SIEMENS

SINUMERIK

SINUMERIK 840D sl / 828D
Pro pokročilé

Programovací příručka

Platí pro:

Řídící systém
SINUMERIK 840D sl / 840DE sl
SINUMERIK 828D

Software	Verze
Programového vybavení CNC	2.7

02/2011
6FC5398-2BP40-1UA0

Právní upozornění

Koncept výstražných upozornění

Tato příručka obsahuje pokyny, které musíte dodržovat z důvodu své osobní bezpečnosti a zamezení materiálním škodám. Upozornění ohledně Vaší osobní bezpečnosti jsou zvýrazněny výstražným trojúhelníkem, upozornění týkající se pouze materiálních škod jsou uvedeny bez výstražného trojúhelníku. Podle stupně ohrožení jsou výstražná upozornění zobrazena v sestupném pořadí následujícím způsobem.

⚠NEBEZPEČÍ

znamená, že **nastane** smrt nebo těžké ublížení na zdraví, když se neučiní příslušná bezpečnostní opatření.

⚠VÝSTRAHA

znamená, že **může** nastat smrt nebo těžké ublížení na zdraví, když se neučiní příslušná bezpečnostní opatření.

⚠POZOR

s výstražným trojúhelníkem znamená, že může nastat lehké ublížení na zdraví, když se neučiní příslušná bezpečnostní opatření.

POZOR

bez výstražného trojúhelníku znamená, že mohou nastat materiální škody, když se neučiní příslušná bezpečnostní opatření.

UPOZORNĚNÍ

znamená, že může dojít k neočekávané události nebo stavu, když se příslušné upozornění nerespektuje.

Při výskytu více stupňů ohrožení bude vždy použito výstražné upozornění s nejvyšším stupněm. Je-li ve výstražném upozornění s výstražným trojúhelníkem výstraha před škodami na zdraví, pak může být v tomtéž výstražném upozornění ještě připojena výstraha před materiálnhími škodami.

Kvalifikovaný personál

Výrobek nebo systém, ke kterému náleží tato dokumentace, může obsluhovat pouze **personál s odpovídající kvalifikací**, který bude při provádění stanovených úkolů dodržovat pokyny uvedené v dokumentaci, zejména pak předpisy týkající se bezpečnosti práce. Kvalifikovaný personál je na základě svého vzdělání a zkušeností způsobilý odhalit rizika v souvislosti s obsluhou těchto výrobků či systémů a zabránit možnému ohrožení.

Používání výrobků Siemens v souladu s určením

Mějte na zřeteli následující:

⚠VÝSTRAHA

Výrobky Siemens se smí používat pouze pro účely uvedené v katalogu a v příslušné technické dokumentaci. Pokud se používají cizí výrobky a komponenty, musí být doporučeny nebo schváleny firmou Siemens. Bezporuchový a bezpečný provoz předpokládá odbornou přepravu, skladování, ustavení, montáž, instalaci, uvedení do provozu, obsluhu a údržbu. Musí se dodržovat přípustné podmínky prostředí. Dodržovat se musí také pokyny v příslušné dokumentaci.

Známky

Všechny názvy označené ochrannou známkou ® jsou zapsané známky firmy Siemens AG. Ostatní názvy v této tiskovině mohou být značkami, jejichž používání třetími subjekty pro své účely může porušovat práva majitelů.

Vyloučení odpovědnosti

Zkontrolovali jsme obsah tiskoviny, zda je v souladu s popsaným hardwarem a softwarem. Přesto nelze vyloučit odchylky, takže nemůžeme převzít odpovědnost za kompletní shodu. Údaje v této tiskovině jsou pravidelně kontrolovány, potřebné opravy jsou uvedeny v následujících vydáních.

Siemens AG
Industry Sector
Postfach 48 48
90026 NÜRNBERG
NĚMECKO

Objednací číslo dokumentu: 6FC5398-2BP40-1UA0
℗ 01.2011

Předmluva

Dokumentace systému SINUMERIK

Dokumentace systému SINUMERIK je rozčleněna do následujících kategorií:

- Všeobecná dokumentace
- Uživatelská dokumentace
- Dokumentace výrobce / servisní dokumentace

Doplňkové informace

Na internetové stránce www.siemens.com/motioncontrol/docu naleznete informace k následujícím tématům:

- Objednávání dokumentace / přehled tištěných materiálů
- Další odkazy pro stažení dokumentů
- Používejte on-line dokumentaci (vyhledávání a prohledávání příruček/informací)

Pokud budete mít dotazy týkající se technické dokumentace (např. návrhy, opravy), zašlete prosím e-mail na tuto adresu:

docu.motioncontrol@siemens.com

My Documentation Manager (MDM)

Pomocí následujícího odkazu naleznete informace, pomocí kterých pak můžete na základě obsahu od firmy Siemens individuálně sestavovat OEM dokumentaci specifického stroje.

www.siemens.com/mdm

Vzdělávání

Pokud budete potřebovat informace o nabídce školení, viz:

- www.siemens.com/sitrain

  SITRAIN - školení firmy Siemens pro produkty, systémy a řešení z oblasti automatizační techniky

- www.siemens.com/sinutrain

  SinuTrain - školicí software pro systémy SINUMERIK

Často kladené otázky

Často kladené otázky naleznete na stránkách Service&Support (Služby a podpora) v rámci podpory pro jednotlivé produkty. http://support.automation.siemens.com

SINUMERIK

Pokud budete potřebovat informace o systému SINUMERIK, využijte následující odkaz:

www.siemens.com/sinumerik

Cílová skupina

Předkládaná dokumentace je určena následujícím pracovníkům:

- Programátoři
- Technici mající na starost konfiguraci systémů

Použití

Pomocí této příručky pro programování mohou pracovníci cílové skupiny vyvíjet, psát, testovat a odstraňovat chyby v programech a v obrazovkách uživatelského rozhraní.

Standardní rozsah

V předkládané příručce pro programování jsou popisovány funkce standardního rozsahu dodávky. Doplnění nebo změny, které byly provedeny výrobcem stroje, jsou popsány v dokumentaci od tohoto výrobce stroje.

V rámci řídícího systému se mohou vyskytovat i další funkce nepopsané v rámci této dokumentace, které lze spustit. S ohledem na tyto funkce však není možné vznést žádný nárok pro případ nové dodávky nebo servisního zásahu.

Z důvodů zachování přehlednosti neobsahuje tato dokumentace všechny podrobné informace ke všem typům produktu a také nemůže pokrýt veškeré myslitelné případy, které se mohou v průběhu instalace, provozování a údržby vyskytnout.

Technická podpora

Specifická telefonní čísla na pracovníky technické podpory v dané zemi naleznete na internetu: http://www.siemens.com/automation/service&support

Informace vztahující se ke struktuře a obsahu

Příručka programování, "Základy" a "Pro pokročilé"

Popisy programování NC systémů jsou rozděleny do dvou příruček:

1. Základy

   Příručka programování "Základy" je určena pro zkušené kvalifikované pracovníky obsluhy stroje a předpokládá odpovídající znalosti pro operace vrtání, frézování a soustružení. Pro vysvětlení příkazů a výrazů, které jsou definovány rovněž podle normy DIN 66025, se používají jednoduché příklady programování.

2. Pro pokročilé

   Příručka programování „Pro pokročilé“ slouží technologům, kteří disponují znalostmi o všech možnostech programování. Řídící systémy SINUMERIK umožňují pomocí speciálního programovacího jazyka vytváření programů pro výrobu složitých obrobků (např. modelované povrchy volných tvarů, koordinace kanálů atd.) a technologům výrazně usnadňuje programování složitých operací.

Dostupnost popisovaných prvků jazyka NC-systému

Všechny prvky jazyka NC-systému, které jsou popisovány v předkládané příručce, jsou pro systém SINUMERIK 840D sl k dispozici. Dostupnost prvků týkající se systému SINUMERIK 828D je zapotřebí zjistit v tabulce "Příkazy: Použitelnost u systému SINUMERIK 828D [Strana 876]".

Obsah

Flexibilní programování NC systémů 1

1.1 Proměnné

1.1.1 Všeobecné informace týkající se proměnných

Prostřednictvím použití proměnných, zejména ve spojení s matematickými funkcemi a řídícími strukturami, mohou být externě sestavovány flexibilní výrobní programy a cykly. Pro tento účel systém dává k dispozici tři odlišné druhy proměnných.

- Systémové proměnné

  Systémové proměnné jsou proměnné, které jsou definovány v systému, mají pevně předem určený význam a jsou uživateli k dispozici. Systémové programové vybavení umožňuje jejich čtení a zápis. Příklad: Strojní parametry

  Smyslem systémových proměnných je v maximální možné míře pevně předem definovat vlastnosti systému. Uživatel ale může změnou definice v malé míře tyto vlastnosti ještě upravit. Viz "Opětovná definice systémových proměnných, uživatelských proměnných a příkazů NC jazyka (REDEF) [Strana 31]".

- Uživatelské proměnné

  Uživatelské proměnné jsou proměnné, jejichž význam není systému znám a které ani nejsou systémem vyhodnocovány. Význam je definován výlučně uživatelem.

  Uživatelské proměnné jsou rozděleny následujícím způsobem:

  - Předdefinované uživatelské proměnné

    Předdefinované uživatelské proměnné jsou proměnné, které již jsou v systému definovány, jejich počet ale musí být ještě nastaven uživatelem pomocí specifických strojních parametrů. Vlastnosti těchto proměnných mohou být uživatelem z větší části ještě přizpůsobeny. Viz "Opětovná definice systémových proměnných, uživatelských proměnných a příkazů NC jazyka (REDEF) [Strana 31]".

  - Uživatelem definované proměnné

    Uživatelem definované proměnné jsou proměnné, které jsou definovány výlučně uživatelem a v systému jsou založeny až při zpracovávání programu. Jejich počet, datový typ, možnosti zobrazování a všechny ostatní vlastnosti jsou definovány výlučně uživatelem.

    Viz "Definice uživatelských proměnných (DEF) [Strana 25]".

Viz také

Systémové proměnné [Strana 18]

Předdefinované uživatelské proměnné: Početní parametr (R) [Strana 21]

Předdefinované uživatelské proměnné: linkové proměnné [Strana 23]

Atribut: Inicializační hodnota [Strana 34]

Atribut: Mezní hodnoty (LLI, ULI) [Strana 37]

Atribut: Fyzikální jednotka (PHU) [Strana 39]

Atribut: Přístupová oprávnění (APR, APW, APRP, APWP, APRB, APWB) [Strana 41]

Přehled definovatelných a znovu definovatelných atributů [Strana 46]

Definice a inicializace proměnných typu pole (DEF, SET, REP) [Strana 47]

Datové typy [Strana 55]

1.1.2 Systémové proměnné

Systémové proměnné jsou v systému předem definované proměnné, které ve výrobních programech a cyklech umožňují přístup k aktuálnímu nastavení parametrů řídícího systému, ale také ke stavovým informacím o stroji, řídícím systému a procesech.

Proměnné předběžného zpracování

Jako proměnné předběžného zpracování jsou označovány systémové proměnné, které jsou čteny nebo do nichž se zapisuje v souvislosti s předběžným zpracováním, tzn. v okamžiku interpretace bloku výrobního programu, v němž je systémová proměnná naprogramována. Proměnné předběžného zpracování nezpůsobují jeho zastavení.

Proměnné hlavního zpracování

Jako proměnné hlavního zpracování jsou označovány systémové proměnné, které jsou čteny nebo do nichž se zapisuje v souvislosti se zpracováním hlavní větve, tzn. v okamžiku zpracovávání bloku výrobního programu, v němž je systémová proměnná naprogramována. Proměnné hlavního zpracování následující:

- Systémové proměnné, které mohou být naprogramovány v synchronních akcích (čtení/zápis)
- Systémové proměnné, které mohou být naprogramovány ve výrobním programu a které spouštějí zastavení předběžného zpracování (čtení/zápis)
- Systémové proměnné, které mohou být naprogramovány ve výrobním programu a jejichž hodnota se zjišťuje v rámci předběžného zpracování, zapisuje se ale až v rámci hlavního zpracování (synchronizovaně s hlavním zpracováním: jen zápis)

Systém předpon

Kvůli zvláštnímu označování systémových proměnných se za normálních okolností k jejich názvu připojuje předpona, která se skládá ze znaku $, za nímž následuje jedno nebo dvě písmena a znak podtržení:

$ + 1. písmeno	**Význam: Druh dat**
Systémové proměnné, jejichž čtení/zápis se uskutečňují v rámci předběžného zpracování	
$M	Strojní parametry [1)]
$S	Nastavované parametry, chráněné oblasti [1)]
$T	Parametry správy nástrojů
$P	Programovatelné hodnoty
$C	Proměnné cyklů v cyklech ISO
$O	Data volitelných doplňků
R	R-Parametry (početní parametry) [2)]
Systémové proměnné, jejichž čtení/zápis se uskutečňují v rámci hlavního zpracování	
$$M	Strojní parametry [1)]
$$S	Nastavované parametry [1)]
$A	Aktuální data hlavního zpracování
$V	Data servomechanismů
$R	R-Parametry (početní parametry) [2)]
[1)] Když jsou strojní a nastavované parametry ve výrobním programu / cyklu používány jako proměnné předběžného zpracování, zapisuje se předpona s jedním znakem $. V případě použití v synchronních akcích jako proměnná hlavního zpracování se předpona zapisuje se dvěma znaky $. [2)] Když jsou R-parametry ve výrobním programu / cyklu používány jako proměnné předběžného zpracování, nezapisuje se žádná předpona. V případě použití v synchronní akci jako proměnná hlavního zpracování se předpona zapisuje s jedním znakem $, např. $R10.	

2. písmeno	**Význam: Zobrazování**
N	Globální proměnná NCK (**N**CK)
C	Kanálová proměnná (**C**hannel)
A	Osová proměnná (**A**xis)

Okrajové podmínky

Výjimky ze systému předpon

Následující systémové proměnné se odlišují od výše popisovaného systému předpon:

- $TC_...: 2. písmeno v tomto případě neodkazuje na kanálové proměnné, nýbrž na systémové proměnné vztahující se k držáku nástroje (TC = Tool Carrier).
- $P_ ...: Kanálové systémové proměnné

Použití strojních a nastavovaných parametrů v synchronních akcích

Při použití strojních a nastavovaných parametrů v synchronních akcích je možné pomocí předpony stanovit, zda se má čtení/zápis daného strojního/nastavovaného parametru uskutečňovat synchronně s předběžným nebo hlavním zpracováním.

Pokud zůstává parametr v průběhu zpracovávání nezměněn, je možno jej číst v průběhu předběžného zpracování. Předpona strojního nebo nastavovaného parametru se pro tento účel zapisuje s jedním znakem $.

Programový kód

```
ID=1 WHENEVER G710 $AA_IM[z] < $SA_OSCILL_REVERSE_POS2[Z]-6 DO $AA_OVR[X]=0
```

Pokud se parametr v průběhu zpracovávání mění, musí jeho čtení/zápis probíhat synchronně s hlavním zpracováním. Předpona strojního nebo nastavovaného parametru se pro tento účel zapisuje se dvěma znaky $.

Programový kód

```
ID=1 WHENEVER $AA_IM[z] < $$SA_OSCILL_REVERSE_POS2[Z]-6 DO $AA_OVR[X]=0
```

Poznámka

Zapisování strojních parametrů

Při zapisování do strojních nebo nastavovaných parametrů je nutno dávat pozor na to, aby aktivní úroveň přístupových oprávnění při zpracovávání výrobního programu / cyklu umožňovala přístup za účelem zápisu a aby byl v platnosti parametr "IMMEDIATE".

Literatura

Pokud budete potřebovat výpis vlastností všech systémových proměnných, viz:

Příručka Seznam systémových proměnných

Viz také

Všeobecné informace týkající se proměnných [Strana 17]

1.1.3 Předdefinované uživatelské proměnné: Početní parametr (R)

Funkce

Početní parametry nebo R-parametry jsou předem definované uživatelské proměnné s označením R definované jako pole s datovým typem `REAL`. Z historických důvodů je pro R-parametry vedle způsobu zápisu s indexem pole, např. `R[10]`, přípustný také způsob zápisu bez indexu pole, např. `R10`.

Při použití v synchronních akcích musí být na začátku uveden znak $, např. `$R10`.

Syntaxe

V případě použití jako proměnná předběžného zpracování:
`R<n>`
`R[<výraz>]`

V případě použití jako proměnná hlavního zpracování:
`$R<n>`
`$R[<výraz>]`

Význam

`R`: Identifikátor při použití jako proměnná předběžného zpracování, např. ve výrobním programu

`$R`: Identifikátor při použití jako proměnná hlavního zpracování, např. v synchronních akcích

Typ: REAL

Rozsah hodnot: V případě neexponenciálního způsobu zápisu:
± (0.000 0001 ... 9999 9999)

Upozornění:
Povoleno je maximálně 8 desetinných míst.

V případě exponenciálního způsobu zápisu:
± ($1*10^{-300}$ ... $1*10^{+300}$)

Upozornění:

- Způsob zápisu: <mantisa>EX<exponent> např. 8.2EX-3
- Je povoleno maximálně 10 znaků včetně znaménka a desetinné tečky.

<n>:	Číslo R-Parametru	
	Typ:	INT
	Rozsah hodnot:	0 - MAX_INDEX **Upozornění** MAX_INDEX vyplývá z počtu R-parametrů, který je definován parametrem: MAX_INDEX = (MD28050 $MN_MM_NUM_R_PARAM) - 1
<výraz>:	Index pole Jako index pole může být uveden libovolný výraz, pokud ale výsledek tohoto výrazu může být převeden na datový typ INT (INT, REAL, BOOL, CHAR).	

Příklad

Přiřazení R-parametru a použití R-parametrů v matematických funkcích:

```
Programový kód                  Komentář
R0=3.5678                       ;   Přiřazení v předběžném zpracování
R[1]=-37.3                      ;   Přiřazení v předběžném zpracování
R3=-7                           ;   Přiřazení v předběžném zpracování
$R4=-0.1EX-5                    ;   Přiřazení v hlavním zpracování: R4 = -0.1 * 10^-5
$R[6]=1.874EX8                  ;   Přiřazení v hlavním zpracování: R6 = 1.874 * 10^8
R7=SIN(25.3)                    ;   Přiřazení v předběžném zpracování

R[R2]=R10                       ;   Nepřímé adresování pomocí R-parametru
R[(R1+R2)*R3]=5                 ;   Nepřímé adresování pomocí matematického výrazu

X=(R1+R2)                       ;   Najíždění osou X na pozici, která vyplývá se součtu
                                    R1 a R2.
Z=SQRT(R1*R1+R2*R2)             ;   Najíždění osou Z na pozici odpovídající druhé
                                    odmocnině z výrazu (R1^2 + R2^2).
```

Viz také

Všeobecné informace týkající se proměnných [Strana 17]

1.1.4 Předdefinované uživatelské proměnné: linkové proměnné

Funkce

Prostřednictvím linkových proměnných mohou být v rámci funkce "NCU-Link" cyklicky vyměňována data mezi jednotkami NCU, které jsou spolu spojeny pomocí sítě. Tyto jednotky přitom umožňují přístup do paměti linkových proměnných v závislosti na formátu těchto dat. Velikost a také datová struktura paměti linkových proměnných je definována uživatelem / výrobcem zařízení tak, aby vyhovovala danému zařízení.

Linkové proměnné jsou globální systémové uživatelské proměnné, jejichž čtení a zápis je možné v případě linkové komunikace nastavené v konfiguraci provádět všemi jednotkami NCU linkového svazku pomocí výrobních programů a cyklů. Oproti globálním uživatelským proměnným (GUD) se mohou linkové proměnné používat také v synchronních akcích.

U zařízení bez aktivního modulu NCU-Link mohou být linkové proměnné používány lokálně v řídícím systému vedle globálních uživatelských proměnných (GUD) jako další globální uživatelské proměnné.

Syntaxe

```
$A_DLB[<Index>]
$A_DLW[<Index>]
$A_DLD[<Index>]
$A_DLR[<Index>]
```

Význam

`$A_DLB`:	Linková proměnná pro datový formát BYTE (1 byte)	
	Datový typ:	UINT
	Rozsah hodnot:	0 ... 255
`$A_DLW`:	Linková proměnná pro datový formát WORD (2 byty)	
	Datový typ:	INT
	Rozsah hodnot:	-32768 ... 32767
`$A_DLD`:	Linková proměnná pro datový formát DWORD (4 byty)	
	Datový typ:	INT
	Rozsah hodnot:	-2147483648 ... 2147483647
`$A_DLR`:	Linková proměnná pro datový formát REAL (8 bytů)	
	Datový typ:	REAL
	Rozsah hodnot:	$\pm(2{,}2*10^{-308} \ldots 1{,}8*10^{+308})$

`<Index>`: Index adresy v bytech, vypočítaný od začátku paměti linkových proměnných

Datový typ: INT

Rozsah hodnot: 0 - MAX_INDEX

Upozornění

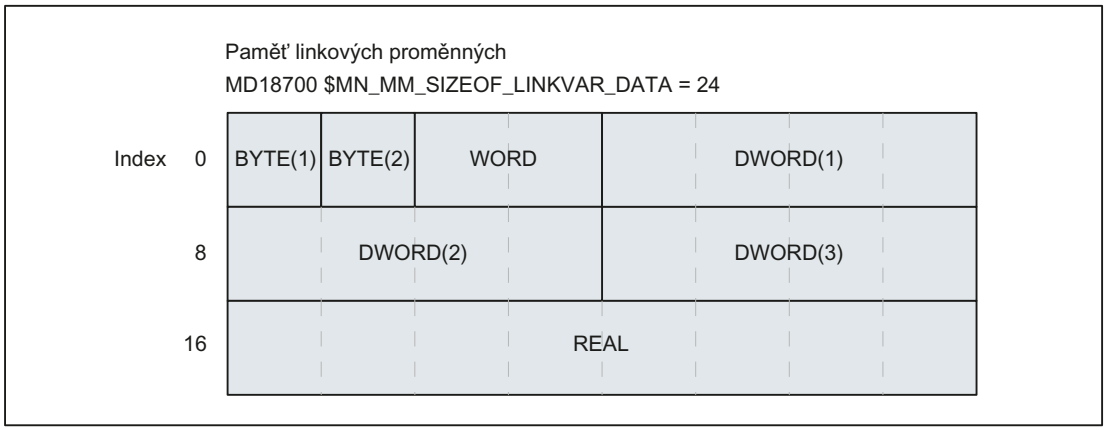
- MAX_INDEX vyplývá z velikosti paměti linkových proměnných, která je definována parametrem: MAX_INDEX = (MD18700 $MN_MM_SIZEOF_LINKVAR_DATA) - 1
- V programech se smí používat jedině indexy, takže byty adresované v paměti linkových proměnných leží na hranici datového formátu ⇒
  Index = n * byty, kde n = 0, 1, 2, ...
  - `$A_DLB[i]`: i = 0, 1, 2, ...
  - `$A_DLW[i]`: i = 0, 2, 4, ...
  - `$A_DLD[i]`: i = 0, 4, 8, ...
  - `$A_DLR[i]`: i = 0, 8, 16, ...

Příklad

V automatickém zařízení jsou 2 jednotky NCU (NCU1 a NCU2). K NCU1 je připojena osa stroje AX2, jejíž pohyb má být ovládán z NCU2 mechanismem linkové osy.

NCU1 cyklicky zapisuje skutečnou hodnotu proudu ($VA_CURR) osy AX2 do paměti linkových proměnných. NCU2 cyklicky čte skutečnou hodnotu proudu přenášenou pomocí linkové komunikace a při překročení mezní hodnoty vypisuje alarm 61000.

Datová struktura v paměti linkových proměnných je uvedena na následujícím obrázku. Skutečná hodnota proudu se přenáší pomocí hodnoty typu REAL.



NCU1

NCU1 cyklicky v taktu IPO zapisuje v rámci statické synchronní akce skutečnou hodnotu proudu osy AX2 prostřednictvím linkové proměnné $A_DLR[16] do paměti linkových proměnných.

Programový kód

```
N111 IDS=1 WHENEVER TRUE DO $A_DLR[16]=$VA_CURR[AX2]
```

NCU2
NCU2 cyklicky v taktu IPO čte v rámci statické synchronní akce skutečnou hodnotu proudu osy AX2 prostřednictvím linkové proměnné $A_DLR[16] z paměti linkových proměnných. Pokud je skutečná hodnota proudu větší než 23,0 A, aktivuje se alarm 61000.

Programový kód

```
N222 IDS=1 WHEN $A_DLR[16] > 23.0 DO SETAL(61000)
```

Viz také

Všeobecné informace týkající se proměnných [Strana 17]

1.1.5 Definice uživatelských proměnných (DEF)

Funkce

Pomocí příkazu `DEF` můžete definovat vlastní proměnné a přiřazovat jim hodnoty. Oproti systémovým proměnným jsou označovány jako uživatelem definované proměnné nebo uživatelské proměnné (User Data).

V závislosti na oblasti jejich platnosti, tzn. oblasti, ve které se proměnná zobrazuje, existují následující kategorie uživatelských proměnných:

- Lokální uživatelské proměnné (LUD)

  Lokální uživatelské proměnné (LUD) jsou proměnné, které jsou definovány ve výrobním programu, který v okamžiku zpracování není hlavním programem. Jsou založeny při vyvolávání výrobního programu a s koncem výrobního programu, příp. při resetu NC systému, jsou vymazány. Přístup k LUD je možný pouze v rámci výrobního programu, v němž jsou definovány.

- Programově globální uživatelské proměnné (PUD)

  Programově globální uživatelské proměnné (PUD) jsou proměnné, které jsou definovány ve výrobním programu, který je používán jako hlavní program. Jsou založeny se spuštěním výrobního programu a s koncem programu, příp. při resetu NC systému, jsou vymazány. Přístup k PUD je k dispozici v hlavním programu a ve všech podprogramech.

- Globální uživatelské proměnné (GUD)

  Globální uživatelské proměnné (GUD) jsou v globálními proměnnými v NC systému, příp. v kanálu, které jsou definovány v datovém modulu (SGUD, MGUD, UGUD, GUD4 ... GUD9) a které zůstávají zachovány i po vypnutí systému. Přístup ke GUD je k dispozici ve všech výrobních programech.

Uživatelské proměnné musí být před svým použitím (čtení / zápis) definovány. Přitom je potřeba dbát na následující pravidla:

- GUD musí být definována v definičním souboru, např. _N_DEF_DIR/_M_SGUD_DEF.
- PUD a LUD musí být definovány v definiční části výrobního programu.
- Definice dat se musí provádět v samostatném bloku.
- V každé definici dat smí být použit jen jeden datový typ.
- V každé definici dat může být definován větší počet proměnných stejného datového typu.

Syntaxe

```
DEF <oblast> <typ> <VL_Stop> <okamžik inicializace> <fyzikální
jednotka> <mezní hodnota> <přístupová oprávnění>
<název>[<hodnota_1>,<hodnota_2>,<hodnota_3>]=<počáteční_hodnota>
```

Význam

DEF:	Příkaz pro definici uživatelských proměnných GUD, PUD, LUD	
<oblast>:	Oblast platnosti, týká se pouze GUD:	
	NCK:	Uživatelská proměnná globální v NC systému
	CHAN:	Uživatelská proměnná globální v kanálu
<typ>:	Datový typ:	
	INT:	Celočíselné hodnoty se znaménkem
	REAL :	Reálné číslo (LONG REAL podle IEEE)
	BOOL:	Boolovská hodnota TRUE (1) / FALSE (0)
	CHAR:	Znaky ASCII
	STRING[<Max. délka>]:	Řetězec znaků definované délky
	AXIS:	Identifikátor osy/vřetena
	FRAME:	Geometrické údaje pro statickou transformaci souřadného systému
	viz "Datové typy [Strana 55]".	
<VL_Stop>:	Zastavení předběžného zpracování, týká se pouze GUD (není nutné)	
	SYNR:	Zastavení předběžného zpracování při čtení
	SYNW:	Zastavení předběžného zpracování při zápisu
	SYNRW:	Zastavení předběžného zpracování při čtení/zápisu

`<okamžik inicializace>`: Okamžik, ve kterém je proměnná znovu inicializována (není nutné)

`INIPO`:	Power On
`INIRE`:	Konec hlavního programu, reset NC systému nebo Power-On
`INICF`:	Příkaz NewConfig nebo konec hlavního programu, reset NC systému nebo Power-On
`PRLOC`:	Konec hlavního programu, reset NC systému po lokální změně nebo Power-On

viz "Atribut: Inicializační hodnota [Strana 34]".

`<fyzikální jednotka>`: Fyzikální jednotka (není nutné)

`PHU` **<jednotka>**:

viz "Atribut: Fyzikální jednotka (PHU) [Strana 39]".

`<mezní hodnota>`: Horní a dolní mezní hodnota (není nutné)

`LLI` **<mezní hodnota>**:	spodní mezní hodnota (lower limit)
`ULI` **<mezní hodnota>**:	horní mezní hodnota (upper limit)

viz "Atribut: Mezní hodnoty (LLI, ULI) [Strana 37]".

`<přístupová oprávnění>`: Přístupová oprávnění pro čtení / zápis GUD prostřednictvím výrobního programu nebo BTSS (není nutné)

`APRP` **<úroveň ochrany>**:	Čtení: Výrobní program
`APWP` **<úroveň ochrany>**:	Zápis: Výrobní program
`APRB` **<úroveň ochrany>**:	Čtení: BTSS
`APWB` **<úroveň ochrany>**:	Zápis: BTSS
Úroveň ochrany	Rozsah hodnot: 0 ... 7

viz "Atribut: Přístupová oprávnění (APR, APW, APRP, APWP, APRB, APWB) [Strana 41]".

`<název>`: Název proměnné

Upozornění

- Maximálně 31 znaků
- První dva znaky musí být písmena a/nebo znak podtržení.
- Znak "$" je vyhrazen pro systémové proměnné a nesmí se používat.

`[<hodnota_1>, <hodnota_2>, <hodnota_3>]`: Údaje o velikosti pole pro 1- až 3-rozměrné proměnné typu pole (není nutné)

`<počáteční hodnota>`: Inicializační hodnota (není nutné)

viz "Atribut: Inicializační hodnota [Strana 34]".

Pokud jde o inicializaci proměnných typu pole:

viz "Definice a inicializace proměnných typu pole (DEF, SET, REP) [Strana 47]".

Příklady

Příklad 1: Definice uživatelských proměnných v datovém modulu pro výrobce stroje

Programový kód

```
%_N_MGUD_DEF                                          ; Modul GUD: Výrobce stroje
$PATH=/_N_DEF_DIR
DEF CHAN REAL PHU 24 LLI 0 ULI 10 STROM_1, STROM_2
; Popis
; Definice dvou GUD: STROM_1, STROM_2
; Oblast platnosti: v kanálu
; Datový typ: REAL
; Zastavení předběžného zpracování: není naprogramováno => předdefinovaná hodnota = žádné zastavení
  předběžného zpracování
; Fyzikální jednotka: 24 = [A]
; Mezní hodnota: Low = 0.0, High = 10.0
; Přístupová oprávnění: není naprogramováno => předdefinovaná hodnota = 7 = poloha přepínače na klíč 0
; Inicializační hodnota: není naprogramováno => předdefinovaná hodnota = 0.0

DEF NCK REAL PHU 13 LLI 10 APWP 3 APRP 3 APWB 0 APRB 2 ZEIT_1=12, ZEIT_2=45
; Popis
; Definice dvou GUD: ZEIT_1, ZEIT_2
; Oblast platnosti: v NCK
; Datový typ: REAL
; Zastavení předběžného zpracování: není naprogramováno => předdefinovaná hodnota = žádné zastavení
  předběžného zpracování
; Fyzikální jednotka: 13 = [s]
; Mezní hodnota: Low = 10.0, High = není naprogramováno => horní hranice definiční oblasti
; Přístupová oprávnění:
;    Výrobní program: Čtení/zápis = 3 = koncový uživatel
;    BTSS: Zápis = 0 = Siemens, Čtení = 3 = koncový uživatel
; Inicializační hodnota: ZEIT_1 = 12.0, ZEIT_2 = 45.0

DEF NCK APWP 3 APRP 3 APWB 0 APRB 3 STRING[5] GUD5_NAME = "COUNTER"
; Popis
; Definice jednoho GUD: GUD5_NAME
; Oblast platnosti: v NCK
; Datový typ: STRING, max. 5 znaků
; Zastavení předběžného zpracování: není naprogramováno => předdefinovaná hodnota = žádné zastavení
  předběžného zpracování
; Fyzikální jednotka: není naprogramováno => předdefinovaná hodnota = 0 = žádná fyzikální jednotka
; Mezní hodnota: není naprogramováno => hranice definiční oblasti: Low = 0, High = 255
; Přístupová oprávnění:
;    Výrobní program: Čtení/zápis = 3 = koncový uživatel
;    BTSS: Zápis = 0 = Siemens, Čtení = 3 = koncový uživatel
; Inicializační hodnota: "COUNTER"
M30
```

Příklad 2: V programu globální a lokální uživatelské proměnné (PUD / LUD)

Programový kód	**Komentář**
PROC MAIN	; Hlavní program
DEF INT VAR1	; Definice PUD
...	
SUB2	; Volání podprogramu
...	
M30	

Programový kód	**Komentář**
PROC SUB2	; Podprogram SUB2
DEF INT VAR2	; Definice LUD
...	
IF (VAR1==1)	; Čtení PUD
VAR1=VAR1+1	; Čtení a zápis PUD
VAR2=1	; Zápis LUD
ENDIF	
SUB3	; Volání podprogramu
...	
M17	

Programový kód	**Komentář**
PROC SUB3	; Podprogram SUB3
...	
IF (VAR1==1)	; Čtení PUD
VAR1=VAR1+1	; Čtení a zápis PUD
VAR2=1	; Chyba: LUD ze SUB2 není známa
ENDIF	
...	
M17	

Příklad 3: Definice a použití uživatelských proměnných datového typu `AXIS`

Programový kód	**Komentář**
DEF AXIS ABSZISSE	; 1. geometrická osa
DEF AXIS SPINDLE	; Vřeteno
...	
IF ISAXIS(1) == FALSE GOTOF WEITER	
ABSZISSE = $P_AXN1	
WEITER:	
...	
SPINDLE=(S1)	1. vřeteno
OVRA[SPINDLE]=80	; Korekce vřetena = 80%
SPINDLE=(S3)	3. vřeteno

Okrajové podmínky

Globální uživatelské proměnné (GUD)

V rámci definice globálních uživatelských proměnných (GUD) je potřeba mít na paměti následující strojní parametry:

Č.	Identifikátor: $MN_	Význam
11140	GUD_AREA_ SAVE_TAB	Doplňkové zálohování modulů GUD
18118 [1]	MM_NUM_GUD_MODULES	Počet souborů GUD v aktivním systému souborů
18120 [1]	MM_NUM_GUD_NAMES_NCK	Počet názvů globálních GUD
18130 [1]	MM_NUM_GUD_NAMES_CHAN	Počet názvů kanálových GUD
18140 [1]	MM_NUM_GUD_NAMES_AXIS	Počet názvů osových GUD
18150 [1]	MM_GUD_VALUES_MEM	Paměťový prostor pro hodnoty globálních GUD
18660 [1]	MM_NUM_SYNACT_GUD_REAL	Počet GUD typu REAL, které mohou být nastaveny v konfiguraci
18661 [1]	MM_NUM_SYNACT_GUD_INT	Počet GUD typu INT, které mohou být nastaveny v konfiguraci
18662 [1]	MM_NUM_SYNACT_GUD_BOOL	Počet GUD typu BOOL, které mohou být nastaveny v konfiguraci
18663 [1]	MM_NUM_SYNACT_GUD_AXIS	Počet GUD typu AXIS, které mohou být nastaveny v konfiguraci
18664 [1]	MM_NUM_SYNACT_GUD_CHAR	Počet GUD typu CHAR, které mohou být nastaveny v konfiguraci
18665 [1]	MM_NUM_SYNACT_GUD_STRING	Počet GUD typu STRING, které mohou být nastaveny v konfiguraci

[1] V systému SINUMERIK 828D je tento MD jen ke čtení!

Programově globální uživatelské proměnné (PUD)

UPOZORNĚNÍ

Viditelnost programových lokálních uživatelských proměnných (PUD)

Programové lokální uživatelské proměnné (PUD) definované v hlavním programu jsou viditelné i v podprogramech jen tehdy, pokud je nastaven následující strojní parametr:

MD11120 $MN_LUD_EXTENDED_SCOPE = 1

Pokud je MD11120 = 0, jsou programové lokální uživatelské proměnné definované v hlavním programu viditelné pouze v tomto hlavním programu.

Použití uživatelských proměnných datového typu AXIS globálních v NCK přes hranice kanálu

V NCK globální uživatelské proměnné datového typu `AXIS`, které byly při definici datového modulu inicializovány identifikátorem osy, mohou být používány v různých kanálech NC systému jedině tehdy, pokud má osa v těchto kanálech stejné číslo kanálové osy.

Jestliže tomu tak není, musí být proměnná načtena na začátku výrobního programu nebo, jako je tomu v následujícím příkladu, se musí použít funkce AXNAME(...).

Programový kód	Komentář
`DEF NCK STRING[5] ACHSE="X"`	`; Definice datového modulu`
`N100 AX[AXNAME(ACHSE)]=111 G00`	`; Použití ve výrobním programu`

Viz také

Všeobecné informace týkající se proměnných [Strana 17]

1.1.6 Opětovná definice systémových proměnných, uživatelských proměnných a příkazů NC jazyka (REDEF)

Funkce

Pomocí příkazu `REDEF` můžete měnit atributy systémových proměnných, uživatelských proměnných a příkazů NC jazyka. Základním předpokladem pro změnu definice je, že musí být uskutečněna časově až po odpovídající definici.

V rámci opětovné definice není možné měnit větší počet atributů současně. Pro každý atribut, který má být změněn, musí být naprogramován vlastní příkaz `REDEF`.

Jestliže je naprogramováno několik vzájemně si odporujících změn atributu, je v platnosti vždycky ta poslední změna.

Atributy, jejichž definice může být změněna

Viz "Přehled definovatelných a znovu definovatelných atributů [Strana 46]".

Lokální uživatelské proměnné (PUD / LUD)

Pro lokální uživatelské proměnné (PUD / LUD) se žádné změny definice nesmí provádět.

Syntaxe

```
REDEF <název> <VL_Stop>
REDEF <název> <fyzikální jednotka>
REDEF <název> <mezní hodnota>
REDEF <název> <přístupová oprávnění>
REDEF <název> <okamžik inicializace>
REDEF <název> <okamžik inicializace> <počáteční hodnota>
```

Význam

`REDEF`: Příkaz pro změnu definice určitého atributu systémové proměnné, uživatelské proměnné a příkazu NC jazyka

`<název>`: Název již definované proměnné nebo příkazu NC jazyka

`<Zastavení předběžného zpracování>`: Zastavení předběžného zpracování

`SYNR`:	Zastavení předběžného zpracování při čtení
`SYNW`:	Zastavení předběžného zpracování při zápisu
`SYNRW`:	Zastavení předběžného zpracování při čtení/ zápisu

`<fyzikální jednotka>`: Fyzikální jednotka

`PHU` **<jednotka>**:

viz "Atribut: Fyzikální jednotka (PHU) [Strana 39]".

Upozornění
Změny definice jsou nepřípustné pro následující objekty:
- Systémové proměnné
- Globální uživatelská data (GUD)
- Datové typy: `BOOL`, `AXIS`, `STRING`, `FRAME`

`<mezní hodnota>`: Spodní a/nebo horní mezní hodnota

`LLI` **<mezní hodnota>**:	spodní mezní hodnota (lower limit)
`ULI` **<mezní hodnota>**:	horní mezní hodnota (upper limit)

viz "Atribut: Mezní hodnoty (LLI, ULI) [Strana 37]".

Upozornění
Změny definice jsou nepřípustné pro následující objekty:
- Systémové proměnné
- Globální uživatelská data (GUD)
- Datové typy: `BOOL`, `AXIS`, `STRING`, `FRAME`

`<přístupová oprávnění>`: Přístupová oprávnění pro čtení / zápis prostřednictvím výrobního programu nebo BTSS

`APX` **<úroveň ochrany>**:	Spouštění: Prvek NC jazyka	
`APRP` **<úroveň ochrany>**:	Čtení: Výrobní program	
`APWP` **<úroveň ochrany>**:	Zápis: Výrobní program	
`APRB` **<úroveň ochrany>**:	Čtení: BTSS	
`APWB` **<úroveň ochrany>**:	Zápis: BTSS	
	Úroveň ochrany	Rozsah hodnot: 0 ... 7

viz "Atribut: Přístupová oprávnění (APR, APW, APRP, APWP, APRB, APWB) [Strana 41]".

<okamžik inicializace>:	Okamžik, ve kterém je proměnná znovu inicializována	
	INIPO:	Zapnutí systému
	INIRE:	Konec hlavního programu, reset NC systému nebo Power-On
	INICF:	Příkaz NewConfig nebo konec hlavního programu, reset NC systému nebo Power-On
	PRLOC:	Konec hlavního programu, reset NC systému po lokální změně nebo Power-On
	viz "Atribut: Inicializační hodnota [Strana 34]".	
<počáteční hodnota>:	Inicializační hodnota Při opětovné definici inicializační hodnoty musí být vždy zadán také časový okamžik inicializace (viz <okamžik inicializace>). viz "Atribut: Inicializační hodnota [Strana 34]". Pokud jde o inicializaci proměnných typu pole: viz "Definice a inicializace proměnných typu pole (DEF, SET, REP) [Strana 47]". **Upozornění** Změny definice jsou nepřípustné pro následující objekty: • Systémové proměnné, s výjimkou nastavovaných parametrů	

Příklad

Opětovná definice systémové proměnné $TC_DPC1 v datovém modulu pro výrobce stroje

Programový kód

```
%_N_MGUD_DEF                                          ; Modul GUD: Výrobce stroje
$PATH=/_N_DEF_DIR
REDEF $TC_DPC1 APWB 2 APWP 3
REDEF $TC_DPC1 PHU 21
REDEF $TC_DPC1 LLI 0 ULI 200
REDEF $TC_DPC1 INIPO (100, 101, 102, 103)
;  Popis
;  Přístupová oprávnění pro zápis: BTSS = úroveň ochrany 2, výrobní program = úroveň ochrany 3
;  Upozornění
;  Jestliže se používají soubory z aplikace ACCESS, musí být opětovná definice přístupových oprávnění
;  _N_MGUD_DEF přenesena na _N_MACCESS_DEF.
;  Fyzikální jednotka = [ % ]
;  Mezní hodnota: spodní = 0, horní = 200
;  Proměnná typu pole se při zapnutí systému (Power On) inicializuje danými čtyřmi hodnotami.
M30
```

Okrajové podmínky

Granularita

Opětovná definice se vždy vztahuje na celé, svými názvy jednoznačně určené proměnné. Není tedy možné např. u proměnných typu pole, přiřazovat jednotlivým prvkům pole odlišné hodnoty atributů.

Viz také

Všeobecné informace týkající se proměnných [Strana 17]

1.1.7 Atribut: Inicializační hodnota

Definice (DEF) uživatelských proměnných

Při definici je možné pro následující uživatelské proměnné předem zadat inicializační hodnotu:

- globální uživatelské proměnné (GUD)
- programově globální uživatelské proměnné (PUD)
- lokální uživatelské proměnné (LUD)

Opětovná definice (REDEF) systémových a uživatelských proměnných

Při opětovné definici je možné pro následující proměnné předem zadat inicializační hodnotu:

- Systémová data
  - Nastavované parametry
- Uživatelská data
  - R-parametry
  - Proměnné synchronních akcí ($AC_MARKER, $AC_PARAM, $AC_TIMER)
  - GUD synchronních akcí (SYG_xy[ ], kde x = R, I, B, A, C, S a y = S, M, U, 4, ..., 9)
  - Parametry EPS
  - Parametry nástroje OEM
  - Parametry zásobníku OEM
  - globální uživatelské proměnné (GUD)

Okamžik opětovné inicializace

Při opětovné definici může být zadán časový okamžik, ve kterém má být proměnná znovu inicializována, tzn. kdy má být znovu dosazena inicializační hodnota:

- `INIPO` (Power On)

  Proměnná se znovu inicializuje při zapnutí systému (Power On).

- `INIRE` (Reset)

  Proměnná bude znovu inicializována při resetu NC systému, resetu BAG, na konci výrobního programu (`M02` / `M30`) nebo při zapnutí systému (Power On).

- `INICF` (NewConfig)

  Proměnná bude znovu inicializována při požadavku NewConfig prostřednictvím HMI, při zpracování příkazu `NEWCONFIG` ve výrobním programu, při resetu NC systému, resetu BAG, na konci výrobního programu (M02 / M30) nebo při zapnutí systému (Power On).

- `PRLOC` (programově lokální změna)

  Proměnná se při resetu NC systému, resetu BAG nebo na konci výrobního programu (M02 / M30) znovu inicializuje jen tehdy, jestliže byla v rámci aktuálního výrobního programu změněna.

  Atribut `PRLOC` se smí používat jedině v souvislosti s programovatelnými nastavovanými parametry (viz následující tabulka).

Tabulka. 1-1 Programovatelný nastavovaný parametr

Číslo	Identifikátor	G-příkaz [1)]
42000	$SC_THREAD_START_ANGLE	`SF`
42010	$SC_THREAD_RAMP_DISP	`DITS` / `DITE`
42400	$SA_PUNCH_DWELLTIME	`PDELAYON`
42800	$SA_SPIND_ASSIGN_TAB	`SETMS`
43210	$SA_SPIND_MIN_VELO_G25	`G25`
43220	$SA_SPIND_MAX_VELO_G26	`G26`
43230	$SA_SPIND_MAX_VELO_LIMS	`LIMS`
43300	$SA_ASSIGN_FEED_PER_REV_SOURCE	`FPRAON`
43420	$SA_WORKAREA_LIMIT_PLUS	`G26`
43430	$SA_WORKAREA_LIMIT_MINUS	`G25`
43510	$SA_FIXED_STOP_TORQUE	`FXST`
43520	$SA_FIXED_STOP_WINDOW	`FXSW`
43700	$SA_OSCILL_REVERSE_POS1	`OSP1`
43710	$SA_OSCILL_REVERSE_POS2	`OSP2`
43720	$SA_OSCILL_DWELL_TIME1	`OST1`
43730	$SA_OSCILL_DWELL_TIME2	`OST2`
43740	$SA_OSCILL_VELO	`FA`
43750	$SA_OSCILL_NUM_SPARK_CYCLES	`OSNSC`
43760	$SA_OSCILL_END_POS	`OSE`
43770	$SA_OSCILL_CTRL_MASK	`OSCTRL`

Tabulka. 1-1 Programovatelný nastavovaný parametr

Číslo	Identifikátor	G-příkaz [1)]
43780	$SA_OSCILL_IS_ACTIVE	OS
43790	$SA_OSCILL_START_POS	OSB
1) pomocí tohoto G-příkazu je nastavovaný parametr vyžádán		

Okrajové podmínky

Inicializační hodnota: globální uživatelské proměnné (GUD)

- Pro globální uživatelské proměnné (GUD), jejichž oblastí platnosti je NCK, je možné jako okamžik inicializace zadat jedině INIPO (Power On).
- Pro globální uživatelské proměnné (GUD), jejichž oblastí platnosti je CHAN, je možné jako okamžik inicializace zadat vedle INIPO (Power On) také INIRE (Reset) nebo INICF (NewConfig).
- U globálních uživatelských proměnných (GUD), které mají oblast platnosti CHAN a okamžik inicializace INIRE (Reset) nebo INICF (NewConfig), jsou v případě resetu NC systému, resetu BAG a příkazu NewConfig proměnné nově inicializovány pouze v těch kanálech, v nichž byla výše uvedená událost uskutečněna.

Inicializační hodnota: Datový typ FRAME

Pro proměnné datového typu FRAME nesmí být žádná inicializační hodnota zadávána. Proměnné datového typu FRAME jsou vždy implicitně inicializovány předdefinovaným framem.

Inicializační hodnota: Datový typ CHAR

Pro proměnné datového typu CHAR může být namísto ASCII kódu (0...255) naprogramován také odpovídající znak ASCII v uvozovkách, např. "A".

Inicializační hodnota: Datový typ STRING

U proměnných datového typu STRING musí být dosazen řetězec znaků v uvozovkách, např.: ...= "MACHINE_1".

Inicializační hodnota: Datový typ AXIS

Pro proměnné datového typu AXIS musí být v případě rozšířeného způsobu zápisu adresy dosazen identifikátor osy v závorkách, např.: ...= (X3).

Inicializační hodnota: Systémové proměnné

Pro systémové proměnné nemohou být prostřednictvím opětovné definice zadávány žádné specifické uživatelské inicializační hodnoty. Inicializační hodnoty systémových proměnných jsou pevně definovány v systému. Opětovnou definicí může být ale změněn okamžik (INIRE, INICF), kdy má být systémová proměnná znovu inicializována.

Implicitní inicializační hodnota: Datový typ AXIS

Pro proměnné datového typu `AXIS` se používají následující implicitní inicializační hodnoty:

- Systémová data: "první geometrická osa"
- GUD synchronních akcí (označení: SYG_A*), PUD, LUD:
  Identifikátor osy ze strojního parametru: MD20082
  $MC_AXCONF_CHANAX_DEFAULT_NAME

Implicitní inicializační hodnota: Data nástrojů a zásobníků

Pro data nástrojů a zásobníků mohou být inicializační hodnoty předem definovány prostřednictvím následujícího strojního parametru: MD17520
$MN_TOOL_DEFAULT_DATA_MASK

UPOZORNĚNÍ
Synchronizace
Za synchronizaci událostí, které spouštějí opětovnou inicializaci globální proměnné, se čtením této proměnné na jiném místě nese veškerou odpovědnost uživatel / výrobce stroje.

Viz také

Všeobecné informace týkající se proměnných [Strana 17]

1.1.8 Atribut: Mezní hodnoty (LLI, ULI)

Horní a dolní mezní hodnotu definiční oblasti je možné zadat jen pro následující datové typy.

- INT
- REAL
- CHAR

Definice (DEF) uživatelských proměnných: Mezní hodnoty a implicitní inicializační hodnoty

Jestliže není při definici uživatelských proměnných jednoho z výše uvedených datových typů definována žádná explicitní inicializační hodnota, bude proměnné dosazena implicitní inicializační hodnota datového typu:

- INT: 0
- REAL: 0.0
- CHAR: 0

Jestliže implicitní inicializační hodnota leží mimo definiční oblast stanovenou naprogramovanými mezními hodnotami, bude proměnná inicializována tou mezní hodnotou, která leží blíže k implicitní inicializační hodnotě:

- implicitní inicializační hodnota < spodní mezní hodnota (LLI) ⇒ inicializační hodnota = spodní mezní hodnota
- implicitní inicializační hodnota > horní mezní hodnota (ULI) ⇒ inicializační hodnota = horní mezní hodnota

Příklady:

Programový kód	**Komentář**
DEF REAL GUD1	; spodní mezní hodnota = hranice definiční oblasti ; horní mezní hodnota = hranice definiční oblasti ; žádná inicializační hodnota není naprogramována ; => implicitní inicializační hodnota = 0.0
DEF REAL LLI 5.0 GUD2	; spodní mezní hodnota = 5.0 ; horní mezní hodnota = hranice definiční oblasti ; => inicializační hodnota = 5.0
DEF REAL ULI –5 GUD3	; spodní mezní hodnota = hranice definiční oblasti ; horní mezní hodnota = -5,0 ; => inicializační hodnota = -5.0

Opětovná definice (REDEF) uživatelských proměnných: mezní hodnoty a aktuální skutečné hodnoty

Jestliže jsou při opětovné definici mezních hodnot uživatelské proměnné tyto mezní hodnoty změněny tak, že aktuální skutečná hodnota leží mimo novou definiční oblast, aktivuje se alarm a mezní hodnoty se nepřevezmou.

Poznámka

Opětovná definice (REDEF) uživatelských proměnných

Při opětovné definici mezních hodnot uživatelské proměnné je potřeba dávat pozor, aby změny následujících hodnot byly konzistentní:

- Mezní hodnoty
- Skutečná hodnota
- Inicializační hodnota při opětovné definici a při automatické opětovné inicializaci na základě INIPO, INIRE nebo INICF

Viz také

Všeobecné informace týkající se proměnných [Strana 17]

1.1.9 Atribut: Fyzikální jednotka (PHU)

Fyzikální jednotka může být zadána pouze pro proměnné následujících datových typů:

- INT
- REAL

Programovatelná fyzikální jednotka (PHU)

Fyzikální jednotka se zadává jako číslo s pevnou řádovou čárkou: `PHU <jednotka>`

Mohou být naprogramovány následující fyzikální jednotky:

<Jednotka>	Význam	Fyzikální jednotka
0	Žádná fyzikální jednotka	-
1	Lineární nebo úhlová poloha 1)2)	[mm], [palce], [stupně]
2	Lineární poloha 2)	[mm], [palce]
3	Úhlová poloha	[stupně]
4	Lineární nebo úhlová rychlost 1)2)	[mm/min], [palce/min], [ot/min]
5	Lineární rychlost 2)	[mm/min]
6	Úhlová rychlost	[ot/min]
7	Lineární nebo úhlové zrychlení 1)2)	[m/s^2], [$palce/s^2$], [ot/s^2]
8	Lineární zrychlení 2)	[m/s^2], [$palce/s^2$]
9	Úhlové zrychlení	[ot/s^2]
10	Lineární nebo úhlový ryv 1)2)	[m/s^3], [$palce/s^3$], [ot/s^3]
11	Lineární ryv 2)	[m/s^3], [$palce/s^3$]
12	Úhlový ryv	[ot/s^3]
13	Čas	[s]
14	Zesílení polohového regulátoru	[16.667/s]
15	Otáčkový posuv 2)	[mm/ot], [palce/ot]
16	Teplotní kompenzace 1)2)	[mm], [palce]
18	Síla	[N]
19	Hmotnost	[kg]
20	Moment setrvačnosti 3)	[kgm^2]
21	Procenta	[%]
22	Frekvence	[Hz]
23	Napětí	[V]
24	Proud	[A]
25	Teplota	[°C]
26	Úhel	[stupně]
27	KV	[1000/min]
28	Lineární nebo úhlová poloha 3)	[mm], [palce], [stupně]
29	Řezná rychlost 2)	[m/min], [stopy/min]

<Jednotka>	Význam	Fyzikální jednotka
30	Obvodová rychlost 2)	[m/s], [stopy/s]
31	Odpor	[ohm]
32	Indukčnost	[mH]
33	Točivý moment 3)	[Nm]
34	Konstanta točivého momentu 3)	[Nm/A]
35	Zesílení proudového regulátoru	[V/A]
36	Zesílení regulátoru otáček 3)	[Nm/(rad*s)]
37	Otáčky	[ot/min]
42	Výkon	[kW]
43	Proud, malý	[µA]
46	Točivý moment, malý 3)	[µNm]
48	Promile	-
49	-	[Hz/s]
65	Průtok	[l/min]
66	Tlak	[bar]
67	Objem 3)	[cm^3]
68	Dráhové zesílení 3)	[mm/(V*min)]
69	Dráhové zesílení regulátoru síly	[N/V]
155	Stoupání závitu 3)	[mm/ot], [palce/ot]
156	Změna stoupání závitu 3)	[mm/ot / ot], [palce/ot / ot]

1) Fyzikální jednotka závisí na typu osy: lineární nebo kruhová osa

2) Přepínání systému jednotek

G70/G71 (palce/metrické jednotky)
Po přepnutí základního systému (MD10240 $MN_SCALING_SYSTEM_IS_METRIC) pomocí příkazu G70/G71 se v případě přístupu k systémovým a uživatelským proměnným souvisejícím s délkou za účelem čtení/zápisu neuskutečňuje **žádné** přepočítávání hodnot (skutečná hodnota, předdefinovaná hodnota a mezní hodnoty).

G700/G710 (palce/metrické jednotky)
Po přepnutí základního systému (MD10240 $MN_SCALING_SYSTEM_IS_METRIC) pomocí příkazu G700/G710 se v případě přístupu k systémovým a uživatelským proměnným souvisejícím s délkou za účelem čtení/zápisu **uskutečňuje** přepočítávání hodnot (skutečná hodnota, předdefinovaná hodnota a mezní hodnoty).

3) Proměnná **není** automaticky přepočítávána do momentálně nastaveného systému jednotek NC systému (palce/metrické jednotky). Za přepočítání nese plnou odpovědnost uživatel / výrobce stroje.

Poznámka

Přeběh roviny v důsledku přepočítání formátu

Interně je pro ukládání všech uživatelských proměnných (GUD / PUD / LUD) s fyzikálními jednotkami vztahujícími se k délce používán metrický formát. Nadměrné používání proměnných tohoto druhu v hlavní větvi NCK, např. v synchronních akcích, může mít při přepnutí systému měřicích jednotek za následek přetečení výpočetního času roviny interpolátoru, alarm 4240.

UPOZORNĚNÍ
Kompatibilita jednotek
Při používání proměnných (přiřazení, porovnávání, výpočty atd.) se neprovádí žádná kontrola, zda jsou jednotky podílející se na operaci kompatibilní. Za přepočítání, které je eventuálně nezbytné, nese plnou odpovědnost uživatel / výrobce stroje.

Viz také

Všeobecné informace týkající se proměnných [Strana 17]

1.1.10 Atribut: Přístupová oprávnění (APR, APW, APRP, APWP, APRB, APWB)

Přístupová oprávnění odpovídají následujícím úrovním ochrany, které se zadávají při programování:

Přístupová oprávnění	Úroveň ochrany
Heslo systému	0
Heslo výrobce stroje	1
Heslo servisní služby	2
Heslo koncového uživatele	3
3. poloha přepínače na klíč	4
2. poloha přepínače na klíč	5
1. poloha přepínače na klíč	6
0. poloha přepínače na klíč	7

Definice (DEF) uživatelských proměnných

Přístupová oprávnění (APR... / APW...) mohou být definována pro následující proměnné:

- Globální uživatelská data (GUD)

Opětovná definice (REDEF) systémových a uživatelských proměnných

Přístupová oprávnění (APR... / APW...) mohou být opětovně definována pro následující proměnné:

- Systémová data
  - Strojní parametry
  - Nastavované parametry
  - FRAME
  - Data procesů
  - Kompenzace chyby stoupání vřetena (EEC)
  - Kompenzace průvěsu (CEC)
  - Kompenzace chyby kvadrantu (QEC)
  - Data zásobníku
  - Parametry nástroje
  - Chráněné oblasti
  - Orientovatelný držák nástroje
  - Kinematické řetězce
  - 3D chráněné oblasti
  - Ohraničení pracovního pole
  - Parametry nástroje ISO
- Uživatelská data
  - R-parametry
  - Proměnné synchronních akcí ($AC_MARKER, $AC_PARAM, $AC_TIMER)
  - GUD synchronních akcí (SYG_xy[ ], kde x = R, I, B, A, C, S a y = S, M, U, 4, ..., 9)
  - Parametry EPS
  - Parametry nástroje OEM
  - Parametry zásobníku OEM
  - globální uživatelské proměnné (GUD)

Poznámka

Při opětovné definici mohou být přístupová oprávnění k určité proměnné volně nastavitelná v rozmezí mezi nejnižším stupněm ochrany 7 až po vlastní stupeň ochrany, např. 1 (výrobce stroje).

Opětovná definice (REDEF) příkazů NC jazyka

Oprávnění k přístupu, příp. ke spouštění (APX) mohou být opětovně definována pro následující příkazy NC jazyka:

- G-funkce / podmínky dráhy

  Literatura:

  /PG/ Příručka programování, Základy; kapitola: G-funkce / podmínky dráhy

- Předem definované funkce

  Literatura:

  /PG/ Příručka programování, Základy; kapitola: Předem definované funkce

- Vyvolávání předem definovaných podprogramů

  Literatura:

  /PG/ Příručka programování, Základy; kapitola: Vyvolávání předem definovaných podprogramů

- Příkaz DO u synchronních akcí
- Programový identifikátor cyklů

  Cyklus musí být uložen v adresáři cyklů a musí obsahovat příkaz PROC.

Přístupová oprávnění týkající se výrobních programů a cyklů (APRP, APWP)

Jednotlivá přístupová oprávnění mají pro přístup ve výrobním programu, příp. v cyklu, následující účinky:

- APRP 0 / APWP 0
  - Při zpracovávání výrobního programu musí být zadáno heslo systému.
  - Cyklus musí být uložen v adresáři _N_CST_DIR (systém).
  - Pro adresář _N_CST_DIR musí být ve strojním parametru MD11160 $MN_ACCESS_EXEC_CST nastaveno oprávnění pro zpracovávání na "systém".
- APRP 1 / APWP 1 příp. APRP 2 / APWP 2
  - Při zpracovávání výrobního programu musí být zadáno heslo výrobce stroje, příp. servisní služby.
  - Cyklus musí být uložen v adresáři _N_CMA_DIR (výrobce stroje) nebo _N_CST_DIR.
  - Pro adresáře _N_CMA_DIR, příp. _N_CST_DIR musí být ve strojních parametrech MD11161 $MN_ACCESS_EXEC_CMA, příp. MD11160 $MN_ACCESS_EXEC_CST nastaveno oprávnění pro zpracovávání minimálně na "výrobce stroje".

- `APRP 3` / `APWP 3`
  - Při zpracovávání výrobního programu musí být zadáno heslo koncového uživatele.
  - Cyklus musí být uložen v adresáři _N_CUS_DIR (uživatel), _N_CMA_DIR nebo _N_CST_DIR.
  - Pro adresáře _N_CUS_DIR, _N_CMA_DIR příp. _N_CST_DIR musí být ve strojních parametrech MD11162 $MN_ACCESS_EXEC_CUS, MD11161 $MN_ACCESS_EXEC_CMA, příp. MD11160 $MN_ACCESS_EXEC_CST nastaveno oprávnění pro zpracovávání minimálně na "koncového uživatele".
- `APRP 4...7` / `APWP 4...7`
  - Při zpracovávání výrobního programu musí být přepínač na klíč nastaven v poloze 3 ... 0.
  - Cyklus musí být uložen v adresáři _N_CUS_DIR, _N_CMA_DIR nebo _N_CST_DIR.
  - Pro adresáře _N_CUS_DIR, _N_CMA_DIR příp. _N_CST_DIR musí být ve strojních parametrech MD11162 $MN_ACCESS_EXEC_CUS, MD11161 $MN_ACCESS_EXEC_CMA, příp. MD11160 $MN_ACCESS_EXEC_CST nastaveno oprávnění pro zpracovávání minimálně na odpovídající polohu přepínače na klíč.

Přístupová oprávnění týkající se BTSS (`APRB`, `APWB`)

Přístupová oprávnění (`APRB`, `APWB`) omezují přístup k systémovým a uživatelským proměnným prostřednictvím BTSS stejnou měrou pro všechny systémové komponenty (HMI, PLC, externí počítač, služby EPS atd.).

Poznámka

Lokální přístupová oprávnění HMI

Při změnách přístupových oprávnění k systémovým datům je nutno dávat pozor, aby byly tyto změny v souladu s přístupovými oprávněními definovanými pomocí mechanismů HMI.

Přístupové atributy `APR` / `APW`

Z důvodů kompatibility jsou atributy `APR` a `APW` implicitně přenášeny do atributů `APRP` / `APRB` a `APWP` / `APWB`.

- `APR x ⇒ APRP x APRB x`
- `APW y ⇒ APWP y APWB y`

Nastavení přístupových oprávnění prostřednictvím souborů typu ACCESS

V případě, že jsou pro předávání přístupových oprávnění používány soubory typu ACCESS, smí být opětovné definice přístupových oprávnění pro systémová data, uživatelská data a příkazy NC jazyka programovány pouze v těchto souborech typu ACCESS. Výjimku tvoří globální uživatelská data (GUD). V jejich případě, jestliže se to jeví jako potřebné, musí být opětovná definice přístupových oprávnění i nadále programována v odpovídajících definičních souborech.

Aby byla přístupová ochrana konzistentní, musí být odpovídajícím způsobem přizpůsobeny strojní parametry pro oprávnění ke spouštění funkcí a pro přístupovou ochranu pro příslušné adresáře.

V principu existují následující postupy:

- Vytvořte potřebné definiční soubory:
  - _N_DEF_DIR/_N_SACCESS_DEF
  - _N_DEF_DIR/_N_MACCESS_DEF
  - _N_DEF_DIR/_N_UACCESS_DEF
- Nastavte parametry oprávnění k zápisu pro tyto definiční soubory na hodnotu, která je zapotřebí pro opětovnou definici:
  - MD11170 $MN_ACCESS_WRITE_SACCESS
  - MD11171 $MN_ACCESS_WRITE_MACCESS
  - MD11172 $MN_ACCESS_WRITE_UACCESS
- Za účelem přístupu k chráněným prvkům z cyklů je nutné přizpůsobit oprávnění pro spouštění funkcí a pro zápis pro adresáře cyklů _N_CST_DIR, _N_CMA_DIR a _N_CST_DIR.

  Oprávnění ke spouštění funkcí
  - MD11160 $MN_ACCESS_EXEC_CST
  - MD11161 $MN_ACCESS_EXEC_CMA
  - MD11162 $MN_ACCESS_EXEC_CUS

  Oprávnění k zápisu
  - MD11165 $MN_ACCESS_WRITE_CST
  - MD11166 $MN_ACCESS_WRITE_CMA
  - MD11167 MN_ACCESS_WRITE_CUS

  Oprávnění pro spouštění funkcí musí být nastaveno minimálně na stejnou úroveň ochrany, jaká je nejvyšší úroveň ochrany používaného prvku.

  Oprávnění pro zápis musí být nastaveno minimálně na stejnou úroveň ochrany jako oprávnění pro spouštění funkcí.
- Oprávnění k zápisu pro lokální adresáře cyklů HMI musí být nastaveno minimálně na stejnou úroveň ochrany, jaká je pro lokální adresáře cyklů v NC systému.

  Literatura

  /BAD/ Návod k obsluze HMI-Advanced,

  kapitola: Systémová oblast Služby > Správa dat > Změnit vlastnosti

Vyvolávání podprogramů v souborech typu ACCESS

Kvůli dalšímu strukturování ochrany proti přístupu mohou být v souborech typu ACCESS také vyvolávány podprogramy (identifikace SPF nebo MPF). Tyto podprogramy přitom dědí oprávnění ke spouštění funkcí ze souboru typu ACCESS, z něhož byly vyvolány.

Poznámka

V souborech typu ACCESS mohou být znovu definována pouze přístupová oprávnění. Všechny ostatní atributy musí být i nadále naprogramovány, příp. znovu definovány v odpovídajících definičních souborech.

Viz také

Všeobecné informace týkající se proměnných [Strana 17]

1.1.11 Přehled definovatelných a znovu definovatelných atributů

Následující tabulky ukazují, u kterých druhů dat mohou být definovány (`DEF`) a/nebo znovu definovány (`REDEF`) jednotlivé druhy atributů.

Systémová data

Druh dat	Počáteční hodnota	Mezní hodnoty	Fyzikální jednotka	Přístupová oprávnění
Strojní parametry	---	---	---	REDEF
Nastavované parametry	REDEF	---	---	REDEF
Data typu FRAME	---	---	---	REDEF
Data procesů	---	---	---	REDEF
Kompenzace chyby stoupání vřetena (EEC)	---	---	---	REDEF
Kompenzace průvěsu (CEC)	---	---	---	REDEF
Kompenzace chyby kvadrantu (QEC)	---	---	---	REDEF
Data zásobníku	---	---	---	REDEF
Parametry nástroje	---	---	---	REDEF
Chráněné oblasti	---	---	---	REDEF
Orientovatelný držák nástroje	---	---	---	REDEF
Kinematické řetězce	---	---	---	REDEF
3D chráněné oblasti	---	---	---	REDEF
Ohraničení pracovního pole	---	---	---	REDEF
Parametry nástroje ISO	---	---	---	REDEF

Uživatelská data

Druh dat	Počáteční hodnota	Mezní hodnoty	Fyzikální jednotka	Přístupová oprávnění
R-parametry	REDEF	REDEF	REDEF	REDEF
Proměnné synchronních akcí ($AC_...)	REDEF	REDEF	REDEF	REDEF
GUD synchronních akcí (SYG_...)	REDEF	REDEF	REDEF	REDEF
Parametry EPS	REDEF	REDEF	REDEF	REDEF
Parametry nástroje OEM	REDEF	REDEF	REDEF	REDEF
Parametry zásobníku OEM	REDEF	REDEF	REDEF	REDEF
globální uživatelské proměnné (GUD)	DEF / REDEF	DEF	DEF	DEF / REDEF
lokální uživatelské proměnné (PUD / LUD)	DEF	DEF	DEF	---

Viz také

Všeobecné informace týkající se proměnných [Strana 17]

1.1.12 Definice a inicializace proměnných typu pole (DEF, SET, REP)

Funkce

Uživatelská proměnná může být definována i jako 1- až maximálně 3-rozměrné pole (Array):

- 1-rozměrné: `DEF <datový typ> <název proměnné>[<n>]`
- 2-rozměrné: `DEF <datový typ> <název proměnné>[<n>,<m>]`
- 3-rozměrné: `DEF <datový typ> <název proměnné>[<n>,<m>,<o>]`

Poznámka

Uživatelské proměnné datového typu STRING mohou být definovány jako maximálně 2-rozměrné pole.

Datové typy

Uživatelské proměnné typu pole mohou být definovány pro následující datové typy: BOOL, CHAR, INT, REAL, STRING, AXIS, FRAME

Přiřazování hodnoty prvku pole

Přiřazování hodnot prvkům pole je možné uskutečňovat v následujících okamžicích:

- při inicializaci pole (inicializační hodnoty)
- v průběhu zpracování programu

Přiřazování hodnot se přitom může provádět následujícími způsoby:

- explicitní zadání prvku pole
- explicitní zadání prvku pole jako prvku počátečního a uvedení seznamu hodnot (`SET`)
- explicitní zadání prvku pole jako prvku počátečního a uvedení hodnoty spolu s četností jejího opakování (`REP`)

Poznámka

Uživatelským proměnným datového typu FRAME není možné přiřadit žádné inicializační hodnoty.

Syntaxe (DEF)

```
DEF <datový typ> <název proměnné>[<n>,<m>,<o>]
DEF  STRING[<délka řetězce>] <název proměnné>[<n>,<m>]
```

Syntaxe (DEF...=SET...)

Použití seznamu hodnot:

- při definici:

  ```
  DEF <datový typ> <název proměnné>[<n>,<m>,<o>] =
  SET(<hodnota1>,<hodnota2>,...)
  ```

  stejný význam má:

  ```
  DEF <datový typ> <název proměnné>[<n>,<m>,<o>] =
  (<hodnota1>,<hodnota2>,...)
  ```

  Poznámka

  Při inicializaci prostřednictvím seznamu hodnot je použití příkazu `SET` nepovinné.

- při přiřazování hodnot:

  ```
  <název proměnné>[<n>,<m>,<o>] = SET(<hodnota1>,<hodnota2>,...)
  ```

Syntaxe (DEF...=REP...)

Použití hodnoty s opakováním

- při definici:

  ```
  DEF <datový typ> <název proměnné>[<n>,<m>,<o>] = REP(<hodnota>)
  ```

  ```
  DEF <datový typ> <název proměnné>[<n>,<m>,<o>] =
  REP(<hodnota>,<počet prvků pole>)
  ```

- při přiřazování hodnot:

  ```
  <název proměnné>[<n>,<m>,<o>] = REP(<hodnota>)
  ```

  ```
  <název proměnné>[<n>,<m>,<o>] = REP(<hodnota>,<počet prvků pole>)
  ```

Význam

`DEF`: Příkaz pro definici proměnné

`<datový typ>`: Datový typ proměnné

Rozsah hodnot:

- u systémové proměnné:

  BOOL, CHAR, INT, REAL, STRING, AXIS

- u proměnných GUD nebo LUD:

  BOOL, CHAR, INT, REAL, STRING, AXIS, FRAME

`<délka řetězce>`: Maximální počet znaků u datového typu STRING

`<název proměnné>`: Název proměnné

`[<n>,<m>,<o>]`: Velikost pole, příp. indexy pole

`<n>`: Velikost pole, příp. index pole pro 1. rozměr

Typ: INT (u systémových proměnných také AXIS)

Rozsah hodnot: Max. velikost pole: 65535
Index pole: $0 \leq n \leq 65534$

`<m>`: Velikost pole, příp. index pole pro 2. rozměr

Typ: INT (u systémových proměnných také AXIS)

Rozsah hodnot: Max. velikost pole: 65535
Index pole: $0 \leq m \leq 65534$

`<o>`: Velikost pole, příp. index pole pro 3. rozměr

Typ: INT (u systémových proměnných také AXIS)

Rozsah hodnot: Max. velikost pole: 65535
Index pole: $0 \leq o \leq 65534$

`SET`: Přiřazení hodnot prostřednictvím seznamu

`(<hodnota1>,<hodnota2>,...)`: Seznam hodnot

`REP`: Přiřazení hodnoty uvedené v poli `<hodnota>`

`<hodnota>`:	Hodnota, která se má při inicializaci zapisovat pomocí příkazu `REP` do prvků pole.
`<počet prvků pole>`:	Počet prvků pole, do kterých se má zapsat hodnota uvedená v poli `<hodnota>`. Pro zbývající prvky pole platí v závislosti na daném okamžiku: • Inicializace při definici pole: → Do zbývajících prvků pole se dosazuje nula. • Přiřazení v průběhu zpracování programu: → Momentální hodnoty prvků pole zůstávají zachovány. Pokud tento parametr není naprogramován, do prvků pole je dosazována hodnota uvedená v poli `<hodnota>`. Jestliže je tento parametr roven nule, platí v závislosti na daném okamžiku: • Inicializace při definici pole: → Všem prvkům je dosazena nulová hodnota • Přiřazení v průběhu zpracování programu: → Momentální hodnoty prvků pole zůstávají zachovány.

Index pole

Implicitní posloupnost prvků pole, např. při přiřazování hodnot pomocí příkazů SET nebo REP, se odvíjí od postupného přičítání indexů pole zprava doleva.

Příklad: Inicializace 3-rozměrného pole s 24 prvky:

```
DEF INT FELD[2,3,4] = REP(1,24)
  FELD[0,0,0] = 1      1. prvek pole
  FELD[0,0,1] = 1      2. prvek pole
  FELD[0,0,2] = 1      3. prvek pole
  FELD[0,0,3] = 1      4. prvek pole
  ...
  FELD[0,1,0] = 1      5. prvek pole
  FELD[0,1,1] = 1      6. prvek pole
  ...
  FELD[0,2,3] = 1      12. prvek pole
  FELD[1,0,0] = 1      13. prvek pole
  FELD[1,0,1] = 1      14. prvek pole
  ...
  FELD[1,2,3] = 1      24. prvek pole
```

čemuž je ekvivalentní:

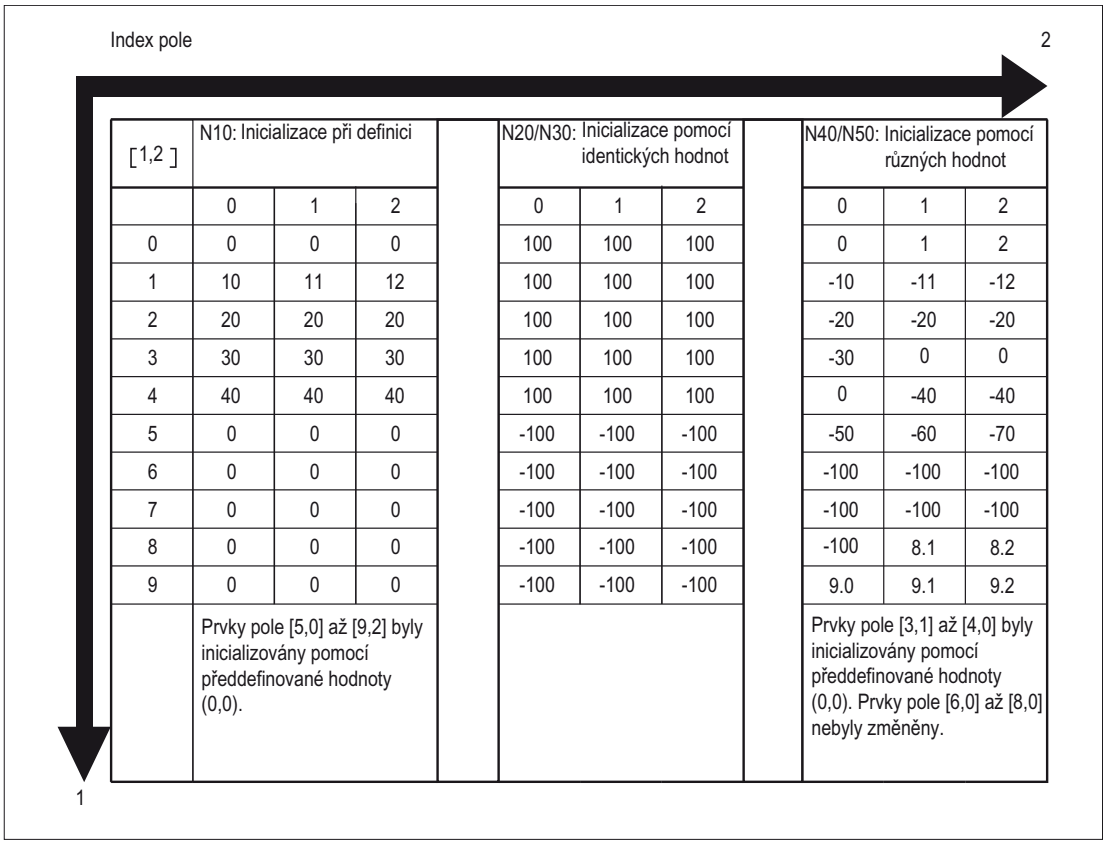
```
FOR n=0 TO 1
  FOR m=0 TO 2
    FOR o=0 TO 3
      FELD[n,m,o] = 1
    ENDFOR
  ENDFOR
ENDFOR
```

Příklad: Inicializace kompletního pole proměnných

Aktuální dosazení viz obrázek.

Programový kód

```
N10 DEF REAL FELD1[10,3]=SET(0,0,0,10,11,12,20,20,20,30,30,30,40,40,40,)
N20 FELD1[0,0]=REP(100)
N30 FELD1[5,0]=REP(-100)
N40 FELD1[0,0]=SET(0,1,2,-10,-11,-12,-20,-20,-20,-30, , , ,-40,-40,-50,-60,-70)
N50 FELD1[8,1]=SET(8.1,8.2,9.0,9.1,9.2)
```



[1,2]	N10: Inicializace při definici			N20/N30: Inicializace pomocí identických hodnot			N40/N50: Inicializace pomocí různých hodnot		
	0	1	2	0	1	2	0	1	2
0	0	0	0	100	100	100	0	1	2
1	10	11	12	100	100	100	-10	-11	-12
2	20	20	20	100	100	100	-20	-20	-20
3	30	30	30	100	100	100	-30	0	0
4	40	40	40	100	100	100	0	-40	-40
5	0	0	0	-100	-100	-100	-50	-60	-70
6	0	0	0	-100	-100	-100	-100	-100	-100
7	0	0	0	-100	-100	-100	-100	-100	-100
8	0	0	0	-100	-100	-100	-100	8.1	8.2
9	0	0	0	-100	-100	-100	9.0	9.1	9.2
	Prvky pole [5,0] až [9,2] byly inicializovány pomocí předdefinované hodnoty (0,0).						Prvky pole [3,1] až [4,0] byly inicializovány pomocí předdefinované hodnoty (0,0). Prvky pole [6,0] až [8,0] nebyly změněny.		

Viz také

Definice a inicializace proměnných typu pole (DEF, SET, REP): Další informace Definice a inicializace proměnných typu pole (DEF, SET, REP): Další informace [Strana 52]

Všeobecné informace týkající se proměnných [Strana 17]

1.1.13 Definice a inicializace proměnných typu pole (DEF, SET, REP): Další informace

Další informace (SET)

Inicializace při definici:

- Počínaje od 1. prvku bude inicializováno hodnotami ze seznamu tolik prvků pole, kolik prvků je naprogramováno v seznamu hodnot.
- Prvkům pole bez explicitně uvedené hodnoty v seznamu (mezery v seznamu hodnot) je dosazována 0.
- U proměnných datového typu AXIS jsou mezery v seznamu hodnot nepřípustné.
- Pokud seznam obsahuje více hodnot, než kolik je definováno prvků pole, aktivuje se alarm.

Přiřazování hodnot během zpracovávání programu

Při přiřazování hodnot v průběhu zpracování programu platí pravidla uvedená výše v rámci definic. Kromě toho existují ještě následující možnosti:

- Jako prvky v seznamu hodnot jsou přípustné také výrazy.
- Přiřazování hodnot začíná u naprogramovaného indexu pole. Takto se dají cíleně obsazovat hodnotami dílčí pole.

Příklad:

Programový kód	**Komentář**
`DEF INT FELD[5,5]`	`; Definice pole`
`FELD[0,0]=SET(1,2,3,4,5)`	`; Přiřazení hodnoty pro prvních 5 prvků pole [0,0] - [0,4]`
`FELD[0,0]=SET(1,2, , ,5)`	`; Přiřazení hodnoty s mezerami pro prvních 5 prvků pole [0,0] - [0,4], prvky pole [0,2] a [0,3] = 0`
`FELD[2,3]=SET(VARIABLE,4*5.6)`	`; Přiřazování hodnot s proměnnou a výrazem od indexu pole [2,3]: [2,3] = VARIABLE [2,4] = 4 * 5.6 = 22.4`

Další informace (REP)

Inicializace při definici:

- Všechny nebo libovolně zadaný počet prvků pole je inicializován uvedenou hodnotou (konstanta).
- Proměnné datového typu FRAME nemohou být inicializovány.

Příklad:

Programový kód	Komentář
`DEF REAL varName[10]=REP(3.5,4)`	`; Definice pole a inicializace prvků pole [0] až [3] s hodnotou 3,5`

Přiřazování hodnot během zpracovávání programu

Při přiřazování hodnot v průběhu zpracování programu platí pravidla uvedená výše v rámci definic. Kromě toho existují ještě následující možnosti:

- Jako prvky v seznamu hodnot jsou přípustné také výrazy.
- Přiřazování hodnot začíná u naprogramovaného indexu pole. Takto se dají cíleně obsazovat hodnotami dílčí pole.

Příklady:

Programový kód	Komentář
`DEF REAL varName[10]`	`; Definice pole`
`varName[5]=REP(4.5,3)`	`; Prvky pole [5] až [7] = 4,5`
`R10=REP(2.4,3)`	`; R-parametr R10 až R12 = 2,4`
`DEF FRAME FRM[10]`	`; Definice pole`
`FRM[5] = REP(CTRANS (X,5))`	`; Prvky pole [5] až [9] = CTRANS(X,5)`

Další informace (všeobecně)

Přiřazování hodnot strojním parametrům os

Strojní parametry os v principu mají index pole datového typu `AXIS`. Při přiřazování hodnot strojnímu parametru osy pomocí příkazu `SET` nebo `REP` je tento index ignorován, příp. není inkrementován.

Příklad: Přiřazování hodnoty strojnímu parametru MD36200 $MA_AX_VELO_LIMIT

```
$MA_AX_VELO_LIMIT[1, AX1] = SET(1.1, 2.2, 3.3)
```

Odpovídá:

```
$MA_AX_VELO_LIMIT[1,AX1] = 1.1
$MA_AX_VELO_LIMIT[2,AX1] = 2.2
$MA_AX_VELO_LIMIT[3,AX1] = 3.3
```

UPOZORNĚNÍ
Přiřazování hodnot strojním parametrům os Při přiřazování hodnot strojním parametrům os pomocí příkazu `SET` nebo `REP` je index pole datového typu `AXIS` ignorován, příp. není inkrementován.

Požadavky na paměť

Datový typ	Požadavky na paměť pro jeden prvek
`BOOL`	1 byte
`CHAR`	1 byte
`INT`	4 byty
`REAL`	8 bytů
`STRING`	(délka řetězce + 1) bytů
`FRAME`	~ 400 bytů, v závislosti na počtu os
`AXIS`	4 byty

Viz také

Definice a inicializace proměnných typu pole (DEF, SET, REP) [Strana 47]

1.1.14 Datové typy

U NC systémů jsou k dispozici následující datové typy:

Datový typ	Význam	Rozsah hodnot
INT	celočíselné hodnoty se znaménkem	-2147483648 ... +2147483647
REAL	Reálné číslo (LONG REAL podle IEEE)	±(~ $2{,}2*10^{-308}$ … ~ $1{,}8*10^{+308}$)
BOOL	Logická hodnota TRUE (1) a FALSE (0)	1, 0
CHAR	Znaky ASCII	Kód ASCII 0 ... 255
STRING	Řetězec znaků definované délky	Maximálně 200 znaků (žádné speciální znaky)
AXIS	Identifikátor osy/vřetena	Identifikátor kanálové osy
FRAME	Geometrické údaje pro statické transformace souřadného systému (posunutí, otočení, změna měřítka, zrcadlové převrácení)	---

Implicitní převody datového typu

Jsou možné následující převody datových typů a při přiřazováních a při přenášení parametrů se implicitně uskutečňují:

z ↓ / do →	REAL	INT	BOOL
REAL	x	o	&
INT	x	x	&
BOOL	x	x	x

x: možné bez omezení

o: Je možná ztráta dat v důsledku překročení rozsahu hodnot ⇒ alarm;
Zaokrouhlení: Hodnota za desetinnou čárkou ≥ 0,5 ⇒ zaokrouhleno nahoru, hodnota za desetinnou čárkou < 0,5 ⇒ zaokrouhleno dolů

&: Hodnota ≠ 0 ⇒ TRUE, hodnota == 0 ⇒ FALSE

Viz také

Všeobecné informace týkající se proměnných [Strana 17]

1.2 Nepřímé programování

1.2.1 Nepřímé programování adres

Funkce

Při nepřímém programování adres je rozšířená adresa (index) nahrazována proměnnou vhodného typu.

Poznámka

Nepřímé programování není možné u těchto adres:

- N (číslo bloku)
- L (podprogram)
- Nastavitelné adresy
  (např. X[1] namísto X1 není přípustné)

Syntaxe

```
<ADRESA>[<Index>]
```

Význam

`<ADRESA>[...]`:	Pevná adresa s rozšířením (index)
`<Index>`:	Proměnná, např. pro číslo vřetena, osy, ...

Příklady

Příklad 1: Nepřímé programování čísla vřetena

Přímé programování:

Programový kód	Komentář
`S1=300`	`; Otáčky 300 ot/min pro vřeteno s číslem 1.`

Nepřímé programování:

Programový kód	Komentář
`DEF INT SPINU=1`	`; Definice proměnné typu INT a přiřazení hodnoty.`
`S[SPINU]=300`	`; Otáčky 300 ot/min pro vřeteno, jehož číslo je uloženo v proměnné SPINU (v tomto příkladu je to vřeteno s číslem 1).`

Příklad 2: Nepřímé programování osy

Přímé programování:

Programový kód	Komentář
FA[U]=300	; Posuv 300 pro osu "U".

Nepřímé programování:

Programový kód	Komentář
DEF AXIS AXVAR2=U	; Definice proměnné typu AXIS a přiřazení hodnoty.
FA[AXVAR2]=300	; Posuv 300 pro osu, jejíž adresový název je uložen v proměnné s názvem AXVAR2.

Příklad 3: Nepřímé programování osy

Přímé programování:

Programování	Komentář
$AA_MM[X]	; Načtení hodnoty změřené měřicí sondou (MCS) pro osu "X".

Nepřímé programování:

Programový kód	Komentář
DEF AXIS AXVAR3=X	; Definice proměnné typu AXIS a přiřazení hodnoty.
$AA_MM[AXVAR3]	; Načtení hodnoty změřené měřicí sondou (MCS) pro osu, jejíž název je uložen v proměnné AXVAR3.

Příklad 4: Nepřímé programování osy

Přímé programování:

Programový kód
X1=100 X2=200

Nepřímé programování:

Programový kód	Komentář
DEF AXIS AXVAR1 AXVAR2	; Definice dvou proměnných typu AXIS.
AXVAR1=(X1) AXVAR2=(X2)	; Přiřazení názvů os.
AX[AXVAR1]=100 AX[AXVAR2]=200	; Pohyb osami, jejichž adresové názvy jsou uloženy v proměnných s názvy AXVAR1 a AXVAR2

Příklad 5: Nepřímé programování osy

Přímé programování:

Programový kód
G2 X100 I20

Nepřímé programování:

Programový kód	Komentář
DEF AXIS AXVAR1=X	; Definice proměnné typu AXIS a přiřazení hodnoty.
G2 X100 IP[AXVAR1]=20	; Nepřímé programování zadání středu pro osu, jejíž adresovací název je uložen v proměnné s názvem AXVAR1.

Příklad 6: Nepřímé programování prvků pole

Přímé programování:

Programový kód	Komentář
DEF INT FELD1[4,5]	; Definice pole 1.

Nepřímé programování:

Programový kód	Komentář
DEFINE DIM1 AS 4	; V případě rozměrů pole musí být velikosti pole uvedeny jako pevné hodnoty.
DEFINE DIM2 AS 5	
DEF INT FELD[DIM1,DIM2]	
FELD[DIM1-1,DIM2-1]=5	

Příklad 7: Nepřímé volání podprogramu

Programový kód	Komentář
CALL "L" << R10	; Volání programu, jehož číslo je uloženo v R10 (skládání řetězců).

1.2.2 Nepřímé programování G-kódů

Funkce

Nepřímé programování G-kódů umožňuje efektivní programování cyklů.

Syntaxe

```
G[<skupina>]=<číslo>
```

Význam

`G[...]`:	G-příkaz s rozšířením (index)
`<skupina>`:	Indexový parametr: Skupina G-funkcí Typ: INT
`<číslo>`:	Proměnná pro číslo G-kódu Typ: INT nebo REAL

Poznámka

Nepřímo mohou být zpravidla programovány pouze G-kódy, které nejsou syntakticky určující.

Ze syntakticky určujících G-kódů jsou možné pouze kódy z 1. skupiny G-funkcí.
G-kódy, které určují syntaxi, ze skupin G-funkcí 2, 3 a 4 možné nejsou.

Poznámka

V nepřímém programování G-kódu nejsou dovoleny žádné aritmetické funkce. Nutný výpočet čísla G-kódu musí proběhnout na samostatném řádku výrobního programu před nepřímým programováním G-kódu.

Příklady

Příklad 1: Nastavitelné posunutí (skupina G-funkcí č. 8)

```
Programový kód                          Komentář
N1010 DEF INT INT_VAR
N1020 INT_VAR=2
...
N1090 G[8]=INT_VAR G1 X0 Y0             ; G54
N1100 INT_VAR=INT_VAR+1                 ; Výpočet G-kódu
N1110 G[8]=INT_VAR G1 X0 Y0             ; G55
```

Příklad 2: Volba roviny (skupina G-funkcí č. 6)

Programový kód	Komentář
N2010 R10=$P_GG[6]	; Načtení aktivní G-funkce ze skupiny G-funkcí č. 6.
...	
N2090 G[6]=R10	

Literatura

Pokud budete potřebovat informace ke skupinám G-funkcí, viz:
Příručka programování, Základy; kapitola "Skupiny G-funkcí".

1.2.3 Nepřímé programování atributů polohy (GP)

Funkce

Atributy polohy, jako např. inkrementální nebo absolutní programování poloh os, mohou být ve spojení s klíčovým slovem `GP` programovány nepřímo jako proměnné.

Aplikace

Nepřímé programování atributů polohy nachází uplatnění v **nahrazovacích cyklech**, protože zde oproti programování atributů polohy jako klíčových slov (např. `IC`, `AC`, ...) existuje následující výhoda:

Díky nepřímému programování, které využívá proměnné, není zapotřebí **žádný** příkaz `CASE`, který by zajišťoval větvení na všechny možné atributy polohy.

Syntaxe

```
<PŘÍKAZ POLOHOVÁNÍ>[<osa/vřeteno>]=
GP(<poloha>,<atribut polohy)
<osa/vřeteno>=GP(<poloha>,<atribut polohy>)
```

Význam

<PŘÍKAZ POLOHOVÁNÍ>[]:	Spolu s klíčovým slovem GP mohou být naprogramovány následující polohovací příkazy: POS, POSA, SPOS, SPOSA Kromě toho je možné zadat: • všechny identifikátory os/vřeten vyskytující se v kanálu: <osa/vřeteno> • proměnný identifikátor osy/vřetena AX
<osa/vřeteno>:	Osa/vřeteno, jehož poloha má být nastavena
GP():	Klíčové slovo pro polohování
<pozice>:	Parametr 1 Pozice osy/vřetena jako konstanta nebo jako proměnná
<atribut polohy>:	Parametr 2 Atribut polohy (např. režim najíždění na pozici) jako proměnná (např. $P_SUB_SPOSMODE) nebo jako klíčové slovo (IC, AC, ...)

Hodnoty předávané proměnnými mají následující význam:

Hodnota	Význam	Přípustné pro:
0	Žádná změna atributu polohy	
1	AC	POS, POSA, SPOS, SPOSA, AX, adresa osy
2	IC	POS, POSA, SPOS, SPOSA, AX, adresa osy
3	DC	POS, POSA, SPOS, SPOSA, AX, adresa osy
4	ACP	POS, POSA, SPOS, SPOSA, AX, adresa osy
5	ACN	POS, POSA, SPOS, SPOSA, AX, adresa osy
6	OC	-
7	PC	-
8	DAC	POS, POSA, AX, adresa osy
9	DIC	POS, POSA, AX, adresa osy
10	RAC	POS, POSA, AX, adresa osy
11	RIC	POS, POSA, AX, adresa osy
12	CAC	POS, POSA
13	CIC	POS, POSA
14	CDC	POS, POSA
15	CACP	POS, POSA
16	CACN	POS, POSA

Příklad

Jestliže je aktivní synchronní vazba mezi řídícím vřetenem S1 a vlečným vřetenem S2, příkazem `SPOS` v hlavním programu se vyvolává následující nahrazovací cyklus, který polohování vřeten zajišťuje.

Polohování se uskutečňuje prostřednictvím příkazu v bloku `N2230`:
`SPOS[1]=GP($P_SUB_SPOSIT,$P_SUB_SPOSMODE)`
`SPOS[2]=GP($P_SUB_SPOSIT,$P_SUB_SPOSMODE)`

Poloha, na kterou se má najet, se načítá ze systémové proměnné $P_SUB_SPOSIT, režim najíždění na pozici se načítá ze systémové proměnné $P_SUB_SPOSMODE.

```
Programový kód                                          Komentář
N1000 PROC LANG_SUB DISPLOF SBLOF
...
N2100 IF($P_SUB_AXFCT==2)
N2110                                                   ; Nahrazení příkazu SPOS / SPOSA / M19 v
                                                          případě aktivní synchronní vazby
N2185 DELAYFSTON                                        ; Začátek oblasti zastavení-prodleva
N2190 COUPOF(S2,S1)                                     ; Deaktivování synchronní vazby mezi vřeteny
N2200                                                   ; Polohování řídícího a vlečného vřetena
N2210 IF($P_SUB_SPOS==TRUE) OR ($P_SUB_SPOSA==TRUE)
N2220                                                   ; Polohování vřetena s příkazem SPOS:
N2230   SPOS[1]=GP($P_SUB_SPOSIT,$P_SUB_SPOSMODE)
        SPOS[2]=GP($P_SUB_SPOSIT,$P_SUB_SPOSMODE)
N2250 ELSE
N2260                                                   ; Polohování vřetena s příkazem M19:
N2270 M1=19 M2=19                                       ; Polohování řídícího a vlečného vřetena
N2280 ENDIF
N2285 DELAYFSTOF                                        ; Konec oblasti zastavení-prodleva
N2290 COUPON(S2,S1)                                     ; Aktivování synchronní vazby mezi vřeteny
N2410 ELSE
N2420                                                   ; Dotaz na další nahrazování
...
N3300 ENDIF
...
N9999 RET
```

Okrajové podmínky

- V synchronních akcích není nepřímé programování atributů polohy možné.

Literatura

Příručka Popis funkcí, Základní funkce; BAG, kanál, zpracování programu, chování při resetu (K1), kapitola: Nahrazování funkcí NC systému pomocí podprogramů

1.2.4 Nepřímé programování řádků výrobního programu (EXECSTRING)

Funkce

Pomocí příkazu výrobního programu `EXECSTRING` je možné spustit zpracování dříve sestavené proměnné typu STRING, jako by se jednalo o řádek výrobního programu.

Syntaxe

Příkaz `EXECSTRING` musí být naprogramován na samostatném řádku výrobního programu.
`EXECSTRING (<proměnná typu string>)`

Význam

`EXECSTRING`:	Příkaz pro zpracování proměnné typu SRTING, jako kdyby šlo o řádek výrobního programu
`<proměnná typu String>`:	Proměnná typu STRING, která obsahuje řádek výrobního programu, který může být samostatně zpracován.

Poznámka

Pomocí příkazu `EXECSTRING` mohou být předávány veškeré konstrukce výrobního programu, které mohou být naprogramovány v programové části výrobního programu. Vyloučeny jsou tedy příkazy `PROC` a `DEF`, jakož i obecně použití souborů INI a DEF.

Příklad

```
Programový kód                              Komentář
N100 DEF STRING[100] BLOCK                  ; Definice proměnné typu String pro
                                              sestavení zpracovatelného řádku
                                              výrobního programu.
N110 DEF STRING[10] MFCT1="M7"
...
N200 EXECSTRING(MFCT1 << "M4711")           ; Zpracování řádku výrobního programu
                                              "M7 M4711".
...
N300 R10=1
N310 BLOCK="M3"
N320 IF(R10)
N330 BLOCK = BLOCK << MFCT1
N340 ENDIF
N350 EXECSTRING(BLOCK)                      ; Zpracování řádku výrobního programu
                                              "M3 M7".
```

1.3 Matematické funkce

Funkce

Matematické funkce jsou přednostně použitelné pro R-parametry a proměnné (nebo konstanty a funkce) typu REAL. Přípustné jsou také typy INT a CHAR.

Operátory / matematické funkce	Význam
+	sečítání
-	odečítání
*	násobení
/	dělení **Pozor**: (typ INT)/(typ INT)=(typ REAL); příklad: 3/4 = 0.75
DIV	celočíselné dělení, pro typ proměnných INT a REAL **Pozor**: (typ INT)DIV(typ INT)=(typ INT); příklad: 3 DIV 4 = 0
MOD	zbytek po dělení (pouze pro typ INT) Příklad: 3 MOD 4 = 3
:	řetězcový operátor (u proměnných typu FRAME)
Sin()	sinus
COS()	kosinus
TAN()	tangens
ASIN()	arkus sinus
ACOS()	arkus kosinus
ATAN2(,)	arkus tangens na druhou
SQRT()	druhá odmocnina
ABS()	absolutní hodnota
POT()	2. druhá mocnina
TRUNC()	celočíselná část Nastavení přesnosti při příkazech porovnávání pomocí funkce TRUNC (viz "Korekce přesnosti při příkazech porovnávání (TRUNC) [Strana 69]")
ROUND()	zaokrouhlování na celá čísla
LN()	přirozený logaritmus
EXP()	exponenciální funkce
MINVAL()	menší hodnota ze dvou proměnných (viz "Minimum, maximum a rozsah proměnných (MINVAL, MAXVAL, BOUND) [Strana 71]")
MAXVAL()	větší hodnota ze dvou proměnných (viz "Minimum, maximum a rozsah proměnných (MINVAL, MAXVAL, BOUND) [Strana 71]")

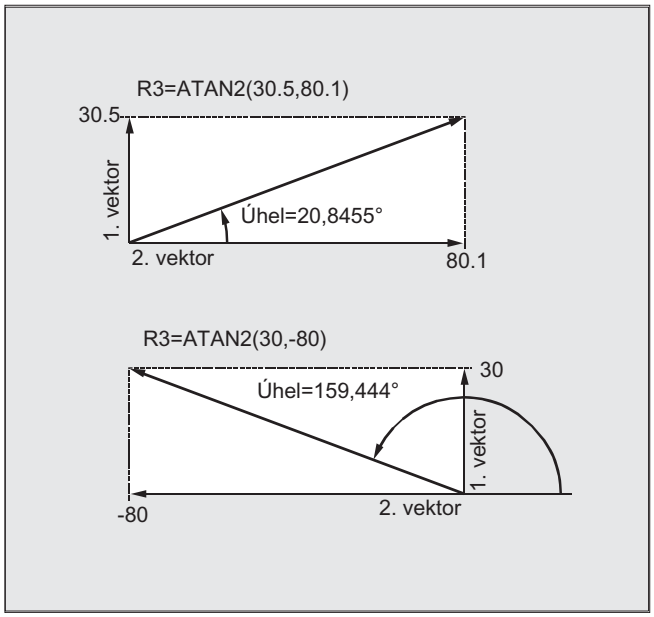
`BOUND()`	Hodnota proměnné, která leží v definovaném intervalu hodnot (viz "Minimum, maximum a rozsah proměnných (MINVAL, MAXVAL, BOUND) [Strana 71]")
`CTRANS()`	Posunutí
`CROT()`	Otočení
`CSCALE()`	změna měřítka
`CMIRROR()`	zrcadlové převrácení

Programování

U matematických operací platí obvyklý matematický způsob zápisu. Priority při zpracování jsou určovány pomocí kulatých závorek. Pro trigonometrické a k nim inverzní funkce platí udání stupňů (pravý úhel = 90°).

Příklady

Příklad 1: ATAN2



Matematická funkce ATAN2 vypočte ze dvou vůči sobě kolmo ležících vektorů úhel součtu těchto vektorů.

Výsledek vždy leží v jednom ze čtyř kvadrantů (-180° < 0 < +180°).

Základem pro úhlový vztah je vždy 2. hodnota v pozitivním směru.

Příklad 2: Inicializace kompletního pole proměnných

Programový kód	**Komentář**
R1=R1+1	; Nová hodnota R1 = stará hodnota R1 + 1
R1=R2+R3 R4=R5-R6 R7=R8*R9	
R10=R11/R12 R13=SIN(25.3)	
R14=R1*R2+R3	; Násobení se provádí před sečítáním.
R14=(R1+R2)*R3	; Napřed se vypočítají závorky.
R15=SQRT(POT(R1)+POT(R2))	; Napřed se vyhodnotí vnitřní závorky: R15 = odmocnina z (R1+R2)
RESFRAME=FRAME1:FRAME2 FRAME3=CTRANS(…):CROT(…)	; Pomocí operátoru pro zřetězení jsou framy sečteny do výsledného framu nebo jsou komponentům framu přiřazovány hodnoty.

1.4 Porovnávací a logické operace

Funkce

Porovnávací operace se mohou používat např. pro formulování podmínky skoku. Přitom je možné porovnávat i složité výrazy.

Relační operace jsou použitelné pro proměnné typů `CHAR, INT`, `REAL` a `BOOL`. U typu `CHAR` je porovnávána kódová hodnota.
U typů `STRING`, `AXIS` a `FRAME` jsou možné operace: == a <>, které se mohou používat také pro operace s typem `STRING` v synchronních akcích.

Výsledek relačních operací je vždy typu `BOOL`.

Logické operátory slouží ke spojení pravdivostních hodnot.

Logické operace mohou být používány jen s proměnnými typu `BOOL`. Prostřednictvím interního převádění typů je možno pracovat i s datovými typy `CHAR, INT` a `REAL`.

U logických (boolovských) operací platí pro datové typy `BOOL`, `CHAR`, `INT` a `REAL`:

- 0 odpovídá hodnotě: FALSE
- cokoli nerovnajícího se 0 odpovídá hodnotě: TRUE

Bitové logické operace

S proměnnými typu `CHAR` a `INT` mohou být prováděny i bitové logické operace. V případě potřeby proběhne konverze typu automaticky.

Programování

Relační operátor	Význam
==	rovná se
<>	nerovná se
>	je větší než
<	je menší než
>=	je větší nebo rovno
<=	je menší nebo rovno

Logický operátor	Význam
`AND`	logické a
`OR`	logické nebo
`NOT`	negace
`XOR`	logické XOR

Bitový logický operátor	Význam
B_AND	bitové logické a
B_OR	bitové logické nebo
B_NOT	bitová negace
B_XOR	bitové logické XOR

Poznámka

V aritmetických výrazech může být pomocí kulatých závorek definováno pořadí zpracování všech operátorů, které může být odlišné od normálních pravidel priority operací.

Poznámka

Mezi boolovskými operandy a operátory musí být zapisována mezera.

Poznámka

Operátor B_NOT se vztahuje pouze na jeden operand. Tento operand se nachází za operátorem.

Příklady

Příklad 1: Relační operátory

```
IF R10>=100 GOTOF ZIEL
```

nebo

```
R11=R10>=100
IF R11 GOTOF ZIEL
```

Výsledek porovnávání R10>=100 se napřed ukládá do pomocné proměnné R11.

Příklad 2: Logické operátory

```
IF (R10<50) AND ($AA_IM[X]>=17.5) GOTOF ZIEL
```

nebo

```
IF NOT R10 GOTOB START
```

NOT se vztahuje pouze na jeden operand.

Příklad 3: Bitové logické operace

```
IF $MC_RESET_MODE_MASK B_AND 'B10000' GOTOF ACT_PLANE
```

1.5 Korekce přesnosti při příkazech porovnávání (TRUNC)

Funkce

Příkaz TRUNC oddělí celočíselnou část operandu po vynásobení faktorem přesnosti.

Nastavitelná přesnost u příkazů porovnávání

Data výrobního programu typu REAL jsou interně zobrazena v IEEE formátu se 64 bity. Na základě této formy zobrazení mohou být vyobrazena nepřesně desetinná čísla, která při porovnávání s ideálně vypočtenými hodnotami mohou vést k neočekávaným výsledkům.

Relativní shodnost

Aby formou zobrazení vyvolané nepřesnosti negativně neovlivňovaly běh programu, kontroluje se u příkazů porovnávání nikoli absolutní shodnost, nýbrž relativní shodnost.

Syntaxe

Korekce přesnosti při chybách porovnávání

```
TRUNC (R1*1000)
```

Význam

`TRUNC`: Odříznutí míst za desetinnou čárkou

Uvažovaná relativní rovnost 10^{-12} při následujících operacích

- Rovnost: (==)
- Nerovnost: (<>)
- Je větší nebo rovno: (>=)
- Je menší nebo rovno: (<=)
- Je větší / menší: (><) s absolutní rovností
- Je větší: (>)
- Je menší: (<)

Kompatibilita

Z důvodu kompatibility může být zkouška relativní shodnosti u operací (>) a (<) nastavením strojního parametru MD10280 $MN_ PROG_FUNCTION_MASK Bit0 = 1 deaktivována.

Poznámka

Porovnávání s daty typu REAL je z výše uvedených důvodů obecně zatíženo určitou nepřesností. V případě neakceptovatelných odchylek musí být použit přepočet na typ INTEGER, při kterém jsou operandy vynásobeny faktorem přesnosti a pak jsou oříznuty pomocí funkce TRUNC.

Synchronní akce

Popisované chování relačních příkazů platí i při synchronních akcích.

Příklady

Příklad 1: Sledování shodnosti

Programový kód	Komentář
N40 R1=61.01 R2=61.02 R3=0.01	; Přiřazení počátečních hodnot
N41 IF ABS(R2-R1) > R3 GOTOF FEHLER	; Skok byl už dříve proveden
N42 M30	; Konec programu
N43 FEHLER: SETAL(66000)	;
R1=61.01 R2=61.02 R3=0.01	; Přiřazení počátečních hodnot
R11=TRUNC(R1*1000) R12=TRUNC(R2*1000) R13=TRUNC(R3*1000)	; Korekce přesnosti
IF ABS(R12-R11) > R13 GOTOF FEHLER	; Skok se už neprovede
M30	; Konec programu
FEHLER: SETAL(66000)	;

Příklad 2: Vytvoření a vyhodnocení koeficientu obou operandů

Programový kód	Komentář
R1=61.01 R2=61.02 R3=0.01	; Přiřazení počátečních hodnot
IF ABS((R2-R1)/R3)-1) > 10EX-5 GOTOF FEHLER	; Skok se už neprovede
M30	; Konec programu
FEHLER: SETAL(66000)	;

1.6 Minimum, maximum a rozsah proměnných (MINVAL, MAXVAL, BOUND)

Funkce

Pomocí příkazů `MINVAL` a `MAXVAL` mohou být mezi sebou porovnávány hodnoty dvou proměnných. Jako výsledek je na výstupu menší hodnota (v případě funkce `MINVAL`), příp. větší hodnota (v případě funkce `MAXVAL`).

Pomocí příkazu `BOUND` je možné zkontrolovat, zda se hodnota zkoušené proměnné nachází v rámci definovaného rozmezí hodnot.

Syntaxe

```
<menší hodnota>=MINVAL(<proměnná1>,<proměnná2>)
<větší hodnota>=MAXVAL(<proměnná1>,<proměnná2>)
<vracená hodnota>=<BOUND>(<minimum>,<maximum>,<zkoušená proměnná>)
```

Význam

`MINVAL`:	Zjišťuje **menší** hodnotu ze dvou proměnných (`<proměnná1>`, `<proměnná2>`)
`<menší hodnota>`:	Proměnná výsledku pro příkaz `MINVAL` Je jí dosazena hodnota menší proměnné.
`MAXVAL`:	Zjišťuje **větší** hodnotu ze dvou proměnných (`<proměnná1>`, `<proměnná2>`)
`<větší hodnota>`:	Proměnná výsledku pro příkaz `MAXVAL` Je jí dosazena hodnota větší proměnné.
`BOUND`:	Zkontroluje, zda proměnná (`<zkoušená proměnná>`) leží v rámci definovaného rozsahu hodnot.
`<minimum>`:	Proměnná, která definuje minimální hodnotu rozsahu hodnot.
`<maximum>`:	Proměnná, která definuje maximální hodnotu rozsahu hodnot.
`<vracená hodnota>`:	Proměnná výsledku pro příkaz `BOUND` Pokud hodnota zkoušené proměnné leží v rámci definovaného rozsahu hodnot, potom se do proměnné výsledku dosadí hodnota zkoušené proměnné. Pokud je hodnota zkoušené proměnné větší než maximální hodnota, potom se do proměnné výsledku dosadí maximální hodnota definovaného rozsahu. Pokud je hodnota zkoušené proměnné menší než minimální hodnota, potom se do proměnné výsledku dosadí minimální hodnota definovaného rozsahu.

Poznámka

Funkce MINVAL, MAXVAL a BOUND mohou být naprogramovány i v synchronních akcích.

Poznámka

Chování v případě rovnosti

V případě rovnosti budou funkce MINVAL/MAXVAL vracet tuto stejnou hodnotu. V případě funkce BOUND bude výsledkem opět hodnota zkoušené proměnné.

Příklad

```
Programový kód                                    Komentář
DEF REAL rVar1=10.5, rVar2=33.7, rVar3, rVar4, rVar5, rValMin, rValMax, rRetVar
rValMin=MINVAL(rVar1,rVar2)                       ; Proměnná rValMin se nastaví na hodnotu 10,5.
rValMax=MAXVAL(rVar1,rVar2)                       ; Proměnná rValMax se nastaví na hodnotu 33,7.
rVar3=19.7
rRetVar=BOUND(rVar1,rVar2,rVar3)                  ; Proměnná rVar3 leží v rámci hranic, rRetVar se nastaví
                                                    na hodnotu 19,7.
rVar3=1.8
rRetVar=BOUND(rVar1,rVar2,rVar3)                  ; Proměnná rVar3 leží pod hranicí minima, rRetVar se
                                                    nastaví na hodnotu 10,5.
rVar3=45.2
rRetVar=BOUND(rVar1,rVar2,rVar3)                  ; Proměnná rVar3 leží nad hranicí maxima, rRetVar se
                                                    nastaví na hodnotu 33,7.
```

1.7 Priorita operací

Funkce

Každému operátoru je přiřazena priorita. Při vyhodnocení výrazu jsou stále nejprve použity operátory vyšší s prioritou. U operátorů stejné úrovně proběhne vyhodnocení zleva doprava.

V aritmetických výrazech může být pomocí kulatých závorek definováno pořadí zpracování všech operátorů, které může být odlišné od normálních pravidel priority operací.

Posloupnost operátorů

Od nejvyšší po nejnižší prioritu

1.	NOT, B_NOT	Negace, bitová negace
2.	*, /, DIV, MOD	Násobení, dělení
3.	+, –	Sečítání, odečítání
4.	B_AND	bitové logické a
5.	B_XOR	bitové logické XOR
6.	B_OR	bitové logické nebo
7.	AND	logické a
8.	XOR	logické XOR
9.	OR	logické nebo
10.	<<	Zřetězení řetězců, typ výsledku STRING
11.	==, <>, >, <, >=, <=	Relační operátory

Poznámka

Operátor zřetězení ":" pro framy se nesmí vyskytnout s ostatními operátory v jednom výrazu. Zařazení tohoto operátoru do stupnice priority proto není nutné.

Příklad: Příkaz IF

```
If (otto==10) and (anna==20) gotof end
```

1.8 Možné převádění typů

Funkce

Konverze typu při přiřazení

Konstantní číselná hodnota, proměnná nebo výraz, který je přiřazen proměnné, musí být kompatibilní s tímto typem proměnné. Je-li tomu tak, je při přiřazení typ automaticky převáděn.

Možné konverze typu

na	REAL	INT	BOOL	CHAR	STRING	AXIS	FRAME
z							
REAL	ano	ano*	ano[1])	ano*	–	–	–
INT	ano	ano	ano[1])	ano[2])	–	–	–
BOOL	ano	ano	ano	ano	ano	–	–
CHAR	ano	ano	ano[1])	ano	ano	–	–
STRING	–	–	ano[4])	ano[3])	ano	–	–
AXIS	–	–	–	–	–	ano	–
FRAME	–	–	–	–	–	–	ano

Vysvětlení

* Při konverzi z typu REAL na typ INT se desetinná část >= 0,5 zaokrouhluje nahoru, jinak dolů (srov. s funkcí ROUND).

1) Hodnotě 0 odpovídá TRUE, hodnotě == 0 odpovídá FALSE

2) Když hodnota leží v přípustném intervalu čísel

3) Když jen jeden znak

4) Délka řetězce 0 = FALSE, jinak TRUE

Poznámka

Je-li při konverzi některá hodnota větší než cílová oblast, bude to mít za následek chybové hlášení.

Objeví-li se v jednom výrazu smíšené typy, tak je automaticky provedeno přizpůsobení typu. Převody datových typů jsou možné i v synchronních akcích, viz kapitola "Pohybové synchronní akce, implicitní konverze typů".

1.9 Operace s řetězci

Operace s řetězci

Vedle klasických operací "přiřazování" a "Porovnávání" je s řetězci možno provádět ještě i následující operace:

- Převod do datového typu STRING (AXSTRING) [Strana 76]
- Převod z datového typu STRING (NUMBER, ISNUMBER, AXNAME) [Strana 77]
- Zřetězení proměnných typu String (<<) [Strana 78]
- Převádění na malá/velká písmena (TOLOWER, TOUPPER) [Strana 79]
- Zjištění délky řetězce (STRLEN) [Strana 80]
- Vyhledávání znaků/řetězců v řetězci (INDEX, RINDEX, MINDEX, MATCH) [Strana 81]
- Vybírání dílčího řetězce (SUBSTR) [Strana 82]
- Vybírání jednotlivých znaků (STRINGVAR, STRINGFELD) [Strana 83]
- Formátování řetězce (SPRINT) [Strana 84]

Zvláštní význam znaku 0

Znak 0 je interně interpretován jako identifikace konce řetězce. Je-li znak nahrazen znakem 0, je tím řetězec zkrácen.

Příklad:

Programový kód	**Komentář**
`DEF STRING[20] STRG="Achse . steht"`	
`STRG[6]="X"`	
`MSG(STRG)`	`; Výpis hlášení "Achse X steht".`
`STRG[6]=0`	
`MSG(STRG)`	`; Výpis hlášení "Achse".`

1.9.1 Převod do datového typu STRING (AXSTRING)

Funkce

Pomocí funkce "Převod do datového typu STRING" se dají využít proměnné rozdílných typů jako součást jednoho hlášení (MSG).

Konverze proběhne při použití operátoru **<<** implicitně pro všechna data typu INT, REAL, CHAR a BOOL (viz "Zřetězení proměnných typu String (<<) [Strana 78]“).

Hodnota typu INT bude převedena do normálně čitelné formy. U hodnot typu REAL bude uvedeno až 10 míst za čárkou.

Pomocí příkazu `AXSTRING` mohou být na proměnné typu STRING převáděny také proměnné typu AXIS.

Syntaxe

```
<STRING_ERG> = << <libovolný typ>
<STRING_ERG> = AXSTRING(<identifikátor osy>)
```

Význam

`<STRING_ERG>`:	Proměnná pro výsledek převodu typu Typ: STRING
`<libovolný typ>`:	Proměnné typu INT, REAL, CHAR, STRING a BOOL
`AXSTRING`:	Výsledkem příkazu `AXSTRING` je specifikovaný identifikátor osy jako řetězec.
`<identifikátor osy>`:	Proměnná pro identifikátor osy Typ: AXIS

Poznámka

Proměnné typu FRAME konvertovat nelze.

Příklady

Příklad 1:

```
MSG("Position:"<<$AA_IM[X])
```

Příklad 2: AXSTRING

Programový kód	Komentář
`DEF STRING[32] STRING_ERG`	
`STRING_ERG=AXSTRING(X)`	`; STRING_ERG == "X"`

1.9.2 Převod z datového typu STRING (NUMBER, ISNUMBER, AXNAME)

Funkce

Prostřednictvím funkce NUMBER se provádí konverze z typu STRING do typu REAL. To, zda je konverze vůbec možná, lze zkontrolovat pomocí příkazu ISNUMBER.

Pomocí příkazu AXNAME je možné proměnnou typu STRING převádět na datový typ AXIS.

Syntaxe

```
<REAL_ERG>=NUMBER("<řetězec>")
<BOOL_ERG>=ISNUMBER("<řetězec>")
<AXIS_ERG>=AXNAME("<řetězec>")
```

Význam

NUMBER:	Výsledkem příkazu NUMBER je řetězcem <řetězec> představované číslo uložené jako reálná hodnota.		
<řetězec>:	Proměnná typu STRING, která má být konvertována		
<REAL_ERG>:	Proměnná pro výsledek převodu typu pomocí příkazu NUMBER		
	Typ:	REAL	
ISNUMBER:	Pomocí příkazu ISNUMBER je možné zkontrolovat, může-li být řetězec <řetězec> převeden na platné číslo.		
<BOOL_ERG>:	Proměnná pro výsledek dotazu pomocí příkazu ISNUMBER		
	Typ:	BOOL	
	Hodnota:	TRUE	Výsledkem funkce ISNUMBER je hodnota TRUE, když řetězec <řetězec> představuje podle pravidel jazyka platné reálné číslo.
		FALSE	Jestliže je výsledkem funkce ISNUMBER hodnota FALSE, je při vyvolání funkce NUMBER se stejným řetězcem <řetězec> spuštěn alarm.
AXNAME:	Funkce AXNAME převádí udaný řetězec <řetězec> na identifikátor osy.		
	Upozornění: Pokud řetězec <řetězec> není možné přiřadit žádnému z identifikátorů os nastavených v konfiguraci, aktivuje se alarm.		
<AXIS_ERG>:	Proměnná pro výsledek převodu typu pomocí příkazu AXNAME		
	Typ:	AXIS	

Příklad

Programový kód	Komentář
DEF BOOL BOOL_ERG	
DEF REAL REAL_ERG	
DEF AXIS AXIS_ERG	
BOOL_ERG=ISNUMBER("1234.9876Ex-7")	; BOOL_ERG == TRUE
BOOL_ERG=ISNUMBER("1234XYZ")	; BOOL_ERG == FALSE
REAL_ERG=NUMBER("1234.9876Ex-7")	; REAL_ERG == 1234.9876Ex-7
AXIS_ERG=AXNAME("X")	; AXIS_ERG == X

1.9.3 Zřetězení proměnných typu String (<<)

Funkce

Funkce "Zřetězení proměnných typu String" poskytuje možnost skládat řetězec z jednotlivých součástí.

Řetězení je realizováno prostřednictvím operátoru "<<". Tento operátor má pro všechny kombinace základních typů CHAR, BOOL, INT, REAL a STRING jako výsledný typ STRING. Případně nutná konverze je provedena dle stávajících pravidel.

Syntaxe

```
<libovolný typ> << <libovolný typ>
```

Význam

<libovolný typ>:	Proměnné typu CHAR, BOOL, INT, REAL nebo STRING
<< :	Operátor pro zřetězení proměnných (<libovolný typ>) do takto poskládaného řetězce znaků (typ STRING). Tento operátor je k dispozici i samostatně jako tzv. „unární“ varianta. Tak je možné provést explicitní změnu do typu STRING (ne pro typy FRAME a AXIS): << <libovolný typ>

Například se tak dá složit hlášení nebo povel z textových seznamů a dají se doplnit parametry (asi jedno jméno prvku):
`MSG(STRG_TAB[LOAD_IDX]<<BAUSTEIN_NAME)`

POZOR
Mezivýsledky při zřetězení typu string nesmí překročit maximální délku řetězce.

Poznámka

Typy FRAME a AXIS nemohou být použity spolu s operátorem "<<".

Příklady

Příklad 1: Zřetězení proměnných typu String

```
Programový kód                                    Komentář
DEF INT IDX=2
DEF REAL VALUE=9.654
DEF STRING[20] STRG="INDEX:2"
IF STRG=="Index:"<<IDX GOTOF NO_MSG
MSG("Index:"<<IDX<<"/Wert:"<<VALUE)               ; Vypíše se:
                                                    "Index:2/Wert:9.654"
NO_MSG:
```

Příklad 2: Explicitní konverze typu pomocí příkazu <<

```
Programový kód          Komentář
DEF REAL VALUE=3.5
<<VALUE                 ; Uvedená proměnná typu REAL je konvertována na typ
                          STRING.
```

1.9.4 Převádění na malá/velká písmena (TOLOWER, TOUPPER)

Funkce

Funkce "Převádění na malá/velká písmena" umožňuje změnit všechna písmena řetězce znaků do jednotné formy.

Syntaxe

```
<STRING_ERG>=TOUPPER("<řetězec>")
<STRING_ERG>=TOLOWER("<řetězec>")
```

Význam

TOUPPER:	Prostřednictvím příkazu TOUPPER jsou všechna písmena v řetězci znaků převedena na **velká** písmena.
TOLOWER:	Prostřednictvím příkazu TOLOWER jsou všechna písmena v řetězci znaků převedena na **malá** písmena.
<řetězec>:	Řetězec znaků, který má být převeden. Typ: STRING
<STRING_ERG>:	Proměnná pro výsledek převodu Typ: STRING

Příklad

Protože uživatelské vstupy mohou být zobrazovány také na uživatelském rozhraní, lze jim dát standardní formu (velká nebo malá písmena):

Programový kód

```
DEF STRING [29] STRG
...
IF "LEARN.CNC"==TOUPPER(STRG) GOTOF LOAD_LEARN
```

1.9.5 Zjištění délky řetězce (STRLEN)

Funkce

Prostřednictvím příkazu `STRLEN` můžete zjistit délku řetězce znaků.

Syntaxe

```
<INT_ERG>=STRLEN("<STRING>")
```

Význam

`STRLEN`:	Prostřednictvím příkazu `STRLEN` se určuje délka zadaného řetězce znaků. Výsledkem bude počet znaků, které – počítáno od začátku řetězce znaků – nejsou znakem 0.
`<řetězec>`:	Řetězec znaků, jehož délka má být určena. Typ: STRING
`<INT_ERG>`:	Proměnná pro výsledek určování Typ: INT

Příklad

Funkce v souvislosti s přístupem k jednotlivým znakům umožňuje stanovit konec řetězce znaků.

Programový kód

```
IF(STRLEN (BAUSTEIN_NAME) > 10) GOTOF FEHLER
```

1.9.6 Vyhledávání znaků/řetězců v řetězci (INDEX, RINDEX, MINDEX, MATCH)

Funkce

Tato funkce umožňuje hledat jednotlivé znaky popř. řetězec v jiné proměnné typu STRING. Výsledky této funkce udávají, na které pozici řetězce byl znak/řetězec v prohledávaném řetězci nalezen.

Syntaxe

`INT_ERG=INDEX(STRING,CHAR)` ; **Typ výsledků: INT**

`INT_ERG=RINDEX(STRING,CHAR)` ; **Typ výsledků: INT**

`INT_ERG=MINDEX(STRING,STRING)` ; **Typ výsledků: INT**

`INT_ERG=MATCH(STRING,STRING)` ; **Typ výsledků: INT**

Sémantika

Vyhledávací funkce: Zobrazují pozici v řetězci (první parametr), kde bylo hledání úspěšné. Jestliže znak/řetězec nemůže být nalezen, je vracena hodnota –1. První znak má přitom pozici 0.

Význam

`INDEX`:	Hledá znak zadaný jako druhý parametr (odpředu) v prvním parametru.
`RINDEX`:	Hledá znak zadaný jako druhý parametr (odzadu) v prvním parametru.
`MINDEX`:	Odpovídá funkci INDEX, s výjimkou toho, že je předán seznam znaků (jako řetězec), ze kterých je předán index prvního nalezeného znaku.
`MATCH`:	Hledá řetězec v řetězci.

Tak se dají řetězce rozložit podle určitých kritérií, např. podle pozic se znakem mezery nebo s oddělovacím znakem („/“) v udání cesty.

Příklad

Rozložení zadání na název cesty a název modulu

```
Programový kód                                          Komentář
DEF INT PFADIDX, PROGIDX
DEF STRING[26] EINGABE
DEF INT LISTIDX
EINGABE = "/_N_MPF_DIR/_N_EXECUTE_MPF"
LISTIDX = MINDEX (EINGABE, "M,N,O,P") + 1               ; Jako hodnota v proměnné LISTIDX je
                                                          vraceno 3, protože "N" je první znak
                                                          od předu v parametru EINGABE ze
                                                          seznamu možností.
```

Programový kód	Komentář
PFADIDX = INDEX (EINGABE, "/") +1	; Přitom tedy platí: PFADIDX = 1
PROGIDX = RINDEX (EINGABE, "/") +1	; Přitom tedy platí: PROGIDX = 12
	S pomocí v další části uvedené funkce SUBSTR se dá proměnná EINGABE rozložit na součásti "cesta" a "modul":
VARIABLE = SUBSTR (EINGABE, PFADIDX, PROGIDX-PFADIDX-1)	; Výsledkem pak je "_N_MPF_DIR"
VARIABLE = SUBSTR (EINGABE, PROGIDX)	; Výsledkem pak je "_N_EXECUTE_MPF"

1.9.7 Vybírání dílčího řetězce (SUBSTR)

Funkce

Tato funkce umožňuje vytáhnout z určitého řetězce nějaký dílčí řetězec. Za tím účelem se zadá index prvního znaku a případně požadovaná délka. Není-li zadána informace o délce, je míněna zbývající část řetězce.

Syntaxe

`STRING_ERG = SUBSTR (STRING,INT)` ; **Typ výsledků: INT**

`STRING_ERG = SUBSTR(STRING,INT, INT)` ; **Typ výsledků: INT**

Sémantika

V prvním případě je dílčí řetězec převzat od pozice, která je určena druhým parametrem, až do konce řetězce.

Ve druhém případě je výsledný řetězec omezen na maximální délku, danou třetím parametrem.

Leží-li počáteční pozice za koncem řetězce, je převzat prázdný (“ “) řetězec.

Je-li počáteční pozice nebo délka negativní, aktivuje se alarm.

Příklad

Programový kód	Komentář
DEF STRING [29] ERG	
ERG = SUBSTR ("QUITTUNG:10 bis 99", 10, 2)	; Přitom tedy platí: ERG == "10"

1.9.8 Vybírání jednotlivých znaků (STRINGVAR, STRINGFELD)

Funkce

Tato funkce umožňuje z řetězce vybírat jednotlivé znaky. Toto se týká jak čtecích, tak zápisových přístupových operací.

Syntaxe

CHAR_ERG = STRINGVAR [IDX] ; **Typ výsledků: CHAR**

CHAR_ERG = STRINGFELD [IDX_FELD, IDX_CHAR] ; **Typ výsledků: CHAR**

Sémantika

Bude načten/zapsán ten znak, který se nachází na udaném místě. Je-li zadání pozice negativní nebo větší než délka řetězce, je aktivován alarm.

Příklad hlášení:

Dosazení identifikátoru osy do předem vytvořeného řetězce.

Programový kód	Komentář
`DEF STRING [50] MELDUNG = "Osa n dosahla pozice"`	
`MELDUNG [6] = "X"`	
`MSG (MELDUNG)`	`; Způsobí výpis hlášení "Osa X dosahla pozice"`

Parametry

Přístup k jednotlivým znakům je možný pouze u uživatelem definovaných proměnných (LUD-, GUD- a PUD-data).

Kromě toho je tento druh přístupu při vyvolání podprogramu možný jen pro parametry typu „Call-By-Value“.

Příklady

Příklad 1: Přístup k jednotlivým znakům systémových a strojních parametrů

Programový kód	Komentář
`DEF STRING [50] STRG`	
`DEF CHAR QUITTUNG`	
…	
`STRG = $P_MMCA`	
`QUITTUNG = STRG [0]`	`; Vyhodnocování potvrzovací složky`

Příklad 2: Přístup k jednotlivým znakům u parametru typu "Call-By-Reference"

Programový kód	Komentář
DEF STRING [50] STRG	
DEF CHAR CHR1	
EXTERN UP_CALL (VAR CHAR1)	; Parametr typu "Call-By-Reference"!
…	
CHR1 = STRG [5]	
UP_CALL (CHR1)	; Call-By-Reference
STRG [5] = CHR1	

1.9.9 Formátování řetězce (SPRINT)

Funkce

Prostřednictvím předdefinované funkce SPRINT mohou být formátovány řetězce znaků a mohou být tak připravovány např. pro výstup na externím zařízení (viz také "Výstup do externího zařízení/souboru (EXTOPEN, WRITE, EXTCLOSE) [Strana 708]").

Syntaxe

```
"<Výsledek_řetězec>"=SPRINT("<Formát_řetězce>",<Hodnota_1>,<Hodnota_
2>,..., <Hodnota_n>)
```

Význam

SPRINT:	Identifikátor předdefinované funkce, jejímž výsledkem je hodnota typu STRING.
"<Formát_řetězce>":	Řetězec znaků, který obsahuje pevné a proměnné složky. Proměnné složky jsou definovány pomocí řídícího formátovacího znaku % a následným popisem formátu.
<Hodnota_1>,<Hodnota_2>,…, <Hodnota_n>:	Hodnota ve formě konstanty nebo proměnné NC systému, která se vkládá na místo, na němž se nachází n-tý řídící formátovací znak v souladu s popisem formátu v parametru <Formát_řetězce>.
"<Výsledek_řetězec>":	Formátovaný řetězec znaků (maximálně 400 bytů)

Popisy formátů, které jsou k dispozici

%B:	Přeměna na řetězec "TRUE", pokud se převáděná hodnota:
	• nerovná 0.
	• nerovná mezeře (v případě hodnot typu STRING).
	Přeměna na řetězec "FALSE", pokud se převáděná hodnota:
	• rovná 0.
	• je mezerou.
	Příklad:
	`N10 DEF BOOL BOOL_VAR=1`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF BOOL_VAR:%B", BOOL_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF BOOL_VAR:TRUE".

%C:	Přeměna na řetězec ASCII znaků.
	Příklad:
	`N10 DEF CHAR CHAR_VAR="X"`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF CHAR_VAR:%C",CHAR_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF CHAR_VAR:X".

%D:	Přeměna na řetězec s celočíselnou hodnotou (INTEGER).
	Příklad:
	`N10 DEF INT INT_VAR=123`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF INT_VAR:%D",INT_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF INT_VAR:123".

%<m>D:	Přeměna na řetězec s celočíselnou hodnotou (INTEGER). Řetězec má minimální délku <m> znaků. Chybějící místa jsou zleva doplněna mezerami.
	Příklad:
	`N10 DEF INT INT_VAR=-123`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF INT_VAR:%6D",INT_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF INT_VAR:xx-123" ("x" v příkladu reprezentuje znak "mezera").

%F:	Přeměna na řetězec s desetinným číslem, které má 6 míst za desetinnou tečkou. Podle potřeby jsou místa za desetinnou tečkou zaokrouhlena nebo doplněna 0.
	Příklad:
	`N10 DEF REAL REAL_VAR=-1.2341234EX+03`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%F",REAL_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF REAL_VAR: -1234.123400".

%<m>F:	Přeměna na řetězec s desetinným číslem, které má 6 míst za desetinnou tečkou a celkovou délku minimálně <m> znaků. Podle potřeby jsou místa za desetinnou tečkou zaokrouhlena nebo doplněna 0. Znaky, které chybějí, aby bylo dosaženo celkové délky <m>, jsou zleva doplněna mezerami.
	Příklad:
	`N10 DEF REAL REAL_VAR=-1.23412345678EX+03`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%15F",REAL_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF REAL_VAR: xxx-1234.123457" ("x" v příkladu reprezentuje znak "mezera").

%.<n>F:	Přeměna na řetězec s desetinným číslem, které má <n> míst za desetinnou tečkou. Podle potřeby jsou místa za desetinnou tečkou zaokrouhlena nebo doplněna 0.
	Příklad:
	`N10 DEF REAL REAL_VAR=-1.2345678EX+03`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%.3F",REAL_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF REAL_VAR: -1234.568".

%<m>.<n>F:	Přeměna na řetězec s desetinným číslem, které má <n> míst za desetinnou tečkou a celkovou délku minimálně <m> znaků. Podle potřeby jsou místa za desetinnou tečkou zaokrouhlena nebo doplněna 0. Znaky, které chybějí, aby bylo dosaženo celkové délky <m>, jsou zleva doplněna mezerami.
	Příklad:
	`N10 DEF REAL REAL_VAR=-1.2341234567890EX+03`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%10.2F",REAL_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF REAL_VAR:xx-1234.12" ("x" v příkladu reprezentuje znak "mezera").

%E:	Přeměna na řetězec s desetinným číslem v exponenciálním způsobu zápisu. Mantisa je normalizována na jedno místo před desetinnou tečkou a 6 míst za desetinnou tečkou. Podle potřeby jsou místa za desetinnou tečkou zaokrouhlena nebo doplněna 0. Exponent začíná klíčovým slovem "EX". Potom následuje znaménko "+" nebo "-") a dvoj- nebo trojmístné číslo.
	Příklad:
	`N10 DEF REAL REAL_VAR=-1234.567890`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%E",REAL_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF REAL_VAR:-1.234568EX+03".

%<m>E:	Přeměna na řetězec s desetinným číslem v exponenciálním formátu, který má celkovou délku minimálně <m> znaků. Chybějící znaky jsou zleva doplněny mezerami. Mantisa je normalizována na jedno místo před desetinnou tečkou a 6 míst za desetinnou tečkou. Podle potřeby jsou místa za desetinnou tečkou zaokrouhlena nebo doplněna 0. Exponent začíná klíčovým slovem "EX". Potom následuje znaménko ("+" nebo "-") a dvoj- nebo trojmístné číslo.
	Příklad:
	`N10 DEF REAL REAL_VAR=-1234.5`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%20E",REAL_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF REAL_VAR:xxxxxx-1.234500EX+03" ("x" v příkladu reprezentuje znak "mezera").

%.<n>E:	Přeměna na řetězec s desetinným číslem v exponenciálním způsobu zápisu. Mantisa je normalizována na jedno místo před desetinnou tečkou a <n> míst za desetinnou tečkou. Podle potřeby jsou místa za desetinnou tečkou zaokrouhlena nebo doplněna 0. Exponent začíná klíčovým slovem "EX". Potom následuje znaménko ("+" nebo "-") a dvoj- nebo trojmístné číslo.
	Příklad:
	`N10 DEF REAL REAL_VAR=-1234.5678`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%.2E",REAL_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF REAL_VAR:-1.23EX+03".

%<m>.<n>E:	Přeměna na řetězec s desetinným číslem v exponenciálním formátu, který má celkovou délku minimálně <m> znaků. Chybějící znaky jsou zleva doplněny mezerami. Mantisa je normalizována na jedno místo před desetinnou tečkou a <n> míst za desetinnou tečkou. Podle potřeby jsou místa za desetinnou tečkou zaokrouhlena nebo doplněna 0. Exponent začíná klíčovým slovem "EX". Potom následuje znaménko ("+" nebo "-") a dvoj- nebo trojmístné číslo.
	Příklad:
	`N10 DEF REAL REAL_VAR=-1234.5678`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%12.2E", REAL_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF REAL_VAR:xx-1.23EX+03" ("x" v příkladu reprezentuje znak "mezera").

%G:	Přeměna na řetězec s desetinným číslem buď v desetinném nebo v exponenciálním způsobu zápisu, v závislosti na rozsahu hodnot: Pokud je zobrazovaná hodnota menší než 1.0EX-04 nebo větší/rovna 1.0EX+06, zvolí se exponenciální způsob zápisu, jinak se použije formát desetinného čísla. Vypisuje se maximálně šest významových míst, v případě nutnosti se číslo zaokrouhlí.
	Příklad s formátem desetinného čísla:
	`N10 DEF REAL REAL_VAR=1.234567890123456EX-04`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%G",REAL_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF REAL_VAR: 0.000123457".
	Příklad s exponenciálním způsobem zápisu:
	`N10 DEF REAL REAL_VAR=1.234567890123456EX+06`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%G",REAL_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF REAL_VAR:1.23457EX+06".

%<m>G:	Přeměna na řetězec s desetinným číslem buď v desetinném nebo v exponenciálním způsobu zápisu, v závislosti na rozsahu hodnot (jako %G). Řetězec má celkovou délku minimálně <m> znaků. Chybějící znaky jsou zleva doplněny mezerami.
	Příklad s formátem desetinného čísla:
	`N10 DEF REAL REAL_VAR=1.234567890123456EX-04`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%15G",REAL_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF REAL_VAR:xxxx0.000123457" ("x" v příkladu reprezentuje znak "mezera").
	Příklad s exponenciálním způsobem zápisu:
	`N10 DEF REAL REAL_VAR=1.234567890123456EX+06`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%15G",REAL_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF REAL_VAR:xxx1.23457EX+06" ("x" v příkladu reprezentuje znak "mezera").

%.<n>G:	Přeměna na řetězec s desetinným číslem buď v desetinném nebo v exponenciálním způsobu zápisu, v závislosti na rozsahu hodnot. Vypisuje se maximálně <n> významových míst, v případě nutnosti se číslo zaokrouhlí. Pokud je zobrazovaná hodnota menší než 1.0EX-04 nebo větší/rovna 1.0EX(+<n>), zvolí se exponenciální způsob zápisu, jinak se použije formát desetinného čísla.
	Příklad s formátem desetinného čísla:
	`N10 DEF REAL REAL_VAR=1.234567890123456EX-04`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%.3G",REAL_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF REAL_VAR: 0.000123".
	Příklad s exponenciálním způsobem zápisu:
	`N10 DEF REAL REAL_VAR=1.234567890123456EX+03`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT = SPRINT("CONTENT OF REAL_VAR:%.3G",REAL_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF REAL_VAR:1.23EX+03".

%<m>.<n>G:	Přeměna na řetězec s desetinným číslem buď v desetinném nebo v exponenciálním způsobu zápisu, v závislosti na rozsahu hodnot (jako %.<n>G). Řetězec má celkovou délku minimálně <m> znaků. Chybějící znaky jsou zleva doplněny mezerami.
	Příklad s formátem desetinného čísla:
	`N10 DEF REAL REAL_VAR=1.234567890123456EX-04`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%12.4G",REAL_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF REAL_VAR:xxx0.0001235" ("x" v příkladu reprezentuje znak "mezera").
	Příklad s exponenciálním způsobem zápisu:
	`N10 DEF REAL REAL_VAR=1.234567890123456EX+04`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%12.4G",REAL_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF REAL_VAR:xx1.235EX+06" ("x" v příkladu reprezentuje znak "mezera").

%.<n>P:	Přeměna hodnoty typu REAL na hodnotu typu INTEGER, přičemž se bere v úvahu <n> míst za desetinnou tečkou. Výsledná hodnota typu INTEGER je reprezentována 32-bitovým binárním číslem. Pokud převáděná hodnota nemůže být zobrazena pomocí 32 bitů, operace se přeruší a aktivuje se alarm.
	Protože posloupnost bytů generovaná formátovacím příkazem %.<n>P může obsahovat také binární nuly, celkový vytvořený řetězec již neodpovídá konvencím pro datový typ STRING NC-systému. Proto jej není možné ukládat do proměnných typu STRING a ani jej nelze dále zpracovávat pomocí příkazů NC jazyka, které jsou určeny pro řetězce. Jediným možným použitím je předávání parametrů do příkazu WRITE s výstupem na odpovídající externí zařízení (viz následující příklad).
	Jestliže parametr <Formát_řetězce> obsahuje popis formátu typu %P, bude na výstup odeslán celý řetězec, s výjimkou binárního čísla generovaného pomocí parametru %.<n>P, v závislosti na nastavení parametru MD10750 $MN_SPRINT_FORMAT_P_CODE ve znakovém kódu ASCII, ISO (DIN 6024) nebo EIA (RS 244). Jestliže je naprogramován znak, který nemůže být převeden, bude zpracování přerušeno a aktivuje se alarm.
	Příklad:
	`N10 DEF REAL REAL_VAR=123.45`
	`N20 DEF INT ERROR`
	`N30 DEF STRING[20] EXT_DEVICE="/ext/dev/1"`
	`...`
	`N100 EXTOPEN(ERROR,EXT_DEVICE)`
	`N110 IF ERROR <> 0`
	`... ; zpracování chyby`
	`N200 WRITE(ERROR,EXT_DEVICE,SPRINT("INTEGER BINARY CODED:%.3P",REAL_VAR)`
	`N210 IF ERROR <> 0`
	`... ; zpracování chyby`
	Výsledek: Řetězec "INTEGER BINARY CODED: 'H0001E23A'" se přenese na výstupní zařízení /ext/dev/1. Hexadecimální hodnota 0x0001E23A odpovídá decimální hodnotě 123450.

%<m>.<n>P:	Převod hodnoty typu REAL na řetězec v souladu s nastavením strojního parametru MD10751 $MN_SPRINT_FORMAT_P_DECIMAL, přičemž tímto řetězcem je:
	• celé číslo s <m> + <n> místy **nebo**
	• desetinné číslo s maximální <m> místy před desetinnou tečkou a přesně <n> místy za desetinnou tečkou.
	Stejně jako v případě popisu formátu %.<n>P se celý řetězec ukládá ve znakovém kódu definovaném v parametru MD10750 $MN_SPRINT_FORMAT_P_CODE.
	Přeměna v případě MD10751 = 0:
	Hodnota typu REAL je převedena na řetězec s celým číslem, které má <m> + <n> míst. V případě potřeby jsou místa za desetinnou tečkou zaokrouhlena na <n> míst nebo doplněna 0. Chybějící místa před desetinnou tečkou jsou doplněna mezerami. Znaménko mínus se připojuje z levé strany, místo znaménka plus se vkládá mezera.
	Příklad:
	`N10 DEF REAL REAL_VAR=-123.45`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("PUNCHED TAPE FORMAT:%5.3P",REAL_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "PUNCHED TAPE FORMAT:-xx123450" ("x" v příkladu reprezentuje znak "mezera").
	Přeměna v případě MD10751 = 1:
	Hodnota typu REAL je převedena na řetězec s desetinným číslem, které má maximálně <m> míst před desetinnou tečkou a přesně <n> míst za desetinnou tečkou. V případě potřeby jsou místa před desetinnou tečkou oříznuta a místa za desetinnou tečkou zaokrouhlena nebo doplněna 0. Jestliže se <n> rovná 0, odpadá i desetinná tečka.
	Příklad:
	`N10 DEF REAL REAL_VAR1=-123.45`
	`N20 DEF REAL REAL_VAR2=123.45`
	`N30 DEF STRING[80] RESULT`
	`N40 RESULT=SPRINT("PUNCHED TAPE FORMAT:%5.3P VAR2:%2.0P",`
	`REAL_VAR1,REAL_VAR2)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "PUNCHED TAPE FORMAT:-123.450 VAR2:23".

%S:	Slouží pro vkládání řetězce.
	Příklad:
	`N10 DEF STRING[16] STRING_VAR="ABCDEFG"`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF STRING_VAR:%S",STRING_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF STRING_VAR:ABCDEFG".

%<m>S:	Vložení řetězce s minimálně <m> znaky. Chybějící místa jsou doplněna mezerami.
	Příklad:
	`N10 DEF STRING[16] STRING_VAR="ABCDEFG"`
	`N20 DEF STRING[80] RESULT`
	`N30 RESULT=SPRINT("CONTENT OF STRING_VAR:%10S",STRING_VAR)`
	Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF STRING_VAR:xxxABCDEFG" ("x" v příkladu reprezentuje znak "mezera").

%.<n>S:	Vložení <n> znaků řetězce (počínaje prvním znakem). **Příklad:** N10 DEF STRING[16] STRING_VAR="ABCDEFG" N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF STRING_VAR:%.3S",STRING_VAR) Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF STRING_VAR:ABC".
%<m>.<n>S:	Vložení <n> znaků řetězce (počínaje prvním znakem). Celková délka sestavovaného řetězce je minimálně <m> znaků. Chybějící místa jsou doplněna mezerami. **Příklad:** N10 DEF STRING[16] STRING_VAR="ABCDEFG" N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF STRING_VAR:%10.5S", STRING_VAR) Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "CONTENT OF STRING_VAR:xxxABCDE" ("x" v příkladu reprezentuje znak "mezera").
%X:	Převod hodnoty typu INTEGER na řetězec v hexadecimálním způsobu zápisu. **Příklad:** N10 DEF INT INT_VAR='HA5B8' N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("INTEGER HEXADECIMAL:%X",INT_VAR) Výsledek: Do proměnné RESULT typu STRING bude zapsán řetězec "INTEGER HEXADECIMAL:A5B8".

Poznámka

Konvence jazyka NC systému týkající se rozlišování velkých a malých písmen v názvech identifikátorů a v klíčových slovech platí také pro popisy formátů. Při programování proto můžete používat velká i malá písmena, aniž by to mělo na funkce nějaký vliv.

Možnosti kombinování

V následující tabulce naleznete přehled o tom, které datové typy NC systémů a které popisy formátování mohou být kombinovány. Platí také pravidla pro implicitní převody datových formátů (viz "Datové typy [Strana 55]").

	Datové typy NC systému						
	BOOL	**CHAR**	**INT**	**REAL**	**STRING**	**AXIS**	**FRAME**
%B	+	+	+	+	+	-	-
%C	-	+	-	-	+	-	-
%D	+	+	+	+	-	-	-
%F	-	-	+	+	-	-	-
%E	-	-	+	+	-	-	-
%G	-	-	+	+	-	-	-
%S	-	+	-	-	+	-	-
%X	+	+	+	-	-	-	-
%P	-	-	+	+	-	-	-

Poznámka

Tato tabulka ukazuje, že NC datové typy AXIS a FRAME není možné pomocí funkce SPRINT převádět přímo. Je ale možné:

- Datový typ AXIS je možné převádět na řetězec pomocí funkce AXSTRING, který je potom možné dále zpracovávat pomocí funkce SPRINT.
- Jednotlivé hodnoty datového typu FRAME je možné načítat pomocí funkcí pro přístup k jednotlivým složkám framu. Tímto způsobem získáte data typu REAL, která je pak možné pomocí příkazu SPRINT dále zpracovávat.

1.10 Programové skoky a větvení

1.10.1 Skok zpátky na začátek programu (GOTOS)

Funkce

Pomocí příkazu GOTOS je možné skočit zpět na začátek hlavního programu nebo podprogramu za účelem jeho opakovaného zpracování.

Prostřednictvím strojních parametrů je možné nastavit, aby se při každém skoku zpět na začátek programu uskutečňovaly následující operace:

- Nastavení doby zpracovávání programu na "0".
- Zvýšení hodnoty počítadla obrobků o "1".

Syntaxe

```
GOTOS
```

Význam

GOTOS: Příkaz skoku, kdy cílem skoku je začátek programu.

Vlastní realizace je ovládána signálem rozhraní NC/PLC:
DB21, ... DBX384.0 (řízení větvení programu)

Hodnota:	Význam:
0	Žádný skok zpět na začátek programu. Zpracovávání programu bude pokračovat následujícím blokem výrobního programu za příkazem GOTOS.
1	Skok zpět na začátek programu. Výrobní program je opakován.

Okrajové podmínky

- Příkaz GOTOS interně spouští STOPRE (zastavení předběžného zpracování).
- U výrobních programů v definicí dat (proměnné LUD) se skok s příkazem GOTOS uskutečňuje na první programový blok za úsekem definic, tzn. definice dat se znovu neprovádějí. Definované proměnné si proto ponechávají hodnotu, kterou měly v bloku s příkazem GOTOS a nejsou jim tedy dosazeny standardní hodnoty naprogramované v úseku definic.
- V synchronních akcích a v technologických cyklech není příkaz GOTOS k dispozici.

Příklad

Programový kód	Komentář
N10 ...	; Začátek programu.
...	
N90 GOTOS	; Skok na začátek programu.
...	

1.10.2 Programové skoky na návěští skoků (GOTOB, GOTOF, GOTO, GOTOC)

Funkce

Do programu je možno vkládat návěští skoků (label), na která lze potom skákat z jiných míst téhož programu pomocí příkazů GOTOF, GOTOB, GOTO, příp. GOTOC. Zpracování programu potom pokračuje příkazem, který se nachází bezprostředně za návěštím skoku. Tímto způsobem je možné realizovat větvení v rámci programu.

Kromě návěští skoků je možno jako cíl skoku používat také čísla hlavních a vedlejších bloků.

Jestliže je před příkazem kroku formulována podmínka skoku (IF ...), potom se programový skok uskuteční jen tehdy, pokud je podmínka skoku splněna.

Syntaxe

```
GOTOB <cíl skoku>
IF <podmínka skoku> = TRUE GOTOB <cíl skoku>
GOTOF <cíl skoku>
IF <podmínka skoku> = TRUE GOTOF <cíl skoku>
GOTO <cíl skoku>
IF <podmínka skoku> = TRUE GOTO <cíl skoku>
GOTOC <cíl skoku>
IF <podmínka skoku> = TRUE GOTOC <cíl skoku>
```

Význam

GOTOB:	Příkaz skoku s cílem skoku hledaným směrem k začátku programu
GOTOF:	Příkaz skoku s cílem skoku hledaným směrem ke konci programu.
GOTO:	Příkaz skoku s vyhledáváním cíle skoku. Vyhledávání se uskutečňuje napřed směrem ke konci programu, potom směrem k začátku programu.
GOTOC:	Tento příkaz funguje stejně jako příkaz GOTO jen s tím rozdílem, že je potlačen alarm 14080 "Cíl skoku nenalezen". To znamená, že se zpracovávání programu v případě neúspěšného vyhledávání cíle skoku nepřeruší, nýbrž bude pokračovat na programovém řádku, který se nachází za příkazem GOTOC.

`<cíl skoku>`:	Parametr cíle skoku Možné údaje jsou následující:	
	<návěští skoku**>**:	Cíl skoku je návěští skoku umístěné v programu, jehož název je definován uživatelem: **<**`návěští skoku`**>:**
	<číslo bloku**>**:	Cíl skoku je číslo hlavního či vedlejšího bloku (např.: `200`, `N300`)
	Proměnná typu STRING:	Proměnný cíl skoku. Proměnná představuje návěští skoku nebo číslo bloku.
`IF`:	Klíčové slovo pro formulování podmínky skoku. Podmínka skoku připouští všechny porovnávací a logické operátory (výsledek TRUE nebo FALSE). Programový skok se uskuteční, jestliže výsledek této operace je TRUE.	

Poznámka

Návěští skoku (label)

Návěští skoku se vždy nachází na začátku bloku. Jestliže existuje také číslo programu, návěští skoku se zapisuje vždy bezprostředně za číslem bloku.

Pro určování názvů návěští skoků platí následující pravidla:

- Počet znaků:
  - minimálně 2
  - maximálně 32
- Povolenými znaky jsou:
  - Písmena
  - Číslice
  - Znak podtržení
- První dva znaky musí být písmena nebo znak podtržení.
- Za názvem návěští skoku následuje dvojtečka (":“).

Okrajové podmínky

- Cílem skoku může být pouze blok s návěštím skoku nebo s číslem bloku, které se nacházejí **uvnitř** daného programu.
- Příkaz skoku bez podmínky skoku musí být naprogramován v samostatném bloku. U příkazů skoku s podmínkou skoku toto omezení neplatí. V jednom bloku může být formulováno několik příkazů skoků.
- U programů s příkazy skoků bez podmínek skoků se může stát, že se konec programu`M2`/`M30` nebude nacházet na konci programu.

Příklady

Příklad 1: Skoky na návěští skoku

Programový kód	Komentář
N10 …	
N20 GOTOF Label_1	; Skok ve směru konce programu na návěští skoku "Label_1".
N30 …	
N40 Label_0: R1=R2+R3	; Návěští skoku "Label_0" nastaveno.
N50 …	
N60 Label_1:	; Návěští skoku "Label_1" nastaveno.
N70 …	
N80 GOTOB Label_0	; Skok ve směru začátku programu na návěští skoku "Label_0".
N90 …	

Příklad 2: Nepřímý skok na číslo bloku

Programový kód	Komentář
N5 R10=100	
N10 GOTOF "N"<<R10	; Skok na blok, jehož číslo bloku je uloženo v R10.
...	
N90 ...	
N100 ...	; Cíl skoku
N110 ...	
...	

Příklad 3: Skok na proměnný cíl skoku

Programový kód	Komentář
DEF STRING[20] ZIEL	
ZIEL = "Marke2"	
GOTOF ZIEL	; Skok ve směru konce programu na proměnný cíl skoku ZIEL.
Marke1: T="Bohrer1"	
...	
Marke2: T="Bohrer2"	; Cíl skoku
...	

Příklad 4: Skok s podmínkou skoku

Programový kód	Komentář
N40 R1=30 R2=60 R3=10 R4=11 R5=50 R6=20	; Přiřazení počátečních hodnot.
N41 LA1: G0 X=R2*COS(R1)+R5 Y=R2*SIN(R1)+R6	; Návěští skoku LA1 nastaveno.
N42 R1=R1+R3 R4=R4-1	
N43 IF R4>0 GOTOB LA1	; Jestliže je podmínka skoku splněna, potom se uskuteční skok ve směru začátku programu na návěští LA1.
N44 M30	; Konec programu

1.10.3 Větvení programu (CASE ... OF ... DEFAULT ...)

Funkce

Funkce CASE nabízí možnost překontrolovat aktuální hodnotu (typ: INT) nějaké proměnné nebo matematické funkce a v závislosti na výsledku skočit v programu na různá místa.

Syntaxe

```
CASE(<výraz>) OF <konstanta_1> GOTOF <cíl skoku_1> <konstanta_2>
GOTOF <cíl skoku_2> ... DEFAULT GOTOF <cíl skoku_n>
```

Význam

CASE:	Příkaz skoku
<výraz>:	Proměnná nebo matematická funkce
OF:	Klíčové slovo pro formulování podmíněného programového větvení.
<konstanta_1>:	První zadaná konstantní hodnota pro proměnnou nebo matematickou funkci Typ: INT
<konstanta_2>:	Druhá zadaná konstantní hodnota pro proměnnou nebo matematickou funkci Typ: INT
DEFAULT:	Pro případy, při kterých se hodnota proměnné nebo matematická funkce nerovná žádné z uvedených konstant, může být cíl skoku určen pomocí příkazu DEFAULT. **Upozornění:** Jestliže příkaz DEFAULT není naprogramován, použije se v těchto případech jako cíl skoku blok, který následuje za příkazem CASE.

GOTOF:	Příkaz skoku s cílem skoku hledaným směrem ke konci programu. Namísto příkazu GOTOF je možné naprogramovat také všechny ostatní příkazy GOTO (viz kapitola "Programové skoky na návěští skoku").
<cíl skoku_1>:	Na tento cíl skoku se odbočí, jestliže hodnota proměnné nebo matematické funkce odpovídá první ze zadaných konstant. Cíl skoku může být zadán následujícím způsobem:
	<návěští skoku>: Cíl skoku je návěští skoku umístěné v programu, jehož název je definován uživatelem: **<**návěští skoku**>:**
	<číslo bloku>: Cíl skoku je číslo hlavního či vedlejšího bloku (např.: 200, N300)
	Proměnná typu STRING: Proměnný cíl skoku. Proměnná představuje návěští skoku nebo číslo bloku.
<cíl skoku_2>:	Na tento cíl skoku se odbočí, jestliže hodnota proměnné nebo matematické funkce odpovídá druhé ze zadaných konstant.
<cíl skoku_n>:	Na tento cíl skoku se odbočí, jestliže hodnota proměnné nebo matematické funkce neodpovídá žádné z uvedených konstant.

Příklad

Programový kód

```
...
N20 DEF INT VAR1 VAR2 VAR3
N30 CASE(VAR1+VAR2-VAR3) OF 7 GOTOF Label_1 9 GOTOF Label_2 DEFAULT GOTOF Label_3
N40 Label_1: G0 X1 Y1
N50 Label_2: G0 X2 Y2
N60 Label_3: G0 X3 Y3
...
```

Příkaz CASE z bloku N30 definuje následující možnosti rozvětvení programu:

1. Jestliže se hodnota matematické funkce VAR1+VAR2-VAR3 = 7, potom se skáče na blok s definicí návěští skoku "Label_1" (→ N40).
2. Jestliže se hodnota matematické funkce VAR1+VAR2-VAR3 = 9, potom se skáče na blok s definicí návěští skoku "Label_2" (→ N50).
3. Jestliže se hodnota matematické funkce VAR1+VAR2-VAR3 nerovní ani 7 ani 9, potom se skáče na blok s definicí návěští skoku "Label_3" (→ N60).

1.11 Opakování části programu (REPEAT, REPEATB, ENDLABEL, P)

Funkce

Opakování části programu umožňuje opětovné zpracování již napsané části v rámci daného programu v libovolné podobě.

Řádky, příp. úsek programu, který se má opakovat, je označen návěštími skoku (label).

Poznámka

Návěští skoku (label)

Návěští skoku se vždy nachází na začátku bloku. Jestliže existuje také číslo programu, návěští skoku se zapisuje vždy bezprostředně za číslem bloku.

Pro určování názvů návěští skoků platí následující pravidla:

- Počet znaků:
  - minimálně 2
  - maximálně 32
- Povolenými znaky jsou:
  - Písmena
  - Číslice
  - Znak podtržení
- První dva znaky musí být písmena nebo znak podtržení.
- Za názvem návěští skoku následuje dvojtečka (":“).

Syntaxe

1. Opakování jednoho jediného programového řádku:

```
<návěští skoku>: ...
...
REPEATB <návěští skoku> P=<n>
...
```

2. Opakování úseku programu mezi návěštím skoku a příkazem REPEAT:

```
<návěští skoku>: ...
...
REPEAT <návěští skoku> P=<n>
...
```

3. Opakování úseku mezi dvěma návěštími skoku:

```
<počáteční návěští skoku>: ...
...
<koncové návěští skoku>: ...
...
REPEAT <počáteční návěští skoku> <koncové návěští skoku> P=<n>
...
```

Poznámka

Není možné, aby se příkaz REPEAT nacházel mezi počátečním a koncovým návěštím. Jestliže se <počáteční návěští skoku> nachází před příkazem REPEAT a pokud <koncové návěští skoku> není dosaženo před příkazem REPEAT, potom bude se opakován úsek programu mezi <počátečním návěštím skoku> a příkazem REPEAT.

4. Opakování úseku mezi návěštím skoku a návěštím ENDLABEL:

```
<návěští skoku>: ...
...
ENDLABEL: ...
...
REPEAT <návěští skoku> P=<n>
...
```

Poznámka

Není možné, aby se příkaz REPEAT nacházel mezi <návěštím skoku> a návěštím ENDLABEL. Jestliže se <počáteční návěští skoku> nachází před příkazem REPEAT a pokud návěští ENDLABEL není dosaženo před příkazem REPEAT, potom bude se opakován úsek programu mezi <počátečním návěštím skoku> a příkazem REPEAT.

Význam

REPEATB:	Příkaz pro opakování řádku programu.
REPEAT:	Příkaz pro opakování úseku programu.
<návěští skoku>:	Řetězec <návěští skoku> označuje: • Řádek programu, který má být opakován (v případě příkazu REPEATB) nebo • Začátek úseku programu, který má být opakován (v případě příkazu REPEAT) Řádek programu označený jako<návěští skoku> se může nacházet před nebo za příkazem REPEAT / REPEATB. Hledání se napřed provádí směrem k začátku programu. Pokud návěští skoku není v tomto směru nalezeno, spustí se hledání směrem ke konci programu. **Výjimka:** Jestliže má být opakován úsek programu mezi návěštím skoku a příkazem REPEAT (viz bod 2. v odstavci Syntaxe), potom se musí být programový řádek označený jako <návěští skoku> nacházet **před** příkazem REPEAT, protože v tomto případě se vyhledávání uskutečňuje **pouze** ve směru začátku programu. Jestliže řádek s návěštím <návěští skoku> obsahuje další příkazy, budou tyto příkazy znovu prováděny při každém opakování.

ENDLABEL:	Klíčové slovo, které označuje konec úseku programu, který se má opakovat. Jestliže řádek s návěštím ENDLABEL obsahuje další příkazy, budou tyto příkazy znovu prováděny při každém opakování. Příkaz ENDLABEL může být v programu použit i vícekrát.
P:	Adresa pro zadání počtu opakování
<n>:	Počet opakování úseku programu Typ: INT Opakovaný úsek programu bude opakován <n>-krát. Po posledním opakování bude zpracování programu pokračovat řádkem, který následuje za příkazy REPEAT / REPEATB. **Upozornění:** Jestliže příkaz P=<n> není uveden, bude opakovaný úsek programu opakován právě jedenkrát.

Příklady

Příklad 1: Opakování jednoho jediného programového řádku

```
Programový kód                          Komentář
N10 POSITION1: X10 Y20
N20 POSITION2: CYCLE(0,,9,8)            ; Cyklus pro polohování
N30 ...
N40 REPEATB POSITION1 P=5               ; Blok N10 provést pětkrát.
N50 REPEATB POSITION2                   ; Blok N20 provést jednou.
N60 ...
N70 M30
```

Příklad 2: Opakování úseku programu mezi návěštím skoku a příkazem REPEAT

```
Programový kód                  Komentář
N5 R10=15
N10 Begin: R10=R10+1            ; Šířka
N20 Z=10-R10
N30 G1 X=R10 F200
N40 Y=R10
N50 X=-R10
N60 Y=-R10
N70 Z=10+R10
N80 REPEAT BEGIN P=4            ; Úsek programu N10 až N70 provést čtyřikrát.
N90 Z10
N100 M30
```

Příklad 3: Opakování úseku mezi dvěma návěštími skoku

Programový kód	Komentář
N5 R10=15	
N10 Begin: R10=R10+1	; Šířka
N20 Z=10-R10	
N30 G1 X=R10 F200	
N40 Y=R10	
N50 X=-R10	
N60 Y=-R10	
N70 END: Z=10	
N80 Z10	
N90 CYCLE(10,20,30)	
N100 REPEAT BEGIN END P=3	; Úsek programu N10 až N70 provést třikrát.
N110 Z10	
N120 M30	

Příklad 4: Opakování úseku mezi návěštím skoku a návěštím ENDLABEL

Programový kód	Komentář
N10 G1 F300 Z-10	
N20 BEGIN1:	
N30 X10	
N40 Y10	
N50 BEGIN2:	
N60 X20	
N70 Y30	
N80 ENDLABEL: Z10	
N90 X0 Y0 Z0	
N100 Z-10	
N110 BEGIN3: X20	
N120 Y30	
N130 REPEAT BEGIN3 P=3	; Úsek programu N110 až N120 provést třikrát.
N140 REPEAT BEGIN2 P=2	; Úsek programu N50 až N80 provést dvakrát.
N150 M100	
N160 REPEAT BEGIN1 P=2	; Úsek programu N20 až N80 provést dvakrát.
N170 Z10	
N180 X0 Y0	
N190 M30	

Příklad 5: Obrábění frézováním, opracování pozic pro vrtání pomocí různých technologií

Programový kód	**Komentář**
N10 ZENTRIERBOHRER()	; Výměna nástroje pro navrtávání středicích důlků.
N20 POS_1:	; Pozice pro vrtání 1
N30 X1 Y1	
N40 X2	
N50 Y2	
N60 X3 Y3	
N70 ENDLABEL:	
N80 POS_2:	; Pozice pro vrtání 2
N90 X10 Y5	
N100 X9 Y-5	
N110 X3 Y3	
N120 ENDLABEL:	
N130 BOHRER()	; Výměna nástroje za vrták a cyklus vrtání.
N140 GEWINDE(6)	; Výměna závitníku M6 a cyklus pro vrtání závitu.
N150 REPEAT POS_1	; Opakování úseku programu od POS_1 do příkazu ENDLABEL jedenkrát.
N160 BOHRER()	; Výměna nástroje za vrták a cyklus vrtání.
N170 GEWINDE(8)	; Výměna závitníku M8 a cyklus pro vrtání závitu.
N180 REPEAT POS_2	; Opakování úseku programu od POS_2 do příkazu ENDLABEL jedenkrát.
N190 M30	

Další informace

- Opakování části programu může být voláno i vnořeným způsobem. Každé takové volání obsazuje jednu úroveň podprogramů.
- Pokud je v průběhu zpracování opakování části programu naprogramován příkaz `M17` nebo `RET`, opakování části programu se přeruší. Zpracování programu bude pokračovat blokem, který následuje za řádkem s příkazem `REPEAT`.
- V okně aktuálního programu se vypisuje opakování části programu jako vlastní úroveň podprogramu.
- Jestliže během zpracování části programu dojde ke zrušení úrovně, program bude pokračovat na místě za voláním opakování úseku programu.

  Příklad:

Programový kód	**Komentář**
`N5 R10=15`	
`N10 BEGIN: R10=R10+1`	`; Šířka`
`N20 Z=10-R10`	
`N30 G1 X=R10 F200`	
`N40 Y=R10`	`; Zrušení úrovně`
`N50 X=-R10`	
`N60 Y=-R10`	
`N70 END: Z10`	
`N80 Z10`	
`N90 CYCLE(10,20,30)`	
`N100 REPEAT BEGIN END P=3`	
`N120 Z10`	`; Zpracovávání programu bude pokračovat zde.`
`N130 M30`	

- Řídící struktury a opakování částí programu mohou být používány kombinovaně. Neměly by se však mezi nimi vyskytovat žádná překrývající se místa. Opakování úseku programu by mělo být uvnitř větve řídící struktury, příp. řídící struktura by měla ležet uvnitř opakování úseku programu.

- V případě směšování skoků a opakování úseku programu jsou bloky zpracovávány čistě sekvenčně. Jestliže se vyskytuje např. skok z opakování části programu, bude se program tak dlouho zpracovávat, dokud nebude nalezen naprogramovaný konec úseku programu.

  Příklad:

Programový kód

```
N10 G1 F300 Z-10
N20 BEGIN1:
N30 X=10
N40 Y=10
N50 GOTOF BEGIN2
N60 ENDLABEL:
N70 BEGIN2:
N80 X20
N90 Y30
N100 ENDLABEL: Z10
N110 X0 Y0 Z0
N120 Z-10
N130 REPEAT BEGIN1 P=2
N140 Z10
N150 X0 Y0
N160 M30
```

Poznámka

Příkaz `REPEAT` by se měl nacházet za bloky posuvu.

1.12 Řídící struktury

Funkce

Řídící systém standardně zpracovává NC bloky v naprogramované posloupnosti.

Tato posloupnost může být naprogramováním alternativních programových bloků a programových smyček změněna. Programování těchto řídících struktur se uskutečňuje pomocí prvků řídících struktur (klíčových slov) `IF...ELSE`, `LOOP`, `FOR`, `WHILE` a `REPEAT`.

⚠ POZOR

Řídící struktury je možné používat pouze v rámci příkazové části programu. Definice v hlavičce programu nemohou být prováděny ani podmíněně, ani opakovaně.

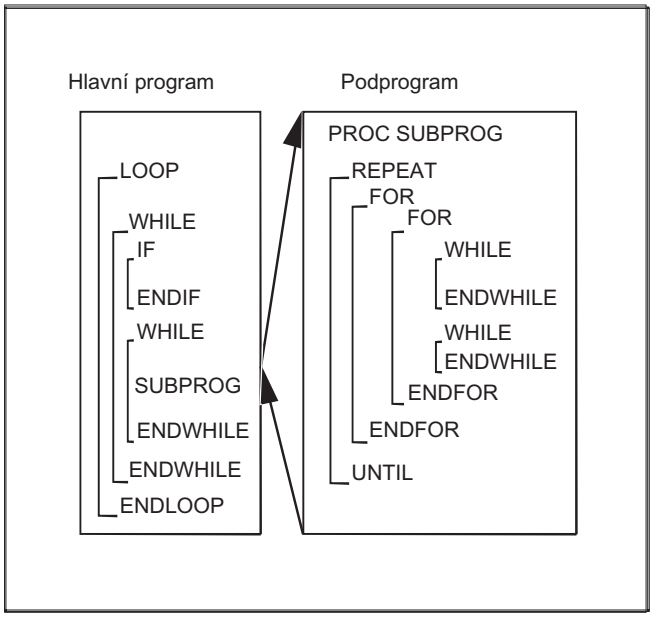
Klíčová slova pro řídící struktury, stejně jako cíle skoku, nesmí být superponovány makry. Kontrola při definici maker se v této záležitosti neprovádí.

Platnost

Řídící struktury mají v programu lokální platnost.

Hloubka vnoření

V rámci každé úrovně podprogramu je možná hloubka vnoření až 16 řídících struktur.



Chování doby zpracování

Při standardně aktivním režimu překladače je možné v důsledku použití programových skoků dosáhnout rychlejšího zpracování než s řídícími strukturami.

U předem přeložených cyklů není mezi programovými skoky a řídícími strukturami žádný rozdíl.

Okrajové podmínky

- Bloky s prvky řídících struktur nemohou být přeskakovány.
- Návěští skoku (label) jsou v blocích s prvky řídících struktur nepřípustné.
- Řídící struktury jsou zpracovávány podle překladače. Když je rozpoznán konec smyčky, bude vyhledán začátek této smyčky a přitom se budou brát v úvahu další nalezené řídící struktury. Z tohoto důvodu není v režimu překladače zkontrolována bloková struktura programu úplně.
- V zásadě se doporučuje nepoužívat řídící struktury a programové skoky současně.
- Při přípravě cyklů může být správné vnoření řídících struktur zkontrolováno.

1.12.1 Programová smyčka s alternativou (IF, ELSE, ENDIF)

Funkce

Konstrukce s příkazy `IF` a `ELSE` se používá tehdy, pokud má programová smyčka obsahovat alternativní programový blok: Jestliže je podmínka `IF` splněna, potom se uskuteční programový blok, který následuje za příkazem `IF`. Jestliže podmínka `IF` splněna**není**, potom se uskuteční alternativní programový blok, který následuje za příkazem `ELSE`.

Poznámka

Jestliže žádná alternativa není zapotřebí, potom může být smyčka IF naprogramována také bez příkazu `ELSE` a bez programového bloku, který za příkazem `ELSE` následuje.

Syntaxe

```
IF <podmínka>
...
ELSE
...
ENDIF
```

Význam

IF:	Uvádí smyčku IF.
ELSE:	Uvádí alternativní programový blok.
ENDIF:	Označuje konec smyčky IF a zajišťuje návrat zpět na začátek smyčky.
<podmínka>:	Podmínka, která rozhoduje o tom, který programový blok se bude zpracovávat.

Příklad

Podprogram pro výměnu nástroje

```
Programový kód                                  Komentář
PROC L6                                         ; Rutina pro výměnu nástroje
N500 DEF INT TNR_AKTUELL                        ; Proměnná pro aktivní T-číslo.
N510 DEF INT TNR_VORWAHL                        ; Proměnná pro předem zvolené T-
                                                  číslo.
                                                ; Zjištění aktuálního nástroje.
N520 STOPRE
N530 IF $P_ISTEST                               ; V režimu testování programu se ...
N540 TNR_AKTUELL = $P_TOOLNO                    ; ... z kontextu programu načte
                                                  "aktuální" nástroj.
N550 ELSE                                       ; V opačném případě se ...
N560 TNR_AKTUELL = $TC_MPP6[9998,1]             ; ... načte nástroj ve vřetenu.
N570 ENDIF
N580 GETSELT(TNR_VORWAHL)                       ; Načtení T-čísla předem zvoleného
                                                  nástroje pro vřeteno.
N590 IF TNR_AKTUELL <> TNR_VORWAHL              ; Pokud předem zvolený nástroj ještě
                                                  není aktuálním nástrojem, potom
                                                  ...
N600 G0 G40 G60 G90 SUPA X450 Y300 Z300 D0      ; ... najíždění na bod pro výměnu
                                                  nástroje ...
N610 M206                                       ; ... a provedení výměny nástroje.
N620 ENDIF
N630 M17
```

1.12.2 Nekonečná programová smyčka (LOOP, ENDLOOP)

Funkce

Nekonečná smyčka nachází uplatnění v nekonečných programech. Na konci smyčky se vždy provádí skok zpět na její začátek.

Syntaxe

```
LOOP
...
ENDLOOP
```

Význam

LOOP:	Uvádí nekonečnou smyčku.
ENDLOOP:	Označuje konec smyčky a způsobuje návrat zpět na její začátek.

Příklad

Programový kód

```
...
LOOP
MSG ("Žádný břit nástroje není aktivní")
M0
STOPRE
ENDLOOP
...
```

1.12.3 Smyčka s počítadlem (FOR ... TO ..., ENDFOR)

Funkce

Smyčka s počítadlem se používá, jestliže má být opakovaně zpracován úsek programu a počet těchto opakování je předem znám.

Syntaxe

```
FOR <proměnná> = <počáteční hodnota> TO <konečná hodnota>
...
ENDFOR
```

Význam

FOR:	Uvádí smyčku s počítadlem.
ENDFOR:	Označuje konec smyčky a způsobuje skok zpět na její začátek, a to tak dlouho, dokud není dosaženo koncové hodnoty počítadla.
<proměnná>:	Počítající proměnná, která načítá nahoru od počáteční až po koncovou hodnotu a s každým průchodem se o hodnotu "1" zvyšuje. Typ INT nebo REAL **Upozornění:** Typ REAL se použije, jestliže je jako počítadlo smyčky naprogramován např. R-parametr. Jestliže je jako počítadlo použita proměnná typu REAL, její hodnota se zaokrouhluje na celé číslo.
<počáteční hodnota>:	Počáteční hodnota pro počítání Podmínka: Počáteční hodnota musí být menší než hodnota konečná.
<konečná hodnota>:	Konečná hodnota pro počítání

Příklady

Příklad 1: Proměnná typu Integer nebo R-parametr jako proměnná počítadla

Proměnná typu INTEGER jako proměnná počítadla:

Programový kód	**Komentář**
DEF INT iVARIABLE1	
R10=R12-R20*R1 R11=6	
FOR iVARIABLE1= R10 TO R11	; Proměnná počítadla = proměnná typu INTEGER
R20=R21*R22+R33	
ENDFOR	
M30	

R-Parametr jako proměnná počítadla:

Programový kód	**Komentář**
R11=6	
FOR R10=R12-R20*R1 TO R11	; Proměnná počítadla = R-parametr (reálná proměnná)
R20=R21*R22+R33	
ENDFOR	
M30	

Příklad 2: Výroba pevně daného počtu kusů

Programový kód	**Komentář**
DEF INT STUECKZAHL	; Definice proměnné typu INT s názvem "STUECKZAHL".
FOR STUECKZAHL = 0 TO 100	; Uvádí smyčku s počítadlem. Proměnná "STUECKZAHL" se bude postupně zvyšovat od počáteční hodnoty "0" až po konečnou hodnotu "100".
G01 …	
ENDFOR	; Konec smyčky s počítadlem.
M30	

1.12.4 Programová smyčka s podmínkou na začátku smyčky (WHILE, ENDWHILE)

Funkce

U smyčky WHILE se podmínka nachází na začátku smyčky. Dokud je podmínka splněna, bude se smyčkou WHILE procházet.

Syntaxe

```
WHILE <podmínka>
...
ENDWHILE
```

Význam

WHILE:	Uvádí programovou smyčku.
ENDWHILE:	Označuje konec smyčky a způsobuje návrat zpět na její začátek.
<podmínka>:	Podmínka, která musí být splněna, aby se smyčkou WHILE procházelo.

Příklad

Programový kód	**Komentář**
...	
WHILE $AA_IW[BOHRACHSE] > -10	; Vyvolání smyčky WHILE za následujících podmínek: Aktuální požadovaná hodnota ve WCS pro osu vrtáku musí být větší než -10.
G1 G91 F250 AX[BOHRACHSE] = -1	
ENDWHILE	
...	

1.12.5 Programová smyčka s podmínkou na konci smyčky (REPEAT, UNTIL)

Funkce

U smyčky REPEAT se podmínka nachází na konci smyčky. Smyčka REPEAT je tedy jednou zpracována a potom je opakována tak dlouho, dokud je podmínka splněna.

Syntaxe

```
REPEAT
...
UNTIL <podmínka>
```

Význam

REPEAT:	Uvádí programovou smyčku.
UNTIL:	Označuje konec smyčky a způsobuje návrat zpět na její začátek.
<podmínka>:	Podmínka, která musí být splněna, aby se smyčkou REPEAT už neprocházelo.

Příklad

Programový kód	**Komentář**
...	
REPEAT	; Vyvolání smyčky REPEAT
...	
UNTIL ...	; Kontrola, zda je podmínka splněna.
...	

1.12.6 Příklad programu se vnořenými řídícími strukturami

Programový kód	**Komentář**
LOOP	
IF NOT $P_SEARCH	; žádné vyhledávání bloku
G01 G90 X0 Z10 F1000	
WHILE $AA_IM[X] <= 100	
G1 G91 X10 F500	; Vrtací vzor
Z-F100	
Z5	
ENDWHILE	
Z10	
ELSE	
MSG("Im Suchlauf wird nicht gebohrt")	
ENDIF	
$A_OUT[1] = 1	; následující vrtaná deska
G4 F2	
ENDLOOP	
M30	

1.13 Koordinace programů (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM)

Funkce

Kanály

Kanál může zpracovávat svůj vlastní program, nezávisle na jiných kanálech. Tímto způsobem mohou být prostřednictvím programu časově řízeny přiřazené osy a vřetena.

Při uvádění do provozu mohou být pro řídící systém nastaveny dva nebo i více kanálů.

Koordinování programů

Jestliže se na opracovávání obrobku podílí více kanálů, potom se může ukázat jako nezbytná synchronizace zpracování programů.

Pro tuto koordinaci programů existují zvláštní příkazy. Všechny vyžadují samostatný blok.

Poznámka

Koordinace programů je možná i ve vlastním kanálu.

Příkazy pro koordinaci programů

- Zadání s absolutním udáním cesty

	Absolutní cesta se přitom skládá podle následujících pravidel:
INIT (n,"/_HUGO_DIR/_N_name_MPF") nebo	- aktuální adresář/_N_name_MPF "aktuální adresář" přitom představuje zvolený adresář obrobku nebo standardní adresář / _N_MPF_DIR.
INIT (n,"/_N_MPF_DIR/_N_name_MPF")	- Aktivování určitého programu pro zpracovávání v určitém kanálu: n: Číslo kanálu, hodnota závisí na konfiguraci - řídícího systému Kompletní název programu
Příklad:	**až do SW 3:**
INIT(2,"/_N_WKS_DIR/_ABRICHT_MPF") G01F0.1 START	Mezi příkazem **init** (bez synchronizace) a operací **NC-Start** se musí nacházet nejméně jeden zpracovatelný blok. U volání podprogramů musí být v údaji cesty doplněno "_SPF".
INIT (2,"/_N_WKS_DIR/ _N_UNTER_1_SPF")	

- Zadání s relativním udáním cesty

Příklad:

INIT(2,"ABRICHT")

INIT(3,"UNTER_1_SPF")

V případě relativního udání cesty platí stejná pravidla jako při volání podprogramů.

U volání podprogramů musí být v názvu programu doplněno "_SPF".

Parametry

Pro výměnu dat mezi programy se mohou používat proměnné, které jsou k dispozici pro všechny kanály společně (specifické globální proměnné NCK). Jinak se sestavování programů uskutečňuje pro každý kanál samostatně.

INIT(n, údaj cesty, režim potvrzení)	Příkaz pro zpracování v kanálu. Volba určitého programu s absolutním nebo relativním udáním cesty.
START (n, n)	Spuštění vybraného programu v jiných kanálech. n,n: Výčet čísel kanálů: Hodnota závisí na konfiguraci řídícího systému.
WAITM (č. značky, n, n, ...)	Nastavování značek "č. značky" ve vlastním kanálu. Předcházející blok se ukončí s přesným najetím. Čekání na značku se stejným "číslem značky" v uvedeném kanálu "n" (nesmí být uveden vlastní kanál). Po synchronizaci se značka vymaže. Současně může být nastaveno max. 10 značek na každý kanál.
WAITMC (č. značky, n, n,	Nastavování značek "č. značky" ve vlastním kanálu. Přesné najetí se uskuteční jen tehdy, pokud v jiném kanálu nebylo značky dosud dosaženo. Čekání na značku se stejným "číslem značky" v uvedeném kanálu "n" (nesmí být uveden vlastní kanál). Jestliže je značky "číslo značky" v uvedeném kanálu dosaženo, zpracování pokračuje, aniž by bylo přesné najetí dokončeno.
WAITE (n, n, ...)	Čekání na konec programu uvedených kanálů (vlastní kanál není uveden). Příklad programování doby prodlevy po příkazu Start. N30 START(2) N31 G4 F0.01 N40 WAITE(2)
SETM (č. značky., č. značky.,	Nastavování značek "č. značky" ve vlastním kanálu, aniž by bylo ovlivněno právě probíhající zpracovávání. Příkaz SETM() zůstává v platnosti, i když je aktivován RESET a NC-START.

`CLEARM (č. značky., č. značky.,`	`Vymazání značek "č. značky" ve vlastním kanálu, aniž by bylo ovlivněno právě probíhající zpracovávání. Příkazem CLEARM() mohou být vymazány všechny značky v kanálu. Příkaz CLEARM (0) vymaže značku "0". Příkaz CLEARM() zůstává v platnosti, i když je aktivován RESET a NC-START.`
`n`	`Číslo nebo název odpovídajícího kanálu`

Poznámka

Všechny výše uvedené příkazy se musí nacházet v samostatných blocích.

Počet značek závisí na instalované CPU.

Čísla kanálu

Pro kanály, které mají být koordinovány, může být uvedeno jako číslo kanálu až 10 kanálů (hodnota typu INT).

Názvy kanálů

Názvy kanálů se musí převádět pomocí proměnných (viz kapitola "Proměnné a početní parametry") na čísla nebo místo čísel kanálů mohou být naprogramovány také názvy kanálů (identifikátory nebo klíčová slova) definované pomocí parametru $MC_CHAN_NAME. Definované názvy musí odpovídat konvencím NC jazyka (tzn. první dva znaky musí být buď písmena nebo znaky podtržení).

⚠ POZOR

Přiřazení čísel musí být chráněno před neuváženými změnami.

Názvy se nesmí krýt s objekty, které v NC systému už mají nějaký jiný význam, jako jsou např. klíčová slova, příkazy jazyka, názvy os atd.

SETM() a CLEARM()

Příkazy `SETM()` a `CLEARM()` mohou být naprogramovány i v rámci synchronních akcí. Viz kapitola "Dosazení/vymazání značky pro čekání: SETM, CLEARM"

Příklad

Kanál s názvem "MASCHINE" má mít číslo kanálu 1,

Kanál s názvem "LADER" má mít číslo kanálu 2:

```
DEF INT MASCHINE=1, LADER=2
```

Proměnné mají stejné názvy jako kanály.

V důsledku toho vypadá například příkaz `START`:

```
START(MASCHINE)
```

Příklad koordinace programů

Kanál 1:

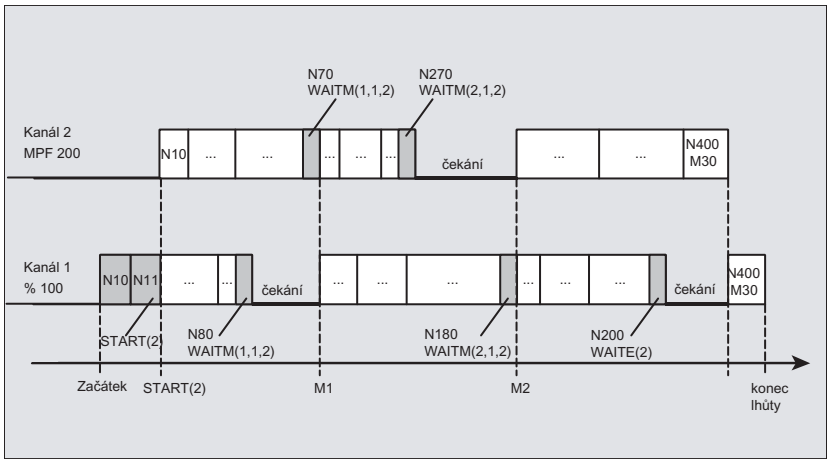
_N_MPF100_MPF

Programový kód	Komentář
N10 INIT(2,"MPF200")	
N11 START(2)	; Zpracování v kanálu 2
...	
N80 WAITM(1,1,2)	; Čekání na značku WAIT 1 v kanálu 1 a v kanálu 2, další zpracování v kanálu 1
...	
N180 WAITM(2,1,2)	; Čekání na značku WAIT 2 v kanálu 1 a v kanálu 2, další zpracování v kanálu 1
...	
N200 WAITE(2)	; Čekání na konec programu v kanálu 2
N201 M30	; Konec programu v kanálu 1, celkový konec
...	

Kanál 2:

_N_MPF200_MPF

Programový kód	Komentář
;$PATH=/_N_MPF_DIR	
	; Zpracování v kanálu 2
N70 WAITM(1,1,2)	; Čekání na značku WAIT 1 v kanálu 1 a v kanálu 2, další zpracování v kanálu 1
...	
N270 WAITM(2,1,2)	; Čekání na značku WAIT 2 v kanálu 1 a v kanálu 2, další zpracování v kanálu 2
...	
N400 M30	; Konec programu v kanálu 2



Příklad: Program z adresáře obrobku

Programový kód
N10 INIT(2,"/_N_WKS_DIR/_N_WELLE1_WPD/_N_ABSPAN1_MPF")

Příklad: Příkaz INIT s relativním udáním cesty

V kanálu 1 je aktivován program /_N_MPF_DIR/_N_MAIN_MPF

Programový kód	Komentář
N10 INIT(2,"MYPROG")	; V kanálu 2 je vybrán program /_N_MPF_DIR/_N_MYPROG_MPF

Příklad: Název kanálu a číslo kanálu s proměnnou typu INT

$MC_CHAN_NAME[0]= "CHAN_X" ;název 1. kanálu
$MC_CHAN_NAME[1]= "CHAN_Y" ;název 2. kanálu

Programový kód	Komentář
START(1, 2)	; Spustit v 1. a ve 2. kanálu

Analogicky k tomu programování s identifikátory kanálů:

Programový kód	Komentář
START(CHAN_X, CHAN_Y)	; Spustit v 1. a ve 2. kanálu ; Identifikátory Kanal_X a Kanal_Y interně reprezentují na základě strojního parametru $MC_CHAN_NAME čísla kanálů 1 a 2. V souladu s tím se uskuteční spuštění s 1. a ve 2. kanálu také.

Programování s proměnnou typu INT:

Programový kód	Komentář
DEF INT chanNo1, chanNo2)	; Definice čísel kanálu
chanNo1=CHAN_X chanNo2=CHAN_Y	
START(chanNo1, chanNo2)	

1.14 Rutiny přerušení (ASUP)

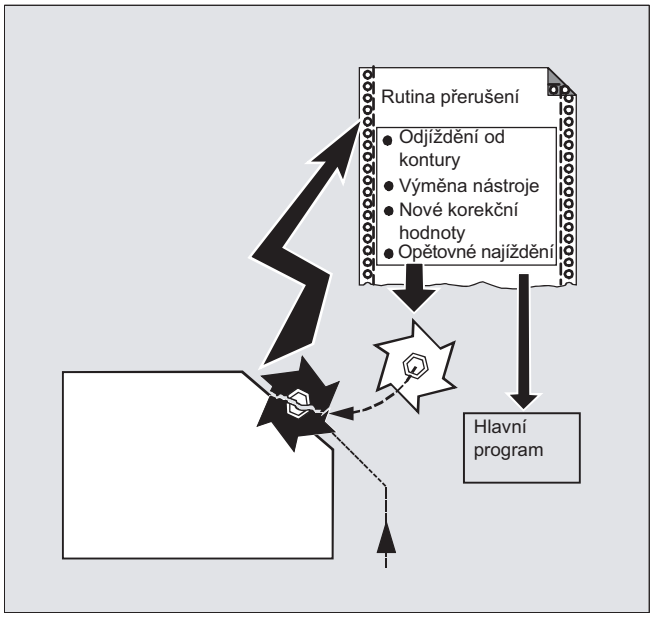
1.14.1 Funkce rutiny přerušení

Poznámka

Pojmy "asynchronní podprogram (ASUP)" a "Rutina přerušení", které se střídavě vyskytují v následujícím popisu, označují stejnou funkci.

Funkce

Funkce rutiny přerušení bude objasněna na základě typického příkladu:



V průběhu zpracování se zlomí nástroj. V důsledku toho se aktivuje signál, který momentálně probíhající obráběcí operaci zastaví a současně spustí podprogram - tak zvanou rutinu přerušení. V tomto podprogramu se nacházejí všechny příkazy, které mají být v tomto případě provedeny.

Jakmile je podprogram zpracován (a v důsledku toho je obnovena připravenost k další práci), skočí řídící systém zpátky do hlavního programu a obrábění bude pokračovat - po aktivování příkazu `REPOS` - na místě, kde došlo k přerušení (viz "Opětovné najíždění na konturu (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [Strana 484]").

⚠ POZOR

Jestliže v podprogramu není žádný příkaz `REPOS` naprogramován, potom se najede do polohy odpovídající koncovému bodu bloku, který odpovídá přerušenému bloku.

Literatura

Příručka Popis funkcí, Základní funkce; BAG, kanál, zpracování programu, chování při resetu (K1), kapitola: "Asynchronní podprogramy (ASUP), rutiny přerušení"

1.14.2 Sestavování rutin přerušení

Sestavení rutiny přerušení jako podprogramu

Při definici je rutina přerušení označena jako podprogram.

Příklad:

Programový kód	**Komentář**
PROC ABHEB_Z	; Název programu "ABHEB_Z"
N10 ...	; Potom následují NC bloky.
...	
N50 M17	; V závěru konec programu a návrat do hlavního programu.

Ukládání modálních G-funkcí (SAVE)

Při definici může být rutina přerušení označena atributem `SAVE`.

Atribut `SAVE` způsobuje, že modální G-funkce, které byly před voláním rutiny přerušení aktivní, se uloží a po skončení rutiny přerušení se znovu aktivují (viz " Ukládání modálních G-funkcí (SAVE) [Strana 170] ").

Díky tomu je možné, aby obrábění po zpracování rutiny přerušení pokračovalo od místa, kde k přerušení došlo.

Příklad:

Programový kód
PROC ABHEB_Z SAVE
N10 ...
...
N50 M17

Další přiřazení rutin přerušení (SETINT)

V rámci rutiny přerušení mohou být naprogramovány příkazy `SETINT` (viz Přiřazování a spouštění rutin přerušení (SETINT, PRIO, BLSYNC) [Strana 122]), takže je možné spouštět další vnořené rutiny přerušení. Spouštění se uskutečňuje až prostřednictvím vstupu.

Literatura

Pokud budete potřebovat další informace týkající se sestavování podprogramů, viz kapitola "Technika podprogramů, technika maker".

1.14.3 Přiřazování a spouštění rutin přerušení (SETINT, PRIO, BLSYNC)

Funkce

Řídící systém má k dispozici signály (vstup 1...8), které umožňují vyvolat přerušení momentálně zpracovávaného programu a spustit odpovídající rutinu přerušení.

Přiřazení, které určuje který vstup spouští který program, se uskutečňuje ve výrobním programu pomocí příkazu `SETINT`.

Jestliže se ve výrobním programu nachází větší počet příkazu `SETINT`, v důsledku čehož se může vyskytnout více signálů současně, musí být nastaveným rutinám přerušení přiřazeny hodnoty priority, které určují posloupnost při jejich zpracovávání. `PRIO=<hodnota>`

Jestliže se v průběhu zpracovávání přerušení vyskytne nový signál, aktuální rutina přerušení je přerušena rutinami s vyšší prioritou.

Syntaxe

```
SETINT(<n>) PRIO=<hodnota> <NAME>
SETINT(<n>) PRIO=<hodnota> <NAME> BLSYNC
SETINT(<n>) PRIO=<hodnota> <NAME> LIFTFAST
```

Význam

`SETINT(<n>)`:	Příkaz: Přiřazení vstupu <n> rutině přerušení. Jakmile se vstup <n> aktivuje, spustí se přiřazená rutina přerušení. **Upozornění:** Jestliže je už obsazenému vstupu přiřazena nová rutina, stává se staré přiřazení automaticky neplatným.
<n>:	Parametr: Číslo vstupu Typ: INT Rozsah hodnot: 1 ... 8
`PRIO=` :	Příkaz: Definice priority
`<hodnota>`:	Hodnota priority Typ: INT Rozsah hodnot: 1 ... 128 Hodnota 1 odpovídá nejvyšší prioritě.
`<NAME>`:	Název podprogramu (rutiny přerušení), který má být zpracován.
`BLSYNC`:	Jestliže je příkaz `SETINT` naprogramován spolu s parametrem `BLSYNC`, potom když se vyskytne signál přerušení, bude momentálně zpracovávaný programový blok ještě dokončen a teprve pak se spustí rutina přerušení.
`LIFTFAST`:	Jestliže je příkaz `SETINT` naprogramován spolu s parametrem `LIFTFAST`, potom když se vyskytne signál přerušení, před spuštěním rutiny přerušení se uskuteční "Rychlé pozvednutí nástroje od kontury (viz "Rychlé pozvednutí od kontury (SETINT LIFTFAST, ALF) [Strana 126]").

Příklady

Příklad 1: Přiřazení rutin přerušení a definice priorit

Programový kód	Komentář
...	
N20 SETINT(3) PRIO=1 ABHEB_Z	; Když se aktivuje vstup 3, pak se má spustit rutina přerušení "ABHEB_Z".
N30 SETINT(2) PRIO=2 ABHEB_X	; Když se aktivuje vstup 2, pak se má spustit rutina přerušení "ABHEB_X".
...	

Jestliže se vstupy aktivují současně, jsou rutiny přerušení zpracovávány postupně po sobě v posloupnosti podle svých hodnot priority: napřed "ABHEB_Z", pak "ABHEB_X".

Příklad 2: Nové přiřazení rutiny přerušení

Programový kód	Komentář
...	
N20 SETINT(3) PRIO=2 ABHEB_Z	; Když se aktivuje vstup 3, pak se má spustit rutina přerušení "ABHEB_Z".
…	
N120 SETINT(3) PRIO=1 ABHEB_X	; Vstupu 3 je přiřazena nová rutina přerušení: Když se aktivuje vstup 3, má se místo "ABHEB_Z" spustit "ABHEB_X".

1.14.4 Deaktivování/opětovné aktivování přiřazení rutiny přerušení (DISABLE, ENABLE)

Funkce

Příkaz `SETINT` může být pomocí příkazu `DISABLE` deaktivován a pomocí příkazu `ENABLE` opět aktivován, aniž by se přiřazení vstup → rutina přerušení ztratilo.

Syntaxe

```
DISABLE(<n>)
ENABLE(<n>)
```

Význam

`DISABLE(<n>)`:	Příkaz: **Deaktivování** přiřazení rutiny přerušení pro vstup **<n>**.
`ENABLE(<n>)`:	Příkaz: **Opětovné aktivování** přiřazení rutiny přerušení pro vstup **<n>**.
<n>:	Parametr: Číslo vstupu
	Typ: INT
	Rozsah hodnot: 1 ... 8

Příklad

```
Programový kód                          Komentář
...
N20 SETINT(3) PRIO=1 ABHEB_Z            ; Když se aktivuje vstup 3, pak se má spustit
                                          rutina přerušení "ABHEB_Z".
...
N90 DISABLE(3)                          ; Příkaz SETINT z bloku N20 se deaktivuje.
...
N130 ENABLE(3)                          ; Příkaz SETINT z bloku N20 se znovu aktivuje
                                          aktivuje.
...
```

1.14.5 Vymazání přiřazení rutiny přerušení (CLRINT)

Funkce

Přiřazení vstup → rutina přerušení definované příkazem `SETINT` může být příkazem `CLRINT` vymazáno.

Syntaxe

```
CLRINT(<n>)
```

Význam

`CLRINT(<n>)`:	Příkaz: Vymazání přiřazení rutiny přerušení pro vstup **<n>**.	
`<n>`:	Parametr: Číslo vstupu	
	Typ:	INT
	Rozsah hodnot:	1 ... 8

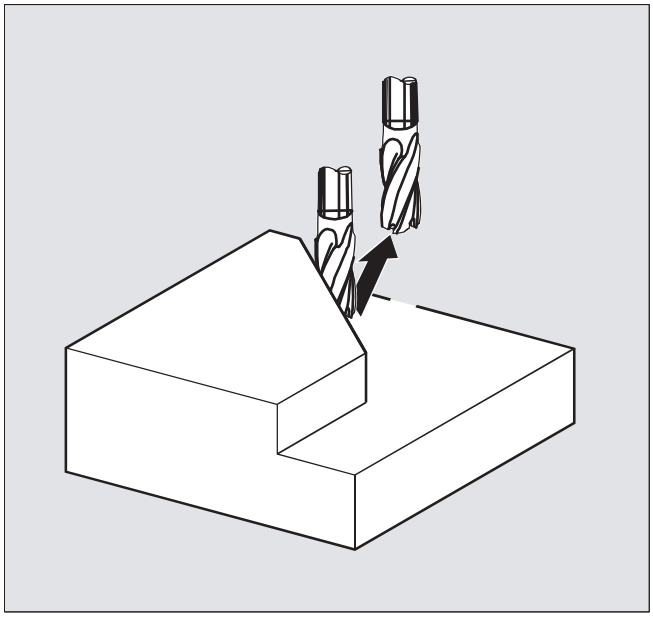
Příklad

```
Programový kód                          Komentář
...
N20 SETINT(3) PRIO=2 ABHEB_Z            ;
...
N50 CLRINT(3)                           ; Přiřazení mezi vstupem "3" a rutinou
                                          přerušení "ABHEB_Z" se vymaže.
...
```

1.14.6 Rychlé pozvednutí od kontury (SETINT LIFTFAST, ALF)

Funkce

Pokud je příkaz `SETINT` naprogramován s parametrem `LIFTFAST`, pak když se aktivuje vstup, nástup se rychlým pozvednutím odsune od kontury obrobku.



Další postup závisí na tom, zda příkaz `SETINT` obsahuje vedle parametru `LIFTFAST` ještě i rutinu přerušení.

S rutinou přerušení:	**Po** rychlém pozvednutí se spustí rutina přerušení.
Bez rutiny přerušení:	Po rychlém pozvednutí se zpracování zastaví a aktivuje se alarm.

Syntaxe

```
SETINT(<n>) PRIO=1 LIFTFAST
SETINT(<n>) PRIO=1 <NAME> LIFTFAST
```

Význam

`SETINT(<n>)`:	Příkaz: Přiřazení vstupu <n> rutině přerušení. Jakmile se vstup <n> aktivuje, spustí se přiřazená rutina přerušení.	
<n>:	Parametr: Číslo vstupu	
	Typ:	INT
	Rozsah hodnot:	1 ... 8
`PRIO=` :	Definice priority	

<hodnota>:	Hodnota priority Rozsah hodnot: 1 ... 128 Hodnota 1 odpovídá nejvyšší prioritě.
<NAME>:	Název podprogramu (rutiny přerušení), který má být zpracován.
LIFTFAST:	Příkaz: Rychlé pozvednutí od kontury
ALF=… :	Příkaz: Programovatelný směr pohybu (nachází se v pohybovém bloku) Pokud budete potřebovat informace o možnostech programování s příkazem ALF, viz kapitola "Směr pohybu při rychlém pozvednutí od kontury [Strana 128]".

Okrajové podmínky

Chování v případě aktivního framu se zrcadlovým převrácením

Při určování směru pozvednutí se kontroluje, zda není aktivní nějaký frame se zrcadlovým převrácením. V takovém případě se u směru pozvedávání vztaženého na směr tečny prohodí směry vlevo a vpravo. Směrové složky v nasměrování nástroje se zrcadlově nepřevracejí. Toto chování se aktivuje nastavením strojního parametru:

MD21202 $MC_LIFTFAST_WITH_MIRROR = TRUE

Příklad

Zlomený nástroj má být automaticky nahrazen náhradním nástrojem. Opracovávání bude potom pokračovat s novým nástrojem.

Hlavní program:

Hlavní program	**Komentář**
N10 SETINT(1) PRIO=1 W_WECHS LIFTFAST	; Když se aktivuje vstup 1, bude nástroj rychlým pozvednutím okamžitě odsunut od kontury (kód č. 7 pro korekci rádiusu nástroje G41). Potom se zpracuje rutina přerušení "W_WECHS".
N20 G0 Z100 G17 T1 ALF=7 D1	
N30 G0 X-5 Y-22 Z2 M3 S300	
N40 Z-7	
N50 G41 G1 X16 Y16 F200	
N60 Y35	
N70 X53 Y65	
N90 X71.5 Y16	
N100 X16	
N110 G40 G0 Z100 M30	

Podprogram:

Podprogram	Komentář
PROC W_WECHS SAVE	; Podprogram s uložením aktuálního provozního stavu
N10 G0 Z100 M5	; Poloha pro výměnu nástroje, zastavení vřetena
N20 T11 M6 D1 G41	; Výměna nástroje
N30 REPOSL RMB M3	; Opětovné najíždění na konturu a skok zpátky do hlavního programu (naprogramuje se v jednom bloku)

1.14.7 Směr pohybu při rychlém pozvednutí od kontury

Zpětný pohyb

Rovina zpětného pohybu je určena prostřednictvím následujících G-kódů:

- LFTXT

  Rovina zpětného pohybu se určuje z tečny ke dráze a ze směru nástroje (standardní nastavení).

- LFWP

  Rovina zpětného pohybu je momentálně aktivní pracovní rovina, která je zvolena jedním z příkazů G17, G18 nebo G19. Směr zpětného pohybu je nezávislý na tečně ke dráze. Tímto způsobem může být naprogramováno rychlé pozvednutí rovnoběžně s osou.

- LFPOS

  Zpětný pohyb osy stanovené příkazem POLFMASK / POLFLIN na absolutní pozici naprogramovanou pomocí příkazu POLF.

  Příkaz ALF nemá žádný vliv na směr pozvednutí pro větší počet os, jakož i pro větší počet os v lineárním vztahu.

  Literatura:
  Příručka programování, Základy; kapitola: "Rychlý zpětný pohyb při řezání závitů"

Programovatelný směr pohybu (ALF=...)

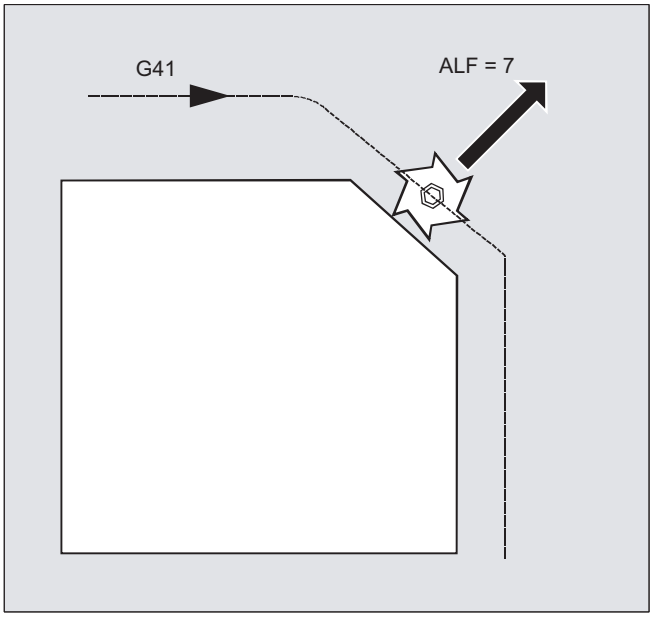
Směr se programuje v diskrétních úhlových krocích 45 stupňů pomocí příkazu ALF v rovině zpětného pohybu.

Možné směry pohybů jsou uloženy v řídícím systému pod speciálními kódovými čísly a pomocí těchto čísel je možné je vyvolat.

Příklad:

Programový kód
N10 SETINT(2) PRIO=1 ABHEB_Z LIFTFAST
ALF=7

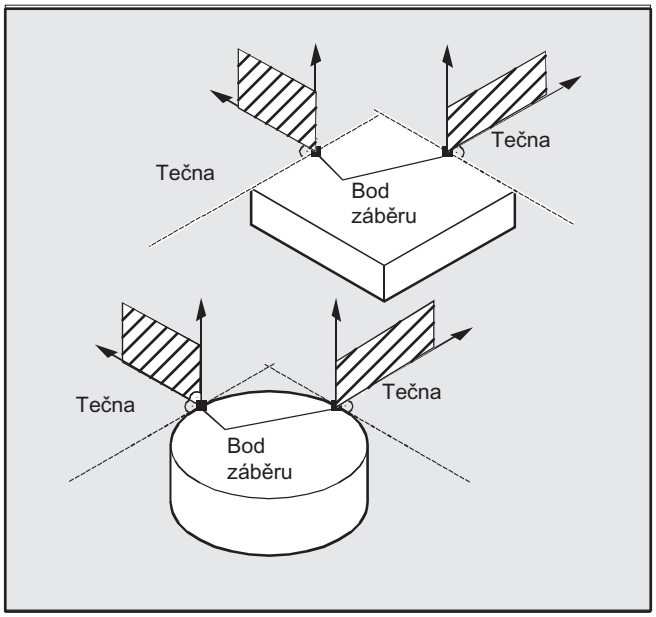
Když je aktivován příkaz `G41` (směr opracování vlevo od kontury), pohybuje se nástroj kolmo od kontury.



Vztažná rovina pro popis směru pohybu u příkazu LFTXT

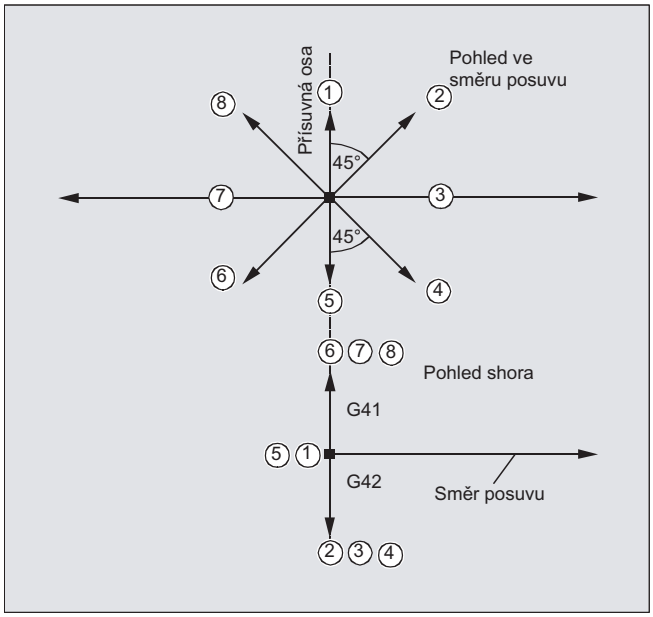
V bodě záběru nástroje na naprogramované kontuře je definována rovina, která slouží jako vztažná rovina pro udávání směru při pozvedávání pomocí odpovídajících kódových čísel.

Vztažná rovina je definována na základě osy přísuvu nástroje (směr přísuvu) a vektoru, který je kolmý jednak na tuto osu a jednak na tečnu v bodě záběru nástroje na kontuře.



Kódová čísla se směry pohybů u příkazu LFTXT

Na následujícím obrázku naleznete kódová čísla se směry pohybů, které se určují na základě vztažné roviny.



Pro `ALF=1` definován návrat ve směru nástroje.

Když je nastaveno `ALF=0`, je funkce "Rychlé pozvednutí" vypnuta.

⚠ POZOR

Když je aktivována korekce rádiusu nástroje:

- v případě příkazu `G41` by se kódy 2, 3, 4
- v případě příkazu `G42` by se kódy 6, 7, 8

neměly používat, protože v těchto případech nástroj přijíždí ke kontuře a dostal by se do kolize s obrobkem.

Kódová čísla se směry pohybů u příkazu LFWP

V případě `LFWP` je směr v pracovní rovině přiřazen podle následujícího uspořádání:

- `G17`: Rovina X/Y

  `ALF=1`: Zpětný pohyb ve směru X

  `ALF=3`: Zpětný pohyb ve směru Y

- `G18`: Rovina Z/X

  `ALF=1`: Zpětný pohyb ve směru Z

  `ALF=3`: Zpětný pohyb ve směru X

- `G19`: Rovina Y/Z

  `ALF=1`: Zpětný pohyb ve směru Y

  `ALF=3`: Zpětný pohyb ve směru Z

1.14.8 Posloupnost pohybů při rutinách přerušení

Rutina přerušení bez příkazu LIFTFAST

Pohyby os na dráze se zabrzdí až do zastavení. Potom se spustí rutina přerušení.

Poloha, kde došlo k zabrždění, se uloží jako místo, kde došlo k přerušení, a při aktivování příkazu `REPOS` s `RMI` na konci rutiny přerušení se na ni najede.

Rutina přerušení s příkazem LIFTFAST

Pohyby os na dráze se zabrzdí. Současně se jako superponovaný pohyb uskuteční pohyb `LIFTFAST`. Jakmile se pohyb po dráze a pohyb `LIFTFAST` úplně zastaví, spustí se rutina přerušení.

Jako místo, kde došlo k přerušení, se uloží pozice na kontuře, na které byl zahájen pohyb `LIFTFAST` a na které byla tedy dráha opuštěna.

Rutina přerušení s parametry `LIFTFAST` a `ALF=0` se chová identicky jako rutina přerušení bez příkazu `LIFTFAST`.

Poznámka

Vzdálenost, o kterou geometrické osy odjedou při rychlém pozvednutí od kontury, může být nastavena pomocí strojního parametru.

1.15 Výměna osy, výměna vřetena (RELEASE, GET, GETD)

Funkce

Interpolace jedné nebo více os, příp. vřeten se smí uskutečňovat vždy jen v jednom kanálu. Pokud musí osa pracovat ve dvou různých kanálech střídavě (např. podavač pro výměnu palet), potom tato osa musí být napřed v aktuálním kanálu uvolněna a pak v jiném kanálu převzata. Kanály si osu vyměňují.

Rozšíření výměny os

Osa/vřeteno mohou být mezi vedlejším a hlavním programem vyměněny se zastavením předběžného zpracování a se synchronizací nebo alternativně i bez zastavení předběžného zpracování. Kromě toho je výměna os možná také pomocí následujících operací:

- Rotace zásobníku os AXCTSWE příp. AXCTWED pomocí implicitních příkazů `GET/GETD`.
- Frame s otočením, jestliže je tato osa spojena vazbou s jinou osou.
- Synchronní akce, viz "Pohybové synchronní akce", "Výměna os `RELEASE`, `GET`".

Výrobce stroje

Věnujte prosím pozornost informacím od výrobce stroje. Prostřednictvím v konfiguraci nastavitelných strojních parametrů musí být osa pro výměnu os ve všech kanálech jednoznačně definována a chování při výměně os může být pomocí strojních parametrů nastaveno tak, aby je bylo možné změnit.

Syntaxe

`RELEASE(název osy, název osy, ...)` nebo `RELEASE(S1)`

`GET(název osy, název osy, ...)` nebo `GET(S2)`

`GETD(název osy, název osy, ...)` nebo `GETD(S3)`

Pomocí příkazu GED (GET Directly) se osa přímo vyzvedne z nějakého jiného kanálu. To znamená, že k tomuto příkazu GETD nemusí být naprogramován žádný odpovídající příkaz REELASE v nějakém jiném kanálu. Znamená to ale také, že nyní musí být s oním kanálem navázána komunikace (např. značky pro čekání).

Význam

`RELEASE (název osy, název osy, ...)`:	Uvolnění jedné nebo více os
`GET (název osy, název osy, ...)`:	Převzetí jedné nebo více os
`GETD (název osy, název osy, ...)`:	Přímé převzetí jedné nebo více os
`název osy`:	Přiřazení osy v systému: AX1, AX2, ... nebo zadání názvů os stroje
`RELEASE(S1)` :	Uvolnění vřetena S1, S2, ...
`GET(S2)` :	Převzetí vřetena S1, S2, ...
`GETD(S3)` :	Přímé převzetí vřetena S1, S2, ...

Požadavek GET bez zastavení předběžného zpracování

Pokud je po požadavku GET **bez** zastavení předběžného zpracování osa opět uvolněna některým z příkazů `RELEASE(osa)` nebo `WAITP(osa)`, má následující příkaz `GET` za následek `GET` **se** zastavením předběžného zpracování.

⚠ POZOR

Osa, příp. vřeteno, které byly příkazem GET převzaty, zůstávají tomuto kanálu přiřazeny i po RESETU, ať už byl vyvolán tlačítkem nebo programem.

Jestliže je osa zapotřebí ve svém základním kanálu, při spuštění nového programu se přiřazení vyměňovaných os, příp. vřeten musí programově-technicky ošetřit.

Při vypnutí a zapnutí systému (Power-On) jsou osy a vřetena přiřazena kanálům podle definice ve strojních parametrech.

Příklady

Příklad 1: Výměna os mezi dvěma kanály

Ze 6 os jsou v kanálu 1 pro obrábění používány: 1., 2., 3. a 4. osa.
5. a 6 osa jsou používány v kanálu 2 pro výměnu obrobku.

Osu 2 má být možno vyměňovat mezi oběma kanály a po zapnutí systému (Power-On) má být tato osa přiřazena kanálu 1.

Program "MAIN" v kanálu 1:

Programový kód	**Komentář**
`INIT (2,"TAUSCH2")`	`; Aktivování programu TAUSCH2 v kanálu 2.`
`N… START (2)`	`; Spuštění programu v kanálu 2.`
`N… GET (AX2)`	`; Převzít osu AX2.`
`...`	
`N… RELEASE (AX2)`	`; Uvolnit osu AX2.`
`N… WAITM (1,1,2)`	`; Čekání na značku WAIT v kanálu 1 a 2 kvůli synchronizaci v obou kanálech.`
`...`	`; Další postup po výměně osy.`
`N… M30`	

Program "TAUSCH2" v kanálu 2:

Programování	**Komentář**
`N… RELEASE (AX2)`	
`N160 WAITM(1,1,2)`	`; Čekání na značku WAIT v kanálu 1 a 2 kvůli synchronizaci v obou kanálech.`
`N150 GET(AX2)`	`; Převzít osu AX2.`
`...`	`; Další postup po výměně osy.`
`N… M30`	

Příklad 2: Výměna osy bez synchronizace

Jestliže osa nemusí být synchronizována, není příkazem GET generováno žádné zastavení předběžného zpracování.

Programování	Komentář
N01 G0 X0	
N02 RELEASE(AX5)	
N03 G64 X10	
N04 X20	
N05 GET(AX5)	; Jestliže není zapotřebí žádná synchronizace, nejedná se o zpracovatelný blok.
N06 G01 F5000	; Žádný zpracovatelný blok.
N07 X20	; Žádný zpracovatelný blok, protože X má stejnou pozici jako v bloku N04.
N08 X30	; První zpracovatelný blok po bloku N05.
...	

Příklad 3: Aktivování výměny os bez zastavení předběžného zpracování

Předpoklad: Výměna os bez zastavení předběžného zpracování musí být nastavena v konfiguraci pomocí strojního parametru.

Programování	Komentář
N010 M4 S100	
N011 G4 F2	
N020 M5	
N021 SPOS=0	
N022 POS[B]=1	
N023 WAITP(B)	; Osa B je nyní neutrální osou.
N030 X1 F10	
N031 X100 F500	
N032 X200	
N040 M3 S500	; Osa nespouští zastavení předběžného zpracování/ příkaz REORG.
N041 G4 F2	
N050 M5	
N099 M30	

Jestliže se vřeteno, příp. osa B bezprostředně po bloku N023 jako **osa PLC** pohybuje na pozici 180 stupňů a zpět na pozici 1 stupeň, potom se tato osa opět stane neutrální osou a v bloku `N40` nebude spouštět žádné zastavení předběžného zpracování.

Předpoklady

Předpoklady pro výměnu osy

- Prostřednictvím strojních parametrů musí být osa definována ve všech kanálech, ve kterých se používá.
- Pomocí strojního parametru pro specifickou **osu** musí být definováno, kterému kanálu má být osa po zapnutí systému (Power-On) přiřazena.

Popis

Uvolnění osy: RELEASE

Při uvolnění osy je potřeba dávat pozor na následující záležitosti:

1. Osa se nesmí podílet na žádné transformaci.
2. V případě spojení os (tangenciální řízení) musí být všechny do svazku seskupené osy uvolněny.
3. Konkurenční polohovací osa nemůže být v tomto stavu vyměňována.
4. V případě řídící osy gantry jsou vyměněny také všechny vlečné osy.
5. V případě vazeb mezi osami (vlečení, spojení řídící hodnotou, elektronická převodovka) může být uvolněna pouze řídící osa vazby.

Převzetí osy: GET

Pomocí tohoto příkazu se uskutečňuje vlastní výměna osy. Veškerá odpovědnost za osu se tím přenáší na kanál, v němž byl příkaz naprogramován.

Důsledky použití příkazu GET:

Výměna osy se synchronizací:

Osa musí být synchronizována vždy, když byla v mezičase přiřazena v nějakém jiném kanálu nebo v PLC a pokud před příkazem GET nebyla uskutečněna žádná synchronizace pomocí příkazu "WAITP", G74 nebo vymazáním zbytkové dráhy.

- Předběžné zpracování se zastaví (stejně jako při příkazu STOPRE).
- Zpracování bude pozastaveno tak dlouho, dokud není výměna úplně dokončena.

Automatický příkaz "GET"

Jestliže je osa v principu k dispozici v nějakém kanálu, v daném okamžiku ale není "kanálovou osou", provede se příkaz "GET" automaticky. Jestliže už jsou osy synchronizovány, k zastavení předběžného zpracování nedochází.

Nastavení měnitelného chování při výměně osy

Okamžik předání osy může být nastaven pomocí strojního parametru, a to následujícím způsobem:

- Automatická výměna osy mezi dvěma kanály se uskuteční také tehdy, pokud byla osa pomocí příkazu WAITP uvedena do neutrálního stavu (stejné chování jak bylo popsáno výše).
- V případě požadavku na rotaci zásobníku os jsou všechny osy zásobníku os, které jsou přiřazeny uvedenému kanálu, převedeny pomocí implicitních příkazů GET, příp. GETD do daného kanálu. Následující výměna osy je znovu umožněna teprve po dokončení rotace zásobníku os.
- Po posunutém pomocném bloku v hlavní větvi programu se zkontroluje, zda je či není nezbytná reorganizace. Jedině v případě, že stavy os v tomto bloku **neodpovídají** momentálním stavům os, je reorganizace potřebná.
- Místo bloku s příkazem GET se zastavením předběžného zpracování a se synchronizací mezi vedlejším a hlavním programem mohou být osy vyměněny bez zastavení předběžného zpracování. Potom je generován jen jeden pomocný blok s příkazem GET. V hlavním programu se při zpracovávání tohoto bloku kontroluje, zda se stavy os v bloku shodují s aktuálními stavy os.

Pokud budete potřebovat další informace o funkcích souvisejících s výměnou os a vřeten, viz /FB2/, Příručka Popis funkcí, Rozšiřovací funkce; BAG, kanály, výměna os (K5).

1.16 Předávání osy jinému kanálu (AXTOCHAN)

Funkce

Pomocí příkazu NC-jazyka `AXTOCHAN` může být pro určitou osu přikázáno, aby se tato osa předala do jiného kanálu. Osa může být do odpovídajícího kanálu přenesena jak z výrobního NC programu, tak i ze synchronní akce.

Syntaxe

```
AXTOCHAN(název osy,číslo kanálu[,název osy,číslo kanálu[,...]])
```

Význam

`AXTOCHAN`:	Vyžádání osy pro určitý kanál
`název osy`:	Přiřazení osy v systému: X, Y, ... nebo zadání názvů os stroje podílejících se na operaci. Jako kanál, který je v příkazu uveden, nesmí být veden vlastní kanál a nesmí se jednat ani o kanál, který v daném okamžiku disponuje interpolačními oprávněními pro danou osu.
`číslo kanálu`:	Číslo kanálu, jemuž má být v dané chvíli osa přiřazena.

Poznámka

Konkurenční polohovací osa a osa ovládaná výlučně prostřednictvím PLC

Osa PLC nesmí být jako konkurenční polohovací osa v kanálu vyměňována. Osa, která je ovládána výlučně prostřednictvím PLC, nesmí být NC programu přiřazena.

Literatura
Příručka Popis funkcí, Rozšiřovací funkce; Polohovací osy (P2)

Příklad

Příkaz AXTOCHAN v NC programu

Osy X a Y jsou známy v kanálu 1 a v kanálu 2. Momentálně disponuje interpolačními oprávněními kanál 1 a v kanálu 1 je spuštěn následující program:

```
Programový kód                  Komentář
N110 AXTOCHAN(Y,2)              ; Osa Y je přesunuta do kanálu 2.
N111 M0
N120 AXTOCHAN(Y,1)              ; Osa Y je přenesena opět zpátky (neutrální stav).
N121 M0
N130 AXTOCHAN(Y,2,X,2)          ; Osa Y a osa X jsou přesunuty do kanálu 2 (osy jsou
                                  neutrální).
N131 M0
N140 AXTOCHAN(Y,2)              ; Osa Y je přesunuta do kanálu 2 (NC program).
N141 M0
```

Další informace

Příkaz AXTOCHAN v NC programu

V tomto případě se příkaz `GET` uskutečňuje a tím pádem se také čeká na skutečnou změnu stavu jen tehdy, vyskytne-li se požadavek na tuto osu pro NC program ve vlastním kanálu. Pokud je osa vyžádána pro jiný kanál nebo jestliže má být ve vlastním kanálu neutrální osou, potom je požadavek odpovídajícím způsobem pouze odložen.

Příkaz AXTOCHAN ze synchronní akce

Jestliže je osa vyžádána pro vlastní kanál, potom se příkaz `AXTOCHAN` ze synchronní akce převádí na příkaz `GET` ze synchronní akce. V tomto případě se osa v rámci prvního vyžádání pro vlastní kanál stává neutrální osou. V případě druhého vyžádání osy z NC programu je osa přiřazena analogicky k příkazu GET v NC programu. Pokud budete potřebovat informace k příkazu GET v synchronních akcích, viz kapitola "Pohybové synchronní akce".

1.17 Aktivování strojních parametrů (NEWCONF)

Funkce

Pomocí příkazu NEWCONF jsou všechny strojní parametry nastaveny do stavu platnosti "NEW _CONFIG". Tato funkce může být aktivována také v uživatelském rozhraní HMI pomocí programového tlačítka "Aktivovat MD".

V rámci zpracovávání funkce "NEWCONF" se provádí implicitní zastavení předběžného zpracování, tzn. pohyb po dráze se přeruší.

Syntaxe

```
NEWCONF
```

Význam

NEWCONF: Příkaz pro aktivování všech strojních parametrů do stavu platnosti "NEW_CONFIG".

Vyvolání příkazu NEWCONF z výrobního programu přes hranice kanálu

Jestliže jsou strojní parametry v důsledku zpracování výrobního programu změněny a potom jsou příkazem NEWCONF aktivovány, nastaví příkaz NEWCONF do aktivního stavu jen ty strojní parametry, které způsobují změny pro kanál výrobního programu.

Poznámka

Aby se všechny změny bezpečně aktivovaly do daného stavu platnosti, musí být příkaz NEWCONF uskutečněn v každém kanálu, ve kterém jsou momentálně vypočítávány také osy nebo funkce, které souvisejí se změněnými strojními parametry.

Příkazem NEWCONF nejsou aktivovány žádné osové strojní parametry.

V případě os ovládaných pomocí PLC musí být spuštěna funkce "reset osy".

Příklad

Obrábění frézováním: Opracování pozic pro vrtání pomocí různých technologií

```
Programový kód                          Komentář
N10 $MA_CONTOUR_TOL[AX]=1.0             ; Změna strojního parametru.
N20 NEWCONF                             ; Prohlášení strojních parametrů za platné.
...
```

1.18 Zápis do souboru (WRITE)

Funkce

Pomocí příkazu `WRITE` je možné zapisovat bloky/data z NC programu na konec zadaného souboru v pasivním systému souborů (protokolový soubor). Může se jednat také o program, který je momentálně zpracováván.

Poznámka

Jestliže soubor, do něhož se má pomocí příkazu `WRITE` zapisovat, ještě v NC systému neexistuje, bude nově založen.

Místem pro ukládání je statická paměť NC systému. V případě systému SINUMERIK 840D sl je to kompaktní Flash-karta. Oproti systému SINUMERIK 840D se v důsledku toho prodlužuje doba potřebná na zpracování příkazu `WRITE` o asi 75 ms.

Pokud na pevném disku existuje soubor stejného názvu, po uzavření souboru (v NC systému) se tento soubor přepíše. Náprava (HMI Advanced): V systémové oblasti "Služby" tento název pomocí programového tlačítka "Vlastnosti" změňte.

Kromě toho je pomocí příkazu `WRITE` možné zapisovat také data z externího NC programu na externí zařízení / do externího souboru (viz také "Výstup do externího zařízení/souboru (EXTOPEN, WRITE, EXTCLOSE) [Strana 708]").

Předpoklady

Momentálně nastavená úroveň ochrany musí být stejná nebo vyšší, než je úroveň oprávnění pro zápis do souboru pomocí příkazu WRITE. Pokud tomu tak není, přístup je odmítnut a vypíše se chybové hlášení (výsledná hodnota v chybové proměnné = 13).

Syntaxe

```
DEF INT <chyba>
...
WRITE(<Chyba>,"<Název souboru>"/"<ExtG>","<Blok/Data>")
```

Význam

`WRITE`: Příkaz pro vložení bloku, příp. dat na konec uvedeného souboru

`<chyba>`: **Parametr 1:** Proměnná pro výslednou chybovou hodnotu

Typ. INT

Hodnota:

0	žádná chyba
1	Cesta není povolena
2	Cesta není nalezena
3	Soubor není nalezen
4	nesprávný datový typ
10	Soubor je plný
11	Soubor je využíván
12	žádné volné kapacity
13	žádná přístupová oprávnění
14	externí zařízení není obsazeno, příp. není otevřeno
15	Chyba při zápisu na externí zařízení
16	naprogramována neplatná externí cesta

<název souboru>: **Parametr 2:** Název souboru v pasivním systému souborů, do kterého se má uvedený blok, příp. uvedená data, vložit.

Typ: STRING

Při zadávání názvu souboru je zapotřebí mít na paměti následující zásady:

- Zadaný název souboru nesmí obsahovat žádné mezery nebo řídící znaky (znaky s kódem ASCII ≤ 32), protože jinak je příkaz WRITE ukončen s chybovým hlášením 1 "Cesta není povolena".
- Název souboru může být zadán pomocí cesty a identifikace souboru:
  - Názvy cesty

    Cesta musí být zadána absolutně, tzn. musí začínat znakem "/".

    Pokud cesta není udána, bude soubor založen v aktuálním adresáři (= adresář zvoleného programu).
  - Přípona souboru

    Pokud název souboru neobsahuje žádnou identifikaci domény ("_N_"), bude odpovídajícím způsobem doplněn.

    Pokud název souboru obsahuje jako čtvrtý znak od konce znak podtržení "_", jsou následující tři znaky interpretovány jako přípona souboru. Aby bylo možné u všech příkazů pro práci se soubory používat stejný název souboru, např. prostřednictvím proměnné typu STRING, smí se používat jedině přípony souborů _SPF a _MPF.

    Pokud žádná přípona "_SPF" nebo "_MPF" není uvedena, bude automaticky doplněna přípona _MPF.
- Název souboru nesmí být delší než 32 bytů, délka řetězce udávajícího cestu smí činit maximálně 128 bytů.

Příklad:

```
"PROTFILE"
"_N_PROTFILE"
"_N_PROTFILE_MPF"
"/_N_MPF_DIR_/_N_PROTFILE_MPF/"
```

<ExtG>: Jestliže mají být data odesílána na externí zařízení/do externího souboru, musí být místo názvu souboru uveden symbolický identifikátor pro externí zařízení/soubor, které se mají otevřít.

Typ: STRING

Pokud budete potřebovat další informace, viz "Výstup do externího zařízení/souboru (EXTOPEN, WRITE, EXTCLOSE) [Strana 708]".

Upozornění:
Tento identifikátor musí být identický s identifikátorem použitým v příkazu EXTOPEN.

<blok/data>: Blok, příp. data, která mají být do uvedeného souboru vložena.

Typ: STRING

Poznámka

Při zápisu do pasivního systému souborů NCK implicitně vkládá příkaz `WRITE` na konec výstupního řetězce znak "LF" (LINE-FEED = konec řádku).

Při odesílání do externího zařízení/souboru toto chování neplatí. Jestliže má být znak "LF" odeslán, musí být explicitně uveden ve výstupním řetězci.

→ K tomu viz příklad 3: Implicitní/explicitní znak "LF"!

Okrajové podmínky

- **Maximální velikost souboru (→ výrobce stroje!)**

  Maximální možná velikost protokolového souboru v pasivním systému souborů se nastavuje pomocí strojního parametru:

  MD11420 $MN_LEN_PROTOCOL_FILE

  Maximální velikost souboru platí pro všechny soubory, které byly založeny příkazem `WRITE` v pasivním systému souborů. V případě překročení této velikosti se aktivuje chybové hlášení a blok, příp. data se do paměti neuloží. Pokud postačuje kapacita paměti, může být založen nový soubor.

Příklady

Příklad 1: Příkaz WRITE do pasivního systému souborů bez absolutního udání cesty

```
Programový kód                                          Komentář
N10 DEF INT ERROR                                       ; Definice chybové
                                                          proměnné.
N20 WRITE(ERROR,"PROT","PROTOKOLL VOM 7.2.97")          ; Zapíše text "PROTOKOLL
                                                          VOM 7.2.97" do souboru
                                                          _N_PROT_MPF.
N30 IF ERROR                                            ; Vyhodnocování chyby.
N40 MSG ("Fehler bei WRITE-Befehl:" << ERROR)
N50 M0
N60 ENDIF
...
```

Příklad 2: Příkaz WRITE do pasivního systému souborů s absolutním udáním cesty

```
Programový kód
...
WRITE(ERROR,"/_N_WKS_DIR/_N_PROT_WPD/_N_PROT_MPF","PROTOKOLL VOM 7.2.97")
...
```

Příklad 3: Implicitní/explicitní znak "LF"

a, Zápis do pasivního systému souborů s implicitně vkládaným znakem "LF"

Programový kód

```
...
N110 DEF INT ERROR
N120 WRITE(ERROR,"/_N_MPF_DIR/_N_MYPROTFILE_MPF","MY_STRING")
N130 WRITE(ERROR,"/_N_MPF_DIR/_N_MYPROTFILE_MPF","MY_STRING")
N140 M30
```

Výsledný výstup:

MY_STRING

MY_STRING

b, Zápis do externího souboru bez implicitně vkládaného znaku "LF"

Programový kód

```
...
N200 DEF STRING[30] DEV_1
N210 DEF INT ERROR
N220 DEV_1="LOCAL_DRIVE/myprotfile.mpf"
N230 EXTOPEN(ERROR,DEV_1)
N240 WRITE(ERROR,DEV_1,"MY_STRING")
N250 WRITE(ERROR,DEV_1,"MY_STRING")
N260 EXTCLOSE(ERROR,DEV_1)
N270 M30
```

Výsledný výstup:

MY_STRINGMY_STRING

c, Zápis do externího souboru s explicitně vkládaným znakem "LF"

Aby bylo dosaženo stejného výsledku jako u případu a, musí být naprogramováno následující:

Programový kód

```
...
N200 DEF STRING[30] DEV_1
N210 DEF INT ERROR
N220 DEV_1="LOCAL_DRIVE/myprotfile.mpf"
N230 EXTOPEN(ERROR,DEV_1)
N240 WRITE(ERROR,DEV_1,"MY_STRING'H0A'")
N250 WRITE(ERROR,DEV_1,"MY_STRING'H0A'")
N260 EXTCLOSE(ERROR,DEV_1)
N270 M30
```

Výsledný výstup:

MY_STRING

MY_STRING

Viz také

Výstup do externího zařízení/souboru (EXTOPEN, WRITE, EXTCLOSE) Výstup do externího zařízení/souboru (EXTOPEN, WRITE, EXTCLOSE) [Strana 708]

1.19 Vymazání souboru (DELETE)

Funkce

Pomocí příkazu `DELETE` je možné vymazat všechny soubory bez ohledu na to, zda tyto soubory vznikly příkazem `WRITE` nebo ne. Pomocí příkazu `DELEE` mohou být mazány také soubory, které byly vytvořeny s vyšší úrovní přístupových práv.

Syntaxe

```
DEF INT <chyba>
DELETE(<chyba>,"<název souboru>")
```

Význam

`DELETE`: Příkaz pro vymazání zadaného souboru.

`<chyba>`: Proměnná pro výslednou chybovou hodnotu

Typ: INT

Hodnota:		
	0	žádná chyba
	1	Cesta není povolena
	2	Cesta není nalezena
	3	Soubor není nalezen
	4	nesprávný datový typ
	11	Soubor je využíván
	12	žádné volné kapacity
	20	jiná chyba

`<název souboru>`: Název souboru, který má být vymazán.

Typ: STRING

Při zadávání názvu souboru je zapotřebí mít na paměti následující zásady:

- Zadaný název souboru nesmí obsahovat žádné mezery nebo řídící znaky (znaky s kódem ASCII ≤ 32), protože jinak je příkaz `DELETE` ukončen s chybovým hlášením 1 "Cesta není povolena".
- Název souboru může být zadán pomocí cesty a identifikace souboru:
  - Názvy cesty

    Cesta musí být zadána absolutně, tzn. musí začínat znakem "/".

    Pokud cesta není udána, bude soubor vyhledán v aktuálním adresáři (= adresář zvoleného programu).
  - Přípona souboru

    Pokud název souboru neobsahuje žádnou identifikaci domény ("_N_"), bude odpovídajícím způsobem doplněn.

    Pokud název souboru obsahuje jako čtvrtý znak od konce znak podtržení "_", jsou následující tři znaky interpretovány jako přípona souboru. Aby bylo možné u všech příkazů pro práci se soubory používat stejný název souboru, např. prostřednictvím proměnné typu STRING, smí se používat jedině přípony souborů _SPF a _MPF.

    Pokud žádná přípona "_SPF" nebo "_MPF" není uvedena, bude automaticky doplněna přípona _MPF.
- Název souboru nesmí být delší než 32 bytů, délka řetězce udávajícího cestu smí činit maximálně 128 bytů.

Příklad:

```
"PROTFILE"
"_N_PROTFILE"
"_N_PROTFILE_MPF"
"/_N_MPF_DIR_/_N_PROTFILE_MPF/"
```

Příklad

Programový kód	**Komentář**
N10 DEF INT ERROR	; Definice chybové proměnné.
N15 STOPRE	; Zastavení předběžného zpracování.
N20 DELETE(ERROR,"/_N_SPF_DIR/_N_TEST1_SPF")	; Vymazání souboru TEST1 v adresáři podprogramu.
N30 IF ERROR	; Vyhodnocování chyby.
N40 MSG("Fehler bei DELETE-Befehl:" <<ERROR)	
N50 M0	
N60 ENDIF	

1.20 Čtení řádků v souboru (READ)

Funkce

Příkaz `READ` načte v zadaném souboru jeden nebo více řádků a načtené informace uloží do pole typu STRING. Každý načtený řádek obsadí v tomto poli jeden jeho prvek.

Poznámka

Soubor se musí nacházet ve statické uživatelské paměti systému NCK (pasivní systém souborů).

Předpoklady

Momentálně nastavená úroveň ochrany musí být stejná nebo vyšší, než je úroveň oprávnění pro čtení souboru pomocí příkazu READ. Pokud tomu tak není, přístup je odmítnut a vypíše se chybové hlášení (výsledná hodnota v chybové proměnné = 13).

Syntaxe

```
DEF INT <chyba>
DEF STRING[<délka řetězce>] <výsledek>[<n>,<m>]
READ(<chyba>,"<název souboru>",<počáteční řádek>,<počet
řádků>,<výsledek>)
```

Význam

`READ`: Příkaz pro čtení řádků ze zadaného souboru a pro uložení těchto řádků do pole proměnných.

`<chyba>`: Proměnná pro výslednou chybovou hodnotu (parametr Call-By-Reference)

Typ: INT

Hodnota:

0	žádná chyba
1	Cesta není povolena
2	Cesta není nalezena
3	Soubor není nalezen
4	nesprávný datový typ
13	Nedostatečná přístupová oprávnění
21	Řádek není k dispozici (parametr `<počáteční řádek>` nebo parametr `<počet řádků>` je větší než počet řádků v zadaném souboru).
22	Délka pole v proměnné pro výsledky (`<výsledek>`) je příliš malá.
23	Oblast řádků je zvolena příliš velká (parametr `<počet řádků>`, takže čtení probíhá až za konec souboru.

`<název souboru>`: Název souboru, z něhož se má číst (parametr Call-By-Value)

Typ: STRING

Při zadávání názvu souboru je zapotřebí mít na paměti následující zásady:

- Zadaný název souboru nesmí obsahovat žádné mezery nebo řídící znaky (znaky s kódem ASCII ≤ 32), protože jinak je příkaz `READ` ukončen s chybovým hlášením 1 "Cesta není povolena".
- Název souboru může být zadán pomocí cesty a identifikace souboru:
  - Názvy cesty

    Cesta musí být zadána absolutně, tzn. musí začínat znakem "/".

    Pokud cesta není udána, bude soubor vyhledán v aktuálním adresáři (= adresář zvoleného programu).
  - Přípona souboru

    Pokud název souboru neobsahuje žádnou identifikaci domény ("_N_"), bude odpovídajícím způsobem doplněn.

    Pokud název souboru obsahuje jako čtvrtý znak od konce znak podtržení "_", jsou následující tři znaky interpretovány jako přípona souboru. Aby bylo možné u všech příkazů pro práci se soubory používat stejný název souboru, např. prostřednictvím proměnné typu STRING, smí se používat jedině přípony souborů _SPF a _MPF.

    Pokud žádná přípona "_SPF" nebo "_MPF" není uvedena, bude automaticky doplněna přípona _MPF.
- Název souboru nesmí být delší než 32 bytů, délka řetězce udávajícího cestu smí činit maximálně 128 bytů.

Příklad:

```
"PROTFILE"
"_N_PROTFILE"
"_N_PROTFILE_MPF"
"/_N_MPF_DIR_/_N_PROTFILE_MPF/"
```

`<počáteční řádek>`: Počáteční řádek oblasti v souboru, z něhož se má číst (parametr Call-By-Value)

Typ: INT

Hodnota:	0	Načte se určitý počet řádků před koncem souboru, přičemž jejich počet je zadán parametrem `<počet řádků>`
	1 … n	Číslo prvního řádku, která se má načíst.

`<počet řádků>`: Počet řádků, které se mají načíst (parametr Call-By-Value)

Typ: INT

`<výsledek>`: Proměnná pro výsledek (parametr "Call-By-Reference")
Pole proměnných, do kterého se ukládá načtený text.
Typ: STRING (max. délka: 255 znaků)
Jestliže parametr `<počet řádků>` specifikuje méně řádků, než kolik odpovídá velikosti pole výsledků `[<n>,<m>]`, potom zůstanou zbývající prvky pole nezměněny.
Konec řádku reprezentovaný řídícími znaky "LF" (Line Feed) nebo "CR LF" (Carriage Return Line Feed) se do pole výsledků **neukládá**.
Jestliže je řádek delší, než je definovaná délka řetězce, jsou načtené řádky oříznuty. Nevypisuje se žádné chybové hlášení.

Poznámka

Binární soubory není možné načítat. V takovém případě je nahlášena chyba "Nesprávný datový typ" (výsledná hodnota v chybové proměnné = 4). Následující datové typy není možné číst: _BIN, _EXE, _OBJ, _LIB, _BOT, _TRC, _ACC, _CYC, _NCK.

Příklad

Programový kód	**Komentář**
`N10 DEF INT ERROR`	`; Definice chybové proměnné.`
`N20 DEF STRING[255] RESULT[5]`	`; Definice proměnné pro výsledky.`
`N30 READ(ERROR,"/_N_CST_DIR/_N_TESTFILE_MPF",1,5,RESULT)`	`; Název souboru s doménou, příponou souboru a udáním cesty.`
`N40 IF ERROR <>0`	`; Vyhodnocování chyby.`
`N50 MSG("FEHLER"<<ERROR<<"BEI READ-BEFEHL")`	
`N60 M0`	
`N70 ENDIF`	
`...`	

1.21 Kontrola existence souboru (ISFILE)

Funkce

Pomocí příkazu `ISFILE` je možné zkontrolovat, zda určitý soubor existuje ve statické uživatelské paměti systému NCK (pasivní systém souborů).

Syntaxe

```
<výsledek>=ISFILE("<název souboru>")
```

Význam

`ISFILE`: Příkaz, kterým se zkontroluje, zda zadaný soubor existuje v pasivním systému souborů.

`<název souboru>`: Název souboru, jehož existence v pasivním systému souborů má být zkontrolována.

Typ: STRING

Při zadávání názvu souboru je zapotřebí mít na paměti následující zásady:

- Uvedený název souboru nesmí obsahovat žádné mezery nebo speciální znaky (znaky s ASCII kódem ≤ 32).
- Název souboru může být zadán pomocí cesty a identifikace souboru:
  - Názvy cesty

    Cesta musí být zadána absolutně, tzn. musí začínat znakem "/".

    Pokud cesta není udána, bude soubor vyhledán v aktuálním adresáři (= adresář zvoleného programu).
  - Přípona souboru

    Pokud název souboru neobsahuje žádnou identifikaci domény ("_N_"), bude odpovídajícím způsobem doplněn.

    Pokud název souboru obsahuje jako čtvrtý znak od konce znak podtržení "_", jsou následující tři znaky interpretovány jako přípona souboru. Aby bylo možné u všech příkazů pro práci se soubory používat stejný název souboru, např. prostřednictvím proměnné typu STRING, smí se používat jedině přípony souborů _SPF a _MPF.

    Pokud žádná přípona "_SPF" nebo "_MPF" není uvedena, bude automaticky doplněna přípona _MPF.
- Název souboru nesmí být delší než 32 bytů, délka řetězce udávajícího cestu smí činit maximálně 128 bytů.

Příklad:

```
"PROTFILE"
"_N_PROTFILE"
"_N_PROTFILE_MPF"
"/_N_MPF_DIR_/_N_PROTFILE_MPF/"
```

<výsledek>:	Proměnná, do které se uloží výsledek kontroly		
	Typ:	BOOL	
	Hodnota:	TRUE	Soubor existuje
		FALSE	Soubor neexistuje

Příklad

Programový kód	**Komentář**
N10 DEF BOOL RESULT	; Definice proměnné pro výsledky.
N20 RESULT=ISFILE("TESTFILE")	
N30 IF(RESULT==FALSE)	
N40 MSG("DATEI NICHT VORHANDEN")	
N50 M0	
N60 ENDIF	
...	

nebo:

Programový kód	**Komentář**
N10 DEF BOOL RESULT	; Definice proměnné pro výsledky.
N20 RESULT=ISFILE("TESTFILE")	
N30 IF(NOT ISFILE("TESTFILE"))	
N40 MSG("DATEI NICHT VORHANDEN")	
N50 M0	
N60 ENDIF	
...	

1.22 Zjišťování informací o souboru (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO)

Funkce

Prostřednictvím příkazů FILEDATE, FILETIME, FILESIZE, FILESTAT a FILEINFO mohou být zjišťovány určité informace o daném souboru, jako jsou datum / přesný čas, kdy došlo k poslednímu přístupu za účelem zápisu, momentální velikost souboru, stav souboru nebo shrnutí všech těchto informací.

Poznámka

Soubor se musí nacházet ve statické uživatelské paměti systému NCK (pasivní systém souborů).

Předpoklady

Momentálně nastavená úroveň ochrany musí být stejná nebo vyšší, než je úroveň oprávnění pro vyvolání výpisu nadřazeného adresáře. Pokud tomu tak není, přístup je odmítnut a vypíše se chybové hlášení (výsledná hodnota v chybové proměnné = 13).

Syntaxe

```
DEF INT <chyba>
DEF STRING[<délka řetězce>] <výsledek>
FILE....(<chyba>,"<název souboru>",<výsledek>)
```

Význam

FILEDATE:	Pomocí příkazu FILEDATE je možné zjistit **datum** posledního přístupu za účelem zápisu do zadaného souboru.
FILETIME:	Pomocí příkazu FILETIME je možné zjistit **přesný čas** posledního přístupu za účelem zápisu do zadaného souboru.
FILESIZE:	Pomocí příkazu FILESIZE se zjišťuje **momentální velikost** zadaného souboru.
FILESTAT:	Příkaz FILESTAT umožňuje zjistit pro uvedený soubor **stavové informace** týkající se oprávnění pro čtení, zápis a spouštění.
FILEINFO:	Příkaz FILEINFO umožňuje pro uvedený soubor vyvolat výpis **shrnutí informací o souboru**, které mohou být zjištěny příkazy FILEDATE, FILETIME, FILESIZE a FILESTAT.

<chyba>: Proměnná pro výslednou chybovou hodnotu (parametr Call-By-Reference)

Typ: INT

Hodnota:		
	0	žádná chyba
	1	Cesta není povolena
	2	Cesta není nalezena
	3	Soubor není nalezen
	4	nesprávný datový typ
	13	Nedostatečná přístupová oprávnění
	22	Délka řetězce v proměnné pro výsledky (<výsledek>) je příliš malá.

<název souboru>: Název souboru, o němž mají být zjištěny požadované informace.

Typ: STRING

Při zadávání názvu souboru je zapotřebí mít na paměti následující zásady:

- Zadaný název souboru nesmí obsahovat žádné mezery nebo řídící znaky (znaky s kódem ASCII ≤ 32), protože jinak je příkaz FILE... ukončen s chybovým hlášením 1 "Cesta není povolena".
- Název souboru může být zadán pomocí cesty a identifikace souboru:
  - Názvy cesty

    Cesta musí být zadána absolutně, tzn. musí začínat znakem "/".

    Pokud cesta není udána, bude soubor vyhledán v aktuálním adresáři (= adresář zvoleného programu).

  - Přípona souboru

    Pokud název souboru neobsahuje žádnou identifikaci domény ("_N_"), bude odpovídajícím způsobem doplněn.

    Pokud název souboru obsahuje jako čtvrtý znak od konce znak podtržení "_", jsou následující tři znaky interpretovány jako přípona souboru. Aby bylo možné u všech příkazů pro práci se soubory používat stejný název souboru, např. prostřednictvím proměnné typu STRING, smí se používat jedině přípony souborů _SPF a _MPF.

    Pokud žádná přípona "_SPF" nebo "_MPF" není uvedena, bude automaticky doplněna přípona _MPF.
- Název souboru nesmí být delší než 32 bytů, délka řetězce udávajícího cestu smí činit maximálně 128 bytů.

Příklad:

```
"PROTFILE"
"_N_PROTFILE"
"_N_PROTFILE_MPF"
"/_N_MPF_DIR_/_N_PROTFILE_MPF/"
```

`<výsledek>`: Proměnná pro výsledek (parametr "Call-By-Reference")
Proměnná, do které se ukládají požadované informace o souboru.

Typ: STRING kde: `FILEDATE`
Formát: "dd.mm.rr"
⇒ řetězec musí být **8** znaků dlouhý.
`FILETIME`
Formát: " hh:mm.ss "
⇒ řetězec musí být **8** znaků dlouhý.
`FILESTAT`
Formát: "rwxsd"
(r: **r**ead, w: **w**rite, x: e**x**ecute, s: **s**how, d: **d**elete)
⇒ řetězec musí být **5** znaků dlouhý.
`FILEINFO`
Formát: "rwxsd nnnnnnnn dd.mm.rr hh:mm:ss"
⇒ řetězec musí být **32** znaků dlouhý.

INT kde: `FILESIZE`
Výpis velikosti souboru v bytech.

Příklad

Programový kód	**Komentář**
`N10 DEF INT ERROR`	`; Definice chybové proměnné.`
`N20 STRING[32] RESULT`	`; Definice proměnné pro výsledky.`
`N30 FILEINFO(ERROR,"/_N_MPF_DIR/_N_TESTFILE_MPF",RESULT)`	`; Název souboru s doménou, příponou souboru a udáním cesty.`
`N40 IF ERROR <>0`	`; Vyhodnocování chyby`
`N50 MSG("FEHLER"<<ERROR<<"BEI FILEINFO-BEFEHL")`	
`N60 M0`	
`N70 ENDIF`	
`...`	

Program v příkladu by mohl do proměnné pro výsledek RESULT uložit například následující řetězec:

"77777 12345678 26.05.00 13:51:30"

1.23 Výpočet kontrolního součtu pole (CHECKSUM)

Funkce

Pomocí příkazu `CHECKSUM` může být vypočítán kontrolní součet pole. Prostřednictvím porovnání tohoto kontrolního součtu s výsledkem předcházejícího výpočtu kontrolního součtu je možné stanovit, jestli se data pole nezměnila.

Aplikace

Kontrola, zda se při oddělování třísky nezměnila vstupní kontura.

Syntaxe

```
DEF INT <chyba>
DEF STRING[<délka řetězce>] <kontrolní součet>
DEF ... <pole>[<n>,<m>,<o>]
<chyba>=CHECKSUM(<kontrolní součet>,"<pole>"[,<počáteční
sloupec>,<koncový sloupec>])
```

Význam

`CHECKSUM`: Příkaz pro výpočet kontrolního součtu pole

`<chyba>`: Proměnná pro výslednou chybovou hodnotu

Typ: INT

Hodnota:

0	žádná chyba
1	Symbol nenalezen
2	žádné pole
3	Index 1 je příliš veliký
4	Index 2 je příliš veliký
5	Neplatný datový typ
10	Přetečení kontrolního součtu

`<kontrolní součet>`: Proměnná pro uložení výsledku výpočtu kontrolního součtu (parametr Call-By-Reference)

Typ: STRING

Potřebná délka řetězce: 16

Kontrolní součet se zobrazuje jako řetězec 16 hexadecimálních číslic. Nejsou však obsaženy žádné formátovací znaky.

Příklad: "A6FC3404E534047C"

<pole>:	Název pole, jehož kontrolní součet má být sestaven (parametr Call-By-Value) Typ: STRING Max. délka řetězce: 32 Přípustná jsou 1- až 3-rozměrná pole následujících typů: BOOL, CHAR, INT, REAL, STRING **Upozornění:** Pole strojních parametrů jsou nepřípustná.
<počáteční sloupec>:	Číslo počátečního sloupce v poli pro výpočet kontrolního součtu (nepovinný parametr)
<koncový sloupec>:	Číslo koncového sloupce v poli pro výpočet kontrolního součtu (nepovinný parametr)

Poznámka

Parametry <počáteční sloupec> a <koncový sloupec> jsou nepovinné. Jestliže žádný index sloupce není udán, bude se kontrolní součet vypočítávat pro celé pole.

Výsledek výpočtu kontrolního součtu je vždy jednoznačný. V případě změny jednoho prvku pole vznikne také jiný výsledný řetězec.

Příklad

Programový kód	Komentář
N10 DEF INT ERROR	; Definice chybové proměnné.
N20 DEF STRING[16] MY_CHECKSUM	; Definice proměnné pro výsledky.
N30 DEF INT MY_VAR[4,4]	; Definice pole.
N40 MY_VAR=...	
N50 ERROR=CHECKSUM(MY_CHECKSUM,"MY_VAR",0,2)	
...	

Program v příkladu by mohl do proměnné pro výsledek MY_CHECKSUM uložit například následující řetězec:

"A6FC3404E534047C"

1.24 Zaokrouhlení (ROUNDUP)

Funkce

Pomocí funkce "ROUNDUP" mohou být vstupní hodnoty typu REAL (desetinná čísla s desetinnou tečkou) zaokrouhlována na nejbližší vyšší celé číslo.

Syntaxe

```
ROUNDUP(<hodnota>)
```

Význam

`ROUNDUP`:	Příkaz pro zaokrouhlení vstupní hodnoty
`<hodnota>`:	Vstupní hodnota typu REAL

Poznámka

Vstupní hodnoty typu INTEGER (celá čísla) se přenesou do výsledku beze změny.

Příklady

Příklad 1: Různé vstupní hodnoty a výsledky jejich zaokrouhlení

Příklad:	Výsledek zaokrouhlení
`ROUNDUP(3.1)`	4.0
`ROUNDUP(3.6)`	4.0
`ROUNDUP(-3.1)`	-3.0
`ROUNDUP(-3.6)`	-3.0
`ROUNDUP(3.0)`	3.0
`ROUNDUP(3)`	3.0

Příklad 2: Funkce ROUNDUP v NC programu

Programový kód

```
N10 X=ROUNDUP(3.5) Y=ROUNDUP(R2+2)
N15 R2=ROUNDUP($AA_IM[Y])
N20 WHEN X=100 DO Y=ROUNDUP($AA_IM[X])
...
```

1.25 Technika podprogramů

1.25.1 Všeobecně

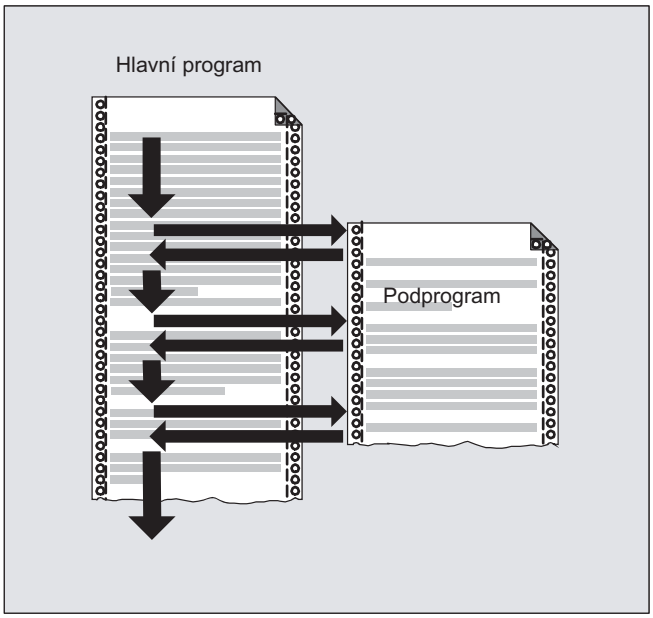
1.25.1.1 Podprogram

Funkce

Označení podprogram pochází ještě z dob, kdy byly výrobní programy pevně rozděleny na hlavní programy a podprogramy. Hlavní programy přitom byly výrobní programy, které se v řídícím systému vybraly ke zpracování a které s potom spouštěly. Podprogramy bývaly výrobní programy, které bývaly vyvolávány z hlavního programu.

Toto pevné rozdělení v dnešním jazyku systému SINUMERIK už neexistuje. V principu může být každý výrobní program vybrán a spuštěn jako program hlavní nebo může být vyvolán z jiného výrobního programu jako podprogram.

Proto jsou v dalším popisu jako podprogramy označovány ty výrobní programy, které jsou vyvolávány z jiného výrobního programu.



Aplikace

Stejně jako u všech vyšších programovacích jazyků se i v jazyku NC systému používají podprogramy k tomu, aby se části programu, které se mají použít vícekrát, uložily do samostatného uzavřeného programu.

Podprogram nabízí následující výhody:

- Výrazně vyšší přehlednost a čitelnost programu
- Zlepšení jakosti díky opětovnému použití testovaných částí programu
- Nabízí možnost vytvořit si specifické knihovny obráběcích programů
- Úspora místa v paměti

1.25.1.2 Názvy podprogramů

Pravidla pro názvy

Při sestavování názvu podprogramu je zapotřebí mít na paměti následující pravidla:

- První dva znaky musí být písmena (A - Z, a - z).
- Následující znaky mohou být libovolnou kombinací písmen, číslic (0 - 9) a znaků podtržení ("_").
- Smí být použito maximálně 31 znaků.

Poznámka

V jazyku NC systému SINUMERIK **nejsou** rozlišována velká a malá písmena.

Přípony názvů programů

Název programu zadaný při jeho sestavování je uvnitř řídícího systému rozšířen o předponu a o příponu.

- Předpona: _N_
- Přípona:
  - Hlavní programy: _MPF
  - Podprogramy: _SPF

Použití názvu programu

Když je název programu použit např. při volání podprogramu, jsou možné všechny kombinace předpony, názvu programu a přípony.

Příklad:

Podprogram s názvem programu "SUB_PROG" může být spuštěn pomocí následujících volání:

```
1. SUB_PROG
2. _N_SUB_PROG
3. SUB_PROG_SPF
4. _N_SUB_PROG_SPF
```

Poznámka

Hlavní program a podprogram stejného názvu

Jestliže existuje hlavní program (_MPF) a podprogram (_SPF) se stejnými názvy, musí být při použití názvu programu ve výrobním programu uvedena příslušná přípona, aby byl program jednoznačně identifikován.

1.25.1.3 Vnoření podprogramů

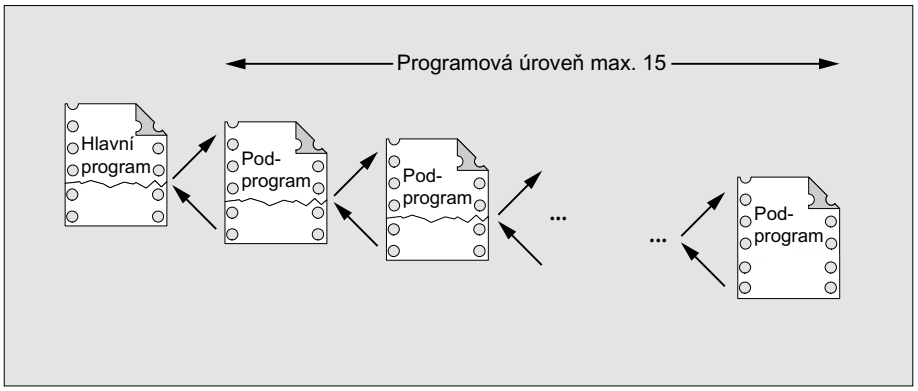
Hlavní program může vyvolávat podprogramy, které mohou opět vyvolávat další podprogramy. Zpracování programů takto může být postupně do sebe vnořeno. Každý program přitom běží na své vlastní programové úrovni.

Hloubka vnoření

Jazyk NC systému má v současnosti k dispozici 16 programových úrovní. Hlavní program vždy běží na nejvyšší programové úrovni 0. Podprogram vždy běží na nejbližší nižší programové úrovni, která následuje za úrovní, z níž byl vyvolán. Programová úroveň 1 je tedy první úrovní pro podprogramy.

Rozdělení programových úrovní:

- Programová úroveň 0: Úroveň hlavního programu
- Programová úroveň 1 - 15: Úroveň podprogramu 1 - 15



Rutiny přerušení (ASUP)

Jestliže je v rámci rutiny přerušení vyvoláván podprogram, nebude zpracováván na programové úrovni, která je momentálně v kanálu aktivní (n), nýbrž rovněž na nejbližší nižší programové úrovni (n+1). By byla tato operace možná i na nejnižší programové úrovni, jsou v souvislosti s rutinami přerušení k dispozici ještě další 2 programové úrovně (16 a 17).

Jestliže jsou zapotřebí více než 2 programové úrovně, je nutné na to explicitně brát zřetel při návrhu struktury výrobního programu, který má být v kanálu zpracováván. To znamená, že se smí počítat jen s maximálně tolika programovými úrovněmi, aby byl pro zpracování přerušení k dispozici dostatečný počet programových úrovní.

Jestliže tedy zpracování přerušení potřebuje např. 4 programové úrovně, musí být výrobní program strukturován tak, aby se obsazovala maximálně 13. programová úroveň. Pokud se potom vyskytne přerušení, budou pro ně potřebné 4 programové úrovně (14 až 17) k dispozici.

Cykly firmy Siemens

Cykly firmy Siemens potřebují 3 programové úrovně. Volání cyklů firmy Siemens se proto musí uskutečňovat maximálně:

- zpracovávání výrobního programu: Programová úroveň 12
- Rutina přerušení: Programová úroveň 14

1.25.1.4 Cesta pro vyhledávání

Při vyvolávání podprogramu bez udání cesty hledá řídící systém v následujících adresářích v uvedené posloupnosti:

Posloupnost	Adresář	Popis
1.	Aktuální adresář	Adresář volajícího programu
2.	/_N_SPF_DIR /	Adresář globálních podprogramů
3.	/_N_CUS_DIR /	Uživatelské cykly
4.	/_N_CMA_DIR /	Cykly výrobce
5.	/_N_CST_DIR /	Standardní cykly

1.25.1.5 Formální a skutečný parametr

O formálních a skutečných parametrech se mluví v souvislosti s definicí a s vyvoláváním podprogramů s předávanými parametry.

Formální parametr

Při definici podprogramu musí být definovány také parametry, které mají být podprogramu předány, což jsou tak zvané formální parametry, spolu s jejich typem a s názvy.

Formální parametry tak definují rozhraní podprogramu.

Příklad:

```
Programový kód                          Komentář
PROC KONTUR (REAL X, REAL Y)            ; Formální parametry: X a Y, oba typu REAL
N20 X1=X Y1=Y                           ; Najíždění osou X1 na pozici X a osou Y1 na
                                          pozici Y
...
N100 RET
```

Skutečné parametry

Podprogramu musí být při jeho volání předány absolutní hodnoty nebo proměnné, což jsou tak zvané skutečné parametry.

Skutečný parametr takto při volání doplní do rozhraní podprogramu skutečné hodnoty.

Příklad:

```
Programový kód                          Komentář
N10 DEF REAL BREITE                     ; Definice proměnných
N20 BREITE=20.0                         ; Přiřazení hodnoty proměnné
N30 KONTUR(5.5, BREITE)                 ; Volání podprogramu se skutečnými parametry:
                                          5.5 a BREITE
...
N100 M30
```

1.25.1.6 Předávání parametrů

Definice podprogramu s předáváním parametrů

Definice podprogramu s předáváním parametrů se uskutečňuje pomocí klíčového slova `PROC` a výpisem úplného seznamu všech podprogramem očekávaných parametrů.

Neúplné předávání parametrů

Při volání podprogramu nemusí být explicitně předávány vždy všechny parametry, které jsou v rozhraní podprogramu definovány. Pokud je nějaký parametr vypuštěn, bude se pro tento parametr předávat standardní hodnota "0".

Aby ale byla posloupnost parametrů jednoznačně identifikována, musí být vždy zapsány čárky, které slouží jako oddělovací znak pro parametry. Výjimku tvoří poslední parametry. Pokud jsou při volání vypuštěny, mohou odpadnout i poslední čárky.

Příklad:

Podprogram:

```
Programový kód                                  Komentář
PROC SUB_PROG (REAL X, REAL Y, REAL Z)          ;  Formální parametry: X, Y a Z
...
N100 RET
```

Hlavní program:

```
Programový kód                      Komentář
PROC MAIN_PROG
...
N30 SUB_PROG(1.0,2.0,3.0)           ; Volání podprogramu s úplným předáváním parametrů:
                                      X=1.0, Y=2.0, Z=3.0
…
N100 M30
```

Příklady pro volání podprogramu v bloku `N30` s neúplným předáváním parametrů:

```
N30 SUB_PROG( ,2.0,3.0)              ; X=0.0, Y=2.0, Z=3.0
N30 SUB_PROG(1.0, ,3.0)              ; X=1.0, Y=0.0, Z=3.0
N30 SUB_PROG(1.0,2.0)                ; X=1.0, Y=2.0, Z=0.0
N30 SUB_PROG( , ,3.0)                ; X=0.0, Y=0.0, Z=3.0
N30 SUB_PROG( , , )                  ; X=0.0, Y=0.0, Z=0.0
```

POZOR

Předávání parametrů typu Call-By-Reference

Parametry, které jsou předávány prostřednictvím Call-by-Reference, nesmí být při volání podprogramu vypuštěny.

POZOR
Datový typ AXIS
Parametry datového typu AXIS nesmí být při volání podprogramu vypuštěny.

Kontrola předávaných parametrů

Prostřednictvím systémové proměnné $P_SUBPAR [n], kde n = 1, 2, ..., může být v podprogramu zkontrolováno, zda se parametr explicitně předává nebo zda byl vypuštěn. Index n se vztahuje na posloupnost formálních parametrů. Index n = 1 se vztahuje na 1. formální parametr, index n = 2 na 2. formální parametr atd.

Následující úsek programu ukazuje na příkladu 1. formálního parametru, jak může být toto přezkoumání realizováno:

Programování	**Komentář**
`PROC SUB_PROG (REAL X, REAL Y, REAL Z)`	`; Formální parametry: X, Y a Z`
`N20 IF $P_SUBPAR[1]==TRUE`	`; Přezkoumání 1. formálního parametru X.`
`...`	`; Tyto operace se uskuteční, jestliže byl formální parametr X explicitně předán.`
`N40 ELSE`	
`...`	`; Tyto operace se uskuteční, jestliže formální parametr X nebyl předán.`
`N60 ENDIF`	
`...`	`; Všeobecné operace`
`N100 RET`	

1.25.2 Definice podprogramu

1.25.2.1 Podprogram bez předávání parametrů

Funkce

Při definici podprogramů bez předávání parametrů může definiční řádek na začátku programu odpadnout.

Syntaxe

```
[PROC <název programu>]
...
```

Význam

PROC:	Definiční příkaz na začátku programu
<název programu>:	Název programu

Příklad

Příklad 1: Podprogram s příkazem PROC

Programový kód	Komentář
PROC SUB_PROG	; Definiční řádek
N10 G01 G90 G64 F1000	
N20 X10 Y20	
...	
N100 RET	; Skok zpátky z podprogramu

Příklad 2: Podprogram bez příkazu PROC

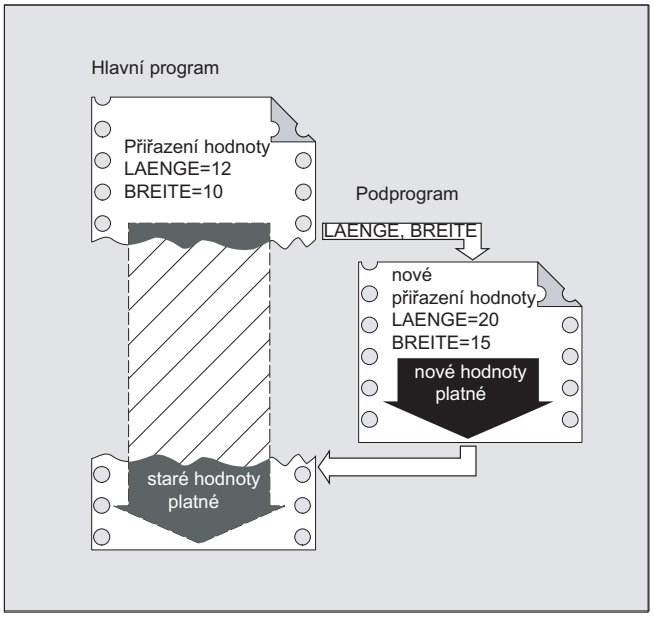
Programový kód	Komentář
N10 G01 G90 G64 F1000	
N20 X10 Y20	
...	
N100 RET	; Skok zpátky z podprogramu

1.25.2.2 Podprogram s předáváním parametrů Call-by-Value (PROC)

Funkce

Definice podprogramu s předáváním parametrů Call-by-Value se uskutečňuje pomocí klíčového slova `PROC`, za kterým následuje název programu a výpisem úplného seznamu všech podprogramem očekávaných parametrů spolu s jejich typy a názvy. Definiční příkaz se musí nacházet na prvním řádku programu.

Předávání parametrů typu Call-by-Value nemá žádný zpětný vliv na volající program. Volající program předává podprogramu pouze hodnoty skutečných parametrů.



Poznámka

Může být předáváno maximálně 127 parametrů.

Syntaxe

```
PROC <název programu> (<typ parametru> <název parametru>, ...)
```

Význam

`PROC`:	Definiční příkaz na začátku programu
`<název programu>`:	Název programu
`<typ parametru>`:	Datový typ parametru (např. REAL, INT, BOOL)
`<název parametru>`:	Název parametru

UPOZORNĚNÍ

Název programu zadaný po klíčovém slově `PROC` se musí shodovat s názvem programu, který je uveden na uživatelském rozhraní.

Příklad

Definice podprogramu se 2 parametry typu REAL.

Programový kód	**Komentář**
`PROC SUB_PROG (REAL LAENGE, REAL BREITE)`	`; Parametr 1: Typ: REAL, název: LAENGE` `Parametr 2: Typ: REAL, název: BREITE`
`...`	
`N100 RET`	`; Skok zpátky z podprogramu`

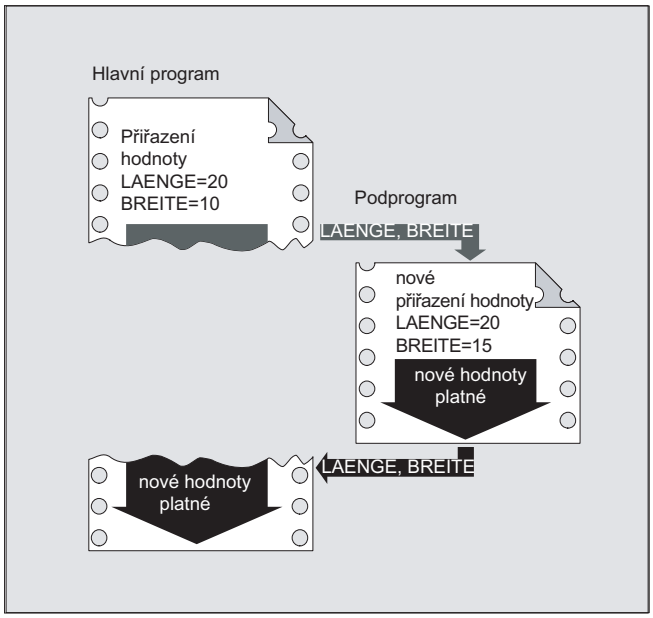
1.25.2.3 Podprogram s předáváním parametrů Call-by-Reference (PROC, VAR)

Funkce

Definice podprogramu s předáváním parametrů Call-by-Reference se uskutečňuje pomocí klíčového slova `PROC`, za kterým následuje název programu a výpisem úplného seznamu všech podprogramem očekávaných parametrů s klíčovým slovem `VAR` a s jejich typy a názvy. Definiční příkaz se musí nacházet na prvním řádku programu.

Při předávání parametrů typu Call-by-Reference mohou být předávány také odkazy na pole.

Předávání parametrů typu Call-by-Reference má zpětný vliv na volající program. Volající program předává podprogramu odkaz na skutečný parametr a umožňuje tak podprogramu přímý přístup k odpovídající proměnné.



Poznámka

Může být předáváno maximálně 127 parametrů.

Poznámka

Předávání parametrů typu Call-by-Reference je zapotřebí jen tehdy, pokud předávaná proměnná byla ve volajícím programu definována (LUD). Proměnné, které jsou globální v kanálu nebo v NC systému, nemusí být předávány, protože podprogram k nim může mít přímý přístup.

Syntaxe

```
PROC <název programu> (VAR <typ parametru> <název parametru>, ...)
PROC <název programu> (VAR <typ pole> <název pole> [<m>,<n>,<o>],
...)
```

Význam

PROC:	Definiční příkaz na začátku programu
VAR:	Klíčové slovo pro předávání parametru typu Call-by-Reference
<název programu>:	Název programu
<typ parametru>:	Datový typ parametru (např. REAL, INT, BOOL)
<název parametru>:	Název parametru
<typ pole>:	Datový typ prvku pole (např. REAL, INT, BOOL)
[<m>,<n>,<o>]:	Velikost pole V současnosti je možné pracovat s maximálně 3-rozměrnými poli: <m>: Velikost pole pro 1. rozměr <n>: Velikost pole pro 2. rozměr <o>: Velikost pole pro 3. rozměr

UPOZORNĚNÍ

Název programu zadaný po klíčovém slově PROC se musí shodovat s názvem programu, který je uveden na uživatelském rozhraní.

Poznámka

Pomocí polí s neurčenou délkou fungujícími jako předávané parametry mohou být podprogramem zpracovávána pole proměnné délky. Za tím účelem se při definici např. dvourozměrného pole jako formální parametr délka v 1. rozměru nezadává. Čárka ale musí být zapsána.

Příklad: `PROC <název programu> (VAR REAL FELD[ ,5])`

Příklad

Definice podprogramu se 2 parametry Call-by-Reference typu REAL:

Programový kód	Komentář
`PROC SUB_PROG(VAR REAL LAENGE, VAR REAL BREITE)`	`; Parametr 1: odkaz na typ: REAL, název: LAENGE` `Parametr 2: odkaz na typ: REAL, název: BREITE`
`...`	
`N100 RET`	

1.25.2.4 Ukládání modálních G-funkcí (SAVE)

Funkce

Atribut `SAVE` způsobuje, že všechny modální G-funkce, které byly před voláním podprogramu aktivní, se uloží a po ukončení podprogramu se znovu aktivují.

POZOR

Přerušení režimu řízení pohybu po dráze

Jestliže je v době aktivního režimu řízení pohybu po dráze vyvolán podprogram s atributem `SAVE`, režim řízení pohybu po dráze se po skončení podprogramu (skok zpět) přeruší.

Syntaxe

```
PROC <název podprogramu> SAVE
```

Význam

`SAVE`: Uložení stavu modálních G-funkcí před voláním podprogramu a jejich opětovné nastavení po skončení podprogramu.

Příklad

V podprogramu KONTUR je aktivní modální G-funkce `G91` (inkrementální rozměry). V hlavním programu je aktivní modální G-funkce `G90` (absolutní rozměry). Když bude v definici podprogramu zadán příznak `SAVE`, po skončení podprogramu bude v hlavním programu znovu v platnosti `G90`.

Definice podprogramu:

Programový kód	Komentář
`PROC KONTUR (REAL WERT1) SAVE`	`; Definice podprogramu s parametrem SAVE`
`N10 G91 ...`	`; Modální G-funkce G91: Řetězové kótování`
`N100 M17`	`; Konec podprogramu`

Hlavní program:

Programový kód	Komentář
N10 G0 X... Y... G90	; Modální G-funkce G90: Absolutní rozměry
N20 ...	
...	
N50 KONTUR (12.4)	; Volání podprogramu
N60 X... Y...	; Modální G-funkce G90 je díky příkazu SAVE znovu aktivována.

Okrajové podmínky

Framy

Chování framů v souvislosti s podprogramy s atributem SAVE závisí na typu framu a může být nastaveno pomocí strojních parametrů.

Literatura

Příručka Popis funkcí, Základní funkce; "Osy, souřadné systémy, framy" (K2),
kapitola: "Návrat zpět z podprogramu s atributem SAVE"

1.25.2.5 Potlačení zpracovávání blok po bloku (SBLOF, SBLON)

Funkce

Potlačení zpracovávání blok po bloku pro celý program

Když je aktivováno zpracování blok po bloku, programy označené atributem SBLOF budou zpracovány kompletně jako jeden blok, tzn. pro celý program se zpracovávání blok po bloku potlačí.

Příkaz SBLOF se nachází na řádku PROCa platí až do konce nebo do přerušení podprogramu. Pomocí příkazu pro návrat se rozhone, zda se má nebo nemá na konci podprogramu zpracování pozastavit.

Návrat s příkazem M17:	Zastavení na konci podprogramu
Návrat s příkazem RET:	Žádné zastavení na konci podprogramu

Potlačení zpracování blok po bloku v rámci programu

Příkaz SBLOF musí být naprogramován v samostatném bloku. Od tohoto bloku je zpracovávání bloku po bloku vypnuto, a to až do:

- následujícího příkazu SBLON

  nebo

- na konci aktivní úrovně podprogramu

Syntaxe

Potlačení zpracovávání blok po bloku pro celý program:
`PROC ... SBLOF`

Potlačení zpracování blok po bloku v rámci programu:

```
SBLOF
...
SBLON
```

Význam

`PROC`:	První příkaz programu
`SBLOF`:	Příkaz pro deaktivování zpracování blok po bloku Příkaz `SBLOF` se může nacházet v bloku s příkazem `PROC` nebo v samostatném bloku.
`SBLON`:	Příkaz pro aktivování zpracování blok po bloku Příkaz `SBLON` se musí nacházet v samostatném bloku.

Okrajové podmínky

- **Potlačení zpracování blok po bloku a vypisování bloků**

  Zobrazování právě zpracovávaného bloku může být v cyklech/podprogramech potlačeno pomocí příkazu `DISPLOF`. Pokud je příkaz `DISPLOF` naprogramován spolu s příkazem `SBLOF`, pak se při zastavení během zpracovávání blok po bloku uvnitř cyklu/podprogramu vypisuje totéž co před voláním cyklu/podprogramu.

- **Potlačení zpracovávání blok po bloku v systémových nebo uživatelských asynchronních podprogramech (ASUP)**

  Jestliže je zastavování při zpracovávání blok po bloku v systémových nebo uživatelských asynchronních podprogramech (ASUP) prostřednictvím nastavení ve strojním parametru MD10702 $MN_IGNORE_SINGLEBLOCK_MASK potlačeno (bit0 = 1 příp. bit1 = 1), potom může být zastavování během zpracovávání blok po bloku opět aktivováno naprogramováním příkazu `SBLON` v programu ASUP.

  Pokud je zastavování při zpracovávání blok po bloku v uživatelských asynchronních podprogramech (ASUP) prostřednictvím nastavení ve strojním parametru MD20117 $MC_IGNORE_SINGLEBLOCK_ASUP potlačeno, potom naprogramováním příkazu `SBLON` v programu ASUP **nemůže** být zastavování během zpracovávání blok po bloku znovu aktivováno.

- **Zvláštnosti potlačení zpracovávání blok po bloku u různých typů zpracování blok po bloku**

  Když je aktivní zpracovávání blok po bloku SBL2 (zastavování po každém bloku výrobního programu), v bloku s příkazem `SBLON`**nedochází** k pozastavení, pokud je v parametru MD10702 $MN_IGNORE_SINGLEBLOCK_MASK (zabránit zastavení při zpracování blok po bloku) nastaven bit 12 na hodnotu "1".

  Když je aktivní zpracovávání blok po bloku SBL3 (zastavení po každém bloku výrobního programu i v cyklech) je příkaz `SBLOF` potlačen.

Příklady

Příklad 1: Potlačení zpracování blok po bloku v rámci programu

```
Programový kód                  Komentář
N10 G1 X100 F1000
N20 SBLOF                       ; Vypnutí zpracování blok po bloku
N30 Y20
N40 M100
N50 R10=90
N60 SBLON                       ; Opětovné zapnutí zpracování blok po bloku
N70 M110
N80 ...
```

Oblast mezi bloky `N20` a `N60` se v režimu zpracování blok po bloku zpracuje jako jeden krok.

Příklad 2: Cyklus se má pro uživatele chovat jako jeden příkaz

Hlavní program:

```
Programový kód
N10 G1 X10 G90 F200
N20 X-4 Y6
N30 CYCLE1
N40 G1 X0
N50 M30
```

Cyklus CYCLE1:

```
Programový kód                              Komentář
N100 PROC CYCLE1 DISPLOF SBLOF              ; Potlačení zpracovávání blok po bloku
N110 R10=3*SIN(R20)+5
N120 IF (R11 <= 0)
N130 SETAL(61000)
N140 ENDIF
N150 G1 G91 Z=R10 F=R11
N160 M17
```

Cyklus CYCLE1 se při aktivním zpracovávání blok po bloku zpracuje najednou, tzn. aby se cyklus CYCLE1 zpracoval, stačí jen jednou stisknout tlačítko Start.

Příklad 3:

ASUP spouštěný z PLC pro aktivování změněného posunutí počátku a korekčních parametrů nástroje se nemá vypisovat.

```
Programový kód
N100 PROC NV SBLOF DISPLOF
N110 CASE $P_UIFRNUM OF        0 GOTOF _G500
                               1 GOTOF _G54
                               2 GOTOF _G55
                               3 GOTOF _G56
                               4 GOTOF _G57
                               DEFAULT GOTOF END
N120 _G54: G54 D=$P_TOOL T=$P_TOOLNO
N130 RET
N140 _G54: G55 D=$P_TOOL T=$P_TOOLNO
N150 RET
N160 _G56: G56 D=$P_TOOL T=$P_TOOLNO
N170 RET
N180 _G57: G57 D=$P_TOOL T=$P_TOOLNO
N190 RET
N200 END: D=$P_TOOL T=$P_TOOLNO
N210 RET
```

Příklad 4: S MD10702 Bit 12 = 1 se nebude zastavovat

Výchozí situace:

- Zpracování blok po bloku je aktivní.
- MD10702 $MN_IGNORE_SINGLEBLOCK_MASK Bit12 = 1

Hlavní program:

Programový kód	Komentář
N10 G0 X0	; Na tomto řádku výrobního programu zastavovat.
N20 X10	; Na tomto řádku výrobního programu zastavovat.
N30 CYCLE	; Cyklem generovaný pohybový blok.
N50 G90 X20	; Na tomto řádku výrobního programu zastavovat.
M30	

Cyklus CYCLE:

Programový kód	Komentář
PROC CYCLE SBLOF	; Potlačení zastavování při zpracovávání blok po bloku
N100 R0 = 1	
N110 SBLON	; Kvůli nastavení MD10702 Bit12=1 se na tomto řádku výrobního programu nebude zastavovat.
N120 X1	; Na tomto řádku výrobního programu se bude zastavovat.

```
Programový kód            Komentář
N140 SBLOF
N150 R0 = 2
RET
```

Příklad 5: Potlačení zpracování blok po bloku v případě vnořených podprogramů

Výchozí situace:

Zpracování blok po bloku je aktivní.

Vnoření podprogramů:

```
Programový kód                                  Komentář
N10 X0 F1000                                    ; Na tomto bloku se bude zastavovat.
N20 UP1(0)
      PROC UP1(INT _NR) SBLOF                   ; Potlačení zastavování při zpracovávání
                                                  blok po bloku.
      N100 X10
      N110 UP2(0)
            PROC UP2(INT _NR)
            N200 X20
            N210 SBLON                          ; Zapnutí zastavování při zpracovávání
                                                  blok po bloku.
            N220 X22                            ; Na tomto bloku se bude zastavovat.
            N230 UP3(0)
                  PROC UP3(INT _NR)
                  N300 SBLOF                    ; Potlačení zastavování při zpracovávání
                                                  blok po bloku.
                  N305 X30
                  N310 SBLON                    ; Zapnutí zastavování při zpracovávání
                                                  blok po bloku.
                  N320 X32                      ; Na tomto bloku se bude zastavovat.
                  N330 SBLOF                    ; Potlačení zastavování při zpracovávání
                                                  blok po bloku.
                  N340 X34
                  N350 M17                      ; Příkaz SBLOF je aktivní.
            N240 X24                            ; Na tomto bloku se bude zastavovat.
                                                  Příkaz SBLON je aktivní.
            N250 M17                            ; Na tomto bloku se bude zastavovat.
                                                  Příkaz SBLON je aktivní.
      N120 X12
      N130 M17                                  ; Na tomto návratovém bloku se
                                                  bude zastavovat. Je aktivní příkaz
                                                  SBLOF z příkazu PROC.
N30 X0                                          ; Na tomto bloku se bude zastavovat.
N40 M30                                         ; Na tomto bloku se bude zastavovat.
```

Další informace

Zablokování zpracování blok po bloku pro asynchronní podprogramy

Aby se program ASUP při zpracování blok po bloku zpracoval v jednom kroku, musí být v programu ASUP naprogramován příkaz `PROC` s příkazem `SBLOF`. To platí také pro funkci "Editovatelný systémový ASUP" (MD11610 $MN_ASUP_EDITABLE).

Příklad pro editovatelný systémový ASUP:

Programový kód	**Komentář**
`N10 PROC ASUP1 SBLOF DISPLOF`	
`N20 IF $AC_ASUP=='H200'`	
`N30 RET`	`; Žádný příkaz REPOS při změně provozního režimu.`
`N40 ELSE`	
`N50 REPOSA`	`; REPOS ve všech ostatních případech.`
`N60 ENDIF`	

Ovlivňování zpracování programu při zpracování blok po bloku

Při zpracovávání blok po bloku může uživatel zpracovat výrobní program jeden blok po druhém. Existují následující druhy nastavení:

- SBL1: Zpracování IPO blok po bloku se zastavením po každém bloku s funkcemi stroje.
- SBL2: Zpracování blok po bloku se zastavením po každém bloku.
- SBL3: Zastavení v cyklu (aktivováním SBL3 se příkaz `SBLOF` potlačí).

Potlačení zpracování blok po bloku v případě vnořených podprogramů

Pokud bylo v podprogramu v příkazu `PROC` naprogramováno `SBLOF`, při návratu z podprogramu pomocí příkazu `M17` bude zpracování pozastaveno. Tím se zabrání, aby se ve volajícím programu ihned zpracoval následující blok. Jestliže je v podprogramu, který nemá příkaz `SBLOF` v příkazu `PROC`, příkazem `SBLOF` aktivováno potlačení zpracování blok po bloku, bude zpracování pozastaveno teprve až po následujícím bloku s funkcemi stroje ve volajícím programu. Je-li toto chování nežádoucí, musí být v podprogramu ještě před návratem (`M17`) znovu naprogramován příkaz `SBLON`. Při návratu pomocí příkazu `RET` se zpracování v nadřazeném programu nezastaví.

1.25.2.6 Potlačení vypisování aktuálního bloku (DISPLOF, DISPLON, ACTBLOCNO)

Funkce

V příslušném okně se standardně vypisuje aktuální programový blok. V cyklech, příp. v podprogramech, může být vypisování aktuálního bloku pomocí příkazu `DISPLOF` potlačeno. Namísto aktuálního bloku se potom vypisuje volání cyklu, příp. podprogramu. Pomocí příkazu `DISPLON` může být potlačení vypisování bloku opět odstraněno.

Příkazy `DISPLOF`, příp. `DISPLON` se programují na řádek programu s příkazem `PROC`, takže platí pro celý podprogram a implicitně také pro všechny podprogramy volané tímto podprogramem, které neobsahují žádný z příkazů `DISPLON`, příp. `DISPLOF`. Toto chování platí také pro programy ASUP.

Syntaxe

```
PROC … DISPLOF
PROC … DISPLOF ACTBLOCNO
PROC … DISPLON
```

Význam

`DISPLOF`: Příkaz pro potlačování vypisování aktuálního bloku.

Umístění: Na konci programového řádku s příkazem `PROC`.

Platnost: Až do návratu z podprogramu nebo do konce programu.

Upozornění:
Jestliže jsou z podprogramu s příkazem `DISPLOF` vyvolávány další podprogramy, potom je vypisování aktuálního bloku potlačeno také v těchto podprogramech, pokud v nich není explicitně naprogramován příkaz `DISPLON`.

`DISPLON`: Příkaz pro odstranění potlačování vypisování aktuálního bloku

Umístění: Na konci programového řádku s příkazem `PROC`.

Platnost: Až do návratu z podprogramu nebo do konce programu.

Upozornění:
Jestliže jsou z podprogramu s příkazem `DISPLON` vyvolávány další podprogramy, potom se aktuální programový blok vypisuje také v těchto podprogramech, pokud v nich není explicitně naprogramován příkaz `DISPLOF`.

`ACTBLOCNO`: Příkaz `DISPLOF` ve spojení s atributem `ACTBLOCNO` způsobuje, že v případě alarmu se bude vypisovat číslo aktuálního bloku, ve kterém se alarm vyskytl. To platí i tehdy, jestliže je na některé z nižších programových úrovní naprogramován příkaz `DISPLOF`.

V případě příkazu `DISPLOF` bez atributu `ACTBLOCNO` se oproti tomu vypisuje číslo bloku s voláním cyklu, příp. podprogramu z poslední programové úrovně, která není opatřena příkazem `DISPLOF`.

Příklady

Příklad 1: Potlačení vypisování aktuálního bloku v cyklu

Programový kód	Komentář
PROC CYCLE(AXIS TOMOV, REAL POSITION) SAVE DISPLOF	; Potlačení vypisování aktuálního bloku. Místo toho se má vypisovat volání cyklu, např.: CYCLE(X,100.0)
DEF REAL DIFF	; Obsah cyklu
G01 ...	
...	
RET	; Návrat zpět z podprogramu. V okně pro vypisování bloku se vypisuje blok, který následuje za voláním cyklu.

Příklad 2: Vypisování bloku v případě aktivování alarmu

Podprogram SUBPROG1 (s atributem ACTBLOCNO):

Programový kód	Komentář
PROC SUBPROG1 DISPLOF ACTBLOCNO	
N8000 R10 = R33 + R44	
...	
N9040 R10 = 66 X100	; Aktivování alarmu 12080
...	
N10000 M17	

Podprogram SUBPROG2 (bez atributu ACTBLOCNO):

Programový kód	Komentář
PROC SUBPROG2 DISPLOF	
N5000 R10 = R33 + R44	
...	
N6040 R10 = 66 X100	; Aktivování alarmu 12080
...	
N7000 M17	

Hlavní program:

Programový kód	Komentář
N1000 G0 X0 Y0 Z0	
N1010 ...	
...	
N2050 SUBPROG1	; Hlášení alarmu = "12080 kanál K1 blok N9040 Syntaktická chyba u textu R10="
N2060 ...	
N2350 SUBPROG2	; Hlášení alarmu = "12080 kanál K1 blok N2350 Syntaktická chyba u textu R10="
...	
N3000 M30	

Příklad 3: Odstranění potlačení vypisování aktuálního bloku

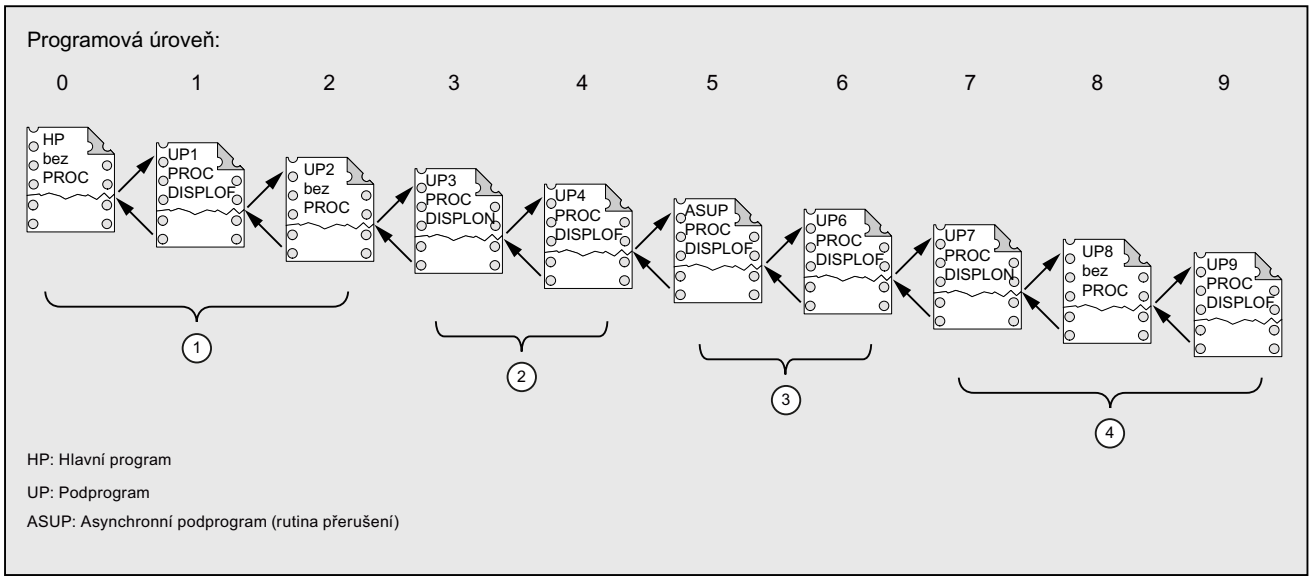
Podprogram SUB1 s potlačením vypisování:

Programový kód	Komentář
`PROC SUB1 DISPLOF`	`; Potlačení vypisování aktuálního bloku v podprogramu SUB1. Místo toho se má vypisovat blok s voláním podprogramu SUB1.`
`...`	
`N300 SUB2`	`; Volání podprogramu SUB2.`
`...`	
`N500 M17`	

Podprogram SUB2 bez potlačení vypisování:

Programový kód	Komentář
`PROC SUB2 DISPLON`	`; Odstranění potlačení vypisování aktuálního bloku v podprogramu SUB2.`
`...`	
`N200 M17`	`; Návrat zpět do podprogramu SUB1. V podprogramu SUB1 je vypisování aktuálního bloku opět potlačeno.`

Příklad 4: Chování vypisování při různých kombinacích příkazů DISPLON/DISPLOF



① V okně výpisu aktuálního bloku se vypisují řádky výrobního programu z programové úrovně 0.

② V okně výpisu aktuálního bloku se vypisují řádky výrobního programu z programové úrovně 3.

③ V okně výpisu aktuálního bloku se vypisují řádky výrobního programu z programové úrovně 3.

④ V okně výpisu aktuálního bloku se vypisují řádky výrobního programu z programové úrovně 7/8.

1.25.2.7 Označení podprogramů s přípravou (PREPRO)

Funkce

Klíčovým slovem `PREPRO` na konci řádku s příkazem `PROC` mohou být označeny všechny soubory, které mají být při náběhu systému předběžně zpracovány.

Poznámka

Tento druh přípravy programu je závislý na odpovídajícím způsobem nastaveném strojním parametru. Věnujte prosím pozornost informacím od výrobce stroje.

Literatura:
Příručka Popis funkcí, Speciální funkce; Přípravné zpracování (V2)

Syntaxe

```
PROC … PREPRO
```

Význam

`PREPRO`: Klíčové slovo pro označení všech souborů, které mají být v rámci náběhu systému předběžně zpracovány a které patří k NC programům uloženým v adresáři cyklů.

Načítání a vyvolávání podprogramů s přípravou

Jak podprogramy s parametry připravené při náběhu systému, tak také při volání podprogramu, se adresáři cyklů prochází ve stejné posloupnosti:

1. _N_CUS_DIR - cykly uživatele
2. _N_CMA_DIR - cykly výrobce
3. _N_CST_DIR - standardní cykly

V případě NC programů stejného názvu a s odlišnými významy se aktivuje první nalezený příkaz `PROC` a ostatní příkazy `PROC` jsou přeskakovány, aniž by bylo generováno alarmové hlášení.

1.25.2.8 Návrat zpět z podprogramu - M17

Funkce

N konci podprogramu se nachází příkaz pro návrat `M17` (příp. příkaz konce výrobního programu `M30`). Způsobuje návrat zpátky do volajícího výrobního programu na blok, který následuje za voláním podprogramu.

Poznámka

S příkazy `M17` a `M30` mají v NC jazyce zcela stejné postavení.

Syntaxe

```
PROC <název programu>
...
M17/M30
```

Okrajové podmínky

Vliv návratu z podprogramu na režim řízení pohybu po dráze

Jestliže se příkaz `M17` (příp. `M30`) nachází v bloku výrobního programu samotný, aktivní režim řízení pohybu po dráze v kanálu se tím přeruší.

Aby se přerušení režimu řízení pohybu po dráze zabránilo, je zapotřebí příkaz `M17` (příp. `M30`) napsat v posledním bloku posuvu. Kromě toho musí být nastaven na "0" následující strojní parametr:

MD20800 $MC_SPF_END_TO_VDI = 0 (žádný výstup příkazu M30/M17 na rozhraní NC/PLC)

Příklad

1. Podprogram s příkazem `M17` v samostatném bloku

Programový kód	**Komentář**
`N10 G64 F2000 G91 X10 Y10`	
`N20 X10 Z10`	
`N30 M17`	`; Návrat s přerušením režimu řízení pohybu po dráze.`

2. Podprogram s příkazem `M17` v posledním bloku posuvu

Programový kód	**Komentář**
`N10 G64 F2000 G91 X10 Y10`	
`N20 X10 Z10 M17`	`; Návrat bez přerušení režimu řízení pohybu po dráze.`

1.25.2.9 Návrat z podprogramu pomocí příkazu RET

Funkce

Jako náhrada příkazu pro návrat z podprogramu `M17` je možné v podprogramu používat také příkaz `RET`. Příkaz `RET` musí být naprogramován v samostatném bloku výrobního programu. Stejně jako příkaz `M17` i příkaz `RET` způsobuje návrat zpátky do volajícího výrobního programu na blok, který následuje za voláním podprogramu.

Poznámka

Naprogramováním parametrů může být chování při návratu pomocí příkazu `RET` změněno (viz "Návrat z podprogramu s nastavitelnými parametry (RET ...) [Strana 183]").

Aplikace

Příkaz `RET` je potřeba používat tehdy, pokud režim řízení pohybu po dráze (G64) (`G641` ... `G645`) nemá být návratem z podprogramu přerušen.

Předpoklady

Příkaz `RET` se může používat pouze v podprogramech, které nejsou definovány s atributem `SAVE`.

Syntaxe

```
PROC <název programu>
...
RET
```

Příklad

Hlavní program:

Programový kód	**Komentář**
PROC MAIN_PROGRAM	; Začátek programu.
...	
N50 SUB_PROG	; Volání podprogramu: SUB_PROG
N60 ...	
...	
N100 M30	; Konec programu

Podprogram:

Programový kód	Komentář
PROC SUB_PROG	
...	
N100 RET	; Návrat se uskuteční na blok N60 v hlavním programu.

1.25.2.10 Návrat z podprogramu s nastavitelnými parametry (RET ...)

Funkce

Obecně platí, že z podprogramu se po jeho skončení pomocí příkazů `RET` nebo `M17` skáče zpátky do programu, ze kterého byl podprogram vyvolán, a zpracování potom pokračuje na programovém řádku, který následuje za voláním podprogramu.

Vedle toho však existují případy použití, při kterých má zpracování programu pokračovat na jiném místě, např.:

- Pokračování zpracováním programu po vyvolání cyklu pro oddělování třísky v režimu dialektu ISO (po popisu kontury).
- Návrat zpět do hlavního programu z libovolné úrovně podprogramu (i po ASUP) při zpracování chybových stavů.
- Návrat zpět přes více programových úrovní pro speciální aplikace v cyklech překladače a v režimu dialektu ISO.

V takových případech se příkaz `RET` programuje spolu s "návratovými parametry".

Syntaxe

```
RET("<cílový blok>")
RET("<cílový blok>",<blok za cílovým blokem>)
RET("<cílový blok>",<blok za cílovým blokem>,<počet úrovní při
návratu>)
RET("<cílový blok>", ,<počet úrovní při návratu>)
RET("<cílový blok>",<blok za cílovým blokem>,<počet úrovní při
návratu>,
<návrat na začátek programu>)
RET( , ,<počet úrovní při návratu>,<návrat na začátek programu>)
```

Význam

`RET`: Konec podprogramu (používá se místo `M17`)

`<cílový blok>`: Návratový parametr 1

Označuje jako cíl skoku blok, na kterém má zpracování programu pokračovat.

Jestliže není naprogramován návratový parametr 3, potom se cíl skoku nachází v programu, z něhož byl aktuální podprogram vyvolán.

Možné údaje jsou následující:

"<číslo bloku>"	Číslo cílového bloku
"<návěští skoku>"	Návěští skoku, které musí být v cílovém bloku nastaveno.
"<řetězec znaků>"	Řetězec znaků, který musí být v programu znám (např. název programu nebo proměnné). Pro programování řetězce znaků v cílovém bloku platí následující pravidla: • **Mezera na konci** (narozdíl od návěští skoku, které je na konci označeno ":"). • **Před řetězcem znaků** smí být uvedeny pouze číslo bloku a/ nebo návěští skoku, **žádné programové příkazy**.

`<blok za cílovým blokem>`: Návratový parametr 2

Vztahuje se na návratový parametr 1.

Typ: INT

Hodnota:	
0	Návrat zpět se uskuteční na blok, který byl udán návratovým parametrem 1.
> 0	Návrat zpět se uskuteční na blok, který následuje za blokem udaným návratovým parametrem 1.

`<počet úrovní při návratu>`:

Návratový parametr 3

Tento parametr uvádí počet úrovní, o které se má skočit zpátky, aby se zpracování dostalo na programovou úroveň, na které má program pokračovat.

Typ: INT

Hodnota:
- 1 Program bude pokračovat na "aktuální programové úrovni - 1" (tedy stejně jako v případě příkazu `RET` bez parametrů).
- 2 Program bude pokračovat na "aktuální programové úrovni - 2", tzn. jedna úroveň bude přeskočena.
- 3 Program bude pokračovat na "aktuální programové úrovni - 3", tzn. dvě úrovně budou přeskočeny.
- ...

Rozsah hodnot: 1 ... 15

`<návrat na začátek programu>`:

Návratový parametr 4

Typ: BOOL

Hodnota:
- 1 Jestliže se uskutečňuje návrat zpět do hlavního programu a jestliže je v něm aktivní **režim dialektu ISO**, skočí se na začátek programu.

Poznámka

Při návratu z podprogramu, při němž je pro vyhledání cílového bloku zadán řetězec znaků, se ve vlaném programu vždy napřed hledá návěští skoku.

Pokud má být cíl skoku jednoznačně definován prostřednictvím řetězce znaků, nesmí proto řetězec znaků odpovídat názvu návěští skoku, neboť potom by se při návratu z podprogramu vždy skákalo na návěští skoku a nikoli na řetězec znaků (viz příklad 2).

Okrajové podmínky

Při návratu zpět přes několik programových úrovní jsou vyhodnocovány příkazy `SAVE` jednotlivých programových úrovní.

Jestliže je při návratu zpět přes několik programových úrovní aktivní modální volání podprogramu a pokud je v některém z přeskakovaných podprogramů naprogramován příkaz pro deaktivování modálního podprogramu `MCALL`, zůstává modální podprogram i nadále aktivní.

⚠ POZOR

Programátor musí dávat pozor na to, aby se při návratu zpět přes několik programových úrovní pokračovalo se správnými modálními nastaveními. Toho je možno dosáhnout např. naprogramováním odpovídajícího hlavního bloku.

Příklady

Příklad 1: Opětovné pokračování v hlavním programu po zpracování programu ASUP

Programování	Komentář
N10010 CALL "UP1"	; Programová úroveň 0 (hlavní program)
N11000 PROC UP1	; Programová úroveň 1
N11010 CALL "UP2"	
N12000 PROC UP2	; Programová úroveň 2
...	
N19000 PROC ASUP	; Programová úroveň 3 (zpracování programu ASUP)
...	
N19100 RET("N10900", ,$P_STACK)	; Skok zpátky z podprogramu
N10900	; Opětovné pokračování v hlavním programu.
N10910 MCALL	; Deaktivování modálního podprogramu.
N10920 G0 G60 G40 M5	; Korekce dalších modálních nastavení.

Příklad 2: Řetězec znaků (<STRING>) jako zadání pro vyhledávání cílového bloku

Hlavní program:

Programový kód	Komentář
PROC MAIN_PROGRAM	
N1000 DEF INT iVar1=1, iVar2=4	
N1010 ...	
N1200 subProg1	; Volání podprogramu "subProg1"
N1210 M2 S1000 X10 F1000	
N1220	
N1400 subProg2	; Volání podprogramu "subProg2"
N1410 M3 S500 Y20	
N1420 ..	
N1500 lab1: iVar1=R10*44	
N1510 F500 X5	
N1520 ...	
N1550 subprog1: G1 X30	; "subProg1" je zde definováno jako návěští skoku.
N1560 ...	
N1600 subProg3	Volání podprogramu "subProg3"
N1610 ...	
N1900 M30	

Podprogram subProg1:

Programový kód	**Komentář**
PROC subProg1	
N2000 R10=R20+100	
N2010 ...	
N2200 RET("subProg2")	; Návrat zpět do hlavního programu na blok N1400

Podprogram subProg2:

Programový kód	**Komentář**
PROC subProg2	
N2000 R10=R20+100	
N2010 ...	
N2200 RET("iVar1")	; Návrat zpět do hlavního programu na blok N1500

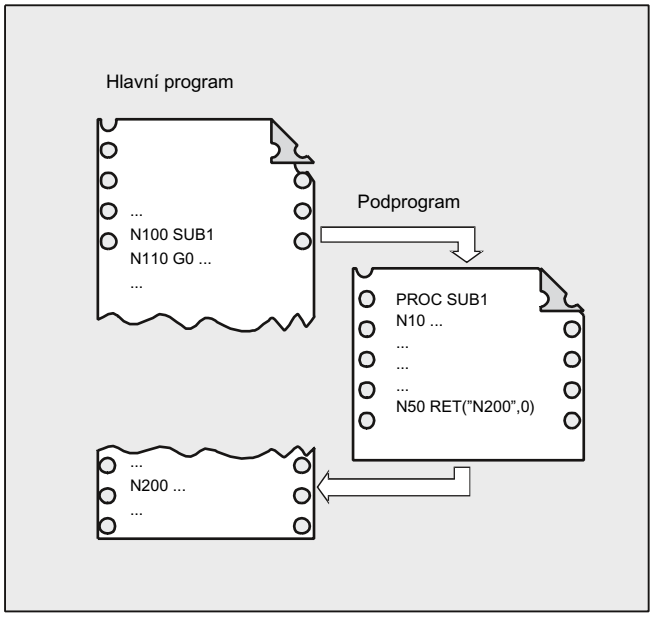
Podprogram subProg3:

Programový kód	**Komentář**
PROC subProg3	
N2000 R10=R20+100	
N2010 ...	
N2200 RET("subProg1")	; Návrat zpět do hlavního programu na blok N1550

Další informace

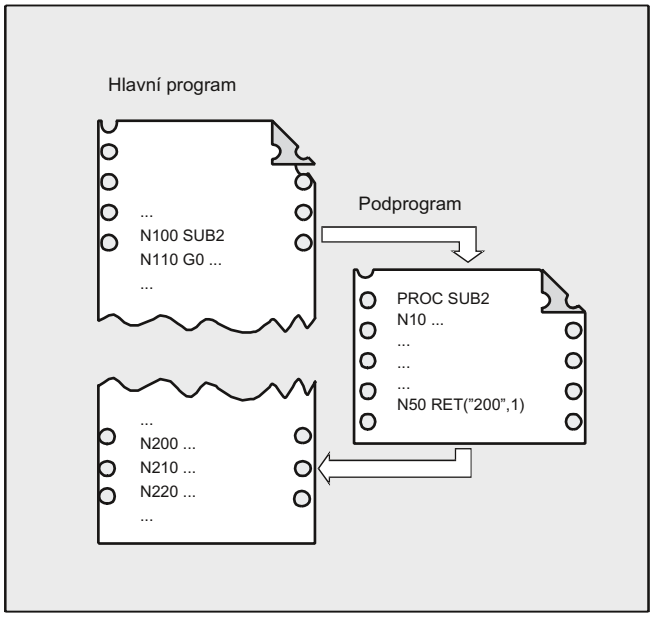
Následující obrázek má objasnit odlišné účinky návratových parametrů 1 až 3.

1. Návratový parametr 1 = "N200", návratový parametr 2 = 0



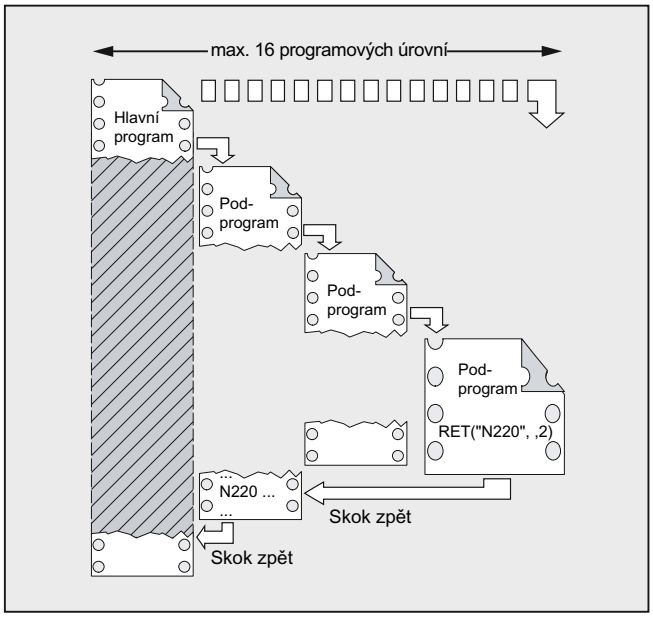
Po příkazu `RET` bude zpracovávání programu pokračovat blokem `N200` v hlavním programu.

2. Návratový parametr 1 = "N200", návratový parametr 2 = 1



Po příkazu `RET` bude zpracovávání programu pokračovat blokem (`N210`), který v hlavním programu následuje za blokem `N200`.

3. Návratový parametr 1 = "N220", návratový parametr 3 = 2



Po příkazu `RET` budou při návratu zpět přeskočeny dvě programové úrovně a zpracovávání programu bude pokračovat blokem `N220`.

1.25.3 Volání podprogramu

1.25.3.1 Volání podprogramu bez předávání parametrů

Funkce

Volání podprogramu se uskutečňuje buď pomocí adresy L a číslem podprogramu nebo zadáním názvu podprogramu.

Jako podprogram je možné volat i hlavní program. Konec programu `M2` nebo `M30` uvedený v hlavním programu bude v tomto případě vyhodnocován jako `M17` (konec programu s návratem zpátky do volajícího programu).

Poznámka

Analogicky je možné spustit podprogram jako program hlavní.

Strategie řídícího systému při vyhledávání:

Existuje *_MPF ?

Existuje *_SPF ?

Z toho vyplývá: Jestliže jsou název volaného podprogramu a název hlavního programu identické, potom bude znovu volán program, který spustil volání. Tento zpravidla nežádoucí efekt je nutno eliminovat jednoznačnou volnou názvu hlavního programu a podprogramu.

Poznámka

Podprogramy, které nevyžadují žádné předávání parametrů, mohou být volány také z inicializačního souboru.

Syntaxe

```
L<číslo>/<název programu>
```

Poznámka

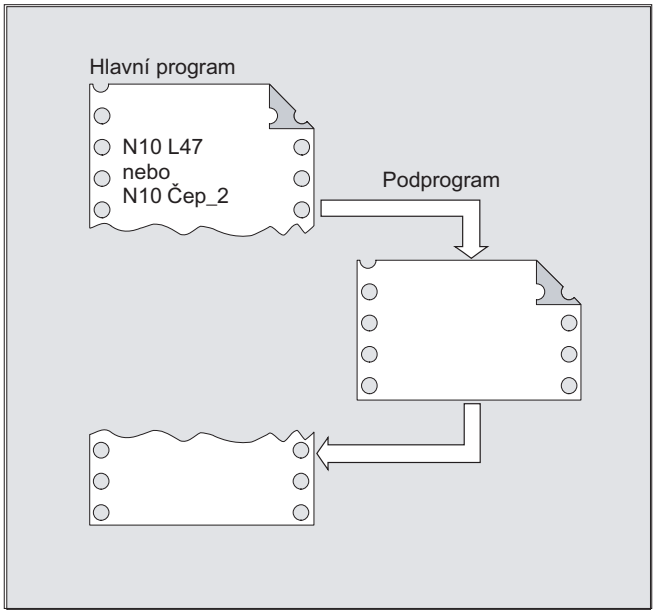
Volání podprogramu musí být vždy naprogramováno v samostatném NC-bloku.

Význam

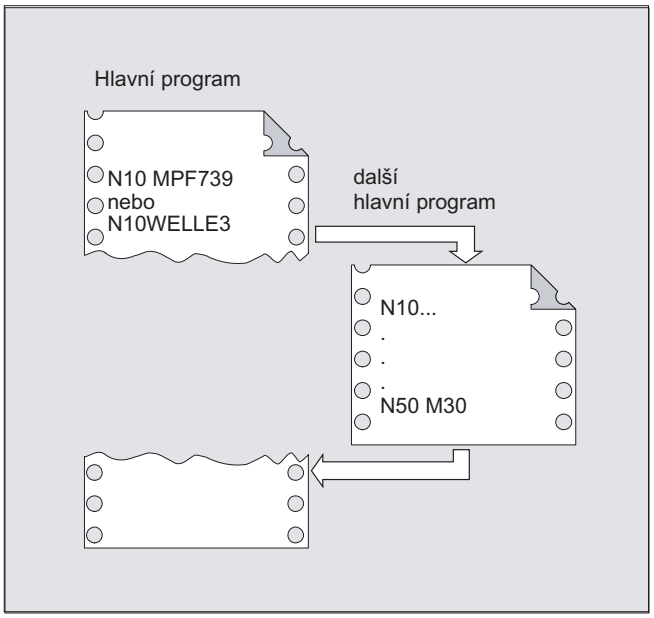
`L`:	Adresa pro volání podprogramu	
`<číslo>`:	Číslo podprogramu	
	Typ:	INT
	Hodnota:	Maximálně 7 míst v desítkové soustavě **Pozor:** Nuly na začátku mají při vyhodnocování názvů svůj význam (⇒ L123, L0123 a L00123 jsou tři různé podprogramy).
`<název programu>`:	Název podprogramu (nebo hlavního programu)	

Příklady

Příklad 1: Volání podprogramu bez předávání parametrů



Příklad 2: Volání hlavního programu jako podprogramu



1.25.3.2 Volání podprogramu s předáváním parametrů (EXTERN)

Funkce

Při volání podprogramu s předáváním parametrů mohou být proměnné nebo hodnoty předávány přímo (nikoli u parametrů typu `VAR`).

Podprogramy s předáváním parametrů musí být před voláním v hlavním programu s příkazem `EXTERN` deklarovány (např. na začátku programu). Uvádí se přitom název podprogramu a typy proměnných v posloupnosti, v jaké jsou předávány.

⚠ POZOR

Jak typy proměnných, tak i posloupnost při předávání se musí shodovat s definicemi, které byly v podprogramu zadány v příkazu `PROC`. Názvy parametrů se mohou v hlavním programu a v podprogramu lišit.

Syntaxe

```
EXTERN <název programu>(<typ_par1>,<typ_par2>,<typ_par3>)
...
<název programu>(<hodnota_par1>,<hodnota_par2>,<hodnota_par3>)
```

⚠ POZOR

Volání podprogramu musí být vždy naprogramováno v samostatném NC-bloku.

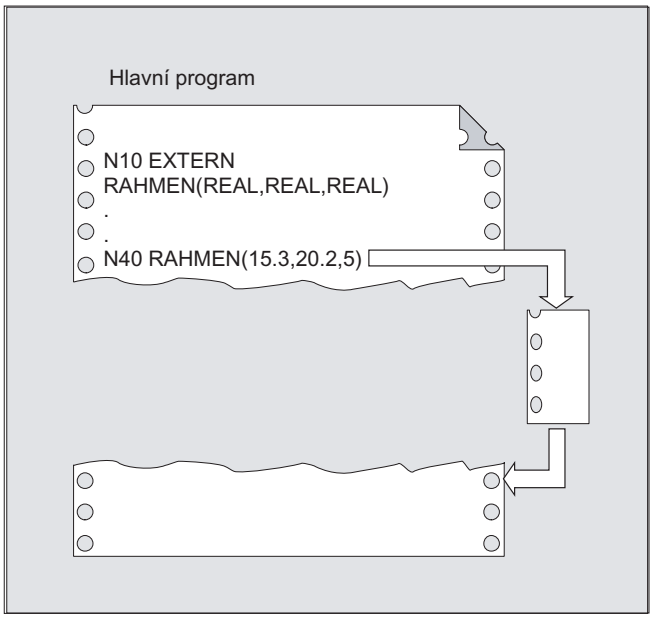
Význam

`<název programu>`:	Název podprogramu
`EXTERN`:	Klíčové slovo pro identifikaci podprogramu s předáváním parametrů **Upozornění:** Příkaz `EXTERN` musí být uváděn jen tehdy, pokud se podprogram nachází v adresáři obrobku nebo v globálním adresáři podprogramů. Při volání cyklů se příkaz `EXTERN` používat nemusí.
`<typ_par1>,<typ_par2>,<typ_par3>`:	Typy proměnných předávaných parametrů v pořadí jejich předávání
`<hodnota_par1>,<hodnota_par2>, <hodnota_par3>`:	Hodnota proměnné pro předávaný parametr

Příklady

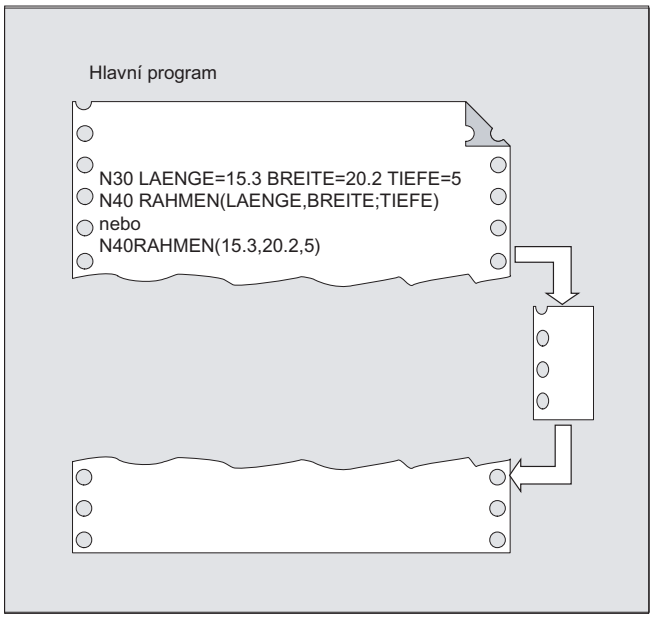
Příklad 1: Volání podprogramu s předcházející deklarací

Programový kód	Komentář
`N10 EXTERN RAHMEN(REAL,REAL,REAL)`	`; Zadání podprogramu.`
`...`	
`N40 RAHMEN(15.3,20.2,5)`	`; Volání podprogramu s předáváním parametrů.`



Příklad 2: Volání podprogramu bez deklarace

Programový kód	Komentář
N10 DEF REAL LAENGE, BREITE, TIEFE	
N20 …	
N30 LAENGE=15.3 BREITE=20.2 TIEFE=5	
N40 RAHMEN(LAENGE,BREITE,TIEFE)	; nebo: N40 RAHMEN(15.3,20.2,5)



1.25.3.3 Počet opakování programu (P)

Funkce

Jestliže má být podprogram zpracován několikrát po sobě, je možné v bloku s voláním podprogramu naprogramovat pomocí adresy `P` požadovaný počet opakování tohoto programu.

⚠ POZOR

Volání podprogramu s opakováním programu a s předáváním parametrů

Parametry se předávají pouze při volání programu, resp. při prvním průchodu. Pro další opakování zůstávají tyto parametry nezměněné. Jestliže si přejete, aby se při opakováních programu tyto parametry měnily, musíte v podprogramu definovat odpovídající opatření.

Syntaxe

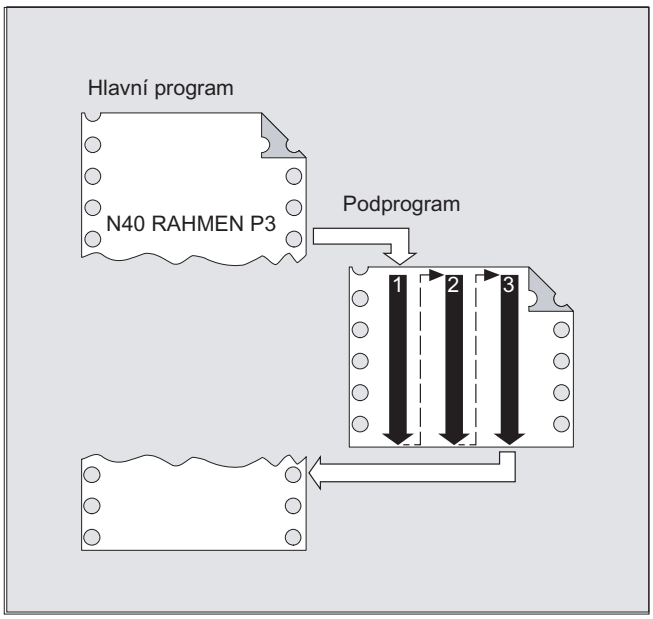
```
<název programu> P<hodnota>
```

Význam

`<název programu>`:	Volání podprogramu	
`P`:	Adresa pro naprogramování opakování programu	
`<hodnota>`:	Počet opakování programu	
	Typ:	INT
	Rozsah hodnot:	1 … 9999 (bez znaménka)

Příklad

Programový kód	**Komentář**
...	
N40 RAHMEN P3	; Podprogram RAHMEN má být zpracován třikrát po sobě.
...	



1.25.3.4 Modální volání podprogramu (MCALL)

Funkce

V případě modálního volání podprogramu pomocí příkazu `MCALL` se podprogram automaticky vyvolá a zpracuje po každém bloku s dráhovým pohybem. Tímto způsobem se dá automatizovat volání podprogramů, které mají být zpracovány na různých místech na obrobku (například při výrobě vrtacích vzorů).

Pro deaktivování této funkce slouží příkaz `MCALL` bez volání podprogramu nebo je možno naprogramovat nové modální volání podprogramu pro nový podprogram.

⚠ POZOR

V jednom zpracovávaném programu může být v dané chvíli v platnosti jen jedno volání typu `MCALL`. Parametry se předávají jen jednou při volání `MCALL`.

Modální podprogram se vyvolává v následujících situacích i bez toho, že by byl naprogramován nějaký pohyb:

- Při naprogramování adres `S` a `F`, když je aktivní některá z funkcí `G0` nebo `G1`.
- Když je v bloku naprogramován pouze příkaz `G0`/`G1` nebo `G`-kódy jiných funkcí.

Syntaxe

```
MCALL <název programu>
```

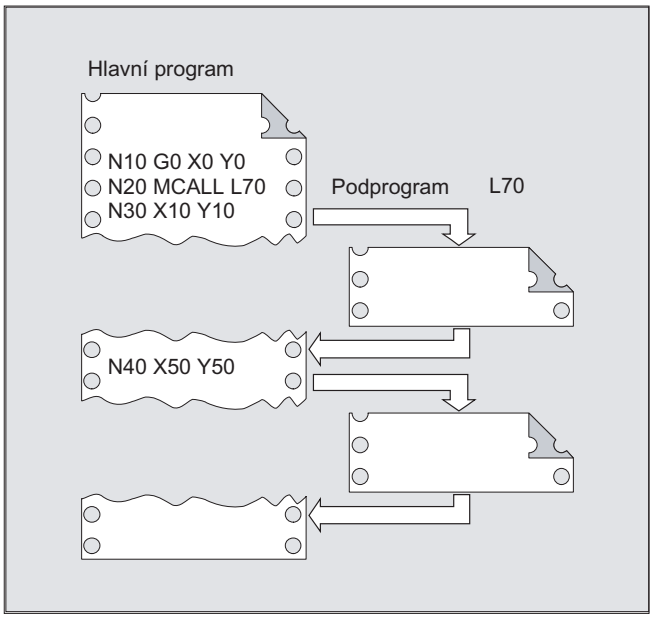
Význam

`MCALL`:	Příkaz pro modální volání podprogramu
`<název programu>`:	Název podprogramu

Příklady

Příklad 1:

Programový kód	Komentář
`N10 G0 X0 Y0`	
`N20 MCALL L70`	`; Modální volání podprogramu.`
`N30 X10 Y10`	`; Najede se na naprogramovanou pozici a potom se zpracuje podprogram L70.`
`N40 X50 Y50`	`; Najede se na naprogramovanou pozici a potom se zpracuje podprogram L70.`



Příklad 2:

Programový kód
N10 G0 X0 Y0
N20 MCALL L70
N30 L80

V tomto příkladu se nacházejí následující NC bloky s naprogramovanými dráhovými osami v podprogramu L80. Podprogram L70 se vyvolává prostřednictvím podprogramu L80.

1.25.3.5 Nepřímé volání podprogramu (CALL)

Funkce

V závislosti na daných podmínkách mohou být na jednom místě vyvolávány různé podprogramy. Za tím účelem se název podprogramu ukládá do proměnné typu STRING. Volání podprogramu se pak uskutečňuje příkazem `CALL` a názvem proměnné.

⚠ POZOR

Nepřímé volání podprogramu je možné pouze pro podprogramy bez předávání parametrů. Pro účely přímého volání podprogramu uložte jeho název do konstanty typu STRING.

Syntaxe

```
CALL <název programu>
```

Význam

`CALL`:	Příkaz pro nepřímé volání podprogramu
`<název programu>`:	Název podprogramu (proměnná nebo konstanta) Typ: STRING

Příklad

Přímé volání s konstantou typu STRING:

Programový kód	**Komentář**
…	
`CALL "/_N_WKS_DIR/_N_SUBPROG_WPD/_N_TEIL1_SPF"`	`; Podprogram TEIL1 s přímým voláním typu CALL.`
…	

Nepřímé volání pomocí proměnné:

Programový kód	**Komentář**
…	
`DEF STRING[100] PROGNAME`	`; Definice proměnné.`
`PROGNAME="/_N_WKS_DIR/_N_SUBPROG_WPD/_N_TEIL1_SPF"`	`; Přiřazení podprogramu TEIL1 proměnné PROGNAME.`
`CALL PROGNAME`	`; Nepřímé volání podprogramu TEIL1 pomocí příkazu CALL a proměnné PROGNAME.`
…	

1.25.3.6 Nepřímé volání podprogramu se zadáním úseku programu, který á být zpracován (CALL BLOCK ... TO ...)

Funkce

Pomocí příkazu `CALL` a kombinace klíčových slov `BLOCK ... TO` se nepřímo vyvolává podprogram a v něm se zpracovává úsek, který začíná počátečním a končí koncovým návěštím.

Syntaxe

```
CALL <název programu> BLOCK <počáteční návěští> TO <koncové návěští>
CALL BLOCK <počáteční návěští> TO <koncové návěští>
```

Význam

`CALL`:	Příkaz pro nepřímé volání podprogramu
`<název programu>`:	Název podprogramu (proměnná nebo konstanta), který obsahuje úsek programu, který má být zpracován (**údaj je nepovinný**). Typ: STRING **Upozornění:** Jestliže parametr `<název programu>` není naprogramován, úsek programu označený návěštími `<počáteční návěští>` a `<koncové návěští>` bude vyhledán v aktuálním programu a bude zpracován.
`BLOCK ... TO ...` :	Kombinace klíčových slov pro nepřímé zpracování úseku programu
`<počáteční návěští`:	Proměnná, která odkazuje na začátek úseku programu, který má být zpracován. Typ: STRING
`<koncové návěští`:	Proměnná, která odkazuje na konec úseku programu, který má být zpracován. Typ: STRING

Příklad

Hlavní program:

```
Programový kód                                  Komentář
...
DEF STRING[20] STARTLABEL, ENDLABEL             ; Definice proměnných pro
                                                  počáteční a koncové návěští.
STARTLABEL="LABEL_1"
ENDLABEL="LABEL_2"
...
```

Programový kód	Komentář
CALL "CONTUR_1" BLOCK STARTLABEL TO ENDLABEL	; Nepřímé volání podprogramu a označení úseku programu, který má být zpracován.
...	

Podprogram:

Programový kód	Komentář
PROC CONTUR_1 ...	
LABEL_1	; Počáteční návěští: začátek zpracovávaného úseku programu
N1000 G1 ...	
...	
LABEL_2	; Koncové návěští: konec zpracovávaného úseku programu
...	

1.25.3.7 Nepřímé volání programu naprogramovaného v jazyce ISO (ISOCALL)

Funkce

Pomocí nepřímého volání programu pomocí příkazu `ISOCALL` je možné vyvolat program naprogramovaný v jazyce ISO. Přitom se aktivuje režim ISO nastavený ve strojních parametrech. Na konci programu se znovu stane platným původní režim zpracování. Jestliže není ve strojních parametrech nastaven žádný režim ISO, uskuteční se volání podprogramu v režimu Siemens.

Pokud budete potřebovat další informace k režimu ISO, viz:
Literatura:
Příručka Popis funkcí, Dialekty ISO

Syntaxe

```
ISOCALL <název programu>
```

Význam

`ISOCALL`:	Klíčové slovo pro nepřímé volání podprogramu, při kterém se aktivuje režim ISO nastavený ve strojních parametrech.
`<název programu>`:	Název programu naprogramovaného v jazyce ISO (proměnná nebo konstanta typu STRING)

Příklad: Vyvolání kontury s programováním cyklu v režimu ISO

Programový kód	**Komentář**
0122_SPF	; Popis kontury v režimu ISO
N1010 G1 X10 Z20	
N1020 X30 R5	
N1030 Z50 C10	
N1040 X50	
N1050 M99	
N0010 DEF STRING[5] PROGNAME = "0122"	; Výrobní program (cyklus) od firmy Siemens
...	
N2000 R11 = $AA_IW[X]	
N2010 ISOCALL PROGNAME	
N2020 R10 = R10+1	; Zpracování programu 0122 v režimu ISO
...	
N2400 M30	

1.25.3.8 Volání podprogramu s udáním cesty a parametrů (PCALL)

Funkce

Pomocí příkazu `PCALL` mohou být vyvolávány podprogramy s absolutním zadáním cesty a s předáváním parametrů.

Syntaxe

```
PCALL <cesta/název programu>(<parametr 1>,…,<parametr n>)
```

Význam

`PCALL`:	Klíčové slovo pro volání podprogramu s absolutním zadáním cesty.
`<cesta/název programu>`:	Absolutní zadání cesty včetně názvu podprogramu začíná znakem "/". Jestliže žádná absolutní cesta nebyla zadána, chová se příkaz `PCALL` stejně jako standardní volání s identifikátorem podprogramu. Identifikátor programu se zadává bez úvodního řetězce _N_ a bez přípony. Jestliže má být název programu naprogramován s úvodním řetězcem a s příponou, potom musí být explicitně vysvětlen i s předponou a s příponou pomocí příkazu `EXTERN`.
`<parametr 1>, ...`:	Aktuální parametry v souladu s příkazem `PROC` v podprogramu.

Příklad

Programový kód
`PCALL/_N_WKS_DIR/_N_WELLE_WPD/WELLE(parametr1,parametr2,…)`

1.25.3.9 Rozšíření cesty pro vyhledávání při volání podprogramu (CALLPATH)

Funkce

Pomocí příkazu `CALLPATH` je možné rozšířit cestu pro vyhledávání při volání podprogramu.

Tímto způsobem mohou být vyvolávány také podprogramy z adresáře obrobku, který není vybrán, aniž by bylo nutno udávat úplný absolutní název cesty tohoto podprogramu.

Rozšíření cesty pro vyhledávání se provádí před zadáním pro uživatelské cykly (_N_CUS_DIR).

Následující události způsobí, že se rozšíření cesty pro vyhledávání opět deaktivuje:

- Příkaz `CALLPATH` s mezerami
- Příkaz `CALLPATH` bez parametrů
- Konec výrobního programu
- Reset

Syntaxe

```
CALLPATH("<název cesty>")
```

Význam

`CALLPATH`:	Klíčové slovo pro programovatelné rozšíření cesty pro vyhledávání Musí být naprogramováno na samostatném řádku výrobního programu.
`<název cesty>`:	Konstanta / proměnná typu STRING. Obsahuje absolutní údaj cesty do adresáře, o který má být cesta pro vyhledávání rozšířena. Zadání cesty začíná znakem "/". Cesta musí být zadána celá, včetně předpon a přípon. Maximální délka cesty je 128 bytů. Pokud parametr <název cesty> obsahuje mezeru nebo pokud je příkaz `CALLPATH` vyvolán bez parametrů, příkaz pro rozšíření cesty pro vyhledávání se opět deaktivuje.

Příklad

Programový kód
`CALLPATH("/_N_WKS_DIR/_N_MYWPD_WPD")`

Tímto příkazem se vytvoří následující cesta pro vyhledávání (položka 5 je nová):

1. Aktuální adresář/identifikátor podprogramu
2. Aktuální adresář/identifikátor podprogramu_SPF
3. Aktuální adresář/identifikátor podprogramu_MPF
4. /_N_SPF_DIR/identifikátor podprogramu_SPF
5. **/_N_WKS_DIR/_N_MYWPD/identifikátor podprogramu_SPF**
6. /N_CUS_DIR/_N_MYWPD/identifikátor podprogramu_SPF
7. /_N_CMA_DIR/identifikátor podprogramu_SPF
8. /_N_CST_DIR/identifikátor podprogramu_SPF

Okrajové podmínky

- Příkaz `CALLPATH` kontroluje, zda naprogramovaný název cesty skutečně existuje. Vyskytne-li se chyba, zpracování výrobního programu se přeruší a aktivuje se alarm korekčního bloku 14009.
- Příkaz `CALLPATH` může být naprogramován i v souborech INI. Potom je v platnosti po celou dobu zpracovávání souboru INI (soubor WPD-INI nebo inicializační program pro data aktivní v NC systému, např. framy v 1. kanálu _N_CH1_UFR_INI). Pak se znovu obnoví původní cesta pro vyhledávání.

1.25.3.10 Zpracovávání externího podprogramu (EXTCALL)

Funkce

Pomocí příkazu EXTCALL je možné načíst výrobní program z externí paměti (lokální jednotka, síťová jednotka, jednotka připojená přes USB) a potom jej zpracovat jako podprogram.

Cesta do externího programového adresáře může být předem určena pomocí nastavovaného parametru:

SD42700 $SC_EXT_PROG_PATH

Spolu s cestou, příp. s identifikátorem programu, které jsou zadány při volání pomocí příkazu EXTCALL, tak vzniká celková cesta k volanému programu.

Poznámka

Cíl skoku

U externích programů, které obsahují příkazy skoku (GOTOF, GOTOB, CASE, FOR, LOOP, WHILE, REPEAT, IF, ELSE, ENDIF atd.), se musí cíle skoků nacházet uvnitř paměti pro načítání tohoto programu. Velikost této vyrovnávací paměti se nastavuje pomocí parametru:

MD18360 MM_EXT_PROG_BUFFER_SIZE

Parametry

Při vyvolávání externího programu mu není možné předávat žádné parametry.

Syntaxe

```
EXTCALL("<cesta/><název programu>")
```

Význam

EXTCALL:	Příkaz pro vyvolání externího podprogramu	
"<cesta/><název programu>":	Konstanta / proměnná typu STRING	
	<cesta/>:	Absolutní nebo relativní zadání cesty (**není nutné**)
	<název programu>:	Název programu se zadává bez předpony "_N_". Příponu souboru ("MPF", "SPF") je možné spolu se znaky "_" nebo "." v názvu programu uvést (**není nutné**). Příklad: "WELLE" nebo "WELLE_SPF" příp. "WELLE.SPF"

Poznámka

Zadání cesty: Zkrácené popisy

Při zadávání cesty se mohou používat následující zkrácené popisy:

- **LOCAL_DRIVE:** pro lokální jednotku
- **CF_CARD:** pro kompaktní Flash-kartu
- **USB:** po konektor USB na čelním panelu

CF_CARD: a **LOCAL_DRIVE:** se mohou používat jako alternativy.

Poznámka

Zpracovávání z externího zdroje pomocí jednotky připojené na USB

Jestliže se mají přenášet externí výrobní programy z externí jednotky USB prostřednictvím rozhraní USB, smí se pro tento účel používat jedině rozhraní s označením X203 a s názvem "TCU_1".

UPOZORNĚNÍ

Zpracovávání z externího zdroje pomocí flash-disku (v konektoru USB na čelním panelu)

Přímé zpracovávání z USB Flash-disku připojeného do konektoru USB na čelném panelu se nedoporučuje, protože přerušení spojení s USB Flash-diskem v průběhu zpracovávání výrobního programu v důsledku problémů s kontaktem, výpadku, přerušení kvůli nárazu nebo kvůli vytažení způsobeného omylem má za následek okamžité zastavení zpracování. Nástroj a/nebo obrobek by se přitom mohly poškodit.

Příklad

Zpracovávání z lokální jednotky

Hlavní program:

Programový kód

```
N010 PROC MAIN
N020 ...
N030 EXTCALL ("SCHRUPPEN")
N040 ...
N050 M30
```

Externí podprogram:

Programový kód

```
N010 PROC SCHRUPPEN
N020 G1 F1000
N030 X= ... Y= ... Z= ...
N040 ...
...
...
N999999 M17
```

Hlavní program "MAIN.MPF" se nachází v paměti NC systému a je vybrán pro zpracování:

Program "SCHRUPPEN.SPF" příp. "SCHRUPPEN.MPF", který má být načítán, se nachází na lokální jednotce v adresáři "/user/sinumerik/data/prog/WKS.DIR/WST1.WPD".

Cesta k podprogramu je předem uložena v nastavovaném parametru SD42700:

SD42700 $SC_EXT_PROG_PATH = "LOCAL_DRIVE:WKS.DIR/WST1.WPD"

Poznámka

Bez udání cesty v parametru SD42700 by musel být příkaz `EXTCALL` pro tento příklad naprogramován následujícím způsobem:

```
EXTCALL("LOCAL_DRIVE:WKS.DIR/WST1.WPD/SCHRUPPEN")
```

Další informace

Volání příkazem EXTCALL s absolutním udáním cesty

Pokud podprogram existuje v adresáři podle zadané cesty, pak se po volání příkazem `EXTCALL` spustí jeho zpracování. Jestliže program neexistuje, zpracovávání programu se přeruší.

Volání příkazem EXTCALL s relativním udáním cesty / bez udání cesty

V případě volání pomocí příkazu `EXTCALL` s relativním zadáním cesty, příp. bez udání cesty, jsou stávající programové paměti prohledávány podle následujícího vzoru:

- Jestliže je v nastavovaném parametru SD42700 $SC_EXT_PROG_PATH předem stanovena nějaká cesta, bude po zadání volání pomocí příkazu `EXTCALL` (název programu, příp. s relativním udáním cesty) vyhledávání začínat v rámci této cesty. Absolutní cesta potom vznikne zřetězením následujících posloupností znaků:
  - Předem nastavená cesta v nastavovaném parametru SD42700
  - Lomítko "/" jako oddělovací znak
  - cesta, příp. identifikátor podprogramu uvedené u příkazu `EXTCALL`
- Jestliže nebyl vyvolávaný podprogram pod předem definovanou cestou nalezen, jako další jsou prohledávány adresáře uživatelské paměti podle zadání v příkazu volání `EXTCALL`.
- Vyhledávání skončí, jakmile je podprogram poprvé nalezen. Jestliže vyhledávání skončí neúspěchem, zpracování programu se přeruší.

Nastavitelná paměť pro načítání (vyrovnávací paměť FIFO)

Pro zpracovávání programu v režimu "zpracování z externího zdroje" (hlavní program nebo podprogram) je v NCK zapotřebí paměť pro načítání tohoto programu. Rozsah této paměti je předem nastaven na 30 kbytů a tato velikost může být stejně jako i další strojní parametry ovlivňující paměť změněna pouze výrobcem stroje, aby se vyhovělo případným požadavkům.

Vyrovnávací paměť pro načítání musí být nastavena pro všechny programy (hlavní programy nebo podprogramy), které jsou současně zpracovávány v režimu "zpracování z externího zdroje".

RESET, POWER ON

Operace RESET a POWER ON způsobují přerušení externího volání podprogramu a vymazání obsahu vyrovnávací paměti, kam se program načítá.

Podprogram vybraný pro "zpracování z externího zdroje" zůstává pro tuto funkci vybrán, i když je proveden RESET nebo nastane konec výrobního programu. Vypnutím a zapnutím systému (Power-On) se toto nastavení ale ztratí.

Literatura

Pokud budete potřebovat další informace týkající se funkce "Zpracování z externího zdroje", viz:

Příručka Popis funkcí, Základní funkce; BAG, kanál, zpracování programu, chování při resetu (K1)

1.25.4 Cykly

1.25.4.1 Dosazování parametrů do uživatelských cyklů

Funkce

Pomocí souborů cov.com a uc.com můžete dosazovat parametry do svých vlastních cyklů:

cov.com	Přehled cyklů
uc.com	Popis volání cyklů

Soubor cov.com je dodáván spolu se standardními cykly a je zapotřebí jej odpovídajícím způsobem doplnit. Soubor uc.com musí uživatel sestavit sám.

Oba soubory je zapotřebí načíst do pasivního systému souborů do adresáře uživatelských cyklů, příp. v programu je nutno udat odpovídající zadání cesty:

```
;$PATH=/_N_CUS_DIR
```

Přizpůsobení souboru cov.com (přehled cyklů)

Soubor cov.com dodávaný spolu se standardními cykly má následující strukturu:

`%_N_COV_COM`	Název souboru
`;$PATH=/_N_CST_DIR`	Název cesty
`;Vxxx 11.12.95 Sca přehled cyklů`	Řádek komentáře
`C1(CYCLE81) Vrtání, navrtávání středicích důlků`	Volání 1. cyklu
`C2(CYCLE82) Vrtání, čelní zahlubování`	Volání 2. cyklu
`...`	
`C24(CYCLE98) Řetězec závitů`	Volání posledního cyklu
`M17`	Konec souboru

Pro každý nově připojovaný cyklus je zapotřebí vložit jeden řádek s následující syntaxí:
`C<číslo> (<název cyklu>) <textový komentář>`

kde:

`<číslo>`:	Libovolné celé číslo, které nesmí být už dříve v tomto souboru použito
`<Název cyklu>`:	název programu připojovaného cyklu
`<Komentář>`:	Textový komentář k cyklu (volitelný)

Příklad:

```
C25 (MEIN_ZYKLUS_1) uživatelský cyklus_1
C26 (SPEZIALZYKLUS)
```

Popis uživatelských cyklů v souboru uc.com

Hlavička každého cyklu:

Stejná jako soubor cov.com. před tím je uvedeno "//":
`//C <číslo> (<název cyklu>) <textový komentář>`

Příklad:
`//C25 (MEIN_ZYKLUS_1) uživatelský cyklus_`

Řádek popisu každého parametru:

`(<identifikace datového typu> / <minimální hodnota> <maximální hodnota> / <předdefinovaná hodnota> /<komentář>)`

kde:

`<Identifikace datového typu>`:	`R`: pro typ REAL `I`: pro typ INTEGER `C`: pro znak (1 znak) `S`: pro typ STRING
`<minimální hodnota>` `<maximální hodnota>`:	definice rozsahu hodnot (může být vypuštěno) Hranice zadávaných hodnot, které jsou během zadávání kontrolovány. Hodnoty mimo tento rozsah není možné zadat. Je možné zadat také hodnoty výčtem, přičemž z těchto hodnot je možné vybírat pomocí přepínacího tlačítka. Tento výčet začíná znakem "*" a v takovém případě jsou jiné hodnoty nepřípustné. Příklad: `(I/*123456/1/druh opracování)` V případě typů STRING a CHARACTER neplatí žádné hranice.
`<Předdefinovaná hodnota>`:	Hodnota, která je při volání cyklu v odpovídající obrazovce předem nastavena (může být vypuštěno) Předdefinovaná hodnota může být obsluhou změněna.
`<Komentář>`:	Textový komentář (maximálně 50 znaků), který se může v obrazovce pro volání cyklu vypisovat před vstupním polem pro daný parametr.

Příklad

Pro následující dvojici cyklů má být nově sestaveno dosazování parametrů cyklům:

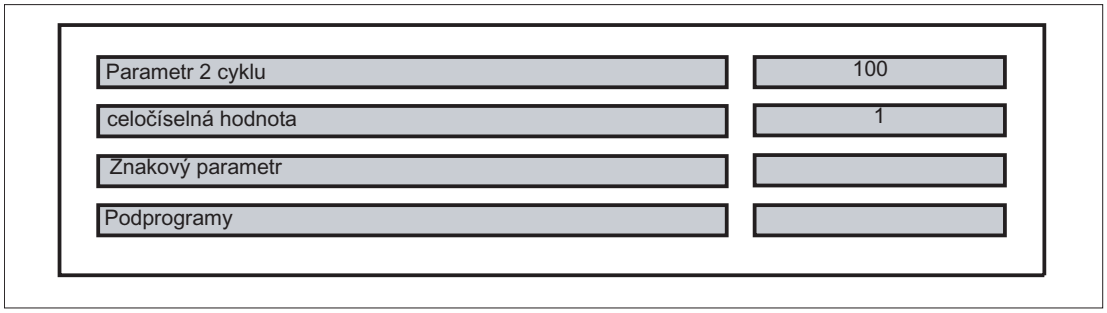
```
PROC MEIN_ZYKLUS_1 (REAL PAR1, INT PAR2, CHAR PAR3, STRING[10] PAR4)
Cyklus má následující předávané parametry::
PAR1:                ; Reálná hodnota v rozsahu -1000.001 <= PAR2 <= 123.456,
                       předdefinované nastavení 100
PAR2:                ; Kladná celočíselná hodnota v rozsahu 0 <= PAR3 <= 999999,
                       předdefinované nastavení 0
PAR3:                ; 1 znak ASCII
PAR4:                ; Řetězec o délce 10 pro název podprogramu
...
M17
```

```
PROC SPEZIALZYKLUS (REAL WERT1, INT WERT2)
Cyklus má následující předávané parametry::
WERT1:               ; Reálná hodnota bez omezení rozsahu hodnot a předdefinovaného
                       nastavení
WERT2:               ; Celočíselná hodnota bez omezení rozsahu hodnot a
                       předdefinovaného nastavení
...
M17
```

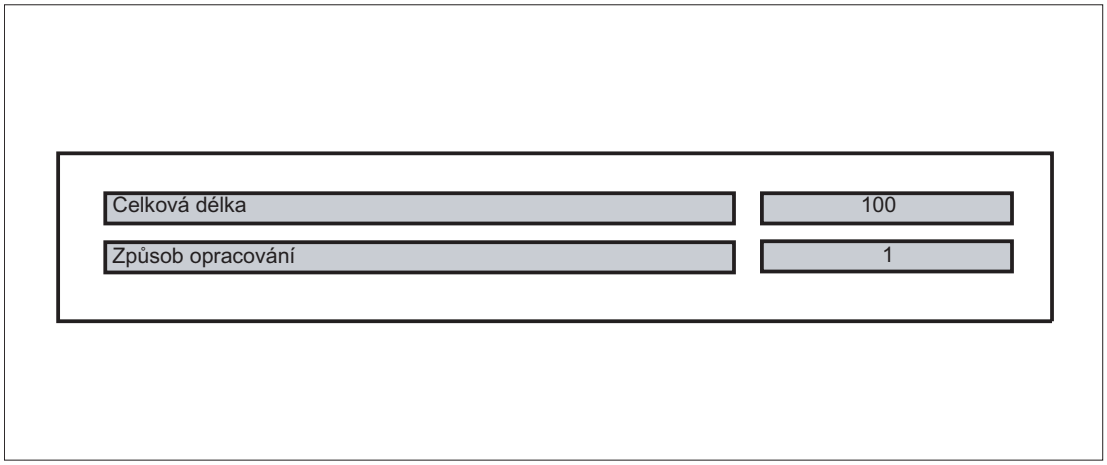
Odpovídající soubor uc.com:

```
%_N_UC_COM
;$PATH=/_N_CUS_DIR
//C25(MEIN_ZYKLUS_1) uživatelský cyklus_1
(R/-1000.001 123.456 / 100 /Parametr_2 cyklu)
(I/0 999999 / 1 / celočíselná hodnota)
(C//"A" / znakový parametr)
(S///název podprogramu)
//C26(SPEZIALZYKLUS)
(R///celková délka)
(I/*123456/3/druh opracování)
M17
```

Zobrazovaná maska pro cyklus `MEIN_ZYKLUS_1`



Zobrazovaná maska pro cyklus `SPEZIALZYKLUS`



1.26 Technika maker (DEFINE ... AS)

⚠ POZOR
Pomocí techniky maker je možné programovací jazyk řídícího systému silně změnit! Techniku maker proto používejte velmi uvážlivě!

Funkce

Makro přestavuje posloupnost jednotlivých příkazů, kterou jsou soustředěny do nového celku, který má vlastní název. Jako makro mohou být připojovány také G-, M- a H-funkce nebo názvy podprogramů. Při vyvolání makra v průběhu zpracování programu jsou postupně zpracovány příkazy, které byly pod název makra naprogramovány.

Aplikace

Posloupnosti příkazů, které se opakují, se naprogramují jen jednou jako makro do vlastního modulu makra (soubor makra) nebo jednou na začátku programu. Makro je potom možné vyvolat v libovolném hlavním programu nebo podprogramu a nechat jej tak zpracovat.

Aktivování

Aby bylo možné makra ze souboru maker v NC-programu používat, musí být soubor maker načten do NC systému.

Syntaxe

Definice makra:
```
DEFINE <název makra> AS <příkaz 1> <příkaz 2> ...
```

Volání v NC programu:
```
<název makra>
```

Význam

`DEFINE ... AS`:	Kombinace klíčových slov pro definici makra
`<název makra>`:	Název makra Jako názvy maker jsou přípustné pouze identifikátory. Pomocí názvu makra se potom v NC programu makro vyvolává.
`<příkaz>`:	Programový příkaz, který má být v makru obsažen.

Pravidla pro definice makra

- V makru mohou být definovány libovolné identifikátory, G-, M-, H-funkce a L-názvy programů.
- Makra mohou být definována také v NC-programu.
- Makra G-funkcí mohou být definována pouze globálně v řídícím systému v modulu maker.
- H- a L-funkce mohou být naprogramovány jako 2-místné.
- M- a G-funkce mohou být naprogramovány jako 3-místné.

⚠ POZOR
Definice klíčových slov a rezervované názvy nesmí být pomocí makra změněny-

Okrajové podmínky

Vnoření maker není možné.

Příklady

Příklad 1: Definice makra na začátku programu

Programový kód	**Komentář**
`DEFINE LINIE AS G1 G94 F300`	`; Definice makra`
`...`	
`...`	
`N70 LINIE X10 Y20`	`; Volání makra`
`...`	

Příklad 2: Definice maker v souboru makra

Programový kód	**Komentář**
`DEFINE M6 AS L6`	`; Při výměně nástroje se bude vyvolávat podprogram, který převezme přenos nezbytných dat. V podprogramu se spouští M-funkce pro vlastní výměna nástroje (např. M106).`
`DEFINE G81 AS DRILL(81)`	`; Dodatečné vytvoření G-funkce podle normy DIN.`
`DEFINE G33 AS M333 G333`	`; Při řezání závitu je požadována synchronizace s PLC. Předcházející G-funkce G33 byla pomocí strojního parametru přejmenována na G333, programování ale zůstává pro uživatele stejné.`

Příklad 3: Externí soubor maker

Po načtení externího souboru maker do řídícího systému musí být soubor maker načten do NC systému. Teprve potom mohou být makra v NC-programech používána.

Programový kód	**Komentář**
%_N_UMAC_DEF	
;$PATH=/_N_DEF_DIR	; Specifická uživatelská makra
DEFINE PI AS 3.14	
DEFINE TC1 AS M3 S1000	
DEFINE M13 AS M3 M7	; Vřeteno se otáčí vpravo, zapnutí chladicí kapaliny
DEFINE M14 AS M4 M7	; Vřeteno se otáčí vlevo, zapnutí chladicí kapaliny
DEFINE M15 AS M5 M9	; Zastavení vřetena, vypnutí chladicí kapaliny
DEFINE M6 AS L6	; Vyvolání programu pro výměnu nástroje
DEFINE G80 AS MCALL	; Deaktivování cyklu pro vrtání
M30	

Správa souborů a programů

2

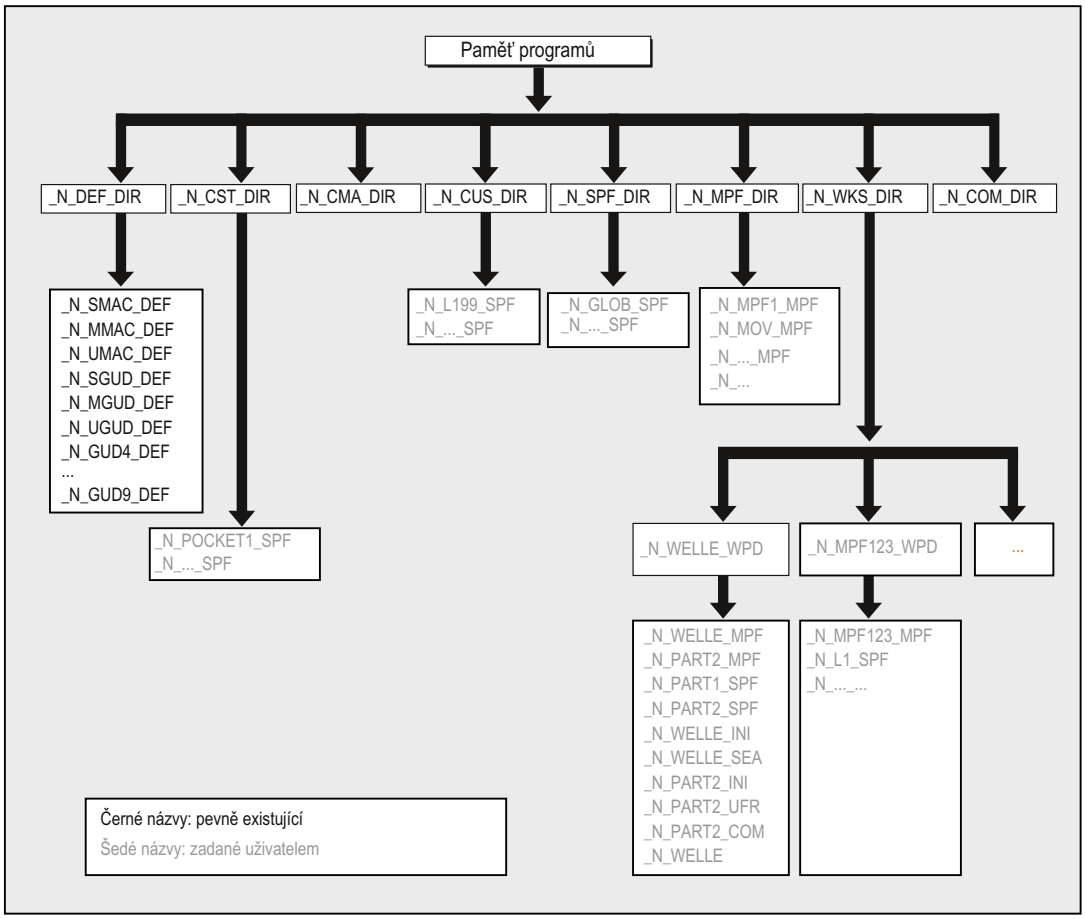
2.1 Paměť programů

Funkce

V paměti programů jsou trvale uloženy soubory a programy (např. hlavní programy a podprogramy, definice maker) (→ pasivní systém souborů).

Literatura:
Příručka Popis funkcí, Rozšiřovací funkce; Konfigurace paměti (S7)

Kromě toho existuje určitý počet typů souborů, které zde mohou být dočasně uloženy a v případě potřeby (např. při opracovávání určitých obrobků) jsou přeneseny do pracovní paměti (např. pro účely inicializace).



Standardní adresáře

Standardně jsou k dispozici následující adresáře:

Adresář	Obsah
_N_DEF_DIR	Datové moduly a moduly maker
_N_CST_DIR	Standardní cykly
_N_CMA_DIR	Cykly výrobce
_N_CUS_DIR	Uživatelské cykly
_N_WKS_DIR	Obrobky
_N_SPF_DIR	Globální podprogramy
_N_MPF_DIR	Hlavní programy
_N_COM_DIR	Komentáře

Typy souborů

Do paměti programů se mohou přenášet následující typy souborů:

Typ souboru	Popis
název_MPF	Hlavní program
název_SPF	Podprogram
název_TEA	Strojní parametry
název_SEA	Nastavované parametry
název_TOA	Korekční parametry nástroje
název_UFR	Posunutí počátku / frame
název_INI	Inicializační soubory
název_GUD	Globální uživatelská data
název_RPA	R-Parametry
název_COM	Komentář
název_DEF	Definice pro globální uživatelská data a makra

Hlavní adresář obrobků (_N_WKS_DIR)

Hlavní adresář obrobků je standardně vytvořen v paměti programů pod označením `_N_WKS_DIR`. Hlavní adresář obrobků obsahuje všechny obrobky, které máte naprogramovány, a odpovídající adresáře obrobků.

Adresáře obrobků (..._WPD)

Kvůli flexibilní manipulaci s daty a s programy mohou být určité soubory a programy spojeny vazbou nebo mohou být ukládány do adresářů jednotlivých obrobků.

Adresář obrobku obsahuje soubory, které jsou nezbytné pro opracovávání obrobku. Může se jednat o hlavní programy, podprogramy, libovolné inicializační programy a soubory komentářů.

Inicializační programy se po zvolení příslušného programu při prvním spuštění výrobního programu jedenkrát zpracují (v závislosti na strojním parametru MD11280 $MN_WPD_INI_MODE).

Příklad:

Adresář obrobku _N_WELLE_WPD, který byl založen pro obrobek WELLE, obsahuje následující soubory:

Soubor	Popis
_N_WELLE_MPF	Hlavní program
_N_PART2_MPF	Hlavní program
_N_PART1_SPF	Podprogram
_N_PART2_SPF	Podprogram
_N_WELLE_INI	Všeobecný inicializační program pro data daného obrobku
_N_WELLE_SEA	Inicializační program nastavovaných parametrů
_N_PART2_INI	Všeobecný inicializační program pro data programu PART2
_N_PART2_UFR	Inicializační program pro data framu pro program PART2
_N_WELLE_COM	Soubor komentářů

Zakládání adresářů obrobku na externím PC

Postup popisovaný v následujících odstavcích se uskutečňuje na externí datové stanici. Pokud budete potřebovat informace o správě souborů a programů (z PC připojeného k řídícímu systému) přímo na řídícím systému, nahlédněte do návodu k obsluze.

Založení adresáře obrobku se zadáním cesty ($PATH=...)

Na druhém řádku souboru je uvedena pomocí příkazu `$PATH=…` cílová cesta. Soubor se potom uloží pod uvedenou cestou.

Příklad:

Programový kód

```
%_N_WELLE_MPF
;$PATH = /_N_WKS_DIR/_N_WELLE_WPD
N10 G0 X… Z…
...
M2
```

Soubor _N_WELLE_MPF se uloží v adresáři /_N_WKS_DIR/_N_WELLE_WPD.

Založení adresáře obrobku bez udání cesty

Pokud údaj cesty chybí, potom se soubory s příponou _SPF uloží do adresáře /_N_SPF_DIR, soubory s příponou _INI se uloží do pracovní paměti a všechny ostatní soubory se uloží do adresáře /_N_MPF_DIR.

Příklad:

Programový kód

```
%_N_WELLE_SPF
...
M17
```

Soubor _N_WELLE_SPF se uloží v adresáři /_N_SPF_DIR.

Volba obrobku pro zpracování

Adresář obrobku může být vybrán pro zpracování v určitém kanálu, Pokud se v tomto adresáři nachází hlavní program **stejného názvu** nebo jen jediný hlavní program (_NPF), bude tento program automaticky vybrán pro zpracování

Literatura:
/BAD/, Návod k obsluze systému HMI-Advanced, kapitoly "Seznam úloh" a "Volba programu pro zpracování"

Cesta pro vyhledávání při volání podprogramu

Pokud cesta pro vyhledávání není ve výrobním programu při vyvolávání podprogramu (nebo inicializačního souboru) explicitně uvedena, bude volaný program hledán podle pevně stanovené cesty pro vyhledávání.

Volání podprogramu s absolutním udáním cesty

Příklad:

Programový kód

```
...
CALL"/_N_CST_DIR/_N_CYCLE1_SPF"
...
```

Volání podprogramu bez absolutního udáním cesty

Programy jsou zpravidla vyvolávány bez uvedení cesty.

Příklad:

Programový kód

```
...
CYCLE1
...
```

Adresáře jsou prohledávány v závislosti na volaném programu v následující posloupnosti:

Č.	Adresář	Popis
1	aktuální adresář / *název*	Hlavní adresář obrobku nebo standardní adresář _N_MPF_DIR
2	aktuální adresář / *název_SPF*	
3	aktuální adresář / *název_MPF*	
4	/_N_SPF_DIR / *název_SPF*	Globální podprogramy
5	/_N_CUS_DIR / *název_SPF*	Uživatelské cykly
6	/_N_CMA_DIR / *název_SPF*	Cykly výrobce
7	/_N_CST_DIR / *název_SPF*	Standardní cykly

Programování cesty pro vyhledávání při volání podprogramu (CALLPATH)

Cesta pro vyhledávání při volání podprogramu může být rozšířena pomocí příkazu výrobního programu `CALLPATH`.

Příklad:

Programový kód

```
CALLPATH("/_N_WKS_DIR/_N_MYWPD_WPD")
...
```

Cesta pro vyhledávání se před položkou 5 (uživatelský cyklus) rozšíří v souladu s příkazem uvedeným v programu.

Pokud budete potřebovat další informace týkající se programovatelné cesty pro vyhledávání při volání podprogramů pomocí příkazu `CALLPATH`, viz kapitola "Rozšíření cesty pro vyhledávání při volání podprogramu pomocí příkazu CALLPATH".

2.2 Pracovní paměť (CHANDATA, COMPLETE, INITIAL)

Funkce

Pracovní paměť obsahuje aktuální systémová a uživatelská data, se kterými systém pracuje (aktivní systém souborů), např.:

- Aktivní strojní parametry
- Korekční parametry nástroje
- Posunutí počátku
- ...

Inicializační programy

V této souvislosti se jedná o programy, které zajišťují dosazování předdefinovaných hodnot pro data v pracovní paměti (inicializace). Za tím účelem se používají následující typy dat:

Typ souboru	Popis
název_TEA	Strojní parametry
název_SEA	Nastavované parametry
název_TOA	Korekční parametry nástroje
název_UFR	Posunutí počátku / frame
název_INI	Inicializační soubory
název_GUD	Globální uživatelská data
název_RPA	R-Parametry

Informace o všech těchto typech dat naleznete v v návodu k obsluze uživatelského rozhraní.

Datové oblasti

Data mohou být rozčleněna do různých oblastí, v nichž mají platit. Řídící systém může například disponovat větším počtem kanálů nebo může být k dispozici více os, což je obvyklé.

Existují:

Označení	Datové oblasti
NCK	Specifická data NCK
CH<n>	Data specifického kanálu (<n> udává číslo kanálu)
AX<n>	Data specifické osy (<n> udává číslo osy stroje)
TO	Parametry nástroje
COMPLETE	Všechna data

Založení inicializačního programu na externím PC

Prostřednictvím identifikace datové oblasti a identifikace datového typu mohou být definovány oblasti, které jsou při ukládání dat považovány za jednu jednotku.

_N_AX5_TEA_INI	Strojní parametry pro osu 5
_N_CH2_UFR_INI	Framy kanálu 2
_N_COMPLETE_TEA_INI	Všechny strojní parametry

Po uvedení řídícího systému do provozu je v pracovní paměti k dispozici datový blok, který zaručuje, že řídící systém pracuje správně.

Postup v případě řídících systémů s více kanály (CHANDATA)

Příkaz `CHANDATA(<číslo kanálu>)` pro větší počet kanálů je přípustný pouze v souboru _N_INITIAL_INI. Jedná se o inicializační soubor, pomocí kterého se uskutečňuje inicializace všech dat v řídícím systému.

Programový kód	**Komentář**
`%_N_INITIAL_INI`	
`CHANDATA(1)`	
	`; Přiřazení os stroje kanálu 1:`
`$MC_AXCONF_MACHAX_USED[0]=1`	
`$MC_AXCONF_MACHAX_USED[1]=2`	
`$MC_AXCONF_MACHAX_USED[2]=3`	
`CHANDATA(2)`	
	`; Přiřazení os stroje kanálu 2:`
`$MC_AXCONF_MACHAX_USED[0]=4`	
`$MC_AXCONF_MACHAX_USED[1]=5`	
`CHANDATA(1)`	
	`; Osové strojní parametry:`
	`; Okno přesného najetí hrubé:`
`$MA_STOP_LIMIT_COARSE[AX1]=0.2`	`; Osa 1`
`$MA_STOP_LIMIT_COARSE[AX2]=0.2`	`; Osa 2`
	`; Okno přesného najetí jemné:`
`$MA_STOP_LIMIT_FINE[AX1]=0.01`	`; Osa 1`
`$MA_STOP_LIMIT_FINE[AX1]=0.01`	`; Osa 2`

⚠ POZOR

Příkaz CHANDATA

Ve výrobním programu smí být příkaz `CHANDATA` použit pouze v kanálu, v němž se daný NC program zpracovává, tzn. tento příkaz se může používat také pro ochranu NC programů, aby nemohly být zpracovávány v kanálu, pro který nejsou určeny.

Pokud se vyskytne chyba, zpracování programu se přeruší.

Poznámka

Soubory INI v seznamu úloh neobsahují žádné příkazy `CHANDATA`.

Ukládání inicializačních programů (COMPLETE, INITIAL)

Soubory z pracovní paměti se mohou ukládat také na externí PC, odkud je možné je opět načíst.

- Soubory se ukládají pomocí příkazu `COMPLETE`.
- Pomocí příkazu `INITIAL` se vytvoří soubor INI ((_N_INITIAL_INI) zahrnující všechny oblasti.

Načtení inicializačního programu

UPOZORNĚNÍ
Pokud je načten soubor s názvem "INITIAL_INI", potom jsou všechna data, do nichž tento soubor nedosazuje žádné hodnoty, inicializována standardními hodnotami. Výjimku z tohoto pravidla představují pouze strojní parametry. Standardní hodnoty (za normálních okolností "NULA" jsou dosazovány také do **nastavovaných parametrů, parametrů nástrojů, posunutí počátku, hodnot GUD, ...**.

Pro načítání jednotlivých strojních parametrů se hodí např. soubor COMPLETE_TEA_INI. V tomto souboru řídící systém očekává strojní parametry. V tomto případě díky tomu zůstávají ostatní datové oblasti nedotčeny.

Načítání inicializačního programu

Pokud programy INI využívají pouze data určitého kanálu, mohou být zvoleny a vyvolány stejně jako výrobní programy. Takto je tedy možné inicializovat také data řízená programem.

2.3 Strukturovací příkaz ve Stepeditoru (SEFORM)

Funkce

Strukturovací příkaz `SEFORM` se ve Stepeditoru (programová podpora založená na editoru) vyhodnocuje, aby z něj bylo možné vygenerovat výpis kroků pro HMI Advanced. Zobrazení ve formě kroků slouží ke zlepšení čitelnosti NC podprogramu.

Syntaxe

```
SEFORM(<název úseku>,<úroveň>,<ikona>)
```

Význam

`SEFORM()`	Vyvolání funkce strukturovacího příkazu s parametry `<název úseku>`, `<úroveň>` a `<ikona>`		
`<název úseku>`	Identifikátor kroku pracovního postupu		
	Typ:	STRING	
`<úroveň>`	Index úrovně hlavního programu nebo podprogramu		
	Typ:	INT	
	Hodnota:	0	Hlavní úroveň
		1, ..., <n>	Úroveň podprogramu 1, ... , úroveň podprogramu <n>
`<ikona>`	Název ikony, která se má pro tento úsek zobrazovat.		
	Typ:	STRING	

Poznámka

Příkazy `SEFORM` jsou generovány ve Stepeditoru.

Řetězec znaků předávaný pomocí parametru `<název úseku>` se analogicky s příkazem `MSG` ukládá synchronně se zpracováním v hlavní větvi do proměnné BTSS. Informace zde zůstávají až do jejich přepsání následujícím příkazem `SEFORM`. Reset a konec výrobního programu způsobí vymazání obsahu této proměnné.

Parametry `<úroveň>` a `<ikona>` se prostřednictvím NCK při zpracovávání výrobního programu kontrolují, dále už zpracovávány nejsou.

Literatura

Pokud budete potřebovat další informace k programové podpoře založené na editoru, viz: Návod k obsluze systému HMI Advanced

Chráněné oblasti

3

3.1 Stanovení chráněné oblasti (CPROTDEF, NPROTDEF)

Funkce

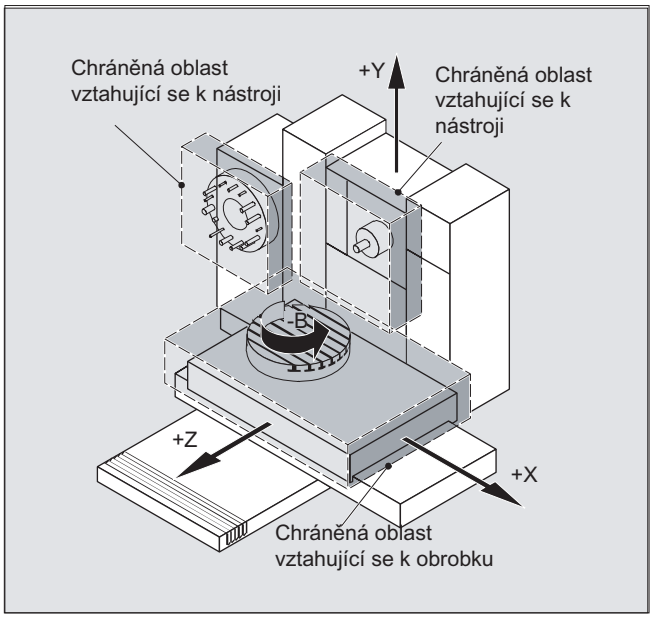
Pomocí chráněných oblastí můžete zajistit ochranu různých prvků na stroji, jeho vybavení, ale i na opracovávaném obrobku před nesprávnými pohyby os.

Chráněné oblasti **vztahující se k nástroji**:

Pro součásti, které patří k nástroji (např. nástroj, držák nástroje).

Chráněné oblasti **vztahující se k obrobku**:

Pro součásti, které patří k obrobku (např. části obrobku, upínací stůl, upínací čelisti, vřetenové sklíčidlo, koník).



Syntaxe

```
DEF INT NOT_USED
G17/G18/G19
CPROTDEF/NPROTDEF(<n>,<t>,<applim>,<applus>,<appminus>)
G0/G1/... X/Y/Z...
...
EXECUTE(NOT_USED)
```

Význam

DEF INT NOT_USED:	Definice lokální proměnné, datový typ INTEGER srov. kapitola "Pohybové synchronní akce [Strana 557]")
G17/G18/G19:	Požadovaná rovina je vybrána před příkazem CPROTDEF, příp. NPROTDEF pomocí příkazu G17/G18/G19 a před příkazem EXECUTE nesmí být změněna. Programování aplikáty mezi příkazy CPROTDEF příp. NPROTDEF a EXECUTE je rovněž nepřípustné.
CPROTDEF:	Definice kanálové chráněné oblasti (pouze v systému NCU 572/573).
NPROTDEF:	Definice strojní chráněné oblasti.
G0/G1/... X/Y/Z... ...:	Kontura chráněné oblasti se zadává s maximálně 11 posuvovými pohyby ve zvolené rovině. První posuvový pohyb je přitom pohyb na konturu. Jako chráněná platí přitom oblast vlevo od kontury. **Upozornění:** Posuvové pohyby nacházející se mezi příkazy CPROTDEF příp. NPROTDEF a EXECUTE se přitom neuskutečňují, nýbrž definují chráněnou oblast..
EXECUTE:	Ukončení definice
<n>:	Číslo definované chráněné oblasti
<t>:	Typ chráněné oblasti TRUE: Chráněná oblast vztahující se k **nástroji**: FALSE: Chráněná oblast vztahující se k **obrobku**:
<applim>:	Druh ohraničení ve 3. rozměru 0: žádné ohraničení 1: Ohraničení v kladném směru 2: Ohraničení v záporném směru 3: Ohraničení v kladném a záporném směru
<applus>:	Hodnota ohraničení v kladném směru ve 3. rozměru
<appminus>:	Hodnota ohraničení v záporném směru ve 3. rozměru
NOT_USED:	Chybová proměnná nemá u chráněných oblastí s příkazem EXECUTE žádný význam.

Okrajové podmínky

V průběhu definice chráněných oblastí:

- Nesmí být aktivní žádná korekce rádiusu frézy, příp. korekce rádiusu břitu nástroje.
- Nesmí být aktivní žádná transformace.
- Nesmí být aktivní žádný frame.

Nesmí být naprogramováno také žádné najíždění na referenční bod (G74), najíždění na pevný bod (G75), zastavení vyhledávání bloku nebo konec programu.

Další informace

Definice chráněných oblastí

K definici chráněných oblastí patří:

- příkaz `CPROTDEF` pro kanálové chráněné oblasti
- Příkaz `NPROTDEF` pro strojní chráněné oblasti
- Popis kontury chráněné oblasti
- Ukončení definice pomocí příkazu `EXECUTE`

Při aktivování chráněné oblasti ve výrobním programu NC systému můžete vztažný bod chráněné oblasti relativně posunovat.

Vztažný bod popisu kontury

Chráněné oblasti vztažené na obrobek jsou definovány v základním souřadném systému.

Chráněné oblasti vztažené k nástroji jsou udávány vůči vztažnému bodu držáku nástroje F.

Přípustné konturové prvky

Pro popis kontury chráněné oblasti jsou přípustné:

- příkazy `G0`, `G1` pro přímé konturové prvky
- `G2` pro kruhové oblouky ve směru hodinových ručiček (pouze pro chráněné oblasti vztažené k obrobku)
- `G3` pro kruhové oblouky proti směru hodinových ručiček

Poznámka

Jestliže má chráněnou oblast popisovat plná kružnice, je nutno ji rozdělit na dva kruhové oblouky. Posloupnost `G2`, `G3` příp. `G3`, `G2` je nepřípustná. V případě potřeby je zde nutno vložit krátký úsek `G1`.

Poslední bod popisu kontury se nesmí krýt s pevním bodem.

Vnější chráněné oblasti

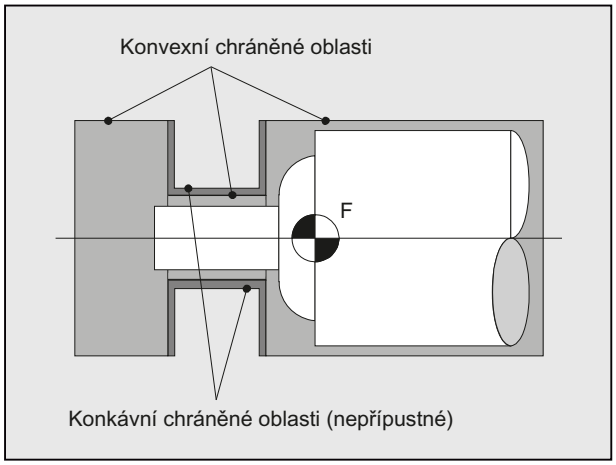
Vnější chráněné oblasti (možné pouze v případě chráněných oblastí vztahujících se na obrobek) je potřeba definovat ve směru hodinových ručiček.

Rotačně symetrické chráněné oblasti

U rotačně symetrických chráněných oblastí (např. vřetenové sklíčidlo) musí být popsána celá kontura (nejen po osu otáčení).

Chráněné oblasti vztahující se k nástroji

Chráněné oblasti vztahující se k nástroji musí být vždy konvexní. Jestliže je požadována konkávní chráněná oblast, je potřeba ji rozložit na několik konvexních chráněných oblastí.



3.2 Aktivování/deaktivování chráněné oblasti (CPROT, NPROT)

Funkce

Aktivování dříve definovaných chráněných oblastí pro protikolizní monitorování, deaktivování předběžně aktivovaných nebo aktivních chráněných oblastí.

Maximální počet chráněných oblastí, které jsou v dané chvíli aktivní v jednom kanálu, se definuje pomocí strojního parametru.

Jestliže není aktivní žádná chráněná oblast vztahující se k nástroji, kontroluje se dráha nástroje vzhledem ke chráněné oblasti vztahující se k obrobku.

Poznámka

Jestliže není aktivní žádná chráněná oblast vztahující se k obrobku, žádné monitorování chráněných oblastí se neuskutečňuje.

Syntaxe

```
CPROT(<n>,<stav>,<xMov>,<yMov>,<zMov>)
NPROT(<n>,<stav>,<xMov>,<yMov>,<zMov>)
```

Význam

`CPROT`:	Vyvolání kanálové chráněné oblasti (pouze v systému NCU 572/573).
`NPROT`:	Vyvolání specifické chráněné oblasti stroje
`<n>`:	Číslo chráněné oblasti
`<stav>`:	Informace o stavu 0: Deaktivovat chráněnou oblast 1: Chráněnou oblast předběžně aktivovat 2: Aktivovat chráněnou oblast 3: Předběžné aktivování chráněné oblasti s podmíněným zastavením
`<xMov>,<yMov>,<zMov>`:	Posunutí již definované chráněné oblasti po geometrických osách

Okrajové podmínky

Monitorování chráněné oblasti v případě aktivní korekce rádiusu nástroje

Pokud je korekce rádiusu nástroje aktivní, je funkční monitorování chráněných oblastí možné jen tehdy, pokud je rovina korekce rádiusu nástroje identická s rovinou, ve které je definována chráněná oblast.

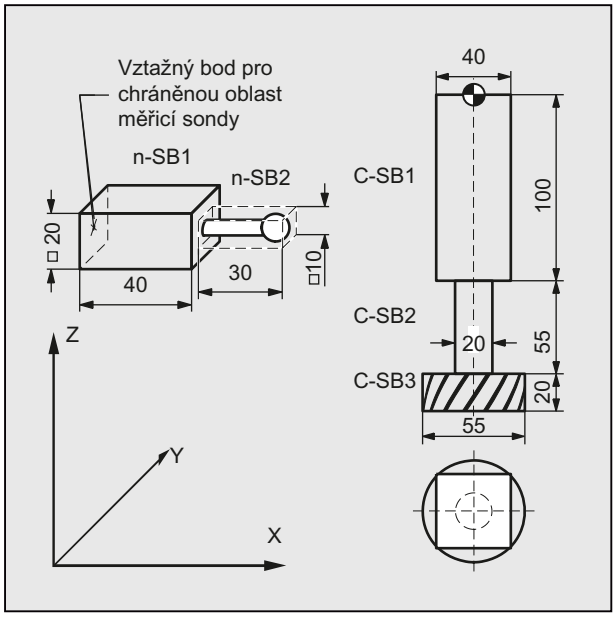
Příklad

V případě frézky mají být monitorovány možné kolize frézy s měřicí sondou. Délka měřicí sondy má být při svém aktivování zadána prostřednictvím posunutí. Pro tento účel jsou definovány následující chráněné oblasti:

- Specifická chráněná oblast daného stroje vztažená na obrobek pro držák měřicí sondy (n-SB1) a pro měřicí sondu samotnou (n-SB2).
- Chráněná oblast daného kanálu vztažená k nástroji pro držák frézy (c-SB1), stopku frézy (c-SB2) a frézu samotnou (c-SB3).

Potvrzení všech chráněných oblastí je ve směru osy Z.

Souřadnice polohy vztažného bodu měřicí sondy při aktivování mají být X = -120, Y = 60 a Z = 80.



Programový kód	**Komentář**
`DEF INT SCHUTZB`	`; Definice pomocné proměnné`
`Definice chráněné oblasti - G17`	`; Nastavení potvrzování`

`NPROTDEF(1,FALSE,3,10,-10)G01 X0 Y-10`
`X40`
`Y10`
`X0`
`Y-10`
`EXECUTE(SCHUTZB)`
	`; Chráněná oblast n-SB1`

`NPROTDEF(2,FALSE,3,5,-5)`
`G01 X40 Y-5`
`X70`
`Y5`
`X40`
`Y-5`
`EXECUTE(SCHUTZB)`
	`; Chráněná oblast n-SB2`

Programový kód	**Komentář**
CPROTDEF(1,TRUE,3,0,-100) G01 X-20 Y-20 X20 Y20 X-20 Y-20 EXECUTE(SCHUTZB)	; Chráněná oblast c-SB1
CPROTDEF(2,TRUE,3,-100,-150) G01 X0 Y-10 G03 X0 Y10 J10 X0 Y-10 J-10 EXECUTE(SCHUTZB)	; Chráněná oblast c-SB2
CPROTDEF(3,TRUE,3,-150,-170) G01 X0 Y-27,5 G03 X0 Y27,5 J27,5 X0 Y27,5 J-27,5 EXECUTE(SCHUTZB)	; Chráněná oblast c-SB3
Aktivování chráněných oblastí:	
NPROT(1,2,-120,60,80)	; Chráněnou oblast n-SB1 aktivovat s posunutím
NPROT(2,2,-120,60,80)	; Chráněnou oblast n-SB2 aktivovat s posunutím
CPROT(1,2,0,0,0)	; Chráněnou oblast c-SB1 aktivovat s posunutím
CPROT(2,2,0,0,0)	; Chráněnou oblast c-SB2 aktivovat s posunutím
CPROT(3,2,0,0,0)	; Chráněnou oblast c-SB3 aktivovat s posunutím

Další informace

Stav aktivování (<stav>)

- **<stav>=2**

  Chráněná oblast je obecně ve výrobním programu aktivována, když má stav = 2.

  Tento stav je vždy vztažen ke kanálu, a to i u chráněných oblastí vztažených ke stroji.

- **<stav>=1**

  Jestliže se prostřednictvím uživatelského programu PLC předpokládá, že chráněná oblast může mát nastavena jako platná pomocí uživatelského programu PLC, provádí se potřebné předběžné aktivování pomocí stavu = 1.

- **<stav>=3**

  Při předběžném aktivování s podmíněným zastavením v zásadě nedochází před narušením chráněné oblasti s předběžnou aktivací k zastavení. K zastavení dojde jen tehdy, pokud byla chráněná oblast nastavena jako platná. To umožňuje nerušené obrábění, pokud jsou chráněné oblasti nastavovány jako platné jen pro určité případy. Je potřeba mít na paměti, že v důsledku brzdné charakteristiky se za určitých okolností najede do chráněné oblasti, jestliže je chráněná oblast nastavena jako platná bezprostředně před najetím do ní.

  Předběžné aktivování s podmíněným zastavením se nastavuje pomocí stavu = 3.

- **<stav>=0**

  Deaktivování a v důsledku toho vypnutí chráněné oblasti se uskutečňuje pomocí stavu = 0. Přitom není zapotřebí žádné posunutí.

Posunutí chráněné oblasti při (předběžné) aktivaci

Posunutí může být provedeno v 1, 2 nebo ve 3 směrech. Údaje posunutí jsou vztaženy na:

- počátek souřadného systému stroje v případě chráněných oblastí vztažených k obrobku
- vztažný bod držáku nástroje F v případě chráněných oblastí vztažených k nástroji

Stav po náběhu systému

Chráněné oblasti mohou být aktivovány již po náběhu systému a po následném najíždění na referenční bod. Za tím účelem musí být systémová proměnná $SN_PA_ACTIV_IMMED[<n>] příp. $SC_PA_ACTIV_IMMED[<n>] nastavena na hodnotu TRUE. Oblasti se vždy aktivují se stavem = 2 a nemají žádné posunutí.

Vícenásobné aktivování chráněných oblastí

Chráněná oblast může být v platnosti i ve více kanálech současně (např. hrotová objímka v případě dvou proti sobě umístěných saní). Monitorování chráněných oblasti se uskutečňuje jen tehdy, pokud bylo všemi geometrickými osami najeto na referenční bod.

Přitom platí:

- Chráněná oblast nemůže být v jednom kanálu aktivována vícekrát pokaždé s jinou hodnotou posunutí.
- Chráněné oblasti vztažené ke stroji musí mít v obou kanálech stejnou orientaci.

3.3 Kontrola narušení chráněné oblasti, ohraničení pracovního pole a softwarových koncových spínačů (CALCPOSI)

Funkce

Funkce CALPOSI umožňuje přezkoumat, jestli když se vyjde z nějakého předem známého počátečního bodu, mohou geometrické osy objet požadovanou dráhu, aniž by došlo k narušení hranic os (softwarové koncové spínače), ohraničení pracovního pole nebo chráněných oblastí.

Pro případ, že požadovanou dráhu nelze bez narušení hranic objet, je výsledkem maximální přípustná hodnota.

Funkce CALCPOSI je předem definovaným podprogramem. Musí se nacházet v samostatném bloku.

Syntaxe

```
Status=CALCPOSI(_STARTPOS, _MOVDIST, _DLIMIT, _MAXDIST, _BASE_SYS,
_TESTLIM)
```

Význam

`Status`

0: Funkce je OK,
předem definovanou dráhu je možné úplně objet.

–: V proměnné _DLIMIT je nejméně jedna složka záporná

–: Ve výpočtu transformace se vyskytla chyba.

Jestliže zadanou dráhu není možné úplně objet, je výsledkem kladná hodnota s desítkovým kódováním:

Místo jednotek (druh narušené hranice):

1: Dráha je omezena softwarovými koncovými spínači.

2: Dráha je omezena ohraničením pracovního pole.

3: Dráha je omezena chráněnými oblastmi.

Jestliže je narušeno více hranic současně (např. softwarové koncové spínače a chráněné oblasti), bude se na místě jednotek uvádět hranice, která způsobuje silnější omezení předem zadané dráhy.

Místo desítek

10:

Počáteční hodnota narušuje hranici

20:

Předem zadaná přímka narušuje hranici. Tato hodnota je výsledkem také tehdy, pokud samotný koncový bod žádnou hranici nenarušuje, ale na dráze z počátečního do koncového bodu k narušení některé z mezních hodnot došlo (např. průjezd chráněnou oblastí, zakřivené polohy softwarových koncových spínačů ve WCS kvůli nelineárním transformacím, např. Transmit).

Místo stovek

100:

Došlo k narušení kladné mezní hodnoty (jen když je na místě jednotek 1 nebo 2, tzn. v případě softwarových koncových spínačů a ohraničení pracovního pole)

100:

Je narušena chráněná oblast NCK (jen když je na místě jednotek 3)

200:

Došlo k narušení záporné mezní hodnoty (jen když je na místě jednotek 1 nebo 2, tzn. v případě softwarových koncových spínačů a ohraničení pracovního pole)

200:

Je narušena chráněná oblast specifického kanálu (jen když je na místě jednotek 3)

Místo tisíců

1000:

Faktor, kterým je vynásobeno číslo osy, která způsobila narušení hranice (jen když je na místě jednotek 1 nebo 2, tzn. v případě softwarových koncových spínačů a ohraničení pracovního pole)

Počítání os začíná 1 a v případě narušení softwarových koncových spínačů (místo jednotek = 1) se vztahuje na osy stroje, zatímco v případě narušení ohraničení pracovního pole (místo jednotek = 2) na geometrické osy.

1000:

Faktor, se kterým je vynásobeno číslo narušené chráněné oblasti (jen když je na místě jednotek 3)

Jestliže je narušeno několik chráněných oblastí, na místě stovek a tisíců je oznámena chráněná oblast, která způsobuje nejsilnější omezení předem zadané dráhy.

`_STARTPOS` Počáteční hodnoty pro abscisu [0], ordinátu[1] a aplikátu [2] ve WCS.

`_MOVEDIST` Inkrementální zadání dráhy pro abscisu [0], ordinátu[1] a aplikátu [2].

`_DLIMIT` [0] - [2]: Minimální vzdálenost, která je přiřazena geometrickým osám.

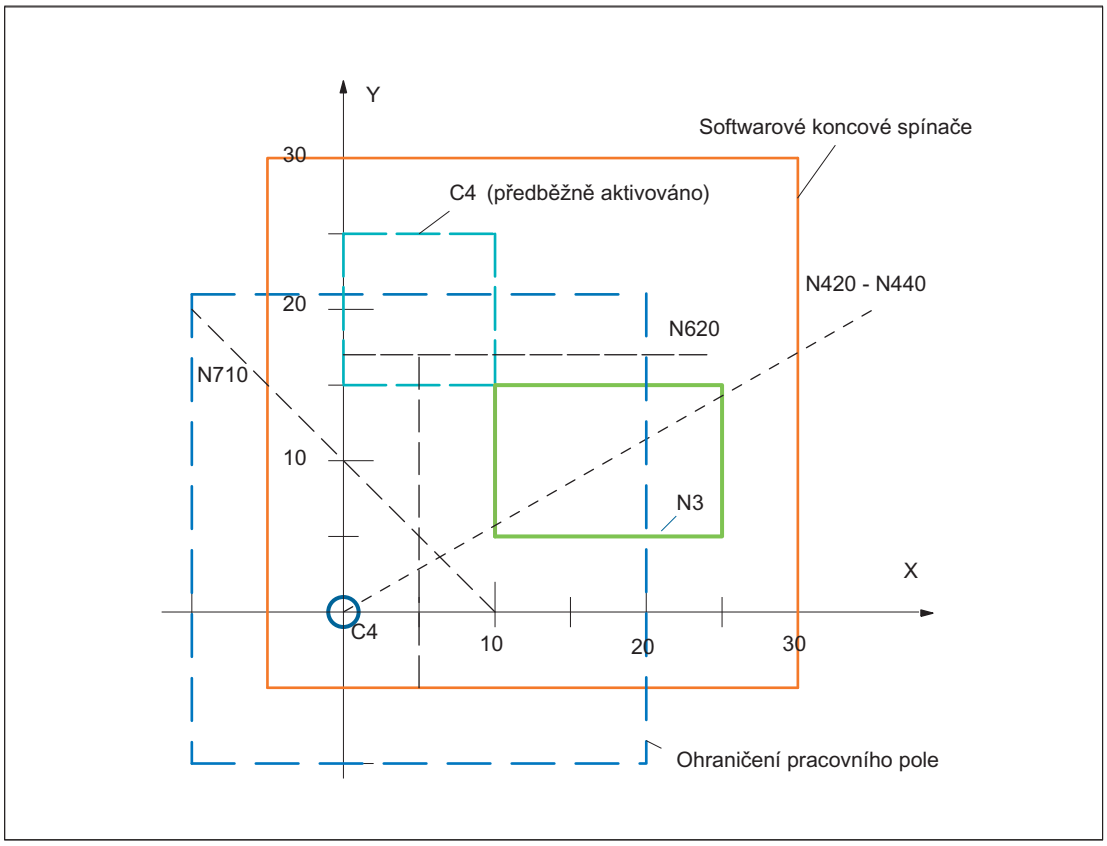
[3]: Minimální vzdálenost, která se přiřazuje lineární ose stroje v případě nelineární transformace, jestliže není možné jednoznačně přiřadit žádnou geometrickou osu.

[4]: Minimální vzdálenost, která se přiřazuje kruhové ose stroje v případě nelineární transformace, jestliže není možné jednoznačně přiřadit žádnou geometrickou osu. Jen v případě speciálních transformací, když se mají monitorovat softwarové koncové spínače.

_MAXDIST	Pole [0] - [2] pro výslednou hodnotu. Inkrementální dráha u všech tří geometrických os, aniž by se překročila zadaná minimální vzdálenost od hranice osy u všech os stroje, které se na operaci podílejí. Pokud dráha není nijak omezena, je obsah tohoto výsledného parametru roven obsahu parametru _MOVDIST.
_BASE_SYS	FALSE nebo parametr není uveden: Při vyhodnocování údajů poloh a délek jsou vyhodnocovány také G-kódy skupiny 13 (G70, G71, G700, G710; palce/ metrické jednotky). Když je aktivní příkaz G70 a základní systém je metrický (příp. když je aktivní příkaz G71 a palce), jsou systémové proměnné WCS $AA_IW[X] a $AA_MW[X]) udávány v základním systému a v případě nutnosti musí být přepočítány, aby mohly být funkcí CALCPOSI použity. TRUE: Při vyhodnocování údajů poloh a délek je vždy používán základní systém řídícího systému bez ohledu na hodnotu aktivní G-funkce ze skupiny 13.
_TESTLIM	Hranice, které se mají kontrolovat (binárně kódované): 1: Monitorování softwarových koncových spínačů 2: Monitorování ohraničení pracovního pole 3: Monitorování aktivovaných chráněných oblastí 4: Monitorování předběžně aktivovaných chráněných oblastí Kombinace se provedou sečítáním hodnot. Předdefinovaná hodnota: 15, všechno kontrolovat.

Příklad

V příkladu (viz obrázek) jsou nakresleny softwarové koncové spínače a ohraničení pracovního pole pro osu X. Kromě toho jsou definovány tři chráněné oblasti, dvojice kanálových chráněných oblastí C2 a C4, jakož i chráněná oblast NCK N3. C2 je kruhová aktivní chráněná oblast s rádiusem 2 mm vztažená na nástroj. C4 je kvadratická, předběžně aktivovaná chráněná oblast vztažená na obrobek s délkou strany 10 mm a N3 je obdélníková aktivní chráněná oblast s délkami stran 10 mm, příp. 15 mm. V následujícím NC programu jsou napřed definovány chráněné oblasti a ohraničení pracovního pole podle nákresu a potom je vyvolána funkce CALCPOSI s různými nastaveními parametrů. Výsledky jednotlivých volání funkce CALCPOSI jsou shrnuty v tabulce na konci příkladu.



Programový kód	**Komentář**
N10 def real _STARTPOS[3]	
N20 def real _MOVDIST[3]	
N30 def real _DLIMIT[5]	
N40 def real _MAXDIST[3]	
N50 def int _SB	
N60 def int _STATUS	
N70 cprotdef(2, true, 0)	; Chráněná oblast vztahující se k nástroji
N80 g17 g1 x-y0	
N90 g3 i2 x2	
N100 i-x-	
N110 execute(_SB)	
N120 cprotdef(4, false, 0)	; Chráněná oblast vztahující se k obrobku
N130 g17 g1 x0 y15	
N140 x10	
N150 y25	
N160 x0	
N170 y15	
N180 execute(_SB)	

Programový kód	**Komentář**
N190 nprotdef(3, false, 0)	; Chráněná oblast vztažená ke stroji
N200 g17 g1 x10 y5	
N210 x25	
N220 y15	
N230 x10	
N240 y5	
N250 execute(_SB)	
N260 cprot(2,2,0, 0, 0)	; Aktivování, příp. předběžné aktivování chráněných oblastí
N270 cprot(4,1,0, 0, 0)	
N280 nprot(3,2,0, 0, 0)	
N290 g25 XX=-YY=-	; Definice ohraničení pracovního pole
N300 g26 xx= 20 yy= 21	
N310 _STARTPOS[0] = 0.	
N320 _STARTPOS[1] = 0.	
N330 _STARTPOS[2] = 0.	
N340 _MOVDIST[0] = 35.	
N350 _MOVDIST[1] = 20.	
N360 _MOVDIST[2] = 0.	
N370 _DLIMIT[0] = 0.	
N380 _DLIMIT[1] = 0.	
N390 _DLIMIT[2] = 0.	
N400 _DLIMIT[3] = 0.	
N410 _DLIMIT[4] = 0.	
;Volání různých funkcí	; Jiný počáteční bod
N420 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST)	
N430 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,,3)	
N440 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,,1)	
N450 _STARTPOS[0] = 5.	; Jiný cíl
N460 _STARTPOS[1] = 17.	
N470 _STARTPOS[2] = 0.	
N480 _MOVDIST[0] = 0.	
N490 _MOVDIST[1] =-.	
N500 _MOVDIST[2] = 0.	
;Volání různých funkcí	
N510 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,,14)	
N520 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,, 6)	

Programový kód	**Komentář**
N530 _DLIMIT[1] = 2.	
N540 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,, 6)	
N550 _STARTPOS[0] = 27.	
N560 _STARTPOS[1] = 17.1	
N570 _STARTPOS[2] = 0.	
N580 _MOVDIST[0] =-.	
N590 _MOVDIST[1] = 0.	
N600 _MOVDIST[2] = 0.	
N610 _DLIMIT[3] = 2.	
N620 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,, 12)	
N630 _STARTPOS[0] = 0.	
N640 _STARTPOS[1] = 0.	
N650 _STARTPOS[2] = 0.	
N660 _MOVDIST[0] = 0.	
N670 _MOVDIST[1] = 30.	
N680 _MOVDIST[2] = 0.	
N690 trans x10	
N700 arot z45	
N710 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST)	
N720 M30	

Výsledky kontrol v příkladu:

Č. bloku N...	_STATUS	_MAXDIST [0] (= X)	_MAXDIST [1] (= Y)	Poznámky
420	3123	8.040	4.594	Chráněná oblast SB N3 se naruší.
430	1122	20.000	11.429	Žádné monitorování chráněných oblastí, narušení ohraničení pracovního pole.
440	1121	30.000	17.143	Je aktivní ještě monitorování softwarových koncových spínačů.
510	4213	0.000	0.000	Počáteční bod narušuje SB C4.
520	0000	0.000	–.000	Předběžně aktivovaná SB C4 není monitorována. Zadanou dráhu je možné objet úplně celou.
540	2222	0.000	–.000	Kvůli nastavení _DLIMIT[1]=2 je dráha omezena kvůli ohraničení pracovního pole..
620	4223	–.000	0.000	Vzdálenost k C4 je kvůli C2 a parametru _DLIMIT[3] celkem 4 mm. Vzdálenost C2 - N3 o hodnotě 0,1 mm nezpůsobuje omezení dráhy pohybu.
710	1221	0.000	21.213	Je aktivní frame se složkou posunu a otočení. Přípustná dráha pohybu v parametru _MOVDIST platí v posunutém a otočeném souřadném systému (WCS).

Zvláštní případy a další podrobnosti

Všechny údaje dráhy jsou vždy hodnotami rádiusu, i když je pro příčnou osu aktivována G-funkce "DIAMON". Jestliže dráha některé osy podílející se na pohybu nemůže být absolvována celá, jsou ve výsledné hodnotě v parametru _MAXDIST odpovídajícím způsobem zkráceny také dráhy ostatních os, takže výsledný koncový bod leží na předem zadané dráze.

Je přípustné, aby pro jednu nebo více od podílejících se na pohybu nebyly definovány žádné softwarové koncové spínače, příp. ohraničení pracovního pole nebo chráněné oblasti. Všechny hranice jsou monitorovány jen tehdy, když bylo osami podílejícími se na pohybu najeto na referenční bod Případně se podílející kruhové osy jsou monitorovány jen tehdy, pokud se nejedná o osy typu modulo.

Monitorování softwarových koncových spínačů a ohraničení pracovního pole probíhá stejně jako v normálním režimu posuvu v závislosti na aktivních nastaveních (signály rozhraní pro aktivování softwarového koncového spínače 1, příp. softwarového koncového spínače 2, funkce WALIMON/WALIMOF, nastavované parametry pro individuální aktivování ohraničení pracovního pole a pro stanovení, zda se při monitorování ohraničení pracovního pole má nebo nemá zohledňovat rádius aktivního nástroje).

Při určitých kinematických transformacích (např. TRANSMIT) se stát, že poloha os stroje nemusí být jednoznačně určena z poloh v souřadném systému obrobku (WCS) (víceznačnost). Při normálních posuvech vyplývá jednoznačnost zpravidla z předcházejícího dění a z podmínky, že spojitý pohyb ve WCS musí odpovídat spojitému pohybu os stroje. Při monitorování softwarových koncových spínačů pomocí funkce CALCPOSI se proto v případech tohoto typu připojuje aktuální poloha stroje, aby se víceznačnost odstranila. V případě potřeby proto musí být před příkazem CALCPOSI naprogramován příkaz **STOPRE**, aby do funkce bylo možno dosadit platné pozice os stroje.

Není zaručeno, že při pohybu po zadané dráze zůstane vzdálenost k chráněným oblastem specifikovaná v parametru _DLIMIT[3] k chráněným oblastem vůbec zachována. Proto nemůže být při prodloužení výsledného koncového bodu v parametru _MOVDIST o tuto vzdálenost žádná chráněná oblast narušena. Přímka se ale může na své dráze dostat libovolně blízko k chráněné oblasti.

Poznámka

Pokud budete potřebovat podrobnosti týkající se ohraničení pracovního pole, viz /PG/, Příručka programování, Základy,

a týkající se softwarových koncových spínačů, viz /FB1/, Příručka Popis funkcí, Základní funkce; Monitorování os, chráněné oblasti (A3).

Speciální příkazy dráhy 4

4.1 Najíždění na kódované pozice (CAC, CIC, CDC, CACP, CACN)

Funkce

Pomocí následujících příkazů můžete lineárními a kruhovými osami pomocí čísel pozic najíždět na pevné polohy os uložené v tabulkách strojních parametrů. Tento druh programování je označován jako "Najíždění na kódované pozice".

Syntaxe

```
CAC(<n>)
CIC(<n>)
CACP(<n>)
CACN(<n>)
```

Význam

`CAC(<n>)`	Najíždění na kódovanou pozici s číslem polohy n.
`CIC(<n>)`	Najíždění na kódovanou pozici, která je uvedena o n míst pozic před (+n) nebo za (-n) od aktuálního čísla pozice.
`CDC(<n>)`	Najíždění na kódovanou pozici s číslem polohy n po nejkratší dráze (jen pro kruhové osy)
`CACP(<n>)`	Najíždění na kódovanou pozici s číslem polohy n v kladném směru (jen pro kruhové osy)
`CACN(<n>)`	Najíždění na kódovanou pozici s číslem polohy n v záporném směru (jen pro kruhové osy)
`<n>`	Číslo pozice v tabulce strojních parametrů Rozsah hodnot: 0, 1, … (max. počet míst v tabulce - 1)

Příklad: Najíždění polohovací osou na kódované pozice

Programový kód	Komentář
`N10 FA[B]=300`	`; Posuv pro polohovací osu B`
`N20 POS[B]=CAC(10)`	`; Najíždění na kódovanou pozici s číslem polohy 10.`
`N30 POS[B]=CIC(-4)`	`; Najíždění na kódovanou pozici s číslem "aktuální číslo polohy" - 4.`

Literatura

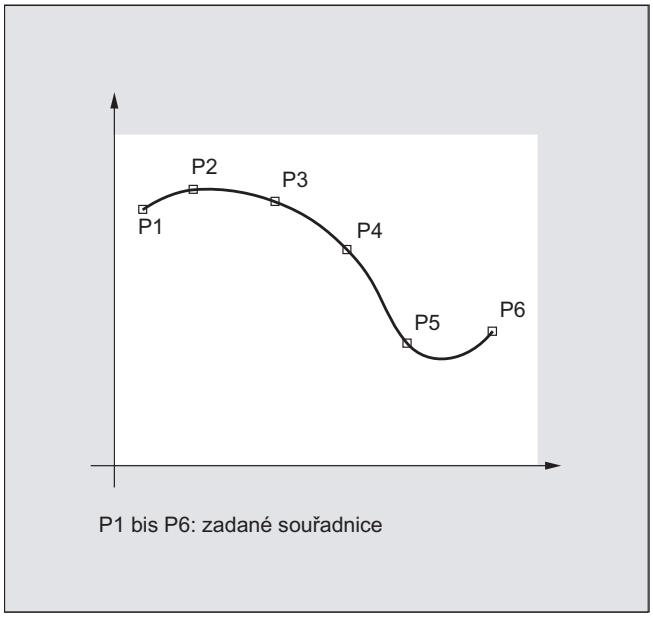
- Příručka Popis funkcí, Rozšiřovací funkce; Osy s dělením (T1)
- Příručka Popis funkcí, Synchronní akce

4.2 Splinová interpolace (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL)

Funkce

Libovolně zakřivené kontury na obrobku nemusí být analyticky popsány úplně přesně. Kontury tohoto druhu jsou proto aproximovány pomocí omezeného počtu uzlových bodů, např. při digitalizaci povrchů. Za účelem vytvoření digitalizovaného povrchu na obrobku musí být uzlové body spojeny, takže vznikne popsaná kontura. To umožňuje splinová interpolace.

Spline definuje křivku, která se skládá z polynomů 2. nebo 3. stupně. Vlastnosti na uzlových bodech splinu mohou být definovány **v závislosti na použitém typu splinu**.



P1 bis P6: zadané souřadnice

U systému SINUMERIK solution line jsou k dispozici následující typy splinů:

- A-Spline
- B-Spline
- C-Spline

Syntaxe

Všeobecně:

```
ASPLINE X... Y... Z... A... B... C...
BSPLINE X... Y... Z... A... B... C...
CSPLINE X... Y... Z... A... B... C...
```

U B-splinů může být ještě navíc naprogramováno:

```
PW=<n>
SD=2
PL=<hodnota>
```

U A- a C-splinů může být ještě navíc naprogramováno:

`BAUTO` / `BNAT` / `BTAN`

`EAUTO` / `ENAT` / `ETAN`

Význam

Typ splinové interpolace:

`ASPLINE`	Příkaz pro aktivování interpolace pomocí A-splinů
`BSPLINE`	Příkaz pro aktivování interpolace pomocí B-splinů
`CSPLINE`	Příkaz pro aktivování interpolace pomocí C-splinů
	Příkazy `ASPLINE`, `BSPLINE` a `CSPLINE` mají modální platnost a patří do skupiny příkazů dráhy.

Uzlové, příp. řídící body:

`X... Y... Z...` `A... B... C...`	Polohy v kartézských souřadnicích.

Váha bodu (jen B-spliny):

`PW`	Pomocí příkazu `PW` je možno pro každý uzlový bod naprogramovat tak zvanou "váhu bodu".		
`<n>`	"Váha bodu"		
	Rozsah hodnot:	0 ≤ n ≤ 3	
	Velikost kroku:	0.0001	
	Působnost:	n > 1	Křivka je silněji přitahována k uzlovému bodu.
		n < 1	Křivka je přitahována k uzlovému bodu méně silně.

Stupeň splinu (jen B-spliny):

`SD`	Standardně se používají se polynomy 3. stupně. Naprogramováním příkazu `SD=2` se mohou ale používat také polynomy 2. stupně.

Vzdálenost mezi uzly (jen B-spliny):

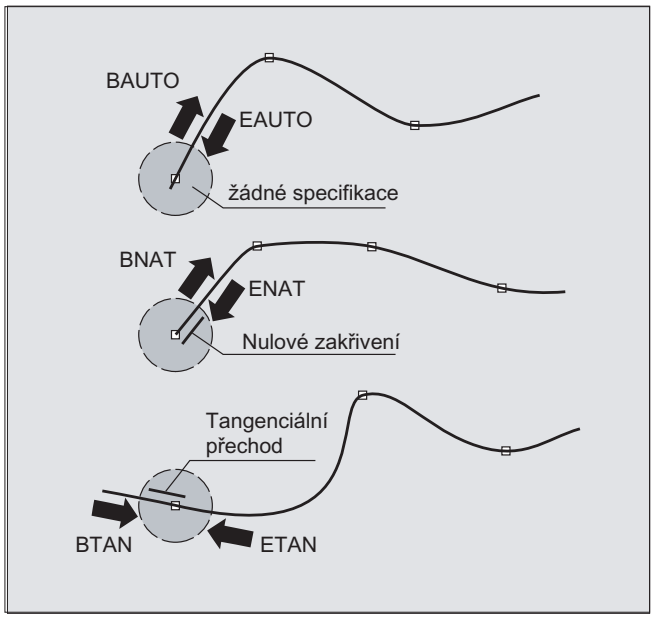
`PL`	Vzdálenosti mezi uzly jsou interně vypočítávány tak, aby byly vhodné. Řídící systém ale může pracovat i se zadanými vzdálenostmi uzlů, které se udávají pomocí příkazu `PL` jako tzv. parametr interval-délka.
`<hodnota>`	Parametr interval-délka Rozsah hodnot: stejně jako rozměrový údaj dráhy

Chování na přechodu na začátku splinové křivky (jen A- a C-spliny):

`BAUTO`	Žádný údaj pro chování na přechodu. Začátek vyplývá z polohy prvního bodu.
`BNAT`	Nulové zakřivení
`BTAN`	Tangenciální přechod k předcházejícímu bloku.

Chování na přechodu na konci splinové křivky (jen A- a C-spliny):

`EAUTO`	Žádný údaj pro chování na přechodu. Konec vyplývá z polohy posledního bodu.
`ENAT`	Nulové zakřivení
`ETAN`	Tangenciální přechod k předcházejícímu bloku.



Poznámka

Naprogramované chování na přechodu nemá na B-spliny žádný vliv. B-spline je v počátečním a koncovém bodě vždy napojen tangenciálně k řídícímu mnohoúhelníku.

Okrajové podmínky

- Korekce rádiusu nástroje může být uplatněna.
- Monitorování kolize se uskutečňuje v projekci v rovině.

Příklady

Příklad 1: B-Spline

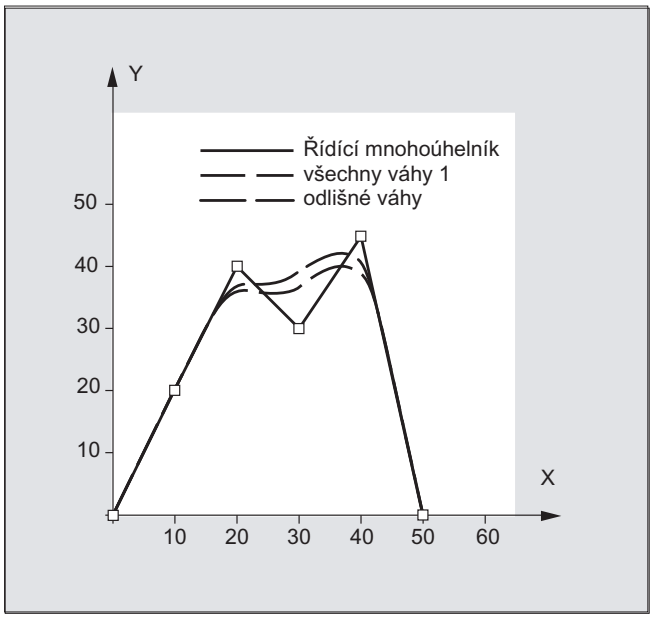
Programový kód 1 (všechny váhy 1)

```
N10 G1 X0 Y0 F300 G64
N20 BSPLINE
N30 X10 Y20
N40 X20 Y40
N50 X30 Y30
N60 X40 Y45
N70 X50 Y0
```

Programový kód 2 (různé váhy)

```
N10 G1 X0 Y0 F300 G64
N20 BSPLINE
N30 X10 Y20 PW=2
N40 X20 Y40
N50 X30 Y30 PW=0.5
N60 X40 Y45
N70 X50 Y0
```

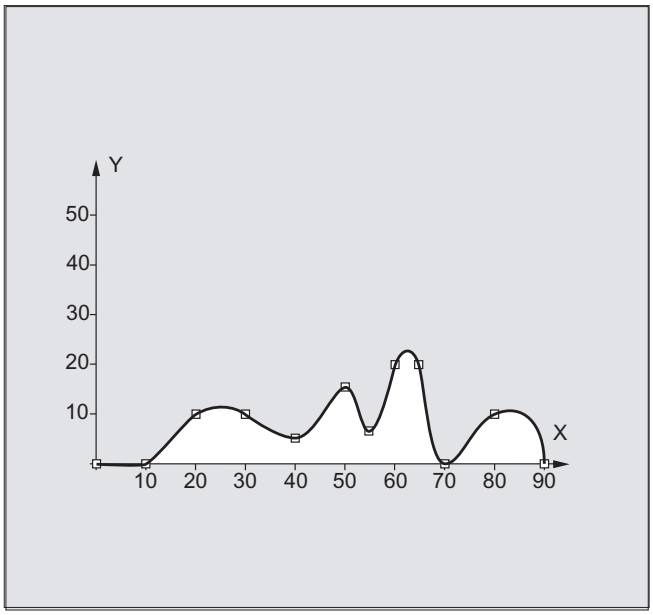
Programová kód 3 (řídící mnohoúhelník)	**Komentář**
N10 G1 X0 Y0 F300 G64	
N20	; odpadá
N30 X10 Y20	
N40 X20 Y40	
N50 X30 Y30	
N60 X40 Y45	
N70 X50 Y0	


Y
Řídící mnohoúhelník
všechny váhy 1
odlišné váhy
50
40
30
20
10
X
10
20
30
40
50
60

Příklad 2: C-spline, na začátku a na konci nulové zakřivení

Programový kód

```
N10 G1 X0 Y0 F300
N15 X10
N20 BNAT ENAT
N30 CSPLINE X20 Y10
N40 X30
N50 X40 Y5
N60 X50 Y15
N70 X55 Y7
N80 X60 Y20
N90 X65 Y20
N100 X70 Y0
N110 X80 Y10
N120 X90 Y0
N130 M30
```



Příklad 3: Splinová interpolace (A-spline)) a transformace souřadného systému (ROT)

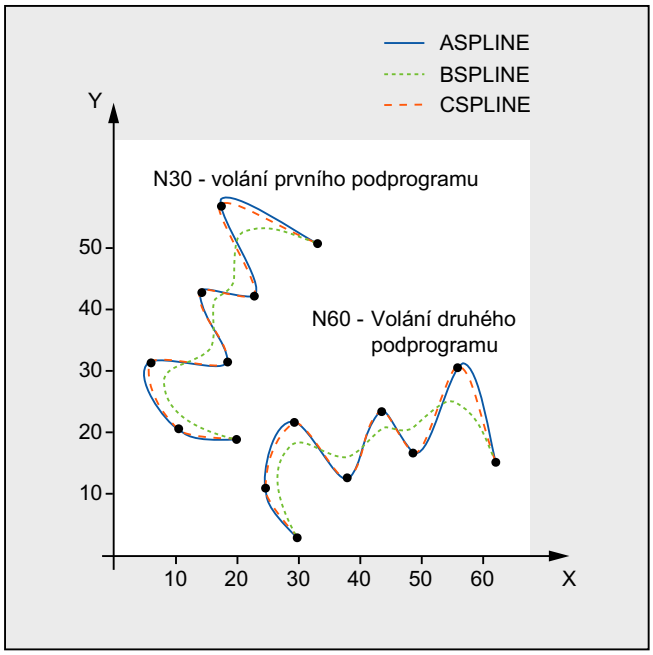
Hlavní program:

Programový kód	Komentář
N10 G00 X20 Y18 F300 G64	; Najetí na počáteční bod.
N20 ASPLINE	; Aktivování typu interpolace A-Spline.
N30 KONTUR	; První volání podprogramu.
N40 ROT Z-45	; Transformace souřadného systému: Otočení WCS o -45° okolo osy Z.
N50 G00 X20 Y18	; Najíždění na počáteční bod kontury.
N60 KONTUR	; Druhé volání podprogramu.
N70 M30	; Konec programu

Podprogram "Kontura" (obsahuje souřadnice uzlového bodu):

Programový kód
N10 X20 Y18
N20 X10 Y21
N30 X6 Y31
N40 X18 Y31
N50 X13 Y43
N60 X22 Y42
N70 X16 Y58
N80 X33 Y51
N90 M1

V následujícím obrázku jsou vedle splinové křivky, která vyplynula z příkladu programování (ASPLINE), obsaženy ještě i splinové křivky, které by vznikly při aktivování interpolace pomocí B- nebo C-splinů (BSPLINE, CSPLINE):



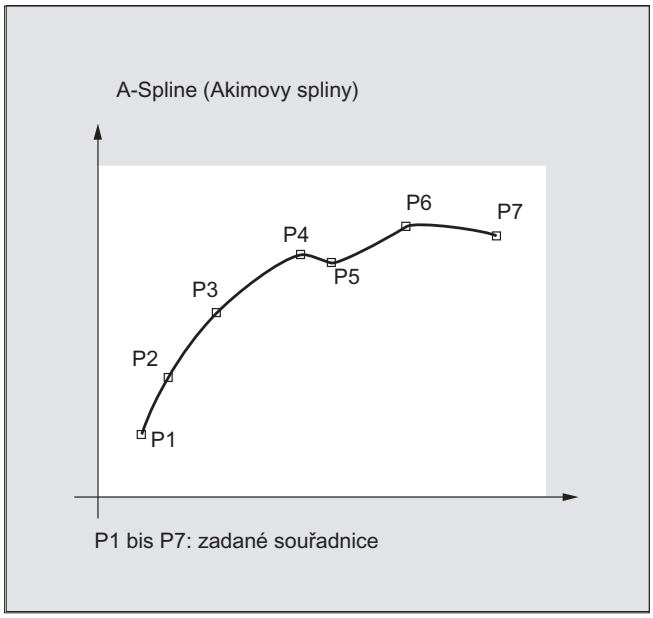
Další informace

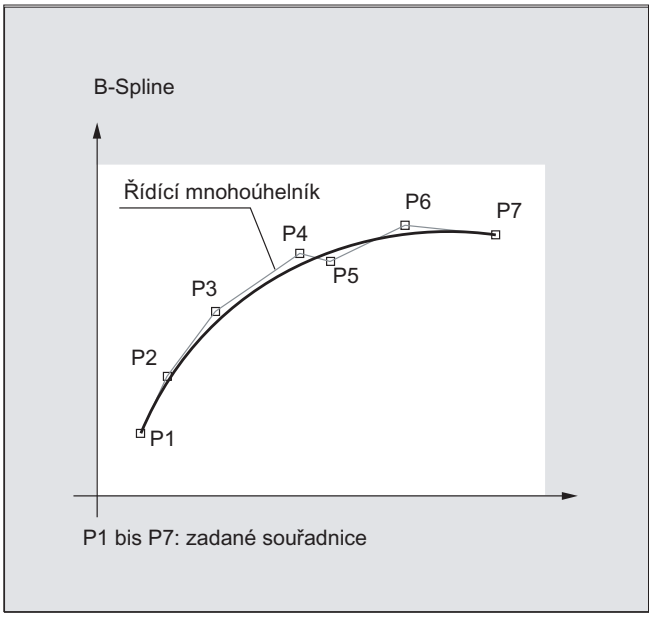
Výhody splinové interpolace

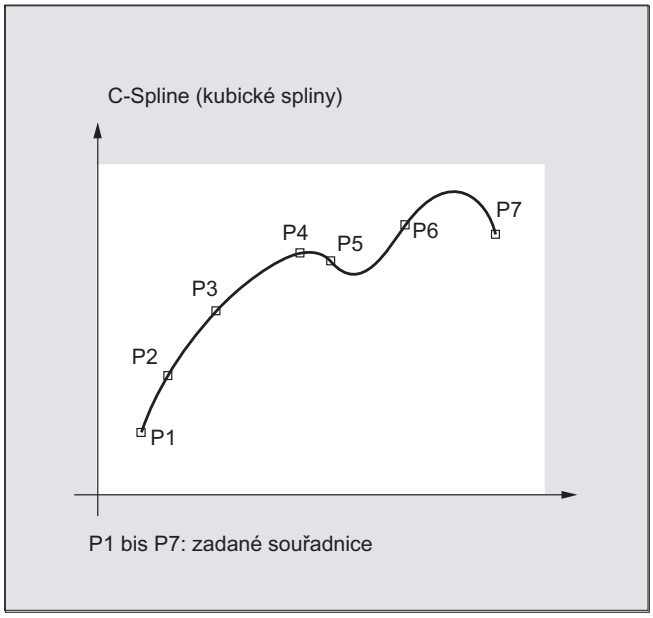
Použitím splinové interpolace se dá oproti použití přímkové interpolace `G01` dosáhnout následujících výhod:

- Snížení počtu potřebných bloků výrobního programu pro popis kontury
- Hladší průběh kontury na přechodech mezi programovými bloky, čímž se šetří mechanika stroje

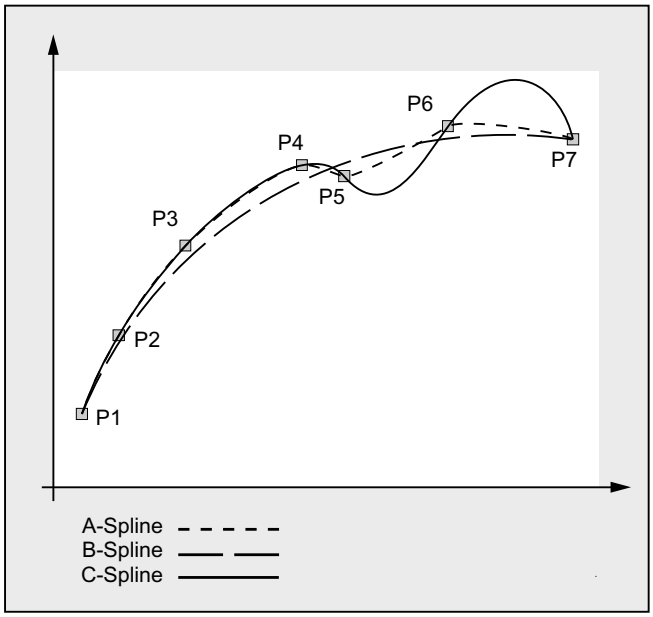
Vlastnosti a použití různých typů splinů

Typ splinu	Vlastnosti a použití
A-Spline	 **Vlastnosti:** • Probíhá přesně zadanými body. • Průběh křivky má tangenciální přechody, nemá však spojité zakřivení • Zřídka způsobuje nežádoucí vibrace. • Oblast vlivu změny uzlového bodu je lokální, tzn. změna jednoho uzlového bodu bude mít vliv pouze na maximálně 6 sousedících uzlových bodů. **Použití:** A-spliny se hodí především pro interpolaci křivkových průběhů s velkými změnami stoupání (např. pro schodovité průběhy křivek).

Typ splinu	Vlastnosti a použití
B-Spline	 **Vlastnosti:** • Křivka neprochází zadanými uzlovými body, nýbrž v jejich blízkosti. Křivka je přitahována k uzlovým bodům. Přiřazováním váhy k uzlovým bodům pomocí faktoru je možné průběh křivky ještě dodatečně ovlivňovat. • Průběh křivky má tangenciální přechody a spojité zakřivení • Nezpůsobuje žádné nežádoucí vibrace. • Oblast vlivu změny uzlového bodu je lokální, tzn. změna jednoho uzlového bodu bude mít vliv pouze na maximálně 6 sousedících uzlových bodů. **Použití:** B-spliny jsou zamýšleny především jako rozhraní pro systémy CAD.

Typ splinu	Vlastnosti a použití
C-Spline	 **Vlastnosti:** • Probíhá přesně zadanými body. • Průběh křivky má tangenciální přechody a spojité zakřivení • Často způsobuje nežádoucí vibrace, a to zejména na místech s velkými změnami stoupání. • Oblast vlivu změny uzlového bodu je globální, tzn. změna jednoho uzlového bodu bude mít vliv celý průběh křivky. **Použití:** C-spliny mohou být výborně uplatněny tehdy, pokud se uzlové body nacházejí na analyticky známé křivce (kruh, parabola, hyperbola).

Porovnání třech typů splinů pomocí stejných uzlových bodů



Minimální počet bloků splinu

G-kódy `ASPLINE`, `BSPLINE` a `CSPLINE` spojují koncové body bloků pomocí splinů.. Za tím účelem musí být při předběžném zpracování současně vypočítána řada bloků (koncových bodů). Velikost vyrovnávací paměti pro tento výpočet standardně činí 10 bloků. Ne každá informace v bloku je koncovým bodem splinu. Řídící systém však potřebuje z 10 bloků určitý počet bloků s koncovými body splinů:

Typ splinu	Minimální počet bloků splinu
A-Spline:	Z každých 10 bloků musí být minimálně **4** bloky splinové. Bloky komentářů a výpočty parametrů se nepočítají.
B-Spline:	Z každých 10 bloků musí být minimálně **6** bloků splinových. Bloky komentářů a výpočty parametrů se nepočítají.
C-Spline:	Potřebný minimální počet splinových bloků vyplývá z následujícího součtu: Hodnota z parametru MD20160 $MC_CUBIC_SPLINE_BLOCKS + 1 Do parametru MD20160 se zadává počet bodů, přes které se splinový úsek vypočítává. Standardní nastavení je 8. Z každých 10 bloků musí proto být ve standardním případě minimálně **9** bloků splinových.

Poznámka

V případě překročení tolerovatelných hodnot se aktivuje alarm, stejně tak, jako když je osa podílející se na splinu naprogramována jako polohovací osa.

Shrnutí krátkých splinových bloků

Při splinové interpolaci mohou vzniknout krátké splinové bloky, což má za následek zbytečné snížení rychlosti pohybu po dráze. Pomocí funkce "Shrnutí krátkých splinových bloků" mohou být tyto bloky soustředěny takový způsobem, že výsledná délka bloků je dostatečně velká a nemá za následek snížení rychlosti pohybu po dráze.

Funkce se aktivuje pomocí kanálového strojního parametru:

MD20488 $MC_SPLINE_MODE (Nastavení pro splinovou interpolaci)

Literatura:
Příručka Popis funkcí, Základní funkce, Režim řízení pohybu po dráze, přesné najetí, funkce Look Ahead (B1), kapitola: Shrnutí krátkých splinových bloků

4.3 Splinový svazek (SPLINEPATH)

Funkce

Pomocí příkazu `SPLINEPATH` jsou vybrány osy, které mají být interpolovány ve splinovém svazku. Při splinové interpolaci je možné pracovat s až osmi dráhovými osami.

Poznámka

Jestliže příkaz `SPLINEPATH` není explicitně naprogramován, zachází se s prvními třemi osami kanálu jako se splinovým svazkem.

Syntaxe

Definice splinového svazku se uskutečňuje v samostatném bloku:

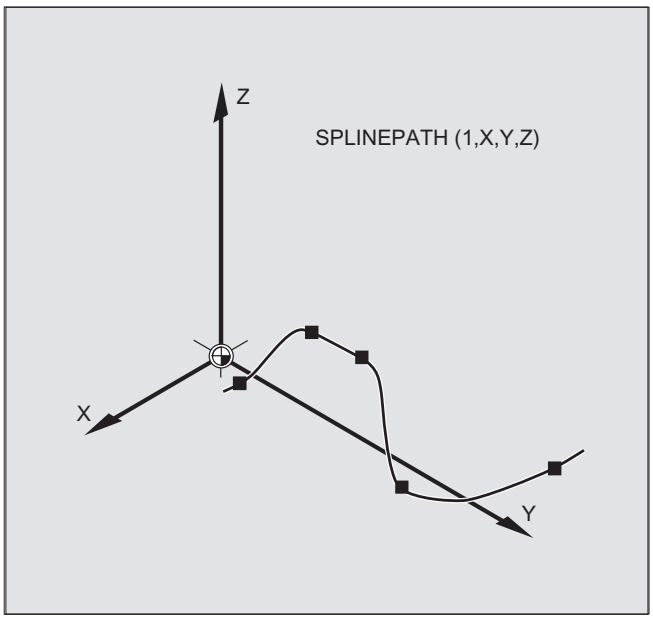
```
SPLINEPATH(n,X,Y,Z,…)
```

Význam

`SPLINEPATH`	Příkaz pro vytvoření splinového svazku.
`n`	=1 (pevná hodnota)
`X,Y,Z,…`	Identifikátory dráhových os, které mají být interpolovány ve splinovém svazku.

Příklad: Splinový svazek se třemi dráhovými osami

Programový kód	Komentář
N10 G1 X10 Y20 Z30 A40 B50 F350	
N11 SPLINEPATH(1,X,Y,Z)	; Splinový svazek
N13 CSPLINE BAUTO EAUTO X20 Y30 Z40 A50 B60	; C-Spline
N14 X30 Y40 Z50 A60 B70	; Uzlové body
…	
N100 G1 X… Y…	; Deaktivování splinové interpolace



4.4 Komprese NC-bloků (COMPON, COMPCURV, COMPCAD, COMPOF)

Funkce

Systémy CAD/CAM zpravidla dodávají posloupnosti lineárních bloků, které zachovávají parametrem nastavenou přesnost. V případě složitých kontur to má za následek velké objemy dat a eventuálně také krátké úseky dráhy. Tyto krátké úseky dráhy omezují rychlost zpracování.

Pomocí funkce kompresoru se provádí aproximace kontury zadané lineárními bloky polynomickými bloky, Z toho vyplývají následující výhody:

- Snížení počtu potřebných bloků výrobního programu pro popis kontury obrobku
- Spojité přechody mezi bloky
- Zvýšení maximální možné rychlosti pohybu po dráze

Jsou Vám k dispozici následující funkce kompresoru:

- COMPON

  Na přechodech mezi bloky je spojitá pouze **rychlost**, zatímco zrychlení os podílejících se na pohybu může na hranicích mezi bloky vykazovat skokové změny.

- COMPCURV

  Na přechodech mezi bloky je **zrychlení spojité**. Tím je zajištěn jak hladší průběh charakteristiky rychlosti, tak také zrychlení všech os na přechodech mezi bloky.

- COMPCAD

  Intenzivní komprese strojového času a paměťového prostoru, která je optimalizována z hlediska jakosti povrchu a rychlosti. Funkce COMCAD by se měla používat jen tehdy, jestliže opatření pro zlepšení jakosti povrchu nemohou být uskutečněna předem v rámci programu CAD/CAM.

Funkce kompresoru se příkazem `COMPOF` ukončí.

Syntaxe

```
COMPON
COMPCURV
COMPCAD
COMPOF
```

Význam

COMPON:	Příkaz pro aktivování funkce kompresoru COMPON.
	Platnost: modální
COMPCURV:	Příkaz pro aktivování funkce kompresoru COMPCURV.
	Platnost: modální
COMPCAD:	Příkaz pro aktivování funkce kompresoru COMPCAD.
	Platnost: modální
COMPOF:	Příkaz pro deaktivování momentálně aktivní funkce kompresoru.

Poznámka

K dalšímu zlepšení jakosti povrchu se mohou použít ještě i funkce pro přechodová zaoblení G642 a funkce pro omezení ryvu SOFT. Tyto příkazy je zapotřebí zapsat na začátku programu.

Okrajové podmínky

- Komprese NC bloků se zpravidla uskutečňuje pro lineární bloky (G1).
- Jsou komprimovány pouze bloky, které mají jednoduchou syntaxi:

  N... G1X... Y... Z... F... ;komentář

  Všechny ostatní bloky jsou zpracovány beze změn (bez komprese).
- Pohybové bloky s rozšířenými adresami, jako např. C=100 nebo A=AC(100) jsou komprimovány také.
- Hodnoty poloh nemusejí být naprogramovány přímo, nýbrž mohou být zadány také nepřímo prostřednictvím přiřazení parametrů, např. X=R1*(R2+R3).
- Jestliže je k dispozici volitelný doplněk "Transformace orientace", pak mohou být komprimovány také NC bloky, ve kterých je prostřednictvím směrových vektorů naprogramována orientace nástroje (a v případě potřeby také otáčení nástroje) (viz "Komprimování orientace (COMPON, COMPCURV, COMPCAD) [Strana 364]").
- Operace komprese se přeruší, pokud se vyskytne jakýkoli jiný příkaz NC jazyka, např. výstup pomocné funkce.

Příklady

Příklad 1: COMPON

Programový kód	Komentář
N10 COMPON	; Zapnutí funkce kompresoru COMPON.
N11 G1 X0.37 Y2.9 F600	; G1 před koncovým bodem a posuv.
N12 X16.87 Y-.698	
N13 X16.865 Y-.72	
N14 X16.91 Y-.799	
…	
N1037 COMPOF	; Vypnutí funkce kompresoru.
…	

Příklad 2: COMPCAD

Programový kód	Komentář
G00 X30 Y6 Z40	
G1 F10000 G642	; Aktivování funkce pro přechodová zaoblení G642.
SOFT	; Zapnutí funkce SOFT pro omezení ryvu.
COMPCAD	; Zapnutí funkce kompresoru COMPCAD.
STOPFIFO	
N24050 Z32.499	
N24051 X41.365 Z32.500	
N24052 X43.115 Z32.497	
N24053 X43.365 Z32.477	
N24054 X43.556 Z32.449	
N24055 X43.818 Z32.387	
N24056 X44.076 Z32.300	
…	
COMPOF	; Vypnutí funkce kompresoru.
G00 Z50	
M30	

Literatura

Příručka Popis funkcí, Základní funkce, Režim řízení pohybu po dráze, přesné najetí, funkce Look Ahead (B1), kapitola: "Komprese NC bloků"

4.5 Polynomická interpolace (POLY, POLYPATH, PO, PL)

Funkce

V zásadě se u polynomické interpolace (`POLY`) nejedná o druh splinové interpolace. V první řadě je míněna jako rozhraní pro programování externě vytvořených splinových křivek. Tímto způsobem mohou být splinové úseky naprogramovány přímo.

Tento druh interpolace zbavuje NC systém nutnosti vypočítávat koeficienty polynomu. Může být optimálně využit tehdy, pokud jsou koeficienty dodávány přímo ze systému CAD nebo ze systému pro dodatečné zpracování.

Syntaxe

Polynom 3. stupně:

```
POLY PO[X]=(xe,a2,a3) PO[Y]=(ye,b2,b3) PO[Z]=(ze,c2,c3) PL=n
```

Polynomy 5. stupně a nová syntaxe polynomu:

```
POLY X=PO(xe,a2,a3,a4,a5) Y=PO(ye,b2,b3,b4,b5) Z=PO(ze,c2,c3,c4,c5)
PL=n
POLYPATH("AXES","VECT")
```

Poznámka

Součet koeficientů polynomu naprogramovaných v NC bloku a os nesmí překročit maximální přípustný počet os najeden blok.

Význam

`POLY`:	Polynomická interpolace se aktivuje blokem s příkazem POLY.
`POLYPATH`:	Polynomická interpolace může být vybrána pro obě skupiny os AXIS nebo VECT
`PO[identifikátor osy/proměnná]`:	Koncové body a koeficienty polynomu
`X, Y, Z`:	Identifikátor osy
`xe, ye, ze`:	Údaj koncové pozice pro příslušné osy, rozsah hodnot a jednotka

`a2, a3, a4, a5:`	Koeficienty a_2, a_3, a_4 a a_5 jsou popsány svou hodnotou, rozsahem hodnot a jednotkou. Poslední koeficient může odpadnout, jestliže má nulovou hodnotu.
`PL:`	Délka intervalu parametru, na kterém je polynom definován (definiční oblast funkce f(p)). Interval má svůj počátek vždy v 0, p může nabývat hodnot od 0 do PL. Teoretický rozsah hodnot pro PL: 0,0001 … 99 999,9999 **Upozornění:** Hodnota PL platí pro blok, ve kterém se nachází. Pokud hodnota PL není naprogramována, platí PL =1.

Aktivování/deaktivování polynomické interpolace.

Polynomická interpolace se ve výrobním programu zapíná příkazem G-funkce `POLY`.

Příkaz G-funkce `POLY` patří spolu s příkazy `G0`, `G1`, `G2`, `G3`, `ASPLINE`, `BSPLINE` a `CSPLINE` do 1. G-skupiny.

Osy, které jsou naprogramovány pouze svým názvem a koncovým bodem (např. `X10`) se pohybují lineárně. Pokud jsou tímto způsobem naprogramovány všechny osy v NC bloku, chová se řídící systém jako při funkci G1.

Naprogramováním jiného příkazu z 1. skupiny (např. `G0`, `G1`) se polynomická interpolace implicitně deaktivuje.

Koeficienty polynomu

Hodnota `PO(PO[]=)`, příp. `...=PO(...)` udává všechny koeficienty polynomu pro danou osu. V souladu se stupněm polynomu se větší počet hodnot zadává oddělených čárkou. V rámci jednoho bloku mohou být pro jednotlivé osy zadány polynomy různých stupňů.

Podprogram POLYPATH

Pomocí příkazu `POLYPATH(...)` může být polynomická interpolace selektivně odblokována pro určité skupiny os.

Jen dráhové osy a doplňkové osy:	`POLYPATH("AXES")`
Jen orientační osy: (při pohybech s transformací orientace)	`POLYPATH("VECT")`

Osy, které nebyly tímto způsobem odblokovány, se pohybují jako osy lineární.

Standardně je polynomická interpolace uvolněna pro obě skupiny os.

Naprogramováním příkazu `POLYPATH( )` bez parametrů se polynomická interpolace pro všechny osy deaktivuje.

Příklad

Programový kód	**Komentář**
N10 G1 X… Y… Z… F600	
N11 POLY PO[X]=(1,2.5,0.7) PO[Y]=(0.3,1,3.2) PL=1.5	; Aktivování polynomické interpolace
N12 PO[X]=(0,2.5,1.7) PO[Y]=(2.3,1.7) PL=3	
...	
N20 M8 H126 …	
N25 X70 PO[Y]=(9.3,1,7.67) PL=5	; smíšené údaje pro osy
N27 PO[X]=(10,2.5) PO[Y]=(2.3)	; hodnota PL není naprogramována, platí PL =1
N30 G1 X… Y… Z.	; Deaktivování polynomické interpolace
…	

Příklad: Nová syntaxe polynomu

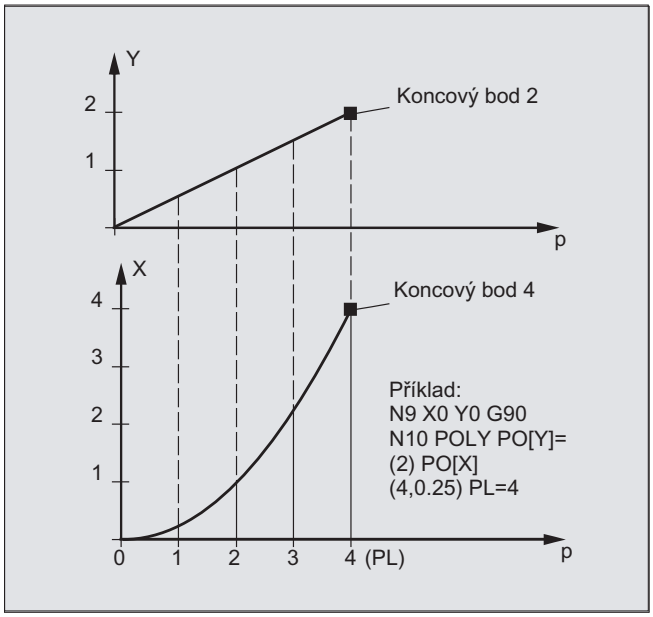
Dále platná syntaxe polynomu	Nová syntaxe polynomu
PO[identifikátor osy]=(.. , ..)	Identifikátor osy=PO(.. , ..)
PO[PHI]=(.. , ..)	PHI=PO(.. , ..)
PO[PSI]=(.. , ..)	PSI=PO(.. , ..)
PO[THT]=(.. , ..)	THT=PO(.. , ..)
PO[]=(.. , ..)	PO(.. , ..)
PO[proměnná]=IC(.. , ..)	proměnná=PO IC(.. , ..)

Příklad: Křivka v rovině X/Y

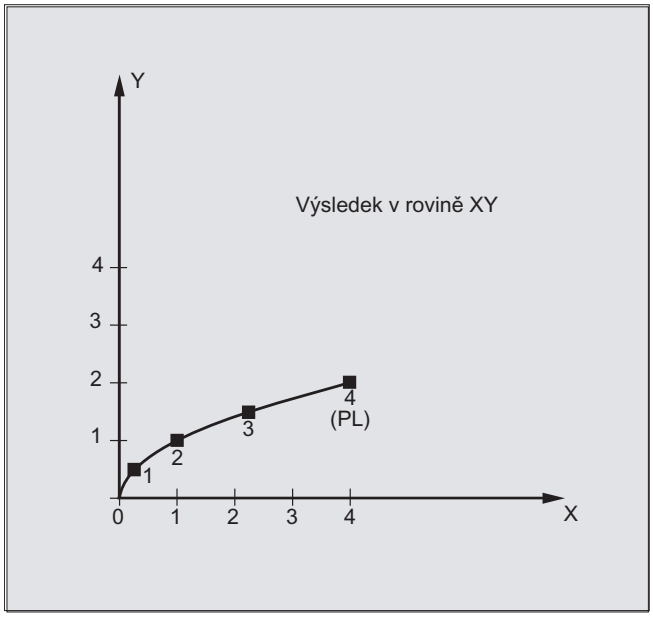
Programování

Programový kód
N9 X0 Y0 G90 F100
N10 POLY PO[Y]=(2) PO[X]=(4,0.25) PL=4

Průběh křivky X(p) a Y(p)



Průběh křivky v rovině XY



Popis

V obecném tvaru má polynomická funkce následující podobu:

$f(p)= a_0 + a_1p + a_2p^2 +. . . + a_np^n$

kde: a_n: konstantní koeficienty
p: Parametry

V řídícím systému je možné programovat maximálně polynomy 5. stupně::

$f(p)= a_0 + a_1p + a_2p^2 + a_3p^3 + a_4p^4 + a_5p^5$

Dosazením konkrétních hodnot koeficientům je možné vytvářet různé průběhy křivek, jako jsou přímky, paraboly a mocninné funkce.

Přímka vzniká dosazením $a_2 = a_3 = a_4 = a_5 = 0$:

$f(p)= a_0 + a_1p$

Kromě toho platí:

$a_{0:}$ Poloha osy na konci předcházejícího bloku

p = PL

$a_1 = (x_E - a_0 - a_2*p^2 - a_3*p^3) / p$

Je možné naprogramovat polynomy, **aniž** by byla polynomická interpolace příkazem G-funkce `POLY` aktivována. V tomto případě nejsou interpolovány naprogramované polynomy, ale na naprogramované koncové body os se najíždí po lineárních drahách (`G1`). Teprve po explicitním aktivování polynomické interpolace ve výrobním programu (`POLY`) jsou pohyby uskutečňovány po křivkových drahách odpovídajících naprogramovaným polynomům.

Zvláštnost: Polynom jmenovatele

Pro geometrické osy je možno pomocí příkazu `PO[]=(…)` bez udání názvu osy naprogramovat také společný polynom jmenovatele, což znamená, že pohyb geometrické osy bude potom interpolován jako podíl dvou polynomů.

Tímto způsobem se dají přesně zobrazit např. kuželosečky (kruh, elipsa, parabola, hyperbola).

Příklad:

Programový kód	**Komentář**
`POLY G90 X10 Y0 F100`	`; Geometrické osy lineárně najíždějí na pozici X10 Y0.`
`PO[X]=(0,-10) PO[Y]=(10) PO[]=(2,1)`	`; Geometrické osy najíždějí po čtvrtkruhu na bod X0 Y10.`

Koeficient konstanty (a_0) polynomu jmenovatele je vždy nastaven na 1. Naprogramovaný koncový bod je nezávislý na příkazech `G90` / `G91`.

Z naprogramovaných hodnot se vypočítají hodnoty X(p) a Y(p):

$X(p) = (10 - 10 * p^2) / (1 + p^2)$

$Y(p) = 20 * p / (1 + p^2)$

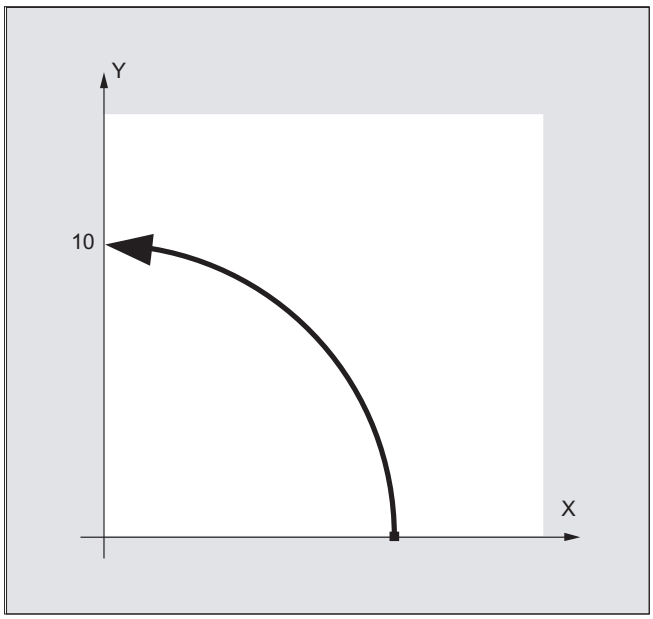
kde $0 \leq p \leq 1$

Na základě naprogramovaných počátečních bodů, koncových bodů, koeficientu a_2 a PL=1 vyplývají následující mezivýsledky:

Čitatel (X) = $10 + 0 * p - 10 * p^2$

Čitatel (Y) = $0 + 20 * p + 0 * p^2$

Jmenovatel = $1 + p^2$



Když je aktivována polynomická interpolace, je naprogramování polynomu jmenovatele s nulovými koeficienty v rámci intervalu `[0,PL]` odmítnuto a aktivuje se alarm. Pohyb doplňkových os nemá na polynom jmenovatele žádný vliv.

Poznámka

Korekce rádiusu nástroje může být v případě polynomické interpolace pomocí příkazů `G41`, `G42` aktivována/deaktivována a může se používat stejně jako při přímkové nebo kruhové interpolaci.

4.6 Nastavitelný vztah k dráze (SPATH, UPATH)

Funkce

V případě polynomické interpolace může uživatel požadovat dva odlišné vztahy mezi osami skupiny FGROUP, které určují rychlost, a zbývajícími dráhovými osami. Tyto zbývající dráhové osy se mají pohybovat buď synchronně k dráze S nebo ke křivkovému parametru U os skupiny FGROUP.

Oba druhy dráhové interpolace jsou zapotřebí v různých případech použití a mohou být aktivovány/přepínány prostřednictvím obou příkazů NC jazyka `SPATH` a `UPATH` s modální platností, které jsou obsaženy ve.45. skupině G-kódů.

Syntaxe

```
SPATH
UPATH
```

Význam

`SPATH`:	Referenční dráha pro osy v FGROUP je délka oblouku
`UPATH`:	Referenční dráha pro osy v FGROUP je křivkový parametr

Poznámka

Funkce `UPATH` a `SPATH` určují také souvislost mezi polynomem F-slova (`FPOLY`, `FCUB`, `FLIN`) a pohybem po dráze.

Okrajové podmínky

Nastavený vztah k dráze nemá žádný význam:

- u lineární a kruhové interpolace
- v závitových blocích
- pokud jsou všechny dráhové osy obsaženy ve skupině FGROUP

Příklady

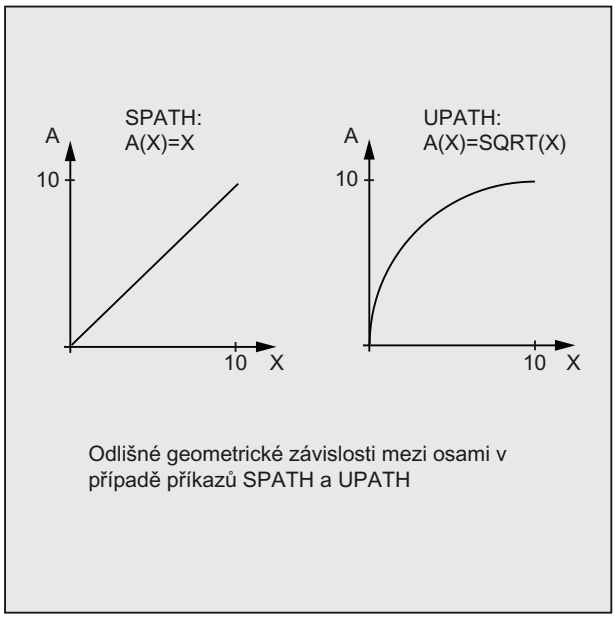
Příklad 1:

V následujícím příkladu se provádí přebroušení čtverce o délce hrany 20 mm s funkcí G643. Maximální odchylka od přesné kontury je přitom pro každou osu definována prostřednictvím strojního parametru MD33100 $MA_COMPRESS_POS_TOL[<n>].

Programový kód	**Komentář**
N10 G1 X… Y… Z… F500	
N20 G643	; Interní blokové přechodové zaoblení pomocí příkazu G643
N30 XO Y0	
N40 X20 Y0	; Délka hrany (mm) pro osy
N50 X20 Y20	
N60 X0 Y20	
N70 X0 Y0	
N100 M30	

Příklad 2:

Následující příklad ilustruje rozdíl mezi oběma druhy řízení pohybu. V obou případech je předem aktivováno nastavení FGROUP(X,Y,Z).



Odlišné geometrické závislosti mezi osami v případě příkazů SPATH a UPATH

Programový kód
N10 G1 X0 A0 F1000 SPATH
N20 POLY PO[X]=(10,10) A10

Příp

Programový kód
N10 G1 X0 F1000 UPATH
N20 POLY PO[X]=(10,10) A10

V bloku N20 závisí dráha S os ve skupině FGROUP na druhé mocnině křivkového parametru U. Z toho vyplývají různé polohy synchronní osy A podél dráhy osy X podle toho, jestli je aktivován příkaz SPATH nebo UPATH.

Další informace

V průběhu polynomické interpole - a tím se myslí vždy polynomická interpolace v užším smyslu slova (POLY), všechny druhy splinové interpolace (ASPLINE, BSPLINE, CSPLINE) a lineární interpolace s funkcí kompresoru /COMPON, COMPCURV) - jsou polohy všech dráhových os dány prostřednictvím polynomu p(U). Křivkový parametr U se přitom v rámci NC bloku pohybuje v hodnotách od 0 do 1, je tedy normovaný.

Prostřednictvím příkazu NC jazyka FGROUP mohou být mezi dráhovými osami vybrány ty osy, na které se má naprogramovaný posuv po dráze vztahovat. Interpolace s konstantní rychlostí na dráze S u těchto os však v průběhu polynomické interpolace zpravidla znamená nekonstantní změnu křivkového parametru U.

Chování řídícího systému při resetu a strojní parametry/parametry doplňků

Po resetu je prostřednictvím strojního parametru MD20150 $MC_GCODE_RESET_VALUES[44] určitý G-kód nastaven jako platný (skupina G-kódů č. 45). Kvůli kompatibilitě s již existujícími zařízeními je v tomto případě předem nastaven jako standardní hodnota příkaz SPATH.

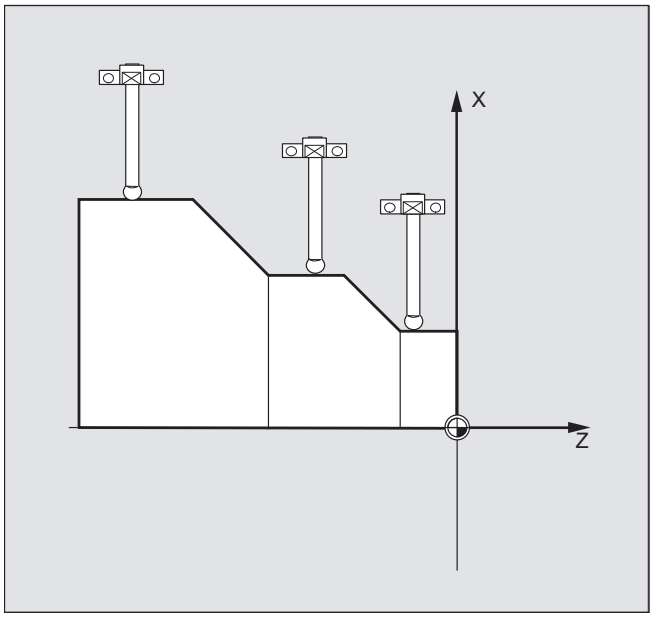
Hodnota základního nastavení pro druh přechodových zaoblení je definována parametrem MD20150 $MC_GCODE_RESET_VALUES[9] (skupina G-kódů č. 10).

Osový strojní parametr MD33100 $MA_COMPRESS_POS_TOL[<n>] má rozšířený význam: Obsahuje tolerance pro funkci kompresoru a pro přechodová zaoblení pomocí příkazu G642.

4.7 Měření se spínací sondou (MEAS, MEAW)

Funkce

Pomocí funkce "Měření se spínací sondou" se najíždí na skutečné polohy na obrobku, a když je zaznamenána skoková změna (sepnutí) signálu spínací sondy, pro všechny osy naprogramované v bloku měření se změří daná pozice a ta se pro každou z těchto os zapíše do odpovídající paměťové buňky.



Programování měřicích bloků

Pro programování této funkce máte k dispozici následující dva příkazy:

- `MEAS`

  Pomocí příkazu `MEAS` se zbytková dráha mezi skutečnou a požadovanou polohou vymaže.

- `MEAW`

  Příkaz `MEAW` se používá pro měřicí úlohy, u kterých se má v každém případě najet na naprogramovanou pozici.

Příkazy `MEAS` a `MEAW` mají blokovou platnost a jsou naprogramovány spolu s příkazy pro pohyb. Posuvy a druhy interpolace (G0, G1, ...), stejně jako počet os, přitom musí být přizpůsobeny konkrétnímu měřicímu problému.

Načtení výsledků měření

Výsledky měření pro osy, které byly měřicí sondou zaznamenány, jsou k dispozici v následujících proměnných:

- `$AA_MM[<osa>]`

  Výsledky měření v souřadném systému stroje

- `$AA_MW[<osa>]`

  Výsledky měření v souřadném systému obrobku

Při načítání těchto proměnných nedochází k žádnému internímu zastavení předběžného zpracování.

Poznámka

Zastavení předběžného zpracování musí být naprogramováno na vhodném místě v NC programu pomocí příkazu `STOPRE`. Jinak jsou načteny nesprávné hodnoty.

Syntaxe

```
MEAS=<TE> G... X... Y... Z...
MEAW=<TE> G... X... Y... Z...
```

Význam

`MEAS`	Příkaz: Měření **s** vymazáním zbytkové dráhy Platnost: bloková
`MEAW`	Příkaz: Měření **bez** vymazání zbytkové dráhy Platnost: bloková
`<TE>`	Událost sepnutí za účelem spuštění měření Typ: INT Rozsah hodnot: -2, -1, 1, 2 **Upozornění:** Existují maximálně 2 měřicí sondy (v závislosti na úrovni vybavení). Význam: (+)1 náběžná hrana signálu měřicí sondy 1 (na měřicím vstupu 1) -1 sestupná hrana signálu měřicí sondy 1 (na měřicím vstupu 1) (+)2 náběžná hrana signálu měřicí sondy 2 (na měřicím vstupu 2) -2 sestupná hrana signálu měřicí sondy 2 (na měřicím vstupu 2) **Upozornění:** Existují maximálně 2 měřicí sondy (v závislosti na úrovni vybavení).
`G...`	Druh interpolace, např. `G0`, `G1`, `G2` nebo `G3`
`X... Y... Z...`	Koncový bod v kartézských souřadnicích

Příklad

Programový kód	Komentář
N10 MEAS=1 G1 F1000 X100 Y730 Z40	; Blok měření s měřicí sondou na prvním měřicím vstupu a přímková interpolace. Je automaticky generováno zastavení předběžného zpracování.
...	

Další informace

Stav měřicí úlohy

Jestliže je v programu zapotřebí vyhodnotit, zda měřicí sonda sepnula nebo ne, může být zjišťován obsah stavové proměnné `$AC_MEA[n]` (n = číslo měřicí sondy).

Hodnota	Význam
0	Měřicí úloha není splněna
1	Měřicí úloha byla úspěšně dokončena)měřicí sonda sepnula).

Poznámka

Pokud je měřicí sonda v programu vychýlena, je proměnná nastavena na 1. Při spuštění bloku měření se proměnná automaticky nastaví podle počátečního stavu sondy.

Odečítání změřených hodnot

Jsou zaznamenávány pozice všech dráhových a polohovacích os v bloku, které se pohybovaly (maximální počet os závisí na konfiguraci řídícího systému). V případě příkazu `MEAS` se pohyb po sepnutí měřicí sondy definovaným způsobem zastaví.

Poznámka

Jestliže je v měřicím bloku naprogramována geometrická osa, jsou změřené hodnoty uloženy pro všechny momentálně definované geometrické osy.

Jestliže je v měřicím bloku naprogramována osa podílející se na nějaké transformaci, jsou změřené hodnoty uloženy i pro všechny na této transformaci se podílející osy.

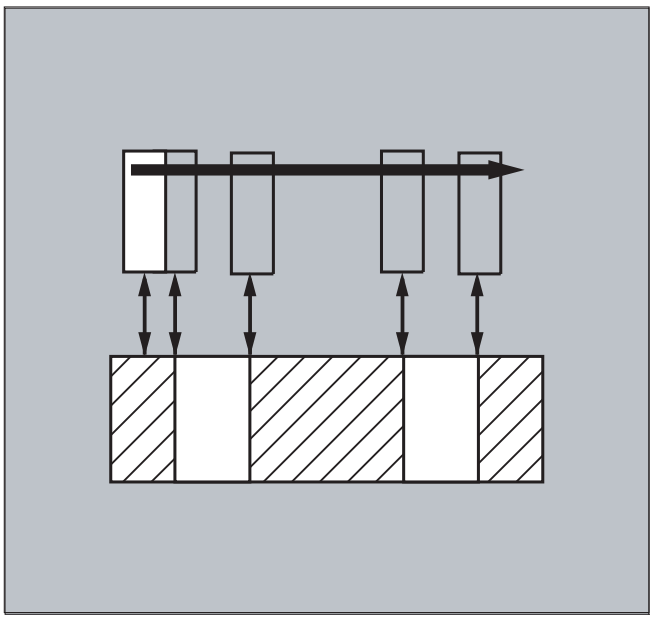
4.8 Rozšířené měřicí funkce (MEASA, MEAWA, MEAC) (volitelný doplněk)

Funkce

V případě měření s jednou osou se může používat větší počet měřicích sond a větší počet měřicích systémů.

Pomocí příkazu `MEASA`, příp. `MEAWA` jsou pro příslušnou naprogramovanou osu zaznamenány v každém měření až čtyři změřené hodnoty, které se pak ukládají podle události spouštění do systémových proměnných.

Kontinuální měřicí úlohy je možné uskutečňovat pomocí příkazu `MEAC`. V tomto případě jsou výsledky měření ukládány do proměnných typu FIFO. Také pro funkci `MEAC` je možné pro každé měření ukládat až čtyři změřené hodnoty.



Načtení výsledků měření

Výsledky měření jsou k dispozici v následujících proměnných:

- $AA_MM1...4[<osa>]

  Výsledky měření v souřadném systému stroje

- $AA_MW1...4[<osa>]

  Výsledky měření v souřadném systému obrobku

Syntaxe

```
MEASA[<osa>]=(<režim>,<TE1>,...,<TE4>)
MEAWA[<osa>]=(<režim>,<TE1>,...,<TE4>)
MEAC[<osa>]=(<režim>,<paměť měření>,<TE1>,...,<TE4>)
```

Poznámka

Příkazy MEASA a MEAWA mají blokovou platnost a mohou být naprogramovány společně v jednom bloku. Pokud je oproti tomu příkaz MEASA/MEAWA naprogramován v jednom bloku spolu s příkazem MEAS/MEAW, je generováno chybové hlášení.

Význam

MEASA
Příkaz: Měření jednou osou s vymazáním zbytkové dráhy
Platnost: bloková

MEAWA
Příkaz: Měření jednou osou bez vymazání zbytkové dráhy
Platnost: bloková

MEAC
Příkaz: Kontinuální měření jednou osou bez vymazání zbytkové dráhy
Platnost: bloková

<osa>
Název kanálové osy použité pro měření

<režim>
Dvoumístné číslo pro zadávání provozního režimu (režim měření a měřicí systém)

Režim měření (místo jednotek)

0 Přerušení měřicí úlohy.

1 Až 4 různé spouštěcí události, které mohou být aktivovány současně.

2 Až 4 postupně aktivovatelné spouštěcí události.

3 Až 4 postupně aktivovatelné spouštěcí události, avšak žádné monitorování spouštěcí události 1 při spuštění (alarmy 21700/21703 jsou potlačeny).

Upozornění:
V případě příkazu MEAC není tento režim možný.

Měřicí systém (místo desítek):

0 (žádný údaj) aktivní měřicí systém

1 Měřicí systém 1

2 Měřicí systém 2

3 Oba měřicí systémy

<TE>
Událost sepnutí za účelem spuštění měření

Typ: INT

Rozsah hodnot: -2, -1, 1, 2

Význam:

(+)1 náběžná hrana měřicí sondy 1

-1 sestupná hrana měřicí sondy 1

(+)2 náběžná hrana měřicí sondy 2

-2 sestupná hrana měřicí sondy 2

<paměť měření>
Číslo FIFO (zásobníková paměť)

Příklady

Příklad 1: Měření jednou osou s vymazáním zbytkové dráhy v režimu 1 (vyhodnocování v časové posloupnosti)

a) s 1 měřicím systémem

Programový kód	Komentář
...	
N100 MEASA[X]=(1,1,-1) G01 X100 F100	; Měření v režimu 1 s aktivním měřicím systémem. Čekání na měřicí signál s náběžnou/sestupnou hranou signálu měřicí sondy 1 na dráze po X=100.
N110 STOPRE	; Zastavení předběžného zpracování
N120 IF $AC_MEA[1]==FALSE GOTOF ENDE	; Kontrola úspěšného měření:
N130 R10=$AA_MM1[X]	; Uložit změřenou hodnotu patřící k první naprogramované spouštěcí události (náběžná hrana).
N140 R11=$AA_MM2[X]	; Uložit změřenou hodnotu patřící ke druhé naprogramované spouštěcí události (sestupná hrana).
N150 ENDE:	

b) se 2 měřicími systémy

Programový kód	Komentář
...	
N200 MEASA[X]=(31,1,-1) G01 X100 F100	; Měření v režimu 1 s oběma měřicími systémy. Čekání na měřicí signál s náběžnou/sestupnou hranou signálu měřicí sondy 1 na dráze po X=100.
N210 STOPRE	; Zastavení předběžného zpracování
N220 IF $AC_MEA[1]==FALSE GOTOF ENDE	; Kontrola úspěšného měření:
N230 R10=$AA_MM1[X]	; Uložení změřené hodnoty z měřicího systému 1 při náběžné hraně signálu.
N240 R11=$AA_MM2[X]	; Uložení změřené hodnoty z měřicího systému 2 při náběžné hraně signálu.
N250 R12=$AA_MM3[X]	; Uložení změřené hodnoty z měřicího systému 1 při sestupné hraně signálu.
N260 R13=$AA_MM4[X]	; Uložení změřené hodnoty z měřicího systému 2 při sestupné hraně signálu.
N270 ENDE:	

Příklad 2: Měření jednou osou s vymazáním zbytkové dráhy v režimu 2 (vyhodnocování v naprogramované posloupnosti)

Programový kód	Komentář
...	
N100 MEASA[X]=(2,1,-1,2,-2) G01 X100 F100	; Měření v režimu 2 s aktivním měřicím systémem. Čekání na měřicí signály v posloupnosti náběžná hrana měřicí sondy 1, sestupná hrana měřicí sondy 1, náběžná hrana měřicí sondy 2, sestupná hrana měřicí sondy 2 na dráze po X=100.
N110 STOPRE	; Zastavení předběžného zpracování
N120 IF $AC_MEA[1]==FALSE GOTOF MESSTASTER2	; Kontrola úspěšného měření měřicí sondou 1:
N130 R10=$AA_MM1[X]	; Uložení změřené hodnoty patřící k první naprogramované spouštěcí události (náběžná hrana měřicí sondy 1).
N140 R11=$AA_MM2[X]	; Uložení změřené hodnoty patřící ke druhé naprogramované spouštěcí události (náběžná hrana měřicí sondy 1).
N150 MESSTASTER2:	
N160 IF $AC_MEA[2]==FALSE GOTOF ENDE	; Kontrola úspěšného měření měřicí sondou 2:
N170 R12=$AA_MM3[X]	; Uložení změřené hodnoty patřící ke třetí naprogramované spouštěcí události (náběžná hrana měřicí sondy 2).
N180 R13=$AA_MM4[X]	; Uložení změřené hodnoty patřící ke čtvrté naprogramované spouštěcí události (náběžná hrana měřicí sondy 2).
N190 ENDE:	

Příklad 3: Kontinuální měření jednou osou v režimu 1 (vyhodnocování v časové posloupnosti)

a) Měření až 100 změřených hodnot

Programový kód	Komentář
...	
N110 DEF REAL MESSWERT[100]	
N120 DEF INT Schleife=0	
N130 MEAC[X]=(1,1,-1) G01 X1000 F100	; Měření v režimu 1 s aktivním měřicím systémem, s ukládáním změřených hodnot do proměnné $AC_FIFO1, čekání na měřicí signál se sestupnou hranou signálu měřicí sondy 1 na dráze po X=1000.
N135 STOPRE	
N140 MEAC[X]=(0)	; Ukončení měření po dosažení polohy osy:
N150 R1=$AC_FIFO1[4]	; Ukládání počtu zaznamenaných změřených hodnot do parametru R1.
N160 FOR Schleife=0 TO R1-1	
N170 MESSWERT[Schleife]=$AC_FIFO1[0]	; Načítání a ukládání změřených hodnot z proměnné $AC_FIFO1.
N180 ENDFOR	

b) Měření s vymazáním zbytkové dráhy po 10 změřených hodnotách

```
Programový kód                                                    Komentář
...
N10 WHEN $AC_FIFO1[4]>=10 DO MEAC[x]=(0) DELDTG(x)                ; Vymazání zbytkové
                                                                    dráhy.
N20 MEAC[x]=(1,1,1,-1) G01 X100 F500
N30 MEAC[X]=(0)
N40 R1=$AC_FIFO1[4]                                               ; Počet změřených
                                                                    hodnot.
...
```

Další informace

Měřicí úloha

Programování měřicích úloh se může uskutečňovat ve výrobním programu nebo ze synchronní akce (viz kapitola "Pohybové synchronní akce"). Pro každou osu přitom může být v jednom a tomtéž okamžiku aktivní jen jedna měřicí úloha.

Poznámka

Posuv je potřeba přizpůsobit příslušnému měřicímu problému.

U příkazů `MEASA` a `MEAWA` mohou být správné výsledky zaručeny jedině při hodnotách posuvu, při kterých se nemůže vyskytnout více než jedna stejná a více než 4 různé spouštěcí události na jeden takt regulátoru polohy.

V případě kontinuálního měření s funkcí `MEAC` nesmí být poměr mezi interpolačním taktem a taktem polohového regulátoru větší než 8:1.

Spouštěcí událost

Spouštěcí událost se skládá z čísla měřicí sondy a ze spouštěcího kritéria (náběžná nebo sestupná hrana) signálu měřicí sondy.

Pro každé měření mohou být zpracovány vždy až 4 spouštěcí události spínající měřicí sondy, tedy až dvě měřicí sondy, každá se dvěma hranami měřicího signálu. Posloupnost zpracování, jakož i také maximální počet spouštěcích událostí jsou přitom závislé na zvoleném režimu.

Poznámka

Pro měřicí režim 1 platí: Stejná spouštěcí událost smí být v dané měřicí úloze naprogramována jen jednou!

Provozní režim

Pomocí první číslice (místo desítek) provozního režimu se vybírá požadovaný měřicí systém. Jestliže je k dispozici jen jeden měřicí systém, je však naprogramován ten druhý, automaticky se použije ten, který je k dispozici.

Pomocí druhé číslice (místo jednotek) se nastavuje požadovaný režim měření. Tímto způsobem může být měřicí operace možnostem příslušného řídícího systému.

- **Režim1**

  Vyhodnocování spouštěcích událostí se uskutečňuje v časové posloupnosti, v jaké se vyskytly. V tomto režimu může být při použití modulů se šesti osami naprogramována jen jedna spouštěcí událost, příp. při zadání většího poštu spouštěcích událostí se režim automaticky přepne do režimu 2 (bez hlášení).

- **Režim 2**

  Vyhodnocování spouštěcích událostí se uskutečňuje v naprogramované posloupnosti.

- **Režim 3**

  Vyhodnocování spouštěcí události se uskutečňuje v naprogramované posloupnosti, neprovádí se však žádné monitorování spouštěcí události 1 při spuštění.

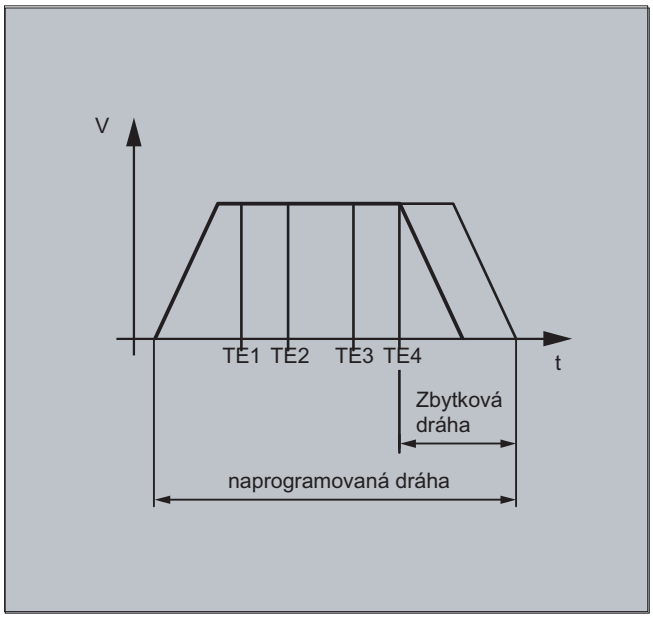
Poznámka

Pokud jsou použity 2 měřicí systémy, mohou být naprogramovány pouze dvě spouštěcí události.

Měření s a bez vymazání zbytkové dráhy

Jestliže je naprogramován příkaz `MEASA`, vymazání zbytkové dráhy se uskuteční až po zaznamenání všech požadovaných měřených hodnot.

Příkaz `MEAWA` se používá v případě speciálních měřicích úloh, u kterých se má v každém případě najet na naprogramovanou pozici.



Poznámka

Příkaz `MEASA` nemůže být naprogramován v synchronních akcích. Místo toho může být jako synchronní akce naprogramován příkaz `MEAWA` plus vymazání zbytkové dráhy.

Pokud je ze synchronní akce spuštěna měřicí úloha s příkazem `MEAWA`, jsou změřené hodnoty k dispozici pouze v souřadném systému stroje.

Výsledky měření pro příkazy MEASA, MEAWA

Výsledky měření jsou k dispozici v následujících systémových proměnných:

- V souřadném systému stroje:

`$AA_MM1[<osa>]`	změřená hodnota z naprogramovaného měřicího systému při spouštěcí události 1
`...`	...
`$AA_MM4[<osa>]`	změřená hodnota z naprogramovaného měřicího systému při spouštěcí události 4

- v souřadném systému obrobku:

`$AA_WM1[<osa>]`	změřená hodnota z naprogramovaného měřicího systému při spouštěcí události 1
`...`	...
`$AA_WM4[<osa>]`	změřená hodnota z naprogramovaného měřicího systému při spouštěcí události 4

Poznámka

Při načítání těchto proměnných nedochází k žádnému internímu zastavení předběžného zpracování. Zastavení předběžného zpracování musí být naprogramováno na vhodném místě pomocí příkazu `STOPRE`. Jinak jsou načteny nesprávné hodnoty.

Geometrické osy / transformace

Jestliže má být měření pomocí jedné osy spuštěno pro geometrickou osu, musí být stejná měřicí úloha explicitně naprogramována také pro všechny zbývající geometrické osy. Totéž platí i pro osy, které se podílejí na transformaci.

Příklad:

```
N10 MEASA[Z]=(1,1) MEASA[Y]=(1,1) MEASA[X]=(1,1) G0 Z100
```

nebo

```
N10 MEASA[Z]=(1,1) POS[Z]=100
```

Měření se 2 měřicími systémy

Jestliže se měřicí úloha uskutečňuje se dvěma měřicími systémy, zaznamenává se pro příslušnou osu každá s obou možných spouštěcích událostí z obou měřicích systémů. Obsazení rezervovaných proměnných je předem definováno takto:

`$AA_MM1[<osa>]`	nebo	`$AA_MW1[<osa>]`	změřená hodnota z měřicího systému 1 při spouštěcí události 1
`$AA_MM2[<osa>]`	nebo	`$AA_MW2[<osa>]`	změřená hodnota z měřicího systému 2 při spouštěcí události 1
`$AA_MM3[<osa>]`	nebo	`$AA_MW3[<osa>]`	změřená hodnota z měřicího systému 1 při spouštěcí události 2
`$AA_MM4[<osa>]`	nebo	`$AA_MW4[<osa>]`	změřená hodnota z měřicího systému 2 při spouštěcí události 2

Stav měřicí sondy

Stav měřicí sondy je k dispozici v následujících systémových proměnných:

$A_PROBE[<n>]

<n>=měřicí sonda

Hodnota	Význam
1	Měřicí sonda je vychýlena
0	Měřicí sonda není vychýlena

Stav měřicí operace pro příkazy MEASA, MEAWA

Pokud je v programu zapotřebí vyhodnocování, pak je možno zjistit stav měřicí operace pomocí proměnné $AC_MEA[<n>], kde <n> = číslo měřicí sondy. Jakmile jsou úspěšně zaznamenány všechny v bloku naprogramované spouštěcí události měřicí sondy <n>, je této proměnné dosazena hodnota 1. Jinak má hodnotu 0.

Poznámka

Pokud je měření spuštěno ze synchronní akce, proměnná $AC_MEA se neaktualizuje. V tomto případě je potřeba zjistit stav nových stavových signálů PLC DB31, … DBX62.3, příp. proměnné $AA_MEAACT[<osa>], která má stejný význam.

Význam:

$AA_MEAACT==1: Měření je aktivní

$AA_MEAACT==0: Měření není aktivní

Kontinuální měření (MEAC)

V případě funkce `MEAC` jsou změřené hodnoty v souřadném systému stroje a ukládají se do zadané paměti typu FIFO[n] (zásobníková paměť). Jestliže je nastaveno, že měření má probíhat se dvěma měřicími sondami, jsou hodnoty změřené druhou sondou ukládány odděleně do pro tento účel vytvořené paměti FIFO [n+1] (nastavitelné pomocí strojního parametru).

Paměť FIFO je zásobníkovou pamětí, do které jsou změřené hodnoty ukládány v proměnné $AC_FIFO na principu "první dovnitř - první ven", viz kapitola "Pohybové synchronní akce".

Poznámka

Obsah paměti FIFO je možné ze zásobníkové paměti přečíst jen jednou. Pokud potřebujete změřené údaje používat vícekrát, musíte je přechodně uložit do uživatelské paměti.

Pokud počet změřených hodnot pro paměť FIFO překročí maximální možný počet definovaný ve strojním parametru, měření se automaticky ukončí.

Nekonečné měření je možné realizovat pomocí cyklického načítání změřených hodnot. Načítání se přitom musí uskutečňovat s minimálně stejnou četností, s jakou probíhá přísun nových změřených hodnot.

Rozpoznané chyby při programování

Jsou rozpoznávány následující chyby v programování a při jejich výskytu se vypíše chyba:

- Příkaz MEASA/MEAWA je naprogramován v jednom bloku spolu s příkazem MEAS/MEAW.

  Příklad:

  ```
  N01 MEAS=1 MEASA[X]=(1,1) G01 F100 POS[X]=100
  ```

- MEASA/MEAWA s počtem parametrů < 2 nebo > 5

  Příklad:

  ```
  N01 MEAWA[X]=(1) G01 F100 POS[X]=100
  ```

- MEASA/MEAWA se spouštěcí událostí nerovnající se 1/ -1/ 2/ -2

  Příklad:

  ```
  N01 MEASA[B]=(1,1,3) B100
  ```

- MEASA/MEAWA s nesprávným režimem

  Příklad:

  ```
  N01 MEAWA[B]=(4,1) B100
  ```

- MEASA/MEAWA s dvakrát naprogramovanou spouštěcí událostí

  Příklad:

  ```
  N01 MEASA[B]=(1,1,-1,2,-1) B100
  ```

- MEASA/MEAWA a chybějící geometrická osa

  Příklad:

  ```
  N01 MEASA[X]=(1,1) MEASA[Y]=(1,1) G01 X50 Y50 Z50 F100 ;GEO-Achse
  X/Y/Z
  ```

- Měřicí úloha u geometrických os bez jednotek

  Příklad:

  ```
  N01 MEASA[X]=(1,1) MEASA[Y]=(1,1) MEASA[Z]=(1,1,2) G01 X50 Y50 Z50
  F100
  ```

4.9 Speciální funkce pro uživatele OEM (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829)

Adresy OEM

Význam adres OEM určuje uživatel OEM. Funkce se instalují prostřednictvím cyklů překladače. Je rezervováno 5 adres OEM (`OMA1` ... `OMA5`). Identifikátory adres jsou nastavitelné. Adresy OEM jsou přípustné v každém bloku.

Rezervovaná volání G-funkcí

Pro uživatele OEM jsou vyhrazeny následující volání G-funkcí:

- `OEMIPO1`, `OEMIPO2` (ze skupiny G-funkcí č. 1)
- `G810` ... `G819` (Skupina G-funkcí č. 31)
- `G820` ... `G829` (Skupina G-funkcí č. 32)

Funkce se instalují prostřednictvím cyklů překladače.

Funkce a podprogramy

Kromě toho mohou uživatelé OEM zakládat také předdefinované funkce a podprogramy s předáváním parametrů.

Poznámka

Simulace obrobku

Až do verze SW 4.4 nebyly při simulaci obrobku podporovány žádné cykly překladače (CC), od verze SW 4.4 jsou podporovány jen vybrané.

Příkazy jazyka ve výrobním programu, které spadají do nepodporovaných cyklů překladače (`OMA1` ... `OMA5`, `OEMIPO1/2`, `G810` ... `G829`, některé procedury a funkce), pokud nejsou individuálně ošetřeny, mají proto za následek alarmové hlášení a přerušení simulace.

Řešení: Pro chybějící specifické prvky jazyka CC ve výrobním programu zajistěte individuální zpracování (zjišťování stavu $P_SIM).
Příklad:

```
N1 G01 X200 F500
IF (1==$P_SIM)
N5 X300 ;při simulaci CC nejsou aktivní
ELSE
N5 X300 OMA1=10
ENDIF
```

4.10 Snížení posuvu se zpožděním v rozích (FENDNORM, G62, G621)

Funkce

V případě automatického zpožďování v rozích se hodnota posuvu krátce před dosažením příslušného rohu snižuje podle křivky zvonovitého tvaru. Kromě toho může být pomocí nastavovaných parametrů definován přídavek rozměru pro odpovídající chování nástroje v rohu. Jedná se o následující:

- Začátek a konec snížené hodnoty posuvu
- Korekce, o kterou je posuv snížen
- Rozpoznávání odpovídajícího rohu

Rohy, kterých se toto snížení posuvu týká, budou ty, jejichž vnitřní úhel je menší než úhel stanovený příslušným nastavovaným parametrem.

Prostřednictvím předdefinované hodnoty `FENDNORM` se funkce automatické korekce v rozích vypíná.

Literatura:
/FBFA/, Příručka Popis funkcí, Dialekty ISO

Syntaxe

```
FENDNORM
G62 G41
G621
```

Význam

`FENDNORM`	Deaktivování automatického zpoždění v rozích
`G62`	Zpoždění na vnitřních rozích při aktivní korekci rádiusu nástroje
`G621`	Zpoždění na všech rozích při aktivní korekci rádiusu nástroje

Příkaz G62 je v platnosti jen na vnitřních rozích, když je

- aktivní korekce rádiusu nástroje `G41`, `G42` a
- aktivní režim řízení pohybu po dráze `G64`, `G641`

Na odpovídající roh se najíždí se sníženou hodnotou posuvu, která vyplývá z následujícího vzorce:

F * (korekce pro snížení hodnoty posuvu) * korekce snížení hodnoty posuvu

Maximálního možného snížení rychlosti posuvu je dosaženo přesně tehdy, když nástroj, vztaženo na dráhu jeho středu, dosáhne bodu, v němž se má uskutečnit změna směru na příslušném rohu.

G621 funguje analogicky k příkazu G62 na každém rohu, na který osy stanovené příkazem `FGROUP` narazí.

4.11 Programovatelné kritérium konce pohybu (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA)

Funkce

Podobně jako kritérium přechodu na další blok v případě dráhové interpolace (G601, G602 a G603) může být ve výrobním programu, příp. v synchronních akcích pro příkazové/PLC osy naprogramováno u interpolace jednotlivých os kritérium konce pohybu

V závislosti na tom, které kritérium konce pohybu je nastaveno, jsou bloky výrobního programu, příp. bloky technologických cyklů s pohyby jednotlivých os ukončeny různě rychle. Totéž platí pro PLC prostřednictvím FC15/16/18.

Syntaxe

```
FINEA[<osa>]
COARSEA[<osa>]
IPOENDA[<osa>]
IPOBRKA(<osa>[,<časový okamžik>])
ADISPOSA(<osa>[,<režim>,<velikost okna>])
```

Význam

`FINEA`: Pohybové kritérium: "jemné přesné najetí"

Platnost: modální

`COARSEA`: Pohybové kritérium: "hrubé přesné najetí"

Platnost: modální

`IPOENDA`: Pohybové kritérium: "zastavení interpolátoru"

Platnost: modální

`IPOBRKA`: Kritérium přechodu na další blok: Brzdná charakteristika

Platnost: modální

`ADISPOSA`: Toleranční okno pro kritérium konce pohybu

Platnost: modální

`<osa>`: Název kanálové osy (X, Y, ...)

`<časový okamžik>`: Časový okamžik přechodu na další blok, vztaženo ke hraně brzdné charakteristiky v %:

- 100% = začátek brzdné charakteristiky
- 0% = Konec brzdné charakteristiky, to má stejný význam jako příkaz `IPOENDA`

Typ: REAL

<režim>:	Vztah tolerančního okna		
	Rozsah hodnot:	0	Toleranční okno není aktivní
		1	Toleranční okno vztaženo na požadovanou polohu
		2	Toleranční okno vztaženo na skutečnou polohu
	Typ:	INT	
<velikost okna>:	Velikost tolerančního okna		
	Typ:	REAL	

Příklady

Příklad 1: Kritérium konce pohybu: "Zastavení interpolátoru"

Programový kód

```
; Najíždění polohovací osou X na pozici 100, rychlost 200 m/min, zrychlení 90%,
; Kritérium konce pohybu: Zastavení interpolátoru
N110 G01 POS[X]=100 FA[X]=200 ACC[X]=90 IPOENDA[X]


; synchronní akce:
; VŽDY KDYŽ: vstup 1 je nastaven
;POTOM najíždění polohovací osou X na pozici 50, rychlost 200 m/min, zrychlení 140%,
;      Kritérium konce pohybu: Zastavení interpolátoru
N120 EVERY $A_IN[1] DO POS[X]=50 FA[X]=200 ACC[X]=140 IPOENDA[X]
```

Příklad 2: Kritérium přechodu na další blok: "Brzdná charakteristika"

Programový kód	Komentář
	; Předdefinovaná nastavení jsou platná
N40 POS[X]=100	; Polohovací pohyb osou X na pozici 100 ; Kritérium přechodu na další blok: Jemné přesné najetí
N20 IPOBRKA(X,100)	; Kritérium přechodu na další blok: "Brzdná charakteristika", ; 100% = začátek brzdné charakteristiky
N30 POS[X]=200	; Přechod na další blok se uskuteční, jakmile osa X začne brzdit.
N40 POS[X]=250	; Osa X už nebrzdí na pozici 200, nýbrž jede dál ; až na pozici 250. ; Jakmile osa začne brzdit, následuje přechod na další blok
N50 POS[X]=0	; Osa X brzdí a pak jede zpět na pozici 0. ; Přechod na další blok se uskutečňuje na pozici 0 a při "jemném přesném najetí".
N60 X10 F100	; Osa X najíždí jako dráhová osa na pozici 10.

Další informace

Systémové proměnné pro kritérium konce pohybu

Kritérium konce pohybu, které je momentálně v platnosti, je možné načíst pomocí systémové proměnné $AA_MOTEND.

Literatura: /LIS2sl/, Popis seznamů, svazek 2

Kritérium přechodu na další blok: "Brzdná charakteristika" (IPOBRKA)

Jestliže je při aktivování kritéria přechodu na další blok "Brzdná charakteristika" pro volitelný bod pro přechod na další blok naprogramována hodnota, bude tato hodnota v platnosti pro následující polohovací pohyb a synchronně se zpracováním hlavní větve se bude zapisovat do strojního parametru. Pokud pro okamžik přechodu na další blok není udána žádná hodnota, použije se aktuální hodnota z nastavovaného parametru.

SD43600 $SA_IPOBRAKE_BLOCK_EXCHANGE

Při následujícím naprogramování kritéria konce pohybu osy (`FINEA`, `COARSEA` , `IPOENDA`) se funkce `IPOBRKA` pro odpovídající osu deaktivuje.

Doplňkové kritérium přechodu na další blok: "Toleranční okno" (ADISPOSA)

Pomocí příkazu `ADISPOSA` může být jako doplňkové kritérium pro přechod na další blok definováno toleranční okno okolo koncového bodu bloku (lze zvolit, zda je to skutečná poloha nebo požadovaná poloha). Pro přechod na další blok musí být splněny obě podmínky:

- Kritérium přechodu na další blok: "Brzdná charakteristika"
- Kritérium přechodu na další blok: "Toleranční okno"

Literatura

Pokud budete potřebovat další informace týkající se kritéria pro přechod polohovacími osami na další blok, viz:

- Příručka Popis funkcí, Rozšiřovací funkce; Polohovací osy (P2)
- Příručka programování, Základy; kapitola "Regulace posuvu"

4.12 Programovatelný blok parametrů servomechnismu (SCPARA)

Funkce

Pomocí příkazu SCPARA je možné ve výrobním programu nebo v synchronních akcích naprogramovat blok parametrů (skládající se ze strojních parametrů) (dříve to bylo možné jen pomocí PLC).

DB3n DBB9 Bit3

Aby nemohlo dojít k žádným konfliktům mezi PLC a NCK, je na rozhraní PLC → NCK definován další bit:

DB3n DBB9 Bit3 "Zadání bloku parametrů příkazem SCPARA blokováno".

Když je zadávání bloku parametrů pro příkaz SCPARA blokováno, nevypisuje se žádné chybové hlášení, i když je naprogramováno.

Syntaxe

```
SCPARA[<osa>]=<hodnota>
```

Význam

SCPARA	definice bloku parametrů
<osa>	Název kanálové osy (X, Y, ...)
<hodnota>	Požadovaná sada parametrů (1<= hodnota <=6)

Poznámka

Aktuální sada parametrů může být zjištěna prostřednictvím systémové proměnné $AA_SCPAR[<osa>].

U příkazů G33, G331, příp. G332 je nejvhodnější sada parametrů vybírána řídícím systémem.

Jestliže má být **datový blok servomechnismuzměněn** jak ve výrobním program, příp. v synchronní akci, tak také v PLC, musí být uživatelský program v PLC rozšířen.

Literatura:
/FB1/, Příručka Popis funkcí – Základní funkce; "Posuvy (V1)",
kapitola "Ovlivňování posuvu"

Příklad

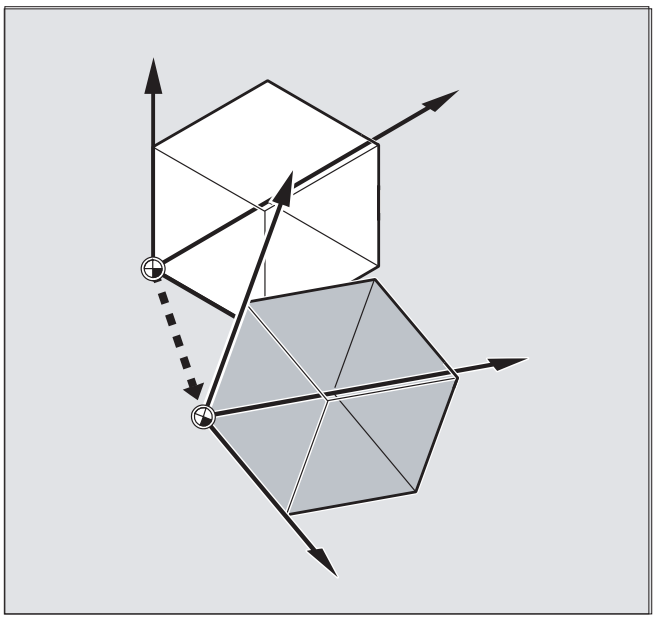
Programový kód	Komentář
...	
N110 SCPARA[X]= 3	; Pro osu X je zvolena 3. sada parametrů.
...	

Transformace souřadného systému (FRAME) 5

5.1 Transformace souřadnic pomocí proměnné typu frame

Funkce

Kromě možností programování, které už byly popsány v dokumentu Příručka programování, Základy, můžete definovat souřadné systémy také pomocí předem stanovených proměnných typu FRAME.



Jsou definovány následující souřadné systémy:

MCS: Souřadný systém stroje

BCS: Základní souřadný systém

BNS: Základní souřadný systém počátku

ENS: Nastavitelný souřadný systém

WCS: Souřadný systém obrobku

Co je to předem definovaná proměnná typu frame?

Předem definované proměnné typu FRAME jsou klíčová slova, pro která je v jazyce řídícího systému již definován odpovídající význam a se kterými lze v NC programu pracovat.

Možné proměnné typu FRAME:

- Základní frame (základní posunutí)
- nastavitelné framy
- Programovatelné framy

Přiřazení hodnot a načítání skutečné hodnoty

Souvislost mezi proměnnou typu FRAME a framem

Transformace souřadného systému může být aktivována přiřazením hodnoty framu proměnné typu FRAME.

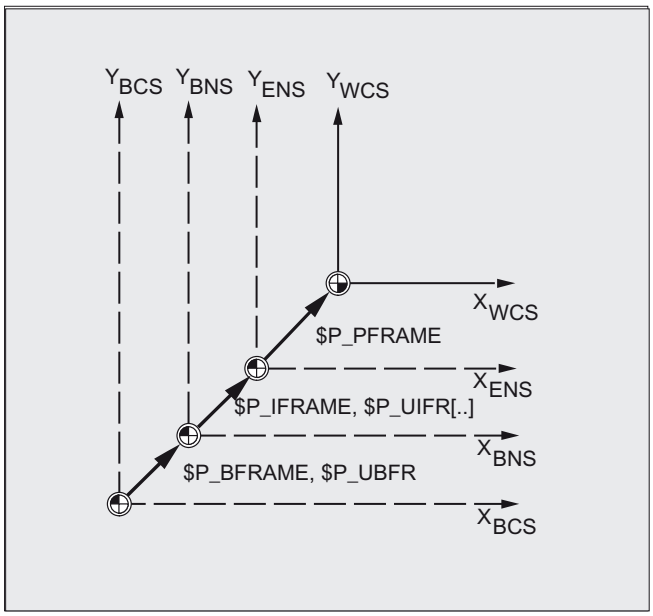
Příklad: `$P_PFRAME=CTRANS(X,10)`

Proměnná typu FRAME:

`$P_PFRAME` znamená: aktuální programovatelný frame.

Frame:

`CTRANS(X,10)` znamená: programovatelné posunutí počátku (nuly) osy X o 10 mm.



Načítání skutečných hodnot

Prostřednictvím předem definované proměnné ve výrobním programu je možné načíst momentální skutečné hodnoty souřadného systému:

$AA_IM[osa]: Načítání skutečné hodnoty v MCS

$AA_IB[osa]: Načítání skutečné hodnoty v BCS

$AA_IBN[osa:] Načítání skutečné hodnoty v BNS

$AA_IEN[osa]: Načítání skutečné hodnoty v ENS

$AA_IW[osa]: Načítání skutečné hodnoty ve WCS

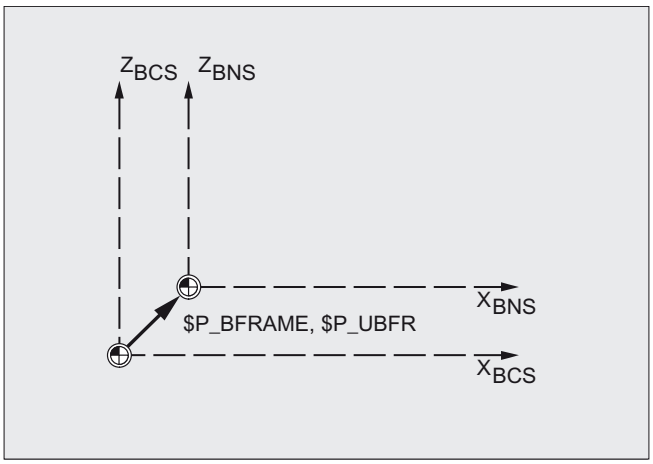
5.1.1 Předdefinované proměnné typu frame ($P_BFRAME, $P_IFRAME, $P_PFRAME, $P_ACTFRAME)

$P_BFRAME

Proměnná aktuálního základního framu, která definuje vztah mezi základním souřadným systémem (BCS) a základním souřadným systémem počátku (nuly) (BNS).

Pokud má být základní frame popsaný pomocí proměnné $P_UBFR v programu okamžitě v platnosti, musí být buď

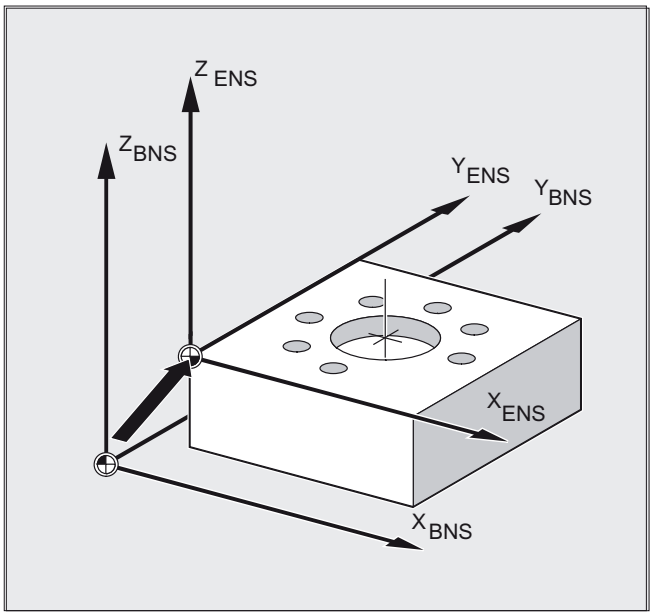
- naprogramován některý z příkazů `G500`, `G54...G599` nebo
- do proměnné `$P_BFRAME` je nutno zapsat `$ $P_UBFR`.



$P_IFRAME

Proměnná aktuálního nastavitelného framu, která definuje vztah mezi základním souřadným systémem počátku (nuly) (BNS) a nastavitelným souřadným systémem (ENS).

- `$P_IFRAME` odpovídá `$P_UIFR[$P_IFRNUM]`
- Proměnná `$P_IFRAME` obsahuje po naprogramování např.. G54 posunutí, otočení, změny měřítka a zrcadlová převrácení definovaná příkazem `G54`.

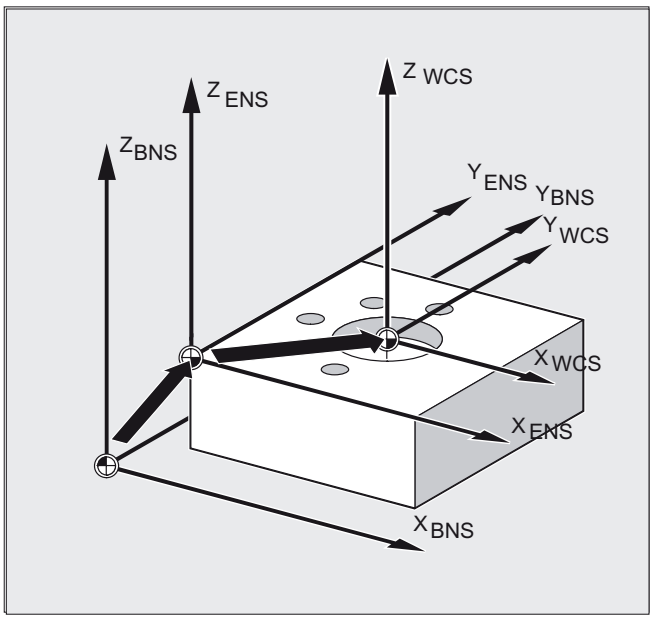


$P_PFRAME

Proměnná aktuálního programovatelného framu, která definuje vztah mezi nastavitelným souřadným systémem (ENS) a souřadným systémem obrobku (WCS).

Proměnná `$P_PFRAME` obsahuje výsledný frame, který vyplývá

- **z naprogramovaných příkazů**`TRANS/ATRANS`, `ROT/AROT`, `SCALE/ASCALE`, `MIRROR/AMIRROR`, příp.
- **z přiřazení** hodnot pomocí příkazů `CTRANS`, `CROT`, `CMIRROR`, `CSCALE` do programovatelného framu



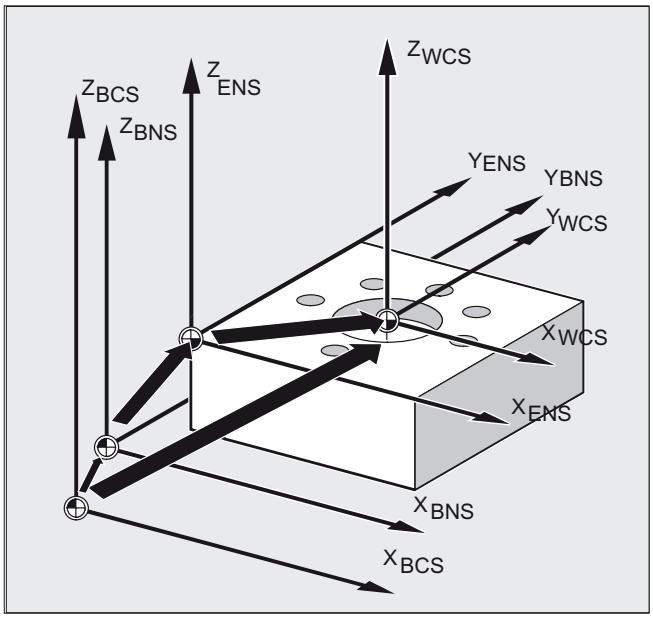
$P_ACTFRAME

Aktuální výsledný celkový frame, který vzniká superpozicí následujících objektů:

- aktuální proměnná základního framu `$P_BFRAME`
- aktuální proměnná nastavitelného framu `$P_IFRAME` se systémovým framem
- aktuální proměnná programovatelného framu `$P_PFRAME` se systémovým framem

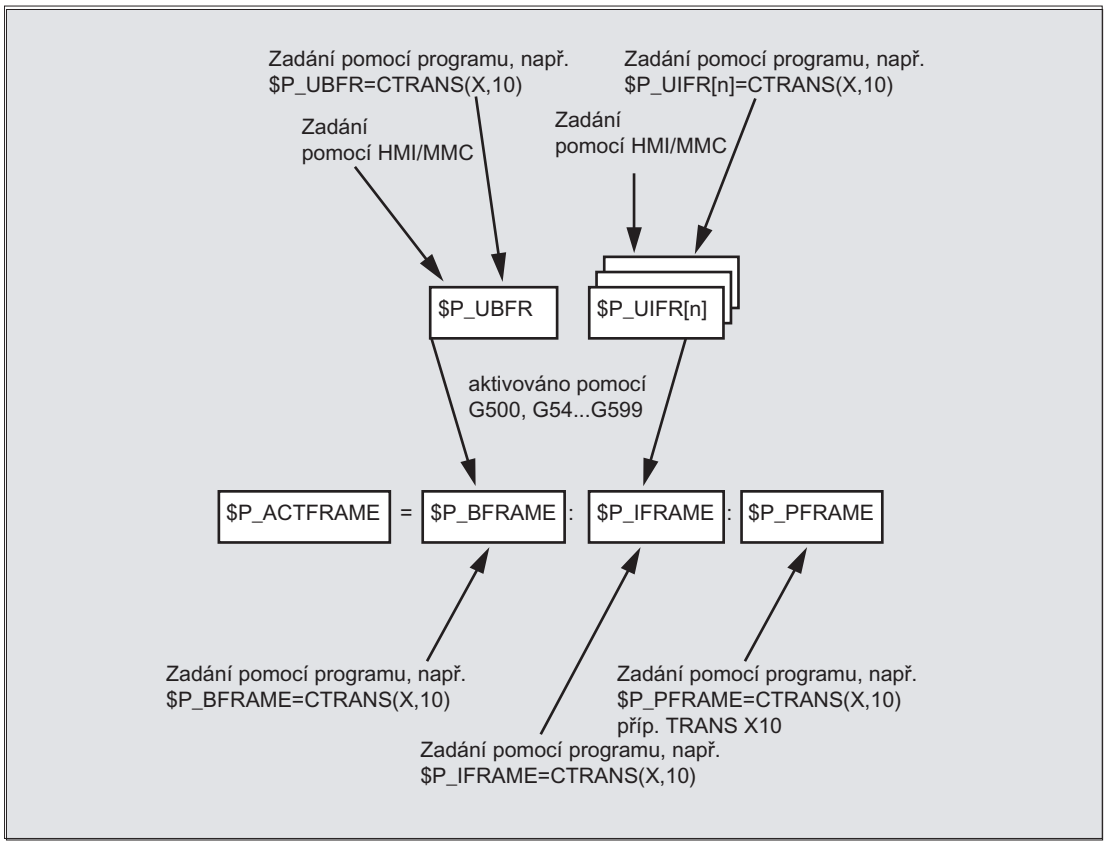
atd. Systémové framy, viz kapitola "Framy aktivní v kanálu".

Proměnná `$P_ACTFRAME` popisuje momentálně platný počátek souřadného systému obrobku.



Jestliže se `$P_BFRAME`, `$P_IFRAME` nebo `$P_PFRAME` změní, hodnota `$P_ACTFRAME` se znovu vypočítá.

`$P_ACTFRAME` odpovídá `$P_BFRAME:$P_IFRAME:$P_PFRAME`



Základní frame a nastavitelný frame zůstávají v platnosti i po resetu, pokud je strojní parametr MD 20110 RESET_MODE_MASK nastaven následujícím způsobem:

Bit0=1, Bit14=1 → `$P_UBFR` (základní frame) je v platnosti

Bit0=1, Bit5=1 → `$P_UIFR[$P_UIFRNUM]` (nastavitelný frame) je v platnosti

Předem definované nastavitelné framy $P_UBFR

Pomocí $P_UBFR se základní frame naprogramuje,nestává se ale současně aktivním ve výrobním programu. Základní frame popisovaný parametrem $P_UBFR se započítá tehdy, pokud nastane některá z následující situací:

- Byl proveden reset a bit 0 a bit 14 v parametru MD RESET_MODE_MASK jsou nastaveny
- byl zpracován některý z příkazů `G500`, `G54...G599`

Předem definované nastavitelné framy $P_UIFR[n]

Prostřednictvím předem definovaných framových proměnných $P_UIFR[n] mohou být ve výrobním programu čtena a zapisována nastavitelná posunutí počátku `G54` až `G599`.

Tato proměnná má strukturu jednorozměrného pole typu FRAME s názvy `$P_UIFR[n]`.

Přiřazení k příkazům F-funkcí

Standardně existuje 5 nastavitelných framů `$P_UIFR[0]... $P_UIFR[4]`, příp. 5 příkazů G-funkcí stejného významu - `G500` a `G54` až `G57`, do jejichž adres mohou být ukládány hodnoty.

`$P_IFRAME=$P_UIFR[0]` odpovídá `G500`

`$P_IFRAME=$P_UIFR[1]` odpovídá `G54`

`$P_IFRAME=$P_UIFR[2]` odpovídá `G55`

`$P_IFRAME=$P_UIFR[3]` odpovídá `G56`

`$P_IFRAME=$P_UIFR[4]` odpovídá `G57`

Prostřednictvím strojního parametru je možné počet framů změnit.

`$P_IFRAME=$P_UIFR[5]` odpovídá `G505`

`... ... ...`

`$P_IFRAME=$P_UIFR[99]` odpovídá `G599`

Poznámka

Tímto způsobem je možné vytvořit celkem 100 souřadných systémů, které pak mohou být i přes hranice programu vyvolávány např. jako počátky souřadného systému pro různá zařízení.

⚠ POZOR

Programování proměnných typu FRAME a framů vyžaduje v NC programu samostatný NC blok. **Výjimka:** Programování nastavitelných framů pomocí příkazů `G54, G55, ...`

5.2 Proměnné typu frame/Přiřazování hodnot framům

5.2.1 Přímé přiřazení hodnoty (hodnota osy, úhlu, měřítko)

Funkce

V NC programu můžete dosazovat hodnoty přímo do framů nebo do proměnných typu FRAME.

Syntaxe

```
$P_PFRAME=CTRANS (X, hodnota osy, Y, hodnota osy, Z, hodnota osy, …)
$P_PFRAME=CROT (X, úhel, Y, úhel, Z, úhel, …)
$P_UIFR[..]=CROT (X, úhel, Y, úhel, Z, úhel, …)
$P_PFRAME=CSCALE (X, měřítko, Y, měřítko, Z, měřítko, …)
$P_PFRAME=CMIRROR (X, Y, Z)
```

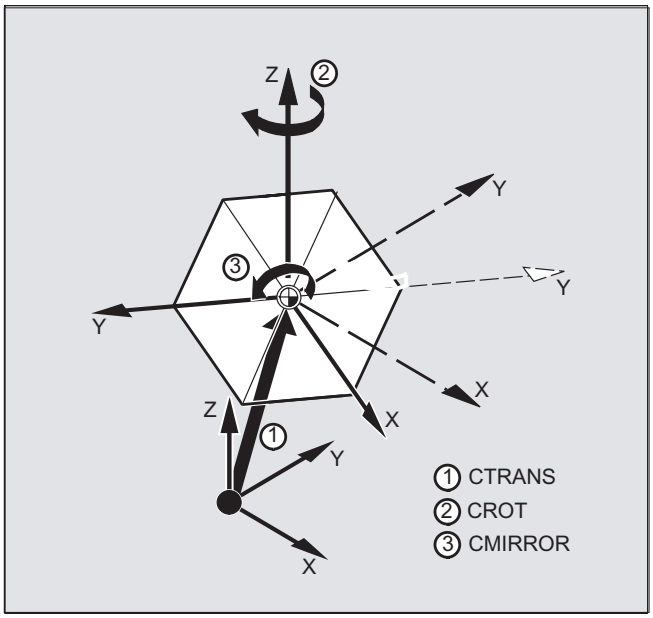
Programování proměnné $P_BFRAME se uskutečňuje analogicky jako $P_PFRAME.

Význam

`CTRANS`	Posunutí v uvedených osách
`CROT`	Otočení okolo uvedených os
`CSCALE`	Změna měřítka v uvedených osách
`CMIRROR`	Zrcadlové převrácení pro uvedenou osu
`X Y Z`	Hodnota posunutí ve směru uvedené geometrické osy
`Hodnota osy`	Přiřazení hodnoty posunutí ose
`Úhel`	Přiřazení úhlu otočení okolo uvedených os
`Měřítko`	Změna měřítka

Příklad

Přiřazením hodnoty do aktuálního programovatelného framu se posunutí, otočení a zrcadlové převrácení aktivují.




```
N10 $P_PFRAME=CTRANS(X,10,Y,20,Z,5):CROT(Z,45):CMIRROR(Y)
```

Dosazení jiných hodnot do složek otočení daného framu

Pomocí příkazu CROT jsou dosazeny hodnoty všem třem komponentům UIFR

Programový kód	**Komentář**
`$P_UIFR[5] = CROT(X, 0, Y, 0, Z, 0)`	
`N100 $P_UIFR[5, y, rt]=0`	
`N100 $P_UIFR[5, x, rt]=0`	
`N100 $P_UIFR[5, z, rt]=0`	

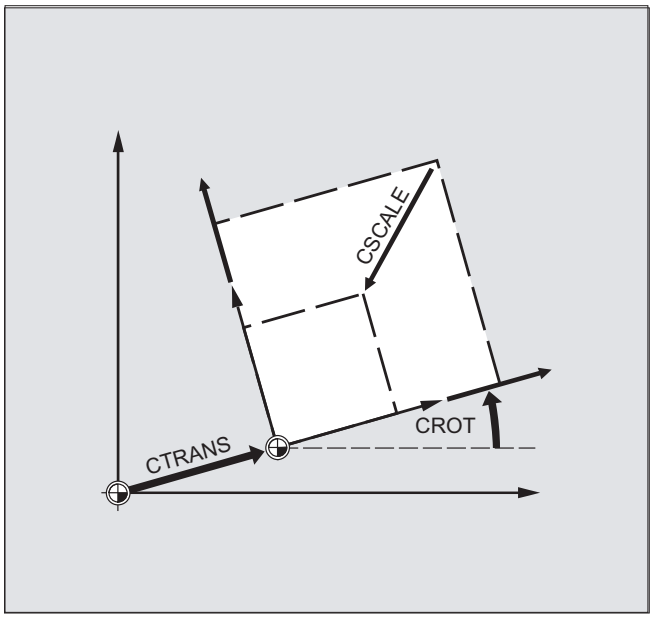
Popis

Můžete naprogramovat také větší počet matematických výrazů za sebou.

Příklad:

```
$P_PFRAME=CTRANS(...):CROT(...):CSCALE...
```

Mějte na paměti, že příkazy musí být vzájemně spojeny operátorem zřetězení, což je dvojtečka (…):(…). Díky tomu jsou příkazy za prvé spolu spojeny a za druhé jsou uskutečněny aditivně v naprogramované posloupnosti.



Poznámka

Hodnoty naprogramované prostřednictvím výše uvedených příkazů jsou přiřazeny framu a uloženy do paměti.

Aktivními se tyto hodnoty stávají teprve tehdy, když je frame přiřazen aktivní proměnné typu FRAME `$P_BFRAME`, příp. `$P_PFRAME`.

5.2.2 Načítání a editace složek framu (TR, FI, RT, SC, MI)

Funkce

Pokud potřebujete, můžete mít přístup k **jednotlivým** datům framu, např. k hodnotám pro určité posunutí nebo úhel otočení. Tyto hodnoty můžete změnit nebo je můžete přiřadit jiné proměnné.

Syntaxe

`R10=$P_UIFR[$P_UIFNUM,X,RT]`	Úhel otočení RT okolo osy X z momentálně platného nastavitelného posunutí počátku $P_UIFRNUM má být přiřazen proměnné R10.
`R12=$P_UIFR[25,Z,TR]`	Hodnota posunutí TR ve směru osy Z z datového bloku nastavitelného framu č. 25 má být přiřazena proměnné R12.
`R15=$P_PFRAME[Y,TR]`	Hodnota posunutí TR ve směru osy Z z aktuálního programovatelného framu má být přiřazena proměnné R15.
`$P_PFRAME[X,TR] = 25`	Hodnota posunutí TR ve směru osy X z aktuálního programovatelného framu má být změněna. Od tohoto okamžiku platí X25.

Význam

`$P_UIFRNUM`	Pomocí této proměnné se automaticky vytváří vztah k momentálně platnému nastavitelnému posunutí počátku.
`P_UIFR[n,…,…]`	Zadáním čísla framu n se otevírá přístup k nastavitelnému framu č. n.
	Zadání složky, která má být načtena nebo změněna.
`TR`	TR - posunutí
`FI`	FI - jemné posunutí
`RT`	RT - otočení
`SC`	SC - změna měřítka
`MI`	MI - zrcadlové převrácení
`X Y Z`	Kromě toho (viz příklady) je zapotřebí zadat odpovídající osy X, Y, Z.

Rozsah hodnot pro otočení RT

Otáčení okolo 1. geometrické osy:	-180° až +180°
Otáčení okolo 2. geometrické osy:	-90° až +90°
Otáčení okolo 3. geometrické osy:	-180° až +180°

Popis

Vyvolání framu

Prostřednictvím systémové proměnné $P_UIFRNUM můžete mít přístup přímo k momentálně nastavenému posunutí počátku, které bylo zadáno pomocí příkazů $P_UIFR příp. G54, G55, … (parametr $P_UIFRNUM obsahuje číslo momentálně nastaveného framu).

Všechny ostatní uložené nastavitelné framy $P_UIFR vyvoláte prostřednictvím zadání odpovídajícího čísla $P_UIFR[n].

Pro předem definované framy a vlastní definované framy zadejte příslušné názvy, např. $P_IFRAME.

Vyvolávání dat

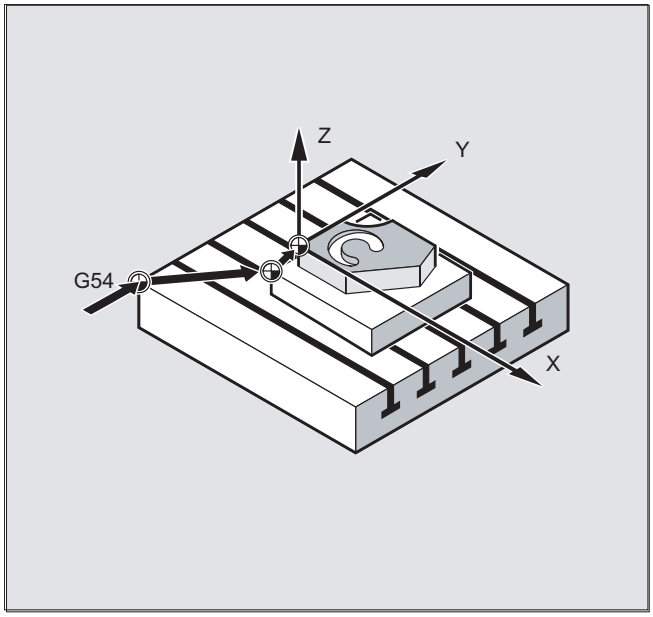
V hranatých závorkách jsou uvedeny hodnoty názvu osy a složky framu, kterým chcete mít přístup nebo které si přejete změnit, např. [X, RT] nebo [Z, MI].

5.2.3 Slučování kompletních framů

Funkce

V NC programu může být kompletní frame přiřazen jinému framu nebo framy mohou být vzájemně zřetězovány.

Zřetězení framů je užitečné např. pro popis většího počtu obrobků, které jsou uspořádány na paletě a které mají být opracovány v rámci jedné výrobní operace.



Za účelem popisu obráběcích úloh na paletách by mohly složky framu obsahovat např. pouze určité dílčí hodnoty, jejichž zřetězením se vygenerují různé počátky souřadného systému obrobku.

Syntaxe

Přiřazování framů

```
DEF FRAME EINSTELLUNG1                    Aktuálnímu programovatelnému framu
EINSTELLUNG1=CTRANS(X,10)                 jsou přiřazeny hodnoty framu
$P_PFRAME=EINSTELLUNG1                    EINSTELLUNG1, který jste si sami
                                          definovali.
DEF FRAME EINSTELLUNG4
EINSTELLUNG4=$P_PFRAME                    Aktuální programovatelný frame se
$P_PFRAME=EINSTELLUNG4                    dočasně ukládá a v případě potřeby
                                          může být znovu vyvolán.
```

Řetězce framů

Framy jsou vzájemně zřetězeny v naprogramované posloupnosti, takže složky framů, jako např. posunutí, otočení atd., jsou uskutečňovány aditivně jedna po druhé.

$P_IFRAME=$P_UIFR[15]:$P_UIFR[16]	Parametr $P_UIFR[15] obsahuje např. data pro posunutí počátku. Potom se zpracují data z parametru $P_UIFR[16], např. data pro otočení, tak, aby byla superponována s původními daty.
$P_UIFR[3]=$P_UIFR[4]:$P_UIFR[5]	Nastavitelný frame 3 vznikne zřetězením nastavitelných framů 4 a 5.

Poznámka

Mějte na paměti, že framy musí být vzájemně spojeny operátorem zřetězení, což je dvojtečka "**:**".

5.2.4 Definice nového framu (DEF FRAME)

Funkce

Vedle již dříve popisovaných předem definovaných, nastavitelných framů máte také možnost vytvářet nové framy. Přitom se jedná o proměnnou typu FRAME, kterou definujete a které můžete přiřadit libovolný název.

Svému framu můžete v NC programu dosazovat hodnoty pomocí funkcí CTRANS, CROT, CSCALE, CMIRROR.

Syntaxe

```
DEF FRAME PALETTE1
PALETTE1=CTRANS(…):CROT(…)…
```

Význam

`DEF FRAME`	Vytvoření nového framu.
`PALETTE1`	Název nového framu
`=CTRANS(...): CROT(...)...`	Přiřazení hodnot možným funkcím.

5.3 Hrubé a jemné posunutí (CFINE, CTRANS)

Funkce

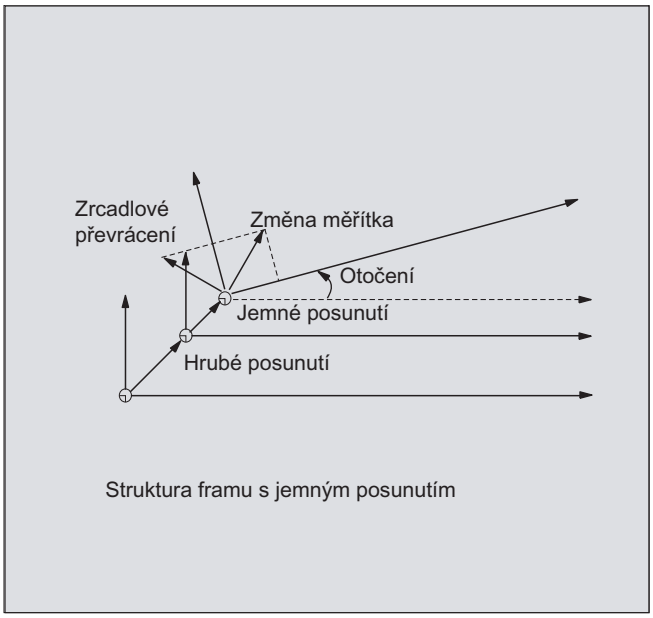
Jemné posunutí

Pomocí příkazu `CFINE(X, ...,Y ...)` může být naprogramováno jemné posunutí základního framu a všech nastavitelných framů.

Jemné posunutí se může uskutečňovat jen tehdy, pokud je nastaveno MD18600 $MN_MM_FRAME_FINE_TRANS=1.

Hrubé posunutí

Pomocí příkazu `CTRANS(...)` je definováno hrubé posunutí..



Struktura framu s jemným posunutím

Hrubé a jemné posunutí se přičítají do celkového posunutí.

Syntaxe

```
$P_UBFR=CTRANS(x, 10) : CFINE(x, 0.1)
: CROT(x, 45)                          ;Zřetězení posunutí,
                                       ;jemného posunutí a otočení
$P_UIFR[1]=CFINE(x, 0.5 y, 1.0, z,
0.1)                                   ;Celkový frame se přepíše příkazem
                                       ;CFINE, včetně hrubého posunutí
```

Přístup k jednotlivým složkám jemného posunutí se uskutečňuje prostřednictvím zadání složky FI (Translation Fine).

```
DEF REAL FINEX                          ;Definice proměnné FINEX
FINEX=$P_UIFR[$P_UIFNUM, x, FI]         ;Načtení jemného posunutí
                                        ;prostřednictvím proměnné FINEX
FINEX=$P_UIFR[3, x, FI]$P               ;Načtení jemného posunutí osy X ve
                                        ;3. framu pomocí proměnné FINEX
```

Význam

`CFINE(x, hodnota, y, hodnota, z, hodnota)`	Jemné posunutí pro více os. Aditivní posunutí (Translation).
`CTRANS(x, hodnota, y, hodnota, z, hodnota)`	Hrubé posunutí pro více os. Absolutní posunutí (Translation).
`x y z`	Posunutí počátku v osách (max. 8 os).
`Hodnota`	Složka posunutí

Výrobce stroje

Pomocí strojního parametru MD18600 $MN_ MM_FRAME_FINE_TRANS mohou být pro konfiguraci jemného posunutí nastaveny tyto varianty:

0:
Jemné posunutí nemůže být zadáno, příp. nemůže být naprogramováno. Příkazy G58 a G59 nelze používat.

1:
Jemné posunutí pro nastavitelné framy, základní frame, programovatelné framy, příkazy G58 a G59 je možné zadávat, příp. je možné je naprogramovat.

Popis

Jemné posunutí změněné prostřednictvím HMI se stává aktivním teprve po aktivování odpovídajícího framu, tzn. aktivování se uskutečňuje příkazy G500, G54...G599. Aktivované jemné posunutí framu je aktivní tak dlouho, dokud je příslušný frame aktivní.

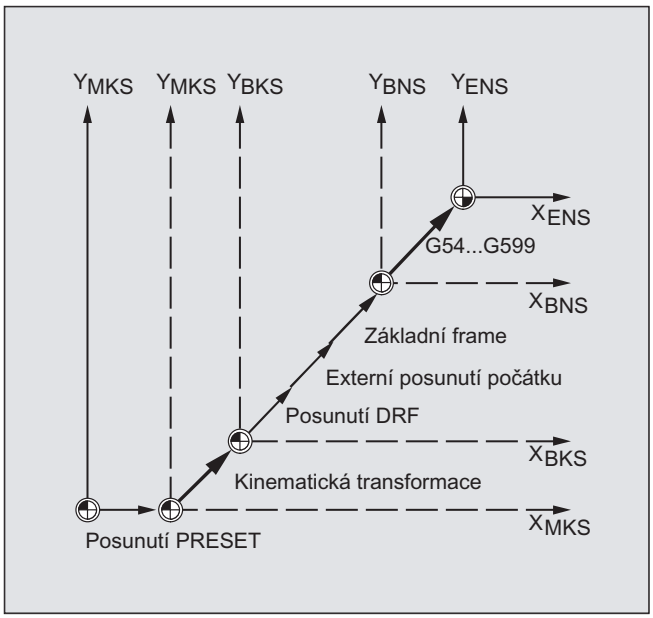
Žádná složka jemného posunutí není součástí programovatelného framu. Pokud je programovatelnému framu přiřazen frame s jemným posunutím, potom se součtem hrubého a jemného posunutí vytvoří jeho celkové posunutí. Při načítání programovatelného framu je jemné posunutí vždycky nulové.

5.4 Externí posunutí počátku

Funkce

Pomocí této funkce máte k dispozici další možnost, jak posouvat počátek souřadného systému mezi základním souřadným systémem a souřadným systémem obrobku.

U externího posunutí počátku mohou být programovány pouze lineární posunutí.



Programování

Programování hodnot posunutí, $AA_ETRANS, se uskutečňuje dosazením příslušných hodnot osovým systémovým parametrům.

Přiřazení hodnot posunutí

```
$AA_ETRANS[osa]=RI
```

RI je matematická proměnná typu REAL, která obsahuje novou hodnotu.

Externí posunutí se zpravidla nezadává ve výrobním programu, nýbrž se nastavuje pomocí PLC.

Poznámka

Hodnota zapisovaná ve výrobním programu se stává platnou teprve tehdy, když je na rozhraní
VDI (rozhraní NCU-PLC) aktivován odpovídající signál.

5.5 Předvolba posunutí (PRESETON)

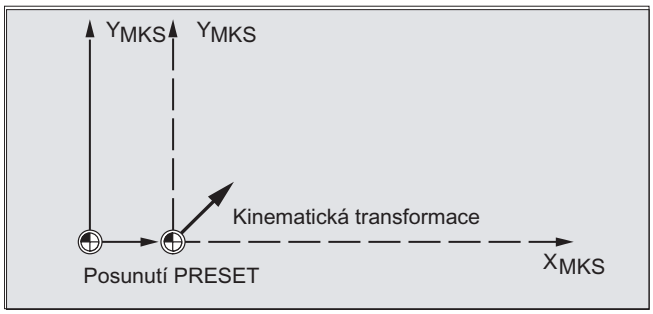
Funkce

Pro speciální aplikace se může ukázat jako užitečné přiřadit strojní ose, kterou už bylo najeto na referenční bod, pomocí příkazu `PRESETON` novou skutečnou hodnotu. Tato hodnota odpovídá posunutí počátku (nuly) v souřadném systému stroje.

⚠ POZOR

Po příkazu `PRESETON` se strojní osa nachází ve stavu "nebylo najeto na referenční bod". Z tohoto důvodu se doporučuje používat tuto funkci pouze pro osy, pro které není najíždění na referenční bod povinné. Jestliže si přejete opět obnovit původní souřadný systém stroje, je nutné příslušnou strojní osou znovu najet na referenční bod, např. pomocí příkazu G74 (Najíždění na referenční bod).

Literatura: Příručka programování, Základy, Doplňkové příkazy, Najíždění na referenční bod (G74)



Syntaxe

```
PRESETON(<Osa>, <Hodnota>, ...)
```

Význam

`PRESETON`	Nastavení skutečné hodnoty
`<osa>`	identifikátor osy stroje
`<hodnota>`	Nová skutečná hodnota strojní osy v souřadném systému stroje

Poznámka

Nastavení skutečné hodnoty prostřednictvím synchronních akcí by se mělo provádět pouze s klíčovým slovem `WHEN` nebo `EVERY`.

Příklad

Geometrická osa: A, odpovídající strojní osa: X1

Programový kód	**Komentář**
`N10 G0 A100`	`; Osa A najíždí na pozici 100.`

`N20 PRESETON(X1,50)`	`; Osa stroje X1 obsahuje na pozici 100 novou skutečnou hodnotu 50 => nová vypisovaná skutečná hodnota:`
	`- Osa X1, MCS: 50`
	`- Osa A, WCS: 50`

`N30 A100`	`; Osa A se pohybu o 50 mm na pozici 100.`

5.6 Výpočet framu na základě 3 změřených bodů v prostoru (MEAFRAME)

Funkce

Příkaz MEAFRAME je rozšířením jazyka systému 840 pro podporu měřicích cyklů.

Funkce MEAFREAME vypočítává frame na základě tří ideálních a vzájemně korespondujících změřených bodů.

Když je obrobek nastaven do polohy pro obrábění, je jeho pozice vzhledem ke kartézskému souřadnému systému stroje vůči jeho ideální poloze zpravidla jednak posunutá, ale i pootočená. Pro přesné opracování nebo měření je zapotřebí uskutečnit buď nákladné fyzické nastavování polohy nebo změny pohybů ve výrobním programu.

Frame může být definován nasnímáním trojice bodů v prostoru, jejichž ideální pozice je známa. Pro snímání polohy se používají buď dotykové nebo optické snímače, speciální na nosné destičce přesně fixované díry nebo měřicí kuličky.

Syntaxe

```
MEAFRAME IDEAL_POINT,MEAS_POINT,FIT_QUALITY)
```

Význam

`MEAFRAME`	Výpočet framu na základě 3 změřených bodů v prostoru.	
`IDEAL_POINT`	Rozměr pole typu REAL, které obsahuje tři souřadnice ideálních bodů.	
`MEAS_POINT`	Rozměr pole typu REAL, které obsahuje tři souřadnice změřených bodů.	
`FIT_QUALITY`	Proměnná typu REAL,	poskytuje následující informace o výsledku:
		Ideální body leží příliš blízko jedné přímky:
	-1:	Frame nemohl být vypočítán. Výsledná proměnná typu FRAME obsahuje neutrální frame.
	-2:	Změřené body leží příliš blízko jedné přímky: Frame nemohl být vypočítán. Výsledná proměnná typu FRAME obsahuje neutrální frame.
	-4:	Výpočet rotační matice skončil z nějakého jiného důvodu neúspěchem.
	Kladná hodnota:	Součet deformací (vzdálenosti mezi body), které jsou zapotřebí pro převedení změřeného trojúhelníku na trojúhelník, který se blíží ideálnímu.

Poznámka

Jakost měření

Aby změřené souřadnice bylo možné pomocí kombinace otočení/posunutí přiřadit ideálním souřadnicím, musí se trojúhelník tvořený změřenými body co možno nejvíce shodovat s ideálním trojúhelníkem. To bude zajištěno kompenzačním algoritmem, který se snaží minimalizovat součet druhých mocnin odchylek, kterými se změřený trojúhelník převádí na ideální trojúhelník.

Případně potřebná deformace poloh změřených bodů může posloužit jako ukazatel jakosti měření a z tohoto důvodu se ukládá jako doplňková proměnná funkce `MEAFRAME`.

Poznámka

Frame vytvořený pomocí funkce `MEAFRAME` může být pomocí funkce `ADDFRAME` transformován do jiného framu v řetězci framů.
Viz příklad: Zřetězení framů "Zřetězení s příkazem ADDFRAME".

Pokud budete potřebovat informace týkající se parametrů k příkazům `ADDFRAME(FRAME, STRING)`, viz: /FB1/, Příručka Popis funkcí, Základní funkce; "Osy, souřadné systémy, framy" (K2), kapitola "Zřetězení framů".

Příklad

Programový kód	Komentář
	; Výrobní program 1
DEF FRAME CORR_FRAME	

Nastavení měřicích bodů

Programování	Komentář
DEF REAL IDEAL_POINT[3,3] = SET(10.0,0.0,0.0, 0.0,10.0,0.0, 0.0,0.0,10.0)	
DEF REAL MEAS_POINT[3,3] = SET (10.1,0.2,-0.2, -0.2,10.2,0.1, -0.2,0.2,9.8)	; pro testování
DEF REAL FIT_QUALITY = 0	
DEF REAL ROT_FRAME_LIMIT = 5	; povolené pootočení polohy součásti max. 5 stupňů
DEF REAL FIT_QUALITY_LIMIT = 3	; povolené posunutí mezi ideálním a změřeným trojúhelníkem max. 3 mm
DEF REAL SHOW_MCS_POS1[3]	
DEF REAL SHOW_MCS_POS2[3]	
DEF REAL SHOW_MCS_POS3[3]	

Programový kód	Komentář
N100 G01 G90 F5000	
N110 X0 Y0 Z0	

Programový kód	**Komentář**
N200 CORR_FRAME=MEAFRAME(IDEAL_POINT,MEAS _POINT,FIT_QUALITY)	
N230 IF FIT_QUALITY < 0	
SETAL(65000)	
GOTOF NO_FRAME	
ENDIF	
N240 IF FIT_QUALITY > FIT_QUALITY_LIMIT	
SETAL(65010)	
GOTOF NO_FRAME	
ENDIF	
N250 IF CORR_FRAME[X,RT] > ROT_FRAME_LIMIT	; Omezení 1. úhlu RPY
SETAL(65020)	
GOTOF NO_FRAME	
ENDIF	
N260 IF CORR_FRAME[Y,RT] > ROT_FRAME_LIMIT	; Omezení 2. úhlu RPY
SETAL(65021)	
GOTOF NO_FRAME	
ENDIF	
N270 IF CORR_FRAME[Z,RT] > ROT_FRAME_LIMIT	; Omezení 3. úhlu RPY
SETAL(65022)	
GOTOF NO_FRAME	
ENDIF	
N300 $P_IFRAME=CORR_FRAME	; Aktivování nasnímaného framu s nastavitelným framem
	; Kontrola framu najetím geometrickými osami na ideální body
N400 X=IDEAL_POINT[0,0] Y=IDEAL_POINT[0,1] Z=IDEAL_POINT[0,2]	
N410 SHOW_MCS_POS1[0]=$AA_IM[X]	
N420 SHOW_MCS_POS1[1]=$AA_IM[Y]	
N430 SHOW_MCS_POS1[2]=$AA_IM[Z]	
N500 X=IDEAL_POINT[1,0] Y=IDEAL_POINT[1,1] Z=IDEAL_POINT[1,2]	
N510 SHOW_MCS_POS2[0]=$AA_IM[X]	
N520 SHOW_MCS_POS2[1]=$AA_IM[Y]	
N530 SHOW_MCS_POS2[2]=$AA_IM[Z]	
N600 X=IDEAL_POINT[2,0] Y=IDEAL_POINT[2,1] Z=IDEAL_POINT[2,2]	
N610 SHOW_MCS_POS3[0]=$AA_IM[X]	
N620 SHOW_MCS_POS3[1]=$AA_IM[Y]	
N630 SHOW_MCS_POS3[2]=$AA_IM[Z]	
N700 G500	; Deaktivování nastavitelného framu nulovým framem (předem není dosazena žádná hodnota)
No_FRAME	; Deaktivování nastavitelného framu nulovým framem (předem není dosazena žádná hodnota)
M0	
M30	

Příklad zřetězení framů

Zřetězení framů MEAFRAME za účelem korekce

Výsledkem funkce `MEAFRAME( )` je korekční frame. Pokud je tento korekční grame zřetězen s nastavitelným framem `$P_UIFR[1]`, který byl aktivní při vyvolání funkce např. `G54`, je takto získán nastavitelný frame pro další přepočítávání pro potřeby posuvů nebo obrábění.

Zřetězení s příkazem ADDFRAME

Pokud se má tento korekční frame uplatnit v řetězci framů na nějakém jiném místě nebo pokud jsou před nastavitelným framem aktivní ještě nějaké jiné framy, potom je možné pro začlenění tohoto framu do řetězce v základním framu kanálu nebo v systémovém framu použít funkci `ADDFRAME( )`.

Ve framu přitom nesmí být aktivní:

- Zrcadlové převrácení s příkazem `MIRROR`
- Změna měřítka s příkazem `SCALE`

Vstupními parametry pro požadovanou a skutečnou hodnotu jsou souřadnice obrobku. V základním systému řídícího systému je třeba zadávat tyto souřadnice vždy

- v metrických jednotkách nebo v palcích `(G71/G70)` a jako
- rozměry vztažené k rádiusu `DIAMOF`

.

5.7 Globální framy v NCU

Funkce

Pro každou NCU existuje jen jeden globální frame platný pro všechny kanály. Globální framy v NCU mohou být ze všech kanálů čteny a všechny kanály do nich mohou také zapisovat. Aktivování globálního framu v NCU se uskutečňuje v příslušném kanálu.

Prostřednictvím globálního framu mohou být pro **kanálové osy a osy stroje** realizována posunutí, změny měřítka a zrcadlová převrácení.

Geometrické souvislosti a řetězce framů

V případě globálních framů neexistuje žádná geometrická souvislost mezi osami. Z tohoto důvodu není možné provádět otočení nebo používat programování identifikátorů geometrických os.

- Na globální framy se nedají používat žádná otočení. Naprogramování otočení je odmítnuto a aktivuje se přitom alarm:"18310 Kanál %1 blok %2 Frame: Otočení nepřípustné".
- Zřetězení globálních framů a kanálových framů je možné. Výsledný frame obsahuje všechny složky framu včetně otočení pro všechny osy. Přiřazení framu se složkami otočení globálnímu gramu je odmítnuto, přičemž se aktivuje alarm "Frame: Otočení nepřípustné".

Globální framy v NCU

Globální základní frame v NCU $P_NCBFR[n]

V konfiguraci může být nastaveno až 8 globálních základních framů NCU.

Současně mohou být k dispozici kanálové základní framy.

Globální framy mohou být ze všech kanálů NCU čteny a všechny kanály NCU do nich mohou také zapisovat. Při zápisu do globálního framu se musí uživatel postarat o koordinaci kanálů. To může být realizováno např. značkami pro čekání `(WAITMC)`.

Výrobce stroje

Počet globálních základních framů se nastavuje v konfiguraci pomocí strojního parametru, viz: /FB1/, Příručka Popis funkcí, Základní funkce; "Osy, souřadné systémy, framy" (K2).

V NCU globální nastavitelné framy $P_UIFR[n]

Všechny nastavitelné framy `G500`, `G54...G599` mohou být konfigurovány buď jako globální v NCU nebo jako kanálové.

Výrobce stroje

Prostřednictvím strojního parametru $MN_MM_NUM_GLOBAL_USER_FRAMES mohou být všechny nastavitelné framy překonfigurovány na globální framy.

Jako identifikátory os se u příkazů pro programování framů mohou používat identifikátory kanálových os a identifikátory os stroje. Programování geometrických os je odmítnuto, přičemž se aktivuje alarm.

5.7.1 Specifické kanálové framy ($P_CHBFR, $P_UBFR)

Funkce

Nastavitelné framy nebo základní framy mohou být čteny nebo se do nich může zapisovat

- prostřednictvím výrobního programu a
- prostřednictvím BTSS

pomocí zásahů např. na HMI Advanced a v PLC.

Pro globální framy je možné také jemné posunutí. Potlačení globálních framů se uskutečňuje stejně jako u kanálových framů, tedy pomocí příkazů G53, G153, SUPA a G500.

Výrobce stroje

Pomocí parametru MD28081 MM_NUM_BASE_FRAMES může být v konfiguraci nastaven počet základních framů v kanálu. Standardní konfigurace je nastavena tak, aby v každém kanálu existoval nejméně jeden základní frame. V každém kanálu je možných maximálně 8 základních framů. Kromě 8 základních framů může v kanálu existovat ještě dalších 8 globálních základních framů v NCU.

Kanálové framy

$P_CHBFR[n]

Pomocí systémové proměnné $P_CHBFR[n] je možné zkladní framy číst a zapisovat do nich. Při zápisu do základního framu není zřetězený celkový základní frame ještě aktivován; toto aktivování se uskuteční teprve se zpracováním některého z příkazů G500, G54...G599. Tato proměnná slouží především jako paměť pro operace zápisu do základního framu z HMI nebo z PLC. Tyto proměnné typu FRAME se ukládají prostřednictvím zálohování dat.

První základní frame v kanálu

Zápis do předem definované proměnné $P_UBFR neaktivuje základní frame s indexem pole 0 v daný okamžik; toto aktivování se uskuteční teprve se zpracováním některého z příkazů G500, G54...G599. Do této proměnné lze zapisovat také v programu a lze ji tam i číst.

$P_UBFR

Proměnná $P_UBFR je identická s proměnnou $P_CHBFR[0]. Standardně existuje v kanálu vždy jeden základní frame, takže systémová proměnná je kompatibilní i se staršími verzemi. Pokud žádný kanálový základní frame neexistuje, při zápisu nebo při čtení se aktivuje alarm: "Frame: Nepřípustný příkaz".

5.7.2 Framy aktivní v kanálu

Funkce

V kanálu platné framy jsou ve výrobním programu zadávány prostřednictvím příslušných systémových proměnných těchto framů. K tomu patří i systémové framy. Pomocí těchto systémových proměnných je možné ve výrobním programu načíst aktuální systémový frame a lze do něj i zapisovat.

Aktuální framy aktivní v kanálu

Přehled

Aktuální systémové framy	pro:
$P_PARTFRAME	TCARR a PAROT
$P_SETFRAME	Dosazení skutečné hodnoty a škrábnutí
$P_EXTFRAME	Externí posunutí počátku
$P_NCBFRAME[n]	Aktuální globální základní framy NCU
$P_CHBFRAME[n]	Aktuální kanálové základní framy
$P_BFRAME	Aktuální 1. základní frame v kanálu
$P_ACTBFRAME	Celkový základní frame
$P_CHBFRMASK a $P_NCBFRMASK	Celkový základní frame
$P_IFRAME	Aktuální nastavitelný frame
Aktuální systémové framy	pro:
$P_TOOLFRAME	TOROT a TOFRAME
$P_WPFRAME	Vztažné body obrobku
$P_TRAFRAME	Transformace
$P_PFRAME	Aktuální programovatelný frame
Aktuální systémový frame	pro:
$P_CYCFRAME	Cykly
P_ACTFRAME	Aktuální celkový frame
Zřetězení framů	Aktuální frame se skládá z celkového základního framu

$P_NCBFRAME[n] - Aktuální globální základní framy NCU

Pomocí systémové proměnné `$P_NCBFRAME[n]` je možné prvky pole aktuálního globálního základního framu číst a zapisovat do nich. Výsledný celkový základní frame se prostřednictvím operace zápisu v kanálu započítá.

Změněný frame se aktivuje pouze v kanálu, v němž byl frame naprogramován. Jestliže má být frame změněn pro všechny kanály dané NCU, musí být data zapsána současně do obou proměnných $P_NCBFR[n] a $P_NCBFRAME[n]. Ostatní kanály musí ještě frame aktivovat, např. příkazem G54. Při zápisu základního framu se celkový základní frame nově vypočítá.

$P_CHBFRAME[n] - Aktuální kanálové základní framy

Pomocí systémové proměnné `$P_CHBFRAME[n]` je možné prvky pole aktuálního kanálového základního framu číst a zapisovat do nich. Výsledný celkový základní frame se prostřednictvím operace zápisu v kanálu započítá. Při zápisu základního framu se celkový základní frame nově vypočítá.

$P_BFRAME - Aktuální 1. základní frame v kanálu

Prostřednictvím předem definované proměnné `$P_BFRAME` typu frame je možné ve výrobním programu číst aktuální základní frame s indexem 0, který je v kanálu platný, a lze do něj i zapisovat. Zapsaný základní frame se okamžitě započítá.

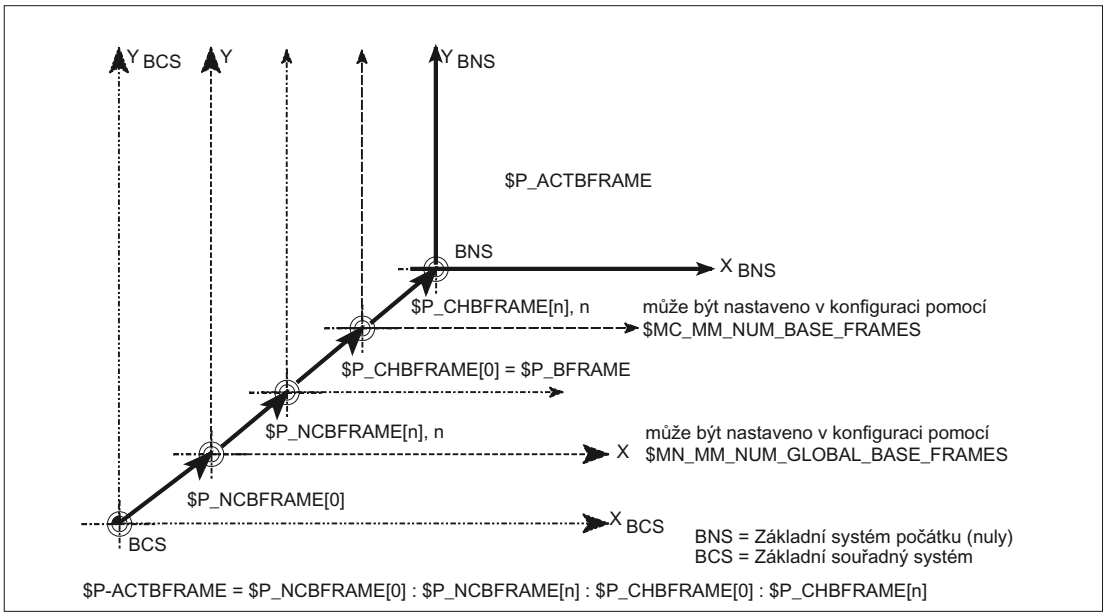
Proměnná `$P_BFRAME` je identická s proměnnou `$P_CHBFRAME[0]`. Tato systémová proměnná má standardně vždy platnou hodnotu. Pokud žádný kanálový základní frame neexistuje, při zápisu nebo při čtení se aktivuje alarm: "Frame: Nepřípustný příkaz".

$P_ACTBFRAME - celkový základní frame

Proměnná `$P_ACTFRAME` zjišťuje zřetězený celkový základní frame. Tato proměnná je jen pro čtení.

Proměnná `$P_ACTFRAME` odpovídá:

`$P_NCBFRAME[0] : ... : $P_NCBFRAME[n] : $P_CHBFRAME[0] : ... : $P_CHBFRAME[n].`



$P_CHBFRMASK a $P_NCBFRMASK - celkový základní frame

Pomocí systémových proměnných `$P_CHBFRMASK` a `$P_NCBFRMASK` může uživatel zvolit, které základní framy by chtěl zahrnout do výpočtu "celkového" základního framu. Proměnné mohou být naprogramovány jen v programu a mohou být čteny pomocí BTSS. Hodnota proměnné je interpretována jako bitová maska a udává, který prvek pole základního framu z proměnné `$P_ACTFRAME` je vtažen do výpočtu.

Pomocí parametrů `$P_CHBFRMASK` (kanálové základní framy) a `$P_NCBFRMASK` (základní framy globální v NCU) může být zadáno, které framy se započítávají.

Naprogramováním těchto proměnných se celkový základní frame a celkový frame znovu vypočítají. Po resetu a v základním nastavení je hodnota

`$P_CHBFRMASK = $MC_CHBFRAME_RESET_MASK` a

`$P_NCBFRMASK = $MC_CHBFRAME_RESET_MASK`.

např.:

```
$P_NCBFRMASK = 'H81' ;$P_NCBFRAME[0] : $P_NCBFRAME[7]
$P_CHBFRMASK = 'H11' ;$P_CHBFRAME[0] : $P_CHBFRAME[4]
```

$P_IFRAME - Aktuální nastavitelný frame

Prostřednictvím předem definované proměnné `$P_IFRAME` typu frame je možné ve výrobním programu číst aktuální nastavitelný frame, který je v kanálu platný, a lze do něj i zapisovat. Zapsaný nastavitelný frame se okamžitě započítá.

V případě v NCU globálních nastavitelných framů se změněný frame se uplatňuje pouze v kanálu, v němž byl frame naprogramován. Jestliže má být frame změněn pro všechny kanály dané NCU, musí být data zapsána současně do obou proměnných `$P_UIFR[n]` a `$P_IFRAME`. Ostatní kanály musí ještě odpovídající frame aktivovat, např. příkazem G54.

$P_PFRAME - Aktuální programovatelný frame

Proměnná `$P_PFRAME` je programovatelný frame, který vznikne naprogramováním příkazů `TRANS/ATRANS`, `G58/G59`, `ROT/AROT`, `SCALE/ASCALE`, `MIRROR/AMIRROR`, příp. přiřazením hodnot pomocí příkazů `CTRANS`, `CROT`, `CMIRROR`, `CSCALE` programovatelnému framu.

Aktuální programovatelná proměnná typu FRAME, která vytváří vztah mezi nastavitelným

- souřadným systémem (ENS) a
- souřadným systémem obrobku (WCS).

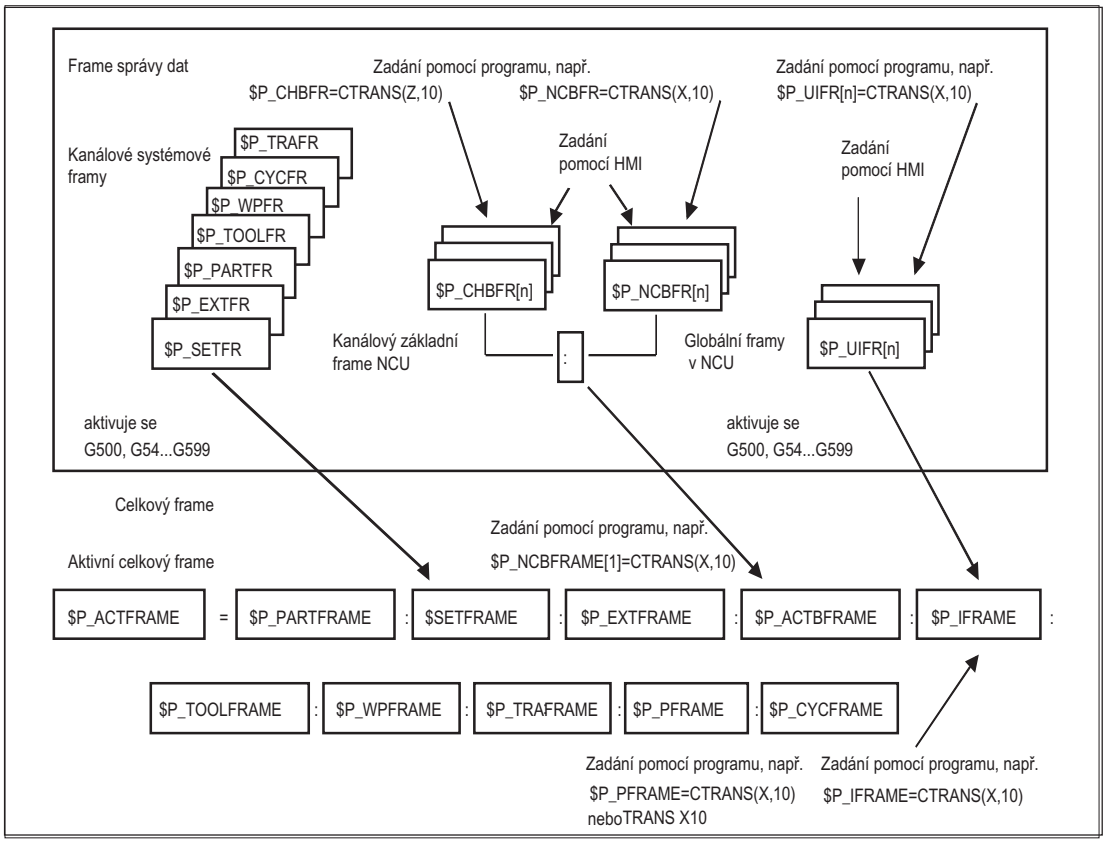
P_ACTFRAME - Aktuální celkový frame

Aktuální výsledný celkový frame $P_ACTFRAME nyní vzniká jako zřetězení všech základních framů. aktuálního nastavitelného framu a programovatelného framu. Aktuální frame se aktualizuje, kdykoli se změní některá ze složek tohoto framu.

Proměnná `$P_ACTFRAME` odpovídá:

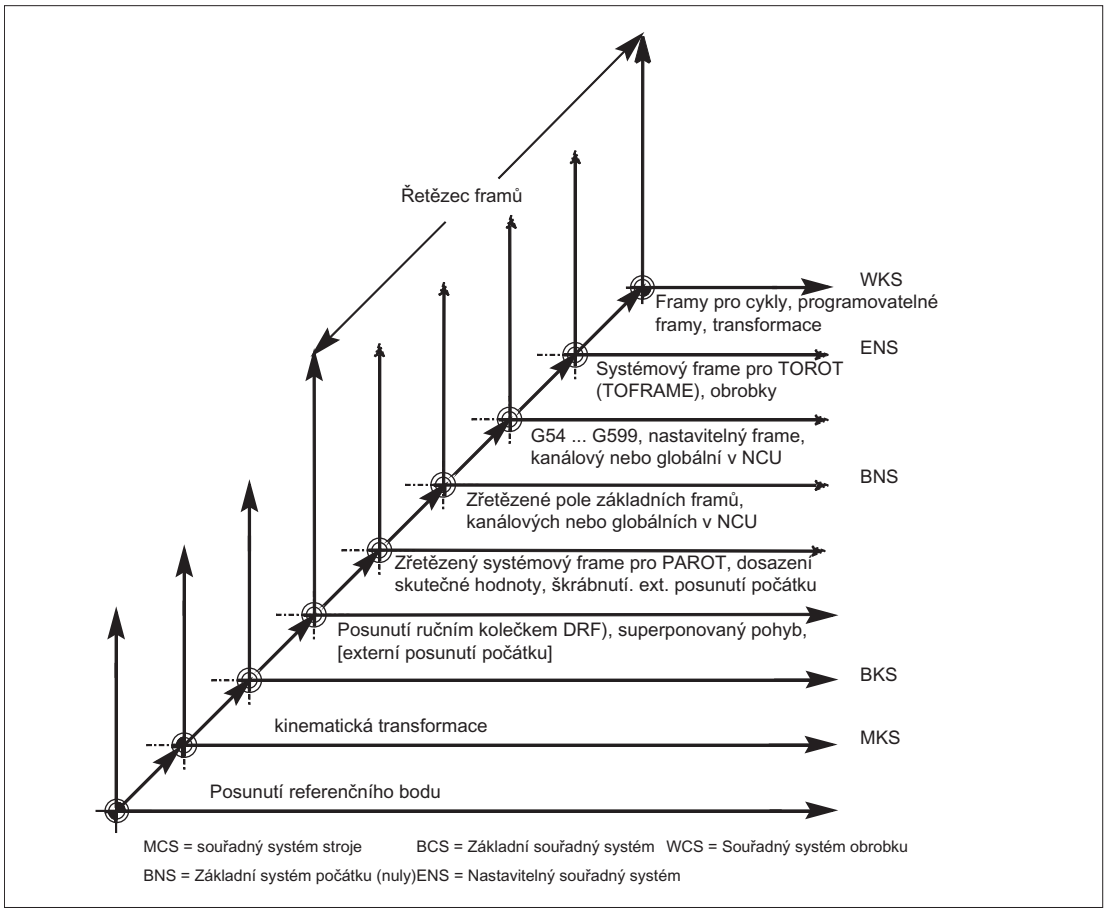
`$P_PARTFRAME : $P_SETFRAME : $P_EXTFRAME : $P_ACTBFRAME : $P_IFRAME :`

`$P_TOOLFRAME : $P_WPFRAME : $P_TRAFRAME : $P_PFRAME : $P_CYCFRAME`



Zřetězení framů

Aktuální frame se skládá z celkového základního framu, nastavitelného framu, systémového framu a z programovatelného framu podle výše uvedeného aktuálního celkového framu.



Transformace 6

6.1 Všeobecné programování různých druhů transformace

Všeobecná funkce

Aby bylo možné řídící systém přizpůsobit různým kinematikám stroje, je možné naprogramovat celou řadu různých transformací s vhodnými parametry. Pomocí těchto parametrů je možno pro vybranou transformaci odpovídajícím způsobem nastavit jak orientaci nástroje v prostoru, tak také orientační pohyby kruhových os.

V případě transformací tří, čtyř a pěti os se naprogramované údaje polohy vždy vztahují na špičku nástroje, který je nastavován tak, aby byl vždy ortogonálně vůči obráběné ploše nacházející se v prostoru. Kartézské souřadnice jsou ze základního souřadného systému přepočítány do souřadného systému stroje a jsou vztaženy na geometrické osy. Tyto souřadnice popisují pracovní bod. Virtuální kruhové osy popisují orientaci nástroje v prostoru a programují se pomocí funkce TRAORI.

V případě kinematické transformace mohou být programovány polohy v kartézském souřadném systému. Řídící systém transformuje posuvové pohyby naprogramované pomocí příkazů TRANSMIT, TRACYL a TRAANG v kartézském souřadném systému na posuvové pohyby reálných os stroje.

Programování

Transformace ve třech, čtyřech a pěti osách (TRAORI)

Deklarovaná transformace orientace se aktivuje příkazem TRAORI se třemi možnými parametry pro číslo transformace, vektor orientace a offset kruhové osy.

```
TRAORI(číslo transformace, vektor orientace, offset kruhové osy)
```

Kinematické transformace

Ke kinematickým transformacím patří deklarované transformace `TRANSMIT(číslo transformace)`

```
TRACYL(pracovní průměr, číslo transformace)
```

```
TRAANG(úhel šikmo postavené osy, číslo transformace)
```

Deaktivování aktivní transformace

Pomocí příkazu `TRAFOOF` mohou být momentálně aktivní transformace deaktivovány.

Transformace orientace

Transformace ve třech, čtyřech a pěti osách (TRAORI)

Za účelem optimálního opracování prostorově tvarovaných ploch v pracovním prostoru stroje potřebují obráběcí stroje kromě tří lineárních os X, Y a Z ještě další osy. Tyto doplňkové osy popisují orientaci v prostoru a jsou proto nazývány orientačními osami. U mnoha typů strojů s různou kinematikou jsou k dispozici jako kruhové osy.

1. Naklápěcí hlavička se dvěma osami, např. kardanová nástrojová hlavička s kruhovou osou rovnoběžně s lineární osou s pevným nástrojovým stolem.
2. Otočný stůl se dvěma osami, např. pevná naklápěcí hlavička s otočným nástrojovým stolem okolo dvou os.
3. Naklápěcí hlavička s jednou osou a otočný stůl s jednou osou, např. otočná naklápěcí hlavička s otočeným nástrojem a s otočným nástrojovým stolem okolo jedné osy.
4. Naklápěcí hlavička se dvěma osami a otočný stůl s jednou osou, např. s otočným nástrojovým stolem okolo jedné osy a s otočnou naklápěcí hlavičkou s nástrojem, který se může otáčet okolo sebe sama.

3- a 4-osá transformace jsou zvláštní případy 5-osé transformace a programují se analogicky k této 5-osé transformaci.

"Generická 3-/4-/5-/6-osá transformace pokrývá svou funkcí pravoúhle uspořádané kruhové osy, stejně jako transformace pro kardanovou frézovací hlavičku a může být aktivována příkazem TRAORI stejně jako jakákoli jiná transformace orientace i pro tyto typy strojů se čtyřmi osami. V případě generické 5-/6-osé transformace má orientace nástroje ještě další třetí stupeň volnosti, který umožňuje, že se nástroj nacházející se v libovolném směru v prostoru může ještě otáčet okolo své vlastní osy.

Literatura: /FB3/ Příručka Popis funkcí, Speciální funkce; 3- až 5-osá transformace (F2)

Základní nastavení orientace nástroje nezávislé na kinematice

ORIRESET

Pokud je aktivní transformace orientace TRAORI, potom je možné pomocí příkazu ORIRESET zadat základní polohy až 3 orientačních os s volitelnými parametry A, B, C. Přiřazení posloupnosti naprogramovaných parametrů kruhovým osám se uskutečňuje podle posloupnosti orientačních os definované prostřednictvím transformace. Naprogramování příkazu ORIRESET(A, B, C) způsobí, že orientační osy lineárně a synchronně najedou ze své momentální polohy do zadané základní polohy.

Kinematické transformace

TRANSMIT a TRACYL

V případě frézovacích prací na soustruzích mohou být naprogramovány pro deklarované transformace buď

1. opracování na čelní ploše při upnutí pro soustružení (TRANSMIT) nebo
2. obrábění drážek libovolného průběhu na válcových tělesech pomocí

transformace TARCYL.

TRAANG

Pokud má být přísuvná osa např. pro technologii broušení přisouvána také šikmo, může být pomocí transformace TRAANG pro deklarovanou transformaci naprogramován příslušný úhlový parametr.

Posuv PTP v kartézských souřadnicích

Ke kinematickým transformacím patří také "Posuv PTP" v kartézských souřadnicích", při kterém může být naprogramováno až 8 různých kloubových nastavení STAT=. Polohy jsou naprogramovány v kartézském souřadném systému, přičemž pohyb se uskutečňuje v souřadném systému stroje.

Literatura:
/FB2/, Příručka Popis funkcí, Rozšiřovací funkce; Kinematická transformace (M1)

Zřetězené transformace

Mohou být za sebou poskládány vždy maximálně dvě transformace. U takto zřetězené druhé transformace se pohybové složky os přebírají z první transformace.

Jako první transformaci je možno zadat následující:

- Transformace orientace TRAORI
- Polární transformace TRANSMIT
- Transformace válcového pláště TRACYL
- Transformace Šikmá osa TRAANG

Druhou transformací musí být Šikmá osa TRAANG.

6.1.1 Pohyby provádějící změnu orientace při transformacích

Posuvové pohyby a orientační pohyby

Posuvové pohyby programovatelných orientací závisí především na typu stroje. V případě tří-, čtyř- a pětiosé transformace s funkcí TRAORI popisují rotační osy nebo naklápěcí lineární osy orientační pohyby nástroje.

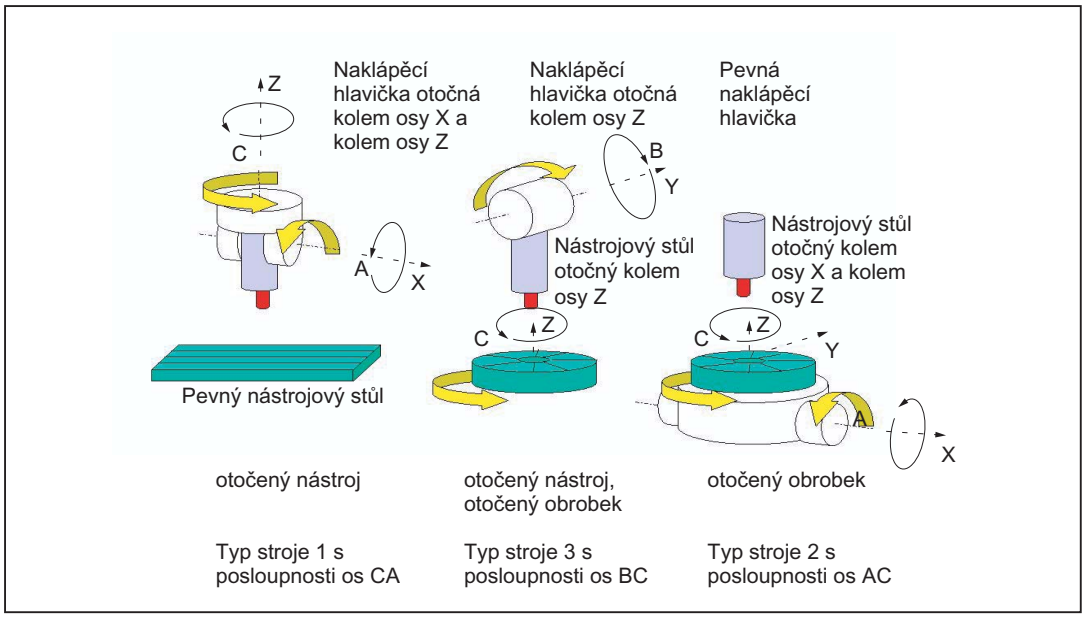
Změny poloh kruhových os podílejících se na transformaci orientace mají za následek kompenzační pohyby zbývajících strojních os. Poloha špičky nástroje zůstává přitom nezměněna.

Orientační pohyby nástroje mohou být v závislosti na použití naprogramovány prostřednictvím identifikátorů virtuálních kruhových os A..., B..., C... buď prostřednictvím zadání Eulerova úhlu nebo úhlu RPY nebo směrových vektorů nebo normálových vektorů plochy, normovaných vektorů pro osu kužele nebo pro pomocnou orientaci na kuželovou plášťovou plochu.

Při kinematické transformaci pomocí příkazů TRANSMIT, TRACYL a TRAANG řídící systém transformuje posuvové pohyby naprogramované v kartézském souřadném systému na posuvové pohyby reálných os stroje.

Kinematika stroje při transformace ve třech, čtyřech a pěti osách (TRAORI)

Otočný může být buď nástroj nebo nástrojový stůl s až maximálně dvěma kruhovými osami. Možné jsou také kombinace naklápěcí hlavičky s jednou osou a otočného stolu.



Obrázek. 6-1

Typ stroje	Programování orientace
Tříosá transformace u strojů typu 1 a 2	Programování orientace nástroje jen v rovině, která je kolmá na osu otáčení Existují **dvě** translační osy (lineární osy) a **jedna** rotační osa (kruhová osa).
Čtyřosá transformace u strojů typu 1 a 2	Programování orientace nástroje jen v rovině, která je kolmá na osu otáčení Existují **tři** translační osy (lineární osy) a **jedna** rotační osa (kruhová osa).
Pětiosá transformace, typ stroje 3 Naklápěcí hlavička s jednou osou a otočný stůl s jednou osou	Programování transformace orientace. Kinematika se **třemi** lineárními osami a **dvěma** ortogonálními kruhovými osami. Kruhové osy jsou rovnoběžné se dvěma ze tří lineárních os. První kruhová osa se pohybuje dvěma kartézskými lineárními osami. Otáčí třetí lineární osu s nástrojem. Druhá kruhová osa otáčí obrobkem.

Generické 5/6-osé transformace

Typ stroje	Programování transformace orientace
Generická Pěti-/šestiosá transformace, typ stroje 4 Naklápěcí hlavička se dvěma osami a s nástrojem, který se může otáček okolo sebe sama, a otočný stůl s jednou osou	Programování transformace orientace. Kinematika se **třemi** lineárními osami a **třemi** ortogonálními kruhovými osami. Kruhové osy jsou rovnoběžné se dvěma ze tří lineárních os. První kruhová osa se pohybuje dvěma kartézskými lineárními osami. Otáčí třetí lineární osu s nástrojem. Druhá kruhová osa otáčí obrobkem. Základní orientace nástroje může být naprogramována pomocí dodatečného otočení nástroje okolo jeho osy o úhel THETA.

Při vyvolávání "generické tří-, čtyř- a pěti-/šestiosé transformace" může být navíc ještě předávána také základní orientace nástroje. Omezení týkající se směrů u kruhových os už neplatí. Jestliže kruhové osy nejsou uspořádány přesně kolmo na sebe nebo pokud se stávající kruhové osy nenacházejí přesně rovnoběžně s lineárními osami, je možné pomocí "generické pěti-/šestiosé transformace dosáhnout lepších výsledků orientace nástroje.

Kinematické transformace TRANSMIT, TRACYL a TRAANG

Pro frézovací práce na soustruzích nebo pro šikmo nastavitelnou osu při broušení platí v závislosti na transformaci ve standardním případě pro osy následující požadavky:

TRANSMIT	Aktivování polární transformace
Obrábění na čelní ploše při upnutí pro soustružení	kruhová osa přísuvná osa kolmo na kruhovou osu podélná osa rovnoběžně s kruhovou osou

TRACYL	Aktivování transformace válcového pláště
Obrábění drážek libovolného průběhu na válcovém tělese	kruhová osa přísuvná osa kolmo na kruhovou osu podélná osa rovnoběžně s kruhovou osou

TRAANG	Aktivování transformace šikmé osy
Obrábění s šikmou přísuvnou osou	kruhová osa přísuvná osa pod úhlem, který může být nastaven pomocí parametru podélná osa rovnoběžně s kruhovou osou

Posuv PTP v kartézských souřadnicích

Pohyb stroje se uskutečňuje pomocí souřadnic stroje a pro jeho programování se používá:

TRAORI	Aktivování transformace
Najíždění PTP od bodu k bodu	Poloha v kartézském souřadném systému (MCS)
CP	Pohyb po dráze kartézskými osami (BCS)
STAT	Nastavení polohy kloubu závisí na transformaci
TU	O jaký úhel mají osy najíždět po nejkratší dráze

Najíždění PTP při generické 5/6-osé transformaci

Pohyb stroje se uskutečňuje v souřadnicích stroje a orientaci nástroje je možné naprogramovat jak pomocí poloh kruhových os, tak také pomocí Eulerova vektoru, příp. úhlu RPY nezávislých na kinematické stroje nebo pomocí směrových vektorů.

Přitom jsou možné interpolace kruhových os, vektorové interpolace s využitím interpolace největší kružnice koule nebo interpolace vektoru orientace na kuželové válcové ploše.

Příklad tří- až pětiosé transformace u kardanové frézovací hlavičky

Obráběcí stroj má minimálně 5 os, z nichž jsou:

- tři osy lineární s přímkovými pohyby, které mohou najet na pracovní bod v jakékoli libovolné poloze v pracovním prostoru
- dvě otočné naklápěcí osy, které jsou uspořádány pod úhlem nastaveným v konfiguraci (většinou 45 stupňů) a které umožňují nastavování orientace nástroje v prostoru, které je v případě uspořádání pod úhlem 45 stupňů omezeno na polokouli

6.1.2 Přehled transformace orientace TRAORI

Možné druhy programování v souvislosti s transformací TRAORI

Typ stroje	Programování při aktivní transformaci TRAORI
Typy strojů 1, 2 nebo 3. Naklápěcí hlavička se dvěma osami nebo otočný stůl se dvěma osami nebo kombinace naklápěcí hlavičky a otočného stolu, které mají po jedné ose.	Posloupnost orientačních os a směrů orientace nástroje je buď **vztažena ke stroji** nastavuje se v konfiguraci pomocí strojních parametrů v závislosti na kinematice stroje nebo **vztaženo na souř. systém obrobku** s programovatelnou orientací nezávisle na kinematice stroje Směr otáčení orientačních os se ve vztažném systému naprogramuje pomocí příkazů - ORIMKS vztažný systém = souřadný systém stroje - ORIWKS vztažný systém = souřadný systém obrobku Základní nastavení je ORIWKS. Programování orientačních os zadáním těchto dat: A, B, C - polohy os stroje přímo A2, B2, C2 - programování úhlu virtuálních os pomocí - ORIEULER - pomocí Eulerova úhlu (standardní) - ORIRPY - pomocí úhlu RPY - ORIVIRT1 - pomocí virtuálních orientačních os (definice 1) - ORIVIRT2 - pomocí virtuálních orientačních os (definice 2) s rozlišením druhu interpolace: **lineární interpolace** - ORIAXES pro orientační osy nebo osy stroje **interpolace pomocí největší kružnice koule** (interpolace vektoru orientace) - ORIVECT pro orientační osy Programování orientačních os zadáním těchto dat: A3, B3, C3 - vektorové složky (směr/normála k ploše) Programování výsledné orientace nástroje A4, B4, C4 - normálové vektory plochy na začátku bloku A5, B5, C5 - normálové vektory plochy na konci bloku LEAD - předozadní úhel pro orientaci nástroje TILT - úhel bočního naklopení pro orientaci nástroje
	Interpolace vektoru orientace na kuželové plášťové ploše Změny orientace na kuželové plášťové ploše nacházející se **libovolně v prostoru** prostřednictvím interpolace: - ORIPLANE v rovině (interpolace pomocí největší kružnice koule) - ORICONCW - po ploše pláště kužele ve směru hodinových ručiček - ORICONCCW - po ploše pláště kužele proti směru hodinových ručiček A6, B6, C6 - směrové vektory (osa otáčení kužele) -OICONIO - Interpolace na kuželové plášťové ploše pomocí: A7, B7, C7 - pomocné vektory (počáteční a koncová orientace) nebo - ORICONTO - na ploše pláště kuželu s tangenciálním přechodem Změny orientace vztahující se **na dráhu** s příkazy - ORICURVE - zadání pohybu přes dva uzlové body pomocí PO[XH]=(xe, x2, x3, x4, x5) - polynom orientace až do 5. stupně PO[YH]=(ye, y2, y3, y4, y5) - polynom orientace až do 5. stupně PO[ZH]=(ze, z2, z3, z4, z5) - polynom orientace až do 5. stupně - ORIPATHS - vyhlazování charakteristiky orientace A8, B8, C8 - Fáze změny orientace nástroje odpovídá: Směru a délce dráhy nástroje při pozvedávacím pohybu

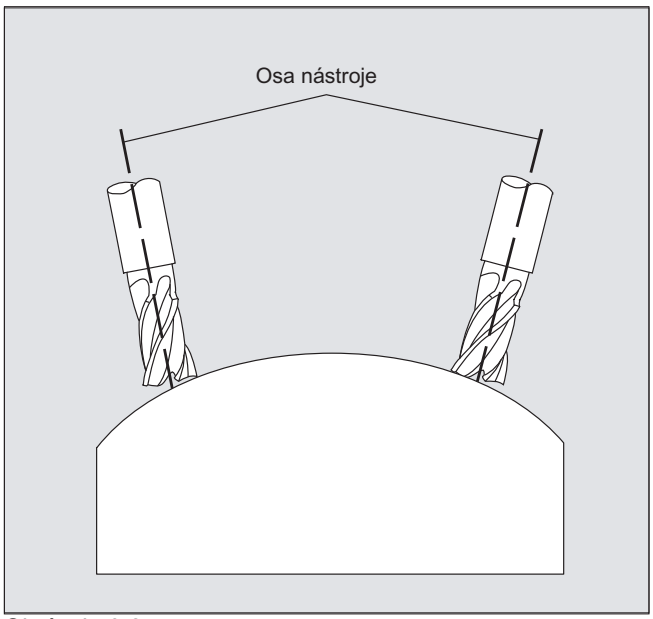
Typ stroje	Programování při aktivní transformaci TRAORI
Typy stroje 1 a 3 Další typy strojů, které umožňují otáčení nástroje okolo sebe sama, vyžadují 3. osu Transformace orientace, jako např. generická 6-osá transformace. Otáčení vektoru orientace.	Programování **otočení** orientace nástroje pomocí následujících parametrů: LEAD - předozadní úhel vzhledem k normálovému vektoru plochy PO[PHI] - programování polynomu až 5. stupně TILT - boční úhel otočení k tečně ke dráze (směr Z) PO[PSI] - programování polynomu až 5. stupně THETA - úhel otočení (otáčení okolo směru nástroje ve směru Z) THETA= hodnota, která bude dosažena na konci bloku THETA=AC(...) - blokové přepnutí na zadávání absolutních rozměrů THETA=AC(...) - blokové přepnutí na zadávání inkrementálních rozměrů THETA=Θ_e - interpolace naprogramovaného úhlu G90/G91 PO[THT]=(..) - programování polynomu až 5. stupně Programování vektoru otočení - ORIROTA - absolutní otočení - ORIROTR - relativní vektor otočení - ORIROTT - tangenciální vektor otočení
Orientace vztahující se k dráze pro popis změn orientace vzhledem ke dráze nebo otáčení vektoru tangenciálně k dráze	Změny orientace **vztažené k dráze** pomocí příkazů: - ORIPATH - Orientace nástroje vztažená na dráhu - ORIPATHS - navíc v případě skokové změny na charakteristice orientace Programování vektoru otočení - ORIROTC - tangenciální vektor otočení, otáčení k tečně dráhy

6.2 Transformace ve třech, čtyřech a pěti osách (TRAORI)

6.2.1 Všeobecné souvislosti v případě kardanové nástrojové hlavičky

Funkce

Aby bylo dosaženo optimálních řezných podmínek při obrábění prostorově zakřivených ploch, musí být možné měnit úhel nastavení nástroje.



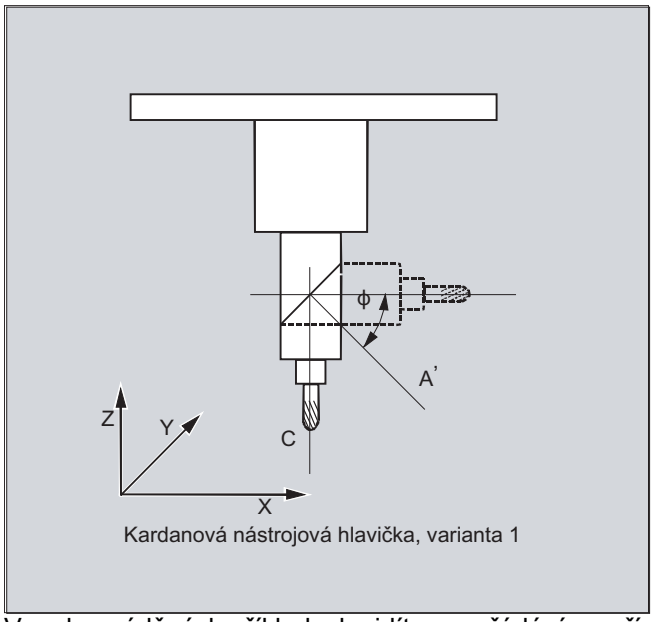
Obrázek. 6-2

To, se kterými konstrukcemi stroje se toho dosáhne, je uloženo v parametrech os.

5-osá transformace

Kardanová nástrojová hlavička

V tomto případě definují tři lineární osy (X, Y, Z) a dvě orientační osy (C, A) úhel nastavení a pracovní bod nástroje. Jedna z obou orientačních os je nastavena jako šikmá osa, zde v tomto příkladu A' - v mnoha případech jako uspořádání pod úhlem 45°.

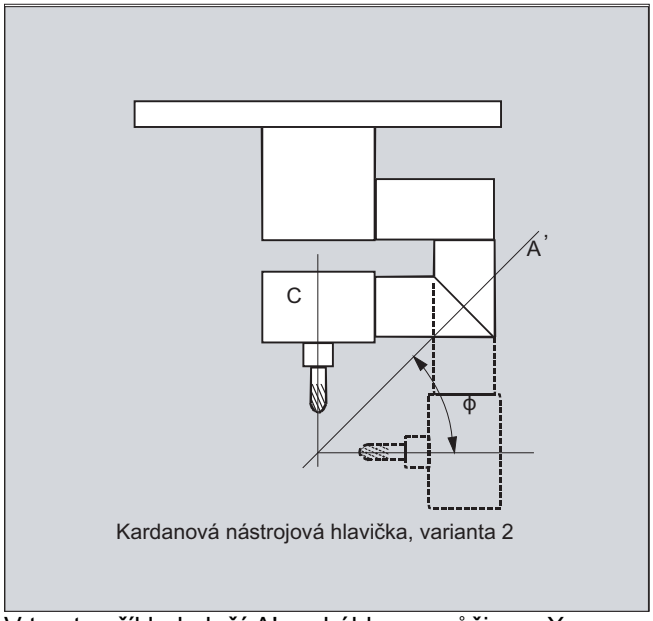


Kardanová nástrojová hlavička, varianta 1

Ve zde uváděných příkladech vidíte uspořádání na případu kinematiky stroje CA s kardanovou nástrojovou hlavičkou!

Výrobce stroje

Posloupnost orientačních os a směr orientace nástroje mohou být pomocí strojních parametrů nastaveny nezávisle na kinematice stroje.



Kardanová nástrojová hlavička, varianta 2

V tomto příkladu leží A' pod úhlem φ vůči ose X

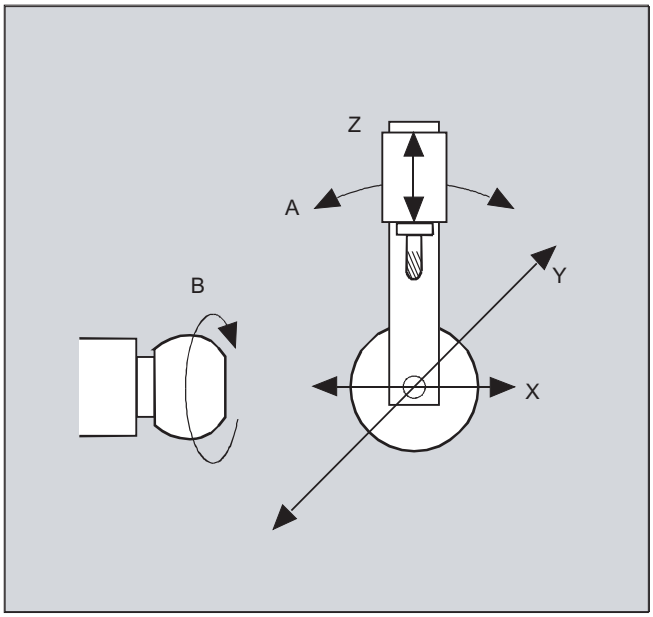
Obecně platí následující možné souvislosti:

A' leží pod úhlem φ vůči ose	Osa X
B' leží pod úhlem φ vůči ose	Osa Y
C' leží pod úhlem φ vůči ose	Osa Z

Úhel φ může být v rozsahu 0° až +89° nastaven v konfiguraci pomocí strojních parametrů.

S naklápěcí lineární osou

V tomto případě se jedná o uspořádání s pohyblivým obrobkem a pohyblivým nástrojem. Kinematika se skládá ze tří lineárních os (X, Y, Z) a tří kruhových os uspořádaných vůči sobě navzájem v pravých úhlech. První kruhová osa se pohybuje např. nad křížovým suportem dvou lineárních os, nástroj se nachází rovnoběžně se třetí lineární osou. Druhá kruhová osa otáčí obrobkem. Třetí lineární osa (naklápěcí osa) leží v rovině křížového suportu.



Posloupnost rotační os a směr orientace nástroje mohou být pomocí strojních parametrů nastaveny nezávisle na kinematice stroje.

Platí následující možné souvislosti:

Osy:	**Posloupnost os:**
1. kruhová osa	A A B B C C
2. kruhová osa	B C A C A B
Naklopená lineární osa	Z Y Z X Y X

Pokud budete potřebovat další vysvětlení týkající se možností konfigurace posloupností os pro směr orientace nástroje, viz:

Literatura: /FB3/ Příručka Popis funkcí, Speciální funkce; 3- až 5-osá transformace (F2), kapitola "Kardanová frézovací hlavička, "Dosazování parametrů".

6.2.2 Transformace ve třech, čtyřech a pěti osách (TRAORI)

Funkce

Uživatel může nastavit v konfiguraci dvě, příp. tři lineární osy a jednu osu kruhovou. Transformace vychází z toho, že se kruhová osa nachází kolmo na orientační rovinu.

Orientace nástroje je možná pouze v rovině, která je kolmá na kruhovou osu. Tato transformace podporuje typy strojů s pohyblivým nástrojem a pohyblivým obrobkem.

Konfigurace a programování tří- a čtyřosé transformace se provádí analogicky k pětiosé transformaci.

Literatura:
Příručka Popis funkcí, Speciální funkce; Transformace ve více osách (F2)

Syntaxe

```
TRAORI(<n>)
TRAORI(<n>,<X>,<Y>,<Z>,<A>,<B>)
TRAFOOF
```

Význam

`TRAORI`:	Aktivuje první deklarovanou transformaci orientace
`TRAORI(<n>)`:	Aktivuje n-tou deklarovanou transformaci orientace
`<n>`:	Číslo transformace Hodnota 1 nebo 2 : Příklad: TRAORI(1) aktivuje transformaci orientace 1
`<X>,<Y>,<Z>`:	Složky vektorů orientace, do kterých ukazuje nástroj
`<A>,<B>`:	Programovatelný offset pro kruhové osy
`TRAFOOF`:	Deaktivování transformace

Orientace nástroje

V závislosti na zvolených směrech orientace nástroje musí v NC programu nastavena aktivní pracovní rovina (G17, G18, G19) takovým způsobem, aby korekce délky nástroje působila ve směru orientace nástroje.

Poznámka

Po aktivování transformace jsou údaje o poloze (X, Y, Z) vztaženy vždy na špičku nástroje. Změna polohy kruhových os podílejících se na transformaci vede ke kompenzačním pohybům zbývajících os stroje, čímž se zajistí, že poloha špičky nástroje zůstane nezměněna.

Transformace orientace je vždy definována od špičky nástroje k jeho upevnění.

Offset pro orientační osy

Při aktivování transformace orientace je možné přímo naprogramovat doplňkový offset pro orientační osy.

Pokud je při programování dodržena správná posloupnost, smí být některé parametry vypuštěny.

Příklad:

`TRAORI(, , , ,A,B)` ; jestliže se mají zadat jen některé offsety.

Další možností vedle přímého naprogramování, jak mohou být doplňkové offsety pro orientační osy také zadány, je automatické převzetí z momentálně aktivního posunutí počátku. Tato funkce pro přebírání se nastavuje v konfiguraci pomocí strojních parametrů.

Příklady

```
TRAORI(1,0,0,1)          ; V základní orientaci je nástroj nasměrován v ose Z

TRAORI(1,0,1,0)          ; V základní orientaci je nástroj nasměrován v ose Z

TRAORI(1,0,1,1)          ; V základní orientaci nástroj ukazuje ve směru Y/Z
                           (odpovídá poloze -45°)
```

6.2.3 Varianty programování orientace a základního nastavení (ORIRESET)

Programování orientace nástroje pomocí funkce TRAORI

Ve spojení s programovatelnou transformací orientace TRAORI mohou být kromě lineárních os X, Y, Z pomocí identifikátorů kruhových os programovány také polohy os A..., B..., C... nebo virtuální osy s úhly nebo komponenty vektorů. Pro orientační osy a osy stroje jsou možné různé druhy interpolace. Nezávisle na tom, které polynomy orientace PO[úhel] a polynomy os PO[osa] jsou právě aktivní, může být naprogramován větší počet různých druhů polynomů, jako např. G1, G2, G3, CIP nebo POLY.

Změna orientace nástroje může být naprogramována také pomocí vektorů orientace. Orientace na konci každého bloku přitom může být zadávána buď přímým naprogramováním vektorů nebo naprogramováním poloh kruhových os.

Poznámka

Varianty programování orientace v případě tří- až pětiosých transformací

V případě tří- až pětiosých transformací se následující varianty

1. A, B, C - přímé zadávání poloh os stroje
2. A2, B2, C2 - programování úhlu virtuálních os pomocí Eulerova úhlu nebo úhlu RPY
3. A3, B3, C3 - zadávání vektorových složek
4. LEAD, TILT - zadávání předozadního a bočního úhlu vztažené na dráhu a plochu
5. A4, B4, C4 a A5, B5, C5 - normálový vektor plochy na začátku bloku a na konci bloku
6. A6, B6, C6 a A7, B7, C7 - interpolace vektoru orientace na kuželové plášťové ploše
7. A8, B8, C8 - změna orientace nástroje, směr a délka dráhy pohybu při pozvednutí

vzájemně vylučují.

Jestliže je naprogramováno smíšené zadání, je odmítnuto a aktivuje se alarmové hlášení.

Základní nastavení orientace nástroje - ORIRESET

Naprogramováním příkazu ORIRESET(A, B, C) orientační osy lineárně a synchronně najedou ze své momentální polohy do zadané základní polohy.

Jestliže pro danou osu není naprogramována žádná základní poloha, potom se použije poloha definovaná pomocí odpovídajícího strojního parametru $MC_TRAFO5_ROT_AX_OFFSET_1/2. Eventuálně aktivní framy kruhových os přitom nejsou nijak zohledňovány.

Poznámka

Jen tehdy, pokud je transformace orientace pomocí příkazu TRAORI(...) aktivní, může být naprogramována základní poloha orientace nástroje pomocí příkazu ORIRESET(...) nezávisle na kinematice stroje, aniž by se aktivoval alarm 14101.

Příklady

```
1. příklad kinematiky stroje CA (názvy kanálových os C, A)
ORIRESET(90, 45)       ;C v poloze 90 stupňů, A v poloze 45 stupňů
ORIRESET(, 30)         ;C v poloze podle $MC_TRAFO5_ROT_AX_OFFSET_1/2[0], A v poloze
                       30 stupňů
ORIRESET( )            ;C v poloze podle $MC_TRAFO5_ROT_AX_OFFSET_1/2[0],
                       ;A v poloze podle $MC_TRAFO5_ROT_AX_OFFSET_1/2[1]
2. příklad kinematiky stroje CAC (názvy kanálových os C, A, B)
ORIRESET(90, 45, 90)   ;C v poloze 90 stupňů, A v poloze 45 stupňů, B v poloze 90
                       stupňů
ORIRESET( )            ;C v poloze podle $MC_TRAFO5_ROT_AX_OFFSET_1/2[0],
                       ;A v poloze podle $MC_TRAFO5_ROT_AX_OFFSET_1/2[1],
                       ;B v poloze podle $MC_TRAFO5_ROT_AX_OFFSET_1/2[2]
```

Programování otočení LEAD, TILT a THETA

Otočení orientace nástroje se v případě tří- až pětiosé transformace programují pomocí předozadního úhlu LEAD a pomocí bočního úhlu TILT.

V případě transformace se třetí kruhovou osou jsou povoleny jak orientace pomocí vektorových složek, tak také doplňková programování C2 pomocí zadávání úhlů LEAD, TILT (otočení vektorů orientace).

Pomocí doplňkové třetí kruhové osy může být naprogramováno otočení nástroje okolo sebe sama o úhel THETA.

6.2.4 Programování orientace nástroje (A..., B..., C..., LEAD, TILT)

Funkce

Pro programování orientace nástroje existují následující možnosti:

1. Přímé programování pohybu kruhových os. Změna orientace se vždy uskutečňuje v základním souřadném systému nebo v souřadném systému stroje. Orientační osy se pohybují jako synchronní osy.
2. Naprogramování Eulerových úhlů nebo úhlů RPY podle definice prostřednictvím `A2, B2, C2`.
3. Naprogramování směrových vektorů pomocí `A3, B3, C3`. Směrový vektoru ukazuje od špičky nástroje ve směru jeho unášeče (držáku).
4. Programování normálových vektorů plochy na začátku bloku pomocí `A4, B4, C4` na na konci bloku pomocí `A5, B5, C5` (frézování na čelní ploše).
5. Naprogramování pomocí předozadního úhlu `LEAD` a bočního úhlu `TILT`.

6. Naprogramování osy kužele jako normovaného vektoru pomocí `A6, B6, C6` nebo pomocné orientace na kuželové plášťové ploše pomocí `A7, B7, C7`
   Viz kapitola "Programování orientace podél kuželové plášťové plochy (ORIPLANE, ORICONxx)".
7. Programování změny orientace, směru a délky dráhy nástroje v průběhu pohybu při pozvednutí pomocí `A8, B8, C8`,
   viz kapitola "Vyhlazení průběhu orientace (ORIPATHS A8=, B8=, C8=)"

Poznámka

Ve všech případech je programování orientace přípustné jen tehdy, pokud je transformace orientace aktivována.

Výhoda: Tyto programy jsou přenositelné na jakoukoli kinematiku stroje.

Definice orientace nástroje pomocí G-kódu

Poznámka
Výrobce stroje

Pomocí strojního parametru je možné přepínat mezi Eulerovými úhly a úhly RPY. V případě odpovídajících nastavení strojních parametrů je přepínání možné jak nezávisle, tak i v závislosti na aktivním G-kódu ze skupiny 50. Můžete si vybrat z následujících možností nastavení:

1. Jestliže jsou oba strojní parametry pro definici orientačních os a definici orientačních úhlů nastaveny pomocí G-kódů na nulu:
   Úhel naprogramovaný pomocí `A2, B2, C2` je **v závislosti na strojním parametru** pro definici úhlu pro programování orientace interpretován buď jako Eulerův úhel nebo jako úhel RPY.
2. Pokud je strojní parametr pro definici orientačních os nastaven pomocí G-kódu na jedna, provádí se přepínání
   v závislosti na aktivním G-kódu ze skupiny 50:
   Úhly naprogramované pomocí `A2, B2, C2` jsou interpretovány podle aktivního G-kódu `ORIEULER, ORIRPY, ORIVIRT1, ORIVIRT2, ORIAXPOS` a `ORIPY2`. Hodnoty naprogramované prostřednictvím orientačních os jsou interpretovány v souladu s aktivním G-kódem ze skupiny 50 také jako orientační úhel.
3. Jestliže je strojní parametr pro definici úhlu orientace prostřednictvím G-kódu nastaven na jedničku a strojní parametr pro definici orientačních os pomocí G-kódu je nastaven na nulu, provádí se přepínání
   nezávisle na aktivním G-kódu ze skupiny 50.
   Úhly naprogramované pomocí `A2, B2, C2` jsou interpretovány podle aktivního G-kódu `ORIEULER, ORIRPY, ORIVIRT1, ORIVIRT2, ORIAXPOS` a `ORIPY2`. Hodnoty naprogramované prostřednictvím orientačních os jsou interpretovány nezávisle na aktivním G-kódu ze skupiny 50 vždy jako polohy kruhových os.

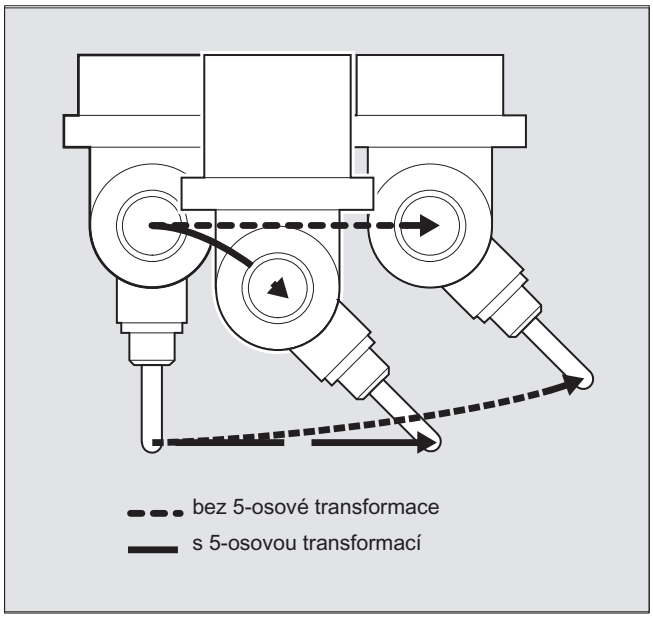
Programování

`G1 X Y Z A B C`	Programování pohybu kruhových os
`G1 X Y Z A2= B2= C2=`	Programování Eulerových úhlů
`G1 X Y Z A3== B3== C3==`	Programování směrových vektorů
`G1 X Y Z A4== B4== C4==`	Programování normálového vektoru plochy na začátku bloku.
`G1 X Y Z A5== B5== C5==`	Programování normálového vektoru plochy na konci bloku.
`LEAD=`	Předozadní úhel pro programování orientace nástroje
`TILT=`	Boční úhel pro programování orientace nástroje

Parametry

`G....`	Zadání druhu pohybu kruhových os
`X Y Z`	Zadání pro lineární osy
`A B C`	Zadání poloh kruhových os stroje
`A2 B2 C2`	Programování úhlů (Eulerův úhel nebo úhel RPY) virtuálních os, příp. orientačních os
`A3 B3 C3`	Zadání vektorových složek směrového vektoru
`A4 B4 C4`	Zadání složek normálového vektoru plochy na začátku bloku, např. při frézování na čelní ploše
`A5 B5 C5`	Zadání složek normálového vektoru plochy na konci bloku, např. při frézování na čelní ploše
`LEAD`	Úhel je vztažen k normálovému vektoru plochy, a svírají jej tečna dráhy a normálový vektor plochy v dané rovině
`TILT`	Úhel v rovině, která je kolmá na tečnu ke dráze vzhledem k normálovému vektoru plochy

Příklad ukazující rozdíl mezi situacemi s pětiosou transformací a bez ní



Popis

5-osé programy jsou zpravidla vytvářeny v systémech CAD/CAM a nejsou zadávány na řídícím systému. Z tohoto důvodu jsou následující vysvětlení určena především pro programátory postprocesorových systémů.

Druh programování orientace je definován G-kódem ze skupiny 50.

`ORIEULER` pomocí Eulerova úhlu
`ORIRPY` pomocí úhlu RPY (posloupnost otáčení ZYX)
`ORIVIRT1` pomocí virtuálních orientačních os (definice 1)
`ORIVIRT2` pomocí virtuálních orientačních os (definice 2)
`ORIAXPOS` pomocí virtuálních orientačních os s polohováním kruhové osy
`ORIPY2` pomocí úhlu RPY (posloupnost otáčení XYZ)

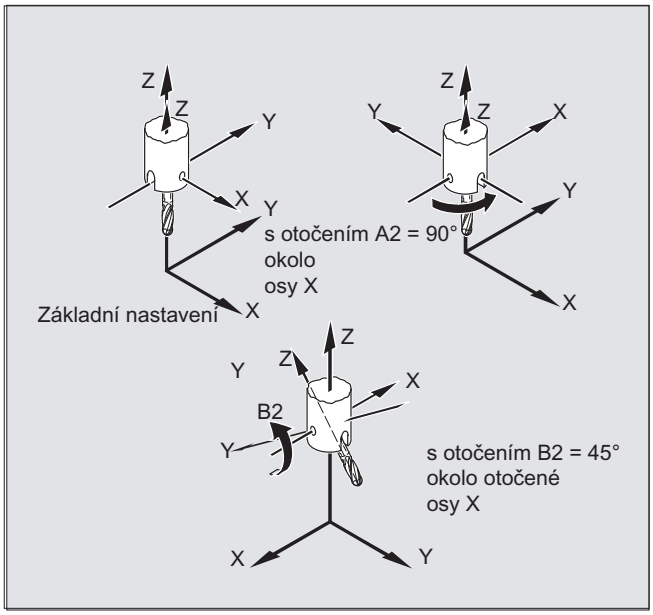
Výrobce stroje

Prostřednictvím strojních parametrů mohou být výrobcem stroje definovány různé varianty. Věnujte prosím pozornost informacím od výrobce stroje.

Programování Eulerových úhlů - ORIEULER

Hodnoty naprogramované pomocí `A2, B2, C2` při programování orientace jsou interpretovány jako Eulerův úhel (ve stupních).

Vektor orientace vzniká tak, že vektor ve směru osy Z se napřed pootočí o `A2` okolo osy Z, potom o `B2` se pootočí okolo nové osy X a nakonec se otočí o `C2` okolo nové osy Z.



V tomto případě nemá hodnota `C2` (otočení okolo nové osy Z) žádný význam a nemusí být proto naprogramována.

Programování v úhlech RPY - ORIRPY

Hodnoty naprogramované pomocí `A2, B2, C2` při programování orientace jsou interpretovány jako úhel RPY (ve stupních).

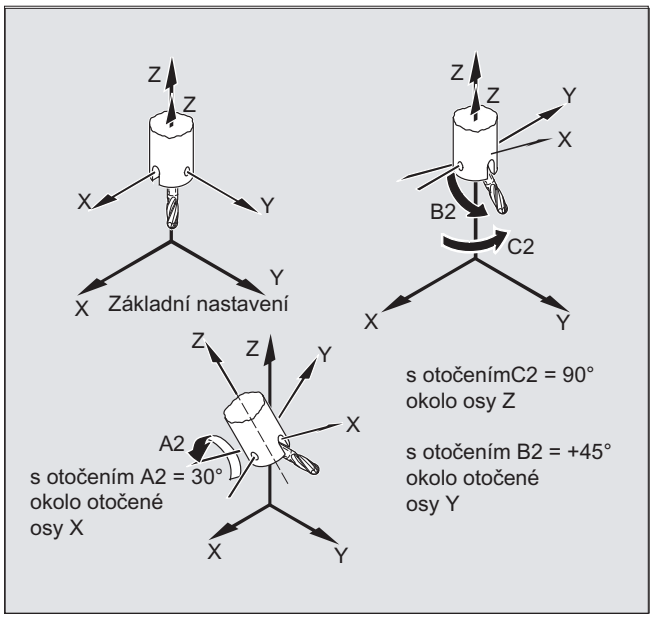
Poznámka

Oproti programování Eulerova úhlu zde mají na vektor orientace vliv všechny tři hodnoty.

Výrobce stroje

Při definici úhlu pomocí úhlu orientace prostřednictvím úhlu RPY platí pro orientační osy s nastavením $MC_ORI_DEF_WITH_G_CODE = 0

Vektor orientace vzniká tak, že vektor ve směru osy Z se napřed pootočí o C2 okolo osy Z, potom o B2 se pootočí okolo nové osy Y a nakonec se otočí o A2 okolo nové osy X.

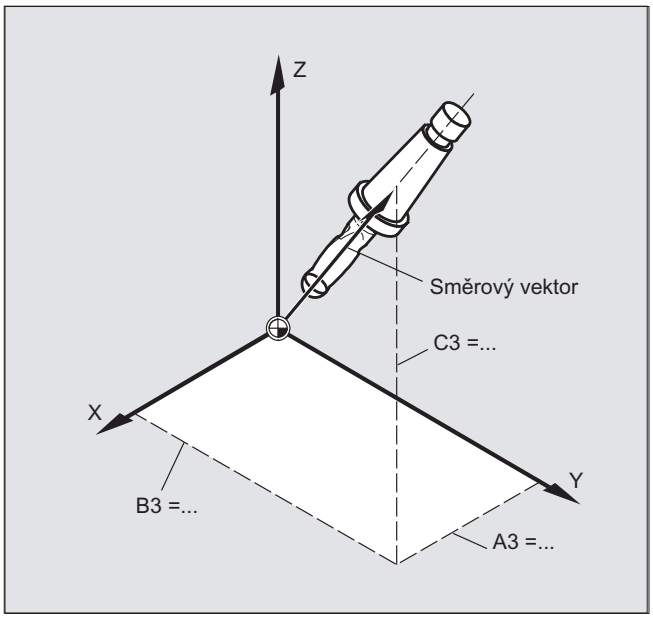


Pokud je strojní parametr pro definici orientačních os pomocí G-kódu $MC_ORI_DEF_WITH_G_CODE = 1, potom platí:
Vektor orientace vzniká tak, že vektor ve směru osy Z se napřed pootočí o A2 okolo osy Z, potom o B2 se pootočí okolo nové osy Y a nakonec se otočí o C2 okolo nové osy X.

Programování směrových vektorů

Komponenty směrového vektoru jsou naprogramovány pomocí `A3, B3, C3`. Vektor ukazuje ve směru upnutí nástroje, délka vektoru přitom nemá žádný význam.

Nenaprogramovaným složkám vektoru je dosazena nulová hodnota.



Programování orientace nástroje pomocí příkazů LEAD= a TILT=

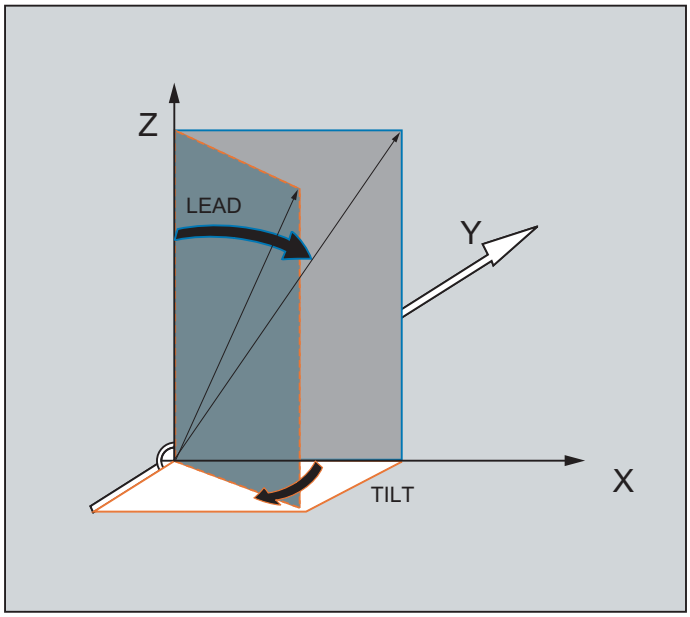
Výsledná orientace nástroje je zjišťována z těchto údajů:

- Tečna ke dráze
- Normálový vektor plochy
  na začátku bloku `A4, B4, C4` a na konci bloku `A5, B6, C5`
- Předozadní úhel `LEAD`
  v rovině definované tečnou ke dráze a normálovým vektorem plochy
- Úhel bočního naklopení `TILT` na konci bloku
  kolmo na tečnu ke dráze a vzhledem k normálovému vektoru plochy

Chování na vnitřních rozích (v případě 3D korekce nástroje)

Jestliže je blok ve vnitřním rohu zkrácený, je výsledné orientace nástroje rovněž dosaženo na konci bloku.

Definice orientace nástroje pomocí příkazů LEAD= a TILT=

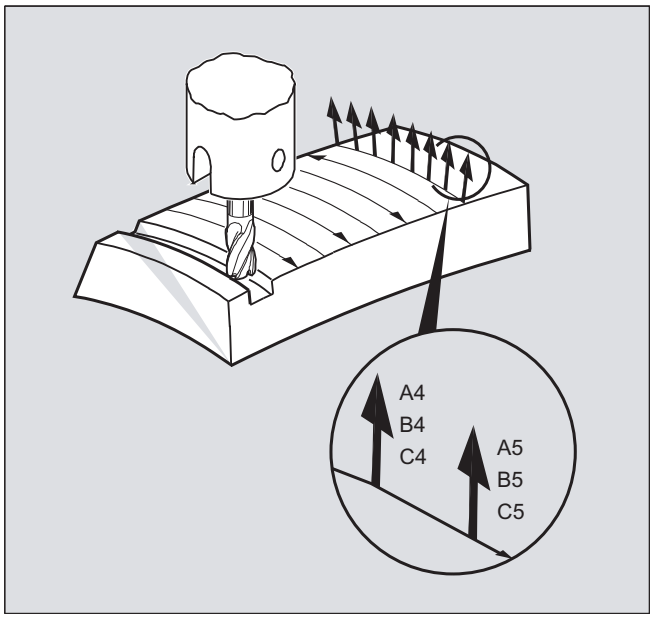


Obrázek. 6-3

6.2.5 Frézování na čelní ploše (3D frézování A4, B4, C4, A5, B5, C5)

Funkce

Frézování na čelní ploše slouží pro opracovávání libovolně zakřivených povrchů.



Pro tento druh 3D frézování potřebujete popis řádek po řádku jednotlivých 3D drah na povrchu obrobku.

Při výpočtech se obvykle uskutečňuje v systému CAM a zohledňuje se při něm tvar nástroje a rozměry nástroje. Nahotovo vypočítané NC bloky jsou potom prostřednictvím postprocesoru načteny do řídícího systému.

Programování zakřivení dráhy

Popis ploch

Pro popis zakřivení dráhy se využívá normálových vektorů plochy s následujícími složkami:

`A4, B4, C4` - počáteční vektor na začátku bloku

`A5, B5, C5` - koncový vektor na konci bloku

Jestliže se v bloku nachází pouze počáteční vektor, zůstává normálový vektor plochy po celý blok konstantní. Pokud se v bloku nachází pouze koncový vektor, uskuteční se interpolace pomocí největší kružnice koule z koncové hodnoty předcházejícího bloku do naprogramované koncové hodnoty.

Jestliže jsou naprogramovány jak počáteční, tak i koncový vektor, mezi oběma směry se rovněž uskuteční interpolace pomocí největší kružnice koule. Takto se dají vytvářet spojité hladké pohyby po dráze.

V základním nastavení ukazují normálové vektory plochy nezávisle na aktivní rovině G17 až G19 ve směru osy Z.

Délka vektoru nemá žádný význam.

Nenaprogramovaným složkám vektoru je dosazena nulová hodnota.

Když je aktivní příkaz ORIWKS, viz kapitola "Vztah orientačních os (ORIWKS, ORIMKS)", jsou normálové vektory plochy vztaženy na aktivní frame a při otáčení framu se otáčejí také.

Výrobce stroje

Normálový vektor plochy musí být v rámci mezních hodnot nastavených pomocí strojních parametrů kolmý na tečnu ke dráze, jinak se aktivuje alarm.

6.2.6 Vztah orientačních os (ORIWKS, ORIMKS)

Funkce

Při programování orientace v souřadném systému obrobku pomocí

- Eulerova úhlu nebo úhlu RPY nebo
- vektoru orientace

může být pomocí funkcí `ORIMKS`/`ORIWKS` nastaven průběh otáčivého pohybu.

Poznámka
Výrobce stroje
Druh interpolace pro orientaci je definován pomocí strojního parametru:
MD21104 $MC_ORI_IPO_WITH_G_CODE
= FALSE: Vztah určují G-funkce ORIWKS a ORIMKS
= TRUE: Vztah je určen G-funkcemi z 51. skupiny (ORIAXES, ORIVECT, ORIPLANE, ...)

Syntaxe

```
ORIMKS=...
ORIWKS=...
```

Význam

`ORIMKS`	Otáčení v souřadném systému stroje
`ORIWKS`	Otáčení v souřadném systému obrobku

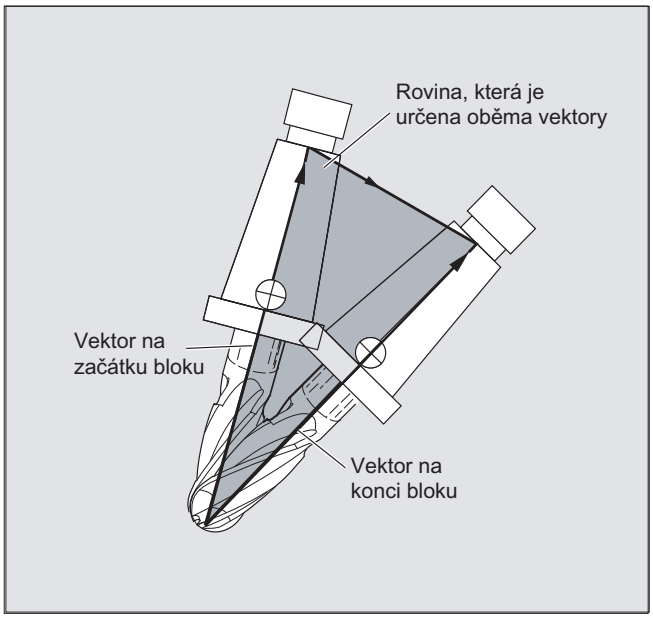
Poznámka
Základním nastavením je ORIWKS. Jestliže není u programu pracujícího s pěti osami předem jasné, na jakém stroji má být zpracováván, mělo by být v zásadě zvoleno ORIWKS. To, jaké pohyby stroje budou ve skutečnosti provedeny, závisí na kinematice stroje.

Pomocí `ORIMKS` mohou být programovány skutečné pohyby stroje, např. aby se zabránilo kolizi s různými zařízeními atd.

Popis

V případě `ORIMKS` je prováděný pohyb nástroje **závislý** na kinematice stroje. Při změně orientace, kdy je špička nástroje na pevném místě v prostoru, se mezi dvěma polohami kruhové osy uskutečňuje lineární interpolace.

V případě `ORIWKS` je prováděný pohyb nástroje **nezávislý** na kinematice stroje. V případě změny orientace, při které špička nástroje zůstává pevně na jednom místě, se nástroj pohybuje v rovině definované počátečním a koncovým vektorem.



Singulární polohy

Poznámka
ORIWKS

Orientační pohyby v oblasti singulární polohy u stroje s pěti osami vyžadují velké pohyby os stroje. (Například v případě otočné naklápěcí hlavičky s osou C jako otočnou osou a osou A jako naklápěcí osou jsou singulární všechny polohy, kdy A=0.)

Výrobce stroje

Aby nebyly osy stroje příliš přetíženy, regulační systém v blízkosti singulárních poloh výrazně snižuje rychlost pohybu po dráze.

Pomocí strojních parametrů

`$MC_TRAFO5_NON_POLE_LIMIT`

`$MC_TRAFO5_POLE_LIMIT`

mohou být pro transformaci dosazeny takové parametry, aby se orientační pohyby v blízkosti pólů byly provedeny skrz pól a aby tak bylo umožněno plynulé opracování.

Singulární místa jsou ošetřena pouze strojním parametrem `$MC_TRAFO5_POLE_LIMIT` .

Literatura:
/FB3/, Příručka Popis funkcí, Speciální funkce; 3- až 5-osá transformace (F2),
kapitola "Singulární místa a jejich ošetření".

6.2.7 Programování orientačních os (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2)

Funkce

Funkce orientačních os popisuje orientaci nástroje v prostoru a daného nastavení se dosahuje naprogramováním offsetů kruhových os. Dalšího třetího stupně volnosti může být dosahováno dalším otáčením nástroje okolo sebe sama. Tyto orientace nástroje se libovolně v prostoru nastavují pomocí třetí kruhové osy a vyžadují transformaci se šesti osami. Vlastní otočení nástroje okolo sebe sama je definováno v závislosti na druhu interpolace vektorů otočení pomocí úhlu otočení THETA, viz kapitola "Otáčení orientace nástroje (ORIROTA/TR/TT, ORIROTC, THETA)".

Programování

Orientační osy jsou programovány prostřednictvím identifikátorů os A2, B2, C2.

`N... ORIAXES` nebo `ORIVECT` `N... G1 X Y Z A B C`	Lineární interpolace nebo interpolace pomocí největší kružnice koule
nebo	nebo
`N... ORIPLANE`	Interpolace orientace roviny
nebo	nebo
`N... ORIEULER` nebo `ORIRPY, příp. ORIRPY2` `N... G1 X Y Z A2= B2= C2=`	Úhel orientace - Eulerův úhel / úhel RPY. Programování úhlů virtuálních os
nebo	nebo
`N... ORIVIRT1` nebo `ORIVIRT2` `N... G1 X Y Z A3= B3= C3=`	Definice virtuálních orientačních os 1 nebo 2, programování směrového vektoru

Pro změny orientace podél kuželové plášťové plochy nacházející se v prostoru mohou být naprogramovány další offsety kruhových orientačních os, viz kapitola "Programování orientace podél kuželové plášťové plochy (ORIPLANE, ORICONxx).

Parametry

`ORIAXES`	`Lineární interpolace os stroje nebo orientačních os`
`ORIVECT`	`Interpolace pomocí největší kružnice koule (identická s ORIPLANE)`
`ORIMKS`	`Otáčení v souřadném systému stroje`
`ORIWKS`	`Otáčení v souřadném systému obrobku`
	`Popis viz kapitola "Otáčení orientace nástroje"`
`A= B= C=`	`Programování poloh os stroje`
`ORIEULER`	`Programování orientace pomocí Eulerova úhlu`

ORIRPY	Programování orientace pomocí úhlu RPY. Posloupnost při otáčení je X Y Z, přičemž platí: A2 je úhel otočení okolo osy X B2 je úhel otočení okolo osy Y C2 je úhel otočení okolo osy Z
ORIRPY2	Programování orientace pomocí úhlu RPY. Posloupnost při otáčení je Z Y X, přičemž platí: A2 je úhel otočení okolo osy Z B2 je úhel otočení okolo osy Y C2 je úhel otočení okolo osy X
A2= B2= C2=	Programování úhlů virtuálních os
ORIVIRT1	Programování orientace pomocí virtuálních orientačních os
ORIVIRT2	(Definice 1), stanovení podle MD $MC_ORIAX_TURN_TAB_1
	(Definice 2), stanovení podle MD $MC_ORIAX_TURN_TAB_2
A3= B3= C3=	Programování směrového vektoru směrové osy

Popis

Výrobce stroje

Pomocí strojního parametru $MC_ORI_DEF_WITH_G_CODE je stanoveno, jak jsou definovány naprogramované úhly A2, B2, C2:

Definice se uskutečňuje podle strojního parametru $MC_ORIENTATION_IS_EULER (standardní) nebo se definice uskutečňuje podle G-funkcí ze skupiny 50 (ORIEULER, ORIRPY, ORIVIRT1, ORIVIRT2).

Pomocí strojního parametru $MC_ORI_IPO_WITH_G_CODE se určuje, jaký druh interpolace se uplatňuje: ORIWKS/ORIMKS nebo ORIAXES/ORIVECT.

Provozní režim JOG

V tomto provozním režimu je úhel orientace interpolován vždy lineárně. Při spojitých a inkrementálních posuvech prostřednictvím tlačítek posuvů je možno pohybovat jen jednou orientační osou. Pomocí ručních koleček je možné pohybovat orientačními osami současně.

Pro manuální pohyby orientačními osami se uplatňuje kanálový korekční spínač posuvu, příp. korekční spínač rychlého posuvu.

Pomocí následujících strojních parametrů je možné oddělené nastavení rychlosti:

$MC_JOG_VELO_RAPID_GEO

$MC_JOG_VELO_GEO

$MC_JOG_VELO_RAPID_ORI

$MC_JOG_VELO_ORI

Poznámka

SINUMERIK 840D s "modulem pro práci s transformacemi"

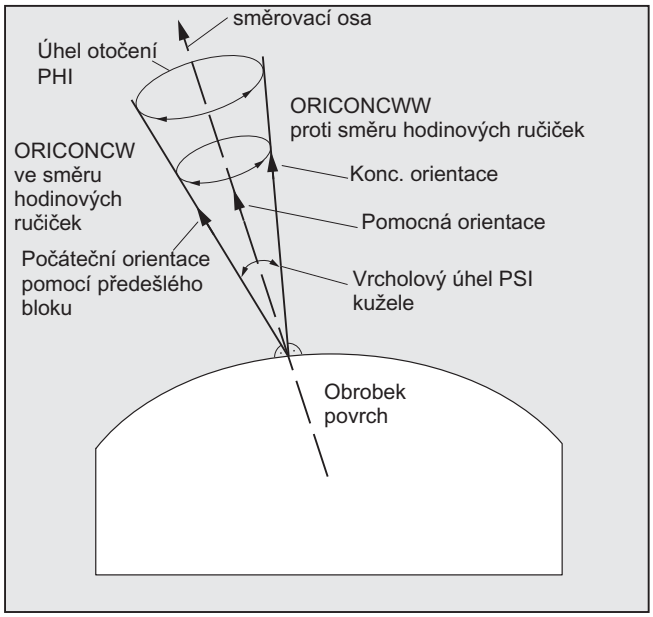
Pomocí funkce "Kartézské manuální pohyby" je možné v provozním režimu JOG od sebe odděleně nastavovat posunutí geometrických os ve vztažných systémech MCS, WCS a TCS.

Literatura:
/FB2/, Příručka Popis funkcí, Rozšiřovací funkce; Kinematická transformace (M1)

6.2.8 Programování orientace podél kuželové plášťové plochy (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO)

Funkce

Pomocí rozšířené orientace je možné uskutečňovat změny orientace podél kuželové plášťové plochy nacházející se v prostoru. Interpolace vektorů orientace se na kuželové plášťové ploše uskutečňuje pomocí modálních příkazů ORICONxx. Pro interpolaci v rovině může být naprogramována koncová orientace ORIPLANE. Obecně je počáteční orientace určena na základě předcházejících bloků.



Obrázek. 6-4

Programování

Koncová orientace se určuje buď prostřednictvím naprogramováním zadaného Eulerova úhlu nebo úhlu RPY pomocí proměnných A2, B2, C2 nebo naprogramováním poloh kruhových os A, B, C. Pro orientační osy podél kuželové plášťové plochy jsou zapotřebí další naprogramované údaje:

- Osa kužele jako vektor pomocí proměnných `A6, B6, C6`
- Úhel kužele `PSI` s identifikátorem `NUT`
- Pomocná orientace na plášti kužele pomocí proměnných `A7, B7, C7`

Poznámka

Programování směrového vektoru A6, B6, C6 pro osu kužele

Naprogramování koncové orientace není bezpodmínečně nutné. Jestliže koncová orientace není uvedena, potom se bude provádět interpolace na celém plášti kužele s úhlem 360 stupňů.

Programování úhlu kužele pomocí parametru NUT = úhel

Zadání koncové orientace je bezpodmínečně nutné.

Úplný plášť kužele s úhlem 360 stupňů není možné tímto způsobem interpolovat.

Programování pomocná orientace na plášti kužele pomocí proměnných A7, B7, C7

Zadání koncové orientace je bezpodmínečně nutné. Změna orientace a směr otáčení jsou jednoznačně určeny pomocí tří vektorů počáteční, koncové a pomocné orientace. Všechny tři vektory se přitom musí od sebe lišit. Jestliže je pomocná orientace rovnoběžná s počáteční nebo s koncovou orientací, potom se uskuteční lineární interpolace orientace pomocí největší kružnice koule v rovině, která je určena počátečním a koncovým vektorem.

Rozšířená interpolace orientace na kuželové plášťové ploše

`N... ORICONCW` nebo `ORICONCCW`	**Interpolace** na plášti kuželu s těmito parametry
`N... A6= B6= C6= A3= B3= C3=`	Směrový vektor kuželu ve/proti směru hodinových ručiček a koncová orientace
`nebo`	nebo
`N... ORICONTO`	tangenciální přechod a
`N... G1 X Y Z A6= B6= C6=`	zadání koncové orientace
nebo	nebo
`N... ORICONIO`	zadání koncové orientace a
`N... G1 X Y Z A7= B7= C7=`	pomocné orientace na plášti kužele s
`N... PO[PHI]=(a2, a3, a4, a5)`	těmito parametry
`N... PO[PSI]=(b2, b3, b4, b5)`	Polynomy pro úhel otočení a Polynomy pro úhel kužele

Parametry

ORIPLANE	Interpolace v rovině (interpolace pomocí největší kružnice koule)
ORICONCW	Interpolace po ploše pláště kužele ve směru hodinových ručiček
ORICONCCW	Interpolace po ploše pláště kužele proti směru hodinových ručiček
ORICONTO	Interpolace na ploše pláště kuželu s tangenciálním přechodem
A6= B6= C6=	Programování osy kužele (normovaný vektor)
NUT=úhel	Úhel kužele ve stupních
NUT=+179	Úhel pohybu menší nebo roven 180 stupňům
NUT=-181	Úhel pohybu větší nebo roven 180 stupňům
ORICONIO	Interpolace na kuželové plášťové ploše
A7= B7= C7=	Pomocná orientace (programuje se jako normovaný vektor)
PHI	Úhel otočení orientace okolo směrové osy kužele
PSI	Úhel kužele
možné polynomy PO[PHI]=(a2, a3, a4, a5) PO[PSI]=(b2, b3, b4, b5)	Kromě příslušných úhlů mohou být naprogramovány také polynomy maximálně 5. stupně

Příklad pro odlišné změny orientace

```
…
N10 G1 X0 Y0 F5000
N20 TRAORI(1)                              ; Aktivování transformace orientace.
N30 ORIVECT                                ; Interpolace orientace nástroje jako
                                             vektor.
…                                          ; Orientace nástroje v rovině.
N40 ORIPLANE                               ; Aktivování interpolace pomocí největší
                                             kružnice koule.
N50 A3=0 B3=0 C3=1
N60 A3=0 B3=1 C3=1                         ; Orientace v rovině Y/Z otočená o 45
                                             stupňů, na konci bloku bude dosaženo
                                             orientace (0,1/√2,1/√2).
…
N70 ORICONCW                               ; Programování orientace na plášti
                                             kužele:
N80 A6=0 B6=0 C6=1 A3=0 B3=0 C3=1          ; Interpolace vektoru orientace na plášti
                                             kužele se směrem (0,0,1) až do orientace
                                             (1/√2,0,1/√2) ve směru hodinových
                                             ručiček, úhel přitom činí 270 stupňů.
N90 A6=0 B6=0 C6=1                         ; Orientace nástroje proběhne celou jednu
                                             otáčku po stejném plášti kuželu.
```

Popis

Jestliže mají být popsány změny orientace na kuželové plášťové ploše nacházející se libovolně v prostoru, potom musí být znám vektor, okolo kterého se má orientace nástroje otáčet. Kromě toho musí být zadána počáteční a koncová orientace. Počáteční orientace vyplývá z předcházejícího bloku a koncová orientace musí být buď naprogramována nebo musí být stanovena na základě dalších podmínek.

Programování v rovině ORIPLANE odpovídá funkci ORIVECT

Naprogramování interpolace pomocí největší kružnice koule spolu s polynomy popisujícími úhel odpovídá lineární a polynomické interpolaci kontur. Orientace nástroje je interpolována v rovině, která je definována počáteční a koncovou orientací. Pokud je kromě toho naprogramován polynom, může se vektor orientace naklápět i mimo tuto rovinu.

Programování kruhových drah v rovině pomocí příkazů G2/G3, CIP a CT

Rozšířená orientace odpovídá interpolaci kruhových oblouků v rovině. Pokud budete potřebovat odpovídající informace o možnostech programování kruhových oblouků pomocí středu nebo rádiusu (G2/G3), pomocí vnitřního bodu CIP a kruhových oblouků s tangenciálním napojením, viz:

Literatura: Příručka programování, Základy; "Programování příkazů dráhy"

Programování orientace

Interpolace vektoru orientace na kuželové plášťové ploše (ORICONxx)

Pro interpolaci orientace na kuželové plášťové ploše mohou být používány čtyři různé druhy interpolace ze skupiny G-kódů č. 51.

1. Interpolace na plášti kužele ve směru hodinových ručiček `ORICONCW` se zadáním koncové orientace a směru kužele nebo úhlu kužele. Směrový vektor se programuje pomocí identifikátorů `A6, B6, C6` a vrcholový úhel kužele se programuje pomocí identifikátoru NUT= s intervalem hodnot 0 až 180 stupňů.
2. Interpolace na plášti kužele proti směru hodinových ručiček `ORICONCWW` se zadáním koncové orientace a směru kužele nebo úhlu kužele. Směrový vektor se programuje pomocí identifikátorů `A6, B6, C6` a vrcholový úhel kužele se programuje pomocí identifikátoru NUT= s intervalem hodnot 0 až 180 stupňů.
3. Interpolace na plášti kužele pomocí příkazu `ORICONIO` se zadáním koncové orientace a pomocné orientace, která se programuje pomocí identifikátorů `A7, B7, C7`.
4. Interpolace po ploše pláště kužele pomocí příkazu `ORICONTO` s tangenciálním přechodem a se zadáním koncové orientace. Směrový vektor se programuje pomocí identifikátorů `A6, B6, C6`.

6.2.9 Zadání orientace pomocí druhého styčného bodu (ORICURVE, PO[XH]=, PO[YH]=, PO[ZH]=)

Funkce

Programování změny orientace pomocí druhé prostorové křivky ORICURVE

Další možnost, jak naprogramovat změny orientace, spočítá v tom, že se kromě pohybu špičky nástroje podél prostorové křivky naprogramuje pomocí příkazu `ORICURVE` také pohyb druhého styčného bodu nástroje. Tímto způsobem mohou být jednoznačně definovány změny orientace nástroje stejně jako při naprogramování vektoru nástroje samotného.

Výrobce stroje

Věnujte prosím pozornost pokynům výrobce stroje, které se týkají identifikátorů osy, které jsou nastavitelné pomocí strojních parametrů a které slouží pro programování 2. dráhy pro popis orientace nástroje.

Programování

U tohoto druhu interpolace mohou být pro obě prostorové křivky naprogramovány body pomocí příkazů `G1`, příp. polynomy pomocí příkazu `POLY`. Kruhové oblouky a evolventy jsou nepřípustné. Kromě toho mohou být aktivovány splinová interpolace pomocí příkazu `BSPLINE` a funkce "Shrnutí krátkých splinových bloků"

Literatura:
/FB1/, Příručka Popis funkcí, Základní funkce, Režim řízení pohybu po dráze, přesné najetí a funkce Look Ahead (B1), kapitola: Shrnutí krátkých splinových bloků

Použití jiných druhů splinů `ASPLINE` a `CSPLINE`, stejně jako aktivování kompresoru pomocí příkazů `COMPON`, `COMPCURV` nebo `COMPCAD` nejsou přípustné.

Souřadnice pro pohyb dvou styčných bodů nástroje může být zadán naprogramováním polynomu orientace až maximálně 5. stupně.

Rozšířená interpolace orientace pomocí doplňkové prostorové křivky a polynomu pro souřadnice

```
N... ORICURVE
N... PO[XH]=(xe, x2, x3, x4, x5)
N... PO[YH]=(ye, y2, y3, y4, y5)
N... PO[ZH]=(ze, z2, z3, z4, z5)
```

Zadání pohybu druhého styčného bodu nástroje a doplňkový polynom příslušné souřadnice

Parametry

ORICURVE	Interpolace orientace s udáním pohybu dvou styčných bodů nástroje.
XH YH ZH	Identifikátory souřadnic druhého kontaktního bodu nástroje, které popisují doplňkovou konturu jako prostorovou křivku
možné polynomy PO[XH] = (xe, x2, x3, x4, x5) PO[YH] = (ye, y2, y3, y4, y5) PO[ZH] = (ze, z2, z3, z4, z5)	Kromě odpovídajících koncových bodů mohou být prostorové křivky naprogramovány také pomocí polynomů.
xe, ye, ze	Koncové body prostorové křivky
xi, yi, zi	Koeficienty polynomu maximálně 5. stupně

Poznámka

Identifikátory XH YH ZH pro programování 2. dráhy orientace

Tyto identifikátory musí být zvoleny tak, aby nemohlo dojít ke konfliktu s jinými identifikátory lineárních os

X Y Z - osy

a kruhovými osami, jako např.

A2 B2 C2 - Eulerův úhel, příp. úhel RPY

A3 B3 C3 - směrové vektory

A4 B4 C4 příp. A5 B5 C5 - normálové vektory plochy

A6 B6 C6 - vektory otáčení, příp. A7 B7 C7 souřadnice vnitřního bodu

nebo jinými interpolačními parametry.

6.3 Polynomický popis orientace (PO[úhel], PO[souřadnice])

Funkce

Nezávisle na tom, která polynomická interpolace ze skupiny G-kódů č.1 je momentálně aktivní, mohou být při tří- až pětíosé transformaci naprogramovány dva různé typy polynomů pro popis orientace až maximálně 5. stupně.

1. Polynom pro **úhel**: Předozadní úhel LEAD, boční úhel TILT
   vztažený na rovinu, která je definována počáteční a koncovou orientací.
2. Polynom pro **souřadnice**: XH, YH, ZH druhé prostorové křivky pro orientaci nástroje pomocí vztažného bodu na nástroji.

V případě transformace se šesti osami může být pro účely orientace nástroje naprogramováno navíc ještě i otáčení vektoru otočení THT pomocí polynomu maximálně 5. stupně pro otáčení nástroje okolo sebe sama.

Syntaxe

Polynom orientace pro typ 1 pro **úhel**

`N… PO[PHI]=(a2, a3, a4, a5)`	Tří- až pětiosá transformace
`N… PO[PSI]=(b2, b3, b4, b5)`	Tří- až pětiosá transformace

Polynom orientace pro typ 2 pro **souřadnice**

`N… PO[XH]=(xe, x2, x3, x4, x5)`	Identifikátor pro souřadnice druhé dráhy orientace pro orientaci nástroje
`N… PO[YH]=(ye, y2, y3, y4, y5)`	
`N… PO[ZH]=(ze, z2, z3, z4, z5)`	

Navíc může být v obou případech naprogramován polynom pro **otáčení** v případě transformace se šesti osami pomocí

`N… PO[THT]=(c2, c3, c4, c5)`	Interpolace otáčení vztažená k dráze
nebo	
`N… PO[THT]=(d2, d3, d4, d5)`	absolutní, relativní a tangenciální interpolace pro změnu orientace

vektoru orientace. To je možné tehdy, pokud je podporována transformace vektoru otočení s offsetem, který může být naprogramován a interpolován pomocí úhlu otočení THETA.

Význam

PO[PHI]	Úhel v rovině mezi počáteční a koncovou orientací
PO[PSI]	Popisuje úhel naklopení orientace ven z roviny mezi počáteční a koncovou orientací
PO[THT]	Úhel otočení, který vzniká otočením vektoru naprogramovaného pomocí G-kódu THETA ze skupiny 54
PHI	Předozadní úhel LEAD
PSI	Úhel bočního naklopení TILT
THETA	Otočení okolo směru nástroje v ose Z
PO[XH]	Souřadnice X vztažného bodu na nástroji
PO[YH]	Souřadnice Y vztažného bodu na nástroji
PO[ZH]	Souřadnice Z vztažného bodu na nástroji

Popis

Polynomy orientace nemusí být naprogramovány,

- pokud jsou aktivní splinové interpolace ASPLINE, BSPLINE, CSPLINE.
  Polynomy typu 1 pro úhel orientace jsou možné pro každý druh interpolace kromě splinů, tzn. při lineární interpolaci s rychlým posuvem G00, příp. s pracovním posuvem G01, při polynomické interpolaci POLY a při kruhové, příp. evolventní interpolaci s příkazy G02, G03, CIP, CT, INVCW a INCCCW.
  Polynom typu 2 pro souřadnice orientace jsou oproti tomu možné jen tehdy, pokud jsou aktivní lineární interpolace s rychlým posuvem G00, příp. s pracovním posuvem G01 nebo polynomická interpolace POLY.
- jestliže je orientace interpolována pomocí osové interpolace ORIAXES. V tomto případě mohou být přímo pomocí příkazů PO[A] a PO[B] naprogramovány polynomy pro orientační osy A a B.

Polynomy orientace typu 1 s příkazy ORIVECT, ORIPLANE a ORICONxx

V případě interpolace pomocí největší kružnice koule a interpolace plochy pláště kužele pomocí příkazů ORIVECT, ORIPLANE a ORICONxx je možné používat jedině polynomy orientace typu 1.

Polynom orientace pro typ 2 s příkazem ORICURVE

Jestliže je aktivní interpolace s doplňkovou prostorovou křivkou ORICURVE, jsou interpolovány kartézské složky vektoru orientace a tato operace je možná pouze s polynomem orientace typu 2.

6.4 Otáčení orientace nástroje (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA)

Funkce

Jestliže má u typů obráběcích strojů s pohyblivým nástrojem existovat možnost měnit orientaci nástroje, potom se v každém bloku programuje koncová orientace. V závislosti na kinematice stroje mohou být naprogramovány buď směr orientace orientačních os nebo směr otáčení vektoru orientace THETA. Pro tyto vektory otočení mohou být naprogramovány různé druhy interpolace:

- ORIROTA: Úhel otočení vztažen ke směru otáčení zadanému absolutně.
- ORIROTR: Úhel otočení relativně vůči rovině mezi počáteční a koncovou orientací.
- ORIROTT: Úhel otočení relativně vůči změně vektoru orientace.
- ORIROTC: Tangenciální úhel otočení k tečně dráhy.

Syntaxe

Jen tehdy, pokud je aktivován druh interpolace `ORIROTA`, může být úhel otočení nebo vektor otočení naprogramován jedním z následujících čtyř možných způsobů:

1. Přímé polohování kruhových os `A, B, C`
2. Eulerův úhel (ve stupních) pomocí `A2, B2, C2`
3. Úhel RPY (ve stupních) pomocí `A2, B2, C2`
4. Směrový vektor pomocí `A3, B3, C3` (úhel otočení pomocí `THETA=hodnota`)

Pokud jsou aktivní funkce `ORIOTR` nebo `ORIOTT`, může být úhel otočení naprogramován ještě i přímo pomocí příkazu `THETA`.

Otočení může být naprogramováno také samotně v bloku, aniž by se uskutečňovala změna orientace. Příkazy `ORIROTR` a `ORIROTT` přitom nemají žádný význam. V tomto případě je úhel otočení vždy interpretován ve vztahu k absolutnímu směru (`ORIROTA`).

`N... ORIROTA`
`N... ORIROTR`
`N... ORIROTT`
`N... ORIROTC`
	Definice interpolace vektoru otočení

`N... A3= B3= C3= THETA=hodnota`	Definice otáčení vektoru orientace
`N... PO[THT]=(`d_2`, `d_3`, `d_4`, `d_5`)`	Interpolace úhlu otočení pomocí polynomu 5. stupně

Význam

`ORIROTA`	Úhel otočení vztažen ke směru otáčení zadanému absolutně
`ORIROTR`	Úhel otočení relativně vůči rovině mezi počáteční a koncovou orientací
`ORIROTT`	Úhel otočení jako tangenciální vektor otočení ke změně orientace
`ORIROTC`	Úhel otočení jako tangenciální vektor otočení vůči tečně ke dráze
`THETA`	Otočení vektoru orientace
`THETA=hodnota`	Úhel otočení ve stupních, který bude dosažen na konci bloku
`THETA=Θe`	Úhel otočení s Eulerovým úhlem Θ_e tohoto vektoru
`THETA=AC(…)`	Blokové přepnutí na zadávání absolutních rozměrů
`THETA=AC(…)`	Blokové přepnutí na zadávání inkrementálních rozměrů
`Θe`	Koncový úhel vektoru otočení buď jako absolutní rozměr, když je aktivní G90, nebo jako relativní rozměr (inkrementální), když je aktivní G91
`PO[THT]=(....)`	Polynom pro úhel otočení

Příklad pro otáčení orientace

Programový kód	**Komentář**
`N10 TRAORI`	`; Aktivování transformace orientace`
`N20 G1 X0 Y0 Z0 F5000`	`; Orientace nástroje`
`N30 A3=0 B3=0 C3=1 THETA=0`	`; ve směru Z s úhlem otočení 0`
`N40 A3=1 B3=0 C3=0 THETA=90`	`; ve směru X a otočení o 90 stupňů`
`N50 A3=0 B3=1 C3=0 PO[THT]=(180,90)`	`; Orientace`
`N60 A3=0 B3=1 C3=0 THETA=IC(-90)`	`; ve směru Y a otočení do polohy 180 stupňů`
`N70 ORIROTT`	`; zůstává konstantní a otočení do polohy 90 stupňů`
`N80 A3=1 B3=0 C3=0 THETA=30`	`; Úhel otočení relativně vůči změně orientace`
	`; Vektor otočení pod úhlem 30 stupňů vůči rovině X-Y`

Při interpolaci bloku
`N40` se úhel otočení lineárně interpoluje od počáteční hodnoty 0 stupňů do koncové hodnoty 90 stupňů. V bloku `N50` se mění úhel otočení o 90 stupňů do polohy 180 stupňů podle paraboly $\theta(u) = +90u^2$. V bloku `N60` může být uvedeno také otočení, aniž by se konala nějaká změna orientace.

V bloku `N80` se orientace nástroje otáčí ze směru Y do směru X. Změna orientace přitom leží v rovině X-Y a vektor otočení svírá s touto rovinou úhel 30 stupňů.

Popis

ORIROTA

Úhel otočení `THETA` bude interpolován vzhledem k absolutně definovanému směru v prostoru. Základní směr otočení se zadává pomocí strojních parametrů.

ORIROTR

Úhel otočení `THETA` je interpretován vzhledem k rovině, která je definována počáteční a koncovou orientací.

ORIROTT

Úhel otočení `THETA` je interpretován vzhledem ke změně orientace- Pro případ `THETA=0` je vektor otočení interpolován tangenciálně vůči změněn orientace a od situace s příkazem `ORIROTR` se liší jen tehdy, pokud byl pro orientaci naprogramován minimálně jeden polynom pro "Úhel naklopení PSI". Z toho pak vyplývá změna orientace, která neprobíhá v dané rovině. Prostřednictvím dále naprogramovaného úhlu otočení `THETA` může být potom interpolován např. vektor otočení tak, že neustále zaujímá určitou hodnotu vůči změně orientace.

ORIROTC

Vektor otočení je interpolován vzhledem k tečně ke dráze s offsetem naprogramovaným pomocí úhlu `THETA`. Pro úhel offsetu přitom může být naprogramovaný také polynom `PO[THT]=(c2, c3, c4, c5)` maximálně 5. stupně.

6.5 Orientace vzhledem k dráze

6.5.1 Druhy orientace vztažené k dráze

Funkce

Pomocí této rozšiřovací funkce je dosahováno orientace vztažené nejen na konec bloku, nýbrž po celé dráze. Orientace dosažená v předcházejícím bloku se pomocí interpolace pomocí největší kružnice koule převádí do naprogramované koncové orientace. V zásadě existují dvě možnosti, jak naprogramovat požadovanou orientaci vzhledem k dráze:

1. Interpolace orientace nástroje také jako otáčení nástroje pomocí příkazů ORIPATH, ORPATHTS vzhledem k dráze.
2. Vektor orientace je naprogramován a interpolován stejně jako dříve. Pomocí příkazu ORIROTC se nastavuje otočení vektoru orientace vzhledem k tečně ke dráze.

Syntaxe

Druh interpolace orientace a otočení nástroje se programují pomocí těchto příkazů:

`N... ORIPATH`	Orientace vzhledem k dráze
`N... ORIPATHS`	Orientace vzhledem ke dráze s vyhlazováním charakteristiky orientace
`N... ORIROTC`	Interpolace vektoru otočení vzhledem ke dráze

Skoková změna orientace vyvolaná místem, kde se na dráze nachází roh, může být vyhlazena pomocí příkazu `ORIPATHS`. Směr a délka dráhy pohybu při pozvednutí nástroje se programuje pomocí vektoru se složkami `A8=X, B8=Y C8=Z`.

Pomocí příkazů `ORIPATH/ORIPATHS` mohou být naprogramovány různé vztahy vůči tečně ke dráze prostřednictvím tří úhlů

- `LEAD=` - zadání předozadního úhlu vztaženého na dráhu a na povrch
- `TILT=` - zadání úhlu bočního naklopení vztaženého na dráhu a na povrch
- `THETA=` - úhel otočení

pro celou dráhu. Pro úhel otočení `THETA` je možné pomocí příkazu `PO[THT]=(...)` naprogramovat navíc ještě i polynom maximálně 5. stupně.

Poznámka
Výrobce stroje

Věnujte prosím pozornost informacím od výrobce stroje. Prostřednictvím konfigurace strojních a nastavovaných parametrů mohou být realizována ještě i další nastavení týkající se druhu orientace vztaženého k dráze. Další vysvětlení viz:

Literatura:
/FB3/, Příručka Popis funkcí, Speciální funkce; 3- až 5-osá transformace (F2), kapitola "Orientace"

Význam

Interpolace úhlů `LEAD` a `TILT` může být pomocí strojního parametru nastavena různě:

- Vztažný směr orientace nástroje naprogramovaný pomocí příkazů `LEAD` a `TILT` zůstává po celou dobu trvání bloku zachován.
- Předozadní úhel `LEAD`: Otočení okolo směru, který je kolmý na tečnu a normálový vektor `TILT`: Otočení orientace okolo normálového vektoru.
- Předozadní úhel `LEAD`: Otočení okolo směru, který je kolmý na tečnu a normálový vektor, úhel bočního naklopení `TILT`: Otočení orientace okolo směru tečny ke dráze.
- Úhel otočení `THETA`: Otočení nástroje okolo sebe sama s doplňkovou třetí kruhovou osou, která představuje orientační osu u transformace se šesti osami.

Poznámka
Nastavování orientace vztažené ke dráze je při použití příkazů OSC, OSS, OSSE, OSD, OST nepřípustné

Interpolaci orientace vztahující se ke dráze pomocí příkazů `ORIPATH`, příp. `ORIPATHS` a `ORIOTC` není možné naprogramovat spolu s vyhlazením průběhu orientace pomocí G-kódů ze skupiny 34. Pro tento účel musí být aktivován příkaz OSOF.

6.5.2 Otáčení orientace nástroje vzhledem k dráze (ORIPATH, ORIPATHS, úhel otočení)

Funkce

V případě transformace se šesti osami může být orientace nástroje nastavována libovolně v prostoru a kromě toho se nástroj může pomocí třetí kruhové osy otáčet okolo sebe sama. V případě otáčení orientace nástroje pomocí příkazů ORIPATH, příp. ORIPATHS vztaženého ke dráze může být doplňkové otočení naprogramováno pomocí úhlu otočení THETA. Kromě toho existuje možnost naprogramovat pomocí příkazů LEAD a TILT vektor, který leží v rovině kolmé na směr nástroje.

Výrobce stroje

Věnujte prosím pozornost informacím od výrobce stroje. Pomocí strojního parametru může být interpolace úhlů LEAD a TILT nastavena odlišně.

Syntaxe

Otáčení orientace nástroje a nástroje samotného

Druh orientace nástroje vzhledem ke dráze se aktivuje pomocí příkazu ORIPATH nebo ORIPATHS.

`N... ORIPATH`	Aktivování druhu orientace vztaženého na dráhu
`N... ORIPATHS`	Aktivování druhu orientace vztaženého na dráhu s vyhlazením průběhu orientace

Aktivování tří možných úhlů způsobujících následující otočení:

`N... LEAD=`	Úhel pro naprogramovanou orientaci vzhledem k normálovému vektoru plochy
`N... TILT=`	Úhel pro naprogramovanou orientaci v rovině, která je kolmá na tečnu ke dráze vzhledem k normálovému vektoru plochy
`N... THETA=`	Úhel otočení třetí kruhové osy vzhledem ke změně orientace okolo směru nástroje

Hodnota úhlu na konci bloku se naprogramuje pomocí příkazů `LEAD=hodnota`, `TILT=hodnota`, příp. `THETA=hodnota`. Kromě konstantních úhlů mohou být pro všechny tři úhly naprogramovány polynomy maximálně 5. stupně.

`N... PO[PHI]=(a2, a3, a4, a5)`	Polynom pro předozadní úhel LEAD
`N... PO[PSI]=(b2, b3, b4, b5)`	Polynom pro úhel bočního naklopení TILT
`N... PO[THT]=(d2, d3, d4, d5)`	Polynom pro úhel otočení THETA

Při programování mohou být vypuštěny vyšší koeficienty polynomu, které jsou nulové. Příklad: `PO[PHI]=a2` udává pro předozadní úhel `LEAD` parabolu.

Význam

Orientace nástroje vztahující se ke dráze

`ORIPATH`	Orientace nástroje vztažená na dráhu
`ORIPATHS`	Orientace nástroje vztažená na dráhu, zlom v průběhu orientace se vyhladí
`LEAD`	Úhel je vztažen k normálovému vektoru plochy, a svírají jej tečna dráhy a normálový vektor plochy v dané rovině
`TILT`	Otočení orientace okolo směru Z, příp. otočení okolo tečny ke dráze
`THETA`	Otočení okolo směru nástroje (Z)
`PO[PHI]`	Polynom orientace pro předozadní úhel LEAD
`PO[PSI]`	Polynom orientace pro úhel bočního naklopení TILT
`PO[THT]`	Polynom orientace pro úhel otočení THETA

Poznámka
Úhel otočení THETA

Pro otočení nástroje okolo sebe sama pomocí třetí kruhové osy jako orientační osy je zapotřebí transformace se šesti osami.

6.5.3 Interpolace otočení nástroje vzhledem k dráze (ORIROTC, THETA)

Funkce

Interpolace s vektory otočení

K otočení nástroje vzhledem k tečně ke dráze naprogramovanému pomocí příkazu ORIROTC může být vektor otočení interpolován s offsetem, který je možno naprogramovat pomocí úhlu otočení THETA. Pro offsetová úhel přitom může být pomocí příkazu PO[THT] naprogramován polynom maximálně 5. stupně.

Syntaxe

`N... ORIROTC`	Nastavení otočení nástroje vzhledem k tečně ke dráze
`N... A3= B3= C3= THETA=hodnota`	Definice otáčení vektoru orientace
`N... A3= B3= C3= PO[THT]=(c2, c3, c4, c5)`	Interpolace úhlu offsetu pomocí polynomu 5. stupně

Otočení může být naprogramováno také samotně v bloku, aniž by se uskutečňovala změna orientace.

Význam

Interpolace otočení nástroje vztažená k dráze v případě transformace se šesti osami

`ORIROTC`	Nastavení tangenciálního vektoru otočení k tečně dráhy
`THETA=hodnota`	Úhel otočení ve stupních, který bude dosažen na konci bloku
`THETA=θe`	Úhel otočení s Eulerovým úhlem Θ_e tohoto vektoru
`THETA=AC(…)`	Blokové přepnutí na zadávání absolutních rozměrů
`THETA=IC(…)`	Blokové přepnutí na zadávání inkrementálních rozměrů
`PO[THT]=(c2, c3, c4, c5)`	Interpolace offsetového úhlu pomocí polynomu 5. stupně

Poznámka

Interpolace vektoru otočení ORIROTC

Jestliže má být vedle směru orientace nástroje nastavováno také otočení nástroje vzhledem k tečně ke dráze, potom je pro tuto operaci nutné používat transformaci se šesti osami.

Když je aktivní příkaz ORIROTC

Vektor otočení ORIROTA nemůže být naprogramován. V případě naprogramování se aktivuje alarm 14128 "Absolutní programování otočení nástroje při aktivním příkazu ORIROTC".

Směr orientace nástroje v případě tří- až pětiosé transformace

Směr orientace nástroje může být naprogramován stejně, jak je obvyklé v případě tří- až pětiosé transformace, pomocí Eulerova úhlu, příp. pomocí úhlu RPY nebo pomocí směrových vektorů. Jsou možné také změny orientace nástroje v prostoru prostřednictvím naprogramování interpolace pomocí největší kružnice koule s příkazem ORIVECT, lineární interpolace orientačních os ORIAXES, všech interpolací na kuželové plášťové ploše ORICONxx, jakož i interpolace pomocí dvou styčných bodů nástroje pomocí příkazu ORICURVE s další prostorovou křivkou.

`G....`	Zadání druhu pohybu kruhových os
`X Y Z`	Zadání pro lineární osy
`ORIAXES`	Lineární interpolace os stroje nebo orientačních os
`ORIVECT`	Interpolace pomocí největší kružnice koule (identická s ORIPLANE)
`ORIMKS`	Otáčení v souřadném systému stroje
`ORIWKS`	Otáčení v souřadném systému obrobku
	Popis viz kapitola "Otáčení orientace nástroje"
`A= B= C=`	Programování poloh os stroje
`ORIEULER`	Programování orientace pomocí Eulerova úhlu
`ORIRPY`	Programování orientace pomocí úhlu RPY.
`A2= B2= C2=`	Programování úhlů virtuálních os

`ORIVIRT1` `ORIVIRT2`	Programování orientace pomocí virtuálních orientačních os (Definice 1), stanovení podle MD $MC_ORIAX_TURN_TAB_1 (Definice 2), stanovení podle MD $MC_ORIAX_TURN_TAB_2
`A3= B3= C3=`	Programování směrového vektoru směrové osy
`ORIPLANE`	Interpolace v rovině (interpolace pomocí největší kružnice koule)
`ORICONCW`	Interpolace po ploše pláště kužele ve směru hodinových ručiček
`ORICONCCW`	Interpolace po ploše pláště kužele proti směru hodinových ručiček
`ORICONTO`	Interpolace na ploše pláště kuželu s tangenciálním přechodem
`A6= B6= C6=`	Programování osy kužele (normovaný vektor)
`NUT=úhel`	Úhel kužele ve stupních
`NUT=+179`	Úhel pohybu menší nebo roven 180 stupňům
`NUT=-181`	Úhel pohybu větší nebo roven 180 stupňům
`ORICONIO`	Interpolace na kuželové plášťové ploše
`A7= B7= C7=`	Pomocná orientace (programuje se jako normovaný vektor)
`ORICURVE` `XH YH ZH z.B. s polynomem PO[XH]=(xe, x2, x3, x4, x5)`	Interpolace orientace s udáním pohybu dvou styčných bodů nástroje. Kromě odpovídajících koncových bodů mohou být další prostorové křivky naprogramovány pomocí polynomu.

Poznámka

Pokud je orientace nástroje s aktivním příkazem ORIAXES interpolována pro orientační osy, potom je nastavení úhlu otočení vztažené ke dráze splněno jen na konci bloku.

6.5.4 Vyhlazení průběhu orientace (ORIPATHS A8=, B8=, C8=)

Funkce

V případě změn orientace s konstantním zrychlením na kontuře jsou přerušení pohybu po dráze, která se mohou vyskytnout zejména na rozích kontury, nežádoucí. Takto vznikající skoková změna na průběhu orientace může být vyhlazena vložením samostatného pomocného bloku. Jestliže je v průběhu změny orientace aktivní také příkaz ORIPATHS, změna orientace se potom uskutečňuje s konstantním zrychlením. V této fázi může být uskutečněno pozvednutí nástroje.

Výrobce stroje

Věnujte prosím pozornost informacím od výrobce stroje, pokud jde o případně se vyskytující předem definované strojní a nastavované parametry, s nimiž se tato funkce aktivuje.

Prostřednictvím strojního parametru může být nastaveno, jak je interpretován vektor pozvednutí.

1. V souřadném systému nástroje je souřadnice Z definována prostřednictvím směru nástroje.
2. V souřadném systému obrobku je souřadnice Z definována prostřednictvím aktivní roviny.

Pokud budete potřebovat další vysvětlení týkající se funkce "Orientace vzhledem k dráze", viz:
Literatura: /FB3/ Příručka Popis funkcí, Speciální funkce; 3- až 5-osá transformace (F2)

Syntaxe

Aby byly orientace nástroje vztažené na celkovou dráhu spojité, jsou zapotřebí v rozích kontury další programové údaje. Směr a délka dráhy tohoto pohybu se programují pomocí vektoru se složkami A8=X, B8=Y C8=Z.

```
N... ORIPATHS A8=X B8=Y C8=Z
```

Význam

`ORIPATHS`	Orientace nástroje vztažená na dráhu, zlom v průběhu orientace se vyhladí.
`A8= B8= C8=`	Vektorové složky pro směr a délku dráhy
`X, Y, Z`	Pohyb pro pozvednutí ve směru nástroje

Poznámka

Programování směrového vektoru A8, B8, C8

Pokud je délka tohoto vektoru rovna nule, žádné pozvednutí nástroje se neuskutečňuje.

ORIPATHS

Orientace nástroje vztažená ke dráze se aktivuje pomocí příkazu ORIPATHS. Jinak je orientace interpolována od počáteční do koncové orientace prostřednictvím lineární interpolace pomocí největší kružnice koule.

6.6 Komprimování orientace (COMPON, COMPCURV, COMPCAD)

Funkce

NC-programy, ve kterých je aktivní transformace orientace (`TRAORI`) a orientace nástroje (jakéhokoli druhu) jsou naprogramovány, mohou být komprimovány, přičemž jsou dodrženy předem nastavené tolerance.

Programování

Orientace nástroje

Jestliže je aktivní transformace orientace (`TRAORI`), může být u strojů s 5 osami programována transformace orientace následujícím způsobem (nezávisle na kinematice):

- Programování směrových **vektorů** prostřednictvím příkazů:

  ```
  A3=<...> B3=<...> C3=<...>
  ```

- Programování **Eulerova úhlu**, příp. **úhlu RPY** prostřednictvím příkazů:

  ```
  A2=<...> B2=<...> C2=<...>
  ```

Otočení nástroje

U strojů se **6 osami** je možné navíc ještě k orientaci nástroje naprogramovat také otočení nástroje.

Pro programování otočení nástroje se používá příkaz:

```
THETA=<...>
```

Viz "Otáčení orientace nástroje (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) [Strana 354]".

Poznámka

NC bloky, ve kterých je naprogramováno ještě i otočení, mohou být komprimovány jen tehdy, pokud se úhel otočení mění **lineárně**. Tzn. pro úhel otočení nesmí být naprogramován žádný polynom `PO[THT]=(...)`.

Všeobecná forma komprimovatelných NC bloků

Z těchto důvodů mohou mít komprimovatelné NC bloky následující obecnou formu:

```
N... X=<...> Y=<...> Z=<...> A3=<...> B3=<...> C3=<...> THETA=<...> F=<...>
```

nebo

```
N... X=<...> Y=<...> Z=<...> A2=<...> B2=<...> C2=<...> THETA=<...> F=<...>
```

Poznámka

Hodnoty polohy mohou být zadávány buď přímo (např. X90) nebo nepřímo pomocí přiřazení parametrů (např. X=R1*(R2+R3)).

Programování orientace nástroje pomocí poloh kruhových os

Orientace nástroje může být zadávána také pomocí poloh kruhových os, např. ve formě:

```
N... X=<...> Y=<...> Z=<...> A=<...> B=<...> C=<...> THETA=<...> F=<...>
```

V tomto případě se komprimování uskutečňuje dvěma různými způsoby v závislosti na tom, zda se provádí interpolace pomocí největší kružnice koule nebo neprovádí. Jestliže se interpolace pomocí největší kružnice koule nekoná, potom je komprimovaná změna orientace reprezentována obvyklým způsobem pomocí axiálních polynomů pro kruhové osy.

Přesnost kontury

V závislosti na nastaveném režimu komprese (MD20482 $MC_COMPRESSOR_MODE) jsou pro geometrické osy a orientační osy při kompresi uplatňovány buď v konfiguraci nastavené tolerance pro specifické osy (MD33100 $MA_COMPRESS_POS_TOL) nebo následující kanálové tolerance nastavitelné pomocí strojních parametrů:

SD42475 $SC_COMPRESS_CONTUR_TOL (maximální odchylka od kontury)

SD42476 $SC_COMPRESS_ORI_TOL (maximální úhlová odchylka pro orientaci nástroje)

SD42477 $SC_COMPRESS_ORI_ROT_TOL (maximální úhlová odchylka pro úhel otočení nástroje (k dispozici jen u strojů se 6 osami)

Literatura:
Příručka Popis funkcí, Základní funkce; 3- až 5-osá transformace (F2),
kapitola: "Komprimování orientace"

Aktivování / deaktivování

Funkce kompresoru se aktivují promocí G-kódů s modální platností `COMPON`, `COMPCURV`, příp. `COMPCAD`.

Funkce kompresoru se příkazem `COMPOF` ukončí.

Viz "Komprese NC-bloků (COMPON, COMPCURV, COMPCAD, COMPOF) [Strana 258]".

Poznámka

Orientační pohyby budou komprimovány jen tehdy, když je aktivní interpolace pomocí největší kružnice koule (tzn. změna orientace nástroje se uskutečňuje v rovině, která je určena počáteční a koncovou orientací).

Interpolace pomocí největší kružnice koule se provádí, pokud jsou splněny následující podmínky:

- MD21104 $MC_ORI_IPO_WITH_G_CODE = 0,
  ORIWKS je aktivní a
  Orientace je naprogramována pomocí vektorů (pomocí příkazů A3, B3, C3, příp. A2, B2, C2).
- MD21104 $MC_ORI_IPO_WITH_G_CODE = 1 a
  ORIVECT, příp. ORIPLANE je aktivní.

  Orientace nástroje může být naprogramována buď jako směrový vektor nebo pomocí poloh kruhových os. Jestliže je aktivní některý z G-kódů `ORICONxx` nebo `ORICURVE` nebo pokud jsou naprogramovány polynomy pro úhel orientace (`PO[PHI]` a `PO[PSI]`), interpolace pomocí největší kružnice koule se neprovádí.

Příklad

V následujícím příkladu programování bude komprimován kruh, který je aproximován polynomickou křivkou. Orientace nástroje se přitom pohybuje synchronně po plášti kužele. Přestože postupně za sebou naprogramované změny orientace mohou být nespojité, generuje funkce kompresoru hladký průběh orientace.

```
Programování                              Komentář
DEF INT ANZAHL=60
DEF REAL RADIUS=20
DEF INT COUNTER
DEF REAL WINKEL
N10 G1 X0 Y0 F5000 G64

$SC_COMPRESS_CONTUR_TOL=0.05              ; Maximální odchylka od kontury = 0.05 mm
$SC_COMPRESS_ORI_TOL=5                    ; Maximální odchylka orientace = 5 stupňů

TRAORI
COMPCURV
                                          ; Pohyb se bude uskutečňovat po kruhu, který
                                            se skládá z polynomů. Orientace se přitom
                                            pohybuje po plášti kužele okolo osy Z a s
                                            vrcholovým úhlem 45 stupňů.
N100 X0 Y0 A3=0 B3=-1 C3=1
N110 FOR COUNTER=0 TO ANZAHL
N120 WINKEL=360*COUNTER/ANZAHL
N130 X=RADIUS*cos(WINKEL) Y=RADIUS*sin(WINKEL)
A3=sin(WINKEL) B3=-cos(WINKEL) C3=1
N140 ENDFOR
```

6.7 Zapnutí vyhlazování charakteristiky orientace (ORISON, ORISOF)

Funkce

Pomocí funkce "Vyhlazení průběhu orientace (ORISON)" mohou být vyhlazeny výchylky orientace přes několik bloků. Tím je dosaženo hladkého průběhu jak orientace, tak také kontury.

Předpoklady

Funkce "Vyhlazení průběhu orientace (ORISON)" je k dispozici pouze u systémů, v nichž je instalována transformace s 5/6 osami.

Syntaxe

```
ORISON
...
ORISOF
```

Význam

ORISON:	Zapnutí vyhlazování charakteristiky orientace	
	Platnost:	modální
ORISOF:	Vypnutí vyhlazování charakteristiky orientace	
	Platnost:	modální

Nastavované parametry

Vyhlazování průběhu orientace se uskutečňuje tehdy, jsou-li dodrženy:

- předem zadaná maximální tolerance (maximální úhlová odchylka orientace nástroje ve stupních)

  a

- Předem definovaná maximální dráha.

Tyto údaje jsou definovány pomocí nastavovaných parametrů:

- SD42678 $SC_ORISON_TOL (tolerance pro vyhlazení průběhu orientace)
- SD42680 O$SC_ORISON_DIST (dráha pro vyhlazení průběhu orientace)

Příklad

Programový kód	Komentář
...	
TRAORI()	; Aktivování transformace orientace..
ORISON	; Aktivování vyhlazování orientace.
$SC_ORISON_TOL=1.0	; Tolerance vyhlazení orientace = 1,0 stupně.
G91	
X10 A3=1 B3=0 C3=1	
X10 A3=-1 B3=0 C3=1	
X10 A3=1 B3=0 C3=1	
X10 A3=-1 B3=0 C3=1	
X10 A3=1 B3=0 C3=1	
X10 A3=-1 B3=0 C3=1	
X10 A3=1 B3=0 C3=1	
X10 A3=-1 B3=0 C3=1	
X10 A3=1 B3=0 C3=1	
X10 A3=-1 B3=0 C3=1	
...	
ORISOF	; Deaktivování vyhlazování orientace.
...	

Orientace bude naklopena o 90 stupňů v rovině XZ od úhlu -45 stupňů do úhlu +45 stupňů. Prostřednictvím vyhlazování průběhu orientace není už dosahováno maximálních úhlových hodnot orientace -45 stupňů, příp. +45 stupňů.

Další informace

Počet bloků

Vyhlazování průběhu orientace se uskutečňuje přes určitý počet bloků nastavený v konfiguraci, který je uložen ve strojním parametru MD28590 $MC_MM_ORISON_BLOCKS.

Poznámka

Jestliže je vyhlazování průběhu orientace pomocí příkazu `ORISON` aktivováno, aniž by byl v konfiguraci pro tento účel vyčleněn dostatek paměťového prostoru (MD28590 < 4), potom se aktivuje alarmové hlášení a funkci nebude možné uskutečnit.

Maximální délka bloků

Průběh orientace se bude vyhlazovat jen v takových blocích, jejichž dráha posuvu je kratší, než je v konfiguraci nastavená maximální délka dráhy pro jeden blok (MD20178 $MC_ORISON_BLOCK_PATH_LIMIT). V blocích, v nichž je dráha posuvu delší, se vyhlazování přeruší a pohyb se uskuteční, jak bylo naprogramováno.

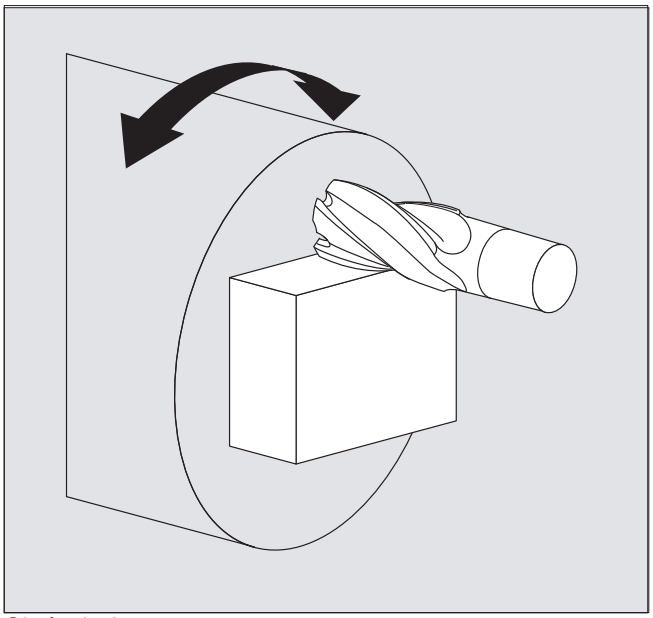
6.8 Kinematická transformace

6.8.1 Frézovací práce na rotačních součástech (TRANSMIT)

Funkce

Funkce TRANSMIT umožňuje provádět následující operace:

- Obrábění na čelní straně na soustružených dílech při upnutí pro soustružení (vrtání, kontury)
- Pro programování těchto obráběcích prací se může používat kartézský souřadný systém.
- Řídící systém transformuje naprogramované pohyby posuvů v kartézském souřadném systému na pohyby posuvů reálných os stroje (standardní případ:
  - Kruhová osa
  - Přísuvná osa kolmo na rotační osu
  - Podélná osa rovnoběžně s kruhovou osou
  - Lineární osy jsou vůči sobě kolmé.
- Posunutí středu nástroje vůči ose otáčení je přípustné.
- Regulace rychlosti bere ohled na omezení definovaná pro rotační pohyby.



Obrázek. 6-5

Typy transformace TRANSMIT

Pro obrábění pomocí funkce TRANSMIT existují dva nastavitelné výrazy:

- `TRANSMIT` ve standardním případě s (TRAFO_TYPE_n = 256)
- `TRANSMIT` s doplňkovou lineární osou Y (TRAFO_TYPE_n = 257)

Rozšířený typ transformace 257 se může používat např. ke kompenzaci korekcí upnutí nástroje s reálnou osou Y.

Syntaxe

`TRANSMIT` nebo `TRANSMIT(n)`

`TRAFOOF`

Kruhová osa

Kruhová osa nemůže být naprogramována, protože je obsazena geometrickou osou A a v důsledku toho nemůže být přímo programována jako osa kanálová.

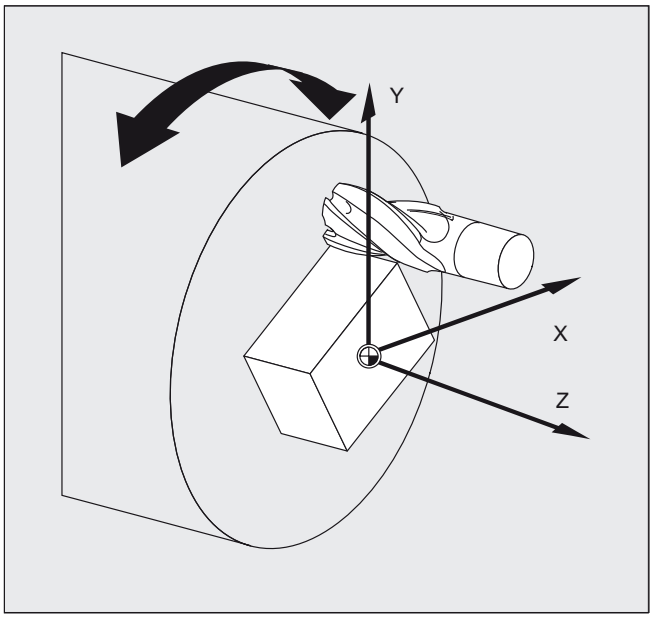
Význam

`TRANSMIT`:	Aktivování první deklarované funkce TRANSMIT. Tato funkce je označována také jako "polární transformace".
`TRANSMIT(n)`:	Aktivování n-té deklarované funkce TRANSMIT; n smí být maximálně 2 (TRANSMIT(1) odpovídá TRANSMIT)
`TRAFOOF`:	Vypnutí aktivní transformace
`OFFN`:	Normální offset kontury: Vzdálenost obrábění na čelní ploše od naprogramované vztažné kontury

Poznámka

Aktivní transformace `TRANSMIT` se vypne také tehdy, když je v příslušném kanálu aktivována některá ze zbývajících transformací (např. `TRACYL`, `TRAANG`, `TRAORI`).

Příklad



Programový kód	Komentář
N10 T1 D1 G54 G17 G90 F5000 G94	; Volba nástroje
N20 G0 X20 Z10 SPOS=45	; Najíždění na výchozí pozici
N30 TRANSMIT	; Aktivování funkce TRANSMIT
N40 ROT RPL=-45 N50 ATRANS X-2 Y10	; Nastavení framu
N60 G1 X10 Y-10 G41 OFFN=1OFFN N70 X-10 N80 Y10 N90 X10 N100 Y-10	; Obrábění čtyřhranu nahrubo, přídavek rozměru pro opracování načisto 1 mm
N110 G0 Z20 G40 OFFN=0 N120 T2 D1 X15 Y-15 N130 Z10 G41	; Výměna nástroje
N140 G1 X10 Y-10 N150 X-10 N160 Y10 N170 X10 N180 Y-10	; Obrábění čtyřhranu načisto
N190 Z20 G40 N200 TRANS N210 TRAFOOF	; Deaktivování framu
N220 G0 X20 Z10 SPOS=45	; Najíždění na výchozí pozici
N230 M30	

Popis

Pól

Pokud potřebujete projet skrz pól, existují dvě možnosti:

- Pohyb samotné lineární osy
- Najetí do pólu s otáčením kruhové osy v pólu a následné vyjetí z pólu

Volba se nastavuje pomocí parametrů MD 24911 a 24951.

TRANSMIT s doplňkovou lineární osou Y (typ transformace 257):

Tato varianta polární transformace využívá u strojů s další lineární osou dalších možností nastavení, aby se zlepšilo uplatňování korekcí nástroje. Pro druhou lineární osu potom platí:

- menší pracovní rozsah a
- druhá lineární osa se nemá používat pro zpracovávání výrobních programů

Pro výrobní program a přiřazení odpovídajících os v systémech BCS nebo MCS se předem nastavují hodnoty do určitých strojních parametrů, viz:

Literatura

/FB2/, Příručka Popis funkcí, Rozšiřovací funkce; Kinematické transformace (M1)

6.8.2 Transformace válcového pláště (TRACYL)

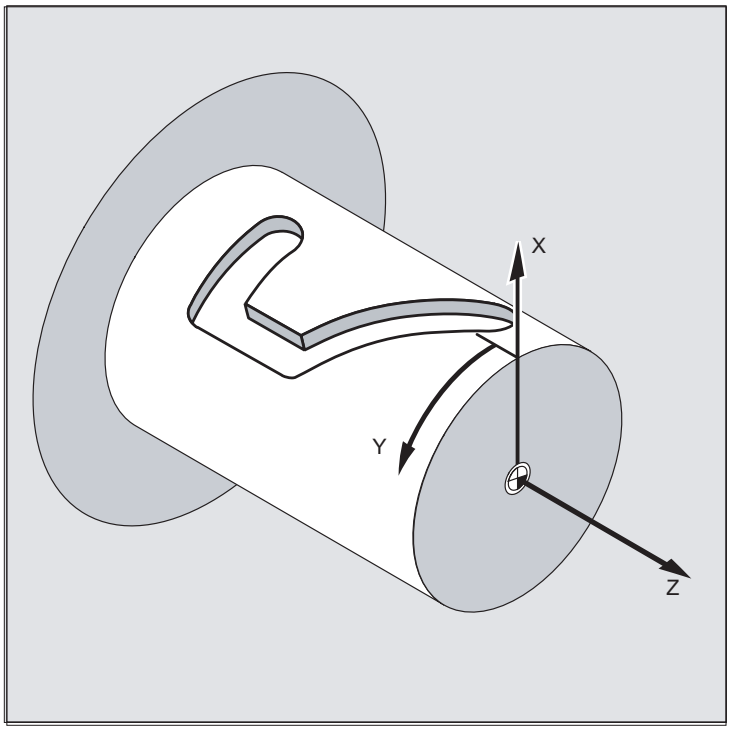
Funkce

Transformace křivek na válcovém plášti TRACYL umožňuje provádět následující výkony:

Opracování

- Podélných drážek na válcových tělesech
- Příčných drážek na válcových tělesech
- Drážek libovolného průběhu na válcových tělesech

Průběh drážek se programuje ve vztahu na rovnou plochu rozvinutého válce.



Typy transformace TRACYL

Transformace souřadnic na válcovém plášti existuje ve dvou podobách:

- `TRACYL` bez korekce stěny drážky: (TRAFO_TYPE_n=512)
- `TRACYL` s korekcí stěny drážky: (TRAFO_TYPE_n=513)
- `TRACYL` s doplňkovou lineární osou a s korekcí stěny drážky: (TRAFO_TYPE_n=514)
  Korekce stěny drážky s příkazem `TRACYL` se nastavuje pomocí třetího parametru.

Při transformaci křivek na válcovém plášti s korekcí stěny drážky by se měla osa používaná pro korekci nastavit do nulové polohy (y=0), aby se při výrobě drážky střed nástroje pohyboval po ose drážky.

Využití os

Následující osy nemohou být používány jako polohovací osy, příp. osy vykonávající kyvný pohyb:

- Geometrické osy v obvodovém směru plášťové plochy válce (osa Y)
- Doplňkové lineární osy při korekci stěny drážky (osa Z)

Syntaxe

`TRACYL(d)` nebo `TRACYL(d, n)` nebo

pro typ transformace 514

`TRACYL(d, n, korekce stěny drážky)`

`TRAFOOF`

Kruhová osa

Kruhová osa nemůže být naprogramována, protože je obsazena geometrickou osou A a v důsledku toho nemůže být přímo programována jako osa kanálová.

Význam

`TRACYL(d)`	Aktivuje první v kanálových strojních parametrech deklarovanou funkci TRACYL. Parametr d udává pracovní průměr.
`TRACYL (d, n)`	Aktivování n-té v kanálových strojních parametrech deklarované funkce TRACYL; n smí být maximálně 2 (TRACYL(d,1) odpovídá TRACYL(d)).
`D`	Tato hodnota udává pracovní průměr. Pracovní průměr odpovídá dvojnásobku vzdálenosti mezi špičkou nástroje a osou otáčení. Tento průměr musí být vždy zadán a musí být větší než 1.
`n`	Nepovinný 2. parametr pro datový blok funkce TRACYL 1 (předdefinované nastavení) nebo 2.
`Korekce stěny drážky`	Nepovinný 3. parametr jehož hodnota pro funkci TRACYL je předem nastavena ve strojních parametrech a odpovídá režimu. Rozsah hodnot: 0: Typ transformace 514 bez korekce stěny drážky, jako dříve 1: Typ transformace 514 s korekcí stěny drážky
`TRAFOOF`	Deaktivování transformace (BCS s MCS jsou opět identické).
`OFFN`	Normální offset kontury: Vzdálenost stěny drážky od naprogramované vztažné kontury

Poznámka

Aktivní transformace `TRACYL` se vypne také tehdy, když je v příslušném kanálu aktivována některá ze zbývajících transformací (např. `TRANSMIT`, `TRAANG`, `TRAORI`).

Příklad: Definice nástroje

Následující příklad ukazuje, jak je možno otestovat nastavení parametrů pro transformaci plášťové plochy válce TRACYL:

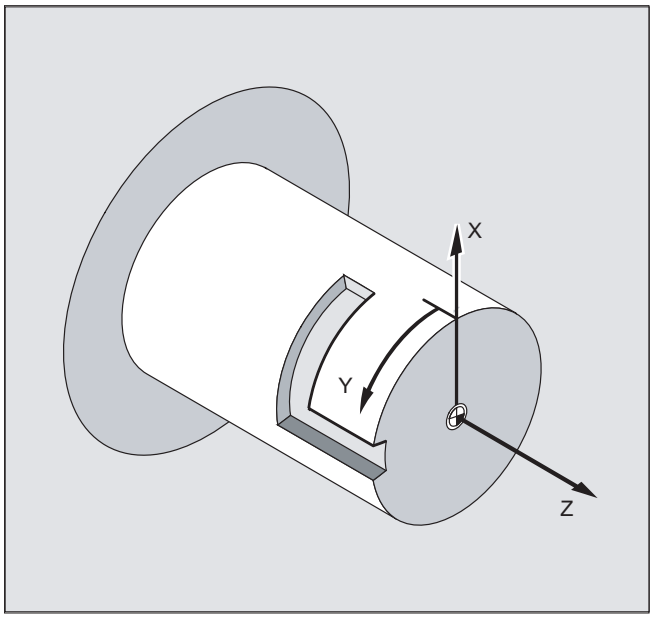
Programový kód	**Komentář**	
Parametry nástroje Číslo (DP)	Význam	Poznámka
$TC_DP1[1,1]=120	Typ nástroje	Fréza
$TC_DP2[1,1]=0	Poloha břitu	jen pro soustružnické nástroje

Programový kód	**Komentář**	
Geometrie	Korekce délky	
$TC_DP3[1,1]=8.	Vektor pro korekci délky	Započítání podle typu
$TC_DP4[1,1]=9.		a roviny
$TC_DP5[1,1]=7.		

Programový kód	**Komentář**	
Geometrie	Rádius	
$TC_DP6[1,1]=6.	Rádius	Rádius nástroje
$TC_DP7[1,1]=0	Šířka drážky b pro drážkovou pilu, rádius zaoblení pro frézovací nástroje	
$TC_DP8[1,1]=0	Přesah k	jen pro drážkovou pilu
$TC_DP9[1,1]=0		
$TC_DP10[1,1]=0		
$TC_DP11[1,1]=0	Úhel pro kuželovité frézovací nástroje	

Programový kód	**Komentář**	
Opotřebení	Korekce délky a rádiusu	
$TC_DP12[1,1]=0	Zbývající parametry až do $TC_DP24=0	Základní rozměr / adaptér

Příklad: Výroba drážky ve tvaru L



Aktivování transformace válcového pláště:

Programový kód	Komentář
N10 T1 D1 G54 G90 F5000 G94	; Volba nástroje, kompenzace upnutí
N20 SPOS=0 N30 G0 X25 Y0 Z105 CC=200	; Najíždění na výchozí pozici
N40 TRACYL (40)	; Aktivování transformace křivek na válcovém plášti
N50 G19	; Volba roviny

Výroba drážky ve tvaru L:

Programový kód	Komentář
N60 G1 X20	; Nastavení nástroje na dno drážky
N70 OFFN=12	; Definice vzdálenost mezi osou a stěnou drážky 12 mm
N80 G1 Z100 G42	; Najíždění na pravou stěnu drážky
N90 G1 Z50	; Úsek drážky rovnoběžný s osou válce
N100 G1 Y10	; Úsek drážky rovnoběžný s obvodem
N110 OFFN=4 G42	; Najíždění na levou stěnu drážky, definice vzdálenosti mezi stěnou a osou drážky 4 mm
N120 G1 Y70	; Úsek drážky rovnoběžný s obvodem
N130 G1 Z100	; Úsek drážky rovnoběžný s osou válce
N140 G1 Z105 G40	; Odjíždění od stěny drážky
N150 G1 X25	; Volné vyjíždění nástroje
N160 TRAFOOF	
N170 G0 X25 Y0 Z105 CC=200	; Najíždění na výchozí pozici
N180 M30	

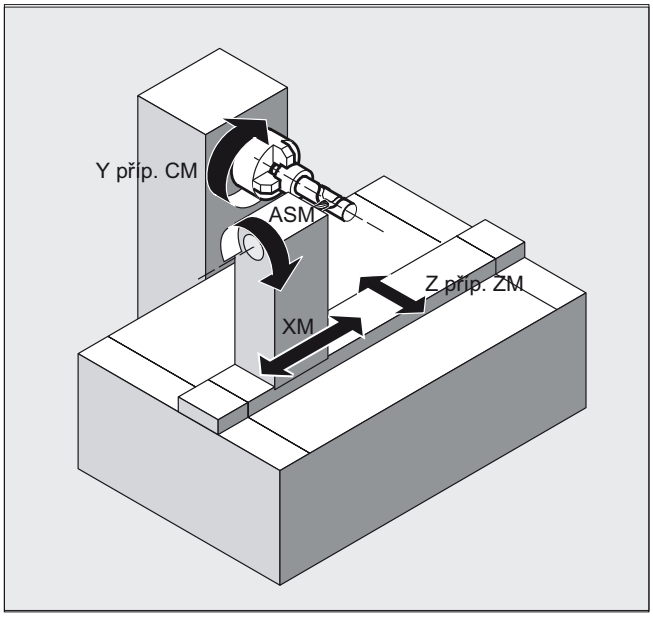
Popis

Bez korekce stěny drážky (typ transformace 512):

Řídící systém transformuje naprogramované pohyby posuvů ve válcovém souřadném systému na pohyby posuvů reálných os stroje:

- Kruhová osa
- Přísuvná osa kolmo na rotační osu
- Podélná osa rovnoběžně s kruhovou osou

Lineární osy jsou vůči sobě kolmé. Přísuvná osa protíná kruhovou osu.

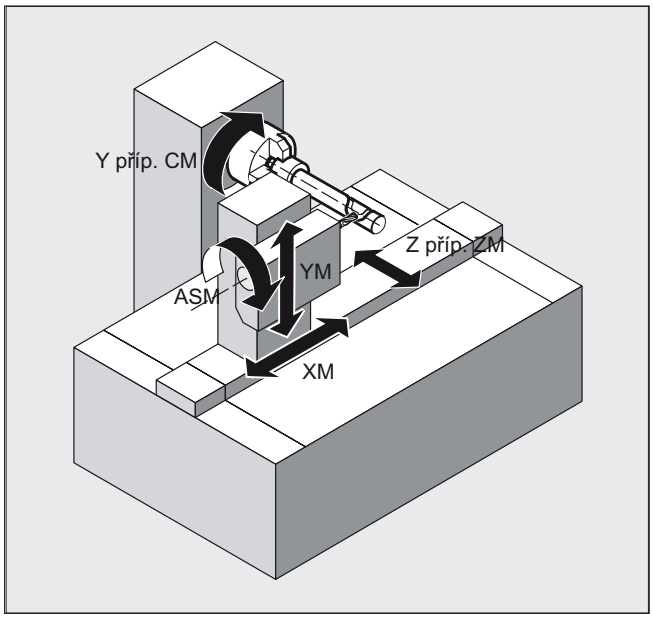


S korekcí stěny drážky (typ transformace 513):

Kinematika viz výše, ale navíc - podélná osa rovnoběžně s obvodovým směrem

Lineární osy jsou vůči sobě kolmé.

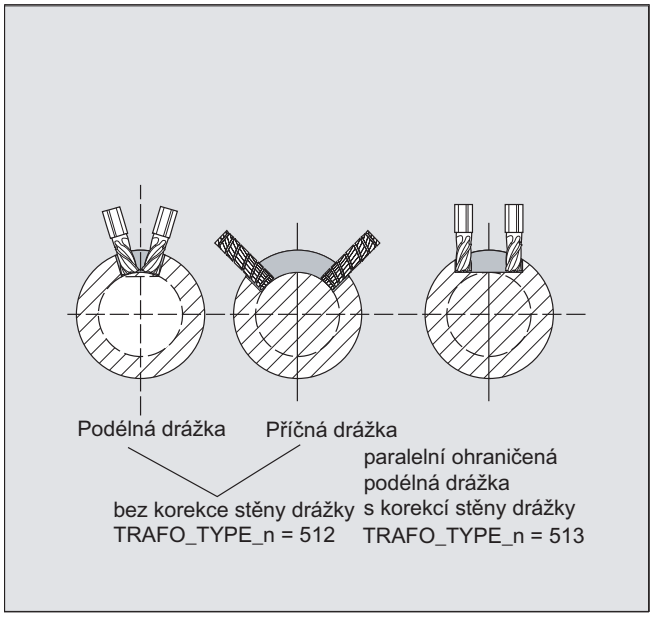
Regulace rychlosti bere ohled na omezení definovaná pro rotační pohyby.



Průřez drážky

V 1. případě konfigurace os jsou drážky v podélném směru s rovnoběžnými stěnami jen tehdy, pokud šířka drážky přesně odpovídá průměru nástroje.

Drážky rovnoběžné s obvodem (příčné drážky) nejsou na začátku a na konci rovnoběžné.



S doplňkovou lineární osou a s korekcí stěny drážky (typ transformace 514):

Tato varianta transformace využívá u strojů s další lineární osou dalších možností nastavení, aby se zlepšilo uplatňování korekcí nástroje. Pro druhou lineární osu potom platí:

- menší pracovní rozsah a
- druhá lineární osa se nemá používat pro zpracovávání výrobních programů

Pro výrobní program a přiřazení odpovídajících os v systémech BCS nebo MCS se předem nastavují hodnoty do určitých strojních parametrů, viz:

Literatura

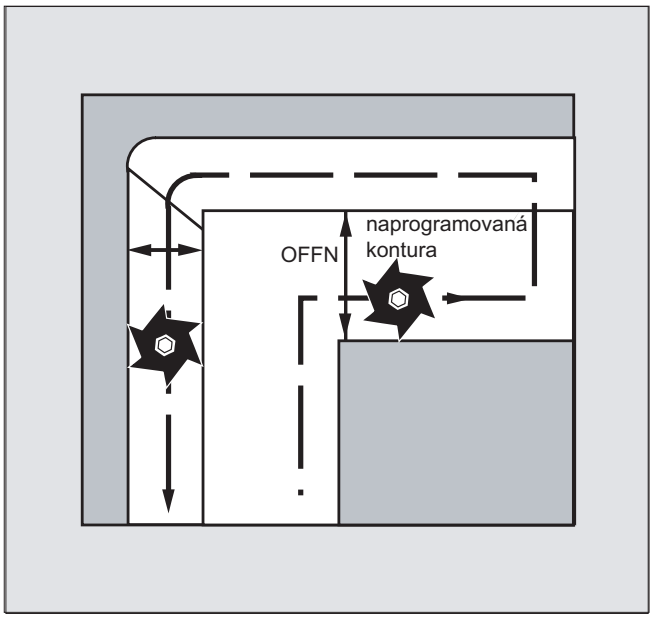
/FB2/, Příručka Popis funkcí, Rozšiřovací funkce; Kinematické transformace (M1)

Offset kontury OFFN (typ transformace 513)

Aby bylo možné pomocí funkce `TRACYL` frézovat drážky, je zapotřebí

- ve výrobním programu naprogramovat osu drážky,
- pomocí příkazu `OFFN` naprogramovat **polovinu šířky drážky**.

Příkaz `OFFN` se uplatní teprve spolu s aktivovanou korekcí rádiusu nástroje, aby se zabránilo poškození stěny drážky. **Kromě toho by mělo být OFFN >= rádius nástroje, aby se vyloučilo poškození protilehlé stěny drážky.**



Výrobní program pro frézování drážky se zpravidla skládá z následujících kroků:

1. Vyberte nástroj
2. Aktivujte funkci TRACYL
3. Zvolte vhodné posunutí počátku souřadného systému (FRAME)
4. Najeďte na požadovanou polohu
5. Naprogramujte OFFN
6. Aktivujte korekci rádiusu nástroje
7. Najížděcí blok (najíždění na korekci rádiusu nástroje a najíždění na stěnu drážky)
8. Kontura osy
9. Deaktivujte korekci rádiusu nástroje
10. Odjížděcí blok (odjíždění od korekce rádiusu nástroje a odjíždění od stěny drážky)
11. Najeďte na požadovanou polohu

12. TRAFOOF

13. Znovu aktivujte původní posunutí počátku souřadného systému (FRAME)

Zvláštnosti

- Aktivování korekce rádiusu nástroje:

  Korekce rádiusu nástroje nebere ohled na stěnu drážky, nýbrž místo toho je naprogramována vzhledem k naprogramované ose drážky. Aby se nástroj pohyboval vlevo od stěny drážky, zadá se příkaz G42 (nikoli G41). Tomu zabráníte tak, že v příkazu OFFN zadáte šířku drážky se záporným znaménkem.

- Příkaz OFFN s funkcí TRACYL funguje jinak než bez funkce TRACYL. Protože se hodnota OFFN započítává i bez funkce TRACYL, když je aktivní korekce rádiusu nástroje, po příkazu TRAFOOF by měla být do OFFN znovu dosazena nula.

- Změny parametru OFFN v rámci výrobního programu jsou možné. Tím by mohla být osa drážky posunuta ze středu (viz obrázek).

- Vodicí drážky:

  Pomocí funkce TRACYL nejsou v případě vodicích drážek vyráběny stejné drážky, jaké by vznikly, kdyby byly vyrobeny nástrojem, jehož průměr odpovídá šířce drážky. V principu není možné vyrábět pomocí malého válcového nástroje stejnou geometrii stěn drážky jako pomocí velkého. Transformace TRACYL minimalizuje chyby. Aby se zabránilo problémům s přesností, měl by být rádius nástroje jen o málo menší než je polovina šířky drážky.

Poznámka
OFFN a korekce rádiusu nástroje

Jestliže je nastaveno TRAFO_TYPE_n = 512, uplatňuje se hodnota zadaná do parametru OFFN jako přídavek rozměru ke korekci rádiusu nástroje.

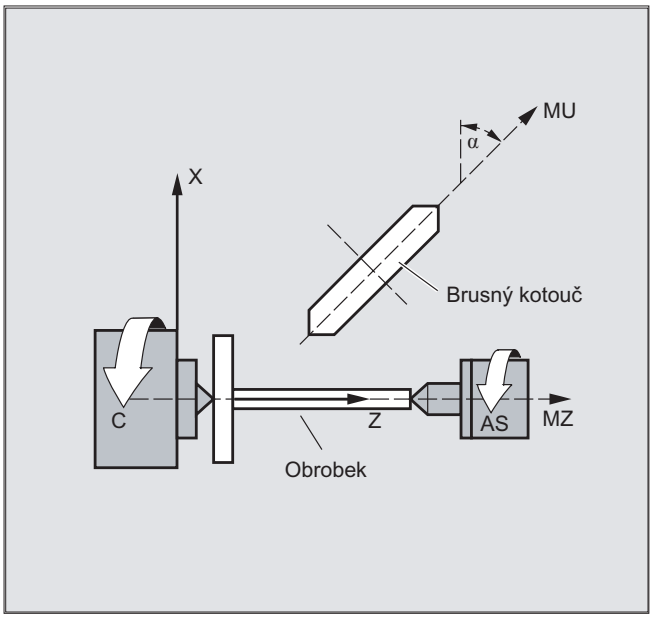
Jestliže je nastaveno TRAFO_TYPE_n = 513, v příkazu OFFN se naprogramuje polovina šířky drážky. Kontura se opracuje s hodnou OFFN - korekce rádiusu nástroje.

6.8.3 Šikmá osa (TRAANG)

Funkce

Funkce Šikmá osa je zamýšlena pro technologii broušení a umožňuje následující výkony:

- Obrábění s šikmou přísuvnou osou
- Pro programování se může používat kartézský souřadný systém.
- Řídící systém transformuje naprogramované pohyby posuvů v kartézském souřadném systému na pohyby posuvů reálných os stroje (standardní případ: šikmá přísuvná osa.



Syntaxe

```
TRAANG(α) nebo TRAANG(α, n)
TRAFOOF
```

Význam

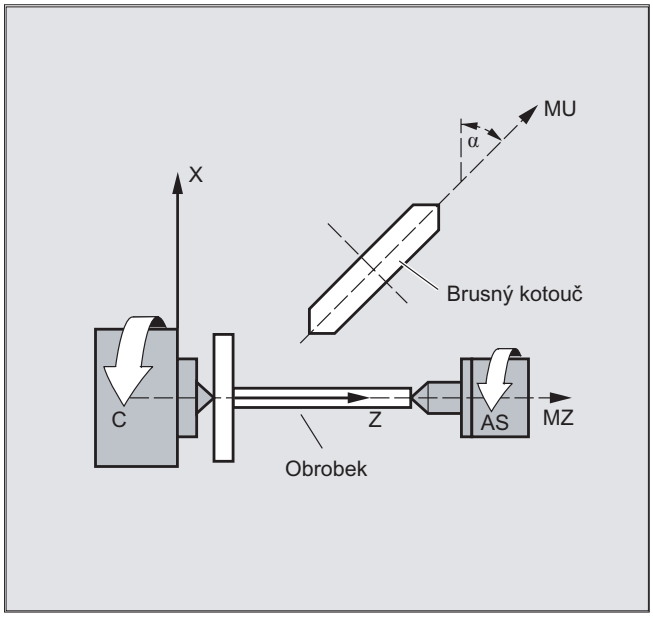
`TRAANG( ) nebo TRAANG( ,n)`	Aktivování transformace se stejným nastavením parametrů, jaké bylo v platnosti u předešlého aktivování.
`TRAANG(α)`	Aktivování první nastavené transformace typu Šikmá osa
`TRAANG(α,n)`	Aktivování n-té nastavené transformace typu Šikmá osa. n smí být maximálně 2. TRAANG(α,1) odpovídá TRAANG(α).
`αA`	Úhel šikmo nastavené osy Přípustné hodnoty pro parametr α jsou: -90 stupňů < α < + 90 stupňů
`TRAFOOF`	Deaktivování transformace
`n`	Počet nastavených transformací

Úhel α je vypuštěn nebo je nulový

Jestliže je úhel α vypuštěn (např. `TRAANG(), TRAANG(, n)`), aktivuje se transformace se stejným nastavením parametrů, jaké bylo v platnosti při předešlém aktivování. Při prvním aktivování se použijí předdefinované hodnoty podle strojních parametrů.

Úhel α = 0 (např. `TRAANG(0), TRAANG(0, n)`) představuje platné nastavení parametrů a neodpovídá již vypuštění parametru, jako tomu bylo u předcházejících verzí.

Příklad



Programový kód	**Komentář**
`N10 G0 G90 Z0 MU=10 G54 F5000 →` `→ G18 G64 T1 D1`	`; Volba nástroje, kompenzace upnutí, Volba roviny`
`N20 TRAANG(45)`	`; Aktivování transformace Šikmá osa`
`N30 G0 Z10 X5`	`; Najíždění na výchozí pozici`
`N40 WAITP(Z)`	`; Uvolnění os pro pohyb tam a zpět`

`N50 OSP[Z]=10 OSP2[Z]=5 OST1[Z]=-2 →`
`→ OST2[Z]=-2 FA[Z]=5000`
`N60 OS[Z]=1`
`N70 POS[X]=4.5 FA[X]=50`
`N80 OS[Z]=0`
	`; Pohyb tam a zpět, dokud není dosaženo rozměru (Pohyb tam a zpět viz kapitola "Kyvný pohyb")`

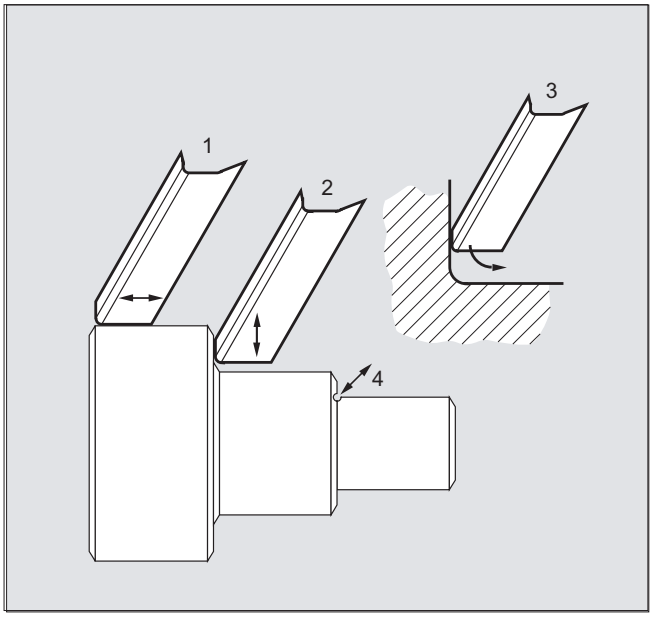
`N90 WAITP(Z)`	`; Přepnutí kyvných os na osy polohovací`
`N100 TRAFOOF`	`; Deaktivování transformace`
`N110 G0 Z10 MU=10`	`; Volné vyjíždění nástroje`
`N120 M30`	`;`

→ Naprogramovat v jednom bloku

Popis

Jsou možné následující druhy obrábění:

1. Podélné broušení
2. Broušení naplocho
3. Broušení určité kontury
4. Šikmé zapichovací broušení



Výrobce stroje

Prostřednictvím strojních parametrů jsou definována následující nastavení:

- Úhel mezi osou stroje a šikmou osou
- Poloha nuly nástroje vztažená na počátek souřadné soustavy, která je pro funkci "Šikmá osa" definována.
- Rezerva rychlosti, která musí zůstat zachována pro paralelní osu kvůli kompenzačním pohybům.
- Rezerva zrychlení, která musí zůstat zachována pro paralelní osu kvůli kompenzačním pohybům.

Konfigurace osy

Aby bylo možné programovat v kartézském souřadném systému, musí být řídícímu systému sdělena souvislost mezi tímto souřadným systémem a ve skutečnosti existujícími osami stroje (MU, MZ).

- Pojmenování geometrických os
- Přiřazení geometrických os kanálovým osám
  - Všeobecný případ (šikmá osa není aktivní)
  - Šikmá osa je aktivní
- Přiřazení kanálových os číslům os stroje
- Označení vřeten
- Přiřazení názvů os stroje

S výjimkou případu "Šikmá osa aktivní" odpovídá tento postup postupu v případě normální konfigurace os.

6.8.4 Programování šikmé osy (G05, G07)

Funkce

V režimu JOG si můžete zvolit, zda chcete, aby se brusný kotouč pohyboval v kartézském systému nebo ve směru šikmé osy (vypisování údajů polohy zůstává kartézské). Pohybuje se pouze reálná osa U, vypisované hodnoty osy Z jsou aktualizovány.

Zpětné najíždění na původní polohu u posunutí REPOS se musí v režimu JOG uskutečňovat kartézsky.

Vyjetí z kartézského ohraničení pracovního pole je v režimu JOG při aktivním příkazu "Posuv PTP" monitorováno, příslušná osa bude zavčasu zabrzděna. Pokud příkaz "Posuv PTP" není aktivní, může osa najet přesně až na ohraničení pracovního pole.

Literatura

/FB2/, Příručka Popis funkcí, Rozšiřovací funkce; Kinematická transformace (M1)

Syntaxe

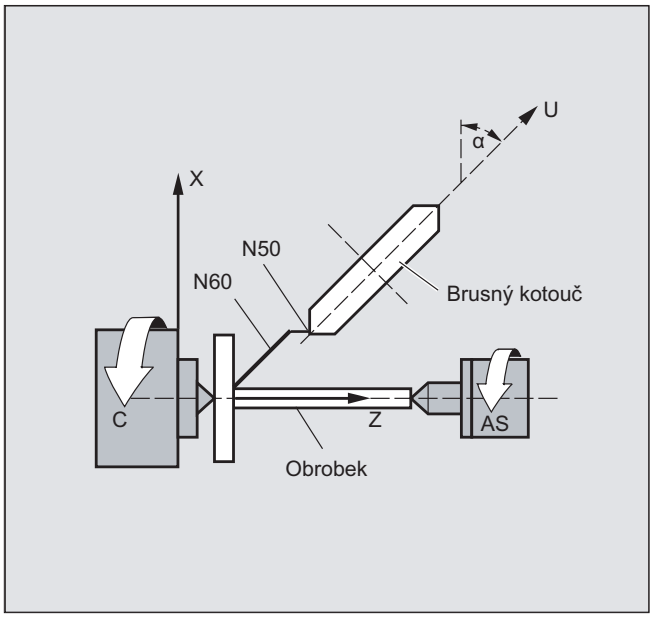
```
G07
G05
```

Příkazy `G05/G07` slouží pro usnadnění programování se šikmou osou. Přitom mohou být programovány a vypisovány údaje poloh v kartézském souřadném systému. Korekční parametry nástroje a posunutí počátku se započítávají kartézsky. Po naprogramování úhlu pro šikmou osu v NC programu je možné najet na počáteční pozici (`G07`) a potom se uskuteční plný šikmý zápich (`G05`).

Význam

G07	Najíždění na počáteční pozici
G05	Aktivuje šikmé zapichování

Příklad



Programování	Komentář
N.. G18	; Programování úhlu pro šikmou osu
N50 G07 X70 Z40 F4000	; Najíždění na počáteční pozici
N60 G05 X70 F100	; Šikmé zapichování
N70 ...	;

6.9 Posuv PTP v kartézských souřadnicích

Funkce

Pomocí této funkce může být naprogramována pozice v kartézském souřadném systému, pohyb stroje se ale uskutečňuje v souřadnicích stroje. Tato funkce může být použita například při změně polohy kloubového mechanismu, jestliže pohyb přitom probíhá přes singularitu.

Poznámka

Funkce má smysl jedině ve spojení s aktivní transformací. Kromě toho je funkce "Posuv PTP" přípustná pouze ve spojení s příkazy G0 a G1.

Syntaxe

```
N... TRAORI
N... STAT='B10' TU='B100' PTP
N... CP
```

Posuv PTP při generické 5/6-osé transformaci

Jestliže je aktivní generická transformace s 5/6 osami a pomocí PTP se uskutečňuje posuv od bodu k bodu v souřadném systému stroje `(ORIMKS)`, potom může být orientace nástroje naprogramována jak pomocí poloh kruhových os

```
N... G1 X Y Z A B C
```

tak také pomocí vektorů Eulerova úhlu, příp. úhlu RPY nezávislých na kinematice

`N... ORIEULER` nebo `ORIRPY`

```
N... G1 X Y Z A2 B2 C2
```

nebo pomocí směrových vektorů

```
N... G1 X Y Z A3 B3 C3
```

Přitom může být aktivní jak interpolace kruhových os, tak také interpolace vektorů s interpolací pomocí největší kružnice koule `ORIVECT` nebo interpolace vektoru orientace na kuželové plášťové ploše `ORICONxx`.

Víceznačnosti orientace pomocí vektorů

Při programování orientace pomocí vektorů existuje u možných poloh kruhových os určitá nejednoznačnost. Polohy kruhových os, na které se najíždí, je přitom možné vybrat naprogramováním příkazu `STAT = <...>`. Jestliže je naprogramováno

`STAT = 0` (což odpovídá standardní poloze,
najíždí se na pozice, které mají nejkratší vzdálenost od počáteční pozice. Pokud je naprogramováno

`STAT = 1`,
najíždí se na pozice, které mají delší vzdálenost od počáteční pozice.

Význam

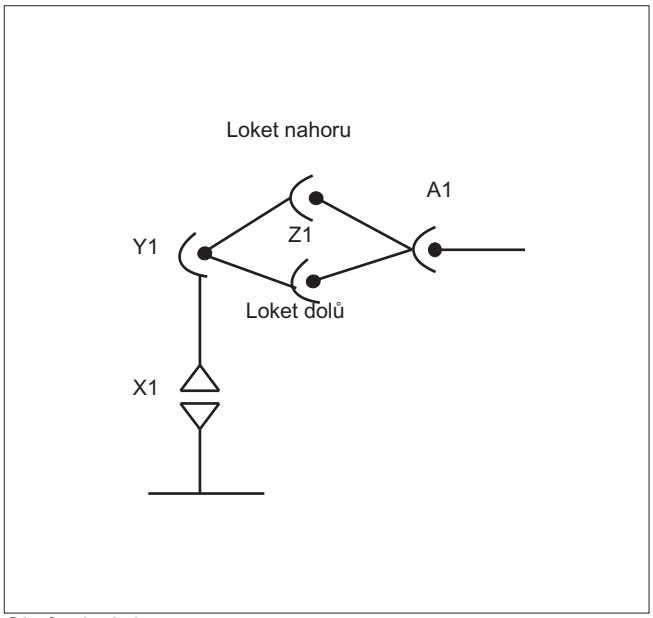
Příkazy `PTP` a `CP` mají modální platnost. `CP` je standardní nastavení.

Zatímco naprogramování hodnoty STAT má modální platnost, má naprogramování TU = <...> platnost blokovou.

Další rozdíl spočívá také v tom, že naprogramování hodnoty STAT se uplatňuje jen při vektorové interpolaci, zatímco programování hodnoty TU se vyhodnocuje i při interpolaci kruhových os.

`PTP`	**P**oint **t**o **P**oint (pohyb od bodu k bodu) Pohyb se uskutečňuje jako synchronní pohyb os, takže nejpomalejší na pohybu se podílející osa, je určující osou pro stanovení rychlosti.
`CP`	**c**ontinuous **p**ath (pohyb po dráze) Pohyb se uskutečňuje jako kartézský pohyb po dráze.
`STAT=`	Nastavení polohy kloubu; hodnota závisí na transformaci.
`TU=`	Informace TURN má blokovou platnost. V důsledku toho je možné najíždět na jednoznačně stanovený úhel v rozsahu od -360 stupňů do +360 stupňů.

Příklad



Obrázek. 6-6

```
N10 G0 X0 Y-30 Z60 A-30 F10000
```
Výchozí nastavení
→ Loket nahoru

```
N20 TRAORI(1)
```
Aktivování transformace

```
N30 X1000 Y0 Z400 A0
```

```
N40 X1000 Z500 A0 STAT='B10'
TU='B100' PTP
```
Změna orientace bez transformace
→ Loket dolů

```
N50 X1200 Z400 CP
N60 X1000 Z500 A20
N70 M30
```

Transformace je znovu aktivní

Příklad - Posuv PTP při generické 5-osé transformaci

Předpoklad: Základem je pravoúhlá kinematika typu CA.

Programový kód	**Komentář**
TRAORI	; Aktivování transformace kinematiky typu CA
PTP	; Aktivování posuvu PTP
N10 A3 = 0 B3 = 0 C3 = 1	; Polohy kruhových os C = 0 A = 0
N20 A3 = 1 B3 = 0 C3 = 1	; Polohy kruhových os C = 90 A = 45
N30 A3 = 1 B3 = 0 C3 = 0	; Polohy kruhových os C = 90 A = 90
N40 A3 = 1 B3 = 0 C3 = 1 STAT = 1	; Polohy kruhových os C = 270 A = -45

Volba jednoznačně určené polohy kruhových os pro najíždění:

V bloku N40 přitom v důsledku programového příkazu `STAT = 1` najíždějí kruhové osy ze svého počátečního bodu (C=90, A=90) do koncového bodu (C=270, A=–45) po delší dráze, místo toho, jak by tomu bylo v případě nejkratší dráhy do koncového bodu (C=90, A=45) při `STAT = 0`.

Popis

Přepínání mezi pohyby v kartézském systému a pohyby os stroje se uskutečňuje pomocí příkazů `PTP` a `CP`.

Posuv PTP při generické 5/6-osé transformaci

Při posuvu PTP nezůstává narozdíl od transformace s 5/6 osami obecně na jednom místě, pokud se mění pouze orientace. Všemi transformačními osami (3 lineární osy a až 3 kruhové osy) se najíždí lineárně na transformované koncové pozice, aniž by přitom ještě byla transformace ve vlastním smyslu slova aktivní.

Posuv PTP se vypíná naprogramováním G-kódu `CP` s modální platností.

O rozmanitých transformacích je pojednáno v dokumentu:
/FB3/, Příručka Popis funkcí, Speciální funkce, Modul pro práci s transformacemi (TE4).

Programování polohy (STAT=)

Samotná poloha stroje není prostřednictvím údajů polohy s kartézskými souřadnicemi a orientací nástroje jednoznačně určena. V závislosti na tom, o jakou kinematiku se jedná, existuje až 8 rozdílných, příp. odlišujících se poloh kloubů. Tyto polohy jsou potom závislé na transformaci. Aby bylo možné kartézskou polohu jednoznačně přepočítat do úhlů os, musí být poloha kloubu zadána příkazem `STAT=`. Příkaz "`STAT`" obsahuje jako binární hodnotu pro každou z možných poloh jeden bit.

Informace o bitech poloh, které je potřeba v příkazu "STAT" naprogramovat, viz:
/FB2/, Příručka Popis funkcí, Rozšiřovací funkce; Kinematická transformace (M1), kapitola "Posuv PTP v kartézských souřadnicích".

Programování úhlu osy (TU=)

Aby bylo možné jednoznačně najet na úhel osy < ± 360 stupňů, musí být tato informace naprogramována pomocí příkazu "TU".

Osy se pohybují po nejkratší dráze:

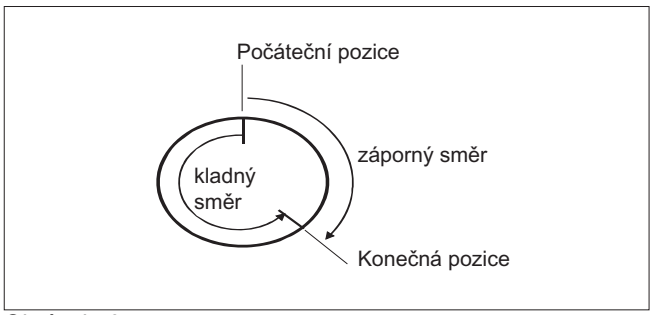
- pokud u dané pozice žádná hodnota TU není naprogramována
- u os, jejichž rozsah posuvu je > ± 360 stupňů

Příklad:

Na cílovou pozici uvedenou na obrázku je možné najet v záporném nebo v kladném směru. Směr je naprogramován v adrese A1.

A1=225°, TU=Bit 0, → kladný směr

A1= − 135°, TU=Bit 1, → záporný směr



Obrázek. 6-7

Příklad vyhodnocování TU pro případ generické transformace s 5/6 osami a cílové pozice

Proměnná TU obsahuje pro každou osu, která vstupuje do transformace, jeden bit, který ukazuje směr pohybu. Přiřazení bitů v TU odpovídá uspořádání kruhových os v kanálu. Informace v TU mohou být vyhodnocovány pouze u až 3 možných kruhových os, které vstupují do transformace:

bit 0: osa 1, bit TU = 0 : 0 stupňů <= úhel kruhové osy < 360 stupňů

bit 1: osa 2, bit TU = 1 : –360 stupňů < úhel kruhové osy < 0 stupňů

Počáteční poloha kruhové osy je C % 0, při zadání programového příkazu C = 270 najíždí kruhová osa na následující cílové pozice:

C = 270: bit TU 0, kladný směr otáčení

C = –90: bit TU 1, záporný směr otáčení

Další chování

Změna provozního režimu

Funkce "Posuv PTP v kartézských souřadnicích" má smysl jen v provozních režimech AUTO a MDA. Při přepnutí do provozního režimu JOG zůstává aktuální nastavení zachováno.

Jestliže je G-kód `PTP` nastaven, uskutečňují se pohyby osami v MCS. Jestliže je G-kód `PTP` nastaven, uskutečňují se pohyby osami v MCS.

Power On/RESET

Po zapnutí systému nebo po resetu je nastavení závislé na strojním parametru `$MC_GCODE_REST_VALUES[48]`. Standardně je nastaven druh posuvu "`CP`".

REPOS

Jestliže se v průběhu bloku přerušení nastavovala poloha pomocí funkce "Posuv PTP v kartézských souřadnicích", bude se `PTP` uplatňovat i při zpětném pohybu.

Superponované pohyby

Posunutí DRF nebo externí posunutí jsou při posuvech PTP v kartézských souřadnicích jen v omezené míře. Při přepnutí z pohybu PTP do pohybu CP se v BCS nesmí vyskytovat žádné superponované pohyby.

Přechodová zaoblení mezi pohyby CP a PTP

Mezi bloky s příkazem `G641` je možné vkládat programovatelná přechodová zaoblení.

Velikost oblasti přechodového zaoblení odpovídá úseku dráhy v mm nebo v palcích, od kterého, příp. po který bude přechod mezi bloky zaoblen. Tuto velikost je možné zadat následujícím způsobem:

- pro bloky s příkazem G0 pomocí příkazu `ADISPOS`
- pro všechny ostatní příkazy dráhy pomocí příkazu `ADIS`

Výpočet pohybu po dráze odpovídá zohlednění F-adres u bloků bez příkazu G0. Posuv zůstane zachován pole os uvedených ve skupině `FGROUP(...)`.

Výpočet hodnoty posuvu

Pro bloky s CP se používají pro výpočet kartézské osy základního souřadného systému.

Pro bloky s PTP se pro výpočet používají odpovídající osy souřadného systému stroje.

6.9.1 PTP u příkazu TRANSMIT

Funkce

Pomocí PTP s funkcí TRANSMIT je možné zajistit, aby bloky s posuvy G0 a G1 byly zpracovávány časově optimálním způsobem. Místo aby se lineárně pohybovaly osy základního souřadného systému (CP), pohybují se lineárně osy stroje (PTP). V blízkosti pólu má tento způsob pohybu os za následek, že koncového bodu bloku je možno dosáhnout výrazně rychleji.

Výrobní program se i nadále zapisuje v kartézském souřadném systému obrobku a všechna posunutí souřadnic, otočení a programové příkazy framů zůstávají v platnosti. Simulace v HMI se rovněž zobrazuje v kartézském souřadném systému obrobku.

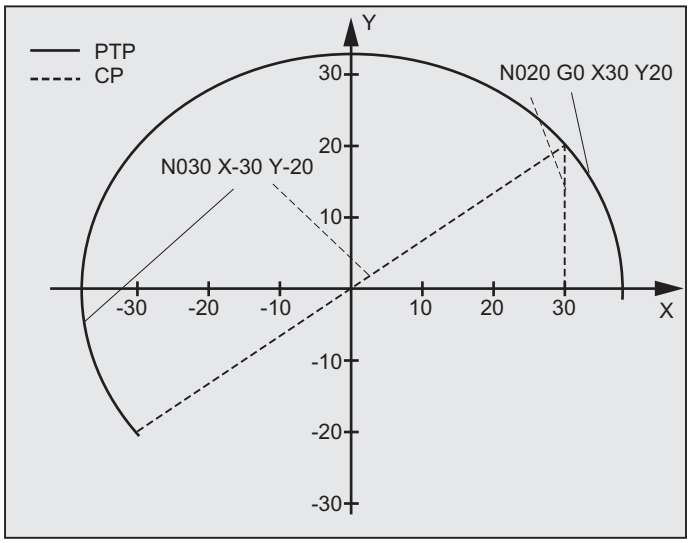
Syntaxe

```
N... TRANSMIT
N... PTPG0
N... G0 ...
...
N... G1 ...
```

Význam

`TRANSMIT`	Aktivování první deklarované funkce TRANSMIT (viz kapitola "Frézovací práce na rotačních součástech: TRANSMIT")
`PTPG0`	**P**oint to **P**oint **G0** (pohyb od bodu k bodu automaticky po každém bloku s G0 a potom opětovné aktivování CP) Protože příkazy STAT a TU mají modální platnost, platí vždy naposled naprogramovaná hodnota.
`PTP`	**P**oint to **P**oint (pohyb od bodu k bodu) Pro funkci TRANSMIT PTP znamená, že v kartézském systému najíždí buď okolo pólu nebo se odjíždí od pólu po archimédovských spirálách. Z toho vyplývající pohyby nástroje mají výrazně jiný průběh než v případě funkce CP a jsou ukázány v příslušných příkladech programování.
`STAT=`	Odstranění víceznačností týkajících se pólu.
`TU=`	TU v případě PTP s funkcí TRANSMIT nemá žádný význam.

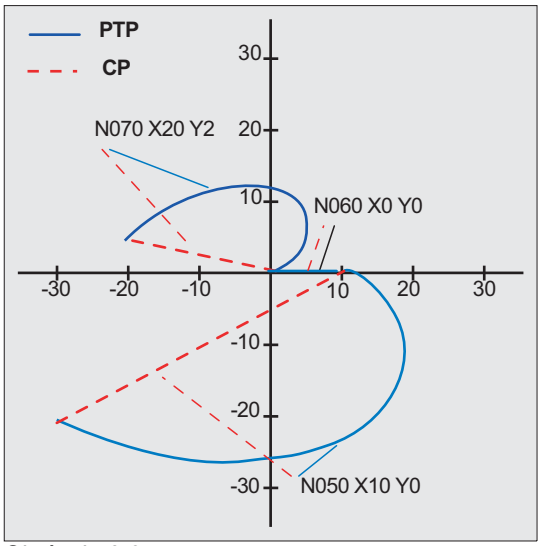
Příklad - Objíždění pólu pomocí PTP a funkce TRANSMIT



Obrázek. 6-8

Programový kód	**Komentář**
N001 G0 X30 Z0 F10000 T1 D1 G90	; Výchozí pozice v absolutních rozměrech
N002 SPOS=0	
N003 TRANSMIT	; Transformace TRANSMIT
N010 PTPG0	; Ke každému bloku s G0 automaticky PTP a potom znovu CP
N020 G0 X30 Y20	
N030 X-30 Y-20	
N120 G1 X30 Y20	
N110 X30 Y0	
M30	

Příklad - Odjíždění od pólu pomocí PTP a funkce TRANSMIT



Obrázek. 6-9

Programování	**Komentář**
`N001 G0 X90 Z0 F10000 T1 D1 G90`	`; Výchozí nastavení`
`N002 SPOS=0`	
`N003 TRANSMIT`	`; Transformace TRANSMIT`
`N010 PTPG0`	`; Ke každému bloku s G0 automaticky PTP a potom znovu CP`
`N020 G0 X90 Y60`	
`N030 X-90 Y-60`	
`N040 X-30 Y-20`	
`N050 X10 Y0`	
`N060 X0 Y0`	
`N070 X-20 Y2`	
`N170 G1 X0 Y0`	
`N160 X10 Y0`	
`N150 X-30 Y-20`	
`M30`	

Popis

PTP a PTPG0

Příkaz `PTPG0` se uplatňuje u všech transformací, které mohou s funkcí `PTP` pracovat. Ve všech ostatních případech nemá příkaz `PTPG0` žádnou funkci.

Bloky s G0 jsou zpracovávány v režimu CP.

Aktivování funkce `PTP`, příp. `PTPG0` se uskutečňuje ve výrobním programu nebo deaktivováním funkce `CP` ve strojním parametru `$MC_GCODE_RESET_VALUES[48]`.

⚠ POZOR

Okrajové podmínky

Pokud jde o pohyby nástroje a kolize, platí větší počet okrajových podmínek a určité funkce se vzájemně vylučují:

V režimu `PTP` nesmí být aktivní žádné korekce rádiusu nástroje.

Když je aktivní funkce `PTPG0` a také korekce rádiusu nástroje, pohyby se uskutečňují s funkcí `CP`.

Měkké najíždění a odjíždění (WAB) není s `PTP` možné.

Když je aktivní funkce `PTPG0` a také měkké najíždění a odjíždění (WAB), pohyby se uskutečňují s funkcí `CP`.

Když je aktivní `PTP`, je zpracování cyklů oddělování třísky (`CONTPRON`, `CONTDCON`) vyloučeno.

Když je aktivní `PTPG0`, probíhá zpracování cyklů oddělování třísky (`CONTPRON`, `CONTDCON`) s `CP`.

Fasety (`CHF`, `CHR`) a zaoblení (`RND`, `RNDM`) jsou ignorovány.

Kompresor není s `PTP` kompatibilní a v blocích s PTP se automaticky deaktivuje.

Superponované pohyby os při interpolaci se v průběhu úseků s PTP nesmí měnit.

V případě `G643` se automaticky přepíná na přechodová zaoblení s axiálním přesným najetím `G642`.

Když je aktivní PTP, nemohou být osy transformace současně polohovacími osami.

Literatura:

/FB2/, Příručka Popis funkcí, Rozšiřovací funkce; Kinematická transformace (M1), kapitola "Posuv PTP v kartézských souřadnicích"

PTP u příkazu TRACON:

`PTP` se může využívat také spolu s funkcí `TRACON`, pokud první zřetězená transformace `PTP` podporuje.

Význam příkazů STAT= a TU= u funkce TRANSMIT

Jestliže má být kruhová osa otočena o 180 stupňů, příp. pokud má kontura procházet pólem, když je aktivní CP, mohou být kruhové osy v závislosti na strojním parametru `$MC_TRANSMIT_POLE_SIDE_FIX_1/2 [48]` otáčeny o -/+ 180 stupňů ve směru nebo proti směru hodinových ručiček. Rovněž může být nastaveno, zda se má projíždět pólem nebo zda se má okolo pólu otáčet.

6.10 Okrajové podmínky při aktivování transformace

Funkce

Aktivování transformací je možné pomocí výrobního programu, příp. MDA. Přitom je potřeba mít na paměti následující:

- Pomocný pohybový blok se nevkládá (fasety/rádiusy).
- Posloupnost splinových bloků musí být uzavřena; pokud není, objeví se hlášení.
- Jemná korekce nástroje musí být deaktivována (FTOCOF); pokud není, objeví se hlášení.
- Korekce rádiusu nástroje musí být deaktivována (G40); pokud není, objeví se hlášení.
- Aktivovaná korekce délky nástroje je řídícím systémem přenášena do transformace.
- Aktuální frame, který byl před transformací aktivní, bude řídícím systémem deaktivován.
- Aktivní ohraničení pracovního pole jsou pro osy podílející se na transformaci řídícím systémem deaktivována (odpovídá příkazu WALOMOF).
- Monitorování chráněných oblastí je deaktivováno.
- Režim řízení pohybu po dráze a přechodová zaoblení se přeruší.
- Všechny osy uvedené ve strojním parametru musí být vzhledem k danému bloku synchronizovány.
- Vyměněné osy musí být vyměněny zpátky, pokud tomu tak není, objeví se hlášení.
- V případě závislých os se objeví hlášení.

Výměna nástroje

Výměna nástroje je přípustná, jen když je korekce rádiusu nástroje deaktivována.

Změna korekce délky nástroje a aktivování/deaktivování korekce rádiusu nástroje nesmí být naprogramovány ve stejném bloku.

Změna framu

Všechny příkazy, které se vztahují jen na základní souřadný systém, jsou přípustné (frame, korekce rádiusu nástroje). Změna framu, když je aktivní funkce G91 (inkrementální rozměry) ale není - narozdíl od situace, kdy transformace není aktivní - nijak zvlášť ošetřena. Inkrement, který se má urazit, se vyhodnocuje v souřadném systému obrobku nového framu - nezávisle na tom, který frame se uplatňoval v předcházejícím bloku.

Výjimky:

Osy, které se podílejí na transformaci, se nemohou používat:

- jako osy s příkazem PRESET (alarm)
- pro najíždění na pevný bod (alarm)
- pro najíždění na referenční bod (alarm)

6.11 Deaktivování transformace (TRAFOOF)

Funkce

Pomocí příkazu `TRAFOOF` jsou všechny aktivní transformace a framy deaktivovány.

Poznámka

Framy, které jsou potom zapotřební, musí být novým naprogramováním přepnuty do aktivního stavu.

Přitom je potřeba mít na paměti následující:

Pro deaktivování transformace platí stejné okrajové podmínky jako pro její aktivování (viz kapitola "Okrajové podmínky při aktivování transformace").

Syntaxe

```
TRAFOOF
```

Význam

`TRAFOOF`	Příkaz pro vypnutí všech aktivních transformací/framů

6.12 Zřetězené transformace (TRACON, TRAFOOF)

Funkce

Mohou být aktivovány (zřetězeny) vždy **dvě** transformace po sobě, takže pohybové složky pro osy z první transformace se stávají vstupními daty pro zřetězenou druhou transformaci. Pohybové složky ze druhé transformace se uplatňují na osy stroje.

Řetězec smí obsahovat **dvě** transformace.

Poznámka

Nástroj je přiřazen vždy první transformaci v řetězci. Následující transformace se potom chová tak, jako kdyby délka aktivního nástroje byla nulová. Pro první transformaci v řetězci jsou uplatňovány pouze základní délky nástroje (_BASE_TOOL_) nastavené pomocí strojních parametrů.

Výrobce stroje

Věnujte prosím pozornost upozorněním od výrobce stroje, které se týkají případných transformací předem definovaných pomocí strojních parametrů.

Transformace a zřetězené transformace jsou volitelnými doplňky. Informace o tom, zda určité transformace v řetězci mohou být zpracovávány v určitých řídících systémech, naleznete v aktuálním katalogu.

Použití

- Broušení kontur, které byly naprogramovány jako křivka na rozvinutém plášti válce (TRACYL) pomocí šikmo uloženého brusného kotouče, např. broušení nástrojů.
- Jemné opracování nezaoblené kontury vyrobené pomocí funkce TRANSMIT pomocí šikmo uloženého brusného kotouče.

Syntaxe

`TRACON(trf,par)`	Zřetězená transformace se aktivuje.
`TRAFOOF`	

Význam

`TRACON`	Zřetězená transformace se aktivuje. Jiná dříve aktivovaná transformace se příkazem TRACON() implicitně zruší.
`TRAFOOF`	Dříve aktivovaná (zřetězená) transformace se vypne.

`trf`	Číslo zřetězené transformace: 0 nebo 1 pro první/jedinou zřetězenou transformaci. Pokud toto místo není naprogramováno, má příkaz stejný význam, jak kdyby byla zadána hodnota 0 nebo 1, tzn. bude aktivována první/jediná transformace. 2 pro druhou zřetězenou transformaci. (Jiné hodnoty než 0 - 2 budou mít za následek alarm.)
`par`	Jeden nebo více parametrů oddělených čárkami pro ty transformace ve zřetězení, které parametry očekávají, např. úhel šikmé osy. Jestliže tyto parametry nejsou nastaveny, uplatní se předdefinovaná nastavení nebo naposled použité parametry. Jestliže mají být pro předtím uvedené parametry uplatněna nastavení předvoleb, musí být nastavením čárek zajištěno, aby byly zadané parametry vyhodnocovány v posloupnosti, ve které jsou očekávány. Zejména v případě zadání nejméně jednoho parametru musí být před tímto parametrem uvedena čárka, a to i když zadání parametru trf není zapotřebí, například (TRACON(, 3.7).

Předpoklady

Druhou transformací musí být transformace **"Šikmá osa"** (`TRAANG`). Jako první transformaci je možno zadat následující:

- Transformace orientace (`TRAORI`), včetně kardanové frézovací hlavičky
- `TRANSMIT`
- `TRACYL`
- `TRAANG`

Aby se mohly používat aktivační příkazy pro zřetězené transformace, musí být splněna předpoklad, že jednotlivé zřetězené transformace a zřetězená transformace, která má být aktivována, jsou definovány pomocí strojních parametrů.

Při využívání v rámci zřetězení je nutno mít na zřeteli také okrajové podmínky uvedené v jednotlivých popisech pro uvedené transformace a speciální případy.

Pokud budete potřebovat informace týkající se konfigurace strojních parametrů transformací, viz:

/FB2/, Příručka Popis funkcí, Rozšiřovací funkce; Kinematické transformace (M1) a

/FB3/, Příručka Popis funkcí, Speciální funkce; 3- až 5-osé transformace (F2).

Korekční parametry nástroje 7

7.1 Paměť korekcí

Funkce

Struktura paměti korekčních parametrů

Každé datové pole může být vyvoláno pomocí T- a D-čísel (s výjimkou "prostého D-čísla") a tato pole obsahují vedle geometrických údajů nástroje ještě i další záznamy, např. typ nástroje.

Struktura prostých D-čísel

"Struktura prostých D-čísel" se používá tehdy, pokud je správa nástrojů realizována mimo NCK. V tomto případě jsou vytvářena D-čísla s odpovídajícími bloky korekčních parametrů bez přiřazení určitému nástroji.

Ve výrobním programu může být naprogramováno T-slovo, to však nemá žádný vztah k naprogramovanému D-číslu.

Uživatelské parametry břitu

Uživatelské parametry břitu mohou být konfigurovány pomocí strojního parametru. Věnujte prosím pozornost informacím od výrobce stroje.

Parametry nástroje

Poznámka

Jednotlivé hodnoty v paměti korekčních parametrů

Jednotlivé hodnoty v paměti korekčních parametrů P1 až P25 je možné číst a zapisovat z programu pomocí systémových proměnných. Všechny zbývající parametry jsou rezervovány.

Korekční parametry nástroje $TC_DP6 až $TC_DP8, $TC_DP10 a $TC_DP11, jakož i $TC_DP15 až $TC_DP17, $TC_DP19 a $TC_DP20 mají v závislosti na typu nástroje odlišný význam.

[1]Platí i u frézovacích nástrojů pro 3D frézování na čelní ploše

[2]U nástroje typu drážková pila

[3]rezervováno: Systémem SINUMERIK 840D není využito

Číslo parametrů nástroje (DP)	Význam systémových proměnných	Poznámka
$TC_DP1	Typ nástroje	Přehled viz seznam
$TC_DP2	Poloha břitu	Jen pro soustružnické nástroje
Geometrie	**Korekce délky**	
$TC_DP3	Délka 1	Započítává se podle
$TC_DP4	Délka 2	Typu a roviny
$TC_DP5	Délka 3	

Číslo parametrů nástroje (DP)	Význam systémových proměnných	Poznámka
Geometrie	**Rádius**	
$TC_DP6[1] $TC_DP6[2]	Rádius 1 / délka 1 Průměr d	Frézovací/soustružnické/brusné nástroje Drážková pila
$TC_DP7[1] $TC_DP7[2]	Délka 2 / Rohový rádius kuželové frézy Šířka drážky b, rohový rádius	Frézovací nástroje Drážková pila
$TC_DP8[1] $TC_DP8[2]	Rádius zaoblení 1 pro frézovací nástroje Přesah k	Frézovací nástroje Drážková pila
$TC_DP9[1,3]	Rádius zaoblení 2	rezervováno
$TC_DP10[1]	Úhel 1 čelní strany nástroje	Kuželové frézovací nástroje
$TC_DP11[1]	Úhel 2 podélné osy nástroje	Kuželové frézovací nástroje
Opotřebení	**Korekce délky a rádiusu**	
$TC_DP12	Délka 1	
$TC_DP13	Délka 2	
$TC_DP14	Délka 3	
$TC_DP15[1] $TC_DP15[2]	Rádius 1 / délka 1 Průměr d	Frézovací/soustružnické/brusné nástroje Drážková pila
$TC_DP16[1] $TC_DP16[3]	Délka 2 / Rohový rádius kuželové frézy, šířka drážky b, rohový rádius	Frézovací nástroje Drážková pila
$TC_DP17[1] $TC_DP17[2]	Rádius zaoblení 1 pro frézovací nástroje Přesah k	Frézování / 3D frézování na čelní ploše Drážková pila
$TC_DP18[1,3]	Rádius zaoblení 2	rezervováno
$TC_DP19[1]	Úhel 1 čelní strany nástroje	Kuželové frézovací nástroje
$TC_DP20[1]	Úhel 2 podélné osy nástroje	Kuželové frézovací nástroje
Základní rozměr / adaptér	**Korekce délky**	
$TC_DP21	Délka 1	
$TC_DP22	Délka 2	
$TC_DP23	Délka 3	
Technologie		
$TC_DP24	Úhel volného řezání	Jen pro soustružnické nástroje
$TC_DP25		rezervováno

Poznámky

Pro geometrické veličiny (např. Délka 1 nebo Rádius) existuje několik vstupních komponent. Tyto komponenty se aditivně započítávají k výsledné veličině (např. celková délka 1, celkový rádius), která se potom uplatňuje při obrábění.

Do nepotřebných parametrů nástroje je nutno dosadit nulu.

Parametry nástroje $TC-DP1 až $TC-DP23 u nástrojů pro obrábění kontur

Poznámka

Parametry nástroje, které v tabulce nejsou uvedeny, jako např. $TC_DP7, nejsou vyhodnocovány, tzn. jejich obsah nemá žádný význam.

Číslo parametrů nástroje (DP)	Význam	Břity Dn		Poznámka
$TC_DP1	Typ nástroje			400 až 599
$TC_DP2	Poloha břitu			
Geometrie	**Korekce délky**			
$TC_DP3	Délka 1			
$TC_DP4	Délka 2			
$TC_DP5	Délka 3			
Geometrie	**Rádius**			
$TC_DP6	Rádius			
Geometrie	**Mezní úhel**			
$TC_DP10	Minimální mezní úhel			
$TC_DP11	maximální mezní úhel			
Opotřebení	**Korekce délky a rádiusu**			
$TC_DP12	Opotřebení – délka 1			
$TC_DP13	Opotřebení – délka 2			
$TC_DP14	Opotřebení – délka 3			
$TC_DP15	Rádius opotřebení			
Opotřebení	**Mezní úhel**			
$TC_DP19	Opotřebení - minimální mezní úhel			
$TC_DP20	Opotřebení - maximální mezní úhel			
Základní rozměr / adaptér	**Korekce délky**			
$TC_DP21	Délka 1			
$TC_DP22	Délka 2			
$TC_DP23	Délka 3			

Základní hodnota a hodnota opotřebení

Výsledné veličiny jsou vždy vypočítávány jako součet základní hodnoty a hodnoty opotřebení (např. $TC_DP6 + $TC_DP15 pro rádius). K délce prvního břitu nástroje se kromě toho přičítá ještě základní rozměr ($TC_DP21 – $TC_DP23). Kromě toho se u těchto délek nástroje uplatňují ještě i další veličiny, které i v případě běžných nástrojů mohou ovlivňovat jejich efektivní délku (adaptér, orientovatelný držák nástroje, nastavované parametry).

Mezní úhel 1 a 2

Mezní úhel 1, příp. mezní úhel 2, se vztahují vždy na vektor od středu břitu ke vztažnému bodu břitu a měří se proti směru hodinových ručiček.

7.2 Aditivní korekce

7.2.1 Aktivování aditivních korekcí (DL)

Funkce

Na aditivní korekce je možné pohlížet jako na procesní korekce programovatelné v průběhu obrábění. Vztahují se na geometrické údaje o břitu a tvoří tedy součást údajů o břitech nástroje.

Údaje aditivních korekcí jsou adresována pomocí DL-čísla (DL: Location Dependent; korekce vztažená na určité místo použití) a zadávají se prostřednictvím uživatelského rozhraní.

Aplikace

Prostřednictvím aditivních korekcí mohou být kompenzovány chyby rozměrů podmíněné místem použití.

Syntaxe

```
DL=<číslo>
```

Význam

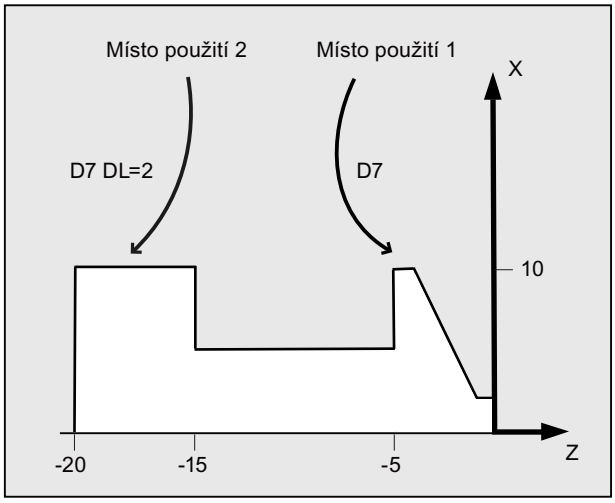
`DL`	Příkaz pro aktivování aditivní korekce
`<číslo>`	Prostřednictvím parametru `<číslo>` se zadává blok aditivních korekčních parametrů nástroje, který má být aktivován.

Poznámka

Definice počtu a aktivování aditivních korekcí se uskutečňuje prostřednictvím strojních parametrů (→ Věnujte prosím pozornost informacím od výrobce stroje!).

Příklad

Pro 2 ložisková sedla se použije stejný břit:



Programový kód	**Komentář**
N110 T7 D7	; Revolver se nastaví na místo č. 7. Aktivují se D7 a DL=1 a v následujícím bloku se použijí.
N120 G0 X10 Z1	
N130 G1 Z-6	
N140 G0 DL=2 Z-14	; Aditivně k D7 se aktivuje DL=2 a v následujícím bloku se použijí.
N150 G1 Z-21	
N160 G0 X200 Z200	; Najíždění na bod pro výměnu nástroje.
...	

7.2.2 Definice opotřebení a seřizovacích hodnot ($TC_SCPxy[t,d], $TC_ECPxy[t,d])

Funkce

Hodnoty opotřebení a seřizovacích parametrů mohou být načítány a zapisovány pomocí systémových proměnných. Logika je přitom orientována na logiku odpovídajících systémových proměnných pro nástroje a břity.

Systémové proměnné

Systémové proměnné	Význam
$TC_SCPxy[<t>,<d>]	Hodnoty opotřebení, které jsou prostřednictvím xy přiřazeny příslušnému geometrickému parametru, přičemž x je číslo hodnoty opotřebení a y odkaz na geometrický parametr
$TC_ECPxy[<t>,<d>]	Seřizovací hodnoty, které jsou prostřednictvím xy přiřazeny příslušnému geometrickému parametru, přičemž x je číslo hodnoty seřizovací a y odkaz na geometrický parametr
<t>: T-číslo nástroje <d>: D-číslo břitu nástroje	

Poznámka

Takto definované hodnoty opotřebení a seřizovací hodnoty s přičítají ke geometrickým parametrům a ostatním korekčním parametrům.

Příklad

Hodnota opotřebení pro Délku 1 pro břit <d> nástroje (t) je stanovena na hodnotu 1,0.

Parametr: $TC_DP**3** (délka 1, soustružnický nástroj)

Hodnoty opotřebení: $TC_SCP1**3** až $TC_SCP6**3**

Seřizovací hodnoty: $TC_ECP1**3** až $TC_ECP6**3**

$TC_SCP4**3** [<t>,<d>] = 1.0

7.2.3 Vymazání aditivních korekcí (DELDL)

Funkce

Příkazem `DELDL` se provádí mazání aditivních korekcí břitů nástroje (vymazání z paměti). Přitom se vymažou jak definované hodnoty opotřebení, tak i seřizovací hodnoty.

Syntaxe

```
DELDL[<t>,<d>]
DELDL[<t>]
DELDL
<Status>=DELDL[<t>,<d>]
```

Význam

`DELDL`	Příkaz pro vymazání aditivních korekcí	
`<t>`	T-číslo nástroje	
`<d>`	D-číslo břitu nástroje	
`DELDL[<t>,<d>]`	Vymažou se všechny aditivní korekce břitu `<d>` nástroje `<t>`.	
`DELDL[<t>]`	Vymažou se všechny aditivní korekce všech břitů nástroje `<t>`	
`DELDL`	Budou vymazány všechny aditivní korekce všech břitů všech nástrojů jednotky TO (pro kanál, ve kterém je příkaz naprogramován).	
`<Status>`	Status vymazání	
	Hodnota:	Význam:
	0	Vymazání bylo úspěšně provedeno.
	-	Vymazání nebylo provedeno (pokud nastavení parametrů označuje přesně jeden břit) nebo vymazání nebylo provedeno úplně (pokud nastavení parametrů popisuje více břitů).

Poznámka

Hodnoty opotřebení a seřizovací hodnoty aktivního nástroje není možné vymazat (analogicky k chování při mazání `D`-čísel a parametrů nástroje).

7.3 Korekce nástroje - speciální zacházení

Funkce

Pomocí nastavovaných parametrů SD 42900 až SD 42960 je možné ovládat vyhodnocování znaménka pro délku a opotřebení nástroje.

To platí jednak pro chování složek opotřebení při zrcadlovém převracení geometrických os, jednak při změně roviny obrábění, ale také pro teplotní kompenzaci ve směru nástroje.

Hodnoty opotřebení

Jestliže jsou v následujících odstavcích zmiňovány hodnoty opotřebení, je tím v každém případě míněn součet vlastních hodnot opotřebení ($TC_DP12 až $TC_DP20) a součtových korekcí s hodnotami opotřebení ($SCPX3 až $SCPX11) a seřizovacími hodnotami ($ECPX3 až $ECPX11).

Pokud budete potřebovat bližší informace o součtových korekčních parametrech, viz:
Literatura:
Příručka Popis funkcí, Správa nástrojů

Nastavované parametry

Nastavovaný parametr	Význam
SD42900 $SC_MIRROR_TOOL_LENGTH	Zrcadlové převracení složek délky nástroje a složek základních rozměrů.
SD42910 $SC_MIRROR_TOOL_WEAR	Zrcadlové převrácení hodnot opotřebení složky délky nástroje.
SD42920 $SC_WEAR_SIGN_CUTPOS	Vyhodnocování znaménka složky opotřebení v závislosti na poloze břitu.
SD42930 $SC_WEAR_SIGN	Invertování znaménka rozměru opotřebení.
SD42935 $SC_WEAR_TRANSFORM	Transformace hodnot opotřebení.
SD42940 $SC_TOOL_LENGTH_CONST	Přiřazení složky délky nástroje geometrické ose.
SD42950 $SC_TOOL_LENGTH_TYPE	Přiřazení složky délky nástroje nezávisle na typu nástroje.
SD42960 $SC_TOOL_TEMP_COMP	Hodnota teplotní kompenzace ve směru nástroje. Funguje, i když je programována orientace nástroje.

Literatura

Příručka Popis funkcí, Základní funkce; "Korekční parametry nástrojů (W1)

Další informace

Aktivování změněných nastavovaných parametrů

Nové vyhodnocení složek nástroje při změně popisovaných nastavovaných parametrů se provádí až tehdy, když je následně aktivován břit nástroje. Pokud je nějaký nástroj už aktivní a vyhodnocování změněných údajů tohoto nástroje se má stát platným, je nutné tento nástroj znovu vyvolat.

Analogický postup se použije v případě, kdy se změní výsledná délka nástroje, protože se změnil stav zrcadlového převrácení osy. Po příkazu zrcadlového převrácení musí být nástroj znovu aktivován, aby se uplatnily změněné složky délky nástroje.

Orientovatelný držák nástroje a nové nastavované parametry

Nastavované parametry SD 42900 až SD 42940 nepůsobí na složky případného aktivního orientovatelného držáku nástroje. Avšak výpočet s orientovatelným držákem nástroje vždy umožňuje pracovat s nástrojem v celé jeho výsledné délce (délka nástroje + opotřebení + rozměr základny nástroje). Při výpočtu výsledné celkové délky se zohledňují všechny změny, jež byly způsobeny nastavovanými parametry; tzn. vektory orientovatelného držáku nástroje jsou nezávislé na rovině opracování.

Poznámka

Při použití orientovatelného držáku nástroje je často užitečné definovat všechny nástroje pro základní systém bez zrcadlového převrácení a pak také ty, které se používají jenom při obrábění v zrcadlovém převrácení. Při obrábění se zrcadlově převrácenými osami se pak držák nástroje otočí tak, aby skutečná poloha nástroje byla popisována správně. Všechny složky délky nástroje jsou potom automaticky přiřazeny správným směrům, takže řídící systém si může ušetřit vyhodnocování jednotlivých složek pomocí nastavovaných parametrů v závislosti na stavu zrcadlového převrácení jednotlivých os.

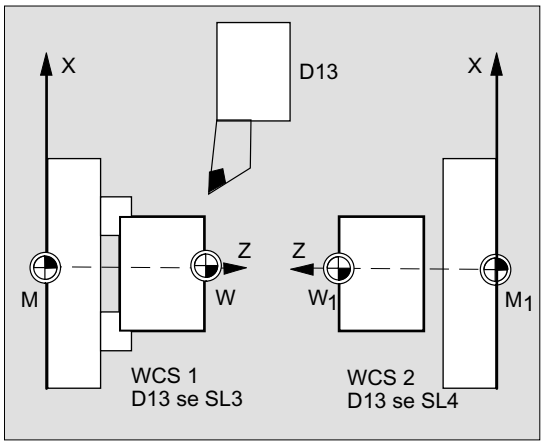
Další možnosti použití

Použití funkce orientovatelného držáku nástroje může být užitečné i tehdy, pokud stroj fyzicky nenabízí žádnou možnost nástroje otáčet do různých poloh, místo toho jsou však pevně instalovány nástroje s různými orientacemi. Při určování rozměrů nástrojů je potom možno pracovat s jednotnou základní orientací a rozměr, který je zapotřebí obrobit, vznikne příslušným otočením virtuálního držáku nástroje.

7.3.1 Zrcadlové převrácení délek nástroje

Funkce

Když je nastavovaným parametrům SD42900 $SC_MIRROR_TOOL_LENGTH a SD42910 $SC_MIRROR_TOOL_WEAR přiřazena nenulová hodnota, můžete provádět zrcadlové převrácení složek délky nástroje a složek základního rozměru odpovídajících příslušným osám.



SD42900 $SC_MIRROR_TOOL_LENGTH

Nastavovaný parametr se **nerovná** nule:

Zrcadlové převrácení – prostřednictvím změny znaménka – se bude týkat složek délky nástroje ($TC_DP3, $TC_DP4 a $TC_DP5) a složek základních rozměrů ($TCDP21, $TC_DP22 a $TC_DP23), jejichž osy jsou zrcadlově převraceny.

Zrcadlové převracení se **netýká** hodnot opotřebení. Pokud mají být převraceny také, musí být aktivován nastavovaný parametr SD42910 $SC_MIRROR_TOOL_WEAR.

SD42910 $SC_MIRROR_TOOL_WEAR

Nastavovaný parametr se **nerovná** nule:

Hodnoty opotřebení složek délky nástroje, pro něž jsou příslušné osy zrcadlově převraceny, se budou změnou znaménka zrcadlově převracet také.

7.3.2 Vyhodnocování znaménka opotřebení

Funkce

Když je nastavovaným parametrům SD42920 $SC_WEAR_SIGN_CUTPOS a SD42930 $SC_WEAR_SIGN přiřazena nenulová hodnota, můžete invertovat vyhodnocování znaménka u složek opotřebení.

SD42920 $SC_WEAR_SIGN_CUTPOS

Nastavovaný parametr se **nerovná** nule:

U nástrojů s relevantní polohou břitu (soustružnické a brusné nástroje – typy nástrojů 400) závisí vyhodnocování znaménka složek opotřebení v rovině obrábění na poloze břitu. U typů nástrojů bez relevantní polohy břitu nemá tento nastavovaný parametr žádný význam.

V následující tabulce jsou značkou X značeny rozměry, jejichž znaménko je nastavovaným parametrem SD42920 (různým od 0) invertováno.

Poloha břitu	Délka 1	Délka 2
1		
2		X
3	X	X
4	X	
5		
6		
7		X
8	X	
9		

Poznámka

Vyhodnocování znaménka prostřednictvím parametrů SD42920 a SD42910 jsou na sobě nezávislá. Jestliže se např. změní znaménko některého rozměru pomocí obou nastavovaných parametrů, zůstane výsledné znaménko nezměněno.

SD42930 $SC_WEAR_SIGN

Nastavovaný parametr se **nerovná** nule:

Znaménka všech rozměrů opotřebení jsou invertována. Tato změna se týká jak délky nástroje, tak i zbývajících veličin, jako jsou rádius nástroje, rádius zaoblení atd.

Jestliže je zadán kladný rozměr opotřebení, nástroj se stane „kratším“ a „tenčím“, viz "Korekční parametry nástroje – zvláštní zacházení"; kapitola „Aktivování změněných nastavovaných parametrů“.

7.3.3 Souřadný systém aktivního obrábění (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS)

Funkce

V závislosti na kinematice stroje nebo možnosti použití orientovatelného držáku nástroje se hodnoty opotřebení změřené v jednom z těchto souřadných systémů převádějí nebo transformují do vhodného souřadného systému.

Souřadné systémy momentálně probíhajícího obrábění

Korekce délky nástroje mohou vycházet z následujících souřadných systémů, které se mohou používat pro začleňování složky délky nástroje „opotřebení“ do aktivního nástroje pomocí odpovídající skupiny G-kódů 56.

- Souřadný systém stroje (MCS)
- Základní souřadný systém (BCS)
- Souřadný systém obrobku (WCS)
- Souřadný systém nástroje (TCS)
- Souřadný systém obrobku kinematické transformace (KCS)

Syntaxe

```
TOWSTD
TOWMCS
TOWWCS
TOWBCS
TOWTCS
TOWKCS
```

Význam

TOWSTD	Počáteční nastavení pro korekce hodnoty opotřebení v délce nástroje
TOWMCS	Korekce délky nástroje v MCS
TOWWCS	Korekce délky nástroje ve WCS
TOWBCS	Korekce délky nástroje v BCS
TOWTCS	Korekce délky nástroje na vztažný bod držáku nástroje (orientovatelný držák nástroje)
TOWKCS	Korekce délky nástroje na hlavu nástroje (kinematická transformace)

Další informace

Rozlišovací charakteristiky

V následující tabulce jsou shrnuty nejdůležitější rozlišovací charakteristiky:

G-kód	Hodnota opotřebení	Aktivní orientovatelný držák nástroje
TOWSTD	Hodnota základního nastavení, délka nástroje	Hodnoty opotřebení podléhají otáčení.
TOWMCS	Hodnoty opotřebení v MCS. TOWMCS a TOWSTD jsou identické, pokud není aktivní žádný orientovatelný držák nástroje.	Otáčí se pouze vektor výsledné délky nástroje bez ohledu na opotřebení.
TOWWCS	Hodnota opotřebení ve WCS se přepočítává na MCS.	Vektor nástroje se počítá bez ohledu na opotřebení stejně jako u TOWMCS.
TOWBCS	Hodnota opotřebení v BCS se přepočítává na MCS.	Vektor nástroje se počítá bez ohledu na opotřebení stejně jako u TOWMCS.
TOWTCS	Hodnota opotřebení v souřadném systému nástroje se přepočítává na MCS.	Vektor nástroje se počítá bez ohledu na opotřebení stejně jako u TOWMCS.

TOWWCS , TOWBCS, TOWTCS: Vektor opotřebení se přičítá k vektoru nástroje.

Lineární transformace

Délka nástroje je v MCS smysluplně definovatelná jen tehdy, pokud MCS vychází z BCS prostřednictvím lineární transformace.

Nelineární transformace

Pokud je např. pomocí TRANSMIT aktivována nelineární transformace, pak bude-li jako požadovaný souřadný systém zadán MCS, automaticky se použije BCS.

Žádná kinematická transformace a žádný orientovatelný držák nástroje

Pokud nejsou aktivní ani kinematická transformace a ani orientovatelný držák nástroje, potom s výjimkou WCS jsou všechny čtyři zbývající souřadné systémy shodné. V důsledku toho se pouze WCS odlišuje od zbývajících. Protože se mají vyhodnocovat pouze délky nástroje, nemají posunutí mezi jednotlivými souřadnými systémy žádný význam.

Literatura:
Pokud budete potřebovat další informace týkající se korekčních parametrů nástroje, viz: Příručka Popis funkcí, Základní funkce; "Korekční parametry nástrojů (W1)

Započítávání hodnot opotřebení

Nastavovaný parametr **SD42935 $SC_WEAR_TRANSFORM** definuje, které z následujících tří složek opotřebení:

- Opotřebení
- Součtová korekce jemná
- Součtová korekce hrubá

mají podléhat rotaci pomocí transformace adaptéru nebo orientovatelného držáku nástroje, pokud je aktivní některý z následujících G-kódů:

- `TOWSTD` Základní nastavení

  pro korekce hodnoty ve směru délky nástroje
- `TOWMCS` Hodnoty opotřebení

  v souřadném systému stroje (MCS)
- `TOWWCS` Hodnoty opotřebení

  v souřadném systému obrobku (WCS)
- `TOWBCS` Hodnoty opotřebení (BCS)

  v základním souřadném systému
- `TOWTCS` Hodnoty opotřebení v souřadném systému nástroje na držáku nástroje (vztažný bod na držáku nástroje T)
- `TOWKCS` Hodnoty opotřebení v souřadném systému hlavy nástroje při kinetické transformaci

Poznámka

Vyhodnocování jednotlivých složek opotřebení (přiřazení geometrickým osám, vyhodnocení znaménka) je ovlivňováno následujícími faktory:

- aktivní rovina
- transformace adaptéru
- následující nastavované parametry
  - SD42910 $SC_MIRROW_TOOL_WEAR
  - SD42920 $SC_WEAR_SIGN_CUTPOS
  - SD42930 $SC_WEAR_SIGN
  - SD42940 $SC_TOOL_LENGTH_CONST
  - SD42950 $SC_TOOL_LENGTH_TYPE

7.3.4 Délka nástroje a změna roviny

Funkce

Když je nastavovanému parametru SD42940 $SC_TOOL_LENGTH_CONST přiřazena nenulová hodnota, můžete jednotlivé složky délky nástroje, jako jsou délka, opotřebení a základní rozměr, přiřadit při změně roviny geometrickým osám pro soustružnické a brusné nástroje.

SD42940 $SC_TOOL_LENGTH_CONST

Nastavovaný parametr se **nerovná** nule:

Přiřazení složek délky nástroje (délka, opotřebení a zkladní rozměr) geometrickým osám se při změně roviny obrábění (`G17` - `G19`) nemění.

Následující tabulka ukazuje přiřazení složek délky nástroje geometrickým osám pro soustružnické a brusné nástroje (typ nástroje 400 – 599):

Obsah	Délka 1	Délka 2	Délka 3
17	Y	X	Z
*)	X	Z	Y
19	Z	Y	X
-17	X	Y	Z
-18	Z	X	Y
-19	Y	Z	X

*) Každá hodnota, která se nerovná nule a ani některé z těchto šesti uváděných hodnot, bude vyhodnocena jako hodnota 18.

Následující tabulka ukazuje přiřazení složek délky nástroje geometrickým osám pro všechny ostatní nástroje (typ nástroje < 400, příp. > 599):

Rovina obrábění	Délka 1	Délka 2	Délka 3
*)	Z	Y	X
18	Y	X	Z
19	X	Z	Y
-17	Z	X	Y
-18	Y	Z	X
-19	X	Y	Z

*) Každá hodnota, která se nerovná nule a ani některé z těchto šesti uváděných hodnot, bude vyhodnocena jako hodnota 17.

Poznámka

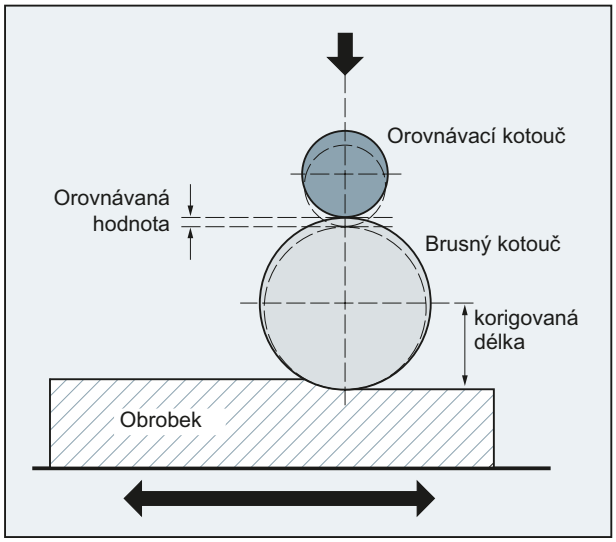
Při zobrazování v tabulkách se vychází z toho, že geometrické osy až 3 jsou označeny X, Y, Z. Pro přiřazení korekce ose je určující ne identifikátor osy, ale posloupnost os.

7.4 On-line korekce nástroje (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF)

Funkce

Když je aktivní funkce "On-line korekce nástroje", bude se u brusných nástrojů korekce délky nástroje, která vyplývá z opracování, započítávat okamžitě.

Příkladem použití je orovnávání CD, při kterém je brusný kotouč orovnáván souběžně s obráběním.



Korekce délky nástroje může být změněna z kanálu, ve kterém probíhá obrábění, nebo z paralelního kanálu (kanál orovnávače).

Pro zápis on-line korekce nástroje se používají různé funkce v závislosti na požadovaném okamžiku operace orovnávání.

- Kontinuální zápis v bloku (`PUTFTOCF`)

  V případě příkazu `PUTFTOCF` se operace orovnávání uskutečňuje souběžně s obráběním.

  Korekce nástroje se v kanálu, v němž probíhá obrábění, kontinuálně mění podle polynomické funkce 1., 2. nebo 3. řádu, která musí být předtím naprogramována pomocí příkazu `FCTDEF`.

  Příkaz `PUTFTOCF` se vždy uplatňuje blokově, tzn. v navazujícím bloku posuvu.
- Kontinuální zápis s modální platností: `ID=1 DO FTOC` (viz "On-line korekce nástroje (FTOC) [Strana 598]")
- Diskrétní zápis (`PUTFTOC`)

  V případě příkazu `PUTFTOC` se operace orovnávání neuskutečňuje souběžně s obráběním z paralelního kanálu. Hodnoty korekcí zadané pomocí příkazu `PUTFTOC` jsou v cílovém kanálu okamžitě v platnosti.

Poznámka

On-line korekce nástroje se může používat pouze u brusných nástrojů.

Syntaxe

Aktivování/deaktivování korekčních parametrů nástroje v cílovém kanálu:

```
FTOCON
...
FTOCOF
```

Zápis do on-line korekčních parametrů nástroje:

- Kontinuálně blokově:

```
FCTDEF(<funkce>,<LLimit>,<ULimit>,<a0>,<a1>,<a2>,<a3>)
PUTFTOCF(<funkce>,<vztažná hodnota>,<parametr nástroje>,<kanál>,<vřeteno>)
...
```

- Diskrétně:

```
PUTFTOC(<hodnota korekce>,<parametr nástroje>,<kanál>,<vřeteno>)
...
```

Význam

FTOCON: Aktivování on-line korekčních parametrů nástroje
Příkaz FTOCON musí být naprogramován v kanálu, v němž se má on-line korekce nástroje uplatňovat.

FTOCOF: Přerušení on-line korekčních parametrů nástroje
V případě příkazu FTOCOF se nebude na zadanou korekci už dále najíždět, ve specifických korekčních parametrech břitu je ale kompletní hodnota zadaná příkazy PUTFTOC/PUTFTOCF opravena.

Upozornění:
Aby byla on-line korekce nástroje s konečnou platností deaktivována, musí být po příkazu FTOCOF provedeno ještě aktivování/deaktivování nástroje (T...).

FCTDEF: Pomocí příkazu FCTDEF je definována polynomická funkce pro příkaz PUTFTOCF.

Parametr:

<funkce>:	Číslo polynomické funkce	
	Typ:	INT
<LLimit>:	Dolní mezní hodnota	
	Typ:	REAL
<ULimit>:	Horní mezní hodnota	
	Typ:	REAL
<a0> ... <a3>:	Koeficienty polynomické funkce	
	Typ:	REAL

`PUTFTOCF`: Vyvolání funkce "Kontinuální zápis on-line korekce nástroje v bloku"

Parametry:

`<funkce>`: Číslo polynomické funkce

Typ: INT

Upozornění:
Musí se shodovat se zadáním v příkazu `FCTDEF`.

`<vztažná hodnota>`: Proměnná vztažná hodnota, od které má být korekce odvozena (např. skutečná hodnota, která se má měnit).

Typ: VAR REAL

`<parametr nástroje>`: Číslo parametru opotřebení (Délka 1, 2 nebo 3), ke kterému se má hodnota korekce přičítat.

Typ: INT

`<kanál>`: Číslo kanálu, v němž má být on-line korekce nástroje v platnosti.

Typ: INT

Upozornění:
Zadání tohoto parametru je zapotřebí jen tehdy, pokud se hodnota korekce nemá uplatňovat v aktivním kanálu.

`<vřeteno>`: Číslo vřetena, pro které má být on-line korekce nástroje v platnosti.

Typ: INT

Upozornění:
Zadání tohoto parametru je zapotřebí jen tehdy, pokud má být korekce uplatňována na neaktivní brusný kotouč a nikoli na nástroj, který je momentálně používán.

`PUTFTOC`: Vyvolání funkce pro "Diskrétní zápis on-line korekce nástroje"

Parametry:

`<hodnota korekce>`: Hodnota korekce, která má být přičtena do parametru opotřebení.

Typ: REAL

`<parametr nástroje>`: viz `PUTFTOCF`

`<kanál>`: Číslo kanálu, v němž má být on-line korekce nástroje v platnosti.

Typ: INT

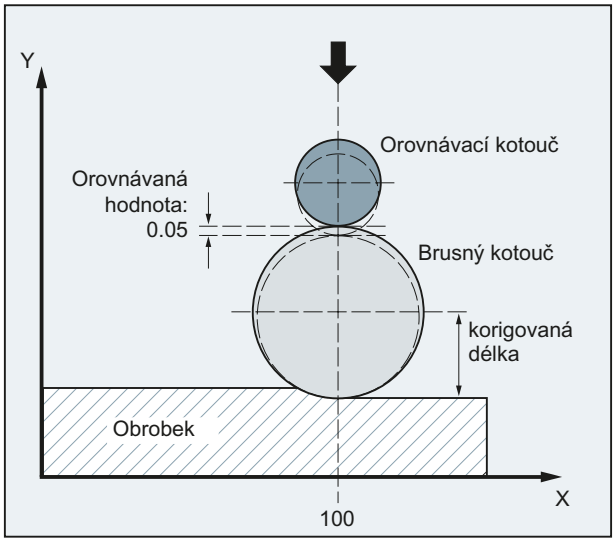
`<vřeteno>`: viz `PUTFTOCF`

Příklad

Obráběcí stroj s plochým brusným kotoučem s osami:

- Y: Přísuvná osa pro brusný kotouč
- V: Přísuvná osa pro orovnávací kotouč
- Kanál pro zpracování: Kanál 1 s osami X, Z, Y
- Kanál orovnávače: Kanál 2 s osou V

Po zahájení brusného pohybu má být na pozici X100 brusný kotouč orovnán o rozměr 0,05. Orovnávaný rozměr má být zadán do parametrů brusného nástroje pomocí funkce "Kontinuální zápis on-line korekce nástroje".



Program pro obrábění v kanálu 1:

Programový kód	Komentář
…	
N110 G1 G18 F10 G90	; Základní nastavení.
N120 T1 D1	; Volba aktuálního nástroje.
N130 S100 M3 X100	; Zapnutí vřetena, najíždění na výchozí pozici.
N140 INIT(2,"ABRICHT","S")	; Aktivování programu pro orovnávání v kanálu 2.
N150 START(2)	; Spuštění programu pro orovnávání v kanálu 2.
N160 X200	; Najíždění na cílovou pozici.
N170 FTOCON	; Aktivování on-line korekce.
N… G1 X100	; Další opracování.
N… M30	

Program pro orovnávání v kanálu 2:

Programový kód	Komentář
…	
N40 FCTDEF(1,-1000,1000,-$AA_IW[V],1)	; Definice funkce: Přímka se stoupáním = 1.
N50 PUTFTOCF(1,$AA_IW[V],3,1)	; Kontinuální zápis do on-line korekčních parametrů nástroje: Na základě pohybů osy V se bude uskutečňovat korekce Délky 3 momentálně používaného brusného kotouče v kanálu 1.
N60 V-0.05 G1 F0.01 G91	; Přísuvný pohyb pro orovnání, příkaz PUTFTOCF se uplatňuje jenom v tomto bloku.
…	
N… M30	

Další informace

Všeobecné informace k on-line korekci nástroje

Při kontinuálním zápisu (v každém taktu IPO) se po zapnutí vyhodnocovací funkce každá změna aditivně započítává do paměti opotřebení (aby se zabránilo skokovým změnám požadované hodnoty).

V každém případě platí: On-line korekce nástroje se může uplatňovat v každém kanálu, pro každé vřeteno a pro parametry opotřebení Délka 1, 2 **nebo** 3.

Přiřazení délek geometrickým osám se uskutečňuje v závislosti na aktuální pracovní rovině.

Když jsou aktivní příkazy `GWPSON`, příp.. `TMON`, probíhá přiřazení vřetena k nástroji prostřednictvím parametrů nástroje, pokud se však nejedná o aktivní brusný kotouč.

Korekce se vždy týká parametrů opotřebení pro aktuální stranu brusného kotouče, příp. pro levou stranu kotouče, pokud nástroj není aktivní.

Poznámka

V případě identické korekce pro více stran brusného kotouče je zapotřebí se pomocí předpisu pro zřetězení postarat o to, aby byla hodnota automaticky převzata i pro druhou stranu.

Pokud jsou pro kanál, v němž probíhá opracování, zadány on-line korekce, potom hodnoty opotřebení pro aktuální nástroj nesmí být v tomto kanálu upravovány z výrobního programu nebo pomocí zásahu obsluhy.

On-line korekce nástroje se uplatňují také v případě aktivování konstantní obvodové rychlosti (`SUG`) a i monitorování nástroje (`TMON`).

7.5 Aktivování 3D korekcí nástroje (CUT3DC..., CUT3DF...)

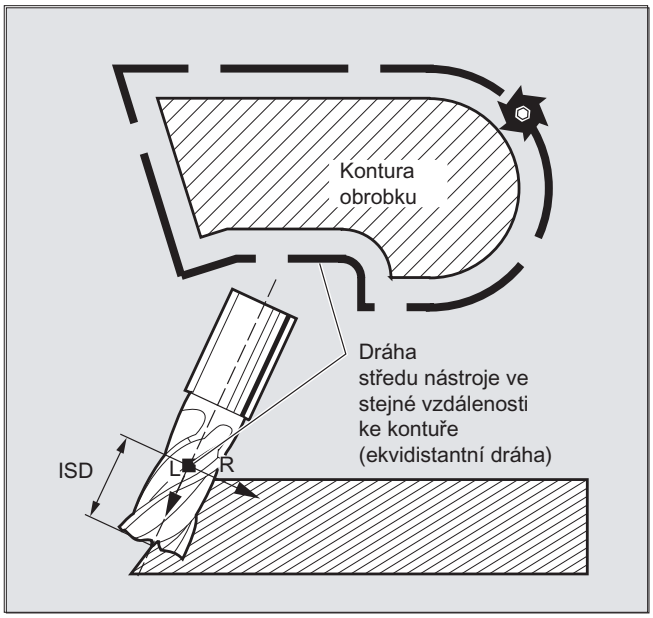
7.5.1 Aktivování 3D korekcí nástroje (CUT3DC, CUT3DF, CUT3DFS, CUT3DFF, ISD)

Funkce

U korekce rádiusu nástroje pro válcové nástroje se zohledňuje změněná orientace tohoto nástroje.

Pro aktivování 3D korekce rádiusu nástroje platí stejné programové příkazy jako v případě 2D korekce rádiusu nástroje. Pomocí příkazů `G41`/`G42` se zadává korekce vlevo/vpravo ve směru opracovávání. Jako chování při najíždění je vždy zvoleno `NORM`. 3D korekce rádiusu nástroje se uplatňuje jen tehdy, když je aktivní 5-osá transformace.

3D-korekce rádiusu nástroje je označována také jako 5D-korekce, protože v tomto případě je pro polohu nástroje v prostoru k dispozici 5 stupňů volnosti.



Rozdíl mezi 2 1/2D a 3D korekcí rádiusu nástroje

V případě 3D korekce rádiusu nástroje se může orientace nástroje měnit. V případě 2 1/2D korekce rádiusu nástroje se pracuje pouze s nástrojem s konstantní orientací.

Syntaxe

```
CUT3DC
CUT3DFS
CUT3DFF
CUT3DF
ISD=<hodnota>
```

Význam

`CUT3DC`	Aktivování 3D korekce rádiusu pro obvodové frézování
`CUT3DFS`	D-korekce nástroje pro frézování na čelní ploše s konstantní orientací Orientace nástroje je definována příkazy `G17` - `G19` a nemůže být ovlivňována pomocí framů.
`CUT3DFF`	D-korekce nástroje pro frézování na čelní ploše s konstantní orientací Orientace nástroje je definována příkazy `G17` - `G19` a v případě potřeby může být pomocí framu pootočena do jiného směru.
`CUT3DF`	D-korekce nástroje pro frézování na čelní straně se změnami orientace (jen když je aktivní 5-osá transformace).
`G40 X... Y... Z...`	Pro deaktivování: Lineární blok s příkazy `G0`/`G1` s geometrickými osami
`ISD`	Hloubka zajíždění nástroje

Poznámka

Příkazy mají modální platnost a nacházejí se ve stejné skupině jako příkazy `CUT2D` a `CUT2DF`. Deaktivování se uskuteční teprve s následujícím pohybem v aktuální rovině. Pro příkaz `G40` platí tato zásada vždy a nezávisle na příkazu `CUT`.

Pomocné vkládané bloky při aktivní 3D korekci rádiusu nástroje jsou přípustné. Platí definice 2 1/2D-korekce rádiusu nástroje.

Okrajové podmínky

- **G450/G451 a DISC**

  Na vnějších rozích se vždy vkládá blok s kruhovým pohybem. Příkazy `G450/G451` nemají žádný význam.
  Příkaz `DISC` se nevyhodnocuje.

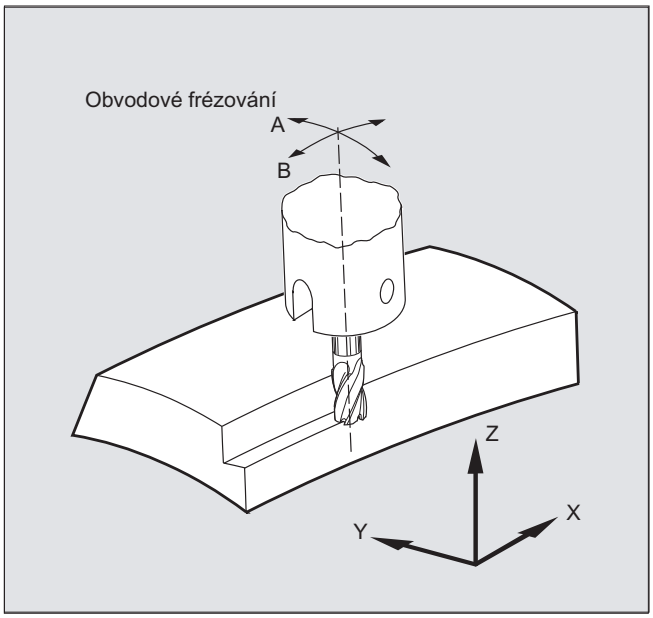
Příklad

```
Programový kód                  Komentář
N10 A0 B0 X0 Y0 Z0 F5000
N20 T1 D1                       ; Vyvolání nástroje, vyvolání korekčních parametrů
                                  nástroje.
N30 TRAORI(1)                   ; Aktivování transformace
N40 CUT3DC                      ; Aktivování 3D korekce rádiusu nástroje
N50 G42 X10 Y10                 ; Aktivování korekce rádiusu nástroje
N60 X60
N70 …
```

7.5.2 3D korekce nástroje: Obvodové frézování, frézování na čelní ploše

Obvodové frézování

Zde využívaná varianta obvodového frézování je realizována zadáním dráhy (řídící křivky) a k ní odpovídající orientace. U tohoto druhu obrábění nemá tvar nástroje na dráze žádný význam. Rozhodující je pouze rádius v bodě záběru nástroje.

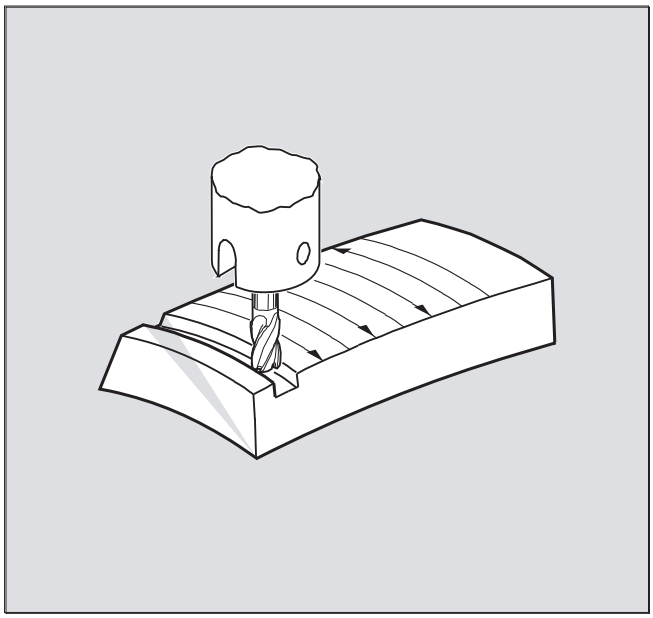


Poznámka

Funkce 3D korekce rádiusu nástroje je omezena jen na válcové nástroje.

Čelní frézování

Pro tento druh 3D frézování potřebujete popis řádek po řádku jednotlivých 3D drah na povrchu obrobku. Při výpočtech se obvykle uskutečňuje v systému CAM a zohledňuje se při něm tvar nástroje a rozměry nástroje. Postprocesor zapisuje do výrobního programu - vedle NC bloků - orientace nástroje (když je aktivní 5-osá transformace) a G-kódy pro požadovanou 3D korekci nástroje. V důsledku toho má obsluha stroje možnost - narozdíl od případu, kdy se nástroj používá pro výpočet jeho drah v NC systému - nepatrně menší nástroje.



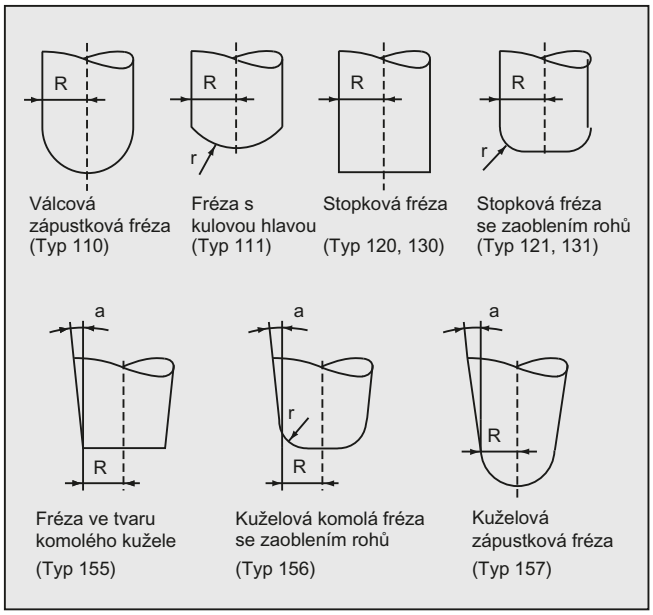
Příklad:

NC-bloky byly vypočítány s frézou 10 mm. Zpracování těchto bloků by mohlo být v tomto případě provedeno i s frézou 9,9 mm, přičemž by potom bylo nutno počítat se změněným profilem drsnosti.

7.5.3 3D korekce nástroje: Tvary nástrojů a parametry nástrojů pro frézování na čelní ploše

Tvary fréz, parametry nástroje

V následujících odstavcích jsou shrnuty možné tvary nástrojů pro frézování na čelní ploše a mezní hodnoty parametrů nástroje. Tvary stopky nástroje nejsou brány v úvahu. Typy nástrojů 120 a 156 mají stejné výsledky.



Jestliže je v NC programu zadáno jiné číslo typu, než jaké je uvedeno na tomto obrázku, systém automaticky použije typ nástroje 110 (válcová zápustková fréza). V případě překročení mezních hodnot pro parametry nástroje se aktivuje alarm.

Typ frézy	Č. typu	R	r	a
Válcová zápustková fréza	110	> 0	-	-
Fréza s kulovou hlavou	111	> 0	> R	-
Stopková fréza, fréza s úhlovou hlavou	120, 130	> 0	-	-
Stopková fréza, fréza s úhlovou hlavou se zaoblením rohů	121, 131	> r	> 0	-
Kuželová komolá fréza	155	> 0	-	> 0
Kuželová komolá fréza se zaoblením rohů	156	> 0	> 0	> 0
Kuželová zápustková fréza	157	> 0	-	> 0

R = rádius stopky (rádius nástroje)

r = rádius v rohu

a = úhel mezi osou nástroje a horním okrajem plochy torusu

- = nevyhodnocuje.se

Parametry nástroje	Parametry nástroje	
Rozměr nástroje	Geometrie	Opotřebení
R	$TC_DP6	$TC_DP15
r	$TC_DP7	$TC_DP16
a	$TC_DP11	$TC_DP20

Korekce délky nástroje

Jako vztažný bod pro korekci délky platí špička nástroje (průsečík podélné osy/povrchu).

3D korekce nástroje, výměna nástroje

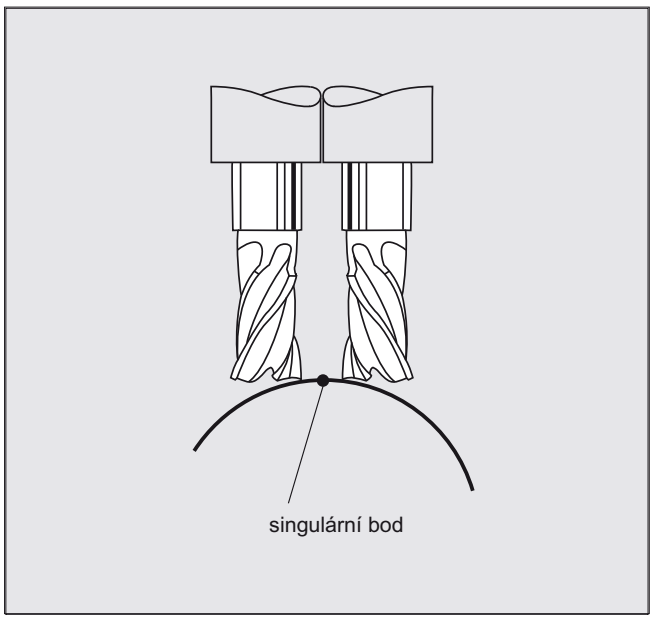
Nový nástroj se změněnými rozměry (R, r, a) nebo s jiným tvarem smí být zadán jedině s naprogramováním příkazu `G41`, příp. `G42` (přepnutí z `G40` do `G41`příp. `G42`, nové naprogramování příkazů `G41`, příp. `G42`). Na všechny ostatní parametry nástroje, např. jeho délku, se toto pravidlo nevztahuje, takže takové nástroje mohou být vyměňovány i bez toho, že by se znovu zadával příkaz `G41`, příp. `G42`

7.5.4 3D korekce nástroje: Korekce na dráze, zakřivení dráhy, hloubka zajíždění nástroje (CUT3DC, ISD)

Funkce

Korekce na dráze

Při frézování na čelní ploše je nutno mít na zřeteli případ, kdy se bod dotyku na povrchu nástroje vzdálí od obráběné plochy, jako je tomu v tomto příkladu při obrábění konvexní plochy s kolmo postaveným nástrojem. Použití uvedené na obrázku je možno považovat za mezní případ.



Tento mezní případ je řídícím systémem sledován, díky čemuž je možno na základě úhlového nastavení mezi nástrojem a normálovým vektorem plochy rozpoznat skokové změny polohy obráběného bodu. Na těchto místech řídící bod vkládá lineární bloky, takže pohyb může být uskutečněn.

Pro výpočet těchto lineárních bloků je ve strojním parametru uložen rozsah přípustných úhlů pro boční naklopení. Jestliže dojde k překročení mezních hodnot pro rozmezí přípustných úhlů definovaných ve strojních parametrech, ohlásí systém alarm.

Zakřivení dráhy

Zakřivení dráhy není monitorováno. Také v těchto případech se doporučuje používat jen takové nástroje, s nimiž je možno pracovat, aniž by došlo k narušení kontury.

Hloubka zajíždění nástroje (ISD)

Hloubka zajíždění nástroje ISD je vyhodnocována, jen když je aktivní 3D korekce rádiusu nástroje.

Pomocí programového příkazu `ISD` (Insertion Depth) se programuje hloubka zajíždění špičky nástroje při obvodovém frézování. Díky tomu je možné měnit polohu obráběcího bodu na ploše pláště nástroje.

Syntaxe

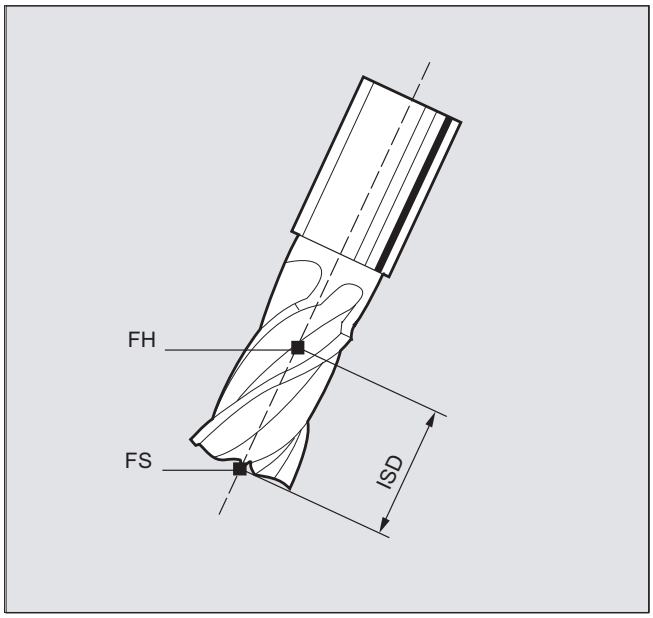
3D korekce nástroje, obvodové frézování
```
CUT3DC
ISD=<hodnota>
```

Význam

`CUT3DC`	Aktivování 3D korekce nástroje pro obvodové frézování, např. pro frézování kapes se šikmými bočními stěnami.
`ISD`	Pomocí příkazu `ISD` se zadává vzdálenost (`<hodnota>`) mezi špičkou frézy (FS) a pomocným bodem frézy (FH).

Pomocný bod frézy

Pomocný bod frézy (FH) vzniká promítnutím naprogramovaného obráběcího bodu na osu nástroje.



Další informace

Frézování kapes se šikmými bočními stěnami pro obvodové frézování s příkazem CUT3DC

Při této 3D korekci rádiusu nástroje se uskutečňuje kompenzace odchylky rádiusu frézy tím, že se provádí přísuv ve směru normálového vektoru obráběné plochy. Jestliže je ponechána hloubka zajíždění nástroje `ISD` na stejné hodnotě, zůstává rovina, ve které leží čelní plocha frézy, přitom nezměněna. Fréza s např. menším rádiusem, než jaký má normovaný nástroj, by potom nedosáhla na dno kapsy, které představuje také hraniční plochu. Aby se uskutečňoval automatický přísuv nástroje, musí být řídícímu systému tyto hraniční plochy známy, viz kapitola "3D obvodové frézování s omezujícími plochami".

Pokud budete potřebovat další informace týkající se monitorování kolizí, viz:
Literatura:
Příručka programování, Základy; kapitola "Korekční parametry nástroje"

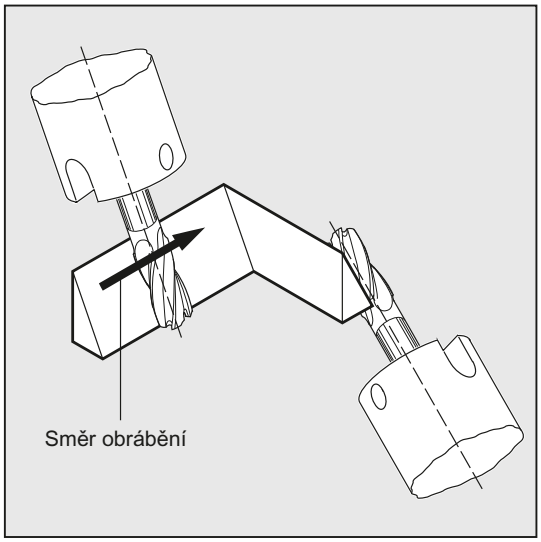
7.5.5 3D korekce nástroje: Vnitřní rohy/vnější rohy a chování v průsečíku (G450, G451)

Funkce

Vnitřní rohy/vnější rohy

Vnější a vnitřní rohy jsou řešeny každý jinak. Označení, zda je roh vnitřní nebo vnější, závisí na orientaci nástroje.

Při změnách orientace v rohu se může vyskytnout případ, že se typ rohu v průběhu opracovávání změní. Pokud taková situace nastane, obrábění se přeruší a aktivuje se chybové hlášení.



Syntaxe

```
G450
G451
```

Význam

`G450`	Přechodový prvek kruh (nástroj objíždí rohy obrobku po kruhové dráze)
`G451`	Průsečík ekvidistantních drah (nástroj řeže ostré rohy obrobku)

Další informace

Najíždění na průsečík pro 3D korekci

Při 3D obvodovém frézování se nyní na vnějších vyhodnocuje G-kód `G450/G451`, tzn. může se najíždět na průsečík offsetových křivek. Až do verze SW 4 byl na vnějších rozích vždy vkládán kruhový oblouk. Realizovatelné chování na průsečících je u typických 3D programů generovaných systémy CAD obzvláště výhodné. Tyto programy se často skládají z krátkých přímkových bloků (pro aproximaci hladkých křivek), u nichž jsou přechody mezi sousedícími bloky téměř tangenciální.

V případě korekce rádiusu nástroje na vnější straně kontury byly až dosud v zásadě vkládány kruhové oblouky, které zajišťovaly objíždění na vnějších rozích. Protože jsou tyto bloky při téměř tangenciálních přechodech velmi krátké, vznikají nežádoucí skokové změny rychlosti.

V těchto případech jsou analogicky s 2 ½ D-korekcí rádiusu obě na pohybu se podílející křivky prodlouženy a najíždí se na průsečík obou prodloužených křivek.

Průsečík se určuje tak, že se offsetové křivky z obou na pohybu se podílejících bodů prodlouží a určí se jejich průsečík v rovině kolmé na orientaci nástroje v rohu. Jestliže žádný takový průsečík neexistuje, bude roh opracován stejně jako dříve, tzn. vloží se kruhový oblouk.

Pokud budete potřebovat další informace o chování v průsečících, viz:
Literatura:
Příručka Popis funkcí, Speciální funkce; 3D korekce rádiusu nástroje (W5)

7.5.6 3D korekce nástroje: 3D obvodové frézování s omezujícími plochami

Přizpůsobení 3D obvodového frézování vlastnostem programů ze systémů CAD

NC programy generované systémy CAD zpravidla aproximují dráhu středu normovaného nástroje pomocí velkého počtu krátkých lineárních bloků. Aby tyto takto vytvořené bloky představující spoustu dílčích kontur co možno nejpřesněji napodobily předcházející původní konturu, je zapotřebí ve výrobním programu uskutečnit určitá přizpůsobení.

Důležité informace, které by byly zapotřebí pro dosažení optimální korekce, které ale ve výrobním programu už nejsou k dispozici, musí být nahrazeny pomocí vhodných opatření. Potom se představí typické metody, aby se vyrovnaly kritické přechody buď přímo ve výrobním programu nebo prostřednictvím vyšetření reálné kontury (např. pomocí přísuvu nástroje).

Použití

Kromě typických případů použití, při kterých namísto normovaného nástroje opisuje středovou dráhu reálný nástroj, se využívá také válcových nástrojů s 3D korekcí nástroje. Naprogramovaná dráha je přitom vztažena na rovinu obrábění. Omezující plocha, která je pro tento účel vhodná, závisí na nástroji. Pro výpočet offsetu kolmého k omezující ploše se používá stejně jako v případě obvyklé korekce rádiusu nástroje celkový rádius.

7.5.7 3D korekce nástroje: Zohlednění hraniční plochy (CUT3DCC, CUT3DCCD)

Funkce

3D obvodové frézování s reálnými nástroji

V případě 3D obvodového frézování se spojitou nebo konstantní změnou orientace nástroje je často programována dráha středu definovaného normovaného nástroje. Protože v praxi se nestává často, že by byl vyhovující normovaný nástroj k dispozici, může se použít nástroj, který se od normovaného nástroje přespříliš neliší.

Pomocí příkazu `CUT3DCCD` je možné pro specifický reálný nástroj zohlednit omezující plochu, která by popisovala naprogramovaný normovaný nástroj. NC program popisuje dráhu středu normovaného nástroje.

Pomocí příkazu `CUT3DCC` je možné při použití válcových nástrojů zohlednit omezující plochu, na kterou by měl naprogramovaný normovaný nástroj dosáhnout. NC program popisuje konturu obráběné plochy.

Syntaxe

```
CUT3DCCD
CUT3DCC
```

Význam

`CUT3DCCD`	Aktivování 3D korekce nástroje pro obvodové frézování s omezujícími plochami se specifickým nástrojem na dráze středu nástroje: Přísuv k omezující ploše.
`CUT3DCC`	Aktivování 3D korekce nástroje pro obvodové frézování s omezujícími plochami s 3D korekcí rádiusu: Kontura na obráběné ploše

Poznámka

Korekce rádiusu nástroje s příkazy G41, G42

Pro korekci rádiusu nástroje s příkazy `G41`, `G42`, když je aktivní funkce `CUT3DCCD` nebo `CUT3DCC` musí být k dispozici volitelný doplněk "Transformace orientace".

Normovaný nástroj se zaoblením rohů

Rohové zaoblení normovaného nástroje je popisováno pomocí parametru nástroje `$TC_DP7`. Z parametru nástroje `$TC_DP16` vyplývá odchylka rohového zaoblení reálného nástroje oproti normovanému nástroji.

Příklad

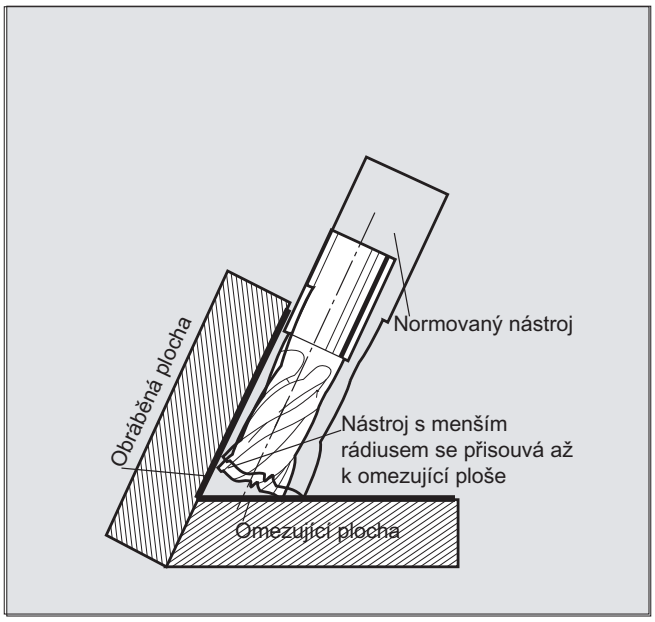
Rozměry nástroje pro Torus frézy se zmenšeným rádiusem oproti normovanému nástroji.

Typ nástroje	R = rádius stopky	r = rádius v rohu
Normovaný nástroj se zaoblením rohů	R = $TC_DP6	r = $TC_DP7
Reálný nástroj se zaoblením rohů Typy nástrojů 121 a 131 Torus frézy (stopkové frézy)	R' = $TC_DP6 + $TC_DP15 + OFFN	r' = $TC_DP7 + $TC_DP16
V tomto příkladu je záporná jak hodnota $TC_DP15 + OFFN, tak také hodnota $TC_DP16. Vyhodnocuje se typ nástroje ($TC_DP1).		
Přípustné jsou pouze typy fréz s válcovou stopkou (válcová nebo stopková fréza), jakož i Torus frézy (typ 121 a 131) a v mezním případě ještě i válcové zápustkové frézy (typ 110).	U těchto přípustných typů frézy je rohový rádius r roven rádiusu stopky R. Všechny ostatní přípustné typy nástrojů jsou interpretovány jako válcové frézy a případně zadaný rozměr pro rohové zaoblení se nevyhodnocuje.	
Přípustné jsou všechny typy nástrojů s čísly 1 - 399 s výjimkou čísel 111 a 155 až 157.		

Další informace

Dráha středu nástroje s přísuvem až po omezující plochu - CUT3DCCD

Jestliže je použit nástroj, který má ve srovnání s vyhovujícím normovaným nástrojem menší rádius, potom se v podélném směru přisouvaná fréza pohybuje ještě dál o takovou hodnotu, dokud znovu nedojde k dotyku se dnem kapsy. Tímto způsobem jsou růžky, které jsou tvořeny mezi obráběnou a omezující plochou, obrobeny až do takové míry, kterou daný nástroj připouští. Přitom se jedná o smíšený způsob obránění skládající se z obvodového a čelního frézování. Analogicky k nástroji s menším rádiusem se v případě použití nástroje se zvětšeným rádiusem přísuv zmenšuje o odpovídající hodnotu v opačném směru.



Oproti všem ostatním korekcím nástroje ze skupiny G-kódů č. 22 nemá pro příkaz `CUT3DCCD` zadávaný parametr nástroje `$TC_DP6` žádný význam pro rádius nástroje a výslednou konturu proto neovlivňuje.

Korekční offset se vypočítá jako součet následujících veličin:

- Hodnota opotřebení rádiusu nástroje (parametr nástroje `$TC_DP15`)
- a offsetu nástroje `OFFN` naprogramovaného pro výpočet kolmého offsetu k omezující ploše.

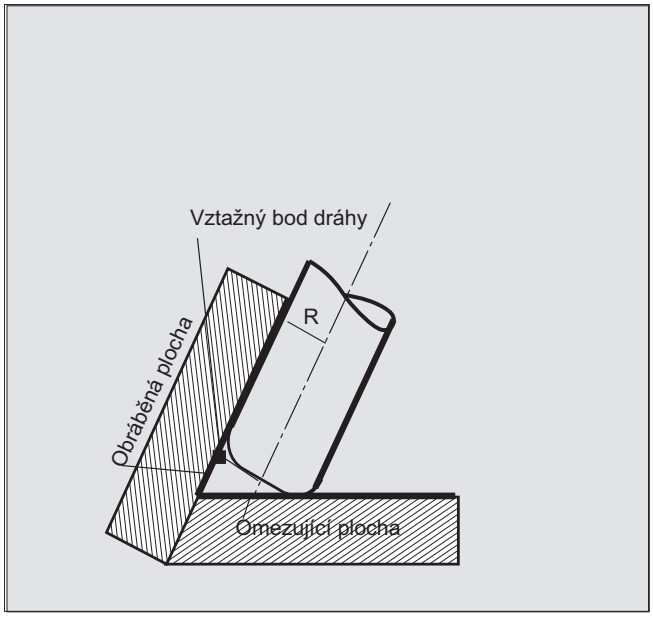
To, zda obráběná plocha leží vlevo nebo vpravo od dráhy, nemůže být ze sestaveného výrobního programu zjištěno. Vychází se proto z kladné hodnoty rádiusu a záporné hodnoty opotřebení původního nástroje. Záporná hodnota opotřebení vždy popisuje nástroj se zmenšeným průměrem.

Použití válcových nástrojů

Při použití válcových nástrojů je přísuv potřebný jedině tehdy, pokud obráběná plocha a omezující plocha mezi sebou svírají ostrý úhel (menší než 90 stupňů). Pokud se používají Torus frézy (válec s rohovým zaoblením), potom tato situace vyžaduje přísuv v podélném směru nástroje tak u ostrých, tak i u tupých úhlů.

3D korekce rádiusu s příkazem CUT3DCC, kontura na obráběné ploše

Pokud je aktivní příkaz `CUT3DCC` a používá se Torus fréza, je naprogramovaná dráha vztažena na fiktivní válcovou frézu stejného průměru. Z toho vyplývající vztažný bod dráhy je v případě použití Torus frézy uveden na následujícím obrázku.



Je přípustné, aby se úhel mezi obráběnou a omezující plochou i v rámci jednoho bloku měnil z ostrého na tupý a obráceně.

Oproti normovanému nástroji smí být použitý reálný nástroj jak větší, tak také menší. Výsledný rohový rádius přitom nesmí být záporný a znaménko výsledného rádiusu nástroje musí zůstat zachováno.

U příkazu `CUT3DCC` se NC program vztahuje na konturu v rovině obrábění. Přitom se používá stejně jako v případě obvyklé korekce rádiusu nástroje celkový rádius, který je tvořen součtem následujících veličin:

- Rádius nástroje (parametr nástroje `$TC_DP6`)
- Hodnota opotřebení (parametr nástroje `$TC_DP15`)
- a offsetu nástroje `OFFN` naprogramovaného pro výpočet kolmého offsetu k omezující ploše.

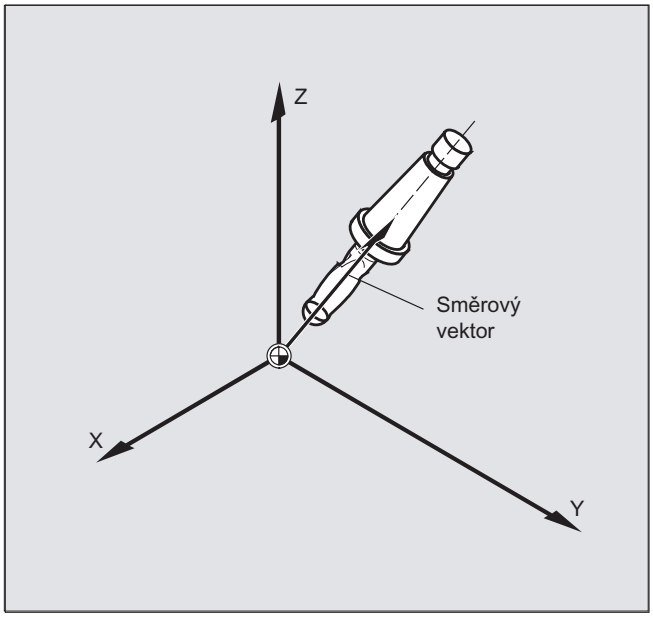
Poloha omezující plochy se určuje jako rozdíl obou těchto hodnot:

- Rozměry normovaného nástroje
- Rádius nástroje (parametr nástroje `$TC_DP6`)

7.6 Orientace nástroje (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST)

Funkce

Pod pojmem orientace nástroje se rozumí geometrické nasměrování nástroje v prostoru. V případě obráběcího stoje s 5 osami může být orientace nástroje nastavována pomocí programových příkazů.



Vyhlazovací pohyby orientace aktivované příkazy `OSD` a `OST` jsou v závislosti na druhu interpolace orientace nástroje realizovány různými způsoby.

Když je aktivní vektorová interpolace, interpoluje se vyhlazený průběh orientace také pomocí vektorové interpolace. Oproti tomu se při aktivní interpolaci kruhových os orientace vyhlazuje přímo pomocí pohybů kruhových os.

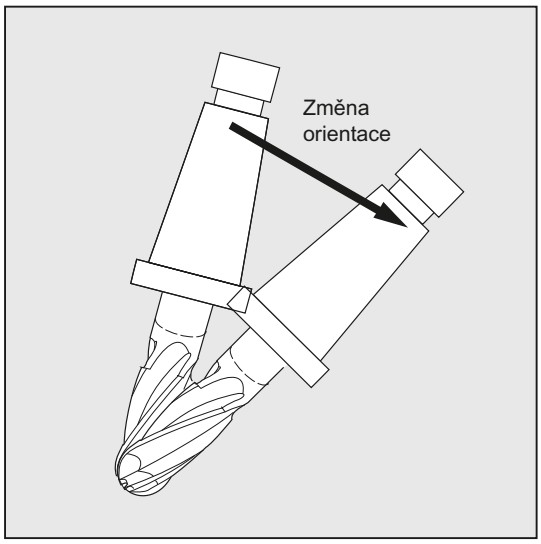
Programování

Programování změny orientace:

Změna orientace nástroje může být naprogramována pomocí těchto prostředků:

- Přímé programování kruhových os `A`, `B`, `C` (interpolace kruhových os)
- Eulerův úhel nebo úhel RPY.
- Směrový vektor (vektorová interpolace pomocí zadání příkazů `A3` nebo `B3` nebo `C3`)
- `LEAD/TILT` (frézování na čelní ploše)

Vztažným souřadným systémem je buď souřadný systém stroje (`ORIMKS`) nebo aktuální souřadný systém obrobku (`ORIWKS`).



Programování orientace nástroje:

Příkaz	Význam
ORIC:	Orientace a dráhový pohyb probíhají souběžně
ORID:	Orientace a dráhový pohyb probíhají jeden po druhém
OSOF:	Žádné vyhlazování orientace
OSC:	Konstantní orientace
OSS:	Vyhlazení orientace jen na začátku bloku
OSSE:	Vyhlazení orientace na počátku a konci bloku
ORIS:	Rychlost změn orientace při aktivovaném vyhlazování orientace ve stupních na mm (platí pro funkce OSS a OSSE)
OSD:	Vyhlazování orientace zadáním délky zaoblení pomocí nastavovaného parametru: SD42674 $SC_ORI_SMOOTH_DIST
OST:	Vyhlazování orientace v případě vektorové interpolace zadáním úhlové tolerance ve stupních pomocí nastavovaného parametru: SD42676 $SC_ORI_SMOOTH_TOL V případě interpolace kruhových os se zadaná tolerance používá jako maximální odchylka orientačních os.

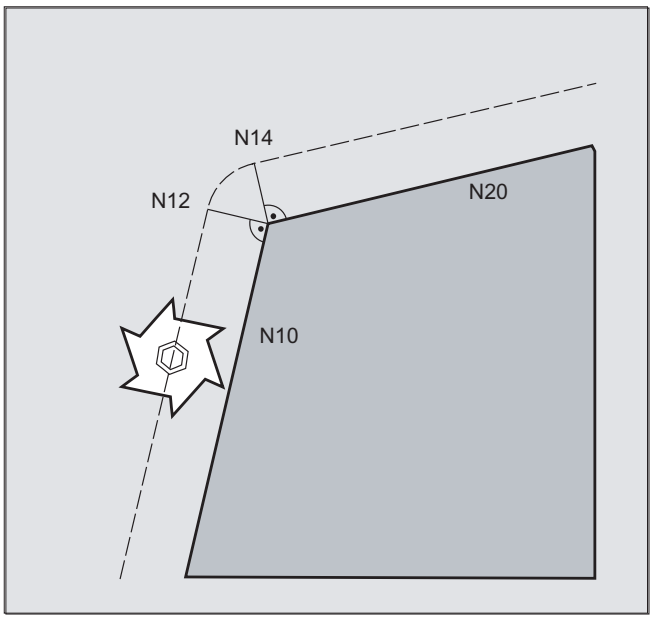
Poznámka

Všechny příkazy pro vyhlazování orientace nástroje (OSOF, OSC, OSS, OSSE, OSD a OST) jsou soustředěny ve skupině G-funkcí č. 34. Mají modální platnost, tzn. může být v platnosti vždy jen jeden z těchto příkazů.

Příklady

Příklad 1: ORIC

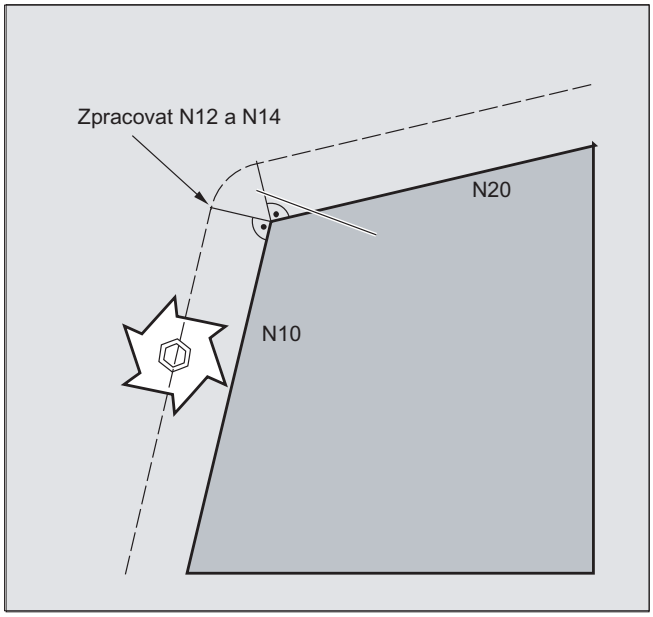
Pokud jsou mezi bloky posuvů `N10` a`N20`naprogramovány dva nebo více bloků se změnami orientace (např. `A2=... B2=... C2=...`) a pokud je aktivní funkce `ORIC`, potom se vkládaný blok kruhového oblouku rozdělí v závislosti na počtu úhlových změn na tyto pomocné bloky.



Programový kód	**Komentář**
`ORIC`	
`N8 A2=… B2=… C2=…`	
`N10 X… Y… Z…`	
`N12 C2=… B2=…` `N14 C2=… B2=…`	`; Blok kruhového oblouku, který se vkládá na vnějších rozích, se rozdělí na N12 a N14 v souladu se změnou orientace. Pohyb po kruhovém oblouku a změny orientace budou přitom prováděny souběžně.`
`N20 X =…Y=… Z=… G1 F200`	

Příklad 2: ORID

Jestliže je aktivní příkaz `ORID`, jsou všechny bloky mezi oběma posuvovými bloky uskutečňovány na konci prvního bloku posuvu. Blok kruhového oblouku s konstantní orientací bude uskutečněn bezprostředně před druhým blokem posuvu.



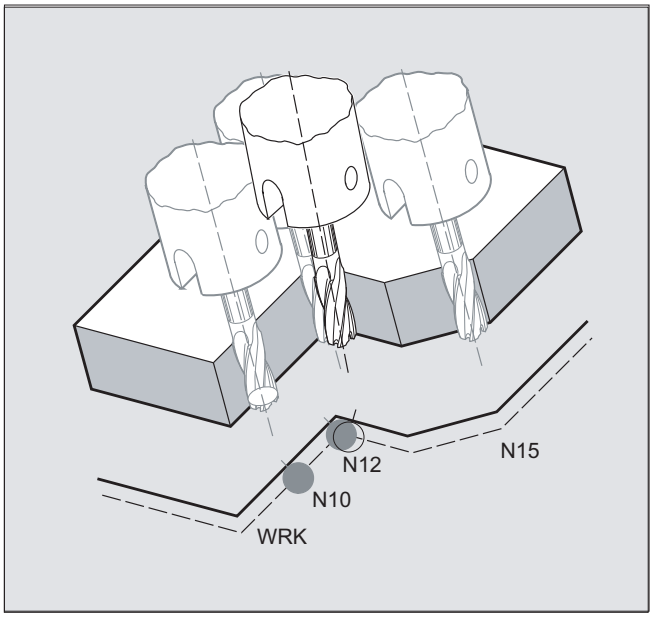
Programový kód	**Komentář**
`ORID`	
`N8 A2=… B2=… C2=…`	
`N10 X… Y… Z…`	
`N12 A2=… B2=… C2=…`	`; Bloky N12 a N14 se uskuteční na konci bloku N10. Potom bude následovat blok kruhového oblouku s momentálně nastavenou orientací.`
`N14 M20`	`; Pomocné funkce atd.`
`N20 X… Y… Z…`	

Poznámka

Pro způsob změny orientace na vnějším rohu je určující programový příkaz, který je aktivní v prvním bloku posuvu na vnějším rohu.

Beze změny orientace: Jestliže orientace na hranici bloku není změněna, potom se nástroj pohybuje po kruhové dráze, která se dotýká obou kontur.

Příklad 3: Změna orientace ve vnitřním rohu



Programový kód

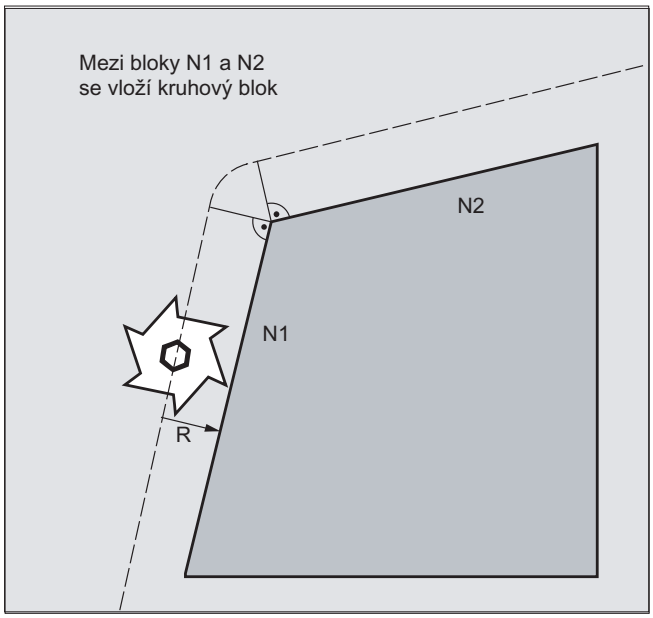
```
ORIC
N10 X …Y… Z… G1 F500
N12 X …Y… Z… A2=… B2=… C2=…
N15 X …Y… Z… A2=… B2=… C2=…
```

Další informace

Chování na vnějších rozích

Na vnějších rozích se vždy vkládá blok kruhového oblouku s rádiusem frézy.

Pomocí programových příkazů `ORIC`, příp. `ORID` může být definováno, zda se změny orientace, které byly naprogramovány mezi bloky `N1` a `N2`, uskuteční před zahájením zpracování vloženého bloku kruhového oblouku nebo současně s ním.



Jestliže je na vnějším rohu nezbytná změna orientace, potom si můžete vybrat, jestli se má uskutečnit souběžně s interpolací nebo odděleně od pohybu po dráze.

Je-li aktivní funkce `ORID`, uskuteční se napřed vložené bloky bez pohybu po dráze. Blok kruhového oblouku se vloží bezprostředně před druhý z obou bloků posuvu, kterými je roh tvořen.

Jestliže je na vnějším rohu vložen větší počet bloků změny orientace a pokud je aktivní funkce `ORIC`, potom se pohyb po kruhovém oblouku rozdělí na jednotlivé vkládané bloky v závislosti na počtu změn orientace.

Vyhlazení průběhu orientace pomocí příkazů OSD, příp. OST

V případě přechodových zaoblení aktivovaných funkcí `G642` se nemůže maximální odchylka pro konturové osy a orientační osy příliš lišit. Menší z obou tolerancí určuje formu pohybu na přechodových zaobleních, příp. toleranci úhlu, aby se průběh orientace relativně silně vyhladil, aniž by však přitom muselo docházet k velkým odchylkám od kontury.

Aktivováním příkazů `OSD`, příp. `OST` je možné zajistit, aby se s předem zadanou délkou přechodového zaoblení, příp. úhlovou tolerancí, probíhalo vyhlazení s jen velmi malými odchylkami průběhu orientace a bez velkých odchylek od kontury.

Poznámka

Narozdíl od vyhlazování kontury (a průběhu orientace) pomocí příkazu `G642` se při vyhlazování orientace pomocí příkazů `OSD`, příp. `OST` nevytváří žádný vlastní blok, nýbrž místo toho se zaoblovací pohyb vkládá přímo do naprogramovaných původních bloků.

Pomocí příkazů `OSD`, příp. `OST` nemohou být zaoblovány žádné přechody mezi bloky, ve který se uskutečňuje změna druhu interpolace pro orientaci nástroje (vektor → kruhové osy, kruhové osy → vektor). Tyto přechody mezi bloky mohou být v případě potřeby zaobleny pomocí běžných funkcí pro přechodová zaoblení `G641`, `G642`, příp. `G643`

7.7 Volné zadávání D-čísel, číslo břitu

7.7.1 Volné zadávání D-čísel, číslo břitu (adresa CE)

D-číslo

D-čísla mohou být stejně jako čísla korekcí zřetězena. Kromě toho je možné pomocí adresy CE určovat číslo břitu. Pomocí systémové proměnné $TC_DPCE může být číslo břitu popsáno.

Předdefinované nastavení: Číslo korekčních parametrů == číslo břitu

Prostřednictvím strojních parametrů jsou definovány maximální počet D-čísel (čísel břitů) a maximální počet břitů na jeden nástroj (→ výrobce stroje). Následující příkazy mají smysl jedině tehdy, pokud byl definován maximální počet břitů (MD18105) větší než počet břitů na jeden nástroj (MD18106). Věnujte prosím pozornost informacím od výrobce stroje.

Poznámka

Kromě zadávání relativních D-čísel mohou být zadávána D-čísla také jako "prostá", příp. "absolutní" D-čísla (1 - 32000) bez jakékoli souvislosti s T-čísly (v rámci funkce"Struktura prostých D-čísel").

Literatura

Příručka Popis funkcí, Základní funkce; "Korekční parametry nástrojů (W1)

7.7.2 Volné zadávání D-čísel: Kontrola D-čísla (CHKDNO)

Funkce

Pomocí příkazu `CHKDNO` zkontrolujete, zda příslušné D-číslo již bylo jednoznačně přiděleno. D-čísla všech v rámci jedné jednotky TO definovaných nástrojů se smí vyskytnout jen jedenkrát. Náhradní nástroje se přitom nezohledňují.

Syntaxe

```
state=CHKDNO(Tno1,Tno2,Dno)
```

Význam

`state`	= TRUE:	D-číslo bylo pro kontrolovaný rozsah jednoznačně přiděleno.
	= FALSE:	Vyskytla se kolize D-čísel nebo nastavení parametrů je neplatné. Pomocí parametrů Tno1, Tno2 a Dno jsou předávány parametry, které měly za následek kolizi.. Tato data mohou být ve výrobním programu vyhodnocována.
`CHKDNO(Tno1,Tno2)`	Budou zkontrolována všechna D-čísla uvedených nástrojů.	
`CHKDNO(Tno1)`	Budou zkontrolována všechna D-čísla nástroje Tno1 proti všem ostatním nástrojům.	
`CHKDNO`	Budou zkontrolována všechna D-čísla všech nástrojů proti všem ostatním nástrojům.	

7.7.3 Volné zadávání D-čísel: Přejmenovávání D-čísel (GETDNO, SETDNO)

Funkce

D-čísla musí být přiřazována jednoznačně. Dva různé břity nástroje nemohou mít stejné D-číslo.

GETDNO

Tento příkaz zjišťuje D-číslo určitého břitu (ce) nástroje s T-číslem t. Jestliže žádné D-číslo k zadanému parametru neexistuje, bude nastaveno d=0. Pokud je D-číslo neplatné, bude výsledkem hodnota větší než 32000.

SETDNO

Pomocí tohoto příkazu přiřadíte hodnotu d D-číslu břitu ce nástroje t. Pomocí stavové proměnné je oznamován výsledek tohoto příkazu (TRUE nebo FALSE). Jestliže žádný datový blok k uvedeným parametrům neexistuje, bude výsledkem hodnota FALSE. Syntaktická chyba bude mít za následek alarm. D-číslo není možné explicitně nastavit na 0.

Syntaxe

```
d = GETDNO (t,ce)

state = SETDNO (t,ce,d)
```

Význam

`d`	D-číslo břitu nástroje
`t`	T-číslo nástroje
`ce`	Číslo břitu (CE-číslo) nástroje
`state`	Udává, zda bylo možné uskutečnit příkaz bezchybně (TRUE nebo FALSE).

Příklad - Přejmenování D-čísla

Programování	Komentář
`$TC_DP2[1,2] = 120`	;
`$TC_DP3[1,2] = 5.5`	;
`$TC_DPCE[1,2] = 3`	; Číslo břitu CE
`...`	;
`N10 def int DNrAlt, DNrNeu = 17`	;
`N20 DNrAlt = GETDNO(1,3)`	;
`N30 SETDNO(1,3,DNrNeu)`	;

Tímto je břitu CE=3 přiřazena nová hodnota D-čísla 17. Nyní je přístup k datům tohoto břitu zajištěn přes D-číslo 17, a to jak prostřednictvím systémových proměnných, tak také při programování pomocí NC adresy.

7.7.4 Volné zadávání D-čísel: Zjišťování T-čísla k zadanému D-číslu (GETACTTD)

Funkce

Pomocí příkazu `GETACTTD` zjistíte T-číslo, které odpovídá k absolutnímu D-číslu. Neuskutečňuje se žádná kontrola jednoznačnosti. Jestliže v rámci jedné jednotky TO existuje větší počet stejných D-čísel, bude výsledkem T-číslo prvního nalezeného nástroje. Při použití "prostých" D-čísel nemá použití tohoto příkazu žádný smysl, protože výsledkem je vždy hodnota "1" (ve správě dat se nenachází žádné T-číslo).

Syntaxe

```
status=GETACTTD(Tnr,Dnr)
```

Význam

`Dnr`	D-číslo, pro které má být vyhledáno T-číslo.	
`Tnr`	Nalezené T-číslo	
`status`	Hodnota:	Význam:
	0	T-číslo bylo nalezeno Parametr Tnr obsahuje hodnotu T-čísla.
	-1	K zadanému D-číslu neexistuje žádné T-číslo, Tnr = 0.
	-2	D-číslo není absolutní. Tnr obsahuje hodnotu prvního nalezeného nástroje, který má D-číslo, jehož hodnota odpovídá parametru Dnr.
	-5	Funkce nemohla být z nějakého jiného důvodu uskutečněna.

7.7.5 Volné zadávání D-čísel: Nastavení D-čísla jako neplatného (DZERO)

Funkce

Příkaz `DZERO` poskytuje podporu během přezbrojení. Takto označené bloky korekčních parametrů už nejsou příkazem `CHKDNO` kontrolovány. Aby se znovu staly přístupnými, musí být D-čísla znovu nastavena pomocí příkazu `SETDNO`.

Syntaxe

```
DZERO
```

Význam

`DZERO` Označení všech D-čísel dané jednotky TO jako neplatné.

7.8 Kinematika držáku nástroje

Předpoklady

Držák nástroje je schopen nastavit nástroj do všech možných prostorových orientací jen tehdy, pokud:

- Jsou k dispozici dvě kruhové osy V_1 a V_2
- Tyto kruhové osy jsou vůči sobě kolmé
- Podélná osa nástroje se nachází kolmo na druhou kruhovou osu V_2.

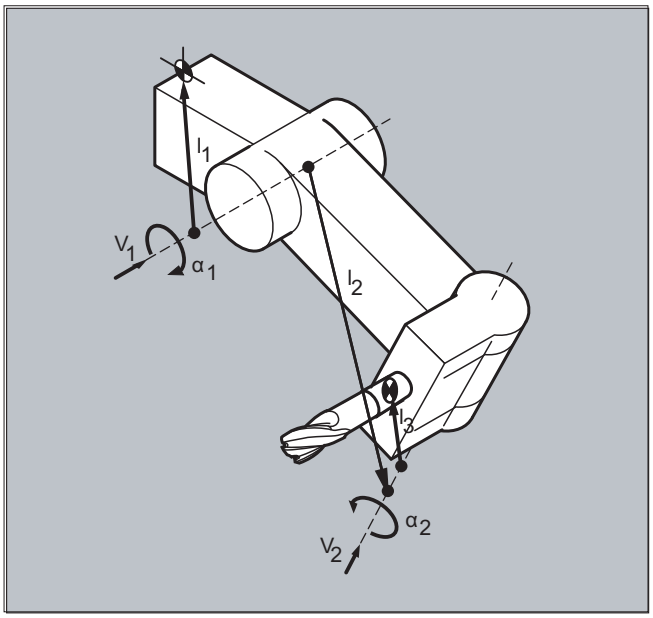
Kromě toho platí u strojů, u nichž musí být nastavitelné všechny možné orientace, následující předpoklad:

- Orientace nástroje nástroje musí být kolmo na první kruhovou osu V_1.

Funkce

Kinematika držáku nástroje s maximálně dvěma otočnými osami V_1 nebo V_2 je popsána pomocí 17 systémových proměnných `$TC_CARR1[m]` až `$TC_CARR17[m]`. Popis držáku nástroje se skládá z těchto částí:

- Vektorová vzdálenost od první kruhové osy po vztažný bod držáku nástroje I_1, vektorová vzdálenost od první ke druhé kruhové ose I_2 a vektorová vzdálenost od druhé kruhové osy ke vztažnému bodu nástroje I_3.
- Směrové vektory obou kruhových os V_1 a V_2
- Úhly otočení α_1, α_2 okolo obou os. Úhly otočení se určují při pohledu ve směru vektorů kruhových os a ve směru hodinových ručiček jsou považovány za kladné.



Pro stroje s **kinematikou s více stupni volnosti** (jak nástroj, tak i obrobek se mohou otáčet) byly systémové proměnné rozšířeny o položky:

- `$TC_CARR18[m]` až `$TC_CARR23[m]`.

Parametry

Funkce systémových proměnných pro orientovatelný držák nástroje			
Popis	**složka x**	**složka y**	**složka z**
l_1 vektor offsetu	$TC_CARR1[m]	$TC_CARR2[m]	$TC_CARR3[m]
l_2 vektor offsetu	$TC_CARR4[m]	$TC_CARR5[m]	$TC_CARR6[m]
v_1 kruhová osa	$TC_CARR7[m]	$TC_CARR8[m]	$TC_CARR9[m]
v_2 kruhová osa	$TC_CARR10[m]	$TC_CARR11[m]	$TC_CARR12[m]
α_1 úhel otočení α_2 úhel otočení	$TC_CARR13[m] $TC_CARR14[m]		
l_3 vektor offsetu	$TC_CARR15[m]	$TC_CARR16[m]	$TC_CARR17[m]

Rozšíření systémových proměnných pro orientovatelný držák nástroje			
Popis	**složka x**	**složka y**	**složka z**
l_4 vektor offsetu	$TC_CARR18[m]	$TC_CARR19[m]	$TC_CARR20[m]
Identifikátor kruhové osy V_1 kruhové osy V_2	Identifikátor kruhových os V_1 a V_2 (předdefinované nastavení je nula) $TC_CARR21[m] $TC_CARR22[m]		
Typ kinematiky **T**ool **P**art **M**ixed mode	$TC_CARR23[m]		
	Typ kinematiky T ->	Typ kinematiky P ->	Typ kinematiky M
	Otočný je jen nástroj (předdefinované nastavení)	Otočný je jen obrobek	Obrobek i nástroj jsou otočné
Offset kruhové osy V_1 kruhové osy V_2	Úhel ve stupních kruhové osy V_1 a V_2 při zaujmutí základní polohy $TC_CARR24[m] $TC_CARR25[m]		
Úhlový offset kruhové osy V_1 kruhové osy V_2	Offset Hirthova ozubení kruhových os V_1 a V_2 ve stupních$TC_CARR26[m] $TC_CARR27[m]		
Úhlový inkrement v_1 kruhová osa v_2 kruhová osa	Inkrement Hirthova ozubení kruhových os V_1 a V_2 ve stupních$TC_CARR28[m] $TC_CARR29[m]		
Min. poloha kruhové osy V_1 kruhové osy V_2	Softwarové koncové spínače pro minimální pozici kruhových os V_1 a V_2 $TC_CARR30[m] $TC_CARR31[m]		
Max. poloha kruhové osy V_1 kruhové osy V_2	Softwarové koncové spínače pro maximální pozici kruhových os V_1 a V_2 $TC_CARR32[m] $TC_CARR33[m]		
Název držáku nástroje	Namísto čísla může být držáku nástroje přiřazen název. $TC_CARR34[m]		

Rozšíření systémových proměnných pro orientovatelný držák nástroje			
Uživatel: Název osy 1 Název osy 2 Identifikace **Pozice**	Zamýšlené použití uživatelem v rámci měřicího systému. $TC_CARR35[m] $TC_CARR36[m] $TC_CARR37[m]		
	$TC_CARR38[m]	$TC_CARR39[m]	$TC_CARR40[m]
Jemné posunutí	Parametr, který se může **přičítat k hodnotám v základních parametrech**.		
l_1 vektor offsetu	$TC_CARR41[m]	$TC_CARR42[m]	$TC_CARR43[m]
l_2 vektor offsetu	$TC_CARR44[m]	$TC_CARR45[m]	$TC_CARR46[m]
l_3 vektor offsetu	$TC_CARR55[m]	$TC_CARR56[m]	$TC_CARR57[m]
l_4 vektor offsetu	$TC_CARR58[m]	$TC_CARR59[m]	$TC_CARR60[m]
v_1 kruhová osa	$TC_CARR64[m]		
v_2 kruhová osa	$TC_CARR65[m]		

Poznámka

Vysvětlivky k parametrům

Pomocí "m" se vždy zadává číslo popisovaného držáku nástroje.

Parametry `$TC_CARR47` až `$TC_CARR54`, jakož i `$TC_CARR61` až `$TC_CARR63` nejsou definovány a v případě pokusu o přístup kvůli jejich čtení nebo zápisu se aktivuje alarm.

Počáteční, příp. koncové body vektorů vzdálenosti na osách mohou být vybrány libovolně. Pro úhly otočení α_1, α_2 okolo obou os je v základním postavení držáku nástroje definována poloha 0°. Kinematika držáku nástroje takto může být popsána libovolným množstvím možností.

Držák nástroje s jen jednou nebo s žádnou kruhovou osou může být popsán dosazením nul směrovým vektorům jedné nebo obou kruhových os.
U držáku nástrojů bez kruhové osy se vektory vzdálenosti uplatňují jako doplňkové korekce nástroje, jejichž komponenty nejsou při přepnutí roviny obrábění (`G17` bis `G19`) ovlivňovány.

Rozšíření parametrů

Parametry kruhových os

Systémové proměnné byly rozšířeny o položky $TC_CARR24[m] až $TC_CARR33[m] a jsou popsány následujícím způsobem:

Offset kruhové osy V_1, V_2	Změna polohy kruhové osy V_1 nebo V_2 v základní poloze orientovatelného držáku nástroje.
Úhlový offset / úhlový inkrement kruhové osy V_1, V_2	Offset nebo úhlový inkrement Hirthova ozubení kruhových os V_1 a V_2. Naprogramovaný nebo vypočítaný úhel se zaokrouhluje na nejbližší hodnotu, která je dána výrazem phi = s + n * d, kde n je celé číslo.
Minimální a maximální poloha kruhové osy V_1, V_2	Minimální pozice/maximální pozice kruhové osy, úhel otočení (softwarový koncový spínač) kruhové osy V1 a V2.

Parametry pro uživatele

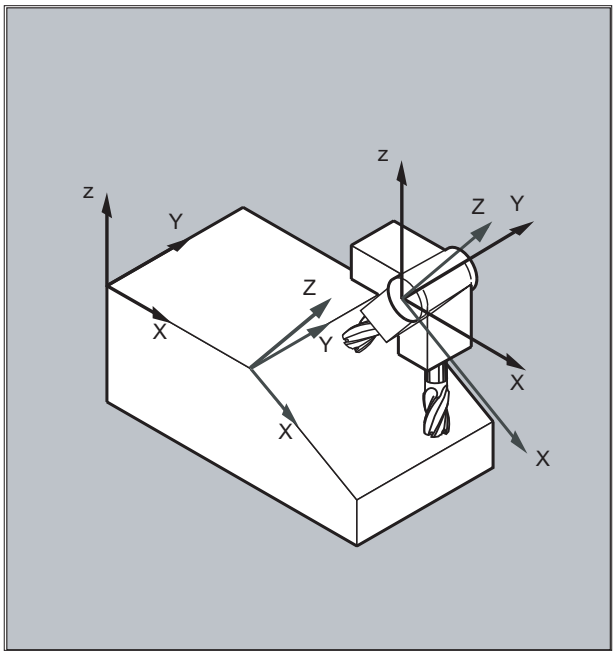
Proměnné $TC_CARR34 až $TC_CARR40 obsahují parametry, které jsou uživateli volně k dispozici a až do verze SW 6.4 standardně nejsou v rámci NCK dále vyhodnocovány a ani nemají žádný význam.

Parametry jemného posunutí

$TC_CARR41 až $TC_CARR65 obsahují parametry jemného posunutí, které se mohou přičítat k hodnotám základních parametrů. K hodnotě jemného posunutí přiřazené k určitému základnímu parametru se dostanete tak, že se k číslu parametru přičte hodnota 40.

Příklad

Držák nástroje používaný v následujícím příkladu se dá úplně popsat pomocí otočení okolo osy Y.



Programový kód	**Komentář**
N10 $TC_CARR8[1]=1	; Definice složky Y první kruhové osy držáku nástroje 1.
N20 $TC_DP1[1,1]=120	; Definice stopkové frézy.
N30 $TC_DP3[1,1]=20	; Definice délky stopkové frézy 20 mm.
N40 $TC_DP6[1,1]=5	; Definice rádiusu stopkové frézy 5 mm.
N50 ROT Y37	; Definice framu s otočením 37° okolo osy Y.
N60 X0 Y0 Z0 F10000	; Najíždění na výchozí pozici.
N70 G42 CUT2DF TCOFR TCARR=1 T1 D1 X10	; Nastavení korekce rádiusu, korekce délky nástroje v otočeném framu, aktivování držáku nástroje 1, nástroje 1.
N80 X40	; Uskutečnění obrábění s otočením 37°.
N90 Y40	
N100 X0	
N110 Y0	
N120 M30	

Další informace

Kinematika s více stupni volnosti

Pro stroje s kinematikou s více stupni volnosti (jak nástroj, tak i obrobek se mohou otáčet) byly systémové proměnné rozšířeny o položky `$TC_CARR18[m]` až `$TC_CARR23[m]`, které jsou popsány následujícím způsobem:

Otočný nástrojový stůl se skládá z těchto částí:

- Vektorová vzdálenost mezi otočnou osou V_2 a vztažným bodem otočného nástrojového stolu I_4 třetí kruhové osy.

Kruhové osy se skládají z těchto součástí:

- Dva kanálové identifikátory pro popis vztahu mezi kruhovými osami V_1 a V_2, jejichž pozic systém v případě potřeby využívá při určování orientace orientovatelného držáku nástroje.

Typ kinematiky s hodnotou T, P nebo M:

- Typ kinematiky T: Otočný je jen nástroj.
- Typ kinematiky P: Otočný je jen obrobek.
- Typ kinematiky M: Obrobek i nástroj jsou otočné.

Vymazání dat držáku nástroje

Pomocí nastavení `$TC_CARR1[0]=0` mohou být data všech datových bloků držáku nástroje vymazána.

Pro typ kinematiky `$TC_CARR23[T]=T` musí být dosazeno jedno ze tří přípustných velkých nebo malých písmen (T, P, M) a z tohoto důvodu by proměnná neměla být vymazána.

Editace dat držáku nástroje

Kterákoli z popisovaných hodnot může být změněna přiřazením nové hodnoty ve výrobním programu. Jakýkoli jiný znak než T, P nebo M bude mít při pokusu o aktivování orientovatelného držáku nástroje za následek aktivování alarmu.

Načítání dat držáku nástroje

Kterákoli z popisovaných hodnot může být čtena přiřazením proměnné ve výrobním programu.

Jemná posunutí

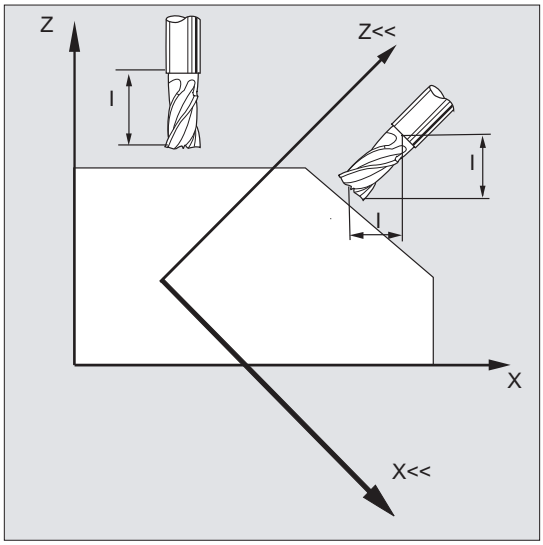
Nepřípustná hodnota jemného posunutí bude rozpoznána až tehdy, když je aktivován orientovatelný držák nástroje, který takovou hodnotu obsahuje, a když je současně nastaven nastavovaný parametr SD42974 $SC_TOCARR_FINE_CORRECTION = TRUE.

Hodnota nepřípustného jemného posunutí je pomocí strojního parametru omezena na maximální přípustnou hodnotu.

7.9 Délková korekce nástroje pro orientovatelný držák nástroje (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ)

Funkce

Při změně prostorové orientace nástroje se mění také jeho složky délky.



Po přestavení, např. manuálním nastavením nebo změnou držáku nástroje s pevným prostorovým nasměrováním, je proto nutné znovu zjistit jednotlivé složky délky nástroje. To se provádí pomocí příkazů dráhy `TCOABS` a `TCOFR`.

Pro orientovatelný držák nástroje aktivního framu je možné při volbě nástroje stanovit pomocí příkazů `TCOFRZ`, `TCOFRY` a `TCOFRX` směr, do nějž má nástroj ukazovat.

Syntaxe

```
TCARR=[<m>]
TCOABS
TCOFR
TCOFRZ
TCOFRY
TCOFRX
```

Význam

`TCARR=[<m>]`:	Vyžádání držáku nástroje s číslem [m]
`TCOABS`:	Výpočet složek délky nástroje z aktuální orientace držáku nástroje
`TCOFR`:	Stanovení složek délky nástroje z orientace aktivního framu
`TCOFRZ`:	Orientovatelný držák nástroje z aktivního framu, jehož nástroj je namířen ve směru osy Z.
`TCOFRY`:	Orientovatelný držák nástroje z aktivního framu, jehož nástroj je namířen ve směru osy Y.
`TCOFRX`:	Orientovatelný držák nástroje z aktivního framu, jehož nástroj je namířen ve směru osy X.

Další informace

Korekce délky nástroje z orientace držáku (TCOABS)

Příkaz `TCOABS` vypočítává korekci délky nástroje z aktuálního úhlu orientace držáku nástroje; ukládá se do systémových proměnných $TC_CARR13 a $TC_CARR14.

Budete-li potřebovat nastudovat definici kinematiky držáku nástroje pomocí systémových proměnných, viz "Kinematika držáku nástroje [Strana 443]".

Aby se uskutečnil nový výpočet korekce délky nástroje při změně framu, musí být nástroj ještě jednou aktivován.

Nasměrování nástroje podle aktivního framu

Orientovatelný držák nástroje může být nastaven tak, aby nástroj ukazoval do následujícího směru:

- pomocí příkazu `TCOFR`, příp. `TCOFRZ` bude natočen ve směru osy Z.
- pomocí příkazu `TCOFRY`bude natočen ve směru osy Y.
- pomocí příkazu `TCOFRX`bude natočen ve směru osy X.

Přepnutí mezi příkazy `TCOFR` a `TCOABS` má za následek nový výpočet korekce délky nástroje.

Vyžádání držáku nástroje (TCARR)

Pomocí příkazu `TCARR` si vyžádáte držák nástroje m spolu s jeho geometrickými údaji (paměť korekcí).

Když bude m=0, bude aktuální držák nástroje deaktivován.

Geometrické údaje o držáku nástroje se aktivují až po vyvolání nástroje. Zvolený nástroj zůstává aktivní i po výměně držáku nástroje.

Aktuální geometrické údaje o držáku nástroje mohou být definovány i ve výrobním programu prostřednictvím odpovídajících systémových proměnných.

Nový výpočet korekce délky nástroje (TCOABS) při změně framu

Aby se uskutečnil nový výpočet korekce délky nástroje při změně framu, musí být nástroj ještě jednou aktivován.

Poznámka

Orientace nástroje musí být manuálně přizpůsobena aktivnímu framu.

Při výpočtu délkové korekce nástroje se v jednom z mezikroků vypočítává také úhel natočení držáku nástroje. Protože u držáků nástrojů se dvěma kruhovými osami obecně existují také dva páry úhlů natočení, pomocí kterých je možné přizpůsobit orientaci nástroje aktuálnímu framu, musí hodnoty úhlu natočení uložené v systémových proměnných alespoň přibližně odpovídat mechanicky nastaveným úhlům natočení.

Poznámka

Orientace nástroje

Může se stát, že řídící systém nebude kontrolovat, jestli je možné úhly natočení vypočítané na základě orientace framu nastavit na stroji.

Pokud jsou kruhové osy držáku nástroje konstrukčně uspořádány tak, že orientace nástroje vypočítané na základě orientace framu nebude možné dosáhnout, bude generován alarm.

Kombinace jemné korekce nástroje a funkcí pro korekci délky nástroje nejsou u pohyblivých držáků nástrojů přípustné. Při pokusu vyvolat obě funkce současně se vypíše chybové hlášení.

Pomocí příkazu `TOFRAME` je možné definovat frame na základě směru orientace zvoleného držáku nástroje. Pokud budete potřebovat přesnější informace, viz kapitola "Framy".

Když je aktivní transformace orientace (transformace ve 3, 4, 5 osách), je možné zvolit držák nástroje s orientací odlišující se od nulové polohy, aniž by byl přitom generován alarm.

Předávané parametry pro standardní a měřicí cykly

Pro předávané parametry standardních a měřicích cyklů platí definované rozsahy hodnot.

V případě hodnot úhlů je rozsah hodnot definován následujícím způsobem:

- Otáčení okolo 1. geometrické osy: -180 stupňů až +180 stupňů
- Otáčení okolo 2. geometrické osy: -90 stupňů až +90 stupňů
- Otáčení okolo 3. geometrické osy: -180 stupňů až +180 stupňů

Viz kapitola Framy, "Programovatelné otočení (ROT, AROT, RPL)".

Poznámka

Při předávání hodnot úhlů standardním a měřicím cyklům je potřeba mít na paměti:

Hodnoty menší než je přesnost výpočtu NC systému jsou zaokrouhleny na nulu!

Přesnost výpočtu úhlových pozic v NC systému ve definována ve strojním parametru:

MD10210 $MN_INT_INCR_PER_DEG

7.10 On-line korekce délky nástroje (TOFFON, TOFFOF)

Funkce

Prostřednictvím systémové proměnné $AA_TOFF[<n>] mohou být trojrozměrně a v reálném čase korigovány efektivní délky nástroje ve všech třech jeho rozměrech.

Jako index <n> se používají tři identifikátory geometrických os. Tímto způsobem je definován počet aktivních směrů korekce prostřednictvím geometrických os, které jsou v daném okamžiku aktivní.

Všechny korekce mohou být aktivní současně.

Funkce On-line korekce délky nástroje se může používat v těchto případech:

- Transformace orientace TRAORI
- Orientovatelný držák nástroje TCARR

Poznámka

On-line korekce délky nástroje je **volitelný doplněk**, který musí být napřed instalován. Tato funkce má smysl jedině tehdy, když je aktivní transformace orientace nebo když je aktivní orientovatelný držák nástroje.

Syntaxe

```
TRAORI
TOFFON(<směr korekce>[,<hodnota offsetu>])
WHEN TRUE DO $AA_TOFF[<směr korekce>]                    ; V synchronních akcích.
...
TOFFOF(<směr korekce>)
```

Pokud budete potřebovat další vysvětlení týkající se programování on-line korekce délky nástroje v pohybových synchronních akcích, viz "On-line korekce délky nástroje ($AA_TOFF) [Strana 601]".

Význam

TOFFON: **Aktivování** on-line korekce délky nástroje

<směr korekce>: Směr nástroje (X, Y, Z), ve kterém se má on-line korekce délky nástroje uplatňovat.

<hodnota offsetu>: Při aktivování může být pro odpovídající směr korekce zadána hodnota offsetu, přičemž na tento offset se bude ihned najíždět.

TOFFOF: **Vynulování** on-line korekce délky nástroje

Hodnoty korekce v zadaném směru budou vynulovány a aktivuje se přitom zastavení předběžného zpracování.

Příklady

Příklad 1: Aktivování korekce délky nástroje

Programový kód	**Komentář**
MD21190 $MC_TOFF_MODE =1 MD21194 $MC_TOFF_VELO[0] =1000 MD21196 $MC_TOFF_VELO[1] =1000 MD21194 $MC_TOFF_VELO[2] =1000 MD21196 $MC_TOFF_ACCEL[0] =1 MD21196 $MC_TOFF_ACCEL[1] =1 MD21196 $MC_TOFF_ACCEL[2] =1	; Najíždí se na absolutní hodnoty.
N5 DEF REAL XOFFSET	
N10 TRAORI(1)	; Aktivování transformace.
N20 TOFFON(Z)	; Aktivování On-line korekce délky nástroje pro směr Z.
N30 WHEN TRUE DO $AA_TOFF[Z]=10 G4 F5	; Pro rozměr nástroje ve směru osy Z se interpoluje korekce délky nástroje 10.
...	
N100 XOFFSET=$AA_TOFF_VAL[X]	; Přiřazení aktuální korekce ve směru osy X.
N120 TOFFON(X,-XOFFSET) G4 F5	; Pro rozměr nástroje ve směru osy X se najede zpátky na hodnotu korekce délky nástroje 0.

Příklad 2: Deaktivování korekce délky nástroje

Programový kód	**Komentář**
N10 TRAORI(1)	; Aktivování transformace.
N20 TOFFON(X)	; Aktivování On-line korekce délky nástroje pro směr X.
N30 WHEN TRUE DO $AA_TOFF[X]=10 G4 F5	; Pro rozměr nástroje ve směru osy X se interpoluje korekce délky nástroje 10.
...	
N80 TOFFOF(X)	; Offset polohy ve směru X nástroje se vymaže. ...$AA_TOFF[X]=0 Žádná osa se nebude pohybovat. K aktuální poloze ve WCS se připočítá offset polohy v souladu s momentální orientací.

Další informace

Příprava bloku

Při přípravě bloku v předběžném zpracování jsou v rámci hlavního zpracování započítávány také momentálně platné offsety délky nástroje. Aby bylo možné v maximální možné míře využívat nejvyšší možnou rychlost pohybu os, je zapotřebí v době, kdy se započítávají offsety nástroje, přerušit přípravu bloku pomocí zastavení předběžného zpracování příkazem `STOPRE`.

Offset nástroje je v okamžiku předběžného zpracování znám také vždy tehdy, pokud se korekce délky nástroje po spuštění programu už nemění nebo pokud bylo po změně korekce délky nástroje zpracováno více bloků, než kolik jich může být uloženo do vyrovnávací paměti IPO mezi předběžným a hlavním zpracováním.

Proměnná $AA_TOFF_PREP_DIFF

Hodnota rozdílu mezi korekcí, která je momentálně v platnosti v interpolátoru, a korekcí, která byla v platnosti v okamžiku přípravy bloku, může být zjištěna v proměnné `$AA_TOFF_PREP_DIFF[<n>]` .

Nastavení strojních parametrů a nastavovaných parametrů

Pro On-line korekci délky nástroje máte k dispozici následující systémové proměnné:

- MD20610 $MC_ADD_MOVE_ACCEL_RESERVE (rezerva zrychlení pro superponovaný pohyb)
- MD21190 $MC_TOFF_MODE

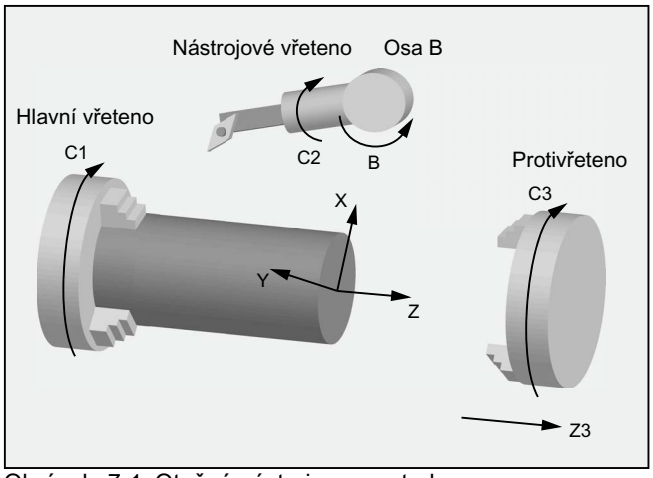
  Na obsah systémové proměnné `$AA_TOFF[<n>]` se buď najede jako na absolutní hodnotu nebo se tato hodnota integruje do výpočtů.
- MD21194 $MC_TOFF_VELO (rychlost on-line korekce délky nástroje)
- MD21196 $MC_TOFF_ACCEL (zrychlení on-line korekce délky nástroje)
- Nastavovaný parametr pro zadání mezních hodnot:
  SD42970 $SC_TOFF_LIMIT (horní mezní hodnota korekce délky nástroje)

Literatura:
Příručka Popis funkcí, Speciální funkce; Transformace ve více osách (F2)

7.11 Editace parametrů břitu u otočných nástrojů (CUTMOD)

Funkce

Pomocí funkce "Editace parametrů břitu u otočných nástrojů" mohou být v korekcích nástroje zohledněny měněné geometrické poměry, které vyplývají při otočení nástroje (především u soustružnických nástrojů, ale i u vrtacích a frézovacích nástrojů) vzhledem k obráběnému obrobku.



Obrázek. 7-1 Otočné nástroje u soustruhu

Momentální otočení nástroje se přitom vždy zjišťuje z polohy aktuálního aktivního orientovatelného držáku nástroje (viz "Délková korekce nástroje pro orientovatelný držák nástroje (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) [Strana 449]").

Funkce se aktivuje pomocí příkazu `CUTMOD`.

Syntaxe

```
CUTMOD=<hodnota>
```

Význam

`CUTMOD` Příkaz pro aktivování funkce "Editace parametrů břitu u otočných nástrojů"

`<hodnota>` Příkazu `CUTMOD` mohou být přiřazeny následující hodnoty:

0 Funkce je deaktivována..
Hodnoty poskytované systémovými proměnnými $P_AD... se rovnají odpovídajícím parametrům nástroje.

> 0 Funkce se aktivuje, jestliže je orientovatelný držák nástroje s uvedeným číslem aktivní, tzn. aktivování je vázáno na určitý orientovatelný držák nástroje.
Hodnoty poskytované systémovými proměnnými $P_AD... jsou oproti odpovídajícím parametrům nástroje v případě potřeby modifikovány v závislosti na aktivním otočení.
Deaktivováním uvedeného orientovatelného držáku nástroje se funkce dočasně deaktivuje, aktivováním jiného orientovatelného držáku nástroje se deaktivuje trvale. V prvním případě se proto funkce při opětovném vyvolání stejného orientovatelného držáku nástroje znovu aktivuje, ve druhém případě je nové vyvolání nezbytné, a to i tehdy, když se orientovatelný držák nástroje s příslušným číslem znovu aktivuje později.
Reset systému nemá na funkci žádný vliv.

-1 Funkce je aktivována vždy, když je aktivní orientovatelný držák nástroje.
Při výměně držáku nástroje nebo při jeho deaktivování a pozdějším opětovném aktivování nemusí být funkce `CUTMOD` znovu nastavována.

-2 Funkce je aktivována vždy, když je aktivován orientovatelný držák nástroje, jehož číslo se rovná momentálně aktivnímu orientovatelnému držáku nástroje.
Jestliže žádný orientovatelný držák není aktivní, má tato situace stejný význam, jako kdyby bylo nastaveno `CUTMOD=0`. Jestliže je aktivní orientovatelný držák, má tato situace stejný význam, jako kdyby bylo bezprostředně zadáno číslo aktuálního držáku nástroje.

< -2 Hodnoty menší než -2 jsou ignorovány, tzn. v tomto případě je situace stejná, jako kdyby příkaz `CUTMOD` nebyl vůbec naprogramován.
Upozornění:
Tento interval hodnot by se vůbec neměl používat, protože je rezervován pro eventuální pozdější rozšíření.

Poznámka

SD42984 $SC_CUTDIRMOD

Funkce, která může být aktivována příkazem `CUTMOD`, nahrazuje funkci, která může být aktivována pomocí nastavovaného parametru SD42984 $SC_CUTDIRMOD. Tato funkce však zůstává i nadále k dispozici nezměněna. Protože ale není rozumné využívat souběžně obě funkce, může být tato funkce aktivována jen tehdy, pokud je `CUTMOD` roven nule.

Příklad

Následující příklad se vztahuje na nástroj s polohou břitu 3 a na orientovatelný držák nástroje, který může nástroj otáčet okolo osy B.

Numerické hodnoty v komentářích uvádějí vždy pozici na konci bloku v souřadném systému stroje (MCS) v posloupnosti X, Y, Z.

```
N10 $TC_DP1[1,1]=500
N20 $TC_DP2[1,1]=3                                          ; Poloha břitu
N30 $TC_DP3[1,1]=12
N40 $TC_DP4[1,1]=1
N50 $TC_DP6[1,1]=6
N60 $TC_DP10[1,1]=110                                       ; Úhel držáku
N70 $TC_DP11[1,1]=3                                         ; Směr břitu
N80 $TC_DP24[1,1]=25                                        ; Úhel volného řezání

N90 $TC_CARR7[2]=0 $TC_CARR8[2]=1 $TC_CARR9[2]=0            ; Osa B
N100 $TC_CARR10[2]=0 $TC_CARR11[2]=0 $TC_CARR12[2]=1        ; Osa C
N110 $TC_CARR13[2]=0
N120 $TC_CARR14[2]=0
N130 $TC_CARR21[2]=X
N140 $TC_CARR22[2]=X
N150 $TC_CARR23[2]="M"

N160 TCOABS CUTMOD=0
N170 G18 T1 D1 TCARR=2                                        X           Y           Z
N180 X0 Y0 Z0 F10000                                        ; 12.000      0.000       1.000

N190 $TC_CARR13[2]=30
N200 TCARR=2
N210 X0 Y0 Z0                                               ; 10.892      0.000       -5.134
N220 G42 Z-10                                               ; 8.696       0.000       -17.330
N230 Z-20                                                   ; 8.696       0.000       -21.330
N240 X10                                                    ; 12.696      0.000       -21.330
N250 G40 X20 Z0                                             ; 30.892      0.000       -5.134

N260 CUTMOD=2 X0 Y0 Z0                                      ; 8.696       0.000       -7.330
N270 G42 Z-10                                               ; 8.696       0.000       -17.330
N280 Z-20                                                   ; 8.696       0.000       -21.330
N290 X10                                                    ; 12.696      0.000       -21.330
N300 G40 X20 Z0                                             ; 28.696      0.000       -7.330

N310 M30
```

Vysvětlení:

V bloku N180 je napřed aktivován nástroj s nastavením CUTMOD=0 a neotočeným orientovatelným držákem nástroje. Protože všechny offsetové vektory orientovatelného držáku nástroje jsou 0, najede se na pozici, která odpovídá délkám nástroje uloženým v parametrech $TC_DP3[1,1] a $TC_DP4[1,1].

V bloku N200 se aktivuje orientovatelný držák nástroje s otočením o 30° okolo osy B. Protože poloha břitu kvůli nastavení CUTMOD=0 není změněna, je stejná jako před starým vztažným bodem břitu. Z tohoto důvodu se v bloku N210 najíždí na pozici, která v počátku (nule) obsahuje starý vztažný bod břitu (tzn. vektor (1, 12) je v rovině Z/X otočen o 30°).

V bloku N260 je narozdíl od bloku N200 aktivní příkaz CUTMOD=2. V důsledku otočení orientovatelného držáku nástroje se stává platnou změněná poloha břitu 8. Z toho vyplývají také odlišné polohy os.

V blocích N220, příp. N270 je aktivována korekce rádiusu nástroje. Odlišné polohy břitu v obou úsecích programu nemají na koncovou pozici v blocích, v nichž je korekce rádiusu nástroje aktivována, žádný vliv, odpovídající polohy jsou proto identické. Teprve po blocích s deaktivováním N260, příp. N300 se odlišné polohy břitu opět projeví.

Další informace

Platnost změněných parametrů břitu

Změněná poloha břitu a změněný vztažný bod břitu jsou při programování okamžitě v platnosti i pro již aktivní nástroj. Nové aktivování nástroje není proto nutné.

Vliv na aktivní pracovní rovinu

Pro stanovení změněné polohy břitu, směru řezání a úhlu držáku, příp. úhlu volného řezání je určující sledování břitu v právě aktivní rovině (G17 - G19).

Jestliže však nastavovaný parametr SD42940 $SC_TOOL_LENGTH_CONST (změna složek délky nástroje v případě změny roviny) obsahuje platnou hodnotu nerovnající se nule (plus nebo mínus 17, 18 nebo 19), potom určuje její obsah rovinu, ve které jsou příslušné veličiny započítány.

Systémové proměnné

Jsou Vám k dispozici následující systémové proměnné:

Systémové proměnné	Význam
$P_CUTMOD_ANG / $AC_CUTMOD_ANG	Výsledkem je (nezaokrouhlený) úhel v aktivní rovině obrábění, který byl brán za základ pro modifikace dat břitu (poloha břitu, směr řezání, úhel volného řezání a úhel držáku) při funkcích aktivovaných pomocí příkazu CUTMOD, příp. $SC_CUTDIRMOD. Proměnná $P_CUTMOD_ANG se vztahuje na aktuální stav při předběžném zpracování, proměnná $AC_CUTMOD_ANG na aktuální blok při hlavním zpracování.
$P_CUTMOD / $AC_CUTMOD	Načítá momentální platnou hodnotu, která byla naposled naprogramována pomocí příkazu CUTMOD (číslo držáku nástroje, pro který má být aktivována úprava parametrů břitu). Jestliže byla naposled naprogramovaná hodnota pro funkci CUTMOD -2 (aktivování s momentálně aktivním orientovatelným držákem nástroje), potom parametr $P_CUTMOD neposkytuje hodnotu -2, ale číslo orientovatelného držáku nástroje, který byl v okamžiku naprogramování aktivní. Proměnná $P_CUTMOD se vztahuje na aktuální stav při předběžném zpracování, proměnná $AC_CUTMOD na aktuální blok při hlavním zpracování.
$P_CUT_INV / $AC_CUT_INV	Pokud je nástroj otočený tak, že směr otáčení vřetena musí být invertován, je výslednou hodnotou TRUE. Za tím účelem musí být v bloku, na který se příslušná operace čtení vztahuje, splněny následující podmínky: 1. Je aktivní soustružnický nebo brusný nástroj (typy nástrojů 400 až 599 a / nebo SD42950 $SC_TOOL_LENGTH_TYPE = 2). 2. Pomocí příkazu NC jazyka CUTMOD bylo aktivováno ovlivňování břitu. 3. Jestliže je aktivní orientovatelný držák nástroje, který byl určen numerickou hodnotou v příkazu CUTMOD. 4. Orientovatelný držák nástroje otáčí nástroj okolo osy v pracovní rovině (za obvyklých okolností je to osa C) tak, aby výsledná normála břitu nástroje byla oproti výchozí poloze otočena o více než 90° (za obvyklých okolností 180°). Pokud alespoň jedna z výše uvedených čtyř podmínek není splněna, obsahuje proměnná hodnotu FALSE. Pro nástroje, jejichž poloha břitu není definována, obsahuje tato proměnná vždy hodnotu FALSE. Proměnná $P_CUT_INV se vztahuje na aktuální stav při předběžném zpracování a proměnná $P_CUT_INV na aktuální blok při hlavním zpracování.

Všechny proměnné hlavního zpracování ($AC_CUTMOD_ANG, $AC_CUTMOD a $AC_CUT_INV) mohou být čteny v synchronních akcích. Přístup kvůli čtení z předběžného zpracování má za následek zastavení předběžného zpracování.

Upravené nastavované parametry:

Jestliže je aktivní otočení nástroje, jsou změněná data k dispozici v následujících systémových proměnných:

Systémové proměnné	Význam
$P_AD[2]	Poloha břitu
$P_AD[10]	Úhel držáku
$P_AD[11]	Směr břitu
$P_AD[24]	Úhel volného řezání

Poznámka

Tyto údaje jsou oproti odpovídajícím parametrům nástroje ($TC_DP2[..., ...] atd.) upravovány vždy tehdy, když byla pomocí příkazu `CUTMOD` aktivována funkce "Editace parametrů břitu u otočných nástrojů" a když je aktivní orientovatelný držák nástroje, který způsobuje otočení nástroje.

Literatura

Pokud budete potřebovat další informace týkající se funkce "Editace parametrů břitu u otočných nástrojů", viz:

Příručka Popis funkcí, Základní funkce; Korekční parametry nástrojů (W1)

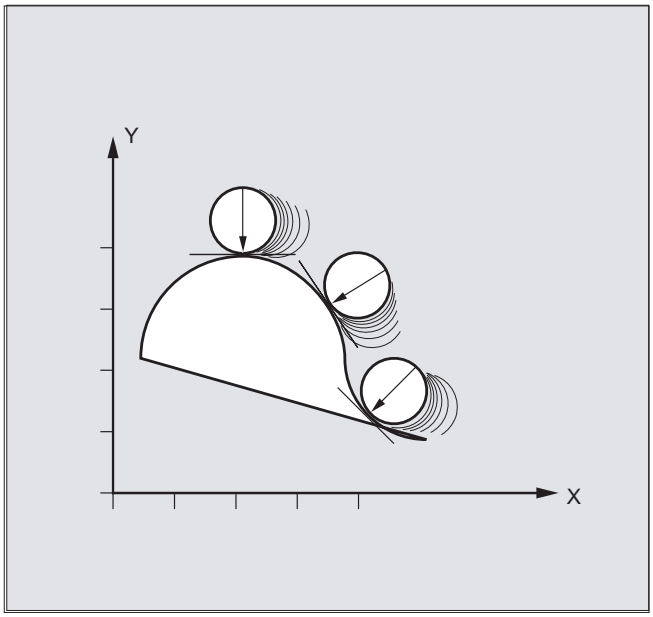
Chování při pohybu po dráze

8

8.1 Tangenciální řízení (TANG, TANGON, TANGOF, TLIFT, TANGDEL)

Funkce

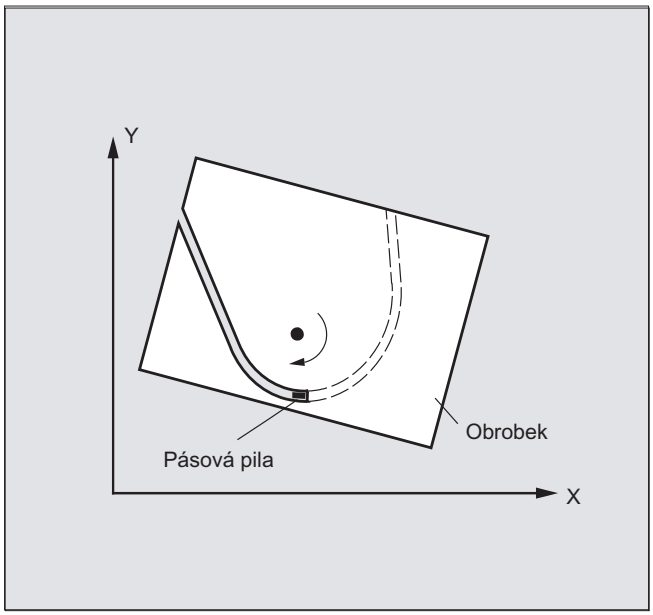
Vlečná osa je naváděna podle tečny k dráze, která je definována řídícími osami. Tímto způsobem může být nástroj nastavován rovnoběžně s konturou. Prostřednictvím úhlu naprogramovaného v příkazu `TANGON` může být nástroj nastavován vzhledem k tečně.



Aplikace

Tangenciální řízení může být použito např. v těchto případech:

- Tangenciální nastavování otočných nástrojů při prostřihování
- Souběžné nastavování nasměrování obrobku při použití pásové pily (viz následující obrázek)
- Nastavování orovnávacího nástroje na brusný kotouč
- Nastavování polohy obvodových nožů při zpracovávání skla nebo papíru
- Tangenciální přivádění drátu při svařování s 5 osami



Syntaxe

Definice tangenciálního vlečení:
```
TANG(<vlečná osa>,<řídící osa 1>,<řídící osa 2>,<faktor
vazby>,<KS>,<Opt>)
```

Aktivování tangenciálního řízení:
```
TANGON(<vlečná osa>,<úhel>,<vzdálenost>,<úhlová tolerance>)
```

Deaktivování tangenciálního řízení:
```
TANGOF(<vlečná osa>)
```

Aktivování funkce "Vložení pomocného bloku v rozích kontury":
```
TLIFT(<vlečná osa>)
```

Příkaz `TLIFT` se zadává v návaznosti na přiřazení os pomocí příkazu `TANG(…)`.

Deaktivování funkce "Vložení pomocného bloku v rozích kontury":

Opakování příkazu `TANG(...)` bez navazujícího příkazu `TLIFT(<vlečná osa>)`.

Vymazání definice tangenciálního vlečení:
```
TANGDEL(<vlečná osa>)
```

Jestliže má být definováno nové tangenciální vlečení se stejnou vlečnou osou v příkazu pro přípravné vyvolání `TANG`, musí být již existující uživatelem definované tangenciální vlečení vymazáno. Vymazání je možné jedině v případě, že vazba je pomocí příkazu `TANGOF(<vlečná osa>)` deaktivována.

Význam

`TANG`:	Přípravný příkaz pro definici tangenciálního vlečení
`TANGON`:	Aktivování tangenciálního vlečení pro uvedenou vlečnou osu.

TANGOF:	Deaktivování tangenciálního vlečení pro uvedenou vlečnou osu.
TLIFT:	Aktivování funkce "Vložení pomocného bloku v rozích kontury"
TANGDEL:	Vymazání definice tangenciálního vlečení
<vlečná osa>:	Vlečná osa: Tangenciálně vlečená doplňková osa
<řídící osa 1>, <řídící osa 2>:	Řídící os: Dráhové osy, na základě kterých se stanoví tečna pro vlečení.
<faktor vazby>:	Faktor vazby: Funkční závislost mezi úhlovou změnou tečny a vlečenou osou
	Předdefinovaná nastavení: 1
	Upozornění: Faktor vazby 1 nemusí být explicitně naprogramován.
<KS>:	Identifikační písmeno pro souřadný systém
	"B": Základní souřadný systém (předdefinované nastavení) **Upozornění:** <KS> = "B" nemusí být explicitně naprogramováno.
	"W": Souřadný systém obrobku (není k dispozici)
<Opt>:	Optimalizace
	"S": Standardní (předdefinované nastavení) **Upozornění:** <Opt> = "S" nemusí být explicitně naprogramováno.
	"P": Automatické přizpůsobení časovému průběhu tangenciální osy a kontury **Upozornění:** Jestliže je nastaveno <Opt> = "P", je na základě omezení rychlosti řídících os zohledňována také dynamika vlečné osy. Toto nastavení je možno doporučit zejména při použití kinematických transformací.
<úhel>:	Úhlový offset vlečné osy
<vzdálenost>:	Dráha zaoblení pro vlečnou osu (zapotřebí při nastavení <Opt> = "P")
<úhlová tolerance>:	Úhlová tolerance pro vlečnou osu (není zapotřebí, vyhodnocování se uskutečňuje pouze při nastavení <Opt> = "P")
	Upozornění: Parametry <vzdálenost> a <úhlová tolerance> cíleně omezují chybu mezi vlečenou osou a tečnou řídících os.

Příklady

Příklad 1: Definice a aktivování tangenciálního vlečení

```
Programový kód                    Komentář
N10 TANG(C,X,Y,1,"B","P")         ; Definice tangenciálního vlečení: Kruhová osa C
                                    má sledovat pohyb geometrických os X a Y.
N20 TANGON(C,90)                  ; Osa C je vlečnou osou. Při každém pohybu
                                    dráhových os se má otočit do polohy 90° vůči
                                    tečně ke dráze.
...
```

Poznámka

Zjednodušené programování

`TANG(C,X,Y,1,"B","P")` může být zjednodušeně naprogramováno jako `TANG(C,X,Y,,,"P")`.

Příklad 2: Změna roviny

```
Programový kód                    Komentář
N10 TANG(A,X,Y,1)                 ; 1. definice tangenciálního vlečení.
N20 TANGON(A)                     ; Aktivování vazby.
N30 X10 Y20                       ; Rádius
...
N80 TANGOF(A)                     ; Deaktivování 1. vazby.
N90 TANGDEL(A)                    ; Vymazání 1. definice.
...
TANG(A,X,Z)                       ; 2. definice tangenciálního vlečení.
TANGON(A)                         ; Aktivování nové vazby.
...
N200 M30
```

Příklad 3: Přepnutí geometrických os a příkaz TANGDEL

Nebude aktivován žádný alarm.

```
Programový kód                    Komentář
N10 GEOAX(2,Y1)                   ; Y1 je geometrická osa 2.
N20 TANG(A,X,Y)                   ; 1. definice tangenciálního vlečení.
N30 TANGON(A,90)                  ; Aktivování vlečení s osou Y1.
N40 G2 F8000 X0 Y0 I0 J50
N50 TANGOF(A)                     ; Deaktivování vlečení s osou Y1.
N60 TANGDEL(A)                    ; Vymazání 1. definice.
N70 GEOAX(2,Y2)                   ; Y2 je nová geometrická osa 2.
N80 TANG(A,X,Y)                   ; 2. definice tangenciálního vlečení.
N90 TANGON(A,90)                  ; Aktivování vlečení s osou Y2.
...
```

Příklad 4: Tangenciální vlečení s automatickou optimalizací

Y1 je geometrická osa 2.

```
Programový kód                                  Komentář
...
N80 G0 C0
N100 F=50000
N110 G1 X1000 Y500
N120 TRAORI
N130 G642                                       ; Přechodová zaoblení při dodržení
                                                  maximální přípustné odchylky od
                                                  dráhy.
N171 TRANS X50 Y50
N180 TANG(C,X,Y,1,,"P")                         ; Definice tangenciálního vlečení s
                                                  automatickou optimalizací
                                                  rychlosti pohybu po dráze.
N190 TANGON(C,0,5.0,2.0)                        ; Aktivování tangenciálního vlečení
                                                  s automatickou optimalizací: Dráha
                                                  přechodového zaoblení 5 mm, úhlová
                                                  tolerance 2 stupně.
N210 G1 X1310 Y500
N215 G1 X1420 Y500
N220 G3 X1500 Y580 I=AC(1420) J=AC(580)
N230 G1 X1500 Y760
N240 G3 X1360 Y900 I=AC(1360) J=AC(760)
N250 G1 X1000 Y900
N280 TANGOF(C)
N290 TRAFOOF
N300 M02
```

Další informace

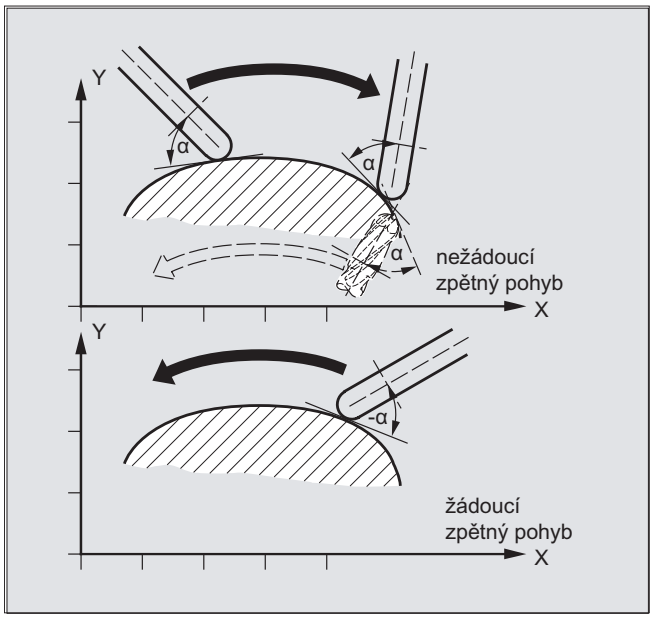
Definice vlečné a řídící osy

Definice vlečné a řídící osy se uskutečňuje pomocí příkazu `TANG`.

Faktor vazby udává souvislost mezi změnou úhlu tečny a vlečenou osou. Jeho hodnota je zpravidla 1 (předdefinovaná hodnota).

Mezní úhel v důsledku ohraničení pracovního pole

Při pohybech po dráze sem a tam provádí tečna skokovou změnu v bodě obratu dráhy o 180° a odpovídajícím způsobem se mění také nasměrování vlečné osy. Toto chování je zpravidla nesmyslné. Zpětný pohyb se má uskutečňovat se stejnou zápornou hodnotou úhlového offsetu jako při pohybu tam.



Z tím účelem musí být pracovní pole vlečné osy omezeno (`G25`, `G26`). Ohraničení pracovního pole musí být aktivováno v okamžiku dosažení bodu obratu na dráze (`WALIMON`). Jestliže leží hodnota úhlového offsetu mimo ohraničení pracovního pole, systém se pokusí dostat se zpátky do přípustné pracovní oblasti se zápornou hodnotou úhlového offsetu.

Vložení pomocného bloku v rozích kontury (TLIFT)

V rohu kontury se tečna a tím pádem také požadovaná poloha vlečné osy skokově mění. Za normálních okolností se osa snaží tento skok vyrovnat svou maximální možnou rychlostí. Přitom ale na určitém úseku dráhy za rohem kontury vzniká odchylka od požadovaného tangenciálního nastavení. Jestliže taková odchylka není z technologických důvodů tolerovatelná, je možné pomocí příkazu `TLIFT` přinutit řídící systém, aby na rohu zpracování pozastavil, aby se v tomto automaticky vytvořeném vloženém bloku otočila vlečená osa do směru nové tečny.

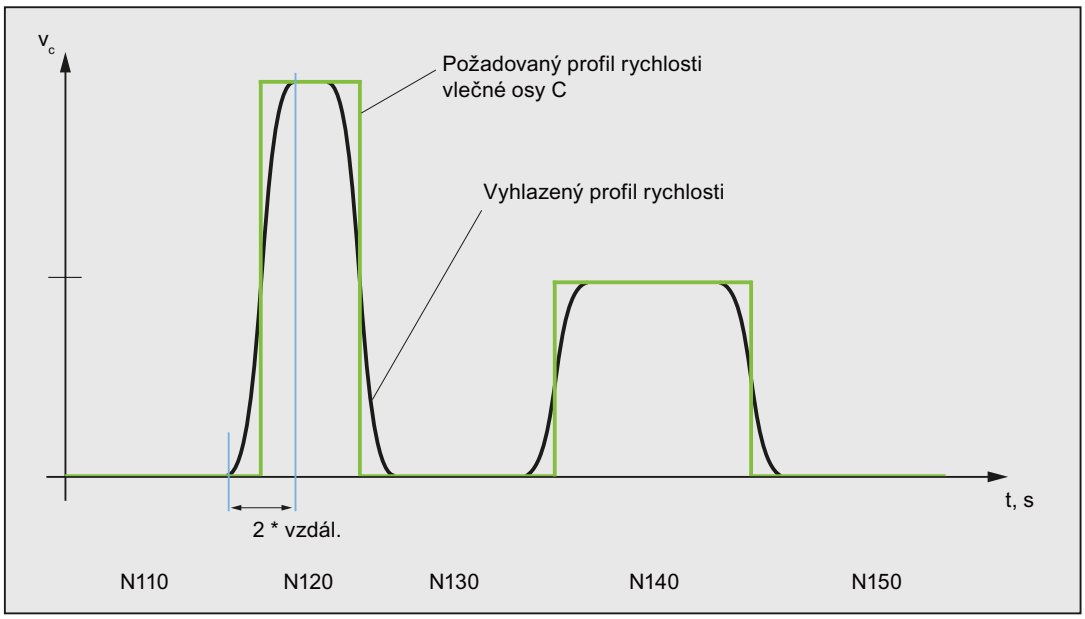
Pokud se vlečená osa už aspoň jedenkrát pohybovala jako dráhová osa, otočení se uskutečňuje s naprogramovanou dráhovou osou. Prostřednictvím funkce `TFGREF[<osa>]=0.001` může být při této operaci dosaženo maximální rychlosti vlečené osy.

Pokud se vlečená osa až dosud jako dráhová osa vůbec nepohybovala, bude tato osa ovládána jako polohovací osa. Rychlost je potom závislá na rychlosti polohování, která je nastavena ve strojním parametru.

Otočení se uskutečňuje s maximální rychlostí vlečné osy.

Možnost optimalizace

Je-li aktivována automatická optimalizace (<Opt>="P") a jsou-li pro vlečnou osu zadány parametry dráhy přechodového zaoblení (<vzdálenost>) a úhlová tolerance (<úhlová tolerance>), potom jsou při tangenciálním vlečení skokové změny rychlosti vlečné osy v důsledku nespojitostí na řídící kontuře zaoblovány, příp. vyhlazovány Přitom je průběh pohybu vlečné osy předvídán (viz obrázek), aby se vznikající odchylka udržela co možno nejmenší.



Definice změny úhlu

Úhlová změna, od které je automatický pomocný blok vkládán, je definována pomocí následujícího strojního parametru:

MD37400 $MA_EPS_TLIFT_TANG_STEP (úhel tečny pro rozpoznání rohu)

Vliv na transformace

Poloha vlečené kruhové osy může být vstupní hodnotou pro transformaci.

Explicitní polohování vlečné osy

Jestliže je svými řídícími osami ovládaná vlečná osa explicitně polohována, potom se tento údaj polohy uplatní aditivně k naprogramovanému úhlovému offsetu.

Všechny způsoby zadání dráhy jsou přípustné (pohyby dráhových a polohovacích os).

Stav vazby

V NC výrobním programu může být stav vazby zjišťován prostřednictvím systémových proměnných $AA_COUP_ACT[<osa>].

Hodnota	Význam
0	Žádná vazba není aktivní
1,2,3	Tangenciální vlečení je aktivní

8.2 Průběh charakteristiky posuvu (FNORM, FLIN, FCUB, FPO)

Funkce

Pro účely pružnějšího zadávání průběhu posuvu bylo jeho programování podle normy DIN 66025 rozšířeno o lineární a kubické průběhy.

Kubické průběhy mohou být naprogramovány buď přímo nebo jako interpolační spliny. Tak je možné naprogramovat - v závislosti na zakřivení opracovávaného obrobku - spojité a hladké funkční průběhy rychlosti.

Tyto funkční průběhy rychlosti umožňují změny zrychlení bez trhavých pohybů a tím tedy i výrobu stejnoměrných povrchů obrobku.

Syntaxe

```
F… FNORM
F… FLIN
F… FCUB
F=FPO(…,…,…)
```

Význam

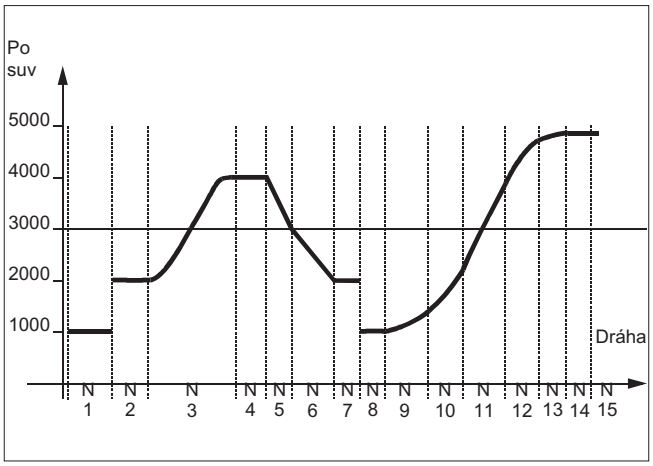
`FNORM`	Základní nastavení. Hodnota posuvu se zadává prostřednictvím dráhového úseku daného bloku a potom platí jako modální hodnota.
`FLIN`	Profil rychlosti pohybu po dráze je **lineární:** Hodnota posuvu se z hodnoty aktuální na začátku bloku do hodnoty na konci bloku po celém úseku dráhy lineárně mění a potom platí jako modální hodnota. Toto chování může být kombinováno s příkazy G93 a G94.
`FCUB`	Profil rychlosti pohybu po dráze je **kubický:** Blokově naprogramované hodnoty F - vztažené na konec bloku - jsou spojovány spliny. Spline začíná a končí s tangenciálním napojením na předešlý, resp.následující údaj posuvu a uplatňuje se s příkazy G93 a G94. Pokud v nějakém bloku F-adresa chybí, použije se pro tento účel naposled naprogramované F-slovo.
`F=FPO…`	Profil rychlosti pohybu po dráze je zadán **pomocí polynomu:** F-adresa popisuje průběh posuvu od aktuální hodnoty do hodnoty na konci bloku pomocí polynomu. Koncová hodnota pak platí jako modální.

Optimalizace posuvu v případě zakřivených úseků dráhy

Pro pohyby s posuvem daným polynomem `F=FPO` a splinem `FCUB` by vždy měla být nastavena konstantní řezná rychlost `CFC`. Tímto způsobem je možné vytvořit profil požadované hodnoty posuvu se spojitým zrychlením.

Příklad: Různé profily hodnoty posuvu

V tomto příkladu naleznete programové příkazy a grafické zobrazení různých profilů hodnoty posuvu.

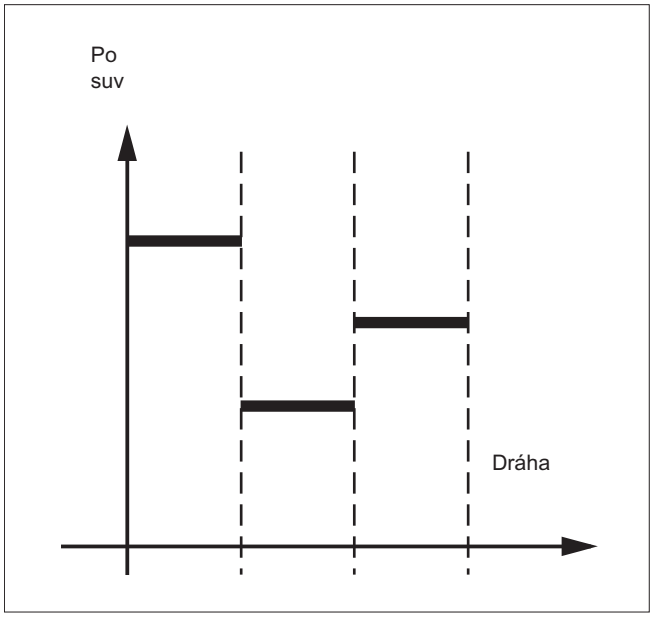


Programový kód	**Komentář**
N1 F1000 FNORM G1 X8 G91 G64	; Konstantní profil hodnoty posuvu, zadávání inkrementálních rozměrů
N2 F2000 X7	; Skoková změna požadované hodnoty rychlosti
N3 F=FPO(4000, 6000, -4000)	; Profil hodnoty posuvu zadaný pomocí polynomu s hodnotou posuvu 4000 na konci bloku
N4 X6	; Posuv 4000 daný polynomem platí jako modální hodnota
N5 F3000 FLIN X5	; Lineární profil posuvu
N6 F2000 X8	; Lineární profil posuvu
N7 X5	Lineární posuv platí jako modální hodnota
N8 F1000 FNORM X5	; Konstantní profil hodnoty posuvu se skokovou změnou zrychlení
N9 F1400 FCUB X8	; Všechny následující blokově naprogramované hodnoty F budou pospojovány spliny
N10 F2200 X6	
N11 F3900 X7	
N12 F4600 X7	
N13 F4900 X5	; Deaktivování splinového profilu
N14 FNORM X5	
N15 X20	

FNORM

Podle normy DIN 66025 označuje adresa posuvu F posuv po dráze jako konstantní hodnotu.

Více informací o této problematice naleznete v Příručce programování, "Základy".

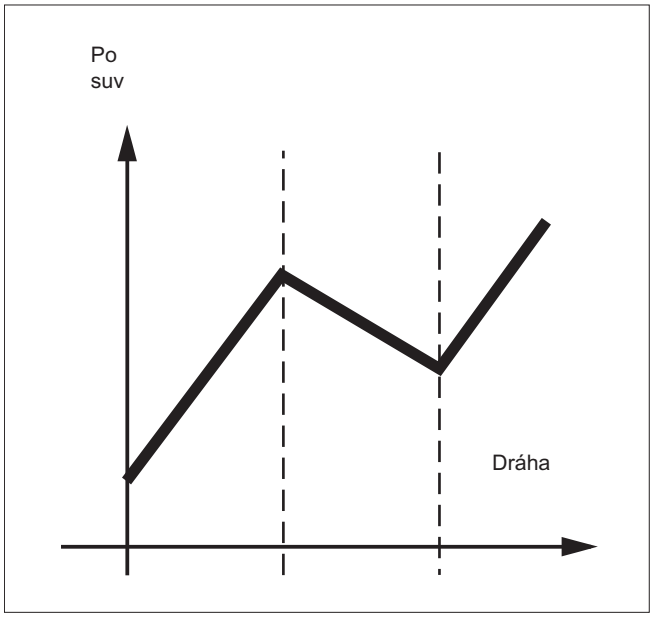


FLIN

Hodnota posuvu se bude lineárně měnit od momentální hodnoty posuvu až do naprogramovaného F-slova, přičemž této hodnoty bude dosaženo na konci bloku.

Příklad:

```
N30 F1400 FLIN X50
```

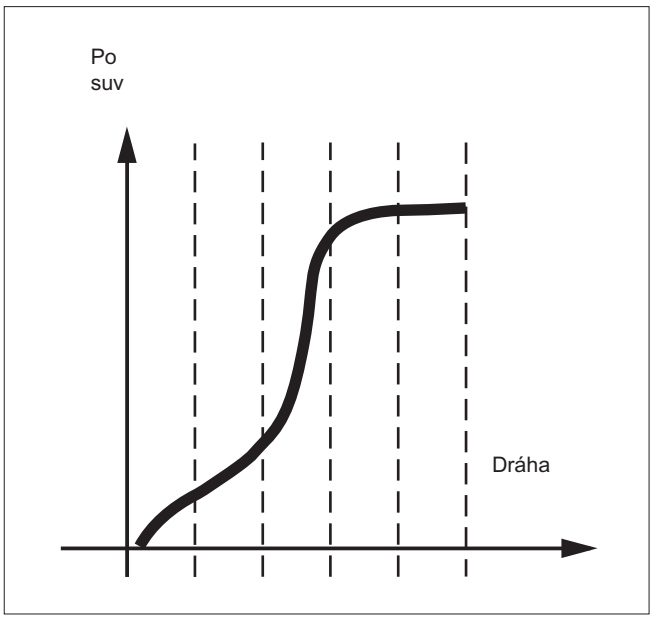


FCUB

Hodnota posuvu se bude měnit od momentální hodnoty posuvu až do naprogramovaného F-slova podle kubické funkce, přičemž této hodnoty bude dosaženo na konci bloku. Řídící systém pomocí splinových funkcí pospojuje všechny hodnoty posuvu naprogramované pomocí příkazu FCUB s blokovou platností. Hodnoty posuvu přitom slouží jako uzlové body pro výpočet splinové interpolace.

Příklad:

```
N50 F1400 FCUB X50
N60 F2000 X47
N70 F3800 X52
```



F=FPO(…,…,…)

Průběh hodnoty posuvu je přímo naprogramován pomocí polynomu. Zadávání koeficientů polynomu se uskutečňuje podobně jako při polynomické interpolaci.

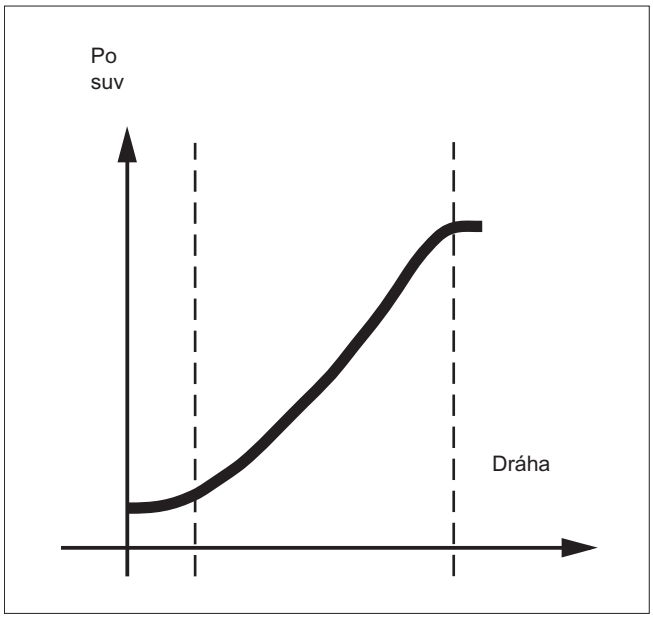
Příklad:

```
F=FPO(endfeed, quadf, cubf)
```

Parametry endfeed, quadf a cubf jsou dříve definované proměnné.

endfeed:	Posuv na konci bloku.
quadf:	Kvadratický koeficient polynomu
cubf:	Kubický koeficient polynomu

Když je aktivní příkaz `FCUB`, napojují se spliny na začátku bloku a na konci bloku tangenciálně na průběh zadaný pomocí funkce `FPO`.



Okrajové podmínky

Nezávisle na naprogramovaném průběhu posuvu platí funkce pro programování chování při pohybu po dráze.

Naprogramovaný průběh posuvu má v zásadě absolutní platnost - nezávisle na příkazech `G90` nebo `G91`.

Průběh hodnoty posuvu FLIN a FCUB se uplatňuje při příkazech

`G93` a `G94`.

Funkce `FLIN` a `FCUB` **se neuplatňují** při příkazech

`G95`, `G96/G961` a `G97/G971`.

Aktivní kompresor COMPON

Když je aktivní kompresor `COMPON`, platí při spojování většího počtu bloků do jednoho splinového úseku následující:

FNORM:

Pro splinový segment platí F-slovo z posledního bloku, který do tohoto segmentu patří.

FLIN:

Pro splinový segment platí F-slovo z posledního bloku, který do tohoto segmentu patří. Naprogramovaná hodnota F platí na konci segmentu a bude jí dosaženo pomocí lineární funkce.

FCUB:

Vytvořený spline posuvu se od naprogramované hodnoty v koncovém bodu odchyluje maximálně o hodnotu definovanou ve strojním parametru `C $MC_COMPRESS_VELO_TOL`.

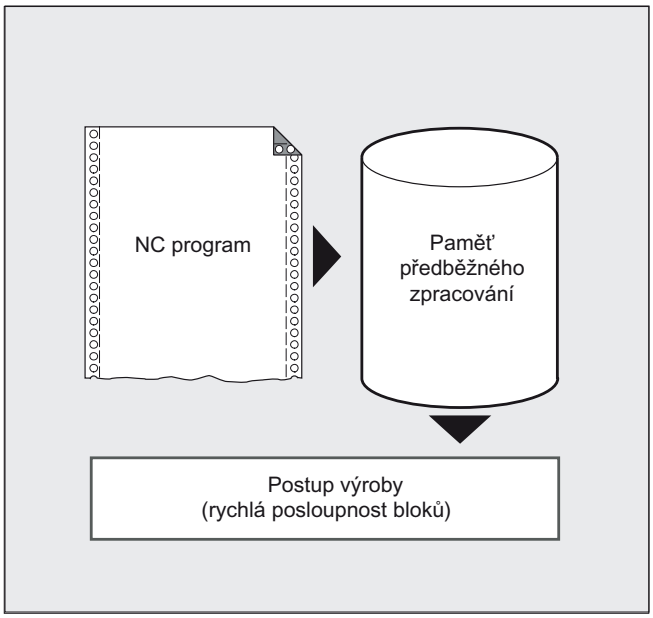
F=FPO(...,...,...)

Tyto bloky nejsou nijak komprimovány.

8.3 Zpracování programu s pamětí předběžného zpracování (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE)

Funkce

V závislosti na úrovni svého vybavení disponuje řídící systém určitým množstvím tzv. paměti pro předběžné zpracování, do které se ukládají připravené hotové bloky před svým zpracování a odkud jsou v průběhu samotného obrábění předávány jako rychlá posloupnost bloků. Díky tomu mohou být krátké bloky obrobeny s vysokými rychlostmi. Jestliže to doba prodlevy řídícího systému připouští, je paměť předběžného zpracování v zásadě naplněna.



Označení zpracovávaného úseku

Zpracovávaný úsek programu, který má být dočasně uložen do paměti předběžného zpracování, je ve výrobním programu označen na začátku příkazem `STOPFIFO` a na konci příkazem `STARTFIFO`. Zpracování připravených a do vyrovnávací paměti uložených bloků začíná teprve po příkazu `STARTFIFO` nebo když je paměť předběžného zpracování plná.

Automatické řízení paměti předběžného zpracování

Automatické řízení paměti předběžného zpracování se vyvolává pomocí příkazu `FIFOCTRL`. Příkaz `FIFOCTRL` z počátku funguje úplně stejně jako příkaz `STOPFIFO`. Při každé sekvenci programových příkazů se čeká, dokud není paměť předběžného zpracování plná, a potom se spouští zpracování. Rozdíl mezi oběma možnostmi spočívá v chování v případě vyprázdnění paměti předběžného zpracování: V případě příkazu `FIFOCTRL` se od úrovně 2/3 naplnění postupně snižuje rychlost pohybu po dráze, aby se zabránilo úplnému vyprázdnění a zabrždění pohybu os až do úplného klidu.

Zastavení předběžného zpracování

Když je v bloku naprogramován příkaz STOPRE, příprava bloků a ukládání do vyrovnávací paměti se zastaví. Následující blok bude zpracován teprve tehdy, až jsou zcela zpracovány všechny bloky, které již dříve byly připraveny a uloženy do paměti. Předcházející blok bude zastaven v přesném najetí (jako při G9).

Syntaxe

Tabulka. 8-1 Označení zpracovávaného úseku:

```
STOPFIFO
...
STARTFIFO
```

Tabulka. 8-2 Automatické řízení paměti předběžného zpracování:

```
...
FIFOCTRL
...
```

Tabulka. 8-3 Zastavení předběžného zpracování:

```
...
STOPRE
...
```

Poznámka

Příkazy STOPFIFO, STARTFIFO, FIFOCTRL a STOPRE musí být naprogramovány v samostatném bloku.

Význam

STOPFIFO: Příkaz STOPFIFO označuje začátek zpracovávaného úseku, který má být dočasně uložen do paměti předběžného zpracování. Pomocí příkazu STOPFIFOse zpracovávání pozastaví a paměť předběžného zpracování se bude plnit, dokud:

- není rozpoznán příkaz STARTFIFO nebo STOPRE.

  nebo

- se paměť předběžného zpracování nenaplní

  nebo

- není dosaženo konce programu.

STARTFIFO: Pomocí příkazu STARTFIFO se spouští rychlé zpracování úseku programu; souběžně s tím probíhá plnění paměti předběžného zpracování.

FIFOCTRL: Spuštění automatického ukládání do paměti předběžného zpracovávání

STOPRE: Zastavení předběžného zpracování

Poznámka

Plnění paměti předběžného zpracování se neprovádí, příp. je přerušeno, pokud zpracovávaný úsek programu obsahuje příkazy, které vyžadují okamžité zpracování (najíždění na referenční bod, měřicí funkce, ...).

Poznámka

Při přístupu ke stavovým údajům stroje ($SA...) generuje řídící systém interní zastavení předběžného zpracování.

⚠ POZOR

Když jsou aktivovány korekce nástroje a splinová interpolace, neměl by být naprogramován žádný příkaz `STOPRE`, protože by došlo k přerušení posloupnosti k sobě patřících bloků.

Příklad: Zastavení předběžného zpracování

Programový kód	**Komentář**
...	
N30 MEAW=1 G1 F1000 X100 Y100 Z50	; Blok měření s měřicí sondou na prvním měřicím vstupu a přímková interpolace.
N40 STOPRE	; Zastavení předběžného zpracování.
...	

8.4 Úseky programu s podmíněným přerušením (DELAYFSTON, DELAYFSTOF)

Funkce

Podmíněně přerušitelné úseky výrobních programů se nazývají oblasti odloženého zastavení (Stop-Delay). V rámci těchto určitých úseků programů nemá být zpracování **zastavováno** a také hodnota **posuvu** se nemá měnit. Krátké úseky programu, které slouží např. pro výrobu závitu, mají být v zásadě chráněny před téměř všemi událostmi způsobujícími zastavení. K případnému zastavení dojde až tehdy, až bude příslušný úsek programu zpracován až do konce.

Syntaxe

```
DELAYFSTON
DELAYFSTOF
```

Příkaz vyžaduje samostatný řádek výrobního programu.

Oba příkazy jsou přípustné jen ve výrobních programech, nikoli v synchronních akcích.

Význam

`DELAYFSTON`	Definice začátku oblasti, ve které budou "měkká" zastavení pozdržena do doby, dokud nebude dosaženo konce oblasti odloženého zastavení (Stop-Delay).
`DELAYFSTOF`	Definice konce oblasti odloženého zastavení (Stop_Delay)

Poznámka

Když je ve strojním parametru nastaveno MD11550 $MN_STOP_MODE_MASK Bit 0 = 0 (předdefinované nastavení), je implicitně definována oblast odloženého zastavení (Stop-Delay), kdykoli je naprogramován příkaz `G331/G332` a pohyb po dráze, příp. `G4`.

Příklad: Události způsobující zastavení

V oblasti odloženého zastavení (Stop-Delay) jsou změny hodnoty **posuvu a blokování posuvu** ignorovány. Tyto příkazy se uplatní až po skončení oblasti odloženého zastavení (Stop-Delay).

Události způsobující zastavení se rozlišují následujícím způsobem:

"Měkké" události způsobující zastavení	Reakce: zpožděná
"Tvrdé" události způsobující zastavení	Reakce: okamžitá

Volba události způsobující zastavení, která zajistí co možno nejrychlejší zastavení:

Název události	Reakce	Parametry přerušení
RESET	okamžitá	NST: DB21,… DBX7.7 a DB11, … DBX20.7
PROG_END	Alarm 16954	Program v NC: M30
INTERRUPT	zpožděná	NST: FC-9 a ASUP DB10, ... DBB1
SINGLEBLOCKSTOP	zpožděná	Zahájení režimu zpracování blok po bloku v oblasti odloženého zastavení (Stop_Delay): NC systém se zastaví na konci 1. bloku mimo oblasti odloženého zastavení (Stop-Delay). Režim zpracování blok po bloku byl již aktivován před oblastí odloženého zastavení (Stop-Delay): NST: "Zastavení NC systému na hranici bloku" DB21, ... DBX7.2
STOPPROG	zpožděná	NST: DB21,… DBX7.3 a DB11, … DBX20.5
PROG_STOP	Alarm 16954	Program v NC: M0 a M1
WAITM	Alarm 16954	Program v NC: WAITM
WAITE	Alarm 16954	Program v NC: WAITE
STOP_ALARM	okamžitý	Alarm: Konfigurace alarmu STOPBYALARM
RETREAT_MOVE_THREAD	Alarm 16954	Program v NC: Alarm 16954, když je aktivní příkaz LFON (zastavení a rychlé pozvednutí v příkazu G33 nejsou možné)
WAITMC	Alarm 16954	Program v NC: WAITMC
NEWCONF_PREP_STOP	Alarm 16954	Program v NC: NEWCONF
SYSTEM_SHUTDOWN	okamžitá	Vypnutí systému u 840Di
ESR	zpožděná	Rozšířené zastavování a odjíždění
EXT_ZERO_POINT	zpožděná	Externí posunutí počátku
STOPRUN	Alarm 16955	BTSS: PI "_N_FINDST" STOPRUN

Vysvětlení reakcí:

okamžitá ("tvrdá" událost způsobující zastavení)	K zastavení dojde okamžitě, i v oblasti odloženého zastavení (Stop-Delay)
zpožděná ("měkká" událost způsobující zastavení)	Zastavení (také krátkodobé) se uskuteční až za oblastí odloženého zastavení (Stop-Delay)..
Alarm 16954	Program se přeruší, protože v oblasti odloženého zastavení (Stop-Delay) byly použity nepovolené programové příkazy.
Alarm 16955	Program bude pokračovat, v oblasti odloženého zastavení (Stop-Delay) se uskutečnila nepovolená akce.
Alarm 16957	Úsek programu (oblast odloženého zastavení (Stop-Delay)), která je uzavřena příkazy DELAYFSTON a DELAYFSTO, nemohla být aktivována. Díky tomu se každý příkaz STOP v oblasti uskuteční ihned a bez zpoždění.

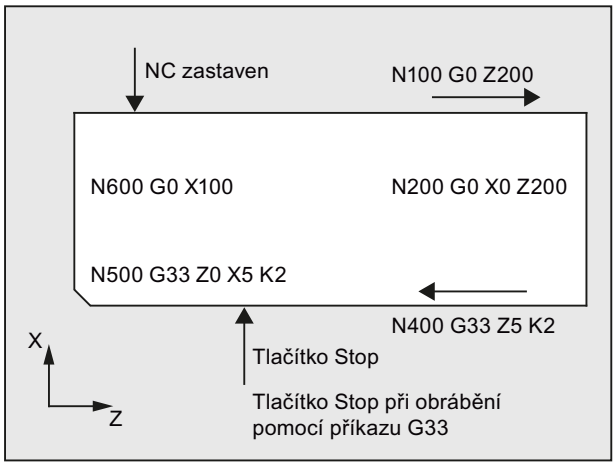
Pokud budete hledat shrnutí dalších reakcí na událost způsobující zastavení, viz:
Literatura:
Příručka Popis funkcí, Základní funkce; BAG, kanál, zpracování programu (K1), kapitola "Ovlivňování a důsledky události způsobující zastavení"

Příklad: Vnoření oblastí odloženého zastavení (Stop-Delay) na dvou programových úrovních

```
Programový kód          Komentář
N10010 DELAYFSTON()     ; Bloky s čísly N10xxx programu 1.
N10020 R1 = R1 + 1
N10030 G4 F1            ; Začátek oblasti odloženého zastavení (Stop-Delay).
...
N10040 Podprogram 2
...
...                     ; Interpretace podprogramu 2.
N20010 DELAYFSTON()     ; Žádná platnosti, opakovaný začátek, 2. rovina.
...
N20020 DELAYFSTOF()     ; Žádná platnost, konec v jiné rovině.
N20030 RET
N10050 DELAYFSTOF()     ; Konec oblasti odloženého zastavení (Stop-Delay) ve stejné
                          rovině.
...
N10060 R2 = R2 + 2
N10070 G4 F1            ; Oblasti odloženého zastavení (Stop-Delay) končí.
                          Zastavení jsou od tohoto okamžiku uskutečňována okamžitě.
```

Příklad: Výpis programu

Ve smyčce se opakuje následující blok:



V obrázku je vidět, že když uživatel v oblasti odloženého zastavení (Stop-Delay) stiskne tlačítko "Stop", zahájí NC systém operaci brždění mimo oblast odloženého zastavení, tzn. v bloku `N100`. Díky tomu se NC systém v oblasti před blokem `N100` zastaví.

```
Programový kód                                  Komentář
...
N99 MY_LOOP:
N100 G0 Z200
N200 G0 X0 Z200
```

Programový kód	Komentář
N300 DELAYFSTON()	
N400 G33 Z5 K2 M3 S1000	
N500 G33 Z0 X5 K3	
N600 G0 X100	
N700 DELAYFSTOF()	
N800 GOTOB MY_LOOP	

Podrobnosti týkající se vyhledávání bloku typu SERUPRO a posuvů ve spojení s příkazy G331/G332Posuv při vrtání závitů bez vyrovnávací hlavičky, viz:

Literatura:
Příručka Popis funkcí, Základní funkce; BAG, kanál, zpracování programu (K1),
Příručka Popis funkcí, Základní funkce; "Posuvy (V1)"

Výhody oblasti odloženého zastavení (Stop-Delay)

Úsek programu se zpracuje se spojitým průběhem rychlosti (bez přerušení).

Pokud uživatel poté, co se systém zastavil, přeruší program pomocí tlačítka RESET, nachází se přerušený blok až za chráněnou oblastí. Tento programový blok se potom hodí jako cíl pro následující operaci vyhledávání.

Dokud se zpracovává oblast odloženého zastavení (Stop-Delay), následující osy hlavního zpracování se nezastavují:

- Příkazové osy a
- Polohovací osy, jejichž pohyb je ovládán příkazem POSA

Příkaz výrobního programu G4 je v oblasti odloženého zastavení (Stop-Delay) přípustný, oproti tomu jiné příkazy výrobního programu, které způsobují přechodné zastavení (např. WAITM) přípustné nejsou.

Příkaz G4, stejně jako pohyby po dráze, oblast odloženého zastavení (Stop-Delay) aktivuje, příp. zachovává její platnost.

Příklad: Zásahy do posuvu

Jestliže je korekčním přepínačem (Override) nastaven před oblastí odloženého zastavení (Stop-Delay) pokles na 6%, zůstává tato korekce v platnosti i v oblasti odloženého zastavení.

Jestliže je korekce snížena ze 100% na 6% uvnitř oblasti odloženého zastavení (Stop-Delay), potom se oblast odloženého zpracování zpracuje s nastavením 100% až do konce a dál zpracování probíhá s nastavením 6%.

Blokování posuvu se v oblasti odloženého zastavení (Stop-Delay) neuplatňuje, zpracování je pozastaveno až po opuštění oblasti odloženého zastavení.

Překrývání/vnoření:

Jestliže se dvě oblasti odloženého zastavení (Stop-Delay) protínají, jedna z příkazů NC jazyka a druhá ze strojního parametru MD 11550: STOP_MODE_MASK, bude vytvořena největší možná oblast odloženého zastavení.

Chování příkazů NC jazyka DELAYFSTON a DELAYFSTOF ve spojení s vnořeními a konci podprogramů se řídí následujícími pravidly:

1. Na konci podprogramu, v němž byl vyvolán příkaz DELAYFSTON, se implicitně aktivuje příkaz DELAYFSTOF.
2. Oblast odloženého zastavení (Stop-Delay) vyvolaná příkazem DELAYFSTON nemá žádný efekt.
3. Pokud podprogram 1 vyvolává v oblasti odloženého zastavení (Stop-Delay) podprogram 2, stává se celý podprogram 2 oblastí odloženého zastavení. Zejména příkazy DELAYFSTOF nemají v podprogramu 2 žádný efekt.

Poznámka

Příkaz REPOSA je konec podprogramu a příkaz DELAYFSTON se v každém případě deaktivuje.

Pokud se v oblasti odloženého zastavení (Stop-Delay) vyskytne událost "tvrdého" zastavení, bude celá oblast "Stop-Delay" deaktivována! To znamená, že pokud se vyskytne v tomto úseku programu další libovolné zastavení, okamžitě se uskuteční. Teprve po novém naprogramování příkazu DELAYFSTON může nová oblast odloženého zastavení začít.

Jestliže je tlačítko Stop stisknuto před začátkem oblasti odloženého zastavení a pokud NCK musí kvůli zabrždění zajet do této oblasti, zastaví se NCK v oblasti odloženého zastavení a oblast odloženého zastavení zůstane deaktivována!

Jestliže se do oblasti odloženého zastavení vstoupí s nastavením korekce (Override) 0%, oblast odloženého zastavení **nebude** akceptována!

To platí pro všechny události "měkkého" zastavení.

V oblasti odloženého zastavení (Stop-Delay) je možné zabrzdit pomocí příkazu STOPALL. Příkazem STOPALL se ale okamžitě aktivují také všechny ostatní události způsobující zastavení, které až dosud byly zpožděny.

Systémové proměnné

Oblast odloženého zastavení může být ve výrobním programu rozpoznána pomocí proměnné $P_DELAYFST. Pokud je bit 0 této systémové proměnné nastaven na 1, nachází se zpracování výrobního programu v daném okamžiku v oblasti odloženého zastavení.

Oblast odloženého zastavení může být v synchronní akci rozpoznána pomocí proměnné $AC_DELAYFST. Pokud je bit 0 této systémové proměnné nastaven na 1, nachází se zpracování výrobního programu v daném okamžiku v oblasti odloženého zastavení.

Kompatibilita

Inicializační nastavení bitu 0 = 0 ve strojním parametru MD 11550: STOP_MODE_MASK vyvolává implicitní oblast odloženého zastavení, kdykoli je zpracováván příkaz ze skupiny G-kódů G331/G332 a když je naprogramován pohyb po dráze, příp. příkaz G4.

Nastavení bit 0 = 1 umožňuje zastavení i během zpracovávání příkazu ze skupiny G-kódů G331/G332 a kdykoli je naprogramován pohyb po dráze, příp. příkaz G4 (stejné chování jako do verze SW 6). Pro definici oblasti odloženého zastavení se musí používat příkazy DELAYFSTON/DELAYFSTOF.

8.5 Zabránění pozice programu pro SERUPRO (IPTRLOCK, IPTRUNLOCK)

Funkce

Pro určité komplikované mechanické situace na stroji je zapotřebí, aby se zabránilo vyhledávání bloku pomocí příkazu SEROPRO.

Pomocí programovatelného ukazatele přerušení existuje možnost, jak při operaci "Vyhledávání místa přerušení" zasáhnout a přesunout se před prohledávané místo.

Ve výrobním programu mohou být definovány také oblasti vyhledávání, do kterých NCK ještě nemůže znovu vstoupit. Při přerušení programu se v NCK zaznamená naposled zpracovávaný blok, který potom může být pomocí uživatelského rozhraní HMI vyhledán.

Syntaxe

```
IPTRLOCK
IPTRUNLOCK
```

Tyto příkazy musí být uvedeny na samostatném řádku výrobního programu a umožňují pracovat s programovatelným ukazatelem přerušení.

Význam

`IPTRLOCK`	Začátek úseku programu pro vyhledávání
`IPTRUNLOCK`	Konec úseku programu pro vyhledávání

Oba příkazy jsou přípustné jen ve výrobních programech, **nikoli** v synchronních akcích.

Příklad

Vnoření úseků programu, které mohou být prohledávány, na dvou programových úrovních s implicitním příkazem IPTRUNLOCK. Implicitní příkaz IPTRUNLOCK v podprogramu 1 končí oblast určenou pro vyhledáváním.

Programový kód	**Komentář**
N10010 IPTRLOCK()	
N10020 R1 = R1 + 1	
N10030 G4 F1	; Blok s pozastavením, úsek programu určený pro vyhledávání začíná.
...	
N10040 Podprogram 2	
...	; Interpretace podprogramu 2.
N20010 IPTRLOCK ()	; Žádná platnosti, opakovaný začátek.
...	
N20020 IPTRUNLOCK ()	; Žádná platnost, konec v jiné rovině.
N20030 RET	
...	
N10060 R2 = R2 + 2	
N10070 RET	; Konec úseku programu pro vyhledávání.
N100 G4 F2	; Hlavní program pokračuje.

Přerušení na řádku 100 potom znovu vytvoří ukazatel přerušení.

Zjišťování a prohledávání oblastí určených pro vyhledávání

Úseky programu určené pro vyhledávání jsou označeny příkazy NC jazyka IPTRLOCK a IPTRUNLOCK.

Příkaz IPTRLOCK zmrazí ukazatele přerušení na určitém bloku (SBL1), který může být realizován v hlavním zpracování. Tento blok se v následujícím označí jako blok se zastavením. Pokud se po příkazu IPTRLOCK vyskytne přerušení programu, pak je možné na uživatelském rozhraní HMI tento tak zvaný blok zastavení vyhledat.

Opětovné nastavení na aktuální blok

Ukazatel přerušení se pomocí příkazu IPTRLOCK pro následující úsek programu nastaví na aktuální blok k bodu, kde došlo k přerušení.

Po nalezení cíle vyhledávání může být pomocí stejného bloku zastavení cíl vyhledávání opakován.

Uživatelem editovaný ukazatel přerušení musí být pomocí HMI znovu odstraněn.

Pravidla pro vnoření

Chování příkazů NC jazyka `IPTRLOCK` a `IPTRUNLOCK` ve spojení s vnořeními a konci podprogramů se řídí následujícími pravidly:

1. Na konci podprogramu, v němž byl vyvolán příkaz `IPTRLOCK`, se implicitně aktivuje příkaz `IPTRUNLOCK`.
2. Oblast určená pro vyhledávání nastavená příkazem `IPTRLOCK` nemá žádný efekt.
3. Pokud podprogram 1 vyvolá v oblasti určené pro vyhledávání podprogram 2, zůstává celý podprogram 2 určený pro vyhledávání. Zejména příkazy `IPTRUNLOCK` nemají v podprogramu 2 žádný efekt.

Pokud budete potřebovat další informace, viz:
/FB1/, Příručka Popis funkcí, Základní funkce; BAG, kanál, zpracování programu (K1).

Systémové proměnné

Oblast určená pro vyhledávání může být ve výrobním programu rozpoznána pomocí proměnné `$P_IPTRLOCK`.

Automatický ukazatel vyhledávání

Funkce automatického ukazatele vyhledávání automaticky definuje dříve nastavený typ vazby jako objekt určený pro vyhledávání. Prostřednictvím strojního parametru se pro

- Elektronickou převodovku (`EGON`)
- Vazbu os pomocí řídící hodnoty (`LEADON`)

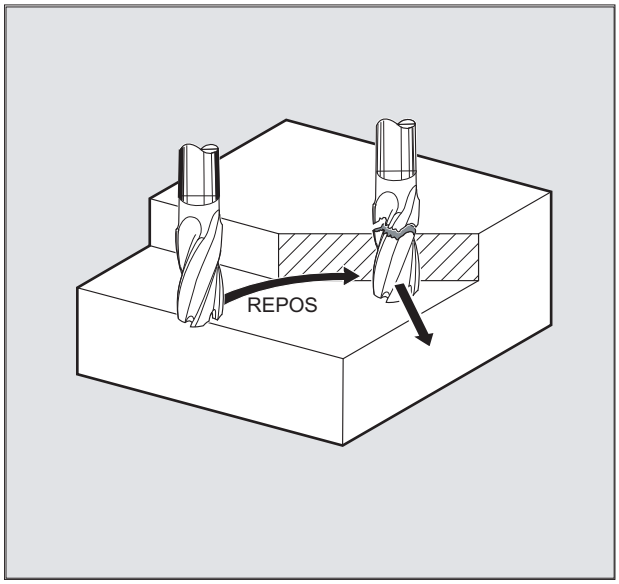
aktivuje automatický ukazatel přerušení. Jestliže se naprogramované a prostřednictvím strojního parametru aktivované automatické ukazatele přerušení překrývají, vytvoří se největší možná oblast určená pro vyhledávání.

8.6 Opětovné najíždění na konturu (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN)

Funkce

Jestliže v průběhu zpracovávání běžící program přerušíte a s nástrojem odjedete od obrobku - například kvůli zlomení nástroje nebo protože potřebujete něco doměřit - můžete na konturu znovu najet ve zvoleném bodě, přičemž tento pohyb je řízený programem.

Příkaz REPOS se chová stejně jako návrat z podprogramu (např. pomocí M17). Následující bloky v rutině přerušení se už neuskuteční.



Pokud budete potřebovat informace týkající se přerušení zpracovávání programu, viz také úsek "Flexibilní programování NC systémů", kapitola "Rutina přerušení" v této příručce programování.

Syntaxe

```
REPOSA RMI DISPR=…
REPOSA RMB
REPOSA RME
REPOSA RMN
REPOSL RMI DISPR=…
REPOSL RMB
REPOSL RME
REPOSL RMN
REPOSQ RMI DISPR=… DISR=…
REPOSQ RMB DISR=…
REPOSQ RME DISR=…
REPOSQA DISR=…
REPOSH RMI DISPR=… DISR=…
REPOSH RMB DISR=…
REPOSH RME DISR=…
REPOSHA DISR=…
```

Význam

Najížděcí dráha

`REPOSA`	Najíždění po přímkách všemi osami
`REPOSL`	Najíždění po přímkách
`REPOSQ DISR=…`	Najíždění po čtvrtkruhu s rádiusem DISR
`REPOSQA DISR=…`	Najíždění všemi osami po čtvrtkruhu s rádiusem DISR
`REPOSH DISR=…`	Najíždění po půlkruhu s průměrem DISR
`REPOSHA DISR=…`	Najíždění všemi osami po půlkruhu s rádiusem DISR

Bod pro opětovné najíždění

`RMI`	Najíždění na bod, kde došlo k přerušení
`RMI DISPR=…`	Vstupní bod ve vzdálenosti DISPR v mm/palcích před bodem, kde došlo k přerušení
`RMB`	Najíždění na počáteční bod bloku
`RME`	Najíždění na koncový bod bloku
`RME DISPR=…`	Najíždění na koncový bod bloku ve vzdálenosti DISPR před koncovým bodem
`RMN`	Najíždění na nejblíže ležící bod dráhy
`A0 B0 C0`	Osy, kterými se má najíždět

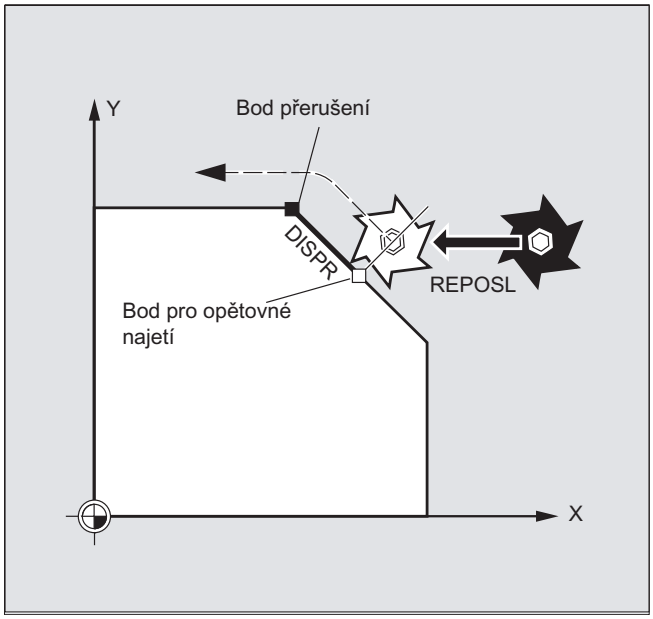
Příklad: Najíždění po přímkách, REPOSA, REPOSL

Nástroj najíždí na bod pro opětovné najetí přímo po přímce.

Pomocí příkazu REPOSA se automaticky najíždí všemi osami. V případě příkazu REPOSL můžete zadat osy, kterými se má najíždět.

Příklad:
```
REPOSL RMI DISPR=6 F400
```

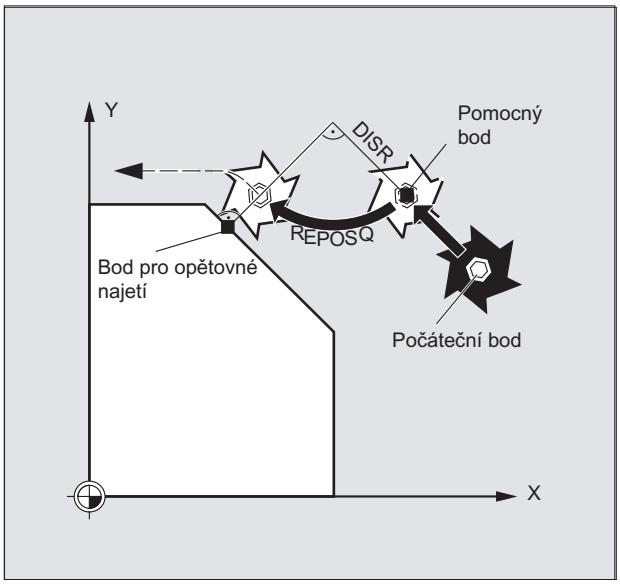
nebo
```
REPOSA RMI DISPR=6 F400
```



Příklad: Najíždění po čtvrtkruhu, REPOSQ, REPOSQA

Nástroj najíždí na bod pro opětovné najetí po čtvrtkruhové dráze s rádiusem `DISR=...`
Potřebný pomocný bod mezi počátečním bodem a bodem pro opětovné najetí vypočítává řídící systém automaticky.

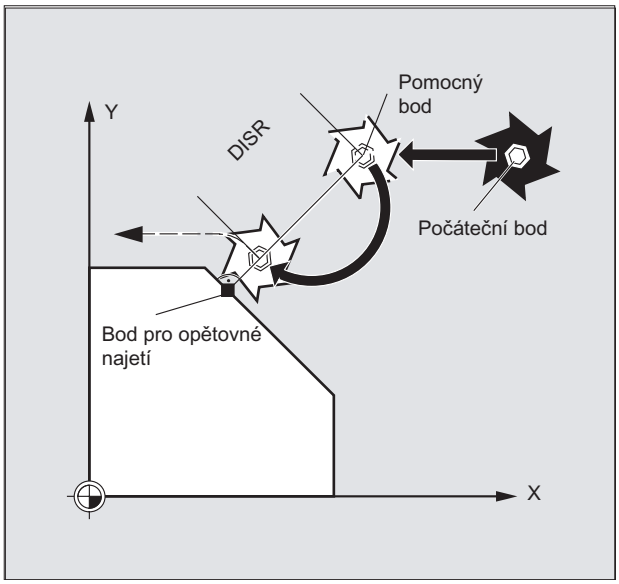
Příklad:
```
REPOSQ RMI DISR=10 F400
```



Příklad: Najížděním nástrojem po půlkruhové dráze, REPOSH, REPOSHA

Nástroj najíždí na bod pro opětovné najetí po půlkruhové dráze s průměrem `DISR=...`
Potřebný pomocný bod mezi počátečním bodem a bodem pro opětovné najetí vypočítává řídící systém automaticky.

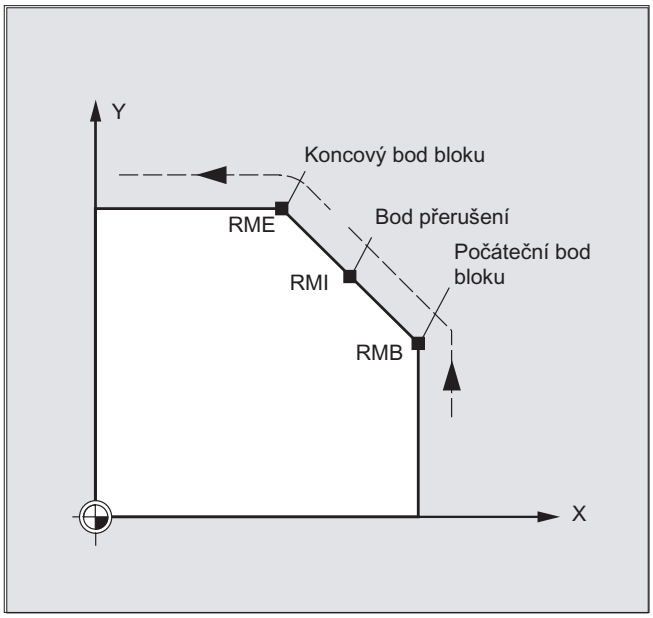
Příklad:
```
REPOSH RMI DISR=20 F400
```



Definice bodu pro opětovné najíždění (nikoli pro najíždění SERUPRO s RMN)

V souvislosti s NC blokem, ve kterém bylo zpracování programu přerušeno, si můžete vybrat mezi třemi body pro opětovné najetí:

- RMI, bod, kde došlo k přerušení
- RMB, počáteční bod bloku, příp. poslední koncový bod
- RME, koncový bod bloku



Pomocí příkazů `RMI DISPR=...`, příp. `RME DISPR=...` můžete definovat bod pro opětovné najíždění, který leží před bodem, v němž došlo k přerušení, příp. před koncovým bodem bloku.

Pomocí příkazu `DISPR=...` zadáváte dráhu na kontuře v mm/palcích, o kterou bod pro opětovné najíždění leží **před** bodem, v němž došlo k přerušení, příp. před koncovým bodem. Tento bod může - i v případě větších hodnot - ležet maximálně v počátečním bodu bloku.

Jestliže není naprogramována žádná hodnota parametru `DISPR=...`, platí nastavení `DISPR=0` a v důsledku toho se použije bod, kde došlo k přerušení (v případě příkazu `RMI`), příp. koncový bod bloku (v případě `RME`).

Znaménko parametru DISPR

Znaménko parametru `DISPR` se vyhodnocuje. Je-li znaménko kladné, funkce se chová, jak bylo popsáno výše.

V případě záporného znaménka se nástroj znovu nasazuje za bod, kde došlo k přerušení, příp. v případě příkazu `RMB` za počáteční bod.

Vzdálenost bodu přerušení a bodu nasazení nástroje vyplývá z hodnoty uložené v parametru `DISPR`. Tento bod může i v případě větších hodnot ležet maximálně v počátečním bodu bloku.

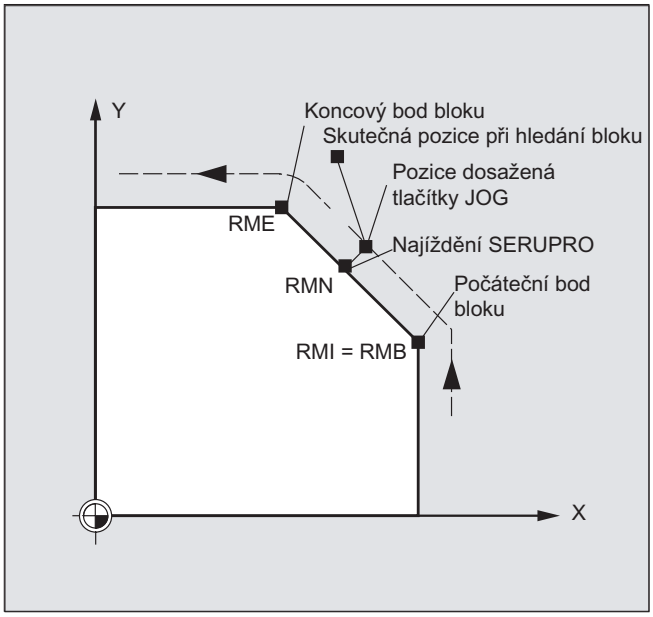
Příklad použití:

Pomocí senzoru bylo zjištěno přiblížení k upínací čelisti. Aktivuje se program `ASUP`, pomocí kterého se upínací čelist objede.

Nakonec se znovu najede na bod za upínací čelistí se zápornou hodnotou parametru `DISPR` a program bude pokračovat.

Najíždění pomocí příkazu SEROPRO s RMN

Jestliže si v průběhu obrábění okolnosti vyžádají na libovolném místě přerušení, pak se pomocí najíždění SERUPRO s `RMN` najede co možno nejblíže před místo, kde došlo k přerušení, aby se nakonec opracovala pouze zbývající dráha. Za tím účelem spouští uživatel operaci SERUPRO na bloku, kde došlo k přerušení, a pomocí tlačítek JOG najíždí před poškozené místo cílového bloku.

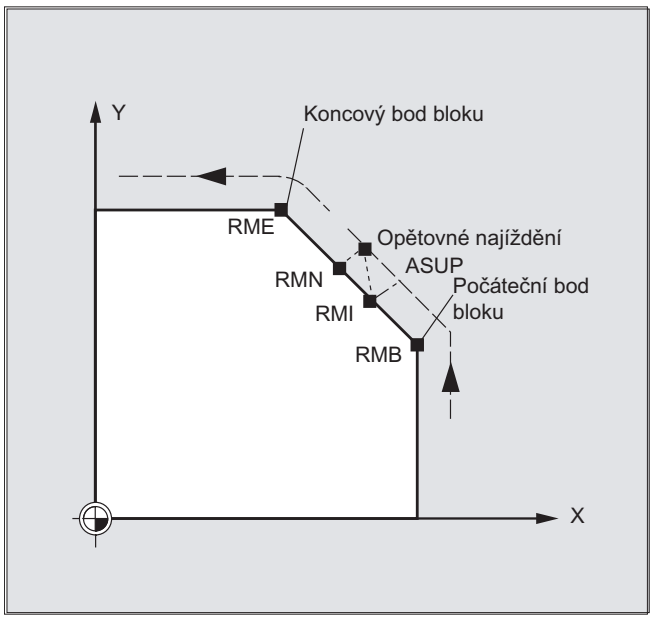


Poznámka
SERUPRO

Pro operaci `SERUPRO` jsou parametry `RMI` a `RMB` identické.Parametr `RMN` není omezen pouze na operaci `SERUPRO`, nýbrž platí všeobecně.

Najíždění před nejblíže ležící bod dráhy RMN

Pokud jde o okamžik interpretace příkazu `REPOSA`, po přerušení není blok pro opětovné najetí s příkazem `RMN` zahájen celý ještě jednou, nýbrž je opracovávána pouze zbytková dráha. Najíždí se na bod dráhy, který leží co možno nejblíže bloku, v němž došlo k přerušení.



Stavové informace pro platný režim REPOS

Platný režim funkce REPOS přerušeného bloku je možné načíst pomocí synchronních akcí prostřednictvím proměnné `$AC_REPOS_PATH_MODE`.

0: Najíždění není definováno

1 `RMB`: Najíždění na začátek

2 `RMI`: Najíždění na místo, kde došlo k přerušení

3 `RME`: Najíždění na koncový bod bloku

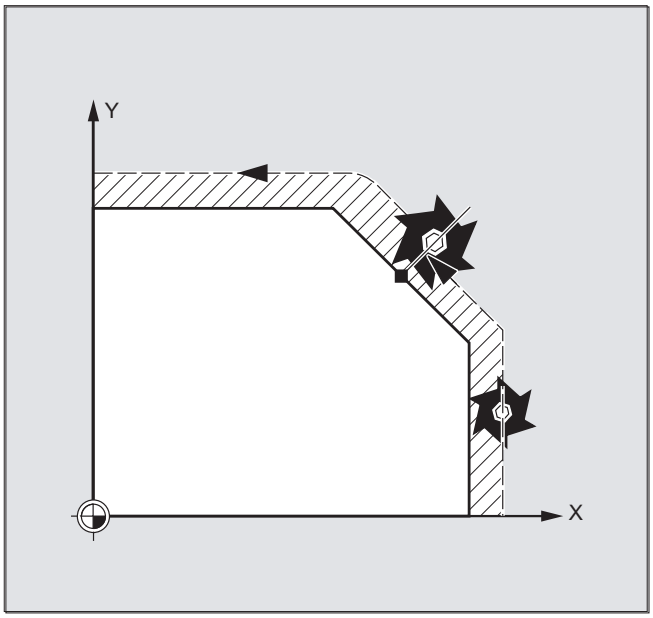
4 `RMN`: Najíždění na bod dráhy, který leží nejblíže bloku, kde došlo k přerušení.

Najíždění s novým nástrojem

Jestliže jste zpracovávání programu přerušili kvůli zlomení nástroje:

Naprogramováním nového D_čísla bude zpracování programu pokračovat od bodu opětovného najetí se změněnými hodnotami korekčních parametrů nástroje.

V případě změněných hodnot korekčních parametrů nástroje se může stát, že na místo, v němž došlo k přerušení, nebude možné najet. V tomto případě se bude najíždět na bod na nové kontuře, který leží co možno nejblíže bodu, v němž došlo k přerušení (v případě potřeby s úpravou podle parametru DISPR).



Najíždění na konturu

Pohyb, se kterým nástroj znovu najíždí na konturu, může být naprogramován. Pro adresy os, které se mají pohybovat, zadejte nulovou hodnotu.

Pomocí příkazů `REPOSA`, `REPOSQA` a `REPOSHA` jsou automaticky všechny osy nastavovány na původní pozici. Zadání os není zapotřebí.

Jestliže je naprogramován příkaz `REPOSL`, `REPOSQ` a `REPOSH`, pohybují se všechny geometrické osy automaticky, aniž by byly v příkazu zadány. Všechny ostatní osy musí být v příkazu uvedenu.

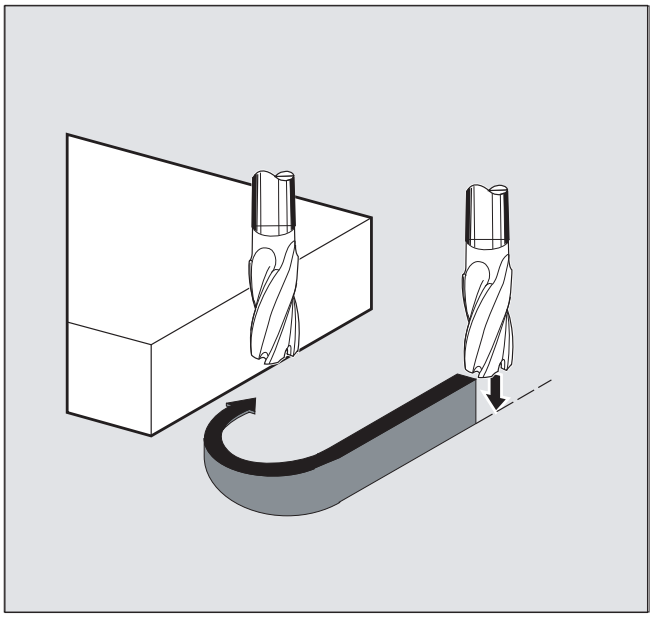
Pro kruhové pohyby s příkazy REPOSH a REPOSQ platí:

Kruhový pohyb se uskutečňuje v zadané pracovní rovině `G17` až `G19`.

Jestliže je v najížděcím bloku uvedena třetí geometrická osa (směr přísuvu), potom se v případě, že si poloha nástroje a naprogramovaná poloha ve směru přísuvu neodpovídají, uskuteční zpětný pohyb na bod pro opětovné najetí po spirále.

V následujících případech se automaticky přepíná na lineární najíždění `REPOSL`:

- Nezadali jste žádnou hodnotu pro parametr `DISR`.
- Neexistuje žádný definovaný směr najíždění (přerušení programu v bloku bez informací o posuvech).
- V případě směru najíždění kolmo na aktuální pracovní rovinu.



8.7 Ovlivňování vedení pohybu

8.7.1 Procentuální korekce ryvu (JERKLIM)

Funkce

Pomocí příkazu NC jazyka `JERKLIM` může být snížen nebo zvýšen prostřednictvím strojního parametru nastavený maximální možný ryv osy při pohybu po dráze v kritických úsecích programu.

Předpoklady

Musí být aktivní režim zrychlení SOFT.

Platnost

Funkce se uplatňuje::

- v automatických provozních režimech
- jen pro dráhové osy

Syntaxe

```
JERKLIM[<osa>]=<hodnota>
```

Význam

`JERKLIM`:	Příkaz pro korekci ryvu
`<osa>`:	Osa stroje, jejíž mezní hodnota ryvu má být přizpůsobena.
`<hodnota>`:	Procentuální hodnota korekce vztahující se na v konfiguraci nastavený maximální ryv osy při pohybu po dráze (MD32431 $MA_MAX_AX_JERK). Rozsah hodnot: 1 ... 200 Hodnota 100 má za následek žádné ovlivňování ryvu.

Poznámka

Chování příkazu JERKLIM na konci výrobního programu a při resetu kanálu se nastavuje v konfiguraci pomocí bitu 0 strojního parametru MD32320 $MA_DYN_LIMIT_RESET_MASK:

- Bit 0 = 0:

  Naprogramovaná hodnota pro příkaz JERKLIM a při resetu kanálu/M30 se nastavuje zpět na 100%.
- Bit 0 = 1:

  Naprogramovaná hodnota pro příkaz JERKLIM zůstává při resetu kanálu/M30 zachována.

Příklad

Programový kód	Komentář
...	
N60 JERKLIM[X]=75	; Osové obrábění načisto ve směru osy X má být uskutečňováno se zrychlením/zpomalením odpovídajícím jen s maximálně 75% ryvu přípustnéo pro osu.
...	

8.7.2 Procentuální korekce rychlosti (VELOLIM)

Funkce

Pomocí příkazu NC jazyka `VELOLIM` může být v kritických úsecích programu omezena prostřednictvím strojního parametru nastavená maximální možná rychlost osy/vřetena v osovém režimu, příp. maximální možné otáčky vřetena, které závisí na stupni převodovky, v režimu vřetena (režim regulace otáček M3, M4, M5 a polohovací režimu SPOS, SPOSA, M19) např. proto, aby se snížila náročnost pro obráběcí stroj nebo aby se zlepšila jakost obrábění.

Platnost

Funkce se uplatňuje::

- v automatických provozních režimech
- na dráhové a polohovací osy
- na vřetena v režimu vřetena/osy

Syntaxe

```
VELOLIM[<osa/vřeteno>]=<hodnota>
```

Význam

`VELOLIM`: Příkaz pro korekci rychlosti

`<osa/vřeteno>`: Osa stroje nebo vřeteno, jejichž mezní hodnota rychlosti nebo otáček má být přizpůsobena.

Funkce VELOLIM pro vřetena

Prostřednictvím strojního parametru (MD30455 $MA_MISC_FUNCTION_MASK, bit 6) může být pro programování výrobních programů nastaveno, zda má být funkce `VELOLIM` uplatňována nezávisle na momentálním použití jako vřeteno nebo jako osa (bit 6 = 1) nebo zda může být naprogramována odděleně pro každý provozní režim (bit 6 = 0). Jestliže je v konfiguraci nastaveno oddělené působení, je možné při programování pomocí identifikátorů vybírat z následujících možností:

- Identifikátor vřetena `S<n>` pro provozní režimy vřetena
- Identifikátor osy, např. "`C`", pro režim osy

`<hodnota>`: Procentuální hodnota korekce

Hodnota korekce se vztahuje:

- v případě os / vřeten v režimu osy (pokud je MD30455 bit 6 = 0):
  na maximální rychlost osy nastavenou v konfiguraci (MD32000 $MA_MAX_AX_VELO).
- v případě vřeten v režimu vřetena/osy (pokud je MD30455 bit 6 = 1):
  na maximální otáčky momentálního převodového stupně (MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT[<n>])

Rozsah hodnot: 1 ... 100

Hodnota 100 má za následek, že rychlost, příp. otáčky nejsou ovlivňovány.

Poznámka

Chování na konci výrobního programu a při resetu kanálu

Chování příkazu VELOLIM na konci výrobního programu a při resetu kanálu se nastavuje v konfiguraci pomocí bitu 0 strojního parametru MD32320 $MA_DYN_LIMIT_RESET_MASK:

- Bit 0 = 0:
  Naprogramovaná hodnota pro příkaz VELOLIM a při resetu kanálu/M30 se nastavuje zpět na 100%.
- Bit 0 = 1:
  Naprogramovaná hodnota pro příkaz VELOLIM zůstává při resetu kanálu/M30 zachována.

Poznámka

Příkaz VELOLIM pro vřetena v synchronních akcích

Když je příkaz `VELOLIM` naprogramován v synchronních akcích, není rozlišováno mezi režimem vřetena a režimem osy. Nezávisle na identifikátoru, který byl použit při programování, jsou otáčky v režimu vřetena a rychlost v režimu osy omezeny stejným způsobem.

Diagnostika

Diagnostika příkazu VELOLIM v režimu vřetena

Aktivní omezení otáček vyvolané příkazem `VELOLIM` (menší než 100%) může být v režimu vřetena rozpoznáno načítáním systémových proměnných $AC_SMAXVELO a $AC_SMAXVELO_INFO.

V případě omezení se v proměnné $AC_SMAXVELO nachází mezní hodnota otáček nastavená příkazem `VELOLIM`. Proměnná $AC_SMAXVELO_INFO obsahuje v tomto případě hodnotu "16", která znamená, že příčinou omezení je příkaz VELOLIM.

Příklady

Příklad 1: Omezení rychlosti osy stroje

```
Programový kód              Komentář
...
N70 VELOLIM[X]=80           ; Osové obrábění načisto ve směru osy X má být
                              uskutečňováno s rychlostí odpovídající jen s maximálně
                              80% rychlosti přípustné pro osu.
...
```

Příklad 2: Omezení otáček vřetena

```
Programový kód          Komentář
N05 VELOLIM[S1]=90      ; Omezení maximálních otáček vřetena 1 na 90% z 1000 ot/min.
...
N50 VELOLIM[C]=45       ; Omezení otáček vřetena na 45% z 1000 ot/min, pokud C je
                          osový identifikátor S1.
...
```

Konfigurační parametry pro vřeteno 1 (AX5):

MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT[1,AX5]=1000	; Maximální otáčky stupně převodovky 1 = 1000 U/min
MD30455 $MA_MISC_FUNCTION_MASK[AX5] = 64	; Bit 6 = 1: Naprogramování příkazu `VELOLIM` se uplatňuje společně pro režim osy a režim vřetena nezávisle na naprogramovaném identifikátoru.

8.7.3 Příklad programování pro příkazy JERKLIM a VELOLIM

Následující program představuje příklad použití pro procentuální omezení ryvu a rychlosti:

```
N1000 G0 X0 Y0 F10000 SOFT G64
N1100 G1 X20 RNDM=5 ACC[X]=20
ACC[Y]=30
N1200 G1 Y20 VELOLIM[X]=5          ; Osové obrábění načisto ve směru osy X má
                                     být uskutečňováno s rychlostí odpovídající
                                     jen s max. 5% rychlosti přípustné pro osu.
JERKLIM[Y]=200                     ; Osové obrábění načisto ve směru osy Y může
                                     být uskutečňováno se zrychlením/zpomalením
                                     odpovídajícím s maximálně 200% ryvu
                                     přípustného pro osu.
N1300 G1 X0 JERKLIM[X]=2           ; Osové obrábění načisto ve směru osy X má
                                     být uskutečňováno se zrychlením/zpomalením
                                     odpovídajícím jen s max. 2% ryvu
                                     přípustného pro osu.
N1400 G1 Y0
M30
```

8.8 Programovatelná tolerance kontury/orientace (CTOL, OTOL, ATOL)

Funkce

Pomocí příkazů `CTOL`, `OTOL` a `ATOL` je možné přizpůsobit tolerance obráběcích prací, které jsou definovány strojními a nastavovanými parametry, pro funkce kompresoru (COMPON, COMPCURV, COMPCAD), pro pohyby na přechodových zaobleních G642, G643, G645, OST a pro vyhlazování orientace pomocí příkazu ORISON v NC programu.

Naprogramované hodnoty platí, dokud nejsou znovu naprogramovány nebo dokud nejsou přiřazením záporné hodnoty vymazány. Kromě toho jsou vymazány na konci programu, při resetu kanálu, resetu BAG, resetu NCK (teplý start) a po vypnutí a zapnutí (studený start). Po vymazání platí znovu hodnoty ze strojních a nastavovaných parametrů.

Syntaxe

```
CTOL=<hodnota>
OTOL=<hodnota>
ATOL[<osa>]=<hodnota>
```

Význam

`CTOL` Příkaz pro programování **tolerance kontury**

Příkaz `CTOL` platí pro:

- všechny funkce kompresoru
- všechny pohyby na přechodových zaobleních, kromě příkazů G641 a G644

`<hodnota>`: Hodnota pro toleranci kontury je délkový údaj.

Typ: REAL

Jednotka: Palce/mm (v závislosti na momentálním nastavení měřicích jednotek)

`OTOL` Příkaz pro programování **tolerance orientace**

Příkaz `OTOL` platí pro:

- všechny funkce kompresoru
- Vyhlazování orientace pomocí příkazu ORISON
- všechny pohyby na přechodových zaobleních, kromě příkazů G641, G644 a OSD

`<hodnota>`: Hodnota pro toleranci orientace je úhlový údaj.

Typ: REAL

Jednotka: stupně

ATOL Příkaz pro programování **tolerance určité osy**

Příkaz ATOL platí pro:

- všechny funkce kompresoru
- Vyhlazování orientace pomocí příkazu ORISON
- všechny pohyby na přechodových zaobleních, kromě příkazů G641, G644 a OSD

<osa>: Název osy, pro kterou má být naprogramována tolerance osy

<hodnota>: Hodnota pro toleranci osy je v závislosti na typu osy (lineární nebo kruhová osa) buď délkovým nebo úhlovým údajem.

Typ: REAL

Jednotka: pro lineární osy: Palce/mm (v závislosti na momentálním nastavení měřicích jednotek)

pro kruhové osy: stupně

Poznámka

Příkazy CTOL a OTOL mají přednost před příkazem ATOL.

Okrajové podmínky

Framy provádějící změnu měřítka

Framy provádějící změnu měřítka ovlivňují naprogramované tolerance stejným způsobem, jako polohy os, tzn. relativní tolerance zůstává stejná.

Příklad

```
Programový kód                  Komentář
COMPCAD G645 G1 F10000          ; Aktivování funkce kompresoru COMPCAD.
X... Y... Z...                  ; Zde se uplatňují strojní a nastavované parametry.
X... Y... Z...
X... Y... Z...
CTOL=0.02                       ; Od tohoto místa se uplatňuje tolerance kontury 0,02 mm.
X... Y... Z...
X... Y... Z...
X... Y... Z...
ASCALE X0.25 Y0.25 Z0.25        ; Od tohoto místa se uplatňuje tolerance kontury
                                  0,005 mm.
X... Y... Z...
X... Y... Z...
X... Y... Z...
CTOL=-1                         ; Od tohoto místa se znovu uplatňují strojní a
                                  nastavované parametry.
X... Y... Z...
X... Y... Z...
X... Y... Z...
```

Další informace

Načítání hodnot tolerance

Pro náročnější případy použití nebo kvůli diagnostice mohou být momentálně platné tolerance pro funkce kompresoru (COMPON, COMPCURV, COMPCAD), pro druhy přechodových zaoblení G642, G643, G645, OST a pro vyhlazování orientace ORISON čteny pomocí systémových proměnných nezávisle na druhu jejich uplatnění.

- V synchronních akcích nebo se zastavením předběžného zpracování ve výrobním programu pomocí systémových proměnných:

$AC_CTOL	Tolerance kontury, která byla v platnosti při přípravě aktuálního bloku v hlavním zpracování Jestliže žádná tolerance kontury není v platnosti, v proměnné `$AC_CTOL` se nachází odmocnina ze součtu druhých mocnin tolerancí geometrických os.
$AC_OTOL	Tolerance orientace, která byla v platnosti při přípravě aktuálního bloku v hlavním zpracování Jestliže žádná tolerance orientace není v platnosti, v proměnné `$AC_OTOL` se v průběhu aktivní transformace orientace nachází odmocnina ze součtu druhých mocnin tolerancí orientačních os, jinak je tam hodnota "-1".
$AA_ATOL[<osa>]	Tolerance osy, která byla v platnosti při přípravě aktuálního bloku v hlavním zpracování Pokud je aktivní tolerance kontury, obsahuje proměnná `$AA_ATOL[<geometrická osa>]` toleranci kontury dělenou odmocninou z počtu geometrických os. Pokud je aktivní tolerance orientace a pokud je aktivní nějaká transformace orientace, obsahuje proměnná `$AA_ATOL[<orientační osa>]` toleranci orientace dělenou odmocninou z počtu orientačních os.

Poznámka

Jestliže nebyly naprogramovány žádné hodnoty tolerance, potom proměnné $A... nejsou dostatečně diferencované, aby bylo možné rozlišit případné různé tolerance jednotlivých funkcí, protože přece mohou reprezentovat jen jednu hodnotu.

Takové případy se mohou vyskytnout, jestliže jsou pomocí strojních a nastavovaných parametrů nastaveny odlišné tolerance pro funkce kompresoru, přechodová zaoblení a vyhlazování orientace. Proměnné potom obsahují největší hodnotu, jaká se vyskytuje u momentálně aktivních funkcí.

Jestliže je např. aktivní funkce kompresoru s tolerancí orientace 0,1 a vyhlazování orientace pomocí funkce ORISON s hodnotou 1°, v proměnné $AC_OTOL je uložena hodnota "1". Když se vyhlazování orientace vypne, je možné načíst ještě hodnotu "0,1".

- Bez zastavení předběžného zpracování ve výrobním programu pomocí systémových proměnných:

$P_CTOL	Programovatelná tolerance kontury
$P_OTOL	Programovatelná tolerance orientace
$PA_ATOL	Programovatelná tolerance osy

Poznámka

Jestliže nejsou naprogramovány žádné hodnoty tolerance, pak proměnné $P... obsahují hodnotu "-1".

8.9 Tolerance u pohybů s funkcí G0 (STOLF)

Toleranční faktor G0

Pohyby s příkazem G0 (rychlý posuv, přísuvné pohyby) mohou být uskutečňovány narozdíl od pohybů při opracovávání obrobků s větší tolerancí. Toto má tu výhodu, že se doby potřebné na vykonání pohybů s příkazem G0 zkracují.

Pro nastavení tolerancí při pohybech s funkcí G0 se používá konfigurace tolerančních faktorů G0 (MD20560 $MC_G0_TOLERANCE_FACTOR).

Toleranční faktor G0 se uplatňuje jen za těchto okolností:

- Je aktivní některá z následujících funkcí:
  - Funkce kompresoru: COMPON, COMPCURV a COMPCAD
  - Funkce pro přechodová zaoblení: G642 a G645
  - Vyhlazování orientace: OST
  - Vyhlazování orientace: ORISON
  - Vyhlazování orientace vztažené k dráze: ORIPATH
- za sebou následuje větší počet (≥ 2) bloků s G0

  V případě osamoceného bloku s G0 se toleranční faktor G0 neuplatňuje, protože **při přechodu** z pohybu, který není zadán pomocí G0, do pohybu s G0 (a obráceně) se v zásadě uplatňuje "**menší tolerance**" (tolerance pro opracovávání obrobku).

Funkce

Naprogramováním příkazu `STOLF` ve výrobním programu je možné dočasně přepsat v konfiguraci nastavený toleranční faktor G0 (MD20560). Hodnota v parametru MD20560 se přitom nemění. Po resetu, příp. po skončení výrobního programu bude opět v platnosti toleranční faktor nastavený v konfiguraci.

Syntaxe

```
STOLF=<toleranční faktor>
```

Význam

`STOLF`:	Příkaz pro naprogramování tolerančního faktoru G0.
`<toleranční faktor>`:	Toleranční faktor G0 Faktor může mít hodnotu větší než 1 i menší než 1. Za normálních okolností však mohou být nastavovány pro pohyby s příkazem G0 větší tolerance. Když je nastaveno `STOLF=1.0` (což odpovídá standardní hodnotě nastavené v konfiguraci), platí pro pohyby s G0 stejné tolerance jako pro pohyby uskutečňované pomocí jiných příkazů než G0.

Systémové proměnné

Toleranční faktor G0 platný ve výrobním programu, příp. v aktuálním bloku IPO, je možné načíst pomocí systémových proměnných.

- V synchronních akcích nebo se zastavením předběžného zpracování ve výrobním programu pomocí systémové proměnné:

  $AC_STOLF Aktivní toleranční faktor G0
  Toleranční faktor G0, který byl v platnosti při přípravě aktuálního bloku v hlavním zpracování.

- Bez zastavení předběžného zpracování ve výrobním programu pomocí systémové proměnné:

  $P_STOLF Naprogramovaný toleranční faktor G0

Jestliže v aktivním výrobním programu není pro funkci `STOLF` naprogramována žádná hodnota, je v obou těchto systémových proměnných uložena hodnota nastavená parametrem MD20560 $MC_G0_TOLERANCE_FACTOR.

Pokud v daném bloku není aktivní žádný rychlý posuv (G0), obsahují tyto systémové proměnné vždy hodnotu 1.

Příklad

Programový kód	**Komentář**
`COMPCAD G645 G1 F10000`	`; Funkce kompresoru COMPCAD`
`X... Y... Z...`	`; Zde se uplatňují strojní a nastavované parametry.`
`X... Y... Z...`	
`X... Y... Z...`	
`G0 X... Y... Z...`	
`G0 X... Y... Z...`	`; Zde se uplatňuje strojní parametr $MC_G0_TOLERANCE_FACTOR (např. =3) a také tolerance pro přechodová zaoblení $MC_G0_TOLERANCE_FACTOR*$MA_COMPRESS_POS_TOL.`
`CTOL=0.02`	
`STOLF=4`	
`G1 X... Y... Z...`	`; Od tohoto místa se uplatňuje tolerance kontury 0,02 mm.`
`X... Y... Z...`	
`X... Y... Z...`	
`G0 X... Y... Z...`	
`X... Y... Z...`	`; Od tohoto místa se uplatňuje toleranční faktor G0 rovnající se 4 a také tolerance kontury rovnající se 0,08 mm.`

Spojení os vazbou

9

9.1 Vlečení (TRAILON, TRAILOF)

Funkce

Při pohybech definované řídící osy se pohybují k ní přiřazené vlečené osy (= závislé osy) po drahách, které jsou odvozeny od pohybů řídící osy, přičemž je zohledňován ještě faktor vazby.

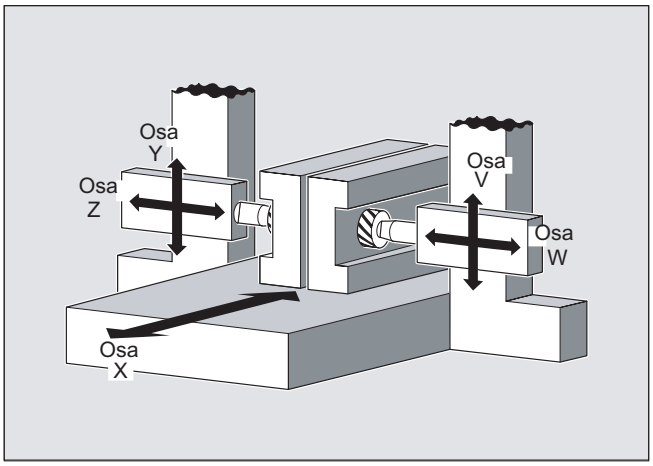
Řídící osa a vlečné osy spolu dohromady tvoří vlečné spojení os.

Oblasti použití

- Ovládání pohybu osy pomocí simulované osy. Řídící osa je simulovanou osou a vlečná osa je reálnou osou. Tímto způsobem je možné ovládat pohyb reálné osy, přičemž je zohledněn faktor vazby.
- Obrábění na dvou stranách se 2 vlečnými spojeními os:

  1. řídící osa Y, vlečná osa V

  2. řídící osa Z, vlečná osa W



Syntaxe

```
TRAILON(<vlečná osa>,<řídící osa>,<faktor vazby>)
TRAILOF(<vlečná osa>,<řídící osa>,<řídící osa 2>)
TRAILOF(<vlečná osa>)
```

Význam

`TRAILON`	Příkaz pro aktivování a definici vlečného spojení os. Platnost: modální
`<vlečná osa>`	Parametr 1: Identifikátor vlečné osy **Upozornění:** Vlečná osa může být také řídící osou pro další vlečené osy. Tímto způsobem je možné vytvářet různá další vlečná spojení os.
`<řídící osa>`	Parametr 2: Identifikátor řídící osy
`<faktor vazby>`	Parametr 3: Faktor vazby Faktor vazby udává požadovaný poměr mezi dráhou vlečné osy a dráhou řídící osy: **<**Faktor vazby**>** = Dráha vlečné osy / Dráha řídící osy Typ: REAL Předdefinovaná nastavení: 1 Zadání záporné hodnoty má za následek, že se vlečná osa bude pohybovat v opačném směru než osa řídící. Jestliže faktor vazby není při programování uveden, automaticky potom platí faktor vazby 1.

`TRAILOF`	Příkaz pro deaktivování vlečného spojení os
	Platnost: modální
	Příkaz `TRAILOF` se 2 parametry vypíná pouze vazbu k uvedené řídící ose:
	`TRAILOF(<vlečná osa>,<řídící osa>)`
	Jestliže je vlečná osa řízena 2 řídícími osami, pro zrušení obou vazeb je možné vyvolat příkaz `TRAILOF` se 3 parametry:
	`TRAILOF(<vlečná osa>,<řídící osa>,<řídící osa 2>)`
	Stejného výsledku dosáhnete tím, že příkaz `TRAILOF` naprogramujete bez udání řídící osy:
	`TRAILOF(<vlečná osa>)`

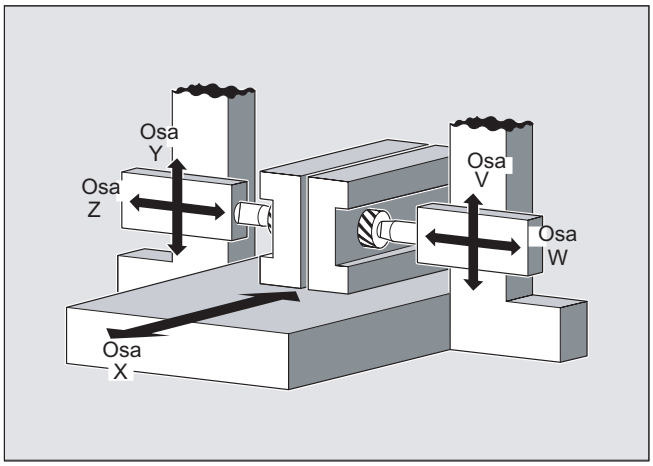
Poznámka

Vlečení se vždy uskutečňuje v základním souřadném systému (BCS).

Počet vlečných spojení, která mohou být aktivována současně, je omezen pouze kombinačními možnostmi os, které jsou na stroji k dispozici.

Příklad

Obrobek má být opracován na obou stranách s konfigurací os uvedenou na obrázku. Za tím účelem jsou vytvořena 2 vlečná spojení os.



Programový kód	**Komentář**
…	
N100 TRAILON(V,Y)	; Aktivování 1. vlečného spojení os
N110 TRAILON(W,Z,-1)	; Aktivování 2. vlečného spojení os. Faktor vazby je záporný: Vlečená osa se pohybuje vždy v opačném směru než osa řídící.
N120 G0 Z10	; Přísuv osy Z a osy W v opačných směrech os.
N130 G0 Y20	; Přísuv osy Z a osy V ve stejném směru.
…	
N200 G1 Y22 V25 F200	; Superpozice závislého a nezávislého pohybu vlečné osy V.
…	
TRAILOF(V,Y)	; Deaktivování 1. vlečného spojení os.
TRAILOF(W,Z)	; Deaktivování 2. vlečného spojení os.

Další informace

Typy os

Vlečné spojení os se může skládat z libovolné kombinace lineárních a kruhových os. Jako řídící osa přitom může být definována také simulovaná osa.

Vlečené osy

Vlečné ose je možné přiřadit maximálně 2 řídící osy současně. Přiřazení se provádí v různých vlečných spojeních.

Pro programování vlečné osy je možné využívat všech pohybových příkazů, které jsou k dispozici (G0, G1, G2, G3, …). Pohyby vlečné osy se skládají z nezávisle definovaných drah, které se skládají s drahami odvozenými od řídících os, jež jsou upraveny pomocí faktoru vazby.

Omezení dynamiky

Omezení dynamiky závisí na druhu aktivování vlečné vazby:

- Aktivování ve výrobním programu

  Pokud je aktivování uskutečněno ve výrobním programu a pokud jsou všechny řídící osy programovými osami v aktivovaném kanálu, potom se při pohybech řídících os zohledňuje dynamika vlečných os tak, aby žádná z vlečných os nebyla přetížena.

  Jestliže je aktivování provedeno ve výrobním programu s řídícími osami, které nejsou aktivní jako programové osy v aktivovaném kanálu ($AA_TYP ≠ 1), potom se při pohybech řídících os na dynamiku vlečné osy nebere zřetel. V důsledku toho může u vlečných os s menší dynamikou, než jaká je pro vazbu potřebná, dojít k přetížení.

- Aktivování v synchronních akcích

  Pokud se aktivování uskuteční v synchronní akci, nebude se při pohybech řídících os dynamika vlečných os zohledňovat. V důsledku toho může u vlečných os s menší dynamikou, než jaká je pro vazbu potřebná, dojít k přetížení.

⚠ POZOR

Pokud je vlečné spojení os aktivováno

- v synchronních akcích
- ve výrobním programu s řídícími osami, které nejsou programovými osami v kanálu vlečné osy,

potom uživatel/výrobce stroje v plné míře odpovídají za to, že budou přijata vhodná opatření, aby v důsledku posuvových pohybů řídící osy nemohlo dojít k přetížení vlečných os.

Stav vazby

Stav vazby dané osy může být ve výrobním programu zjištěn pomocí následujících systémových proměnných:

$AA_COUP_ACT[<osa>]

Hodnota	Význam
0	Žádná vazba není aktivní
8	Je aktivní vlečení

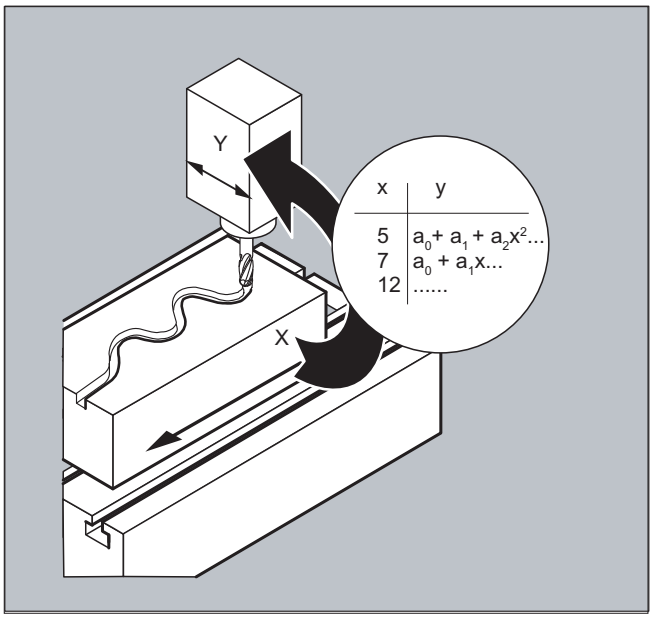
9.2 Tabulky křivek (CTAB)

Funkce

Pomocí tabulek křivek je možné naprogramovat polohové a rychlostní vztahy mezi dvěma osami (řídící a vlečná osa). Definice tabulek křivek se uskutečňuje ve výrobním programu.

Aplikace

Tabulky křivek nahrazují mechanické křivkové vačky. Tabulky křivek přitom tvoří základ pro vazbu řídící hodnotou mezi osami, neboť zajišťují funkční vztah mezi řídící a závislou hodnotou. Při odpovídajícím naprogramování řídící systém vypočítává ze vzájemně přiřazených pozic řídící a závislé osy polynom, který odpovídá křivkové vačce.

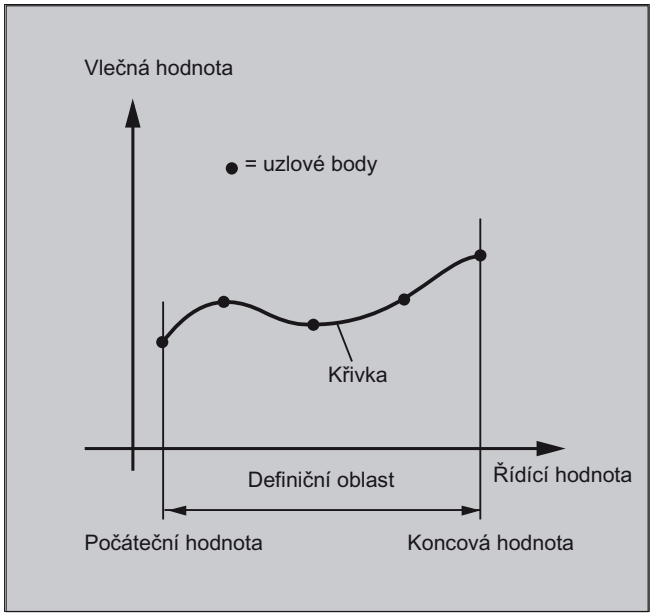


9.2.1 Definice tabulek křivek (CTABDEF, CATBEND)

Funkce

Tabulka křivky představuje výrobní program nebo úsek výrobního programu, který je na začátku uveden příkazem `CTABDEF` a ukončen závěrečným příkazem `CTABEND`.

V rámci tohoto úseku výrobního programu jsou prostřednictvím pohybových příkazů přiřazeny jednotlivým pozicím řídící osy polohy závislé osy, které se pak používají jako uzlové body pro výpočet aproximační křivky ve formě polynomu maximálně 5. stupně.



Předpoklady

Pro definici tabulky křivky musí být prostřednictvím nastavení odpovídajících konfiguračních strojních parametrů vyhrazen patřičný paměťový prostor (→ výrobce stroje!).

Syntaxe

```
CTABDEF(<vlečná osa>,<řídící osa>,<n>,<periodicita>[,<paměťové
místo>])...
CTABEND
```

Význam

`CTABDEF( )`	Začátek definice tabulky křivky
`CTABEND`	Konec definice tabulky křivky
`<vlečná osa>`	Osa, jejíž pohyb se má vypočítávat prostřednictvím tabulky křivky.
`<řídící osa>`	Osa, která poskytuje řídící hodnotu pro výpočet pohybu vlečné osy.
`<n>`	Číslo (**ID**) tabulky křivky Číslo tabulky křivky je jednoznačné a nezávisí na umístění v paměti. Ani ve statické, ani v dynamické paměti NC systému se nemohou nacházet tabulky se stejnými čísly.
`<periodicita>`	Periodicita tabulky `0` Tabulka není periodická (bude zpracována jen jednou, a to i v případě kruhových os) `1` Tabulka je vzhledem k řídící ose periodická. `2` Tabulka je periodická vzhledem k řídící ose i vlečné ose.
`<paměťové místo>`	Údaj místa v paměti (nepovinné) `"SRAM"` Tabulka křivky se ukládá ve **statické** paměti NC systému. `"DRAM"` Tabulka křivky se ukládá v **dynamické** paměti NC systému. **Upozornění:** Pokud pro tento parametr není naprogramována žádná hodnota, potom se použije standardní místo v paměti nastavené parametrem MD20905 $MC_CTAB_DEFAULT_MEMORY_TYPE.

Poznámka
Přepisování

Jestliže se při nové definici tabulky použije číslo (`<n>`) nějaké už existující tabulky křivky, tato tabulka křivky se přepíše (výjimka: tabulka křivky je součástí aktivní vazby mezi osami nebo je zablokována pomocí příkazu `CTABLOCK`). **Při přepisování tabulky se nevypisuje žádná odpovídající výstraha!**

Příklady

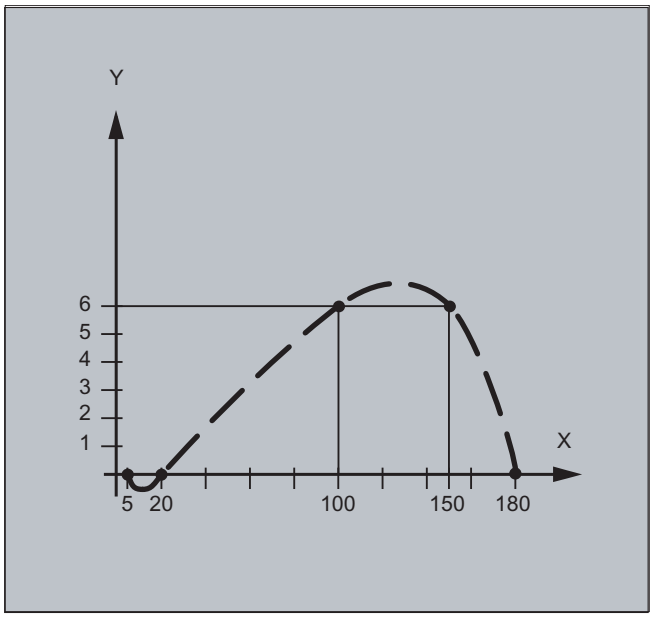
Příklad 1: Úsek programu jako definice tabulky křivky

Úsek programu má být použit nezměněn pro definici tabulky křivky. Příkaz pro zastavení předběžného zpracování `STOPRE`, který se v něm vyskytuje, může zůstat na svém místě a okamžitě se znovu aktivuje, jakmile se úsek programu přestane používat pro definici tabulky a jakmile je pomocí příkazů `CTABDEF` a `CTABEND` odstraněn.

```
Programový kód                  Komentář
…
CTABDEF(Y,X,1,1)                ; Definice tabulky křivky
…
IF NOT ($P_CTABDEF)
STOPRE
```

Programový kód	Komentář
ENDIF	
…	
CTABEND	

Příklad 2: Definice neperiodické tabulky křivky



Programový kód	Komentář
N100 CTABDEF(Y,X,3,0)	; Začátek definice neperiodické tabulky křivky s číslem 3.
N110 X0 Y0	; 1. pohybový příkaz, definuje počáteční hodnotu a 1. uzlové místo: Řídící hodnota: 0, závislá hodnota: 0
N120 X20 Y0	; 2. uzlové místo Řídící hodnota: 0…20, závislá hodnota: počáteční hodnota…0
N130 X100 Y6	; 3. uzlové místo Řídící hodnota: 20…100, závislá hodnota: počáteční hodnota: 0...6
N140 X150 Y6	; 4. uzlové místo Řídící hodnota: 100…150, závislá hodnota: počáteční hodnota: 6...6
N150 X180 Y0	; 5. uzlové místo Řídící hodnota: 150…180, závislá hodnota: počáteční hodnota: 6...0
N200 CTABEND	; Konec tabulky. Tabulka křivky se ve své interní reprezentaci převede na polynom maximálně 5. stupně. Výpočet křivkového průběhu se zadanými uzlovými body závisí na zvoleném modálním druhu interpolace (kruhová, lineární, splinová interpolace). Opět se obnoví stav výrobního programu před zahájením definice.

Příklad 3: Definice periodické tabulky křivky

Definice periodické tabulky křivky s číslem 2, rozmezí řídící hodnoty od 0 do 360, pohyb vlečné osy od 0 do 45 a zpátky do 0.

```
Programový kód                                          Komentář
N10 DEF REAL DEPPOS
N20 DEF REAL GRADIENT
N30 CTABDEF(Y,X,2,1)                                    ; Začátek definice.
N40 G1 X=0 Y=0
N50 POLY
N60 PO[X]=(45.0)
N70 PO[X]=(90.0) PO[Y]=(45.0,135.0,-90)
N80 PO[X]=(270.0)
N90 PO[X]=(315.0) PO[Y]=(0.0,-135.0,90)
N100 PO[X]=(360.0)
N110 CTABEND                                            ; Konec tabulky.
;Text křivky pomocí vazby mezi Y a X:
N120 G1 F1000 X0
N130 LEADON(Y,X,2)
N140 X360
N150 X0
N160 LEADOF(Y,X)

N170 DEPPOS=CTAB(75.0,2,GRADIENT)                       ; Načtení tabulkové funkce s řídící
                                                          hodnotou 75,0.
N180 G0 X75 Y=DEPPOS                                    ; Polohování řídící a vlečné osy.
;Po aktivování vazby už není žádná synchronizace vlečné osy zapotřebí.
N190 LEADON(Y,X,2)
N200 G1 X110 F1000
N210 LEADOF(Y,X)
N220 M30
```

Další informace

Počáteční a koncová hodnota tabulky křivky

Jako počáteční hodnota pro začátek definiční oblasti tabulky křivky platí první údaj k sobě patřících poloh os (první pohybový příkaz) uvnitř definice tabulky křivky. Koncová hodnota definiční oblasti tabulky křivky je analogicky určena prostřednictvím posledního pohybového příkazu.

Rozsah jazykových prostředků, které jsou k dispozici

V rámci definice tabulky křivky je k dispozici celý rozsah NC jazyka.

Poznámka

Následující příkazy jsou v definicích tabulek křivek nepřípustné:

- Zastavení předběžného zpracování
- Skokové pohyby řídící osy (např. při změně transformací)
- Příkaz pohybu pro samotnou závislou osu
- Změna směru pohybu řídící osy na opačný, tzn. poloha řídící osy musí být vždy jednoznačná.
- Příkazy `CTABDEF` a `CTABEND` na různých programových úrovních.

Platnost modálních příkazů

Veškeré příkazy s modální platností, které se vyskytují v rámci definice tabulky křivky, se stávají po ukončení definice tabulky neplatnými. Výrobní program, ve kterém se definice tabulky nachází, se takto nachází před a po definici tabulky ve stejném stavu.

Přiřazování R-Parametrů

Přiřazení R-parametrů v rámci definici tabulky se po příkazu `CTABEND` resetuje.

Příklad:

Programový kód	**Komentář**
`...`	
`R10=5 R11=20`	`; R10=5`
`...`	
`CTABDEF`	
`G1 X=10 Y=20 F1000`	
`R10=R11+5`	`; R10=25`
`X=R10`	
`CTABEND`	
`...`	`; R10=5`

Aktivování funkcí ASPLINE, BSPLINE, CSPLINE

Jestliže je v rámci definice tabulky křivky pomocí příkazů CTABDEF ... CTABEND aktivován některý z příkazů ASPLINE, BSPLINE oder CSPLINE, potom by měl být před tímto příkazem splinu naprogramován minimálně jeden počáteční bod. Okamžitému aktivování po příkazu CTABDEF byste se měli vyhnout, protože jinak by spline závisel na momentální poloze osy před definicí tabulky křivky.

Příklad:

Programový kód

```
...
CTABDEF(Y,X,1,0)
X0 Y0
ASPLINE
X=5 Y=10
X10 Y40
...
CTABEND
```

Opakované použití tabulky křivky

Funkční závislost mezi řídící a vlečnou osou, která byla vypočítána pomocí tabulky křivky, zůstává pod vybraným číslem tabulky i po skončení výrobního programu a po vypnutí a zapnutí systému (Power off), jestliže je ale tabulka uložena ve statické paměti NC systému.

Tabulka, která byla uložena v dynamické paměti (DRAM), se při vypnutí a zapnutí systému (power-on) vymaže a v případě potřeby musí být sestavena ještě jednou.

Jedenkrát sestavená tabulka křivky může být využita libovolnou kombinací řídící a vlečné osy a je nezávislá na tom, které osy byly pro její sestavení použity.

Přepisování tabulek křivek

Tabulka křivky se přepíše, jakmile je při definici tabulky znovu použito její číslo.

Výjimka: Tabulka křivky je aktivní v rámci vazby mezi osami nebo je blokována příkazem CTABLOCK.

Poznámka

Při přepisování tabulky křivky se nevypisuje žádná odpovídající výstraha!

Je definice tabulky křivky aktivní?

Prostřednictvím systémové proměnné $P_CTABDEF je možné z výrobního programu kdykoli zjistit, zda je definice tabulky křivky aktivní.

Ukončení definice tabulky křivky

Úsek výrobního programu je po vymezení definice tabulky křivky, které označují její začátek a konec, opět použitelný jako reálný výrobní program.

Načítání tabulek křivek pomocí funkce "Zpracování z externího zdroje"

Při zpracovávání tabulek křivek z externího zdroje musí být velikost používané vyrovnávací paměti (DRAM) nastavena pomocí parametru
MD18360 $MN_MM_EXT_PROG_BUFFER_SIZE tak, aby bylo možné do této vyrovnávací paměti uložit celou definici tabulky naráz. Zpracovávání výrobního programu se jinak přeruší a aktivuje se alarm.

Skokové změny vlečné osy

V závislosti na nastavení strojního parametru
MD20900 $MC_CTAB_ENABLE_NO_LEADMOTION
mohou být skokové změny vlečné osy v případě chybějícího pohybu řídící osy tolerovány.

9.2.2 Kontrola existence tabulky křivky (CTABEXISTS)

Funkce

Pomocí příkazu `CTABEXISTS` může být zkontrolováno, zda se tabulka s určitým číslem křivky vyskytuje v paměti NC systému.

Syntaxe

```
CTABEXISTS(<n>)
```

Význam

`CTABEXISTS`	Zkontroluje se, zda je tabulka křivky s číslem `<n>` k dispozici ve statické nebo dynamické paměti NC systému. 0 Tabulka neexistuje 1 Tabulka existuje
`<n>`	Číslo (**ID**) tabulky křivky

9.2.3 Mazání tabulek křivek (CTABDEL)

Funkce

Pomocí příkazu `CTABDEL` je možné mazat tabulky křivek.

Poznámka

Tabulky křivky, které jsou aktivní ve vazbě mezi osami, nemohou být vymazány.

Syntaxe

```
CTABDEL(<n>)
CTABDEL(<n>,<m>)
CTABDEL(<n>,<m>,<paměťové místo>)
CTABDEL()
CTABDEL(,,<paměťové místo>)
```

Význam

`CTABDEL`	Příkaz pro vymazání tabulky křivky
`<n>`	Číslo (**ID**) tabulky křivky, která má být vymazána Při mazání intervalu tabulek křivek pomocí příkazu `CTABDEL(<n>,<m>)` se pomocí parametru `<n>` zadává číslo první tabulky křivky tohoto intervalu.
`<m>`	Při mazání intervalu tabulek křivek pomocí příkazu `CTABDEL(<n>,<m>)` se pomocí parametru `<m>` zadává číslo poslední tabulky křivky tohoto intervalu. Parametr `<m>` musí být větší než `<n>`!
`<paměťové místo>`	Údaj místa v paměti (nepovinné) Při mazání **bez** zadání paměťového místa jsou uvedené tabulky křivek vymazány ze statické a dynamické paměti NC systému. Při mazání **se** zadáním paměťového místa jsou vymazány jen ty uvedené tabulky křivek, které se nacházejí v zadané paměti. Zbývající tabulky zůstanou nedotčeny. `"SRAM"` Mazání ve **statické** paměti NC systému `"DRAM"` Mazání v **dynamické** paměti NC systému

Pokud je naprogramován příkaz `CTABDEL` bez udání tabulky křivky, která má být vymazána, potom se vymažou **všechny** tabulky křivek, příp. všechny tabulky křivek z uvedené paměti.

`CTABDEL()`	Vymažou se všechny tabulky křivek ve statické a dynamické paměti NC systému.
`CTABDEL(,,"SRAM")`	Vymažou se všechny tabulky křivek ve statické paměti NC systému.
`CTABDEL(,,"DRAM")`	Vymažou se všechny tabulky křivek v dynamické paměti NC systému.

Poznámka

Pokud je v případě mazání většího počtu křivek pomocí příkazů `CTABDEL(<n>,<m>)` nebo `CTABDEL()` alespoň jedna z mazaných tabulek křivek součástí aktivní vazby, potom se příkaz mazání neuskuteční, tzn. nevymaže se **žádná** ze zadaných tabulek křivek.

9.2.4 Blokování tabulek křivek proti mazání a přepisování (CTABLOCK, CTABUNLOCK)

Funkce

Tabulky křivek mohou být aktivováním blokování chráněny proti nechtěnému vymazání a přepsání. Aktivované blokování může být kdykoli opět zrušeno.

Syntaxe

Aktivování blokování:

```
CTABLOCK(<n>)
CTABLOCK(<n>,<m>)
CTABLOCK(<n>,<m>,<paměťové místo>)
CTABLOCK()
CTABLOCK(,,<paměťové místo>)
```

Deaktivování blokování:

```
CTABUNLOCK(<n>)
CTABUNLOCK(<n>,<m>)
CTABUNLOCK(<n>,<m>,<paměťové místo>)
CTABUNLOCK()
CTABUNLOCK(,,<paměťové místo>)
```

Význam

`CTABLOCK`	Příkaz pro **aktivování** blokování proti vymazání/přepsání
`CTABUNLOCK`	Příkaz pro **deaktivování** blokování proti vymazání/přepsání Příkazem `CTABUNLOCK` se tabulky křivek zablokované příkazem `CTABLOCK` opět odblokují. Tabulky křivky, které se uplatňují v aktivní vazbě, zůstanou i nadále zablokovány a nemohou být vymazány. Blokování příkazem `CTABLOCK` je odstraněno, jakmile pomine ochrana vyvolávaná aktivní vazbou v důsledku toho, že je vazba deaktivována. Potom může být tato tabulka vymazána. Opětovné vyvolání příkazu `CTABUNLOCK` není zapotřebí.
`<n>`	Číslo (**ID**) tabulky křivky, která má být blokována/odblokována Při aktivování/deaktivování blokování intervalu tabulek křivek pomocí příkazů `CTABLOCK(<n>,<m>)`/`CTABUNLOCK(<n>,<m>)` se pomocí parametru `<n>` zadává číslo první tabulky křivky tohoto intervalu.
`<m>`	Při aktivování/deaktivování blokování intervalu tabulek křivek pomocí příkazů `CTABLOCK(<n>,<m>)`/`CTABUNLOCK(<n>,<m>)` se pomocí parametru `<m>` zadává číslo poslední tabulky křivky tohoto intervalu. Parametr `<m>` musí být větší než `<n>`!

<paměťové místo>	Údaj místa v paměti (nepovinné) Při aktivování/deaktivování blokování **bez** zadání paměťového místa jsou uvedené tabulky křivek blokovány/odblokovány ve statické a dynamické paměti NC systému. Při aktivování/deaktivování blokování **se** zadáním paměťového místa jsou blokovány/odblokovány jen ty uvedené tabulky křivek, které se nacházejí v zadané paměti. Zbývajících se blokování/odblokování netýká. "SRAM" Aktivování/deaktivování blokování ve **statické** paměti NC systému "DRAM" Aktivování/deaktivování blokování v **dynamické** paměti NC systému

Pokud je naprogramován příkaz CTABLOCK/CTABUNLOCK bez udání tabulky křivky, jejíž blokování má být aktivováno/deaktivováno, potom se aktivování/deaktivování blokování týká **všech** tabulek křivek, příp. všech tabulek křivek z uvedené paměti.

CTABLOCK()	Zablokují se všechny tabulky křivek ve statické a dynamické paměti NC systému.
CTABLOCK(,,"SRAM")	Zablokují se všechny tabulky křivek ve statické paměti NC systému.
CTABLOCK(,,"DRAM")	Zablokují se všechny tabulky křivek v dynamické paměti NC systému.
CTABUNLOCK()	Odblokují se všechny tabulky křivek ve statické a dynamické paměti NC systému.
CTABUNLOCK(,,"SRAM")	Odblokují se všechny tabulky křivek ve statické paměti NC systému.
CTABUNLOCK(,,"DRAM")	Odblokují se všechny tabulky křivek v dynamické paměti NC systému.

9.2.5 Tabulky křivek: Zjišťování vlastností tabulek (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD)

Funkce

Pomocí těchto příkazů mohou být zjišťovány důležité vlastnosti tabulek křivek (číslo tabulka, stav blokování, místo uložení, periodicita).

Syntaxe

```
CTABID(<p>)
CTABID(<p>,<paměťové místo>)
CTABISLOCK(<n>)
CTABMEMTYP(<n>)
TABPERIOD(<n>)
```

Význam

`CTABID` Výsledkem je **číslo tabulky**, která je v zadané paměti uložena jako `<p>`-tá tabulka křivky.

Příklad:

Výsledkem příkazu `CTABID(1,"SRAM")` bude číslo první tabulky křivky ve statické paměti NC systému. První tabulka křivky přitom odpovídá tabulce křivky s nejvyšším číslem tabulky.

Upozornění:
Jestliže se mezi dvěma po sobě následujícími voláními funkce `CTABID` posloupnost tabulek křivek v paměti změní, např. v důsledku vymazání tabulek křivek pomocí příkazu `CTABDEL`, může se stát, že výsledkem příkazu `CTABID(<p>,...)` se stejným číslem `<p>` bude jiná tabulka křivky než předtím.

`CTABISLOCK` Výsledkem je **stav blokování** tabulky křivky s číslem `<n>`.

0	Tabulka není blokována
1	Tabulka je blokována příkazem `CTABLOCK`
2	Tabulka je blokována díky aktivní vazbě
3	Tabulka je blokována příkazem `CTABLOCK` a díky aktivní vazbě
-1	Tabulka neexistuje

`CTABMEMTYP` Výsledkem je **místo uložení** tabulky křivky s číslem `<n>`.

0	Tabulka ve statické paměti NC systému
1	Tabulka v dynamické paměti NC systému
-1	Tabulka neexistuje

`CTABPERIOD` Výsledkem je **periodicita** tabulky křivky s číslem `<n>`.

0	Tabulka není periodická
1	Tabulka je vzhledem k řídící ose periodická.
2	Tabulka je periodická vzhledem k řídící ose i vlečné ose.
-1	Tabulka neexistuje

`<p>` Číslo položky v paměti

`<n>` Číslo (**ID**) tabulky křivky

`<paměťové místo>` Údaj místa v paměti (nepovinné)

`"SRAM"`	**statická** paměť NC systému
`"DRAM"`	**dynamická** paměť NC systému

Upozornění:
Pokud pro tento parametr není naprogramována žádná hodnota, potom se použije standardní místo v paměti nastavené parametrem MD20905 $MC_CTAB_DEFAULT_MEMORY_TYPE.

9.2.6 Čtení hodnot z tabulky křivky (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX)

Funkce

Ve výrobním programu mohou být čteny následující hodnoty z tabulky křivky:

- Hodnoty vlečné a řídící osy na začátku a na konci tabulky křivky
- Hodnoty vlečné osy na začátku a na konci úseku křivky
- Hodnota vlečné osy pro určitou hodnotu řídící osy
- Hodnota řídící osy pro určitou hodnotu vlečné osy
- Minimální a maximální hodnota vlečné osy
  - v celkovém definičním rozsahu tabulky křivky
    nebo
  - v definovaném intervalu tabulky křivky

Syntaxe

```
CTABTSV(<n>,<gradient>[,<vlečná osa>])
CTABTEV(<n>,<gradient>[,<vlečná osa>])
CTABTSP(<n>,<gradient>[,<řídící osa>])
CTABTEP(<n>,<gradient>[,<řídící osa>])
CTABSSV(<řídící hodnota>,<n>,<gradient>[,<vlečná osa>])
CTABSEV(<řídící hodnota>,<n>,<gradient>[,<vlečná osa>])
CTAB(<řídící hodnota>,<n>,<gradient>[,<vlečná osa>,<řídící osa>]
CTABINV(<vlečná hodnota>,<hodnota
přiblížení>,<n>,<gradient>[,<vlečná osa>,<řídící osa>]
CTABTMIN(<n>[,<vlečná osa>])
CTABTMAX(<n>[,<vlečná osa>])
CTABTMIN(<n>,<a>,<b>[,<vlečná osa>,<řídící osa>])
CTABTMAX(<n>,<a>,<b>[,<vlečná osa>,<řídící osa>])
```

Význam

CTABTSV:	Načtení hodnoty vlečné osy na **začátku** tabulky křivky s číslem <n>.
CTABTEV:	Načtení hodnoty vlečné osy na **konci** tabulky křivky s číslem <n>.
CTABTSP:	Načtení hodnoty řídící osy na **začátku** tabulky křivky s číslem <n>.
CTABTEP:	Načtení hodnoty řídící osy na **konci** tabulky křivky s číslem <n>.
CTABSSV:	Načtení hodnoty vlečné osy na **začátku** úseku křivky, který patří k zadané hodnotě řídící osy (<řídící hodnota>)
CTABSEV:	Načtení hodnoty vlečné osy na **konci** úseku křivky, který patří k zadané hodnotě řídící osy (<řídící hodnota>)
CTAB:	Načtení hodnoty vlečné osy k zadané hodnotě řídící osy (<řídící hodnota>)

CTABINV:	Načtení hodnoty řídící osy k zadané hodnotě vlečné osy (<vlečná hodnota>)
CTABTMIN:	Stanovení **minimální hodnoty** vlečné osy: • v celkovém definičním rozsahu tabulky křivky nebo • v definovaném intervalu <a> ... <b>
CTABTMAX:	Stanovení **maximální hodnoty** vlečné osy: • v celkovém definičním rozsahu tabulky křivky nebo • v definovaném intervalu <a> ... <b>
<n>:	Číslo (**ID**) tabulky křivky
<gradient>:	V parametru <gradient> systém poskytuje údaj o **stoupání** funkce popisované tabulkou křivky v bodě, který je předmětem zájmu
<vlečná osa>:	Osa, jejíž pohyb se má vypočítávat prostřednictvím tabulky křivky (nepovinné)
<řídící osa>:	Osa, která poskytuje řídící hodnotu pro výpočet pohybu vlečné osy (nepovinné)
<vlečná hodnota>:	Hodnota vlečné osy pro načtení k ní náležící hodnoty řídící osy pomocí příkazu CTABINV
<řídící hodnota>:	Hodnota řídící osy • pro načtení k ní náležící hodnoty vlečné osy pomocí příkazu CTAB nebo • pro volbu úseku křivky pomocí příkazu CTABSSV/CTABSEV
<hodnota přiblížení>:	Přiřazení hodnoty řídící osy hodnotě vlečné osy u příkazu CTABINV nemusí být vždy jednoznačné. Příkaz CTABINV potřebuje proto jako parametr hodnotu, která se blíží očekávané hodnotě řídící osy.
<a>:	**Spodní** mezní hodnota intervalu řídících hodnot u příkazů CTABTMIN/CTABTMAX
<b>:	**Horní** mezní hodnota intervalu řídících hodnot u příkazů CTABTMIN/CTABTMAX **Upozornění:** Interval řídících hodnot <a> ... <b> musí ležet v rámci definičního rozsahu tabulky křivky.

Příklady

Příklad 1:

Stanovení hodnot vlečné osy a řídící osy na začátku a na konci tabulky křivky, jakož i maximální a minimální hodnoty vlečné osy v celém definičním rozsahu tabulky křivky.

Programový kód	Komentář
N10 DEF REAL STARTPOS	
N20 DEF REAL ENDPOS	
N30 DEF REAL STARTPARA	
N40 DEF REAL ENDPARA	
N50 DEF REAL MINVAL	
N60 DEF REAL MAXVAL	
N70 DEF REAL GRADIENT	
...	
N100 CTABDEF(Y,X,1,0)	; Začátek definice tabulky
N110 X0 Y10	; Počáteční pozice 1. úseku tabulky
N120 X30 Y40	; Koncová pozice 1. úseku tabulky = počáteční pozice 2. úseku tabulky
N130 X60 Y5	; Koncová pozice 2. úseku tabulky = ...
N140 X70 Y30	
N150 X80 Y20	
N160 CTABEND	; Konec definice tabulky.
...	
N200 STARTPOS=CTABTSV(1,GRADIENT)	; Hodnota vlečné osy na začátku tabulky křivky = 10
N210 ENDPOS=CTABTEV(1,GRADIENT)	; Hodnota vlečné osy na konci tabulky křivky = 20
N220 STARTPARA=CTABTSP(1,GRADIENT)	; Hodnota řídící osy na začátku tabulky křivky = 0
N230 ENDPARA=CTABTEP(1,GRADIENT)	; Hodnota řídící osy na konci tabulky křivky = 80
N240 MINVAL=CTABTMIN(1)	; Minimální hodnota vlečné osy v bodě Y=5
N250 MAXVAL=CTABTMAX(1)	; Maximální hodnota vlečné osy v bodě Y=40

Příklad 2:

Stanovení hodnoty vlečné osy na začátku a na konci úseku křivky, který patří k hodnotě řídící osy X=30.

Programový kód	Komentář
N10 DEF REAL STARTPOS	
N20 DEF REAL ENDPOS	
N30 DEF REAL GRADIENT	
...	
N100 CTABDEF(Y,X,1,0)	; Začátek definice tabulky.
N110 X0 Y0	; Počáteční pozice 1. úseku tabulky
N120 X20 Y10	; Koncová pozice 1. úseku tabulky = počáteční pozice 2. úseku tabulky
N130 X40 Y40	Koncová pozice 2. úseku tabulky = ...
N140 X60 Y10	
N150 X80 Y0	
N160 CTABEND	; Konec definice tabulky.
...	
N200 STARTPOS=CTABSSV(30.0,1,GRADIENT)	; Počáteční pozice Y ve 2. úseku = 10
N210 ENDPOS=CTABSEV(30.0,1,GRADIENT)	; Koncová pozice Y ve 2. úseku = 40

Další informace

Použití v synchronních akcích

Všechny příkazy pro čtení hodnot z tabulky křivky se mohou používat také v synchronních akcích (viz také kapitola "Pohybové synchronní akce").

Při použití příkazů `CTABINV`, `CTABTMIN` a `CTABTMAX` je zapotřebí mít na paměti, aby:

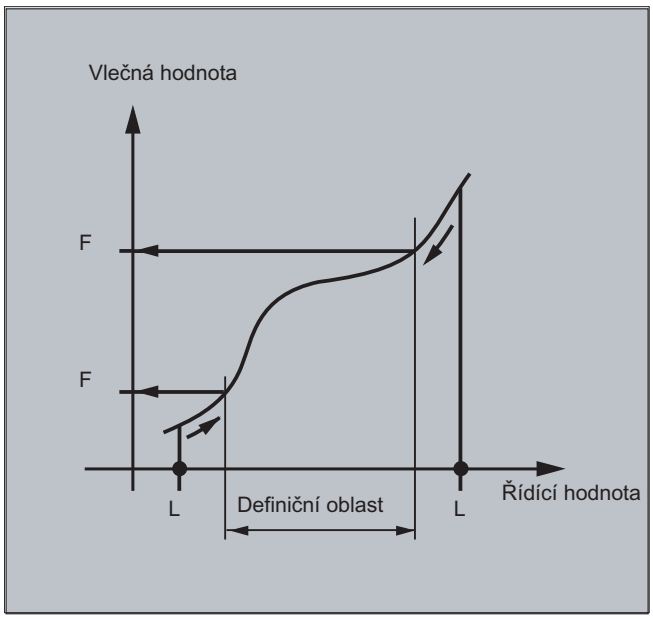
- v okamžiku zpracování těchto příkazů byl k dispozici dostatečný výkon NC systému

  nebo

- aby před vyvoláním příkazu byl zjištěn počet úseků v tabulce křivky, aby bylo možné v případě potřeby příslušnou tabulku rozdělit

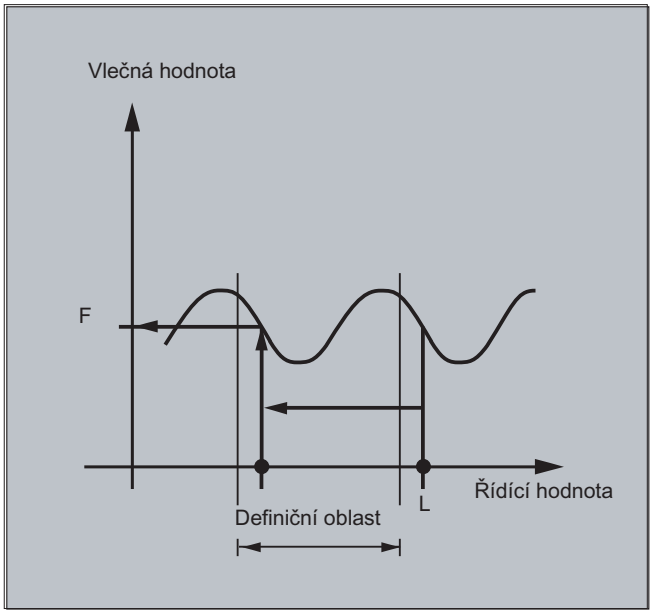
Příkaz CTAB v neperiodických tabulkách křivky

Jestliže hodnota zadaná v parametru `<řídící hodnota>` leží mimo definiční rozsah, bude jako vlečná hodnota uvedena horní, příp. dolní mezní hodnota:



Příkaz CTAB v periodických tabulkách křivky

Jestliže hodnota zadaná v parametru <řídící hodnota> leží mimo definiční rozsah, bude řídící hodnota upravena funkcí Modulo pro definiční rozsah a potom se zjistí odpovídající vlečná hodnota:



Hodnota přiblížení pro příkaz CTABINV

Příkaz CTABINV potřebuje hodnotu, která se blíží očekávané řídící hodnotě. Výsledkem funkce CTABINV je řídící hodnota, která leží nejblíže hodnotě přiblížení. Hodnotou přiblížení může být např. řídící hodnota z předcházejícího interpolačního taktu.

Stoupání funkce dané tabulkou křivky

Zadání stoupání pomocí parametru <gradient>) umožňuje započítat rychlost řídící nebo vlečné osy na odpovídajícím místě charakteristiky.

Zadání řídící nebo vlečné osy

Nepovinné zadání řídící a/nebo vlečné osy je důležité, jestliže jsou pro řídící a vlečnou osu v konfiguraci nastaveny různé jednotky délky.

CTABSSV, CTABSEV

Příkazy CTABSSV a CTABSEV**nejsou** v následujících případech vhodné k tomu, aby se zjišťovaly nějaké hodnoty z naprogramovaných úseků:

- Jsou naprogramovány kruhy nebo evolventy.
- Jsou aktivní fasety, příp. zaoblení pomocí příkazů CHF/RND.
- Jsou aktivní přechodová zaoblení aktivovaná příkazem G643.
- Pomocí příkazů COMPON/COMPCURV/COMPCAD je aktivní komprese NC bloků.

9.2.7 Tabulky křivek: Kontrola využití systémových zdrojů (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL)

Funkce

Pomocí těchto příkazů má programátor možnost získat aktuální informace o využití systémových zdrojů pro tabulky křivek, úseky tabulek a polynomy.

Syntaxe

```
CTABNO
CTABNOMEM(<paměťové místo>)
CTABFNO(<paměťové místo>)
CTABSEGID(<n>,<paměťové místo>)
CTABSEG(<paměťové místo>,<druh úseku>)
CTABFSEG(<paměťové místo>,<druh úseku>)
CTABMSEG(<paměťové místo>,<druh úseku>)
CTABPOLID(<n>)
CTABPOL(<paměťové místo>)
CTABFPOL(<paměťové místo>)
CTABMPOL(<paměťové místo>)
```

Význam

CTABNO	Stanovení celkového počtu **definovaných** tabulek křivek (ve statické a dynamické paměti NC systému)
CTABNOMEM	Stanovení počtu **definovaných** tabulek křivek v zadané paměti <paměťové místo>
CTABFNO	Stanovení počtu **ještě možných** tabulek křivek v zadané paměti <paměťové místo>
CTABSEGID	Stanovení počtu křivkových úseků odpovídajících zadanému <druhu úseku>, které jsou využívány tabulkou křivky s číslem <n>
CTABSEG	Stanovení počtu **používaných** úseků křivky odpovídajících zadanému <druhu úseku>, které jsou uloženy v zadané paměti <paměťové místo>
CTABFSEG	Stanovení počtu **ještě možných** úseků křivky odpovídajících zadanému <druhu úseku>, které jsou uloženy v zadané paměti <paměťové místo>
CTABMSEG	Stanovení počtu **maximálně možných** úseků křivky odpovídajících zadanému <druhu úseku>, které jsou uloženy v zadané paměti <paměťové místo>
CTABPOLID	Stanovení počtu křivkových polynomů, které jsou využívány tabulkou křivky s číslem <n>
CTABPOL	Stanovení počtu **používaných** křivkových polynomů v zadané paměti <paměťové místo>

CTABFPOL	Stanovení počtu **ještě možných** křivkových polynomů v zadané paměti <paměťové místo>
CTABMPOL	Stanovení počtu **maximálně možných** křivkových polynomů v zadané paměti <paměťové místo>
<n>	Číslo (**ID**) tabulky křivky
<paměťové místo>	Údaj místa v paměti (nepovinné) "SRAM" **statická** paměť NC systému "DRAM" **dynamická** paměť NC systému **Upozornění:** Pokud pro tento parametr není naprogramována žádná hodnota, potom se použije standardní místo v paměti nastavené parametrem MD20905 $MC_CTAB_DEFAULT_MEMORY_TYPE.
<druh úseku>	Zadání druhu úseku (nepovinné) "L" Lineární úseky "P" Polynomický úsek **Upozornění:** Jestliže pro tento parametr není naprogramována žádná hodnota, bude výsledkem součet lineárních a polynomikkých úseků.

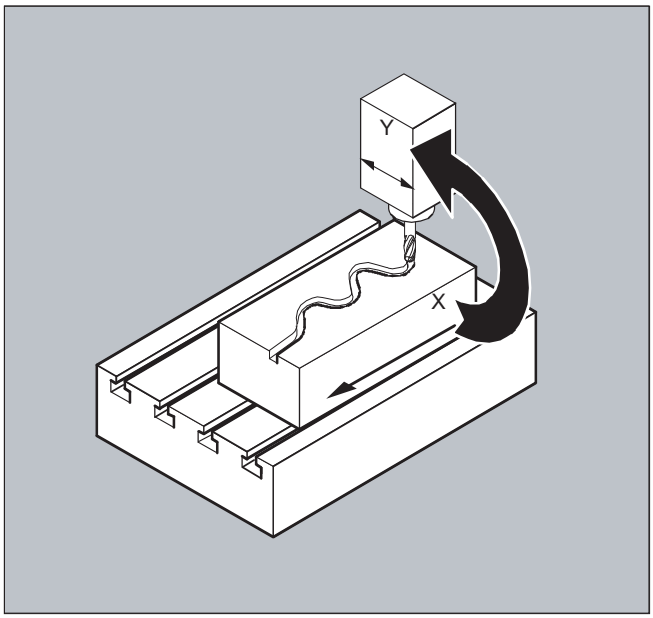
9.3 Osová vazba řídící hodnotou (LEADON, LEADOF)

Poznámka

U systému SINUMERIK 828D není tato funkce k dispozici!

Funkce

V případě osové vazby řídící hodnotou se řídící a vlečná osa pohybují synchronně. Příslušná poloha vlečné osy je přitom prostřednictvím tabulky křivky, příp. z ní vypočítaného polynomu jednoznačně přiřazena poloze - v případě potřeby simulované - řídící osy.



Řídící osou je ta osa, která poskytuje vstupní hodnoty pro tabulku křivky. **Vlečnou osou** je nazývána osa, která zaujímá polohy vypočítané prostřednictvím tabulky křivky.

Vazba pomocí skutečné a požadované hodnoty

Jako řídící hodnoty se mohou používat také výstupní hodnoty pro zjišťování polohy vlečné osy.

- Skutečné hodnoty polohy řídící osy: Vazba pomocí skutečné hodnoty
- Požadované hodnoty polohy řídící osy: Vazba pomocí požadované hodnoty

Vazba pomocí řídící hodnoty platí vždy v základním souřadném systému.

Pokud budete potřebovat informace o tabulkách křivek, viz kapitola "Tabulky křivek".
Pokud budete potřebovat informace o vazbě pomocí řídící hodnoty, viz /FB/, M3, Vlečení a vazba řídící hodnotou.

Syntaxe

```
LEADON(vlečná osa,řídící osa,n)
LEADOF(vlečná osa,řídící osa)
```

nebo zrušení příkazem bez uvedení řídící osy:

```
LEADOF(vlečná osa)
```

Vazba řídící hodnotou může být aktivována a deaktivována jak z výrobního programu, tak také v průběhu pohybu ze synchronních akcí, viz kapitola "Pohybové synchronní akce".

Význam

`LEADON`	Aktivování vazby řídící hodnotou
`LEADOF`	Deaktivování vazby řídící hodnotou
`Vlečná osa`	Vlečná osa
`Řídící osa`	Řídící osa
`n`	Číslo tabulky křivky
`$SA_LEAD_TYPE`	Přepnutí mezi vazbou pomocí požadované a skutečné hodnoty

Deaktivování vazby řídící hodnotou, LEADOF

Po deaktivování vazby řídící hodnotou se vlečná osa znovu stává normální příkazovou osou!

Osová vazba řídící hodnotou a různé provozní stavy, RESET

V závislosti na nastavení ve strojním parametru se vazby řídící hodnotou při RESETu vypínají.

Příklad vazby řídící hodnotou ze synchronní akce

U lisovacího zařízení má být obvyklá mechanická vazba mezi řídící osou (hřídel razníku) a osami podávacího systému, který se skládá z os podavače a pomocných os, nahrazena systémem elektronických vazeb.

Na tomto příkladu je ukázáno, jak se u lisovacího zařízení nahrazuje mechanický podávací systém elektronickým podávacím systémem. Operace vytváření a rozpojování vazeb jsou realizovány jako **statické synchronní akce**.

Na základě řídící osy LW (osa razníku) jsou definovaným způsobem prostřednictvím tabulek křivek řízeny osy podavače a pomocné osy jako vlečné osy.

Vlečné osy

X posuv, příp podélná osa
YL zavírání, příp. příčná osa
ZL osa zdvihu
U posuv válce, pomocná osa
V vyrovnávací hlava, pomocná osa
W mazání, pomocná osa

Akce

V rámci synchronních akcí se uskutečňují např. následující činnosti:

- Navázání vazby, `LEADON(vlečná osa, řídící osa, číslo tabulky křivky)`
- Rozpojení vazby, `LEADOF(vlečná osa, řídící osa)`
- Nastavení skutečné hodnoty, `PRESETON(osa, hodnota)`
- Nastavení značky, `$AC_MARKER[i] = hodnota`
- Druh vazby: reálná/virtuální řídící hodnota
- Najíždění osami na pozice, `POS[osa] = hodnota`

Podmínky

Jako podmínky jsou vyhodnocovány rychlé digitální vstupy, proměnné v reálném čase `$AC_MARKER` a porovnávání pozic, které jsou spojeny pomocí logického operátoru AND.

Poznámka

V následujícím příkladu byly řádkování, řazení bloků a bloky **mazání** použity výlučně k tomu účelu, aby se zvýšila čitelnost programu. Z hlediska řídícího systému je všechno pod jedním číslem bloku považováno za jeden řádek.

Komentář

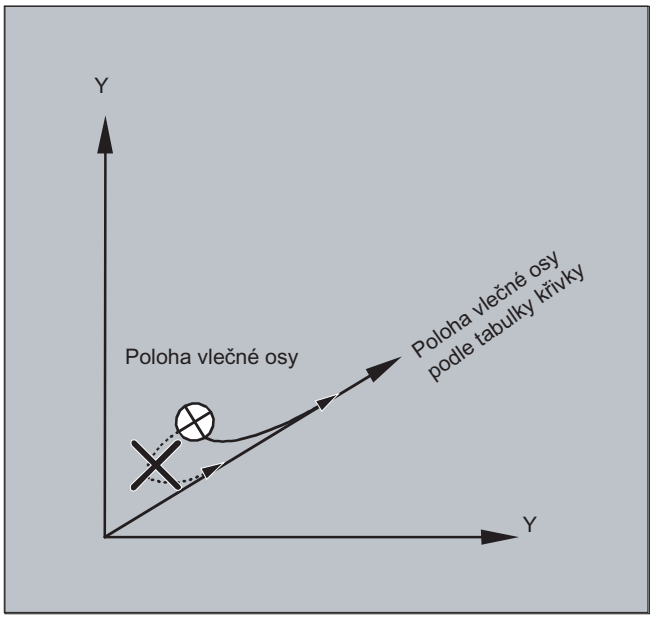
Programový kód	Komentář
	; Definuje veškeré statické synchronní akce.
	; ****Vynulování značek
N2 $AC_MARKER[0]=0 $AC_MARKER[1]=0 $AC_MARKER[2]=0 $AC_MARKER[3]=0 $AC_MARKER[4]=0 $AC_MARKER[5]=0 $AC_MARKER[6]=0 $AC_MARKER[7]=0	
	; **** E1 0=>1 Zapnutí vazby podavače
N10 IDS=1 EVERY ($A_IN[1]==1) AND ($A_IN[16]==1) AND ($AC_MARKER[0]==0) DO LEADON(X,LW,1) LEADON(YL,LW,2) LEADON(ZL,LW,3) $AC_MARKER[0]=1	
	; **** E1 0=>1 Zapnutí vazby posuvu válce
N20 IDS=11 EVERY ($A_IN[1]==1) AND ($A_IN[5]==0) AND ($AC_MARKER[5]==0) DO LEADON(U,LW,4) PRESETON(U,0) $AC_MARKER[5]=1	
	; **** E1 0->1 Zapnutí vazby vyrovnávací hlavy
N21 IDS=12 EVERY ($A_IN[1]==1) AND ($A_IN[5]==0) AND ($AC_MARKER[6]==0) DO LEADON(V,LW,4) PRESETON(V,0) $AC_MARKER[6]=1	
	; **** E1 0->1 Zapnutí vazby mazání
N22 IDS=13 EVERY ($A_IN[1]==1) AND ($A_IN[5]==0) AND ($AC_MARKER[7]==0) DO LEADON(W,LW,4) PRESETON(W,0) $AC_MARKER[7]=1	
	; **** E2 0=>1 Vypnutí vazby
N30 IDS=3 EVERY ($A_IN[2]==1) DO LEADOF(X,LW) LEADOF(YL,LW) LEADOF(ZL,LW) LEADOF(U,LW) LEADOF(V,LW) LEADOF(W,LW) $AC_MARKER[0]=0 $AC_MARKER[1]=0 $AC_MARKER[3]=0 $AC_MARKER[4]=0 $AC_MARKER[5]=0 $AC_MARKER[6]=0 $AC_MARKER[7]=0	
....	
N110 G04 F01	
N120 M30	

Popis

Vazba řídící hodnotou vyžaduje synchronizaci řídící a vlečné osy. Této synchronizace může být dosaženo jen tehdy, pokud se vlečná osa při zapnutí vazby řídící hodnotou nachází v rámci toleranční oblasti průběhu kontury vypočítaného z tabulky křivky.

Toleranční oblast pro polohu vlečné osy je definována prostřednictvím strojního parametru `MD 37200: COUPLE_POS_POL_COARSE A_LEAD_TYPE`.

Jestliže se vlečná osa v okamžiku zapnutí vazby řídící hodnotou ještě nenachází na odpovídající pozici, automaticky se uskuteční synchronizační operace, která zajistí, aby se vypočítaná požadovaná poloha pro vlečnou osu blížila její skutečné poloze. Vlečná osa se přitom v průběhu synchronizační operace pohybuje ve směru, který je definován prostřednictvím požadované hodnoty rychlosti vlečné osy (vypočítá se na základě rychlosti řídící osy a podle tabulky křivky CTAB).

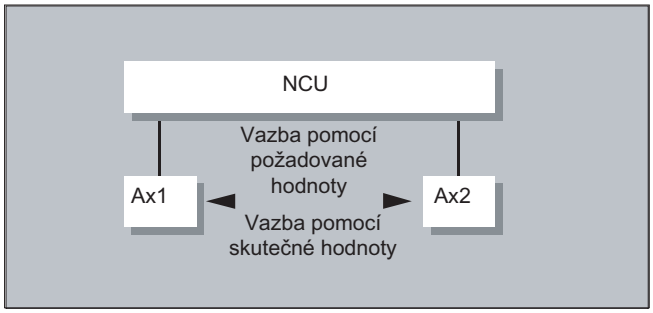


Žádná synchronizační operace

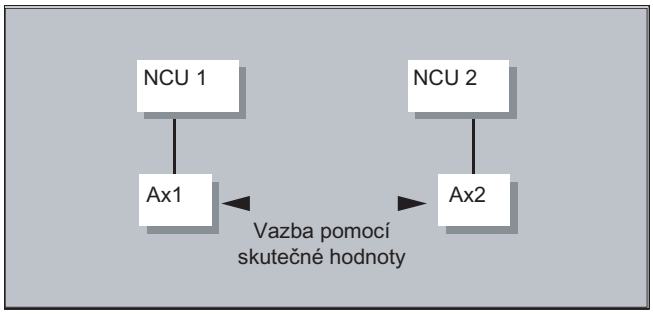
Jestliže se vypočítaná požadovaná poloha vlečné osy při zapnutí vazby řídící hodnotou vzdaluje od aktuální polohy vlečné osy, žádná synchronizační operace se neuskutečňuje.

Vazba pomocí skutečné a požadované hodnoty

Vazba požadovanou hodnotou poskytuje ve srovnání s vaznou pomocí skutečné hodnoty lepší synchronizaci mezi řídící a vlečnou osou a je proto předem nastavena jako standardní.



Vazba pomocí požadované hodnoty je možná jedině tehdy, pokud interpolace řídící a vlečné probíhá ve stejné NCU. V případě externí řídící osy může být vytvořena vazba mezi touto řídící osou a vlečnou osou jen pomocí skutečné hodnoty.



Přepnutí je možné uskutečnit pomocí nastavovaného parametru `$SA_LEAD_TYPE`.

Přepnutí mezi vazbou pomocí skutečné a požadované hodnoty by se mělo vždy provádět, když je vlečná osa v klidu. Důvodem je, že po přepnutí proběhne nová synchronizace jen v klidovém stavu.

Příklad použití

Může se stát, že při velkých otřesech stroje nebude čtení skutečné hodnoty bezchybné. Při použití vazby řídící hodnotou se v případě podavačů u lisů může ukázat jako nezbytné v pracovních krocích s velkými otřesy přepnout z vazby pomocí skutečné hodnoty na vazbu pomocí požadované hodnoty.

Simulace řídící hodnoty ve vazbách pomocí požadované hodnoty

Pomocí strojního parametru je možné oddělit interpolátor pro řídící osu od příslušného servomechanismu. Díky tomu je potom možné v případě vazby pomocí požadované hodnoty vytvořit požadovanou hodnotu, aniž by se řídící osa ve skutečnosti pohybovala.

Řídící hodnoty, které jsou vytvořeny prostřednictvím vazby pomocí požadované hodnoty a které se využívají např. v synchronních akcích, je možné načítat z následujících proměnných:

```
- $AA_LEAD_P                                    Řídící hodnota polohy
- $AA_LEAD_V                                    Řídící hodnota rychlosti
```

Vytváření řídících hodnot

Řídící hodnoty mohou být generovány pomocí libovolných posuvů, které jste sami naprogramovali. Tyto řídící hodnoty vytvořené tímto způsobem se zapisují do proměnných

```
- $AA_LEAD_SP                                    Řídící hodnota polohy
- $AA_LEAD_SV                                    Řídící hodnota rychlosti
```

a je možné je odtud také číst. Aby bylo možné tyto proměnné využívat, musí být nastavovanému parametru přiřazena hodnota `$SA_LEAD_TYPE = 2`.

Stav vazby

Ve výrobním programu v NC systému můžete pomocí následujících systémových proměnných zjistit stav vazby:

`$AA_COUP_ACT[osa]`
0: Žádná vazba není aktivní
16: Vazba řídící hodnotou je aktivní

Správa stavových proměnných při synchronních akcích

Spínací operace a operace s vazbami jsou spravovány pomocí proměnných v reálném čase:

`$AC_MARKER[i] = n`
, přičemž:
i = číslo značky
n = stavová hodnota

9.4 Elektronická převodovka (EG)

Funkce

Pomocí funkce "Elektronická převodovka" je možné ovládat pohyb jedné **vlečné osy** po lineárním pohybovém bloku v závislosti na až pěti **řídících osách**. Vzájemné závislosti mezi řídícími osami a vlečnou osou jsou pro každou řídící osu zvlášť definovány pomocí faktoru vazby.

Vypočítaná pohybová složka vlečné osy se vypočítává součtem pohybových složek jednotlivých lineárních os vynásobených příslušnými faktory vazby. Při aktivování svazku os typu "Elektronická převodovka" se může ukázat jako nezbytné synchronizovat vlečnou osu na určitou definovanou pozici. S převodovou vazbou je možné ve výrobním programu uskutečňovat následující operace:

- definice
- aktivování
- deaktivování
- vymazání.

Podle Vašeho přání může být pohyb vlečné osy odvozen od následujících veličin:

- Požadované hodnoty řídících os
- Skutečné hodnoty řídících os

Jako rozšíření mohou být prostřednictvím **tabulek křivek** (viz kapitola "Chování při pohybu po dráze) realizovány také nelineární vztahy mezi řídícími osami a vlečnou osou. Elektronické převodovky umožňují kaskádovité řazení, tzn. vlečná osa jedné elektronické převodovky se může stát řídící osou pro další elektronickou převodovku.

9.4.1 Definice elektronické převodovky (EGDEF)

Funkce

Svazek os typu "Elektronická převodovka" je definován stanovením vlečné osy a minimálně jedné, avšak nejvýše pěti řídících os s příslušnými faktory vazby.

Předpoklady

Předpoklad pro definici svazku os typu "Elektronická převodovka":

Pro vlečnou osu nesmí být definována žádná další vazba mezi osami (v případě potřeby musí předem být předcházející vazba vymazána pomocí příkazu `EGDEL`).

Syntaxe

```
EGDEF(vlečná osa,řídící osa1,typ vazby1,řídící osa2,typ vazby2,...)
```

Význam

`EGDEF`	Definice elektronické převodovky
`Vlečná osa`	Osa, která je řídícími osami ovlivňována

`řídící osa1`
`,...,`
`řídící osa5`
	Osy, které ovlivňují vlečnou osu

`typ vazby1`
`,...,`
`typ vazby5`
	Typ vazby
	Typ vazby nemusí být pro všechny řídící osy stejný a proto je potřeba jej zadat pro každou řídící osu zvlášť.

Hodnota:	Význam:
0	Vlečná osa je ovlivňována **skutečnou hodnotou** odpovídající řídící osy.
1	Vlečná osa je ovlivňována **požadovanou hodnotou** odpovídající řídící osy.

Poznámka

Při definici svazku os typu "Elektronická převodovka" jsou faktorům vazby dosazeny nuly.

Poznámka

Příkaz `EGDEF` zastavuje předběžné zpracování. Definici převodovky pomocí příkazu `EGDEF` je potřeba v nezměněné podobě používat také v případě systémů, u kterých jedna nebo více řídících os ovlivňují vlečnou osu na základě **tabulky křivky**.

Příklad

Programový kód	**Komentář**
`EGDEF(C,B,1,Z,1,Y,1)`	`; Definice svazku os typu "Elektronická převodovka". Řídící osy B, Z, Y ovlivňují vlečnou osu C pomocí požadované hodnoty.`

9.4.2 Zapnutí elektronické převodovky (EGON, EGONSYN, EGONSYNE)

Funkce

Pro zapínání svazku os typu "Elektronická převodovka" existují 3 varianty:

Syntaxe

Varianta 1:

Svazek os typu "Elektronická převodovka" se zapíná selektivně bez synchronizace pomocí příkazu:

```
EGON(FA,"režim přechodu na další
blok",LA1,Z1,N1,LA2,Z2,N2,...,LA5,Z5,N5)
```

Varianta 2:

Svazek os typu "Elektronická převodovka" se zapíná selektivně se synchronizací pomocí příkazu:

```
EGONSYN(FA,"režim přechodu na další
blok",SynPosFA,[,LAi,SynPosLAi,Zi,Ni])
```

Varianta 3:

Svazek os typu "Elektronická převodovka" se zapíná selektivně se synchronizací a s předem zadaným režimem najíždění pomocí příkazu:

```
EGONSYNE(FA,"režim přechodu na další blok",SynPosFA,režim
najíždění[,LAi,SynPosLAi,Zi,Ni])
```

Význam

Varianta 1:

`FA`	Vlečná osa	
`Režim přechodu na další blok`	Mohou se používat následující režimy:	
	`"NOC"`	Na další blok se přechází okamžitě
	`"FINE"`	Přechod na další blok při „jemném chodu synchronizace“
	`"COARSE"`	Přechod na další blok při „hrubém chodu synchronizace“
	`"IPOSTOP"`	Přechod na další blok při dosažení synchronizace s požadovanou hodnotou
`LA1, ... LA5`	Řídící osy	
`Z1, ... Z5`	Čitatel pro faktor vazby i	
`N1, ... N5`	jmenovatel pro faktor vazby i Faktor vazby i = čitatel i/jmenovatel i	

Smí být naprogramovány jedině řídící osy, které byly předtím specifikovány pomocí příkazu `EGDEF`. Musí být naprogramována nejméně jedna řídící osa.

Varianta 2:

`FA`	Vlečná osa	
`Režim přechodu na další blok`	Mohou se používat následující režimy:	
	`"NOC"`	Na další blok se přechází okamžitě
	`"FINE"`	Přechod na další blok při „jemném chodu synchronizace“
	`"COARSE"`	Přechod na další blok při „hrubém chodu synchronizace“
	`"IPOSTOP"`	Přechod na další blok při dosažení synchronizace s požadovanou hodnotou
`[,LAi,SynPosLAi,Zi,Ni]`	(hranaté závorky nezapisovat) Minimálně. 1, max. 5 posloupností s těmito prvky:	
`LA1, ... LA5`	Řídící osy	
`SynPosLAi`	Synchronizační pozice pro i-tou řídící osu	
`Z1, ... Z5`	Čitatel pro faktor vazby i	
`N1, ... N5`	jmenovatel pro faktor vazby i Faktor vazby i = čitatel i/jmenovatel i	

Smí být naprogramovány jedině řídící osy, které byly předtím specifikovány pomocí příkazu `EGDEF`. Prostřednictvím naprogramovaných "Synchronizačních pozic" pro vlečnou osu (`SynPosFA`) a pro řídící osy (`SynPosLA`) jsou definovány polohy, ve kterých je svazek os považován za *synchronizovaný*. Jestliže se elektronická převodovka při svém aktivování nenachází v synchronizovaném stavu, najede vlečná osa na svou definovanou synchronizační pozici.

Varianta 3:

Parametry odpovídají variantě 2, a kromě toho ještě:

`Způsob najíždění`	Mohou se používat následující režimy:	
	`"NTGT"`	V následujícím mezizubí najíždět s časovou optimalizací
	`"NTGP"`	V následujícím mezizubí najíždět s dráhovou optimalizací
	`"ACN"`	Kruhovou osou najíždět v záporném směru otáčení na absolutní pozici
	`"ACP"`	Kruhovou osou najíždět v kladném směru otáčení na absolutní pozici
	`"DCT"`	Na naprogramovanou synchronizační pozici s časovou optimalizací
	`"DCP"`	Na naprogramovanou synchronizační pozici s dráhovou optimalizací

Varianta 3 se uplatňuje pouze u vlečných os typu Modulo, které jsou navázány na řídící osy typu Modulo. Časová optimalizace bere v úvahu mezní hodnoty rychlosti vlečné osy.

Další informace

Popis variant zapnutí

Varianta 1:

Pozice jak řídící osy, tak i vlečné osy v okamžiku aktivování se ukládají jako "synchronizační pozice". "Synchronizační pozice" mohou být načteny pomocí systémových proměnných `$AA_EG_SYN`.

Varianta 2:

Jestliže jsou ve svazku osy Modulo, jsou hodnoty jejich poloh zmenšeny pomocí funkce Modulo. Tímto způsobem je zajištěno, že se najíždí na nejbližší možnou synchronizační polohu (tzv. *relativní synchronizace*: např. následující mezizubí). Jestliže pro vlečnou osu není nastaven signál rozhraní DB(30 + číslo osy), DBX 26 bit 4 "Uvolnění superpozice vlečné osy", nebude se na synchronní pozici najíždět. Místo toho se program bloku s příkazem EGONSYN pozastaví a aktivuje se alarm 16771 s automatickým potvrzováním, dokud není výše uvedený signál nastaven.

Varianta 3:

Vzdálenost mezi zuby vyplývá z následujícího vztahu: 360 * Zi/Ni. V případě, že je vlečná osa v okamžiku vyvolání zastavena, je chování v případě dráhové a časové optimalizace stejné.

Pokud se vlečná osa už pohybuje, pomocí příkazu `NTGP` se uskuteční synchronizace na následující mezizubí nezávisle na momentální rychlosti vlečné osy. Pokud se vlečná osa už pohybuje, pomocí příkazu `NTGT` se uskuteční synchronizace na následující mezizubí v závislosti na momentální rychlosti vlečné osy. Za tím účelem se osa v případě potřeby zabrzdí.

Tabulky křivek

Jestliže se pro jednu z řídících os používá **tabulka křivky**, pak musí být splněny tyto podmínky:

Ni	Jmenovatel faktoru lineární vazby musí být nastaven na 0. (Hodnota jmenovatele 0 by byla pro lineární vazby nepřípustná.) Nulový jmenovatel řídícímu systému signalizuje, že:
Zi	má být interpretováno jako číslo tabulky křivky ,která se má použít. Tabulka křivky se zadaným číslem musí být už v okamžiku aktivování vazby definována.
LAi	Zadání řídící osy odpovídá údaji řídící osy ve vazbě pomocí faktoru vazby (lineární vazba).

Pokud budete potřebovat další informace týkající se používání tabulek křivek a kaskádového řazení elektronických převodovek a jejich synchronizace, viz:
Literatura:
Příručka Popis funkcí, Speciální funkce; Spojení os a ESR (M3), kapitola "Vlečení a vazba řídící hodnotou".

Chování elektronické převodovky při zapnutí (Power-On), resetu, změně provozního režimu, vyhledávání bloku

- Po zapnutí systému není aktivní **žádná** vazba.
- V případě resetu a změny provozního režimu zůstávají aktivní vazby zachovány.
- Při vyhledávání bloku nejsou příkazy pro aktivování, mazání a definice elektronických převodovek prováděny a nejsou ani shromažďovány, nýbrž jsou přeskakovány.

Systémové proměnné elektronických převodovek

Pomocí systémových proměnných elektronických převodovek mohou být ve výrobním programu zjišťovány aktuální stavy svazku os typu "Elektronická převodovka" a v případě nutnosti je možno na ně reagovat.

Systémové proměnné elektronických převodovek jsou označeny následujícím způsobem:

$AA_EG_ ...

nebo

$VA_EG_ ...

Literatura:
Příručka k systémovým proměnným

9.4.3 Vypnutí elektronické převodovky (EGOFS, EGOFC)

Funkce

Pro vypínání aktivního svazku os typu "Elektronická převodovka" existují 3 varianty:

Programování

Varianta 1:

Syntaxe	Význam
`EGOFS(vlečná osa)`	Elektronická převodovka se vypne. Vlečná osa bude zabrzděna až do úplného klidu. Tento příkaz zastavuje předběžné zpracování.

Varianta 2:

Syntaxe	Význam
`EGOFS(vlečná osa,řídící osa1,…,řídící osa5)`	Tyto programové příkazy umožňují **selektivně** odstraňovat vliv jednotlivých řídících os na pohyby vlečné osy.

Musí být uvedena nejméně jedna řídící osa. Vliv uvedených řídících os na vlečnou osu se cíleně vypne. Tento příkaz zastavuje předběžné zpracování. Pokud zůstávají ještě nějaké aktivní řídící osy, pohybuje se vlečná osa podle jejich působení i nadále. Jestliže jsou tímto způsobem zrušeny všechny vlivy řídících os, vlečná osa se zabrzdí do úplného klidu.

Varianta 3:

Syntaxe	Význam
`EGOFC(vlečná osa1)`	Elektronická převodovka se vypne. Vlečné vřeteno se pohybuje dál s otáčkami/rychlostí, které byly v platnosti v okamžiku vypnutí. Tento příkaz zastavuje předběžné zpracování.

Poznámka

Tato varianta je povolena jen pro vřetena.

9.4.4 Vymazání definice elektronické převodovky (EGDEL)

Funkce

Svazek os typu "Elektronická převodovka" musí být vypnutý, jinak jeho definici není možné vymazat.

Programování

Syntaxe	Význam
`EGDEL(vlečná osa)`	Definice vazby svazku os se vymaže. Dokud není dosaženo maximálního počtu současně aktivních svazků os je možné pomocí příkazu `EGDEF` nově definovat další svazky os. Tento příkaz zastavuje předběžné zpracování.

9.4.5 Otáčkový posuv (G95) / elektronická převodovka (FPR)

Funkce

Pomocí příkazu `FPR` může být i vlečná osa elektronické převodovky zadána jako osa, která určuje hodnotu otáčkového posuvu. V tomto případě se systém chová následujícím způsobem:

- Posuv je závislý na požadované hodnotě rychlosti vlečné osy elektronické převodovky.
- Požadovaná hodnota rychlosti se vypočítává z rychlostí řídícího vřetena a řídících os typu Modulo (které nejsou dráhovými osami) a z jim přiřazených faktorů vazby.
- Složky rychlosti z lineárních os, příp. z jiných os než typu Modulo, a superponované pohyby vlečné osy nejsou zohledňovány.

9.5 Synchronní vřeteno

Funkce

V synchronním režimu tvoří řídící vřeteno (LS) a vlečné vřeteno (FS) tzv. **synchronní pár vřeten**. Když je aktivní vazba (synchronní režim), sleduje vlečné vřeteno pohyby řídícího vřetena v souladu s definovanou funkční závislostí.

Dvojice synchronních vřeten mohou být u každého stroje napevno konfigurovány pomocí specifických kanálových strojních parametrů, mohou být ale definovány také uživatelsky pomocí CNC výrobního programu. V jednom kanálu NC systému mohou být současně připraveny až 2 páry synchronních vřeten.

Vazba může být ve výrobním programu:

- definována, příp. upravována
- aktivována
- deaktivována
- vymazána.

Kromě toho je možné v závislosti na stavu programového vybavení provádět tyto operace:

- Čekání na splnění podmínky pro synchronizaci
- Změna chování na přechodech mezi bloky
- Volba druhu vazby, tzn. buď vazba pomocí požadované nebo pomocí řídící hodnoty, nebo je možno zadat úhlové posunutí mezi řídícím a vlečným vřetenem
- Před aktivováním vazby lze převzít předcházející programové příkazy vlečného vřetena
- Lze provést korekci změřené nebo již známé odchylky synchronního režimu

.

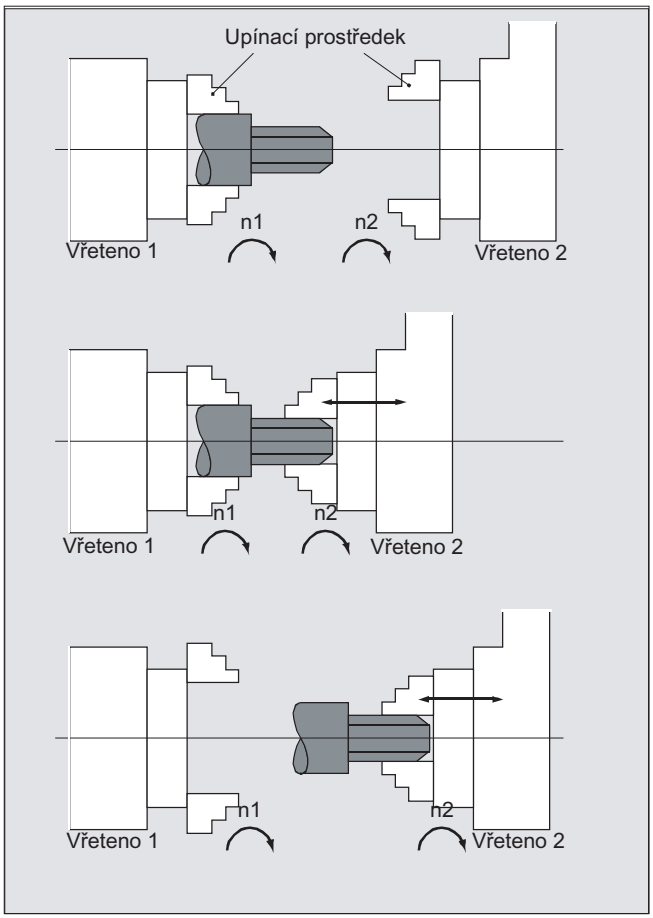
9.5.1 Synchronní vřeteno: Programové příkazy (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC)

Funkce

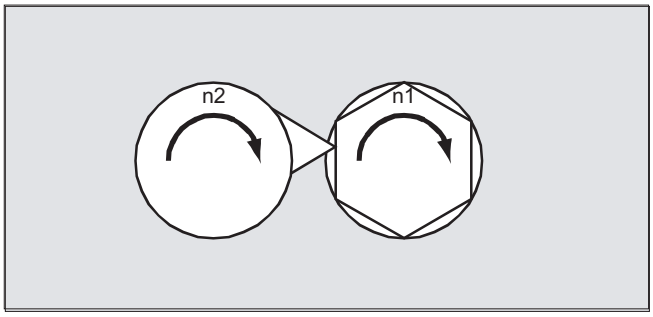
Funkce synchronního vřetena umožňuje synchronní ovládání dvojice vřeten (vlečné vřeteno FS a řídící vřeteno LS), např. za účelem předávání obrobku za pohybu.

Tato funkce nabízí následující režimy:

- Synchronizace otáček ($n_{FS} = n_{LS}$)
- Synchronizace polohy ($\phi_{FS} = \phi_{LS}$)
- Synchronizace polohy s úhlovým offsetem ($\phi_{FS} = \phi_{LS} + \Delta\phi$)



Pomocí zadání převodového poměru mezi řídícím a vlečným vřetenem, který se nerovná 1, je možné dosáhnout také obrábění mnohohranu (soustružení mnohohranu).



Syntaxe

```
COUPDEF(<FS>,<LS>,<ÜFS>,<ÜLS>,<přechod na další blok>,<druh vazby>)
COUPON(<FS>,<LS>,<POSFS>)
COUPONC(<FS>,<LS>)
COUPOF(<FS>,<LS>,<POSFS>,<POSLS>)
COUPOFS(<FS>,<LS>)
COUPOFS(<FS>,<LS>,<POSFS>)
COUPRES(<FS>,<LS>)
COUPDEL(<FS>,<LS>)
WAITC(<FS>,<přechod na další blok>,<LS>,<přechod na další blok>)
```

Poznámka

Zkrácený způsob zápisu

V případě příkazů `COUPOF`, `COUPOFS`, `COUPRES` a `COUPDEL` je možný také zkrácený způsob zápisu bez udání řídícího vřetena.

Význam

`COUPDEF`: Definice/editace specifické uživatelské vazby

`COUPON`: Aktivování vazby. Vlečné vřeteno se synchronizuje podle řídícího vřetena, přičemž se vychází z momentálních otáček.

`COUPONC`: Vazba při svém aktivování převezme předcházející naprogramované příkazy `M3 S...` nebo `M4 S...`.

Rozdíl otáček vlečného vřetena se okamžitě převezme.

`COUPOF`: Deaktivování vazby.

- s okamžitým přechodem na další blok:
  `COUPOF(<S2>,<S1>)`
- Přechod na další blok teprve po přejetí vypínací pozice <`POSFS`>, příp. <`POSLS`>:
  `COUPOF(<S2>,<S1>,<POSFS>)`
  `COUPOF(<S2>,<S1>,<POSFS>,<POSLS>)`

COUPOFS:	Vypnutí vazby se zastavením vlečného vřetena. Co možno nejrychlejší přechod na další blok. COUPOFS(<S2>,<S1>) Přechod na další blok teprve po přejetí vypínací pozice: COUPOFS(<S2>,<S1>,<POSFS>)
COUPRES:	Reset parametrů vazby na hodnoty nastavené v konfiguraci pomocí MD a SD
COUPDEL:	Vymazání uživatelem definované vazby
WAITC:	Čekání na splnění synchronizační podmínky (NOC se při přechodu na další blok uloží do IPO)
<FS>:	Popis vlečného vřetena

Volitelné parametry:

<LS>:	Označení řídícího vřetena Zadání čísla vřetena: např. S2, S1
<ÜFS>, <ÜLS>:	Převodový poměr mezi FS a LS. <ÜFS> = čitatel, <ÜLS> = jmenovatel Předdefinované nastavení: <ÜFS> / <ÜLS> = 1,0 ; uvedení jmenovatele je nepovinné
<přechod na další blok>:	Chování při přechodu na další blok Přechod na další blok se uskuteční: "NOC" okamžitě "FINE" při dosažení "jemného chodu synchronizace" "COARSE" při dosažení "hrubého chodu synchronizace" "IPOSTOP" při dosažení zastavení interpolátoru (IPOSTOP), tzn. po dosažení synchronizace s požadovanou hodnotou (předdefinované nastavení) Chování při přechodu na další blok má modální působnost.
<druh vazby>:	Druh vazby: Vazba mezi FS a LS. "DV" Vazba pomocí požadované hodnoty (předdefinované nastavení) "AV" Vazba pomocí skutečné hodnoty "VV" Vazba pomocí rychlosti Druh vazby má modální působnost.
<POSFS>:	Úhlové posunutí mezi řídícím a vlečným vřetenem Rozsah hodnot: 0°… 359,999°
<POSFS>,<POSLS>:	Vypínací pozice vlečného a řídícího vřetena "Přechod na další blok se odblokuje až po přejetí pozic POS_{FS}, POS_{LS}." Rozsah hodnot: 0°… 359,999°

Příklady

Příklad 1: Práce s řídícím a vlečným vřetenem

Programování	Komentář
	; Řídící vřeteno = Hlavní vřeteno = Vřeteno 1
	; Vlečné vřeteno = Vřeteno 2
N05 M3 S3000 M2=4 S2=500	; Řídící vřeteno se otáčí s otáčkami 3000 ot/min, vlečné vřeteno se otáčí s otáčkami 500 ot/min.
N10 COUPDEF(S2,S1,1,1,"NOC","Dv")	; Definice vazby (může být nastavena také v konfiguraci).
...	
N70 SPCON	; Řídící vřeteno v režimu polohové regulace (vazba pomocí požadované hodnoty).
N75 SPCON(2)	; Vlečné vřeteno v režimu polohové regulace.
N80 COUPON(S2,S1,45)	; Aktivování vazby za pohybu s offsetovou pozicí = 45 stupňů.
...	
N200 FA[S2]=100	; Rychlost polohování = 100 stupňů/min
N205 SPOS[2]=IC(-90)	; Najetí na superponovanou pozici 90 stupňů v záporném směru.
N210 WAITC(S2,"Fine")	; Čekání na "jemné chod synchronizace".
N212 G1 X... Y... F...	; Opracování
...	
N215 SPOS[2]=IC(180)	; Najetí na superponovanou pozici 180 stupňů v kladném směru.
N220 G4 S50	; Doba prodlevy = 50 otáček hlavního vřetena
N225 FA[S2]=0	; Aktivování rychlosti nastavené v konfiguraci.
N230 SPOS[2]=IC(-7200)	; 20 otáček. Pohyb s rychlosti nastavenou v konfiguraci v záporném směru.
...	
N350 COUPOF(S2,S1)	; Zrušení vazby za pohybu, S=S2=3000
N355 SPOSA[2]=0	; Zastavení FS v poloze nula stupňů.
N360 G0 X0 Y0	
N365 WAITS(2)	; Čekání na vřeteno 2.
N370 M5	; Zastavení FS.
N375 M30	

Příklad 2: Programování rozdílných otáček

Programování	Komentář
	; Řídící vřeteno = Hlavní vřeteno = Vřeteno 1
	; Vlečné vřeteno = Vřeteno 2
N01 M3 S500	; Řídící vřeteno se otáčí s otáčkami 500 ot/min.
N02 M2=3 S2=300	; vlečné vřeteno se otáčí s otáčkami 300 ot/min.
...	
N10 G4 F1	; Doba prodlevy pro hlavní vřeteno.
N15 COUPDEF (S2,S1,-1)	; Faktor vazby s převodovým poměrem -1:1
N20 COUPON (S2,S1)	; Aktivování vazby. Otáčky vlečného vřetena vyplývají z otáček řídícího vřetena a z faktoru vazby.
...	
N26 M2=3 S2=100	; Programování rozdílných otáček.

Příklad 3: Příklady převzetí pohybu pro rozdílné otáčky

1. Aktivování vazby pomocí příkazu COUPON, když byly předtím pro vlečnou osu naprogramovány nějaké příkazy

Programování	Komentář
	; Řídící vřeteno = Hlavní vřeteno = Vřeteno 1
	; Vlečné vřeteno = Vřeteno 2
N05 M3 S100 M2=3 S2=200	; Řídící vřeteno se otáčí s otáčkami 100 ot/min, vlečné vřeteno s otáčkami 200 ot/min.
N10 G4 F5	; Doba prodlevy = 5 sekund hlavního vřetena
N15 COUPDEF(S2,S1,1)	; Převodový poměr FS ku LS je 1,0 (předdefinované nastavení)
N20 COUPON (S2,S1)	; Aktivování vazby na řídící vřeteno za pohybu.
N10 G4 F5	; vlečné vřeteno se otáčí s otáčkami 100 ot/min.

2. Aktivování vazby pomocí příkazu COUPONC, když byly předtím pro vlečnou osu naprogramovány nějaké příkazy

Programování	Komentář
	; Řídící vřeteno = Hlavní vřeteno = Vřeteno 1
	; Vlečné vřeteno = Vřeteno 2
N05 M3 S100 M2=3 S2=200	; Řídící vřeteno se otáčí s otáčkami 100 ot/min, vlečné vřeteno s otáčkami 200 ot/min.
N10 G4 F5	; Doba prodlevy = 5 sekund hlavního vřetena
N15 COUPDEF(S2,S1,1)	; Převodový poměr FS ku LS je 1,0 (předdefinované nastavení)
N20 COUPONC(S2,S1)	; Aktivování vazby na řídící vřeteno za pohybu a převzetí předcházejících otáček pro S2.
N10 G4 F5	; S2 se otáčí s rychlostí 100 ot/min + 200 ot/min = 300 ot/min

3. Aktivování vazby pomocí příkazu COUPON, když vlečné vřeteno stojí

Programování	Komentář
	; Řídící vřeteno = Hlavní vřeteno = Vřeteno 1
	; Vlečné vřeteno = Vřeteno 2
N05 SPOS=10 SPOS[2]=20	; Vlečné vřeteno S2 v polohovacím režimu.
N15 COUPDEF(S2,S1,1)	; Převodový poměr FS ku LS je 1,0 (předdefinované nastavení)
N20 COUPON (S2,S1)	; Aktivování vazby na řídící vřeteno za pohybu.
N10 G4 F1	; Vazba je vytvořena, S2 zůstává stát v poloze 20 stupňů.

4. Aktivování vazby pomocí příkazu COUPONC, když vlečné vřeteno stojí

Poznámka

Polohovací nebo osový režim

Jestliže se vlečné vřeteno před zapojením do vazby nachází v polohovacím nebo v osovém režimu, potom se toto vlečné vřeteno chová v případě příkazů COUPON(<FS>,<LS>) a COUPONC(<FS>,<LS>) stejně.

UPOZORNĚNÍ
Řídící vřeteno a osový režim
Jestliže se řídící vřeteno nachází před definicí vazby v režimu osy, po aktivování vazby se bude uplatňovat mezní hodnota rychlosti ze strojního parametru:
MD32000 $MA_MAX_AX_VELO (maximální rychlost osy)
Aby se tomuto chování zabránilo, musí být osa před definicí vazby přepnuta do režimu vřetena (`M3 S...` nebo `M4 S...`).

Další informace

Definice dvojice synchronních vřeten

Vazba nastavená v konfiguraci:

V případě vazby nastavené v konfiguraci jsou řídící a vlečné vřeteno definovány pomocí strojního parametru. Vřetena nastavená v konfiguraci mohou být ve výrobním programu změněna. Parametry vazby je možno ve výrobním programu nastavovat pomocí příkazu `COUPDEF` (předpoklad: ochrana proti zápisu není aktivována).

Uživatelem definovaná vazba:

Pomocí příkazu `COUPDEF` může být vazba ve výrobním programu nově definována nebo změněna. Jestliže už je nějaká vazba aktivní, musí být před definicí nové vazby napřed vymazána pomocí příkazu `COUPDEL`.

Definice vazby: COUPDEF

Vazba je úplně definována pomocí následujícího příkazu:

```
COUPDEF(<FS>,<LS>,<ÜFS>,<ÜLS>, chování na přechodu mezi bloky, druh vazby)
```

Vlečné vřeteno (FS) a řídící vřeteno (LS)

Pomocí názvů os pro vlečné vřeteno (FS) a řídící vřeteno (LS) je vazba jednoznačně definována. Názvy os musí být naprogramovány u každého příkazu `COUPDEF`. Ostatní parametry vazby mají modální platnost a musí být naprogramovány jedině tehdy, pokud jsou měněny.

Příklad:

```
COUPDEF(S2,S1)
```

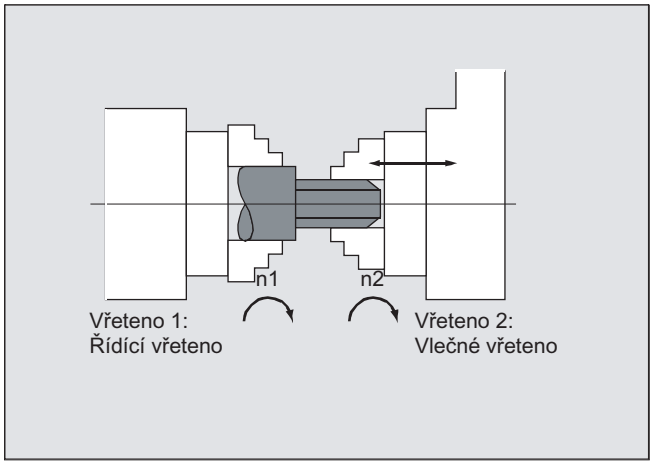
Převodový poměr ÜFS / ÜLS

Převodový poměr se udává jako poměr otáček mezi vlečným vřetenem (čitatel) a řídícím vřetenem (jmenovatel). Čitatel musí být naprogramován. Pokud není naprogramován žádný jmenovatel, dosadí se jako hodnota jmenovatele 1,0.

Příklad:

Vlečné vřeteno S2, řídící vřeteno S1, převodový poměr = 1 / 4 = 0,25.

```
COUPDEF(S2,S1,1.0, 4.0)
```



Poznámka

Převodový poměr může být upravován i v době, kdy je vazba aktivní a vřetena se otáčejí.

Chování na přechodech mezi bloky NOC, FINE, COARSE, IPOSTOP

Při programování chování na přechodech mezi bloky jsou možné následující zkrácené způsoby zápisu:

- `"NO"`: okamžitě (předdefinované nastavení)
- `"FI"`: při dosažení "jemného chodu synchronizace"
- `"CO"`: při dosažení "hrubého chodu synchronizace"
- `"IP"`: při dosažení zastavení interpolátoru (IPOSTOP), tzn. po dosažení synchronizace s požadovanou hodnotou

Druh vazby DV, AV

⚠ POZOR

Druh vazby se smí měnit jedině tehdy, když je vazba vypnutá.

Aktivování synchronizačního režimu, COUPON, POSFS

- Aktivování vazby s libovolným úhlovým vztahem mezi osami LS a FS::
  - `COUPON(S2,S1)`
  - `COUPON(S2,S1,<POSFS>)`
  - `COUPON(S2)`
- Aktivování vazby s úhlovým offsetem <`POSFS>`

  Pro účely vazby s polohovou synchronizací pro profilové obrobky.
  `<POSFS>`:se vztahuje na polohu 0° řídícího vřetena v kladném směru otáčení
  Rozsah hodnot `<POSFS>`: 0°… 359,999°

  - `COUPON(S2,S1,30)`

  Tímto způsobem můžete měnit úhlový offset, i když je vazba už aktivována.

Polohování vlečného vřetena

Když je aktivována vazba synchronního vřetena, je možné také vlečné vřeteno nastavovat do libovolné polohy v rozsahu ±180°, a to nezávisle na pohybu spouštěném řídícím vřetenem.

Nastavování polohy pomocí příkazu SPOS

Vlečné vřeteno může být interpolováno pomocí příkazu `SPOS=...`

Příklad:

```
SPOS[2]=IC(-90)
```

Pokud budete potřebovat podrobnější informace týkající se příkazu `SPOS`, viz:
Literatura:
Příručka programování, Základy

Rozdíl otáček M3 S... nebo M4 S...

Rozdíl otáček vzniká v důsledku superpozice dvou zdrojů otáček, u kterých záleží na znaménku, a pro vlečné vřeteno se jako nová hodnota programuje např. pomocí příkazů `S<n>=...` nebo `M<n>=3`, `M<n>=4` v režimu regulace otáček i v době, kdy je vazba synchronních vřeten aktivní. Přitom je tato složka hodnoty otáček odvozena prostřednictvím faktoru vazby od řídící osy a v souladu se svým znaménkem se přičítá k hodnotě pro vlečné vřeteno.

Poznámka

Pomocí směru otáček `M3` nebo `M4` musí být také otáčky `S...` nově naprogramovány, protože jinak by se v důsledku nesprávného programového příkazu aktivoval alarm.

Pokud budete potřebovat další informace týkající se rozdílu otáček, viz:
Literatura:
Příručka Popis funkcí, Rozšiřovací funkce; Synchronní vřetena (S3)

Rozdíl otáček v případě příkazu COUPONC

Převzetí pohybu pro rozdílné otáčky

Prostřednictvím aktivování vazby synchronních vřeten pomocí příkazu `COUPONC` jsou superponovány momentálně platné otáčky vlečného vřetena (`M3 S...` nebo `M4 S...`).

Poznámka

Odblokování superpozice

Sčítání otáček vřetena (`M3 S...` nebo `M4 S...`) prostřednictvím vazby synchronních vřeten `COUPONC` je v platnosti jen tehdy, pokud je superpozice tohoto druhu povolena.

Omezení dynamiky řídícího vřetena

Dynamika řídícího vřetena musí být omezena natolik, aby při superpozici vlečného vřetena nemohlo dojít k překročení mezních hodnot jeho dynamiky.

Rychlost, zrychlení: FA, ACC, OVRA, VELOLIMA

Rychlost a zrychlení osy vlečného vřetena mohou být naprogramovány pomocí těchto příkazů:

- `FA[SPI(S<n>)]`, příp. `FA[S<n>]` (axiální rychlost)
- `ACC[SPI(S<n>)]`, příp. `ACC[S<n>]` (axiální zrychlení)
- `OVRA[SPI(S<n>)]`, příp. `OVRA[S<n>]` (axiální korekce (override))
- `VELOLIMA[SPI(S<n>)]`, příp. `VELOLIMA[S<n>]` (axiální zvýšení, příp. snížení rychlosti)

kde `<n>` = 1, 2, 3, ... (číslo vlečného vřetena)

Literatura:
Příručka programování, Základy

Poznámka

Složka zrychlení JERKLIMA[S<n>]

Programování dynamiky axiálního zvyšování nebo snižování rychlosti není v současnosti u vřeten možné.

Pokud budete potřebovat podrobnější informace týkající se konfigurace dynamiky os, viz:
Literatura:
Příručka Popis funkcí, Rozšiřovací funkce; Kruhové osy (R2)

Programovatelné chování při přechodech mezi bloky, WAITC

Pomocí příkazu `WAITC` může být zadáno chování na přechodech mezi bloky, např. po změně parametrů vazby nebo při polohovacích operacích, s různými podmínkami chování synchronizace (hrubá, jemná, `IPOSTOP`). Jestliže není žádná podmínka chodu synchronizace uvedena, platí chování při přechodech mezi bloky zadané při definici v příkazu `COUPDEF`.

Příklad:

Čekání na dosažení podmínky chodu synchronizace v souladu s příkazem `COUPDEF`

```
WAITC( )
```

Čekání na dosažení podmínky chodu synchronizace `FINE` u vlečného vřetena S2 a `COARSE` u vlečného vřetena S4:

```
WAITC(S2,"FINE",S4,"COARSE")
```

Deaktivování vazby COUPOF

Pomocí příkazu `CUOPOF` je možné definovat chování při deaktivování vazby:

- Vypnutí vazby s okamžitým přechodem na další blok:
  - `COUPOF(S2,S1)` (s uvedením řídícího vřetena)
  - `COUPOF(S2)` (bez uvedení řídícího vřetena)
- Deaktivování vazby po přejetí vypínací pozice. Přechod na následující blok se provede po přejetí vypínací pozice.
  - `COUPOF(S2,S1,150)` (vypínací pozice FS: 150°)
  - `COUPOF(S2,S1,150,30)` (vypínací pozice FS: 150°, LS: 30°)

Deaktivování vazby se zastavením vlečného vřetena, COUPOFS

Pomocí příkazu `CUOPOFS` je možné definovat chování při deaktivování vazby se zastavením vlečného vřetena:

- Vypnutí vazby se zastavením vlečného vřetena a s okamžitým přechodem na další blok:
  - `COUPOFS(S2,S1)` (s uvedením řídícího vřetena)
  - `COUPOFS(S2)` (bez uvedení řídícího vřetena)
- Deaktivování vazby po přejetí vypínací pozice se zastavením vlečného vřetena. Přechod na následující blok se provede po přejetí vypínací pozice.
  - `COUPOFS(S2,S1,150)` (vypínací pozice FS: 150°)

Vymazání vazby, COUPDEL

Pomocí příkazu `COUPDEL` se vazba vymaže:

- `COUPDEL(S2,S1)` (s uvedením řídícího vřetena)
- `COUPDEL(S2)` (bez uvedení řídícího vřetena)

Obnovení původního nastavení parametrů vazby, COUPRES

Pomocí příkazu `COUPRES` je možno aktivovat hodnoty parametrů vazby, které jsou uloženy ve strojních a nastavovaných parametrech:

- `COUPRES(S2,S1)` (s uvedením řídícího vřetena)
- `COUPRES(S2)` (bez uvedení řídícího vřetena)

Systémové proměnné

Aktuální stavové informace o vazbě vlečného vřetena

Aktuální stavové informace o vazbě vlečného vřetena je možné načítat pomocí následující systémové proměnné:

`$AA_COUP_ACT[<FS>]`

Hodnota	Význam
0	žádná vazba není aktivní
4	synchronní vazba mezi vřeteny je aktivní
Upozornění Ostatní hodnoty systémových proměnných se vztahují na osový režim. **Literatura:** Příručka Seznam systémových proměnných	

Aktuální úhlový offset

Aktuální úhlový offset vlečného vřetena vzhledem k řídícímu vřetenu je možné načítat pomocí následující systémové proměnné:

- `$AA_COUP_OFFS[<FS>]` (úhlový offset vztahující se na požadovanou hodnotu)
- `$VA_COUP_OFFS[<FS>]` (úhlový offset vztahující se na skutečnou hodnotu)

Poznámka

Po odstranění odblokování regulátoru, když jsou vazba a režim vlečení aktivovány, se po opětovném aktivování odblokování regulátoru nastaví jiný offset polohy, než byla původně naprogramovaná hodnota. V tomto případě mohou být změněné hodnoty offsetu polohy načteny a v případě potřeby ve výrobním programu korigovány.

9.6 Spojení master/slave (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS)

Funkce

Vazba typu Master/Slave před verzí SW 6.4 dovolovala zapojení os typu Slave do vazby na jejich osu Master pouze tehdy, když byly všechny podílející se osy v klidu.

Rozšíření od verze SW 6.5 umožňuje zapojování a rozpojování vazeb **točících se** vřeten v režimu regulace otáček a dynamickou konfiguraci.

Syntaxe

`MASLON(Slv1,Slv2,..., )`	
`MASLOF(Slv1,Slv2,..., )`	
`MASLDEF(Slv1,Slv2,..., osa Master)`	Rozšíření pro dynamickou konfiguraci
`MASLDEL(Slv1,Slv2,..., )`	Rozšíření pro dynamickou konfiguraci
`MASLOFS(Slv1, Slv2, ..., )`	Rozšíření pro vřeteno Slave

Poznámka

V případě příkazů `MASLOF/MASLOFS` odpadá implicitní zastavení předběžného zpracování. Protože se neuskutečňuje zastavení předběžného zpracování, neposkytují systémové proměnné $P... pro osy typu Slave až do okamžiku opětovného naprogramování žádné aktualizované hodnoty.

Význam

Všeobecně

`MASLON`	Aktivování dočasné vazby.
`MASLOF`	Rozpojení aktivní vazby. V případě vřeten je nutno mít na zřeteli rozšíření.

Rozšíření pro dynamickou konfiguraci

`MASLDEF`	Vytvoření/editace vazby, která je definována uživatelem pomocí strojních parametrů nebo z výrobního programu.
`MASLOFS`	Rozpojení vazby analogicky s příkazem `MASLOF` a automatické zabrždění vřetena Slave.
`MASLDEL`	Zrušení spojení os typu Master/Slave a vymazání definice tohoto spojení.
`Slv1, Slv2, ...`	Osy typu Slave, které jsou řízeny osami typu Master.
`Osa typu Master`	Osa, které ve spojení typu Master/Slave řídí osy definované jako Slave.

Příklady

Příklad 1: Dynamická konfigurace spojení os typu Master/Slave

Dynamická konfigurace spojení os typu Master/Slave z výrobního programu:

Osa, která se stane relevantní po otočení zásobníku os, se má stát osou Master.

Programový kód	Komentář
MASLDEF(AUX,S3)	; S3 je osou Master pro osu AUX
MASLON(AUX)	; Vazba pro osu AUX
M3=3 S3=4000	; Směr otáčení vpravo
MASLDEL(AUX)	; Vymazání konfigurace a rozpojení vazby
AXCTSWE(CT1)	; Otočení zásobníku

Příklady

Příklad 2: Vazba osy typu Slave pomocí skutečné hodnoty

Vazba osy typu Slave pomocí skutečné hodnoty na stejnou hodnotu osy Master pomocí příkazu PRESETON.

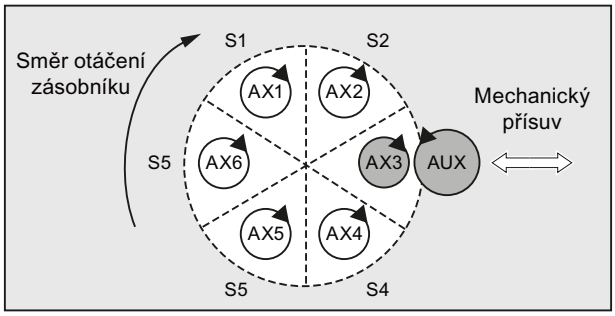
V případě trvalé vazby typu Master/Slave má být skutečná hodnota na ose SLAVE změněna pomocí příkazu PRESETON.

Programový kód	Komentář
N37262 $MA_MS_COUPLING_ALWAYS_ACTIVE[AX2]=0	; Krátké vypnutí trvalé vazby.
N37263 NEWCONF	
N37264 STOPRE	
MASLOF(Y1)	; Dočasné vypnutí vazby.
N5 PRESETON(Y1,0,Z1,0,B1,0,C1,0,U1,0)	; Dosazení skutečné hodnoty osám typu Slave, které nenajely na referenční bod, protože tyto osy jsou aktivovány se zapnutím systému.
N37262 $MA_MS_COUPLING_ALWAYS_ACTIVE[AX2]=1	; Aktivování trvalé vazby.
N37263 NEWCONF	

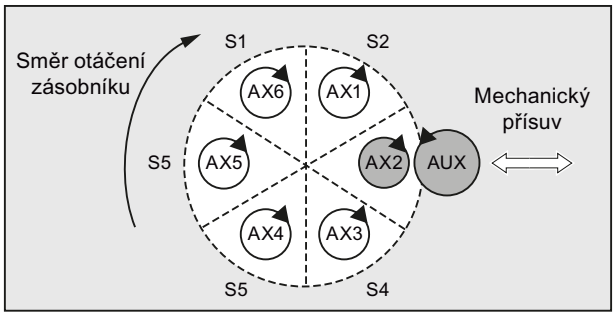
Příklad 3: Sekvence vazeb Poloha 3/zásobník CT1

Aby po otočení zásobníku mohla být vazba připojena na jiné vřeteno, musí být napřed stará vazba rozpojena, její konfigurace vymazána a zadána nová konfigurace vazby.

Výchozí situace:



Nové otočení o jeden slot:



Literatura:

Příručka Popis funkcí, Rozšiřovací funkce, Další ovládací panely a NCU (B3), kapitola: "Osový zásobník"

Další informace

Všeobecně

MASLOF	U vřeten v režimu regulace otáček se tento příkaz okamžitě uskuteční. Vřetena typu Slave, které se v tomto okamžiku otáčejí, si své otáčky zachovají, dokud nebudou naprogramovány nové.

Rozšíření pro dynamickou konfiguraci

MASLDEF	Definice spojení os typu Master/Slave z výrobního programu. Dříve byla možná pouze definice pomocí strojních parametrů.
MASLDEL	Příkaz odstraní přiřazení os typu Slave ose typu Master a současně vazbu rozpojí, analogicky s příkazem MASLOF. Definice spojení Master/Slave konfigurovaná ve strojních parametrech zůstává zachována.
MASLOFS	Příkaz MASLOFS se může používat v situacích, kdy potřebujete vřeteno Slave po rozpojení vazby automaticky zabrzdit. U os a vřeten v režimu polohování se vazba aktivuje a rozpojuje, jen když jsou osy v klidu.

Poznámka

Pro osu Slave může být pomocí příkazu PRESETON skutečná hodnota synchronizována na stejnou hodnotu, jakou má osa Master. Za tím účelem musí být trvalá vazba Master/Slave na krátkou chvíli deaktivována, aby bylo možné ose Slave, která po zapnutí systému (Power-on) nenajela na referenční bod, dosadit hodnotu osy Master. Potom se trvalá vazba znovu obnoví.

Trvalá vazba typu Master/Slave se aktivuje pomocí nastavení strojního parametru MD37262 $MA_MS_COUPLING_ALWAYS_ACTIVE = 1 a nemá žádný vliv na příkazy NC jazyka pro dočasnou osu.

Chování vazby v případě vřeten

U vřeten v režimu regulace otáček je chování vazby v případě příkazů MASLON, MASLOF, MASLOFS a MASLDEL explicitně definováno pomocí strojního parametru MD37263 $MA_MS_SPIND_COUPLING_MODE.

Při standardním nastavení MD37263 = 0 se uskutečňuje aktivování a rozpojování vazby s osami typu Slave jen tehdy, jsou-li všechny podílející se osy v klidu. Příkaz MASLOFS odpovídá příkazu MASLOF.

Jestliže je nastaveno MD37263 = 1 příkazy pro ovládání vazby se uskuteční okamžitě, tedy během pohybu. V případě příkazu MASLON se vazba okamžitě zapojí a v případě příkazů MASLOFS nebo MASLOF se ihned rozpojí. Vřetena typu Slave, které se v tomto okamžiku otáčejí, se v případě příkazu MASLOFS automaticky zabrzdí a v případě příkazu MASLOF si své otáčky zachovají, dokud nebudou naprogramovány nové.

Pohybové synchronní akce 10

10.1 Základy

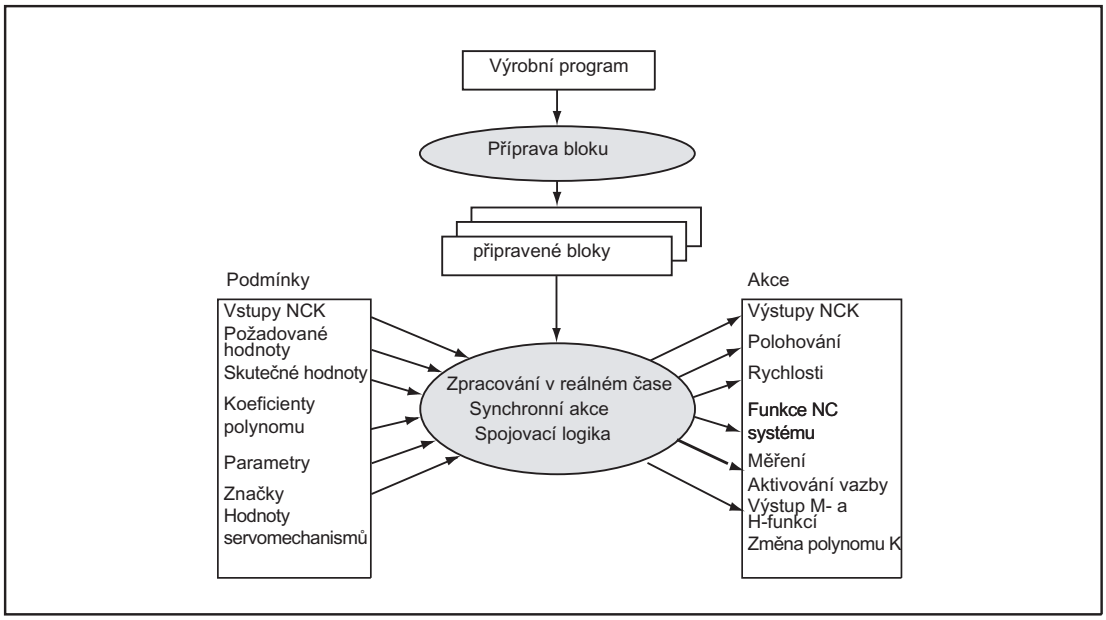
Funkce

Synchronní akce nabízejí možnost uskutečňovat určité akce synchronně se zpracováváním bloků.

Okamžik uskutečnění akcí může být definován prostřednictvím podmínek. Podmínky jsou monitorovány v taktu interpolace. Akce takto představují reakci na události v reálném čase, jejichž uskutečnění není vázáno na hranice bloku.

Kromě toho obsahuje synchronní akce údaje týkající se doby jejího trvání a frekvence zjišťování hodnot pro naprogramované proměnné v hlavním zpracování a v důsledku toho také frekvence uskutečňování akcí, které se mají spouštět. Tímto způsobem může být akce spuštěna jen jednou nebo cyklicky (pokaždé v interpolačním taktu).

Možné případy použití

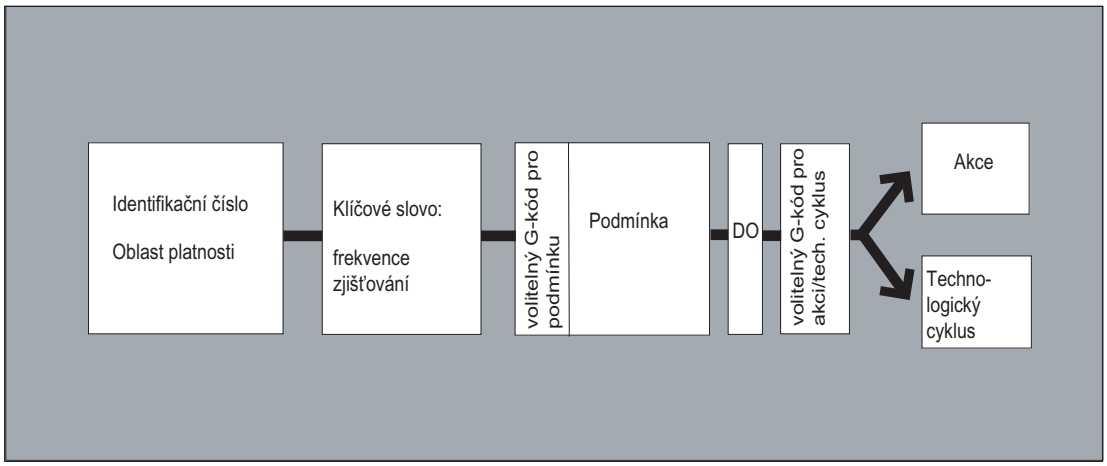


- Optimalizace aplikací kritických z hlediska doby zpracování (např. výměna nástroje)
- Rychlá reakce na externí události
- Programování AC-regulace
- Instalace bezpečnostních funkcí
- ....

Programování

Synchronní akce vyžaduje samostatný blok a uplatňuje se od následujícího zpracovatelného bloku s funkcemi stroje (např. pohyby s příkazy `G0`, `G1`, `G2`, `G3`).

Synchronní akce se skládají z až 5 příkazových prvků s odlišnými úkoly:



Syntaxe:

```
DO <akce1> <akce2> …
<SCHLÜSSELWORT> <podmínka> DO <akce1> <akce2> …
ID=<n> <SCHLÜSSELWORT> <podmínka> DO <akce1> <akce2> …
IDS=<n> <SCHLÜSSELWORT> <podmínka> DO <akce1> <akce2> …
```

Význam:

`DO`	Příkaz pro spuštění naprogramované akce nebo akcí Uskuteční se, jen když je `<podmínka>` splněna (pokud je naprogramována). → viz " akce "

`<akce1>`
`<akce2>`
…
	Akce, které se mají spustit
	Příklady:
	• Přiřazení proměnné
	• Spuštění technologického cyklu

`<SCHLÜSSELWORT>`	Pomocí klíčového slova (`WHEN`, `WHENEVER`, `FROM` nebo `EVERY`) je definována cyklická kontrola `<podmínky>` synchronní akce. → viz " Cyklická kontrola podmínky "
`<podmínka>`	Spojovací logický výraz pro proměnné hlavního zpracování Podmínka je kontrolována v taktu IPO.

`ID=<n>`
, příp.
`IDS=<n>`
	Identifikační číslo
	Pomocí identifikačního čísla je definována oblast platnosti a poloha v rámci posloupnosti při zpracovávání.
	→ viz " Oblast platnosti a posloupnost při zpracovávání "

Koordinace synchronních akcí/technologických cyklů

Pro koordinaci synchronních akcí/technologických cyklů jsou k dispozici následující příkazy:

Příkaz	Význam
`CANCEL(<n>)`	Mazání synchronních akcí → viz " Mazání synchronních akcí "
`LOCK(<n>)`	Blokování synchronních akcí
`UNLOCK(<n>)`	Odblokování synchronních akcí
`RESET`	Reset technologického cyklu
	Pokud budete potřebovat informace o příkazech `LOCK`, `UNLOCK` a `RESET`: → viz " Blokování, odblokování, reset "

Příklad

Programový kód	**Komentář**
`WHEN $AA_IW[Q1]>5 DO M172 H510`	`; Jestliže je překročena skutečná hodnota osy Q 1,5 mm, na rozhraní PLC se aktivují pomocné funkce M172 a H510.`

10.1.1 Oblast platnosti a posloupnost při zpracovávání (ID, IDS)

Funkce

Oblast platnosti

Oblast platnosti synchronní akce je definována prostřednictvím identifikace `ID`, příp. `IDS`:

Žádná modální identifikace:	Synchronní akce s blokovou platností v automatickém režimu
`ID`:	Synchronní akce s modální platností v automatickém režimu až do konce programu
`IDS`:	Statické synchronní akce s modální platností v každém provozním režimu, i po skončení programu

Použití

- AC-broušení v režimu JOG.
- Spojovací logika pro Safety Integrated
- Monitorovací funkce, reakce na stavové informace stroje ve všech provozních režimech

Posloupnost při zpracování

Synchronní akce s modální a statickou platností jsou zpracovávány v interpolačním taktu v posloupnosti svých čísel `ID`, příp. `IDS` (`ID=<n>`, příp. `IDS=<n>`).

Synchronní akce s blokovou platností (bez identifikačního čísla) jsou zpracovány po zpracování synchronních akcí s modální platností v posloupnosti, v jaké byly naprogramovány.

Poznámka

Prostřednictvím nastavení strojních parametrů mohou být synchronní akce s modální platností chráněny proti změnám nebo vymazání (→ výrobce stroje!).

Programování

Syntaxe	Význam
žádná modální identifikace	Synchronní akce je v platnosti pouze v **automatickém režimu**. Platí jen pro následující zpracovatelný blok (blok s pohybovým příkazem nebo s jinou funkcí stroje), má tedy **blokovou** platnost. Příklad: `WHEN $A_IN[3]==TRUE DO $A_OUTA[4]=10`
`ID=<n> ...`	Synchronní akce má v následujících blocích **modální** platnost a může být vypnuta příkazem `CANCEL(<n>)` nebo může být přepsána naprogramováním nové synchronní akce se stejným identifikačním číslem. Synchronní akce, které jsou aktivní v bloku s příkazem `M30`, způsobují zpoždění ukončení programu. Identifikační číslo (ID) synchronní akce se uplatňuje pouze v **automatickém režimu**. Rozsah hodnot pro **<n>**: 1 ... 255 Příklad: `ID=2 EVERY $A_IN[1]==1 DO POS[X]=0`
`IDS=<n>`	Statické synchronní akce se uplatňují **modálně ve všech provozních režimech**. Zůstávají aktivní i po skončení programu a mohou být aktivovány přímo po zapnutí systému (Power-On) pomocí funkce ASUP. Tímto způsobem je možné aktivovat akce, které mají být v NC systému zpracovány nezávisle na zvoleném provozním režimu. Rozsah hodnot pro **<n>**: 1 ... 255 Příklad: `IDS=1 EVERY $A_IN[1]==1 DO POS[X]=100`

10.1.2 Cyklická kontrola podmínky (WHEN, WHENEVER, FROM, EVERY)

Funkce

Prostřednictvím klíčového slova je definována cyklická kontrola podmínky pro spuštění synchronní akce. Jestliže není žádné klíčové slovo naprogramováno, uskutečňují se příkazy synchronní akce v každém taktu IPO.

Klíčová slova

žádné klíčové slovo	Provádění akce není vázáno na žádnou podmínku. Akce se uskutečňuje cyklicky v každém interpolačním taktu.
`WHEN`	Podmínka je v každém interpolačním taktu vyhodnocována tak dlouho, dokud není jedenkrát splněna; odpovídající akce se potom právě jedenkrát uskuteční.
`WHENEVER`	Podmínka je cyklicky vyhodnocována v každém interpolačním taktu. Pokud je podmínka splněna, odpovídající akce se uskuteční v každém interpolačním taktu.
`FROM`	Podmínka se v každém interpolačním taktu kontroluje, dokud není jedenkrát splněna. Akce se potom provádí tak dlouho, dokud je synchronní akce aktivní, tzn. i tehdy, když už podmínka splněna není.
`EVERY`	Podmínka je vyhodnocována v každém interpolačním taktu. Pokud je podmínka splněna, potom se daná akce vždy jedenkrát provede. Řízení podle náběžné nebo sestupné hrany signálu: Akce se spouští vždy, když podmínka mění svůj výsledek ze stavu FALSE do stavu TRUE.

Proměnné hlavního zpracování

Používané proměnné jsou vyhodnocovány v interpolačním taktu. Proměnné hlavního zpracování v synchronních akcích nezpůsobují zastavení předběžného zpracování.

Vyhodnocování:

Jestliže se ve výrobním programu vyskytují proměnné hlavního zpracování (např. skutečná hodnota, stav digitálního vstupu nebo výstupu atd.), předběžné zpracování se pozastaví, dokud není zpracován předcházející blok a dokud nejsou proměnné hlavního zpracování k dispozici.

Příklady

Příklad 1: Žádné klíčové slovo

Programový kód	Komentář
`DO $A_OUTA[1]=$AA_IN[X]`	`; Přenášení skutečné hodnoty na analogový výstup.`

Příklad 2: WHENEVER

Programový kód	Komentář
`WHENEVER $AA_IM[X] > 10.5*SIN(45) DO …`	`; Porovnávání s výrazem vypočteným v rámci předběžného zpracování.`
`WHENEVER $AA_IM[X] > $AA_IM[X1] DO …`	`; Porovnávání s další proměnnou hlavního zpracování`
`WHENEVER ($A_IN[1]==1) OR ($A_IN[3]==0) DO ...`	`; Dvě spolu spojená porovnávání.`

Příklad 3: EVERY

Programový kód	Komentář
`ID=1 EVERY $AA_IM[B]>75 DO POS[U]=IC(10) FA[U]=900`	`; Vždy když je překročena skutečná hodnota osy B v MCS rovnající se 75, má se osa U nastavit axiálním posuvem dál o 10.`

Další informace

Podmínka

Podmínka představuje logický výraz, který může být libovolně sestaven pomocí booleovských operátorů. Booleovské výrazy by měly být vždy uvedeny v závorkách.

Podmínka je vyhodnocována v interpolačním taktu.

Před podmínkou může být uveden G-kód. Tímto způsobem může být dosaženo, aby nezávisle na momentálně aktivním stavu výrobního programu existovaly pro vyhodnocování podmínky a akci/technologický cyklus, které se mají uskutečnit, definovaná nastavení. Je nutno zajistit, aby synchronní akce nebyla ovlivňována okolními programovými vlivy, protože synchronní akce mají v libovolných okamžicích na základě splněné spouštěcí podmínky realizovat své činnosti v definovaném výchozím stavu.

Případy použití

Stanovení měřicího systému pro vyhodnocování podmínky a akce prostřednictvím G-kódů `G70`, `G71`, `G700`, `G710`.

G-kód zadaný u podmínky platí pro její vyhodnocování a také pro akci samotnou, jestliže v rámci akce není žádný vlastní G-kód uveden.

Na jednu část podmínky smí být naprogramován jen **jeden G-kód** ze skupiny G-kódů.

Možné podmínky

- Porovnávání proměnných hlavního zpracování (analogové/digitální vstupy/výstupy atd.).
- Booleovské spojení výsledků porovnávání
- Výpočet výrazů v reálném čase
- Čas/vzdálenost od začátku bloku
- Vzdálenost od konce bloku
- Změřené hodnoty, výsledky měření
- Hodnoty servomechanismů
- Rychlosti, stavové informace os

10.1.3 Akce (DO)

Funkce

V synchronních akcích může být naprogramována jedna nebo více akcí. Veškeré akce naprogramované v jednom bloku budou aktivní ve stejném interpolačním taktu.

Syntaxe

```
DO <akce1> <akce2> …
```

Význam

`DO`	Když je podmínka splněna, spouští akci nebo technologický cyklus.
`<Akce>`	Akce, které se spouští, když je splněna podmínka, jako např. přiřazení proměnné, aktivování vazby mezi osami, nastavení výstupu NCK, spuštění M-, S- a H-funkce, zadání naprogramovaného G-kódu, ...

G-kódy mohou být v synchronních akcích naprogramovány pro akce/technologické cykly. Tento G-kód v případě potřeby zadává pro všechny akce v bloku a v technologických cyklech jiný G-kód, než jaký je nastaven u podmínky. Pokud jsou technologické cykly v akční části, potom G-kód platí dál i po skončení technologického cyklu pro všechny potom následující akce až po následující G-kód s modální platností.

Na jednu akční část smí být naprogramován jen **jeden G-kód** ze skupiny G-kódů (`G70`, `G71`, `G700`, `G710`).

Příklad: Synchronní akce se dvěma akcemi

Programový kód	**Komentář**
`WHEN $AA_IM[Y]>=35.7 DO M135 $AC_PARAM=50`	`; Jestliže je podmínka splněna, na PLC se aktivuje funkce M135 a korekce (override) se nastaví na 50%.`

10.2 Operátory pro podmínky a akce

Porovnávání (==, <>, <, >, <=, >=)	V podmínkách mohou být porovnávány proměnné nebo dílčí výrazy. Výsledek je vždy datového typu BOOL. Všechny známé porovnávací operace jsou přípustné.
Booleovské operátory (NOT, AND, OR, XOR)	Proměnné, konstanty nebo porovnávané výrazy mohou být spolu spojovány známými Booleovskými operátory.
Bitové operátory (B_NOT, B_AND, B_OR, B_XOR)	Mohou se používat také bitové operátory B_NOT, B_AND, B_OR, B_XOR.
Základní aritmetické operace (+, -, *, /, DIV, MOD)	Proměnné hlavního zpracování mohou být spolu nebo s konstantami spojovány pomocí základních aritmetických operací.
Matematické funkce (SIN, COS, TAN, ASIN, ACOS, ABS, TRUNC, ROUND, LN, EXP, ATAN2, POT, SQRT, CTAB, CTABINV).	S proměnnými datového typu REAL se mohou používat matematické funkce.
Indexování	Indexování je ve výrazech hlavního zpracování možné.

Příklad

- **Spojení základních aritmetických funkcí**

Platí přednost násobení a dělení před sečítáním a odečítáním, použití závorek ve výrazech je přípustné. Operátory DIV a MOD jsou přípustné i pro proměnné typu REAL.

Programování	**Komentář**
DO $AC_PARAM[3] = $A_INA[1]-$AA_IM[Z1]	; ;Odečítání dvou ;proměnných hlavního zpracování
WHENEVER $AA_IM[x2] < $AA_IM[x1]-1.9 DO $A_OUT[5] = 1	; ;Odečtení konstanty od proměnné
DO $AC_PARAM[3] = $INA[1]-4*SIN(45.7 $P_EP[Y])*R4	;Konstantní výraz, vypočítává se v rámci předběžného zpracování

- **Matematické funkce**

Programování	**Komentář**
DO $AC_PARAM[3] = COS($AC_PARAM[1])	; ;

- **Výrazy vyhodnocované v reálném čase**

Programování	Komentář
`ID=1 WHENEVER ($AA_IM[Y]>30) AND ($AA_IM[Y]<40) DO $AA_OVR[S1]=80`	`; Volba okna pro pozici`
`ID=67 DO $A_OUT[1]=$A_IN[2] XOR $AN_MARKER[1]`	`; Vyhodnocení 2 booleovských signálů`
`ID=89 DO $A_OUT[4]=$A_IN[1] OR ($AA_IM[Y]>10)`	`; Výstup výsledku porovnávání`

- **Indexování proměnných hlavního zpracování**

Programování	Komentář
`WHEN…DO $AC_PARAM[$AC_MARKER[1]] = 3`	`;`
`Nepřípustné je`	`;`
`$AC_PARAM[1] = $P_EP[$AC_MARKER]`	`;`

10.3 Proměnné hlavní větve programu pro synchronní akce

10.3.1 Systémové proměnné

Funkce

Pomocí systémových proměnných je možné provádět čtení a zápis dat NC systému. Systémové proměnné se dělí na proměnné předběžného a hlavního zpracování. Proměnné předběžného zpracování jsou vždy uvedeny v okamžiku předběžného zpracování. Proměnné hlavního zpracování zjišťují svou hodnotu vždy vzhledem k aktuálnímu stavu hlavního zpracování.

Názvy

Název systémových proměnných začíná většinou znakem $.

Proměnné předběžného zpracování:

$M...	Strojní parametry
$S...	Nastavované parametry, chráněné oblasti
$T...	Parametry správy nástrojů
$P...	Programovatelné hodnoty, data předběžného zpracování
$C...	Proměnné cyklů v cyklech ISO
$O...	Data volitelných doplňků
R ...	R-Parametry

Proměnné hlavního zpracování:

$$A...	Aktuální data hlavního zpracování
$$V...	Data servomechanismů
$R...	R-Parametry

2. znak popisuje možnosti přístupu k proměnné:

N...	Hodnota globální v NCK (všeobecně platná hodnota)
C...	Kanálová hodnota
A...	Osová hodnota

2. znak se většinou vyhodnocuje pouze u proměnných hlavního zpracování. Proměnné předběžného zpracování, jako např. $P_... se většinou uvádějí bez 2. znaku.

Za předponou (znak $, za nímž jsou jedno nebo dvě písmena) následuje vždy znak podtržení a potom název proměnné (většinou jako anglické označení nebo zkratka).

Datové typy

Proměnné hlavního zpracování mohou být následujících typů:

INT	Celočíselné hodnoty (integer) se znaménkem
REAL	Reálná čísla s desetinnou tečkou
BOOL	Booleovské hodnoty TRUE a FALSE
CHAR	Znaky ASCII
STRING	Řetězce znaků s alfanumerickými znaky
AXIS	Adresy os a vřeten

Proměnné předběžného zpracování mohou být kromě toho ještě následujícího typu:

FRAME	Transformace souřadného systému

Pole proměnných

Systémové proměnné mohou být založeny také jako 1- až 3-rozměrná pole.

Podporovány jsou následující datové typy: BOOL, CHAR, INT, REAL, STRING, AXIS

Indexy mohou být datových typu INT a AXIS, přičemž ty mohou být setřízeny libovolně.

Proměnné typu STRING mohou být pouze 2-rozměrné.

Příklady definice polí:

```
DEF BOOL $AA_NEWVAR[x,y,2]
DEF CHAR $AC_NEWVAR[2,2,2]
DEF INT $AC_NEWVAR[2,10,3]
DEF REAL $AA_VECTOR[x,y,z]
DEF STRING $AC_NEWSTRING[3,3]
DEF AXIS $AA_NEWAX[x,3,y]
```

Poznámka

Zobrazování 3-rozměrných systémových proměnných je možné neomezeně, pokud k dané systémové proměnné existuje proměnná BTSS.

10.3.2 Implicitní převádění typu

Funkce

Při přiřazování hodnot a předávání parametrů mohou být přiřazovány nebo předávány proměnné různých datových typů.

Implicitní převádění typu spouští interní konverzi typu dané hodnoty.

Možné konverze typu

na	REAL	INT	BOOL	CHAR	STRING	AXIS	FRAME
z							
REAL	ano	ano*	ano[1])	–	–	–	–
INT	ano	ano	ano[1])	–	–	–	–
BOOL	ano	ano	ano	–	–	–	–

Vysvětlení

* Při konverzi z typu REAL na typ INT se desetinná část >= 0,5 zaokrouhluje nahoru, jinak dolů (srov. s funkcí ROUND).
V případě překročení rozsahu hodnot se aktivuje alarm.

1) Hodnotě 0 odpovídá TRUE, hodnotě == 0 odpovídá FALSE

Výsledky

```
Převod z typu REAL nebo INTEGER do typu BOOL
Výsledek BOOL = TRUE          jestliže se hodnota typu REAL nebo INTEGER nerovná nule
Výsledek BOOL = FALSE         jestliže se hodnota typu REAL nebo INTEGER rovná nule
Převod z typu BOOL na typ REAL nebo INTEGER
Výsledek REAL TRUE            jestliže hodnota typu BOOL = TRUE (1)
Výsledek INTEGER = TRUE       jestliže hodnota typu BOOL = TRUE (1)
Převod z typu BOOL na typ REAL nebo INTEGER
Výsledek REAL FALSE)          jestliže hodnota typu BOOL = FALSE (0)
Výsledek INTEGER = FALSE      jestliže hodnota typu BOOL = FALSE (0)
```

Příklady implicitních převodů typu

```
Převod z typu z INTEGER do typu BOOL
$AC_MARKER[1] = 561
ID=1 WHEN $A_IN[1] == TRUE DO $A_OUT[0]=$AC_MARKER[1]


Převod z typu z REAL do typu BOOL
R401 = 100.542
WHEN $A_IN[0] == TRUE DO $A_OUT[2]=$R401


Převod z typu z BOOL do typu INTEGER
ID=1 WHEN $A_IN[2] == TRUE DO $AC_MARKER[4] = $A_OUT[1]]


Převod z typu z BOOL do typu REAL
R401 = 100.542
WHEN $A_IN[3] == TRUE DO $R10 = $A_OUT[3]
```

10.3.3 Proměnné GUD

Proměnné GUD, které jsou použitelné v synchronních akcích

Vedle specifických systémových proměnných se mohou v synchronních akcích používat také předem definované globální uživatelské proměnné pro synchronní akce (GUD pro synchronní akce). Počet GUD pro synchronní akce, které jsou uživateli k dispozici, může být nastaven prostřednictvím následujících strojních parametrů podle datových typů a přístupových oprávnění.

- MD18660 $MM_NUM_SYNACT_GUD_**REAL**[<x>] = <počet>
- MD18661 $MM_NUM_SYNACT_GUD_**INT**[<x>] = <počet>
- MD18662 $MM_NUM_SYNACT_GUD_**BOOL**[<x>] = <počet>
- MD18663 $MM_NUM_SYNACT_GUD_**AXIS**[<x>] = <počet>
- MD18664 $MM_NUM_SYNACT_GUD_**CHAR**[<x>] = <počet>
- MD18665 $MM_NUM_SYNACT_GUD_**STRING**[<x>] = <počet>

Prostřednictvím indexu <x> se zadává datový modul (přístupová oprávnění), pomocí hodnoty <počet> se udává počet GUD pro synchronní akce příslušného datového typu (REAL, INT, ...). V odpovídajícím datovém modulu se poté pro každý datový typ založí 1-rozměrná proměnná typu pole, jejíž název bude respektovat následující schéma: SYG_<datový typ><přístupová oprávnění>[<index>]:

Index <x>		Datový typ (MD18660 ... MD18665)					
	Modul	**REAL**	**INT**	**BOOL**	**AXIS**	**CHAR**	**STRING**
0	SGUD	SYG_RS[i]	SYG_IS[i]	SYG_BS[i]	SYG_AS[i]	SYG_CS[i]	SYG_SS[i]
1	MGUD	SYG_RM[i]	SYG_IM[i]	SYG_BM[i]	SYG_AM[i]	SYG_CM[i]	SYG_SM[i]
2	UGUD	SYG_RU[i]	SYG_IU[i]	SYG_BU[i]	SYG_AU[i]	SYG_CU[i]	SYG_SU[i]
3	GUD4	SYG_R4[i]	SYG_I4[i]	SYG_B4[i]	SYG_A4[i]	SYG_C4[i]	SYG_S4[i]
4	GUD5	SYG_R5[i]	SYG_I5[i]	SYG_B5[i]	SYG_A5[i]	SYG_C5[i]	SYG_S5[i]
5	GUD6	SYG_R6[i]	SYG_I6[i]	SYG_B6[i]	SYG_A6[i]	SYG_C6[i]	SYG_S6[i]
6	GUD7	SYG_R7[i]	SYG_I7[i]	SYG_B7[i]	SYG_A7[i]	SYG_C7[i]	SYG_S7[i]
7	GUD8	SYG_R8[i]	SYG_I8[i]	SYG_B8[i]	SYG_A8[i]	SYG_C8[i]	SYG_S8[i]
8	GUD9	SYG_R9[i]	SYG_I9[i]	SYG_B9[i]	SYG_A9[i]	SYG_C9[i]	SYG_S9[i]
kde i = 0 až (<počet> - 1) modul: _N_DEF_DIR/_N_ ... _DEF, např. pro SGUD => _N_DEF_DIR/_N_**SGUD**_DEF							

Vlastnosti

GUD pro synchronní akce mají následující vlastnosti:

- Čtení a zápis do GUD pro synchronní akce může probíhat v synchronních akcích a ve výrobních programech / cyklech.
- Ke GUD pro synchronní akce je možno mít přístup pomocí BTSS.
- GUD pro synchronní akce se vypisují na uživatelském rozhraní HMI v systémové oblasti "Parametry".
- GUD pro synchronní akce se mohou na HMI používat v aplikaci Wizard, v obrazovce proměnných a v protokolu proměnných.
- Velikost pole u GUD pro synchronní akce typu STRING je pevně definována na 32 (31 znaků + \0).
- I když manuálně nebyly založeny žádné definiční soubory pro globální uživatelská data (GUD), mohou být GUD pro synchronní akce definované pomocí strojních parametrů čteny z HMI v příslušném modulu GUD.

UPOZORNĚNÍ

Uživatelské proměnné (GUD, PUD, LUD) se stejným názvem jako GUD pro synchronní akce mohou být definovány jen tehdy (`DEF ... SYG_xy`), jestliže nejsou nastaveny parametry žádné GUD pro synchronní akce se stejným názvem (MD18660 - MD18665) . Tyto uživatelem definované GUD se **nesmí** používat v synchronních akcích.

Přístupová oprávnění

Přístupová oprávnění definovaná v definičním souboru GUD zůstávají i nadále v platnosti a vztahují se pouze na proměnné GUD definované v tomto definičním souboru GUD.

Chování při mazání

Jestliže je obsah určitého definičního souboru GUD nově aktivován, starý datový modul GUD v aktivním systému souborů se napřed vymaže. V konfiguraci nastavené GUD pro synchronní akce jsou přitom vráceny do svého původního stavu také. Tuto operaci je možné spustit také pomocí HMI v systémové oblasti "Služby" > "Definice a aktivování uživatelských dat".

10.3.4 Předdefinovaný identifikátor osy (NO_AXIS)

Funkce

Proměnné nebo parametry typu AXIS, kterým při inicializaci nebyla dosazena žádná hodnota, mohou být opatřeny definovanou předem stanovenou hodnotou. Nedefinované proměnné typu AXIS jsou inicializovány touto předdefinovanou hodnotou.

Platné názvy os, které nebyly inicializovány, je možno v synchronních akcích rozpoznat pomocí dotazu na proměnnou "NO_AXIS". Tento identifikátor neinicializované osy se přiřazuje identifikátoru předdefinované osy, která je nastavena v konfiguraci pomocí strojního parametru.

Výrobce stroje

Pomocí strojních parametrů musí být definován nejméně jeden platný už existující identifikátor osy, kterému musí být dosazena nějaká hodnota. Hodnoty ale mohou být dosazeny i všem existujícím platným identifikátorům os. Věnujte prosím pozornost informacím od výrobce stroje.

Poznámka

Nově založená proměnná nyní automaticky dostává přiřazenu hodnotu pro předdefinovaný název osy, která je uložena ve strojním parametru.

Pokud budete potřebovat další informace týkající se platné definice podle strojního parametru, viz:
Literatura:
/FBSY/, Příručka Popis funkcí, Synchronní akce

Syntaxe

```
PROC UP(AXIS PAR1=NO_AXIS, AXIS PAR2=NO_AXIS)
IF PAR1 <>NO_AXIS...
```

Význam

PROC	Definice podprogramu
UP	Název podprogramu pro rozpoznání
PARn	Parametr n
NO_AXIS	Inicializace formálního parametru s předdefinovaným identifikátorem osy

Příklad: Definice proměnné typu AXIS v hlavním programu

Programový kód

```
DEF AXIS AXVAR
UP( , AXVAR)
```

10.3.5 Ukazatel pro synchronní akce ($AC_MARKER[n])

Funkce

Proměnnou typu pole $AC_MARKER[n] je možné v synchronních akcích číst a i zapisovat do ní. Tyto proměnné mohou být uloženy buď v paměti aktivního nebo pasivního systému souborů.

Proměnná synchronní akce: Datový typ INT

$AC_MARKER[n]	Kanálový ukazatel/čítač datového typu INTEGER
$MC_MM_NUM_AC_MARKER	Strojní parametr pro nastavení počtu kanálových ukazatelů pro pohybové synchronní akce
n	Index pole proměnných 0-n

Příklad čtení a zápisu do proměnné typu ukazatel

Programový kód

```
WHEN ... DO $AC_MARKER[0] = 2
WHEN ... DO $AC_MARKER[0] = 3
WHENEVER $AC_MARKER[0] == 3 DO $AC_OVR=50
```

10.3.6 Parametry synchronních akcí ($AC_PARAM[n])

Funkce

Parametry synchronní akce $AC_PARAM[n] slouží pro výpočty a jako pomocná paměť v synchronních akcích. Tyto proměnné mohou být uloženy buď v paměti aktivního nebo pasivního systému souborů.

Proměnná synchronní akce: Datový typ REAL

Tyto parametry jsou k dispozici pod stejným názvem jedenkrát v každém kanálu.

`$AC_PARAM[n]`	Matematické proměnné pro pohybové synchronní akce (REAL)
`$MC_MM_NUM_AC_PARAM`	Strojní parametr pro nastavení počtu parametrů pro pohybové synchronní akce, až maximálně 20000.
`n`	Index pole parametrů 0n

Příklad pro parametry synchronních akcí $AC_PARAM[n]

Programový kód

```
$AC_PARAM[0]=1.5
$AC_MARKER[0]=1
ID=1 WHEN $AA_IW[X]>100 DO $AC_PARAM[1]=$AA_IW[X]
ID=2 WHEN $AA_IW[X]>100 DO $AC_MARKER[1]=$AC_MARKER[2]
```

10.3.7 Početní parametr ($R[n])

Funkce

Toto statické pole proměnných slouží pro výpočty ve výrobním programu a v synchronních akcích.

Syntaxe

Programování ve výrobním programu:

```
REAL R[n]
REAL Rn
```

Programování v synchronních akcích:

```
REAL $R[n]
REAL $Rn
```

Početní parametry

Použití početních parametrů umožňuje následující:

- Ukládání do paměti hodnot, které mají zůstat zachovány i po skončení programu, přes reset NC systému a vypnutí a zapnutí celého systému.
- Zobrazování uložených hodnot na obrazovce R-parametrů.

Příklady

Programový kód	**Komentář**
`WHEN $AA_IM[X]>=40.5 DO $R10=$AA_MM[Y]`	`; Použití parametru R10 v synchronní akci.`
`G01 X500 Y70 F1000`	
`STOPRE`	`; Zastavení předběžného zpracování`
`IF R10>20`	`; Vyhodnocování početní proměnné.`

Programový kód

```
SYG_AS[2]=X
SYG_IS[1]=1
WHEN $AA_IM[SGY_AS[2]]>10 DO $R3=$AA_EG_DENOM[SYG_AS[1]],SYG_AS[2]]
WHEN $AA_IM[SGY_AS[2]]>12 DO $AA_SCTRACE[SYG_AS[2]]=1
SYG_AS[1]=X
SYG_IS[0]=1
WHEN $AA_IM[SGY_AS[1]]>10 DO $R3=$$MA_POSCTRL_GAIN[SYG_IS[0]],SYG_AS[1]]
WHEN $AA_IM[SGY_AS[1]]>10 DO $R3=$$MA_POSCTRL_GAIN[SYG_AS[1]]
WHEN $AA_IM[SGY_AS[1]]>15 DO $$MA_POSCTRL_GAIN[SYG_AS[0]], SYG_AS[1]]=$R3
```

10.3.8 Čtení a zápis strojních a nastavovaných parametrů NC systému

Funkce

Čtení a zápis strojních a nastavovaných parametrů NC systému je možno provádět i ze synchronních akcí. Při čtení a zápisu prvků pole strojních parametrů může být při programování index vypuštěn. Pokud se to stane ve výrobním programu, bude při čtení zjištěn obsah **prvního** prvku pole a při zápisu se daná hodnota zapíše do všech prvků pole.

V synchronních akcích se oproti tomu v tomto případě čte na zapisuje jen do **prvního** prvku.

Definice

MD, SD

`$`: Čtení hodnoty v okamžiku interpretace synchronní akce

`$$`: Čtení hodnoty v hlavním zpracování

Čtení hodnot MD a SD v okamžiku předběžného zpracování

Tyto parametry jsou adresovány ze synchronní akce pomocí znaku $ a jsou vyhodnocovány v okamžiku předběžného zpracování.

```
ID=2 WHENEVER $AA_IM[z]<$SA_OSCILL_REVERSE_POS2[Z]-6 DO $AA_OVR[X]=0
;Tady se aktivuje oblast bodu obratu 2 pro oscilační pohyb, která je považována za
neměnnou.
```

Čtení hodnot MD a SD v okamžiku hlavního zpracování

Tyto parametry jsou adresovány ze synchronní akce pomocí znaku $$ a jsou vyhodnocovány v okamžiku hlavního zpracování.

```
ID=1 WHENEVER $AA_IM[z]<$$SA_OSCILL_REVERSE_POS2[Z]-6 DO $AA_OVR[X]=0
;Zde se vychází z toho, že bod obratu by mohl být v průběhu zpracování
prostřednictvím obsluhy změněn.
```

Zápis hodnot do parametrů MD a SD v okamžiku hlavního zpracování

Momentálně nastavená přístupová oprávnění musí dovolovat přístup za účelem zápisu. Platnost pro všechny parametry MD a SD je uvedena, viz **Literatura:** /LIS/, Seznamy (svazek 1).

Parametry `MD` a `SD`, do nichž se má zapisovat, musí být uvedeny znaky `$$`.

Příklad

Programový kód	**Komentář**
`ID=1 WHEN $AA_IW[X]>10 DO $$SN_SW_CAM_PLUS_POS_TAB_1[0]=20`	`; Změna spínací pozice softwarové vačky. Upozornění: Spínací pozice musí být změněna 2-3 takty IPO před dosažením místa, kde by mohlo k sepnutí dojít.`
`$$SN_SW_CAM_MINUS_POS_TAB_1[0]=30`	

10.3.9 Proměnné časovače ($AC_Timer[n])

Funkce

Systémová proměnná $AC_TIMER[n] umožňuje spouštění určitých akcí po uplynutí definované čekací lhůty.

Proměnná typu časovač: Datový typ REAL

`$AC_TIMER[n]`	Kanálový časovač datového typu REAL
`s`	Údaj v sekundách
`n`	Index proměnné typu časovač

Nastavování časovače

Počítání proměnné typu časovač směrem nahoru se spouští prostřednictvím přiřazení hodnoty:
`$AC_TIMER[n] = hodnota`

n:	Číslo časové proměnné
hodnota:	Počáteční hodnota (zpravidla "0")

Zastavení časovače

Počítání proměnné typu časovač směrem nahoru se zastavuje prostřednictvím přiřazení záporné hodnoty:
`$AC_TIMER[n]=-1`

Čtení časovače

Momentální časový údaj je možné z proměnné typu časovač přečíst bez ohledu na to, jestli je časovač zastaven nebo spuštěn. Po zastavení proměnné typu časovač pomocí přiřazení hodnoty -1 zůstává poslední aktuální časová hodnota zachována a je možné ji i později přečíst.

Příklad

Výstup skutečné hodnoty prostřednictvím analogového výstupu 500 ms po zachycení signálu digitálního vstupu:

Programový kód	**Komentář**
`WHEN $A_IN[1]==1 DO $AC_TIMER[1]=0` `WHEN $AC_TIMER[1]>=0.5 DO $A_OUTA[3]=$AA_IM[X] $AC_TIMER[1]=-1`	`; Vynulování a spuštění časovače`

10.3.10 Proměnné typu FIFO ($AC_FIFO1[n] ... $AC_FIFO10[n])

Funkce

Pro ukládání posloupností na sebe navazujících dat je k dispozici 10 proměnných typu FIFO (cyklická paměť).
Datový typ: REAL

Použití:

- Cyklická měření
- Průběžné opracovávání

Za účelem čtení a zápisu je možné mít přístup ke kterémukoli prvku.

Proměnná typu FIFO

Počet proměnných typu FIFO, které jsou k dispozici, je definován pomocí strojního parametru MD28260 $MC_NUM_AC_FIFO.

Počet hodnot, které mohou být do proměnné typu FIFO zapsány, je nastavován strojním parametrem MD28264 $MC_LEN_AC_FIFO. Všechny proměnné typu FIFO mají stejnou délku.

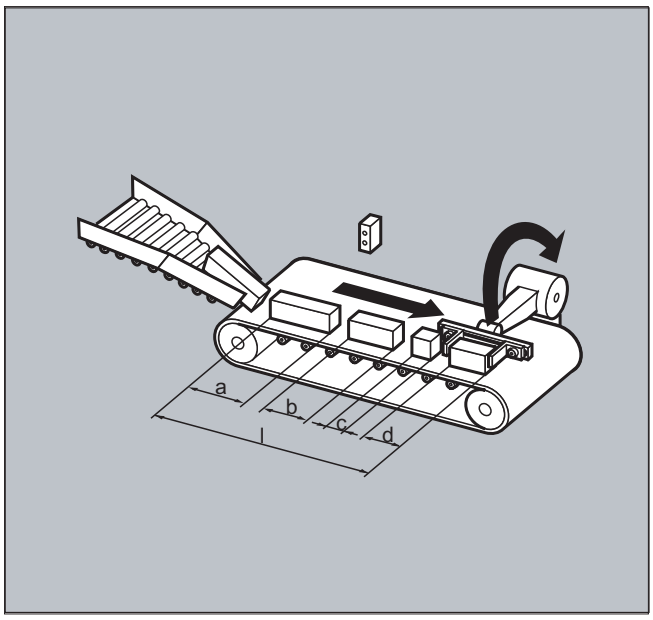
Součet všech prvků FIFO se generuje jen tehdy, pokud je nastaveno MD28266 $MC_MODE_AC_FIFO bit 0.

Indexy **0 až 5** mají zvláštní význam.

Index	Význam	
0	Při zápisu:	Nová hodnota se uloží do proměnné typu FIFO.
	Při čtení:	Nejstarší prvek se přečte a odstraní z paměti FIFO.
1	Přístup k nejstaršímu uloženému prvku	
2	Přístup k nejmladšímu uloženému prvku	
3	Součet všech prvků FIFO	
4	Počet prvků, které jsou ve FIFO k dispozici Za účelem čtení a zápisu je možné mít přístup ke kterémukoli prvku FIFO. Vynulování proměnných typu FIFO se uskutečňuje dosazením nuly pro počet prvků, např. pro první proměnnou typu FIFO: $AC_FIFO1[4] = 0	
5	Aktuální index zápisu vzhledem k počátku paměti FIFO	
6 až n_{max}	Přístup k n-tému prvku paměti FIFO	

Příklad: Cyklická paměť

V průběhu výrobního procesu se pro transport produktů používá pásový dopravník s různými délkami (a, b, c, d). Na pásovém dopravníku s danou přepravní délkou se proto současně přepravuje různý počet příslušných produktů v závislosti na jejich délce. Jestliže je rychlost dopravníku stále stejná, musí být odběr produktů z pásu přizpůsoben proměnným časům jejich příchodu.



Programový kód	**Komentář**
DEF REAL ZWI=2.5	; Konstantní vzdálenost mezi odkládanými produkty.
DEF REAL GESAMT=270	; Vzdálenost mezi polohou pro měření délky a polohou pro odběr.
EVERY $A_IN[1]==1 DO $AC_FIFO1[4]=0	; Vynulování proměnné FIFO na začátku procesu.
EVERY $A_IN[2]==1 DO $AC_TIMER[0]=0	; Jakmile produkt přeruší světelnou závoru, spustí se měření času.
EVERY $A_IN[2]==0 DO $AC_FIFO1[0]=$AC_TIMER[0]*$AA_VACTM[B]	; Když se světelná závora uvolní, ze změřeného času a rychlosti pásového dopravníku se vypočítá délka produktu, která se uloží do paměti FIFO.
EVERY $AC_FIFO1[3]+$AC_FIFO1[4]*ZWI>=GESAMT DO POS[Y]=-30 $R1=$AC_FIFO1[0]	; Jakmile je součet všech délek produktu a vzdáleností mezi nimi vetší/roven vzdálenosti mezi místem odkládání a místem odebírání, odebere se produkt nacházející se na místě pro odběr z pásového dopravníku, odpovídající délka produktu se přečte z paměti FIFO.

10.3.11 Informace o typu bloku v interpolátoru ($AC_BLOCKTYPE, $AC_BLOCKTYPEINFO, $AC_SPLITBLOCK)

Funkce

V synchronních akcích jsou k dispozici následující systémové proměnné, které obsahují informace o aktuálním bloku, který se právě nachází v hlavním zpracování.

- $AC_BLOCKTYPE
- $AC_BLOCKTYPEINFO
- $AC_SPLITBLOCK

Proměnné pro typ bloku a informace o typu bloku

$AC_BLOCKTYPE		$AC_BLOCKTYPEINFO				
Hodnota:		Hodnota:				
0	nerovná se 0	T	H	Z	E	Význam:
Původní blok	Pomocný blok					Spouštěcí prvek pro pomocný blok:
	1	1	0	0	0	Interně generovaný blok, žádné další informace
	2	2	0	0	1	Fasety/zaoblení: Přímka
	2	2	0	0	2	Fasety/zaoblení: Kruh
	3	3	0	0	1	WAB: Najíždění po přímce
	3	3	0	0	2	WAB: Najíždění po čtvrtkruhu
	3	3	0	0	3	WAB: Najíždění po půlkruhu
						Korekční parametry nástroje:
	4	4	0	0	1	Blok najíždění po STOPRE
	4	4	0	0	2	Spojovací bloky v případě nenalezeného průsečíku
	4	4	0	0	3	Bodový kruh na vnitřních rozích (jen u TRACYL)
	4	4	0	0	4	Oblouková dráha na vnějších rozích (příp. kuželosečka)
	4	4	0	0	5	Najížděcí bloky při potlačení korekce
	4	4	0	0	6	Najížděcí bloky při opětovném aktivování korekce rádiusu nástroje
	4	4	0	0	7	Rozštěpení blokuv důsledku příliš vysokého zakřivení
	4	4	0	0	8	Vyrovnávací bloky při 3D frézování na čelní ploše (vektor nástroje || vektorem plochy)

$AC_BLOCKTYPE		$AC_BLOCKTYPEINFO				
Hodnota:		Hodnota:				
0	nerovná se 0	T	H	Z	E	Význam:
Původní blok	Pomocný blok					Spouštěcí prvek pro pomocný blok:
						Přechodová zaoblení pomocí příkazu:
	5	5	0	0	1	G641
	5	5	0	0	2	G642
	5	5	0	0	3	G643
	5	5	0	0	4	G644
						Blok TLIFT s:
	6	6	0	0	1	Lineární pohyb tangenciální osy a bez pozvedávacího pohybu
	6	6	0	0	2	Nelineární pohyb tangenciální osy (polynom) a bez pozvedávacího pohybu
	6	6	0	0	3	Pozvednutí nástroje, pohyb tangenciální osy a pozvednutí se spouští současně
	6	6	0	0	4	Pozvednutí nástroje, tangenciální osa se spouští napřed, jakmile je dosaženo určité pozice pozvednutí.
						Rozdělení dráhy:
	7	7	0	0	1	Naprogramované rozdělení dráhy bez prostřihování nebo lisování je aktivní
	7	7	0	0	2	Naprogramované rozdělení dráhy s aktivním lisováním nebo prostřihováním
	7	7	0	0	3	Automaticky interně generované rozdělení dráhy
						Cykly překladače:
	8	ID aplikace				ID aplikace kompilačních cyklů, kterou blok vytvořil.

T: Místo tisíců
H: Místo stovek
Z: Místo desítek
E: Místo jednotek

Poznámka

Proměnná $AC_BLOCKTYPEINFO obsahuje na místě tisíců (T) vždy také hodnotu pro typ bloku pro případ, že by existoval pomocný blok. Do proměnné $AC_BLOCKTYPE nerovnající se 0 se místo tisíců nepřebírá.

$AC_SPLITBLOCK	
Hodnota:	**Význam:**
0	Nezměněný naprogramovaný blok (s blokem vygenerovaným kompresorem se zachází stejně jako s naprogramovaným blokem)
1	Jedná se o interně vygenerovaný blok nebo zkrácený původní blok
3	Jedná se o poslední blok v řetězci interně vygenerovaných bloků nebo zkrácených původních bloků

Příklad: Počítání bloků přechodových zaoblení

```
Programový kód          Komentář

$AC_MARKER[0]=0
$AC_MARKER[1] = 0
$AC_MARKER[2] = 0
...
                        ; Definice synchronní akce, pomocí které
                          jsou bloky přechodových zaoblení počítány.
                        ; Všechny bloky přechodových zaoblení se počítají v proměnné $AC_MARKER[0]:
ID=1 WHENEVER ($AC_TIMEC==0) AND ($AC_BLOCKTYPE==5) DO $AC_MARKER[0]=$AC_MARKER[0]+1
...
                        ; Bloky přechodových zaoblení vytvořené pomocí G641 se počítají v proměnné
                          $AC_MARKER[1]:
ID=2 WHENEVER ($AC_TIMEC==0) AND ($AC_BLOCKTYPEINFO==5001) DO $AC_MARKER[1]=$AC_MARKER[1]+1
                        ; Bloky přechodových zaoblení vytvořené pomocí G642 se počítají v proměnné
                          $AC_MARKER[2]:
ID=3 WHENEVER ($AC_TIMEC==0) AND ($AC_BLOCKTYPEINFO==5002) DO $AC_MARKER[2]=$AC_MARKER[2]+1
...
```

10.4 Akce v synchronních akcích

10.4.1 Přehled možných akcí v synchronních akcích

Akce v synchronních akcích se skládají z přiřazení hodnot, volání funkcí nebo parametrů, klíčových slov nebo technologických cyklů. Pomocí operátorů lze vytvořit i složité zpracovávané celky.

Možné případy použití jsou následující:

- Výpočet složitých výrazů v taktu IPO
- Pohyby os a řízení vřeten
- On-line změny nebo vyhodnocování nastavovaných parametrů ze synchronních akcí (např. předávání poloh a časů softwarových vaček na PLC nebo periferií NC systému)
- Výstup pomocných funkcí do PLC
- Instalace doplňkových bezpečnostních funkcí
- Nastavování superponovaných pohybů, on-line korekcí nástroje a regulace vzdálenosti
- Provádění akcí ve všech provozních režimech
- Ovlivňování synchronních akcí z PLC
- Provádění technologických cyklů
- Předávání digitálních a analogových signálů
- Zaznamenávání informací o provádění synchronních akcí v interpolačním taktu a záznam doby výpočtu regulátoru polohy pro vyhodnocování zatížení
- Diagnostické možnosti na uživatelském rozhraní

Synchronní akce	Popis
DO $V…= DO $A...=	Přiřazení (hodnoty servomechanismů) Přiřazení proměnné (proměnná hlavního zpracování)
DO $AC…[n]= DO $AC_MARKER[n]= DO $AC_PARAM[n]=	Speciální proměnná hlavního zpracování Čtení nebo zápis maker synchronních akcí Čtení nebo zápis parametrů synchronních akcí
DO $R[n]=	Čtení nebo zápis početní proměnné
DO $MD...= DO $$SD...=	Čtení proměnné typu MD v okamžiku interpolace Zápis hodnoty SD v hlavním zpracování
DO $AC_TIMER[n]=počáteční hodnota	Časovač
DO $AC_FIFO1[n] …FIFO10[n]=	Proměnná typu FIFO
DO $AC_BLOCKTYPE= DO $AC_BLOCKTYPEINFO= DO $AC_SPLITBLOCK=	Interpretace aktuálního bloku (proměnná hlavního zpracování)
DO M-, S a H z. B. M07	Výstup pomocných M-, S- a H-funkcí
DO RDISABLE	Aktivování blokování operace načtení
DO STOPREOF	Odblokování zastavení předběžného zpracování

Synchronní akce	Popis
DO DELDTG	Rychlé vymazání zbytkové dráhy bez zastavení předběžného zpracování
FTCDEF(Polyn., LL, UL , Koefic.) DO SYNFCT(Polyn., Output, Input)	Definice polynomů Aktivování synchronních funkcí: AC-regulace
DO FTOC	On-line korekce nástroje
DO G70/G71/G700/G710	Definice systému měřicích jednotek pro úlohy nastavování polohy (jednotkami jsou palce nebo metrické jednotky)
DO POS[osa]= / DO MOV[osa]= DO SPOS[vřeteno]=	Spouštění/nastavování polohy/zastavování příkazových os Spouštění/nastavování polohy/zastavování vřeten
DO MOV[osa]=hodnota	Spuštění/zastavení nekonečného pohybu příkazové osy
DO POS[osa]= FA [osa]=	Posuv osy FA
ID=1 ... DO POS[osa]= FA [osa]= ID=2 ... DO POS[osa]= $AA_IM[osa] FA [osa]=	Nastavování polohy ze synchronních akcí
PRESETON(osa, hodnota)	Nastavení skutečné hodnoty (předvolba ze synchronních akcí)
ID=1 EVERY $A_IN[1]=1 DO M3 S… ID=2 EVERY $A_IN[2]=1 DO SPOS=	Spouštění/nastavování polohy/zastavování vřeten
DO TRAILON(FA,LA,faktor vazby) DO LEADON(FA,LA,NRCTAB,OVW)	Aktivování vlečení Aktivování vazby řídící hodnotou
DO MEAWA(osa)= DO MEAC(osa)=	Aktivování axiálního měření Aktivování kontinuálního měření
DO [pole n, m]=SET(hodnota, hodnota, ...) DO [pole n, m]=REP(hodnota, hodnota, ...)	Inicializace proměnných typu pole pomocí seznamu hodnot Inicializace proměnných typu pole pomocí stejné hodnoty
DO SETM(č. značky) DO CLEARM(č. značky)	Nastavování čekacích značek Vymazání čekací značky
DO SETAL(č. alarmu)	Aktivování alarmu cyklu (doplňková bezpečnostní funkce)
DO FXS[osa]= DO FXST[osa]= DO FXSW[osa]= DO FOCON[osa]= DO FOCOF[osa]=	Aktivování najíždění na pevný doraz Změna blokovacího momentu Změna monitorovacího okna Aktivování najíždění s omezeným momentem/silou (modální) FOC Deaktivování najíždění s omezeným momentem/silou (synchronní akce se uplatňuje v daném bloku)
ID=2 EVERY $AC_BLOCKTYPE==0 DO $R1=$AC_TANEB	Úhel mezi tečnou ke dráze v koncovém bodě aktuálního bloku a tečnou ke dráze v počátečním bodě naprogramovaného následujícího bloku
DO $AA_OVR= DO $AC_OVR= DO $AA_PLC_OVR DO $AC_PLC_OVR DO $AA_TOTAL_OVR DO $AC_TOTAL_OVR	Override (korekce) osy Korekce (override) dráhy Override (korekce) osy zadaná z PLC Override (korekce) dráhy zadaná z PLC Výsledná override (korekce) osy Výsledná override (korekce) dráhy

Synchronní akce	Popis
$AN_IPO_ACT_LOAD=	Aktuální výpočetní čas IPO
$AN_IPO_MAX_LOAD=	Nejdelší výpočetní čas IPO
$AN_IPO_MIN_LOAD=	Nejkratší výpočetní čas IPO
$AN_IPO_LOAD_PERCENT=	Aktuální výpočetní čas IPO v poměru k taktu IPO
$AN_SYNC_ACT_LOAD=	Aktuální výpočetní čas pro synchronní akci přes všechny kanály
$AN_SYNC_MAX_LOAD=	Nejdelší výpočetní čas pro synchronní akci přes všechny kanály
$AN_SYNC_TO_IPO=	Procentuální podíl celé synchronní akce
DO TECCYCLE	Vyvolání technologického cyklu
DO LOCK(n, n, ...)	Zablokování
DO UNLOCK(n, n, ...)	Odblokování
DO RESET(n, n, ...)	RESET technologického cyklu
CANCEL(n, n, ...)	Vymazání modální synchronní akce s označením ID(S) ve výrobním programu

10.4.2 Výstup pomocných funkcí

Funkce

Okamžik výstupu

Výstup pomocných funkcí se v synchronní akci uskutečňuje bezprostředně po okamžiku výstupu akce. Okamžik výstupu pro pomocné funkce definovaný pomocí strojních parametrů nemá žádný význam.

Pokud je podmínka splněna, okamžik výstupu je potom v platnosti.

Příklad:

Zapnutí chladicí kapaliny na určité pozici osy:

```
WHEN $AA_IM[X]>=15 DO M07 POS[X]=20 FA[X]=250
```

Povolená klíčová slova v synchronních akcích s blokovou platností (bez modálního ID)

Pomocné funkce v synchronních akcích s blokovou platností (bez modálního ID) mohou být programovány pouze s klíčovými slovy `WHEN` nebo `EVERY`.

Poznámka

Následující pomocné funkce jsou v synchronní akci nepřípustné:

- `M0`, `M1`, `M2`, `M17`, `M30`: zastavení/konec programu (`M2`, `M17`, `M30` může být v technologickém cyklu)
- `M6`, příp. pomocí strojního parametru nastavené M-funkce pro výměnu nástroje

Příklad

Programový kód	Komentář
`WHEN $AA_IW[Q1]>5 DO M172 H510`	`; Pokud je překročena skutečná hodnota osy Q1 5 mm, do PLC jsou odeslány pomocné funkce M172 a H510.`

10.4.3 Aktivování blokování načítání (RDISABLE)

Funkce

Když je splněna podmínka, pomocí funkce `RDISABLE` se další příprava bloků v hlavním programu pozastaví. Naprogramované pohybové synchronní akce jsou zpracovávány dále, následující bloky jsou dále připravovány.

Na konci bloku s příkazem `RDISABLE` se aktivuje přesné najetí, a to bez ohledu na to, zda je zablokování načítání v platnosti nebo není. Přesné najetí se aktivuje také tehdy, pokud se řídící systém nalézá v režimu řízení pohybu po dráze (G64, G641 ... G645).

Použití

Pomocí příkazu `RDISABLE` je např. možné spouštět určitý program v interpolačním taktu nezávisle na externích vstupech.

Příklad

Programový kód	Komentář
`WHENEVER $A_INA[2]<7000 DO RDISABLE`	`; Jestliže je na vstupu 2 překročeno napětí 7 V, další pokračování zpracování programu bude pozastaveno (předpoklad: hodnota 1000 odpovídá 1 V).`
`...`	
`N10 G01 X10`	`; Příkaz RDISABLE na konci bloku N10 se uskuteční, pokud je v průběhu zpracování splněna jeho podmínka.`
`N20 Y20`	
`...`	

Okrajové podmínky

Chování příkazu RDISABLE při výměně osy

Jestliže platí příkaz RDISABLE v bloku, v němž se uskutečňuje také výměna osy, uplatňuje se příkaz RDISABLE také na na blok s příkazem REPOSA vyvolávaný výměnou osy.

Příklad programu:

Programový kód

```
N100 G0 G60 X300 Y300
N105 WHEN TRUE DO POS[X]=20 FA[X]=20000
N110 WHENEVER $AA_IM[X]<>20 DO RDISABLE
N115 G0 Y20
N120 Y-20
N125 M30
```

Prostřednictvím synchronní akce osa X odjíždí od dráhy a spouští se příkaz REORG (REPOSA). Funkce RDISABLE platí pro operaci REPOSA. V důsledku toho napřed najíždí na svou pozici osa X a potom se v bloku N115 najíždí na Y20.

Příkazu REORG se lze vyhnout, jestliže se v bloku N101 naprogramuje příkaz `RELEASE(X)` nebo `WAITP(X)`, protože v důsledku toho je osa X uvolňována pro pohyb např. jako příkazová osa.

Programový kód

```
N100 G0 G60 X300 Y300
N101 RELEASE(X)
N105 WHEN TRUE DO POS[X]=20 FA[X]=20000
...
```

10.4.4 Zrušení zastavení předběžného zpracování (STOPREOF)

Funkce

V případě explicitně naprogramovaného zastavení předběžného zpracování (STOPRE) nebo implicitně aktivovaném zastavení předběžného zpracování, které bylo vyvoláno aktivní synchronní akcí, pak pokud je splněna podmínka, příkaz STOPREOF po následujícím zpracovávaném bloku zastavení předběžného zpracování odstraní.

Poznámka

Příkaz STOPREOF musí být naprogramován pomocí klíčového slova `WHEN` a s blokovou platností (bez identifikačního (ID) čísla).

Příklad

Rychlé větvení programu na konci bloku.

Programový kód	Komentář
`WHEN $AC_DTEB<5 DO STOPREOF`	`; Jestliže je překročena vzdálenost od konce ;bloku 5 mm, odstranit zastavení předběžného zpracování.`
`G01 X100`	`; Po uskutečnění lineární interpolace se zastavení předběžného zpracování odstraní.`
`IF $A_INA[7]>500 GOTOF MARKE1=X100`	`; Jestliže je překročeno napětí 5 V na vstupu 7, ;skočit na návěští 1.`

10.4.5 Vymazání zbytkové dráhy (DELDTG)

Funkce

V závislosti na podmínce může být provedeno vymazání zbytkové dráhy pro dráhu a pro zadané osy.

K dispozici jsou následující možnosti:

- Rychlé připravené vymazání zbytkové dráhy
- Vymazání zbytkové dráhy bez přípravy

Připravené vymazání zbytkové dráhy pomocí příkazu DELDTG umožňuje velmi rychlou reakci na spouštěcí událost, a proto se používá v časově kritických situacích, např. pokud:

- čas mezi vymazáním zbytkové dráhy a spuštěním následujícího bloku má být velmi krátký
- podmínka pro vymazání zbytkové dráhy je s velmi vysokou pravděpodobností splněna

Poznámka

Příkaz DELDTG, za nímž je v závorkách uveden název osy, platí jen pro **jednu** polohovací osu.

Syntaxe

Vymazání zbytkové dráhy pro dráhu
```
DO DELDTG
```

Vymazání zbytkové dráhy pro osy
```
DO DELDTG(osa1) DELDTG(osa2) ...
```

Příklad: Rychlého vymazání zbytkové dráhy pro dráhu

Programový kód	Komentář
`WHEN $A_IN[1]==1 DO DELDTG`	
`N100 G01 X100 Y100 F1000`	`;` Jestliže je vstup nastaven, pohyb se přeruší
`N110 G01 X…`	
`IF $AA_DELT>50…`	

Příklad: Rychlého vymazání zbytkové dráhy pro osy

Programový kód	Komentář
`Přerušení polohovacího pohybu:`	
`ID=1 WHEN $A_IN[1]==1 DO MOV[V]=3 FA[V]=700`	`;` Spuštění osy
`WHEN $A_IN[2]==1 DO DELDTG(V)`	`;` Vymazání zbytkové dráhy, pozastavení osy pomocí příkazu MOV=0.
`V závislosti na napětí vstupu vymazání zbytkové dráhy:`	
`WHEN $A_INA[5]>8000 DO DELDTG(X1)`	`;` Jakmile dojde k překročení napětí 8 V na vstupu 5, vymaže se zbytková dráha osy X1. Pohyb po dráze pokračuje dále.
`POS[X1]=100 FA[X1]=10 G1 Z100 F1000`	

Další informace

Na konci pohybového bloku, ve kterém bylo spuštěno připravené vymazání zbytkové dráhy, se implicitně aktivuje zastavení předběžného zpracování.

Režim řízení pohybu po dráze, příp. pohyby polohovacích os se v důsledku toho na konci bloku s rychlým vymazáním zbytkové dráhy přeruší, příp. zastaví.

Poznámka

Připravené vymazání zbytkové dráhy:

- nemůže být aplikováno, když je aktivní korekce rádiusu nástroje
- smí být naprogramováno jenom v synchronních akcích s blokovou platností (bez identifikačního (ID) čísla)

10.4.6 Definice polynomu (FCTDEF)

Funkce

Pomocí příkazu FCTDEF mohou být definovány polynomy 3. stupně ve formě $y=a_0+a_1x+a_2x^2+a_3x^3$. Tyto polynomy jsou využívány on-line korekcí nástroje FTOC a vyhodnocovací funkcí SYNFCT.

Syntaxe

```
FCTDEF(č. polynomu,LLIMIT,ULIMIT,a0,a1,a2,a3)
```

Význam

`č. polynomu`	Číslo polynomu 3. řádu
`LLIMIT`	Spodní mez pro hodnotu funkce
`ULIMIT`	Horní mez pro hodnotu funkce
`a0, a1, a2, a3`	Koeficienty polynomu

K těmto hodnotám je přístup i prostřednictvím systémové proměnné.

`$AC_FCTLL[n]`	Spodní mez pro hodnotu funkce
`$AC_FCTUL[n]`	Horní mez pro hodnotu funkce
`$AC_FCT0[n]`	a_0
`$AC_FCT1[n]`	a_1
`$AC_FCT2[n]`	a_2
`$AC_FCT3[n]`	a_3

Poznámka

Zápis do systémových proměnných

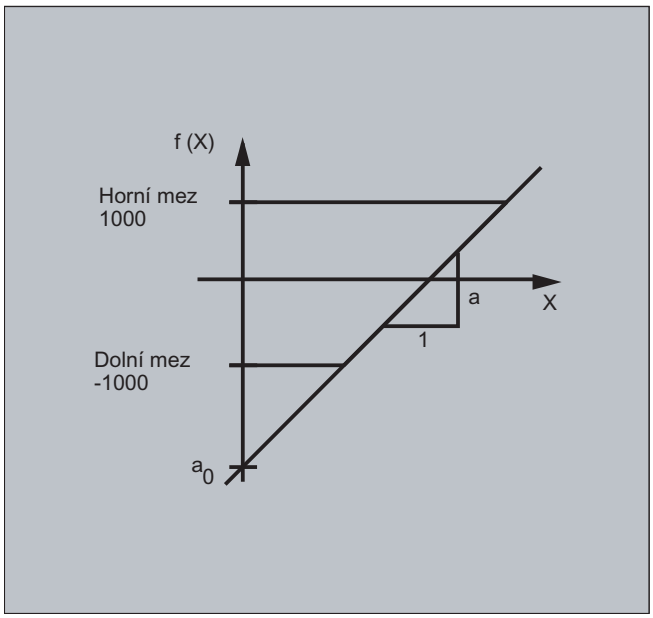
- Hodnoty je možné do systémových proměnných zapisovat z výrobního programu nebo i ze synchronní akce. Při zápisu z výrobního programu je nutno se prostřednictvím programového příkazu STOPRE postarat o to, aby zápis proběhl synchronně s blokem.
- Systémové proměnné `$AC_FCTLL[n]`, `$AC_FCTUL[n]`, `$AC_FCT0[n]` až `$AC_FCTn[n]` mohou být upravovány i ze synchronních akcí.

Při zápisu ze synchronní akce jsou koeficienty polynomu a meze hodnoty funkce okamžitě v platnosti.

Příklad: Polynomu pro přímkový úsek

S horní mezní hodnotou 1000, spodní mezní hodnotou -1000, s úsekem ordináty a_0=$AA_IM[X] a se směrnicí přímky 1 zní definice polynomu takto:

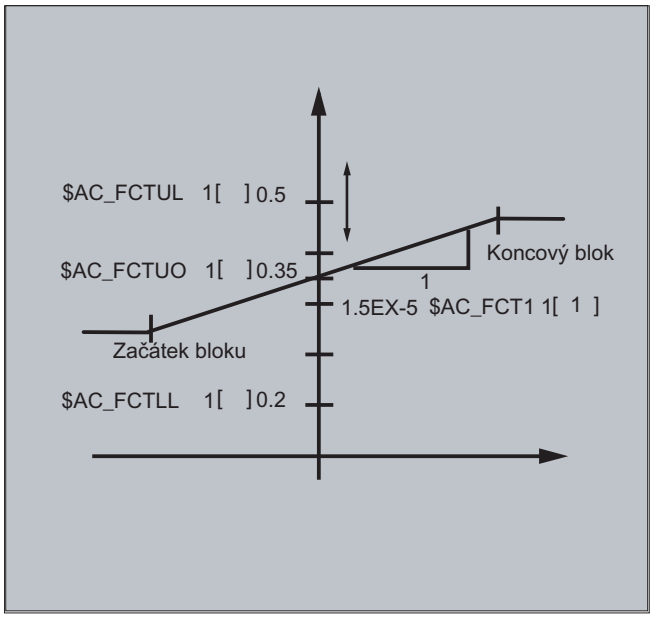
```
FCTDEF(1, -1000,1000,$AA_IM[X],1)
```



Příklad: Řízení výkonu laseru

Jedním z možných použití definice polynomu je řízení výkonu laseru.

Řízením výkonu laseru je míněno následující:
Ovlivňování analogového výstupu v závislosti např. na rychlosti pohybu po dráze.



Programový kód	**Komentář**
`$AC_FCTLL[1]=0.2`	`; Definice koeficientů polynomu`
`$AC_FCTUL[1]=0.5`	
`$AC_FCT0[1]=0.35`	
`$AC_FCT1[1]=1.5EX-5`	
`STOPRE`	
`ID=1 DO $AC_FCTUL[1]=$A_INA[2]*0.1 +0.35`	`; On-line změna horní mezní hodnoty.`
`ID=2 DO SYNFCT(1,$A_OUTA[1],$AC_VACTW)`	`; V závislosti na rychlosti pohybu po dráze (uložené v proměnné $AC_VACTW) je prostřednictvím analogového výstupu 1 řízen výkon laseru.`

Poznámka

Výše definovaný polynom je použit pomocí příkazu SYNFCT.

10.4.7 Synchronní funkce (SYNFCT)

Funkce

Funkce SYNFCT vypočítává výstupní hodnotu polynomu 3. stupně váženou pomocí vstupních proměnných. Výsledek se nachází ve výstupní proměnné a je shora a zdola omezen.

Vyhodnocovací funkce nachází uplatnění v těchto oblastech:

- AC regulace (Adaptive Control - adaptivní řízení)
- Řízení výkonu laseru
- Přepínání pozic

Syntaxe

```
SYNFCT(č. polynomu, proměnná hlavního zpracování-výstup, proměnná
hlavního zpracování-vstup)
```

Význam

Jako výstupní proměnné mohou být zvoleny proměnné, které

- s aditivním ovlivňováním
- s multiplikativním ovlivňováním
- jako offset polohy
- přímo

vstupují do operace opracování.

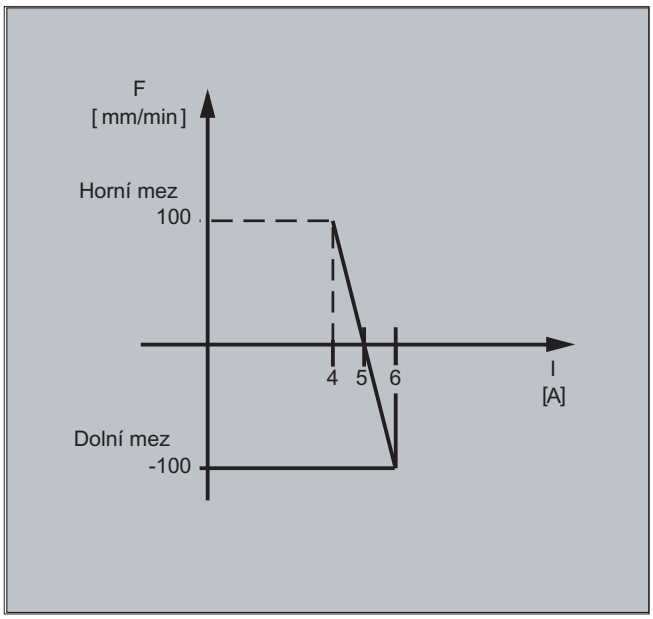
`DO SYNFCT`	Aktivování vyhodnocovací funkce
`č. polynomu`	Polynom definovaný příkazem FCTDEF (viz podkapitola "Definice polynomu")
`proměnná hlavního zpracování-výstup`	Proměnná hlavního zpracování pro zápis
`proměnná hlavního zpracování-vstup`	Proměnná hlavního zpracování pro čtení

Příklad: AC regulace (aditivní)

Aditivní ovlivňování naprogramovaného posuvu

Naprogramovaný posuv má být regulován aditivně prostřednictvím proudu osy X (přísuvná osa):

Posuv se má měnit v rozsahu +/- 100 mm/min, přičemž proud +/- 1 A kolísá okolo pracovního bodu 5 A.



1. definice polynomu

Stanovení koeficientů

$y = f(x) = a_0 + a_1x + a_2x^2 + a_3x^3$

a_1 = -100 mm / 1 min A

a_0 = -(-100)*5 = 500

$a_2 = a_3 = 0$ (žádný kvadratický a kubický člen)

Horní mez = 100

Spodní mez = -100

Z toho vyplývá:

```
FCTDEF(1,-100,100,500,-100,0,0)
```

2. Aktivování AC regulace

```
ID=1 DO SYNFCT(1,$AC_VC,$AA_LOAD[x])
```

;Prostřednictvím parametru `$AA_LOAD[x]` se načítá aktuální zatížení
;osy (% max. proudu pohonu), přičemž se započítává výše definovaný polynom korekce posuvu po dráze.

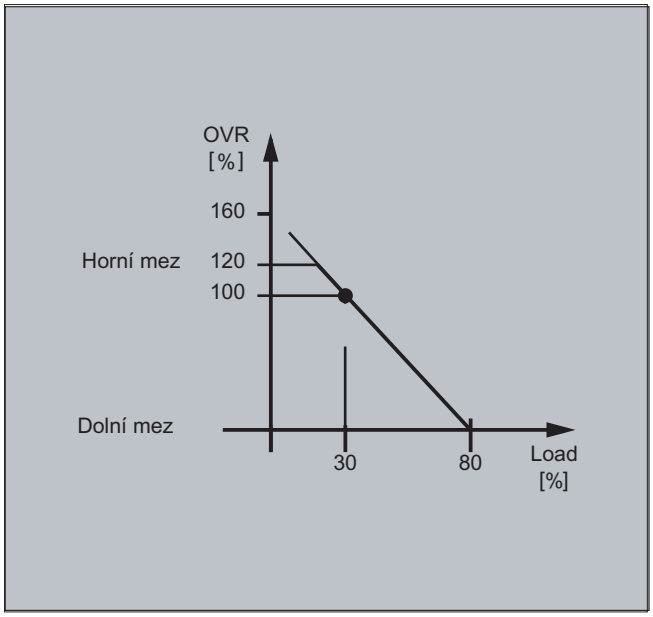
Příklad: AC regulace (multiplikativní)

Multiplikativní ovlivňování naprogramovaného posuvu

Naprogramovaný posuv má být multiplikativně ovlivňován, přičemž posuv - v závislosti na zatížení pohonu - nemá překračovat určité mezní hodnoty:

- Při zatížení pohonu 80% se má posuv zastavit: Override = 0.
- Při zatížení pohonu 30% se smí najíždět s naprogramovaným posuvem. Override = 100%.

Rychlost posuvu smí být překročena o 20%.
Max. Override = 120%.



1. definice polynomu

Stanovení koeficientů

$y = f(x) = a_0 + a_1x + a_2x^2 + a_3x^3$

$a_1 = -100\%/(80-30)\% = -2$

$a_0 = 100 + (2*30) = 160$

$a_2 = a_3 = 0$ (žádný kvadratický a kubický člen)

Horní mez = 120

Spodní mez = 0

Z toho vyplývá:

```
FCTDEF(2,0,120,160,-2,0,0)
```

2. Aktivování AC regulace

```
ID=1 DO SYNFCT(2,$AC_OVR,$AA_LOAD[x])
```

;Prostřednictvím parametru `$AA_LOAD[x]` se načítá aktuální zatížení
;osy (% max. proudu pohonu), přičemž se započítává výše definovaný polynom korekce posuvu.

10.4.8 Regulace vzdálenosti s omezenou korekcí ($AA_OFF_MODE)

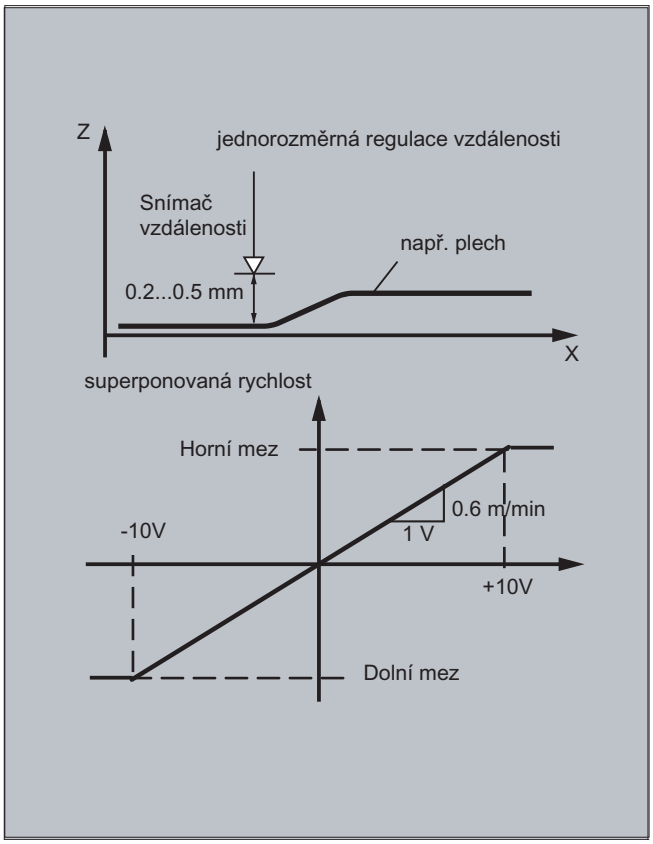
Poznámka

U systému SINUMERIK 828D není tato funkce k dispozici!

Funkce

Integrovaný výpočet hodnoty vzdálenosti probíhá s kontrolou hraniční oblasti:

$AA_OFF_MODE = 1



UPOZORNĚNÍ
Zesílení superponované regulační smyčky je závislé na nastavení taktu IPO. Pomoc: Načtení a započítání MD pro takt IPO.

Poznámka

Omezení rychlosti superponovaného interpolátoru prostřednictvím MD32020 JOG_VELO v případě taktu IPO 12 ms.

Vzorec pro rychlost:

$$\frac{0.120\text{mm}}{0.6\text{ms}} / \text{mV} = 0.6 \frac{\text{m}}{\text{min}} / \text{V}$$

Příklad

Podprogram "AON": Regulace vzdálenosti zapnuta

Programový kód	Komentář
PROC AON	
$AA_OFF_LIMIT[Z]=1	; Definice mezní hodnoty
FCTDEF(1, -10, +10, 0, 0.6, 0.12)	; Definice polynomu.
ID=1 DO SYNFCT(1,$AA_OFF[Z],$A_INA[3])	; Regulace vzdálenosti je aktivní
ID=2 WHENEVER $AA_OFF_LIMIT[Z]<>0 DO $AA_OVR[X] = 0	; Při překročení mezní oblasti zablokovat osu X.
RET	
ENDPROC	

Podprogram "AOFF": Vypnutí regulace vzdálenosti

Programový kód	Komentář
PROC AOFF	
CANCEL(1)	; Vymazání synchronní akce pro regulaci vzdálenosti
CANCEL(2)	; Vymazání kontroly mezní oblasti
RET	
ENDPROC	

Hlavní program "MAIN"

Programový kód	Komentář
AON	; Regulace vzdálenosti zapnuta
...	
G1 X100 F1000	
AOFF	; Vypnutí regulace vzdálenosti
M30	

Další informace

Offset polohy v základním souřadném systému

Pomocí systémové proměnné $AA_OFF[osa] může být pro každou osu v kanálu superponován další pohyb. Ten se uplatňuje jako offset polohy v základním souřadném systému.

Takto naprogramovaný offset polohy se v odpovídající ose superponuje okamžitě, nezávisle na tom, zda je nebo není pro osu naprogramován pohyb.

Omezení výstupní proměnné hlavního zpracování

Hodnotu, na kterou se má uplatňovat absolutní korekce (výstupní proměnná hlavního zpracování) je možné omezit na hodnotu, která je uložena v nastavovaném parametru SD43350 $SA_AA_OFF_LIMIT.

Pomocí strojního parametru MD36750 $MA_AA_OFF_MODE je stanoven druh superpozice vzdálenosti:

Hodnota	Význam
0	Proporcionální vyhodnocování
1	Integrační vyhodnocování

Pomocí systémové proměnné $AA_OFF_LIMIT[osa] je možné v závislosti na směru zjistit, zda se hodnota korekce nachází v hraniční oblasti. Tato systémová proměnná může být kontrolována ze synchronních akcí a při dosažení mezní hodnoty může být např. osa zastavena nebo aktivován alarm.

0: Hodnota korekce mimo hraniční oblast

1 Dosaženo mezní hodnoty korekce v kladném směru

-1: Dosaženo mezní hodnoty korekce v záporném směru

10.4.9 On-line korekce nástroje (FTOC)

Funkce

Příkaz `FTOC` umožňuje superponovaný pohyb geometrické osy podle polynomu naprogramovaného pomocí příkazu `FCTDEF` v závislosti na vztažné hodnotě, která může být např. skutečnou hodnotou osy.

Koeficient a_0 definice funkce v příkazu `FCTDEF(...)` se ve funkci `FTOC` vyhodnocuje. Horní a dolní mezní hodnoty jsou závislé na hodnotě a_0.

Pomocí funkce `FTOC` mohou být jako synchronní akce naprogramovány modální on-line korekce nástroje nebo regulace vzdálenosti.

Tato funkce nachází své uplatnění při opracovávání obrobků a při orovnávání brusného kotouče ve stejném kanálu nebo v různých kanálech (kanál pro obrábění a kanál pro orovnávání).

U funkce `FTOC` platí analogické okrajové podmínky a definice pro orovnávání brusného kotouče jako v případě on-line korekce nástroje s funkcí `PUTFTOCF` (viz "On-line korekce nástroje (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) [Strana 414]").

Syntaxe

```
FCTDEF(<funkce>,<LLimit>,<ULimit>,<a0>,<a1>,<a2>,<a3>)
FTOC(<funkce>,<vztažná hodnota>,<parametr nástroje>,<kanál>,<vřeteno>)
...
```

Význam

`FCTDEF`: Pomocí příkazu `FCTDEF` je definována polynomická funkce pro příkaz `FTOC`.

Parametry:

`<funkce>`: Číslo polynomické funkce
Typ: INT
Rozsah hodnot: 1 ... 3

`<LLimit>`: Dolní mezní hodnota
Typ: REAL

`<ULimit>`: Horní mezní hodnota
Typ: REAL

`<a0> ... <a3>`: Koeficienty polynomické funkce
Typ: REAL

`DO FTOC`: Vyvolání funkce "Kontinuální modální zápis on-line korekce nástroje"

Parametry:

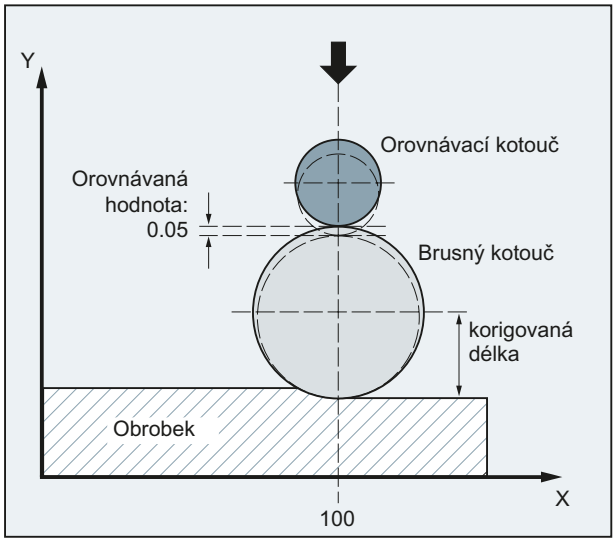
`<funkce>`:	Číslo polynomické funkce Typ: INT Rozsah hodnot: 1 ... 3 **Upozornění:** Musí se shodovat se zadáním v příkazu `FCTDEF`.
`<vztažná hodnota>`:	Proměnná hlavního zpracování, ke které se má započítávat hodnota polynomické funkce definovaná pomocí funkce `FCTDEF`. Typ: VAR REAL
`<parametr nástroje>`:	Číslo parametru opotřebení (Délka 1, 2 nebo 3), ke kterému se má hodnota korekce přičítat. Typ: INT
`<kanál>`:	Číslo kanálu, v němž má být on-line korekce nástroje v platnosti. Typ: INT **Upozornění:** Zadání tohoto parametru je zapotřebí jen tehdy, pokud se hodnota korekce nemá uplatňovat v aktivním kanálu.
`<vřeteno>`:	Číslo vřetena, pro které má být on-line korekce nástroje v platnosti. Typ: INT **Upozornění:** Zadání tohoto parametru je zapotřebí jen tehdy, pokud má být korekce uplatňována na neaktivní brusný kotouč a nikoli na nástroj, který je momentálně používán.

Poznámka

V cílovém kanálu musí být aktivována funkce `FTOCON`.

Příklad

Aktivování délky, o kterou má být korigován brusný kotouč nacházející se v záběru.



Programový kód	**Komentář**
`FCTDEF(1,-1000,1000,-$AA_IW[V],1)`	`; Definice funkce.`
`ID=1 DO FTOC(1,$AA_IW[V],3,1)`	`; Aktivování on-line korekce nástroje: Skutečná hodnota osy V je vstupní hodnotou pro polynom 1. Výsledek se přičítá v kanálu 1 jako hodnota korekce k parametru Délka 3 aktivního brusného kotouče.`
`WAITM(1,1,2)`	`; Synchronizace s kanálem, kde probíhá obrábění.`
`G1 V-0.05 F0.01 G91`	`; Přísuvný pohyb pro orovnávání.`
`G1 V-0.05 F0.02`	
`...`	
`CANCEL(1)`	`; Deaktivování on-line korekce.`
`...`	

10.4.10 On-line korekce délky nástroje ($AA_TOFF)

Funkce

Prostřednictvím systémové proměnné $AA_TOFF[ ] mohou být trojrozměrně a v reálném čase korigovány efektivní délky nástroje ve všech třech jeho rozměrech.

Jako index se používají tři identifikátory geometrických os. Tímto způsobem je definován počet aktivních směrů korekce prostřednictvím geometrických os, které jsou v daném okamžiku aktivní.

Všechny korekce mohou být aktivní současně.

Syntaxe

```
N... TRAORI
N... TOFFON(X,<hodnota offsetu>)
N... WHEN TRUE DO $AA_TOFF[X]
N... TOFFON(Y,<hodnota offsetu>)
N... WHEN TRUE DO $AA_TOFF[Y]
N... TOFFON(Z,<hodnota offsetu>)
N... WHEN TRUE DO $AA_TOFF[Z]
```

Význam

TOFFON:	**Aktivování** on-line korekce délky nástroje	
	X, Y, Z:	Směr nástroje, ve kterém se má on-line korekce délky uplatňovat.
	<hodnota offsetu>:	Při aktivování může být pro odpovídající směr korekce zadána hodnota offsetu, přičemž na tento offset se bude ihned najíždět.
TOFFOF:	**Vynulování** on-line korekce délky nástroje Hodnoty korekce v zadaném směru budou vynulovány a aktivuje se přitom zastavení předběžného zpracování.	
$AA_TOFF[X]=<hodnota>:	Superpozice ve směru X	
$AA_TOFF[Y]=<hodnota>:	Superpozice ve směru Y	
$AA_TOFF[Z]=<hodnota>:	Superpozice ve směru Z	

Příklady

Příklad 1: Aktivování korekce délky nástroje

Programový kód	Komentář
N10 TRAORI(1)	; Aktivování transformace.
N20 TOFFON(Z)	; Aktivování On-line korekce délky nástroje pro směr Z.
N30 WHEN TRUE DO $AA_TOFF[Z]=10 G4 F5	; Pro rozměr nástroje ve směru osy Z se interpoluje korekce délky nástroje 10.
N40 TOFFON(X)	; Aktivování On-line korekce délky nástroje pro směr X.
N50 ID=1 DO $AA_TOFF[X]=$AA_IW[X2] G4 F5	; Pro směr nástroje X se uskuteční korekce v závislosti na poloze osy X2.
...	
	; Přiřazení aktuální korekce ve směru osy X. Pro rozměr nástroje ve směru osy X se najede zpátky na hodnotu korekce délky nástroje 0:
N100 XOFFSET=$AA_TOFF_VAL[X] N120 TOFFON(X,-XOFFSET) G4 F5	

Příklad 2: Deaktivování korekce délky nástroje

Programový kód	Komentář
N10 TRAORI(1)	; Aktivování transformace.
N20 TOFFON(X)	; Aktivování On-line korekce délky nástroje pro směr X.
N30 WHEN TRUE DO $AA_TOFF[X]=10 G4 F5	; Pro rozměr nástroje ve směru osy X se interpoluje korekce délky nástroje 10.
...	
N80 TOFFOF(X)	; Offset polohy ve směru X nástroje se vymaže. ...$AA_TOFF[X]=0 Žádná osa se nebude pohybovat. K aktuální poloze ve WCS se připočítá offset polohy v souladu s momentální orientací.

10.4.11 Polohovací pohyby

Funkce

Polohování os ze synchronních akcí může probíhat zcela asynchronně vzhledem k výrobnímu programu. Programování polohovacích os ze synchronních akcí se doporučuje pro cyklické operace nebo pro činnosti, které jsou silně řízeny na základě nějakých událostí. Osy, které jsou programovány ze synchronních akcí, se nazývají **příkazové osy**.

Programování

Literatura:
/PG/, Příručka programování, Základy; kapitola "Zadávání dráhy"
/FBSY/ Příručka Popis funkcí, Synchronní akce; "Spouštění příkazových os"

Parametry

Systém měřicích jednotek pro zadávání poloh v synchronních akcích je definován pomocí G-kódů `G70`/`G71`/`G700`/`G710`.

Naprogramováním G-funkcí v synchronní akci může být vyhodnocování měřicích jednotek (palce/mm) pro synchronní akci nastaveno nezávisle na kontextu výrobního programu.

10.4.12 Polohování osy (POS)

Funkce

Pohyb pro nastavení osy do určité polohy tohoto typu nemá oproti programování z výrobního programu žádný vliv na zpracovávání výrobního programu.

Syntaxe

```
POS[osa] = hodnota
```

Význam

`DO POS`	Spouštění/nastavování polohy příkazové osy
`Osa`	Název osy, která se má pohybovat
`Hodnota`	Zadání hodnoty, na kterou se má najet (v závislosti na režimu pohybu)

Příklady

Příklad 1:

Programový kód	**Komentář**
`ID=1 EVERY $AA_IM[B]>75 DO POS[U]=100`	`; Najetí osou U v závislosti na režimu pohybu buď inkrementálně o 100 (palce/mm) nebo na pozici 100 (palce/mm) od počátku řídícího systému.`
	`; Pohyb osy U o dráhu vypočítanou podle proměnné hlavního zpracování:`
`ID=1 EVERY $AA_IM[B]>75 DO POS[U]=$AA_MW[V]-$AA_IM[W]+13.5`	

Příklad 2:

Okolní programové příkazy ovlivňují polohovací dráhu polohovací osy (žádná G-funkce v akční části synchronní akce)

Programový kód	Komentář
N100 R1=0	
N110 G0 X0 Z0	
N120 WAITP(X)	
N130 ID=1 WHENEVER $R==1 DO POS[X]=10	
N140 R1=1	
N150 G71 Z10 F10	; Z=10mm X=10mm
N160 G70 Z10 F10	; Z=254mm X=254mm
N170 G71 Z10 F10	; Z=10mm X=10mm
N180 M30	

Příkaz `G71` v akční části synchronní akce jednoznačně určuje polohovací dráhu polohovací osy (metrické jednotky) nezávisle na okolních programových příkazech:

Programový kód	Komentář
N100 R1=0	
N110 G0 X0 Z0	
N120 WAITP(X)	
N130 ID=1 WHENEVER $R==1 DO G71 POS[X]=10	
N140 R1=1	
N150 G71 Z10 F10	; Z=10mm X=10mm
N160 G70 Z10 F10	; Z=254mm X=10mm (X je polohována vždy na hodnotu 10mm)
N170 G71 Z10 F10	; Z=10mm X=10mm
N180 M30	

Jestliže se pohyb osy nemá spouštět na začátku bloku, může být korekce (override) pro tuto osu podržena pomocí synchronní akce na 0 až do požadovaného okamžiku spuštění:

Programový kód	Komentář
WHENEVER $A_IN[1]==0 DO $AA_OVR[W]=0 G01 X10 Y25 F750 POS[W]=1500 FA=1000	
	; Polohovací osa bude pozastavena tak dlouho, dokud digitální vstup 1=0.

10.4.13 Pozice v předdefinované referenční oblasti (POSRANGE)

Funkce

Pomocí funkce POSRANGE ( ) je možno zjistit, zda se požadovaná poloha osy, jejíž interpolace právě probíhá, nachází v okně okolo předem zadané referenční pozice. Údaje polohy mohou být vztaženy na souřadné systémy, jež mohou být určeny předem.

Při zjišťování skutečné polohy osy typu Modulo se automaticky provádí korekce pomocí funkce Modulo.

Poznámka

Funkce může být vyvolávána pouze ze synchronní akce. V případě vyvolání z výrobního programu se aktivuje alarm 14091 %1 blok %2 Funkce je nepřípustná, index: %3 s indexem 5.

Syntaxe

```
BOOL POSRANGE(osa, Refpos, Winlimit,[Coord])
```

Význam

`BOOL POSRANGE`	Aktuální poloha příkazové osy v okně okolo předem zadané referenční pozice.
`AXIS <osa>`	Identifikátor strojní, kanálové nebo geometrické osy
`REAL Refpos`	Referenční pozice v souřadném systému Coord
`REAL Winlimit`	Údaj, který udává mezní hodnotu pro polohovací okno
`INT Coord`	Standardně je aktivní MCS. K dispozici jsou následující možnosti: 0: MCS (souřadný systém stroje) 1: BCS (základní souřadný systém) 2: ENS (nastavitelný souřadný systém) 3: WCS (souřadný systém obrobku)

Hodnota funkce

Aktuální požadovaná poloha v závislosti na údaji pozice v zadaném souřadném systému

`Hodnota funkce: TRUE`	pokud Refpos(Coord) - abs(Winlimit) ≤ Actpos(Coord) ≤ Refpos(Coord) + abs(Winlimit)
`Hodnota funkce: FALSE`	v ostatních případech

10.4.14 Zastavení/spuštění osy (MOV)

Funkce

Pomocí příkazu MOV[osa]=hodnota může být příkazová osa spuštěna bez zadání koncové pozice. Příslušná osa se pohybuje v naprogramovaném směru, dokud není zadán nový pohybový nebo polohovací příkaz pro nějaký jiný pohyb nebo dokud není osa příkazem Stop zastavena.

Syntaxe

```
MOV[osa] = hodnota
```

Význam

`DO MOV`	Spuštění pohybu příkazové osy
`Osa`	Název osy, která se má spustit
`Hodnota`	Příkaz pro spuštění/zastavení pohybu Znaménko určuje směr pohybu. Datový typ hodnoty je INTEGER.
`Hodnota > 0` `(obvykle +1)`	kladný směr
`Hodnota < 0` `(obvykle -1)`	záporný směr
`Hodnota == 0`	Zastavení pohybu osy

Poznámka

Jestliže je osa s diskrétními polohami zastavena pomocí příkazu `MOV[osa] = 0`, bude osa zastavena na následující diskrétní pozici.

Příklad

Programový kód	**Komentář**
`... DO MOV[U]=0`	`; Osa U se zastaví.`

10.4.15 Výměna osy (RELEASE, GET)

Funkce

Za účelem výměny nástroje mohou být příslušné příkazové osy vyžádány prostřednictvím akce s příkazem GET(osa) v rámci synchronní akce. Typ osy přiřazený tomuto kanálu a tím pádem také v tomto okamžiku platná interpolační oprávnění mohou být zjištěny pomoc systémové proměnné $AA_AXCHANGE_TYP. V závislosti na vlastním stavu a stavu kanálu, který vlastní aktuální interpolační oprávnění k této ose, jsou možné různé způsoby, jak může tato operace proběhnout.

Je-li výměna nástroje ukončena, potom může být tato příkazová osa pomocí akce s příkazem RELEASE(osa) v rámci synchronní akce pro daný kanál uvolněna.

Výrobce stroje

Příslušná osa musí být pomocí strojních parametrů přiřazena danému kanálu. Věnujte prosím pozornost informacím od výrobce stroje

Syntaxe

`GET(osa[,osa{,...}])` Vyžádání osy

`RELAESE(osa[,osa{,...}])` Uvolnění osy

Význam

`DO RELEASE`	Uvolnění osy, aby se stala neutrální
`DO GET`	Převzetí osy pro její výměnu
`Osa`	Název osy, která se má spustit

Příklad - Posloupnost programových příkazů pro výměnu osy mezi dvěma kanály

Osa Z je známá v kanálu 1 a v kanálu 2.

Posloupnost programových příkazů v 1. kanálu:

```
Programový kód                                  Komentář
WHEN TRUE DO RELEASE(Z)                         ; Osa Z je nyní neutrální osou
WHENEVER($AA_TYP[Z]==1) DO RDISABLE             ; Blokování načítání, dokud osa Z
                                                  není programovou osou
N110 G4 F0.1
WHEN TRUE DO GET(Z)                             ; Osa Z je znovu osou v NC programu
WHENEVER($AA_TYP[Z]<>1) DO RDISABLE             ; Blokování načítání, dokud je osa
                                                  Z programovou osou
N120 G4 F0.1
WHEN TRUE DO RELEASE(Z)                         ; Osa Z je nyní neutrální osou
WHENEVER($AA_TYP[Z]==1) DO RDISABLE             ; Blokování načítání, dokud osa Z
                                                  není programovou osou
N130 G4 F0.1                                    ;
N140 START(2)                                   ; Spustit 2. kanál
```

Posloupnost programových příkazů v 2. kanálu:

Programový kód	Komentář
WHEN TRUE DO GET(Z)	; Převzetí osy Z ve 2. kanálu
WHENEVER($AA_TYP[Z]==0) DO RDISABLE	; Blokování načítání, dokud je osa Z v jiném kanálu
N210 G4 F0.1	
WHEN TRUE DO GET(Z)	; Osa Z je osou v NC programu
WHENEVER($AA_TYP[Z]<>1) DO RDISABLE	; Blokování načítání, dokud je osa Z programovou osou
N220 G4 F0.1	
WHEN TRUE DO RELEASE(Z)	; Osa Z ve 2. kanálu je neutrální osou
WHENEVER($AA_TYP[Z]==1) DO RDISABLE	; Blokování načítání, dokud osa Z není programovou osou
N230 G4 F0.1	
N250 WAITM(10, 1, 2)	; Synchronizace s kanálem 1

Další posloupnost programových příkazů v 1. kanálu:

Programový kód	Komentář
N150 WAIM(10, 1, 2)	; synchronizace s kanálem 2
WHEN TRUE DO GET(Z)	; Převzetí osy Z v tomto kanálu
WHENEVER($AA_TYP[Z]==0) DO RDISABLE	; Blokování načítání, dokud je osa Z v jiném kanálu
N160 G4 F0.1	
N199 WAITE(2)	
N999 M30	; Čekání na konec programu v kanálu 2

Příklad výměny osy v technologickém cyklu

Osa U ($MA_AUTO_GET_TYPE=2) je v 1. kanálu a ve 2. kanálu známá a momentálně jsou interpolační oprávnění přiřazena kanálu 1. V kanálu 2 se spouští následující technologický cyklus:

Programový kód	Komentář
GET(U)	; Převzetí osy U v kanálu
POS[U]=100	; Osa U má najet na pozici 100

Řádek pro pohyb příkazové osy POS[U] se uskuteční teprve tehdy, když byla osa U v kanálu 2 převzata.

Postup

Typ osy, která byla pro účely výměny osy vyžádána v okamžiku aktivování akce `GET(osa)`, může být načten pomocí systémové proměnné ($AA_AXCHANGE_TYP[<osa>].

- 0: Osa je přiřazena NC programu
- 1: Osa je přiřazena PLC nebo je aktivní jako příkazová nebo oscilační osa
- 2: Interpolační oprávnění jsou přiřazena jinému kanálu
- 3: Osa je neutrální osou
- 4: Neutrální osa je ovládána pomocí PLC
- 5: Interpolační oprávnění jsou přiřazena jinému kanálu, osa je vyžádána pro NC program
- 6: Interpolační oprávnění jsou přiřazena jinému kanálu, osa je vyžádána jako neutrální osa
- 7: Osa je aktivní jako osa PLC nebo jako příkazová nebo oscilační osa, osa je vyžádána pro NC program
- 8: Osa je aktivní jako osa PLC nebo jako příkazová nebo oscilační osa, osa je vyžádána jako neutrální osa

Okrajové podmínky

Příslušná osa musí být pomocí strojních parametrů přiřazena danému kanálu.

Osa, která je ovládána výlučně prostřednictvím PLC, nesmí být přiřazena NC programu.

Literatura:
/FB2/, Příručka Popis funkcí, Rozšiřovací funkce; Polohovací osy (P2)

Vyžádání osy z jiného kanálu pomocí akce GET

Jestliže má v okamžiku aktivování akce `GET` pro danou osu přidělena **oprávnění k zápisu jiný kanál** (interpolační oprávnění) ($AA_AXCHANGE_TYP[<osa>] == 2), potom je osa prostřednictvím výměny os z tohoto kanálu vyžádána ($AA_AXCHANGE_TYP[<osa>]==6), a jakmile je to možné, je přiřazena kanálu, který ji vyžádal.

Osa se přitom dostane do stavu neutrální osy ($AA_AXCHANGE_TYP[<osa>]==3).

V kanálu, který si osu vyžádal, se žádná reorganizace neuskutečňuje.

Přiřazení **jako osa NC programu s reorganizací:**

Jestliže osa byla už v okamžiku aktivování akce `GET` vyžádána jako neutrální osa ($AA_AXCHANGE_TYP[<osa>]==6), potom je osa vyžádána pro NC program ($AA_AXCHANGE_TYP[<osa>]==5) a jakmile je to možné, je přiřazena NC programu v kanálu ($AA_AXCHANGE_TYP[<osa>]==0).

Osa je již přiřazena kanálu, který ji vyžádal

Přiřazení **jako osa NC programu s reorganizací:**

Jestliže je požadovaná osa v okamžiku aktivování už přiřazena kanálu, který o ni žádal, a pokud se nachází ve stavu neutrální osy - nesmí být ovládána z PLC - ($AA_AXCHANGE_TYP[<osa>]==3), potom bude přiřazena NC programu ($AA_AXCHANGE_TYP[<osa>]==0).

Osa ve stavu neutrální osy je řízena prostřednictvím PLC

Pokud se osa nachází ve stavu neutrální osy a pokud je řízena prostřednictvím PLC ($AA_AXCHANGE_TYP[<osa>]==4), potom je osa vyžádána jako neutrální osa ($AA_AXCHANGE_TYP[<osa>] == 8), ale přitom se osa v závislosti na bitu 0 ve strojním parametru MD 10722: AXCHANGE_MASK pro automatickou výměnu osy mezi kanály zablokuje (bit 0 == 0). To odpovídá nastavení ($AA_AXCHANGE_STAT[<osa>] == 1).

Osa je aktivní jako neutrální příkazová osa, příp. jako oscilační osa, nebo je přiřazena PLC

Pokud je osa aktivní jako příkazová osa, příp. jako oscilační osa nebo pokud jsou její pohyby ovládány z PLC, osa PLC == konkurenční polohovací osa ($AA_AXCHANGE_TYP[<osa>]==1), potom je osa vyžádána jako neutrální osa ($AA_AXCHANGE_TYP[<osa>] == 8), ale přitom se osa v závislosti na bitu 0 ve strojním parametru MD 10722: AXCHANGE_MASK pro automatickou výměnu osy mezi kanály zablokuje (bit 0 == 0). To odpovídá nastavení ($AA_AXCHANGE_STAT[<osa>] == 1).

Nová akce s příkazem GET potom osu vyžádá pro NC program ($AA_AXCHANGE_TYP[<osa>] == 7).

Osa je již přiřazena NC programu

Jestliže je již osa přiřazena NC programu daného kanálu ($AA_AXCHANGE_TYP[<osa>]==0) nebo pokud je již takové přiřazení vyžádáno, např. z NC programu se spustila výměna osy ($AA_AXCHANGE_TYP[<osa>]==5, příp. $AA_AXCHANGE_TYP[<osa>] == 7), k žádné změně stavu nedochází.

10.4.16 Posuv osy (FA)

Funkce

Posuv pro příkazové osy má modální platnost.

Syntaxe

```
FA[<osa>]=<hodnota>
```

Příklad

Programový kód	Komentář
`ID=1 EVERY $AA_IM[B]>75 DO POS[U]=100 FA[U]=990`	`; Pevně zadaná hodnota posuvu.`
	`; Nastavení hodnoty posuvu podle proměnné hlavního zpracování:`
`ID=1 EVERY $AA_IM[B]>75 DO POS[U]=100 FA[U]=$AA_VACTM[W]+100`	

10.4.17 Softwarový koncový spínač

Funkce

Ohraničení pracovního pole naprogramovaná pomocí příkazů G25/G26 se pro příkazové osy zohledňují v závislosti na nastavení parametru $SA_WORKAREA_PLUS_ENABLE.

Aktivování a deaktivování ohraničení pracovního pole prostřednictvím G-funkcí WALIMON/ WALIMOF ve výrobním programu pro příkazové osy neplatí.

10.4.18 Koordinace os

Funkce

Za obvyklých okolností jsou pohyby osy ovládány buď z výrobního programu nebo je osa ovládána jako polohovací osa ze synchronní akce.

Jestliže se však má tatáž osa pohybovat střídavě jako dráhová osa z výrobního programu nebo jako polohovací osa ze synchronní akce, je potřeba koordinované předávání mezi oběma typy pohybů osy.

Jestliže mají být následně pohyby příkazové osy ovládány z výrobního programu, je nutné reorganizovat přípravu zpracování. Tím je opět podmiňováno přerušení zpracovávání výrobního programu, které je srovnatelné se zastavením předběžného zpracování.

Příklad - Ovládání pohybů osy X volitelně z výrobního programu nebo ze synchronní akce

Programový kód	Komentář
N10 G01 **X100** Y200 F1000	; Osa X je naprogramována ve výrobním programu.
…	
N20 ID=1 WHEN $A_IN[1]==1 DO **POS[X]=150** FA[X]=200	; Spuštění polohování ze synchronní akce, jakmile ; se aktivuje digitální vstup
…	
CANCEL(1)	; Deaktivování synchronní akce
…	
N100 G01 **X240** Y200 F1000	; Osa X je dráhovou osou, před spuštěním pohybu se vyskytuje čekací doba v důsledku předávání osy, jestliže byl aktivován digitální vstup 1 a poloha osy X byla nastavována ze synchronní akce.

Příklad - Změna příkazu pohybu pro tutéž osu

Programový kód	Komentář
ID=1 EVERY $A_IN[1]>=1 DO POS[V]=100 FA[V]=560	; Spuštění polohování ze synchronní akce, jakmile je na digitálním vstupu hodnota >= 1
ID=2 EVERY $A_IN[2]>=1 DO POS[V]=$AA_IM[V] FA[V]=790	; Osa se pohybuje, 2. vstup je nastaven, tzn. koncová poloha a posuv pro osu V se v případě dvou současně aktivních synchronních akcí plynule dosadí, zatímco pohyb probíhá.

10.4.19 Dosazení skutečné hodnoty (PRESETON)

Funkce

Při zpracování příkazu PRESTON (osa, hodnota) se aktuální poloha osy nemění, je jí jen přiřazena nová hodnota.

Příkaz PRESETON je možné ze synchronních akcí používat pro tyto osy:

- Kruhové osy typu Modulo, které se spouštěly z výrobního programu
- Všechny příkazové osy, které se spouštěly ze synchronní akce

Syntaxe

```
PRESETON(osa, hodnota)
```

Význam

`DO PRESETON`	Dosazení skutečné hodnoty v synchronních akcích
`Osa`	Osa, jejíž počátek řídícího systému má být změněn
`Hodnota`	Hodnota, o kterou má být počátek řídícího systému změněn.

Omezení pro osy

Příkaz `PRESETON` není možné použít pro osy, které se podílejí na transformaci.

Jednou a toutéž osou se může pohybovat pouze s časovým posunem z výrobního programu nebo ze synchronní akce, proto se mohou při programování osy z výrobního programu vyskytnout čekací doby, pokud tato osa byla již předtím naprogramována v synchronní akci.

Jestliže je stejná osa používána střídavě, potom se mezi oběma pohyby osy uskutečňuje koordinované předávání. Zpracování výrobního programu musí být za tímto účelem přerušeno.

Příklad

Posunutí počátku souřadného systému osy

Programový kód	**Komentář**
`WHEN $AA_IM[a] >= 89.5 DO PRESETON(a4,10.5)`	`; Počátek řídícího systému osy a se posune o 10,5 délkových jednotek (palce, příp. mm) v kladném směru osy`

10.4.20 Zrušení uvolnění pro otáčení osového zásobníku (AXCTSWEC)

Funkce

Pomocí příkazu `AXCTSWEC` může být již aktivované uvolnění pro otáčení osového zásobníku opět odvoláno. Příkaz spouští zastavení předběžného zpracování s reorganizací (`STOPRE`).

Platnost

Aby mohlo být uvolnění pro otáčení osového zásobníku v daném kanálu opět odvoláno, musí být splněny následující podmínky:

- Uvolnění pro otáčení osového zásobníku v daném kanálu musí již být uskutečněno:
  - `AXCTSWE(<osový zásobník>)`
  - `$AC_AXCTSWA[<osový zásobník>] == 1`
- Otáčení osového zásobníku ještě nebylo zahájeno:
  - `$AN_AXCTSWA[<osový zásobník>] == 0`

Jako zpětné hlášení pro dokončené odvolání uvolnění se resetuje specifická kanálová systémová proměnná:

`$AC_AXCTSWA[<osový zásobník>] == 0`

Literatura:
Pokud budete potřebovat informace o systémových proměnných, viz "Osový zásobník (AXCTSWE, AXCTSWED, AXCTSWEC) [Strana 685]".

Syntaxe

```
DO AXCTSWEC(<osový zásobník>)
```

Význam

`DO AXCTSWEC`:	Zrušení uvolnění pro otáčení osového zásobníku pro daný kanál	
`<osový zásobník>`:	Identifikátor osového zásobníku Možné údaje jsou následující:	
	`CT<číslo zásobníku>`:	Ke kombinaci písmen CT je připojeno číslo osového zásobníku. Příklad: `CT3`
	`<název zásobníku>`:	Parametrem MD12750 $MN_AXCT_NAME_TAB nastavený individuální název osového zásobníku. Příklad: `A_CONT3`
	`<název osy>`:	Název osy osového zásobníku, který je v příslušném kanálu známý.

Příklad

Příklad programu:

Programový kód	Komentář
N100 Id=1 DO CTSWEC	; Technologický cyklus, viz níže.
; inicializace	
NEXT:	
N200 G0 X30 Z1	
N210 G95 F.5	
N220 M3 S1000	
N230 G0 X25	
N240 G1 Z-10	
N250 G0 X30	
N260 M5	
N270 AXCTSWE(S1)	; Uvolnění otáčení osového zásobníku prostřednictvím protivřetena A1, které je v kanálu známé.
N280 GOTO NEXT	

Technologický cyklus CTSWEC:

Programový kód	Komentář
CTSWEC PROC(_ex_CT="CT1" STRING _ex_CTsl_BITmask=1H LONG _ex_CT_SL_Number=1 _ex_WAIT_number_of_IPOs=1000) DISPLOF ICYCOF	
DEFINE _ex_CT[3]	; Název zásobníku
DEFINE _ex_CTsl_BITmask	; Slot-Bit
DEFINE _ex_CT_SL_Number LONG =1	
DEFINE _ex_WAIT_number_of_IPOs INTEGER =1000	
DEFINE _ex_number_of_IPOs $AC_MARKER[0]	
N110 IF (($A_STOP_COND[0] OR $A_STOP_COND[1] OR $A_STOP_COND[2] OR $A_STOP_COND[3] OR $A_STOP_COND[4] OR $A_STOP_COND[5] OR $A_STOP_COND[6] OR $A_STOP_COND[7] OR $A_STOP_COND[8] OR $A_STOP_COND[9] OR $A_STOP_COND[10]) is true))	
N120 _ex_number_of_IPOs=_ex_number_of_IPOs+1	; Jestliže existuje nějaký podmínka zastavení delší než 1000 taktů IPO
N130 IF _ex_number_of_IPOs >= _ex_WAIT_number_of_IPOs AND $AN_AXCTSWEC[_ex_CT] == _ex_CTsl_BITmask	; a k uvolnění vlastního slotu ještě nedošlo:
N140 AXCTSWEC	; Odstranit uvolnění.
; … ELSE	; Čekat, zda se podmínka zastavení sama nezruší.
N150 ENDIF	
N160 ELSE	
N170 _ex_number_of_IPOs=0	
N180 ENDIF	
N190 RET	

Okrajové podmínky

Použití osy ze zásobníku před voláním funkce AXCTSWEC

Protože zpracovávání programu není příkazem AXCTSWE pozastaveno, je zapotřebí mít při programování synchronní akce DO AXCTSWEC na paměti následující:

Příklad:

Programový kód	**Komentář**
N10 AXCTSWE(CT3)	; Uvolnění otáčení osového zásobníku.
N20 AX_A10	; AX_A = osa ze zásobníku. ; Čeká se na konec otáčení osového zásobníku: $AN_AXCTSWA[CT3]==0
WHEN <podmínka> DO AXCTSWEC(AX_A)	; Odvolání uvolnění. **Nemá žádný efekt!**
N30 G4 F1	

Protože se po bloku N10 s uvolněním otáčení osového zásobníku používá v bloku N20 osa z tohoto osového zásobníku (AX_A) a protože toto použití má za následek, že se čeká na dokončení otáčení tohoto osového zásobníku, uskuteční se synchronní akce teprve společně s programovým blokem N30 ve zpracování hlavní větve programu a v důsledku toho nemá příkaz žádný efekt.

Pomoc:

Programový kód	**Komentář**
N11 AXCTSWE(CT3)	; Uvolnění otáčení osového zásobníku.
WHEN <podmínka> DO AXCTSWEC(AX_A)	; Odvolání uvolnění.
N21 ...	; Zpracovatelný NC blok.
N31 AX_A10	; Čeká se na konec otáčení osového zásobníku: $AN_AXCTSWA[CT3]==0

UPOZORNĚNÍ

Bez zpracovatelného bloku N21 se synchronní akce uskuteční teprve po skončení otáčení osového zásobníku spolu s následujícím zpracovatelným programovým blokem N31 v hlavní větvi programu a byla by stejně jako ve výše uvedeném příkladu bez jakéhokoli efektu.

10.4.21 Pohyby vřetena

Funkce

Polohování vřeten ze synchronních akcí může probíhat zcela asynchronně vzhledem k výrobnímu programu. Tento druh programování se doporučuje pro cyklické operace nebo pro činnosti, které jsou silně řízeny na základě nějakých událostí.

Jestliže jsou v důsledku současně aktivních synchronních akcí pro jedno vřeteno zadány vzájemně si odporující příkazy, platí ten příkaz vřetena, který byl přijat jako poslední.

Příklad - Spouštění/zastavování/nastavování polohy vřetena

Programový kód	Komentář
`ID=1 EVERY $A_IN[1]==1 DO M3 S1000`	`; Nastavení směru otáčení a otáček`
`ID=2 EVERY $A_IN[2]==1 DO SPOS=270`	`; Polohování vřetena`

Příklad nastavení směru otáčení/otáček, nastavování polohy vřetena

Programový kód	Komentář
`ID=1 EVERY $A_IN[1]==1 DO M3 S300`	`; Nastavení směru otáčení a otáček`
`ID=2 EVERY $A_IN[2]==1 DO M4 S500`	`; Zadání nového směru otáčení a nových otáček`
`ID=3 EVERY $A_IN[3]==1 DO S1000`	`; Zadání nových otáček`
`ID=4 EVERY ($A_IN[4]==1) AND ($A_IN[1]==0) DO SPOS=0`	`; Polohování vřetena`

10.4.22 Vlečení (TRAILON, TRAILOF)

Funkce

Při aktivování vazby ze synchronní akce může být řídící osa v pohybu. Vlečná osa se v tomto případě urychlí na požadovanou rychlost. Poloha řídící osy v okamžiku synchronizace rychlostí je počáteční polohou pro vlečení. Funkce vlečení je popisována v kapitole "Chování při pohybu po dráze".

Syntaxe

Aktivování vlečení

```
DO TRAILON(vlečná osa, řídící osa, faktor vazby)
```

Deaktivování vlečení

```
DO TRAILOF (vlečná osa, řídící osa, řídící osa 2)
```

Význam

Aktivování asynchronního vlečení:

`... DO TRAILON(FA, LA, Kf)`	kde: FA: Vlečná osa LA: Řídící osa Kf: Faktor vazby

Deaktivování asynchronního vlečení:

`... DO TRAILOF(FA, LA, LA2)`	kde: FA: Vlečná osa LA: Řídící osa, volitelné LA2: Řídící osa 2, volitelné
`... DO TRAILOF(FA)`	Všechny vazby s vlečnou osou se zruší.

Příklad

Programový kód	Komentář
`$A_IN[1]==0 DO TRAILON(Y,V,1)`	`;` Aktivování 1. svazku os s vlečením, pokud je digitální vstup 1 nastaven
`$A_IN[2]==0 DO TRAILON(Z,W,-1)`	`;` Aktivování 2. vlečného spojení os
`G0 Z10`	`;` Přísuv osy Z a osy W v opačných směrech os
`G0 Y20`	`;` Přísuv osy Z a osy V ve stejném směru osy
`...`	
`G1 Y22 V25`	`;` Superpozice závislého a nezávislého pohybu vlečné osy "V".
`...`	
`TRAILOF(Y,V)`	`;` Deaktivování 1. vlečného spojení os
`TRAILOF(Z,W)`	`;` Deaktivování 2. vlečného spojení os

Příklad zabránění konfliktu pomocí příkazu TRAILOF

Aby bylo možné osu zapojenou do vazby znovu uvolnit, aby k ní byl přístup jako ke kanálové ose, musí být napřed vyvolána funkce `TRAILOF`. Ještě předtím, než si kanál vyžádá příslušnou osu, musí být zaručeno, že byl příkaz `TRAILOF` zpracován. V následujícím příkladu tomu tak není.

…

```
N50 WHEN TRUE DO TRAILOF(Y,X)
N60 Y100
```

…

V tomto případě není osa v pravý čas uvolněna, protože synchronní akce s příkazem `TRAOLOF` s blokovou platností je aktivní synchronně s blokem `N60`, viz kapitola Pohybové synchronní akce, "Struktura, všeobecné základy".
Aby se konfliktním situacím zabránilo, mělo by se

postupovat následujícím způsobem:

…

```
N50 WHEN TRUE DO TRAILOF(Y,X)
N55 WAITP(Y)
N60 Y100
```

10.4.23 Vazba řídící hodnotou (LEADON, LEADOF)

Poznámka

U systému SINUMERIK 828D není tato funkce k dispozici!

Funkce

Osová vazba řídící hodnotou může být programována v synchronních akcích bez jakýchkoli omezení. Změna tabulky křivky u již existující vazby bez předcházející nové synchronizace je možná jen v synchronních akcích.

Syntaxe

Aktivování vazby řídící hodnotou

```
DO LEADON(vlečná osa, řídící osa, č. tab. křivky, OVW)
```

Deaktivování vazby řídící hodnotou

```
DO LEADOF (vlečná osa, řídící osa, řídící osa 2)
```

Význam

```
Aktivování vazby os řídící hodnotou:
```

`...DO LEADON(FA, LA, NR, OVW)`	kde: FA: Vlečná osa LA: Řídící osa NR: Číslo uložené tabulky křivky OVW: Přepsání již existující vazby pomocí změněné tabulky křivky povoleno

```
Deaktivování vazby os řídící hodnotou:
```

`...DO LEADOF(FA, LA)`	kde: FA: Vlečná osa LA: Řídící osa, volitelné
`... DO LEADOF(FA)`	zkrácená forma bez uvedení řídící osy

Odblokování přístupu pomocí synchronní akce, RELEASE

Aby bylo možné osu zapojenou do vazby prostřednictvím synchronní akce uvolnit, musí být napřed pro vlečnou osu zapojenou do vazby vyvolána funkce RELEASE.

Příklad:

```
RELEASE(XKAN)
ID=1 every SR1==1 to LEADON(CACH,XKAN,1)
```

OVW=0 (předdefinované nastavení)

Bez nové synchronizace nemůže být pro již existující vazbu zadána žádná nová tabulka křivky. Změna tabulky křivky vyžaduje, aby se napřed již existující vazba deaktivovala a aby se znovu zapnula se změněným číslem tabulky křivky. Tím se zajistí nová synchronizace vazby.

Změna tabulky křivky u již existující vazby s nastavením OVW=1

Když je nastaveno `OVW=1`, může být pro již existující vazbu zadána nová tabulka křivky. Neprovádí se žádná nová synchronizace. Vlečná osa se pokusí co možno nejrychleji sledovat hodnoty polohy zadané pomocí nové tabulky křivky.

Příklad - Rozpojení za pohybu

Kolejový materiál, který se soustavně pohybuje skrz pracovní prostor řezacího zařízení, má být řezán na stejně velké kusy.

Osa X: Osa, ve které se kolejový materiál pohybuje. WCS
Osa X1: Osa stroje kolejového materiálu, MCS
Osa Y: Osa, ve které se spolu s kolejovým materiálem pohybuje řezací zařízení

Předpokládá se, že přísuv řezacího zařízení a jeho ovládání jsou řízeny prostřednictvím PLC. Pro definici synchronního chování mezi kolejovým materiálem a řezacím zařízením mohou být vyhodnocovány signály rozhraní PLC.

Akce
Aktivování vazby, `LEADON`
Rozpojení vazby, `LEADOF`
Nastavení skutečné hodnoty, `PRESETON`

Programový kód	**Komentář**
N100 R3=1500	; Délka kusu, který má být odřezán
N200 R2=100000 R13=R2/300	
N300 R4=100000	
N400 R6=30	; Počáteční pozice osy Y
N500 R1=1	; Počáteční podmínka pro pásovou osu
N600 LEADOF(Y,X)	; Vymazání případně existující vazby
N700 CTABDEF(Y,X,1,0)	; Definice tabulky
N800 X=30 Y=30	; Dvojice hodnot
N900 X=R13 Y=R13	
N1000 X=2*R13 Y=30	
N1100 CTABEND	; Konec definice tabulky
N1200 PRESETON(X1,0)	; PRESET na začátek
N1300 Y=R6 G0	; Počáteční poloha osy Y, osa je lineární
N1400 ID=1 WHENEVER $AA_IW[X]>$R3 DO PESETON(X1,0)	; PRESET po délce R3, nový začátek po oddělení
N1500 RELEASE(Y)	
N1800 ID=6 EVERY $AA_IM[X]<10 DO LEADON(Y,X,1)	; Vytvoření vazby podle tabulky 1 navázané na X v případě X < 10
N1900 ID=10 EVERY $AA_IM[X]>$R3-30 DO EADOF(Y,X)	; > 30 před odjetou oddělovanou délkou vytvořit vazbu
N2000 WAITP(X)	
N2100 ID=7 WHEN $R1==1 DO MOV[X]=1 FA[X]=$R4	; Nastavit osu koleje do trvalého pohybu
N2200 M30	

10.4.24 Měření (MEAWA, MEAC)

Funkce

Ve srovnání s použitím v pohybových blocích výrobního programu může být měřicí funkce prostřednictvím synchronní akce libovolně zapínána a vypínána.

Pokud budete potřebovat další informace týkající se měření, viz Speciální příkazy dráhy "Rozšířené měřicí funkce".

Syntaxe

Měření jednou osou bez vymazání zbytkové dráhy

```
MEAWA[osa] = (režim, spouštěcí událost_1, ..._4)
```

Kontinuální měření bez mazání zbytkové dráhy

```
MEAC[osa] = (režim, paměť měření, spouštěcí událost_1, ..._4))
```

Význam

Programový kód	Komentář		
DO MEAWA	;	Aktivování axiálního měření	
DO MEAC	;	Aktivování kontinuálního měření	
Osa	;	Název osy, pro kterou se měření provádí	
Režim	;	Údaj na místě **desítek** 0: aktivní měřicí systém	Údaj na místě **jednotek** 0: Přerušení měřicí úlohy
		Počet měřicích systémů (v závislosti na režimu) 1: 1. Měřicí systém 2: 2. Měřicí systém 3: Oba měřicí systémy	Až 4 aktivovatelné spouštěcí události 1: současně 2: postupně 3: stejně jako 2., ale bez monitorování spouštěcí události 1 na začátku
Spouštěcí událost_1 až_4	;	: náběžná hrana měřicí sondy 1 -1: sestupná hrana měřicí sondy 1 (volitelné) 2: náběžná hrana měřicí sondy 2 (volitelné) -2: sestupná hrana měřicí sondy 2 (volitelné)	
Paměť měření	;	Číslo zásobníkové paměti FIFO	

10.4.25 Inicializace proměnných typu pole (SET, REP)

Funkce

V synchronních akcích mohou být proměnné typu pole inicializovány nebo do nich mohou být zapisovány určité hodnoty.

Poznámka

Možné jsou pouze takové proměnné, do nichž lze v synchronních akcích zapisovat. Strojní parametry se takto inicializovat nedají. Hodnota NO_AXIS nemůže být proměnným typu AXIS zadávána.

Syntaxe

```
DO FELD[n,m]=SET(<hodnota1>,<hodnota2>,...)
DO FELD[n,m]=REP(<hodnota>)
```

Význam

`FELD[n,m]`	Programovatelné indexy pole Inicializace začíná u naprogramovaných indexů pole. V případě 2-rozměrných polí se napřed inkrementuje 2. index. V případě osových indexů se tento proces neuskuteční.
`SET(<hodnota1>, <hodnota2>,...)`	Inicializace se seznamem hodnot Do pole se od naprogramovaných indexů pole zapíší parametry uvedené v příkazu `SET`. Je přiřazeno tolik prvků pole, kolik je programováno hodnot. Je-li programováno více hodnot než je k dispozici zbývajících prvků pole, spustí se alarm.
`REP(<hodnota>)`	Inicializace pomocí stejných hodnot Do pole se od naprogramovaných indexů pole až do konce pole bude opakovaně zapisovat parametr (`<hodnota>`) uvedený v příkazu `REP`.

Příklad

Programový kód	**Komentář**
`WHEN TRUE DO SYG_IS[0]=REP(0)`	`; Výsledek:`

`WHEN TRUE DO SYG_IS[1]=SET(3,4,5)`	`; SYG_IS[0]=0`
	`SYG_IS[1]=3`
	`SYG_IS[2]=4`
	`SYG_IS[3]=5`
	`SYG_IS[4]=0`

10.4.26 Dosazení/vymazání značky pro čekání (SETM, CLEARM)

Funkce

V synchronních akcích mohou být čekací značky nastavovány, příp. mazány, aby se např. kanály koordinovaly mezi sebou.

Syntaxe

```
DO SETM(<číslo značky>)
DO CLEARM(<číslo značky>)
```

Význam

`SETM`	Příkaz pro nastavení čekací značky pro daný kanál Příkaz `SETM` může být zapsán jak ve výrobním programu, tak také v akční části synchronní akce. Nastavuje značku (`<číslo značky>`) pro kanál, v němž je příkaz zpracován.
`CLEARM`	Příkaz pro vymazání čekací značky pro daný kanál Příkaz `CLEARM` může být zapsán jak ve výrobním programu, tak také v akční části synchronní akce. Vymaže značku (`<číslo značky>`) pro kanál, v němž je příkaz zpracován.
`<číslo značky>`	Čekací značka

10.4.27 Reakce na chybu (SETAL)

Funkce

Pomocí synchronních akcí mohou být naprogramovány reakce na chybu. Přitom jsou zjišťovány hodnoty stavových proměnných a spouští se odpovídající akce.

Možné reakce na chybové stavy jsou následující:

- Zastavení osy (override = 0)
- Aktivování alarmu

  Pomocí příkazu `SETAL` mohou být aktivovány alarmy cyklů ze synchronních akcí.
- Nastavení výstupu
- Veškeré akce možné v synchronních akcích

Syntaxe

Aktivování alarmů cyklů:

```
DO SETAL(<číslo alarmu>)
```

Význam

`SETAL`	Příkaz pro aktivování alarmu cyklu	
`<Číslo alarmu>`	Číslo alarmu	
	Interval alarmů cyklů pro uživatele:	65000 až 69999

Příklad

Programový kód	**Komentář**
`ID=67 WHENEVER ($AA_IM[X1]-$AA_IM[X2])<4.567 DO $AA_OVR[X2]=0`	`; Jestliže je bezpečnostní vzdálenost mezi osami X1 a X2 příliš malá, osu X2 zastavit.`
`ID=67 WHENEVER ($AA_IM[X1]-$AA_IM[X2])<4.567 DO SETAL(65000)`	`; Jestliže je bezpečnostní vzdálenost mezi osami X1 a X2 příliš malá, aktivovat alarm 65000.`

10.4.28 Najíždění na pevný doraz (FXS, FXST, FXSW, FOCON, FOCOF)

Funkce

Příkazy pro funkci "Najíždění na pevný doraz" se v synchronních akcích/technologických cyklech programují pomocí příkazů používaných ve výrobním programu `FXS`, `FXST` a `FXSW`.

Aktivování se může uskutečňovat bez pohybů, moment bude okamžitě omezen. Pokud se má osa pohybovat přes nastavený bod, monitorování dorazu se aktivuje.

Najíždění s omezeným momentem/silou (FOC)

Tato funkce dovoluje prostřednictvím synchronních akcí kdykoli změnit moment/sílu a může být aktivována jak s modální, tak i blokovou platností.

Syntaxe

```
FXS[<osa>]
FXST[<osa>]
FXSW[<osa>]
FOCON[<osa>]
FOCOF[<osa>]
```

Význam

`FXS`	Aktivování jedině v systémech s digitálními pohony (VSA, HSA, HLA)
`FXST`	Změna momentu sevření FXST
`FXSW`	Změna monitorovacího okna FXSW
`FOCON`	Aktivování omezení momentu/síly s modální platností
`FOCOF`	Deaktivování omezení momentu/síly
<osa>	Identifikátor osy Přípustné je následující: • Identifikátor geometrické osy • Identifikátor kanálové osy • Identifikátor osy stroje

Poznámka

Aktivování se smí uskutečnit jen jednou.

Příklady

Příklad 1: Najíždění na pevný doraz (FXS) spouštěné prostřednictvím synchronní akce

Programový kód	**Komentář**
Osa Y:	; Statické synchronní akce
aktivování:	
N10 IDS=1 WHENEVER (($R1==1) AND $AA_FXS[y]==0)) D $R1=0 FXS[Y]=1 FXST[Y]=10 FA[Y]=200 POS[Y]=150	
	; Nastavením parametru $R1=1 se pro osu Y aktivuje FXS, platný moment je snížen na 10% a spouští se pohyb pracovním posuvem ve směru dorazu.
N11 IDS=2 WHENEVER ($AA_FXS[Y]==4) DO FXST[Y]=30	; Jakmile byl pevný doraz rozpoznán ($AA_FXS[Y]==4), zvyšuje se moment na 30%.
N12 IDS=3 WHENEVER ($AA_FXS[Y]==1) DO FXST[Y]=$R0	; Po dosažení dorazu je moment řízen v závislosti na parametru R0.
N13 IDS=4 WHENEVER (($R3==1) AND $AA_FXS[Y]==1)) DO FXS[Y]=0 FA[Y]=1000 POS[Y]=0	
	; Deaktivování v závislosti na R3 a zpětný pohyb.
N20 FXS[Y]=0 G0 G90 X0 Y0	; Normální zpracování programu:
N30 RELEASE(Y)	; Osa Y je uvolněna pro pohyb v rámci synchronní akce.
N40 G1 F1000 X100	; Pohyb jiné osy.
N50 ...	
N60 GET(Y)	; Osa Y je znovu převzata do svazku dráhových os.

Příklad 2: Aktivování omezení momentu/síly (FOC)

Programový kód	**Komentář**
N10 FOCON[X]	; Modální aktivování omezení.
N20 X100 Y200 FXST[X]=15	; Osa X se pohybuje se zmenšeným momentem (15%).
N30 FXST[X]=75 X20	; Změna momentu na 75%, osa X se pohybuje s tímto omezeným momentem.
N40 FOCOF[X]	; Deaktivování omezení momentu.

Další informace

Několikanásobné aktivování

Jestliže je v důsledku chyby v programu funkce po svém aktivování (`FXS[<osa>]=1`) vyvolána ještě jednou, aktivuje se následující alarm:

Alarm 20092 "Najíždění na pevný doraz je ještě aktivní"

Programové sekvence, ve kterých je v podmínce zjišťována hodnota buď proměnné $AA_FXS[ ] nebo vlastní značka (v tomto případě R1), zabraňují, aby funkce "Fragment výrobního programu" byla aktivována vícekrát.

Programový kód

```
N10 R1=0
N20 IDS=1 WHENEVER ($R1==0 AND
$AA_IW[AX3] > 7) DO R1=1 FXST[AX1]=12
```

Synchronní akce vztahující se k bloku

Naprogramováním synchronní akce vztahující se k bloku může být najíždění na pevný doraz připojeno v průběhu najížděcího pohybu.

Příklad:

Programový kód	**Komentář**
`N10 G0 G90 X0 Y0`	
`N20 WHEN $AA_IW[X] > 17 DO FXS[X]=1`	`; Jestliže osa X dosáhne polohy větší než 17 mm, aktivuje se FXS.`
`N30 G1 F200 X100 Y110`	

Statické a blokové synchronní akce

Ve statických a blokových synchronních akcích mohou být používány stejné příkazy `FXS`, `FXST` a `FXSW` jako při zpracovávání normálního výrobního programu. Hodnoty, které se přiřazují, mohou vzniknout i výpočtem.

10.4.29 Stanovení úhlu tečny k dráze v synchronních akcích

Funkce

Systémová proměnná $AC_TANEB (**T**angent **AN**gel at **E**nd of **B**lock), která může být čtena v synchronních akcích, obsahuje úhel mezi tečnou ke dráze v koncovém bodě aktuálního bloku a tečnou ke dráze v počátečním bodě následujícího naprogramovaného bloku.

Parametry

Úhel tečny ke dráze je vždy kladný v rozsahu od 0,0 do 180,0 stupňů. Jestliže už žádný následující blok v hlavním zpracování neexistuje, bude dosazen úhel -180,0 stupňů.

Systémová proměnná `$AC_TANEB` by neměla být čtena pro bloky, které jsou generovány systémem (pomocné vkládané bloky). Pro rozhodování, zda se jedná o naprogramovaný blok (hlavní blok), slouží systémová proměnná `$AC_BLOCKTYPE`.

Příklad

```
ID=2 EVERY $AC_BLOCKTYPE==0 DO $SR1 = $AC_TANEB
```

10.4.30 Stanovení aktuální korekce typu override

Funkce

Aktuální korekce typu override

(NC-složka) Pro čtení a pro zápis je možné používat následující systémové proměnné:

`$AA_OVR` Override (korekce) osy

`$AA_OVR` Korekce (override) dráhy

v synchronních akcích.

Korekce (override) zadávené z PLC jsou připraveny v systémových proměnných pro synchronní akce:

`$AA_PLC_OVR` Override (korekce) osy

`$AC_PLC_OVR` Korekce (override) dráhy

ale jen pro čtení.

Výsledná korekce (override)

pro systémové akce je připravena v systémových proměnných:

`$AA_TOTAL_OVR` Override (korekce) osy

`$AC_TOTAL_OVR` Korekce (override) dráhy

ale jen pro čtení.

Výsledná korekce (override) se vypočítá jako:

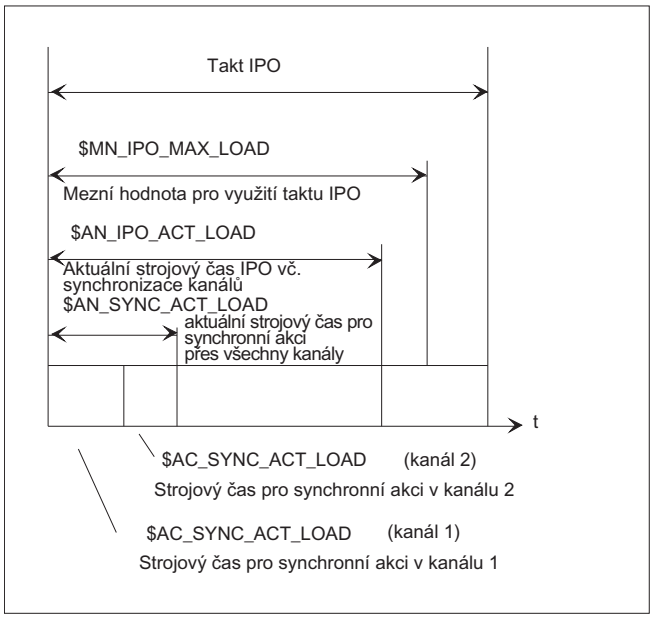
`$AA_OVR * $AA_PLC_OVR`, příp.

`$AC_OVR * $AC_PLC_OVR`

10.4.31 Vyhodnocování vytížení pomocí časové náročnosti synchronních akcí

Funkce

V rámci interpolačního taktu musí být jednak interpretovány synchronní akce, současně ale se vypočítávají pohyby atd. z NC systému. Prostřednictvím systémových proměnných uvedených v následujících odstavcích se mohou synchronní akce informovat o aktuálních časových podílech synchronních akcí na interpolačním taktu a na strojovém času regulátoru polohy.



Význam

Proměnné mají platné hodnoty jedině tehdy, pokud je strojní parametr `$MN_IPO_MAX_LOAD` větší než 0. Jinak jak u systémů SINUMERIK powerline, tak také solution line, udávají tyto proměnné vždy čistý strojový čas, který už nezohledňuje přerušení, která byla vyvolána prostřednictvím HMI. Čistý strojový čas se skládá z následujících složek:

- čas synchronní akce
- čas regulátoru polohy
- zbývající strojový čas IPO bez přerušení podmíněných HMI

Systémové proměnné vždy obsahují hodnoty týkající se **předcházejícího** taktu IPO

$AN_IPO_ACT_LOAD	Aktuální strojový čas IPO (včetně synchronních akcí všech kanálů)
$AN_IPO_MAX_LOAD	Nejdelší strojový čas IPO (včetně synchronních akcí všech kanálů)
$AN_IPO_MIN_LOAD	Nejkratší strojový čas IPO (včetně synchronních akcí všech kanálů)
$AN_IPO_LOAD_PERCENT	Aktuální strojový čas IPO v poměru k taktu IPO (%)
$AN_SYNC_ACT_LOAD	Aktuální strojový čas pro synchronní akce přes všechny kanály
$AN_SYNC_MAX_LOAD	Nejdelší strojový čas pro synchronní akce přes všechny kanály
$AN_SYNC_TO_IPO	Procentuální podíl celkových synchronních akcí na celkovém strojovém čase IPO (přes všechny kanály)
$AC_SYNC_ACT_LOAD	Aktuální strojový čas pro synchronní akce v kanálu
$AC_SYNC_MAX_LOAD	Nejdelší strojový čas pro synchronní akce v kanálu
$AC_SYNC_AVERAGE_LOAD	Průměrný strojový čas pro synchronní akce v kanálu
$AN_SERVO_ACT_LOAD	Aktuální strojový čas regulátoru polohy
$AN_SERVO_MAX_LOAD	Nejdelší strojový čas regulátoru polohy
$AN_SERVO_MIN_LOAD	Nejkratší strojový čas regulátoru polohy

Proměnné pro zjišťování vytížení:

Prostřednictvím strojního parametru $MN_IPO_MAX_LOAD se nastavuje, od kterého čistého strojového času IPO (v % z taktu IPO) se má systémová proměnná $AN_IPO_LOAD_LIMIT nastavovat na hodnotu TRUE. Jestliže aktuální vytížení poklesne opět pod tuto hranici, nastaví se proměnná znovu na hodnotu FALSE. Pokud má tento strojní parametr hodnotu 0, je celá diagnostická funkce deaktivována.

Na základě vyhodnocování proměnné $AN_IPO_LOAD_LIMIT může uživatel definovat svou vlastní strategii, aby se přeběhu roviny zabránilo.

10.5 Technologické cykly

Funkce

Jako akce v synchronních akcích mohou být vyvolávány také programy, které však smí být sestaveny výhradně z funkcí, které jsou přípustné také jako akce v synchronních akcích. Takto sestavené programy se nazývají technologické cykly.

Technologické cykly jsou v řídícím systému uloženy jako podprogramy.

V jednom kanálu může být souběžně zpracováváno i několik technologických cyklů nebo akcí.

Programování

Pro programování technologických cyklů platí následující pravidla:

- Konec programu je naprogramován pomocí příkazů `M02`/`M17`/`M30`/`RET`.
- V rámci programové úrovně mohou být všechny akce uvedené v příkazu `ICYCOF` zpracovány v jednom taktu bez čekacích cyklů.
- Na jednu synchronní akci může být postupně otestováno až 8 technologických cyklů.
- Technologické cykly jsou možné také v synchronních akcích s blokovou platností.
- Mohou být naprogramovány jak řídící struktury s příkazem `IF`, tak také příkazy skoků `GOTO`, `GOTOF` a `GOTOB`.
- Pro bloky s příkazy `DEF` a `DEFINE` platí:
  - Příkazy `DEF` a `DEFINE` jsou v technologických cyklech zpracovávány.
  - V případě, že jejich syntaxe není správná nebo je neúplná, mají za následek alarmové hlášení.
  - Mohou být bez alarmového hlášení zpracovány, aniž by samy byly založeny.
  - Jako technologický cyklus s přiřazením hodnot jsou úplně celé zohledňovány.

Předávání parametrů

Předávání parametrů do technologických cyklů je možné. Jsou zohledňovány jak jednoduché datové typy, které jsou předávány jako formální parametry "Call by Value", tak také standardní nastavení, které jsou při vyvolávání technologických cyklů v platnosti. Jedná se o následující:

- Naprogramované standardní hodnoty, jestliže není naprogramován žádný předávaný parametr.
- Dosazení inicializačních hodnot do standardních parametrů.
- Předávání neinicializovaných aktuálních parametrů se standardní hodnotou.

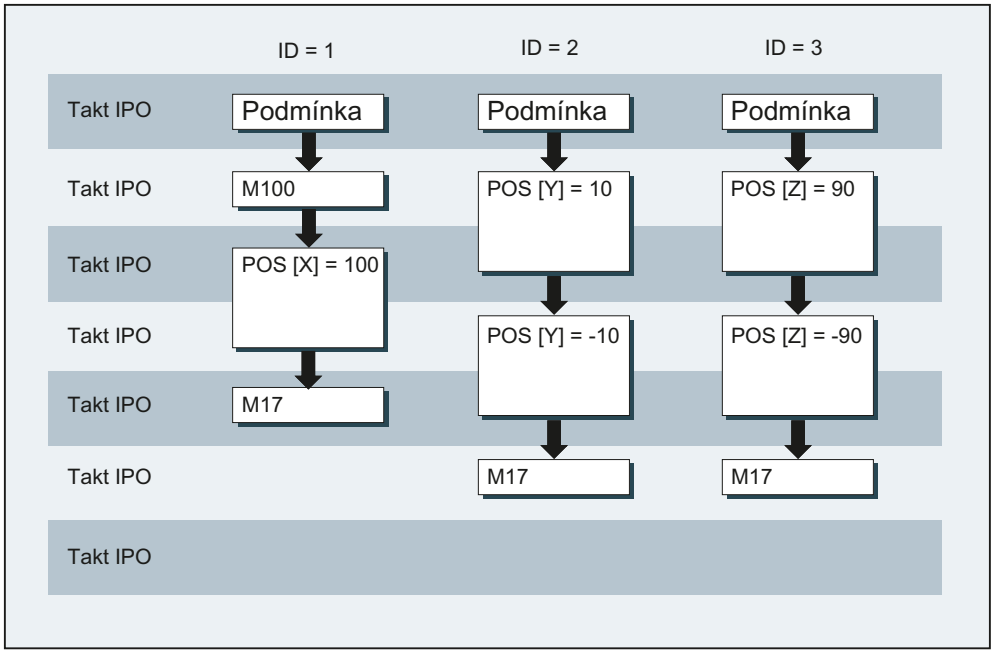
Postup

Technologické cykly se spouští, jakmile je splněna jejich podmínka. Každý řádek technologického cyklu se zpracovává v samostatném taktu IPO. V případě polohovacích os je pro uskutečnění zapotřebí více taktů IPO. V rámci jednoho taktu jsou zpracovávány i jiné funkce. V technologickém cyklu probíhá zpracovávání bloků jeden po druhém.

Jestliže jsou ve stejném interpolačním taktu vyvolávány akce, které se vzájemně vylučují, potom je aktivní ta akce, která byla vyvolána ze synchronní akce s vyšším identifikačním (ID) číslem.

Příklady

Příklad 1: Nastavením digitálních vstupů se spouští programy pro osy



Hlavní program:

Programový kód	**Komentář**
`ID=1 EVERY $A_IN[1]==1 DO ACHSE_X`	`; Jestliže je vstup 1 nastaven na 1, spustit program pro osu ACHSE_X.`
`ID=2 EVERY $A_IN[2]==1 DO ACHSE_Y`	`; Jestliže je vstup 2 nastaven na 1, spustit program pro osu ACHSE_Y.`
`ID=3 EVERY $A_IN[3]==1 DO ACHSE_Z`	`; Jestliže je vstup 3 nastaven na 1, spustit program pro osu ACHSE_Z.`
`M30`	

Program pro osu ACHSE_X:

```
Programový kód
M100
POS[X]=100 FA[X]=300
M17
```

Program pro osu ACHSE_Y:

```
Programový kód
POS[Y]=10 FA[Y]=200
POS[Y]=-10
M17
```

Program pro osu ACHSE_Z:

```
Programový kód
POS[Z]=90 FA[Z]=250
POS[Z]=-90
M17
```

Příklad 2: Různé programové sekvence v technologickém cyklu

```
Programový kód
PROC CYCLE
N10 DEF REAL WERT=12.3
N15 DEFINE ABC AS G01
```

Oba bloky jsou zpracovány bez alarmu a bez založení proměnných, příp. maker.

```
Programový kód
PROC CYCLE
N10 DEF REAL
N15 DEFINE ABC G01
```

Oba bloky kromě toho mají za následek alarm NC systému, protože syntakticky nejsou správně napsány.

```
Programový kód
PROC CYCLE
N10 DEF AXIS ACHSE1=XX2
```

Pokud osa XX2 není známa, aktivuje se alarm 12080. Jinak se blok zpracuje bez alarmu a bez založení proměnných.

```
Programový kód
PROC CYCLE
N10 DEF AXIS ACHSE1
N15 G01 X100 F1000
N20 DEF REAL WERT1
```

Blok `N20` má vždy za následek alarm 14500, protože po 1. programovém příkazu nesmí následovat žádný definiční příkaz.

10.5.1 Kontextová proměnná ($P_TECCYCLE)

Funkce

Pomocí proměnných $P_TECCYCLE mohou být odděleny programy v programech synchronních akcí a programy pro předběžné zpracování. Díky tomu je možné syntakticky správně napsané bloky nebo programové sekvence zpracovávat alternativně také jako cyklus ve výrobním programu.

Interpretace kontextové proměnné

Systémová proměnná $P_TECCYCLE umožňuje řídit specifickou kontextovou interpretaci úseků programů v technologických cyklech:

```
IF $P_TECCYCLE==TRUE
...                          ; Úsek programu pro technologický cyklus v synchronní
                               akci.
ELSE
...                          ; Úsek programu pro cyklus ve výrobním programu.
ENDIF
```

Poznámka

Blok s nesprávnou nebo nepovolenou syntaxí programových příkazů, stejně jako neznámá přiřazení hodnot mají i v cyklu výrobního programu za následek alarmové hlášení.

Příklad

Programová sekvence s vyhodnocování proměnné $P_TECCYCLE v technologickém cyklu:

Programový kód	**Komentář**
PROC CYCLE	
N10 DEF REAL WERT1	; V technologickém cyklu se zpracuje.
N15 G01 X100 F1000	
N20 IF $P_TECCYCLE==TRUE	
...	; Úsek programu pro technologický cyklus (bez proměnné WERT1).
N30 ELSE	
...	; Úsek programu pro cyklus výrobního programu (proměnná WERT1 je k dispozici).
N40 ENDIF	

10.5.2 Volání parametru hodnotou

Funkce

Technologické cykly mohou být definovány s parametry typu Call-by-Value. Jako parametry jsou možné jednoduché datové typy, jako jsou INT, REAL, CHAR, STRING, AXIS a BOOL.

Poznámka

Formálními parametry, které se předávají metodou Call-by-Value, nemohou být žádná pole.

Skutečné parametry mohou existovat i jako předdefinované parametry (viz "Inicializace předdefinovaných parametrů [Strana 638]").

Syntaxe

```
ID=1 WHEN $AA_IW[X]>50 DO TEC(IVAL,RVAL,,SVAL,AVAL)
```

V případě, že aktuální parametry nejsou inicializovány, předávají se předdefinované hodnoty:

```
ID=1 WHEN $AA_IW[X]>50 DO TEC(IVAL,RVAL,,SYG_SS[0],AVAL)
```

10.5.3 Inicializace předdefinovaných parametrů

Funkce

Předdefinovaným parametrům může být v příkazu PROC dosazována také inicializační hodnota.

Syntaxe

Přiřazení předdefinovaných parametrů v technologickém cyklu:

```
PROC TEC (INT IVAL=1, REAL RVAL=1.0, CHAR CVAL='A', STRING[10]
SVAL="ABC", AXIS AVAL=X, BOOL BVAL=TRUE)
```

Jestliže aktuální parametr existuje z předdefinovaného parametru, bude se předávat inicializační hodnota z příkazu PROC. To platí jak ve výrobním programu, tak také v synchronních akcích.

Příklad

Programový kód	**Komentář**
`TEC (IVAL, RVAL, SVAL, AVAL)`	`;  v případě proměnných CVAL a BVAL platí inicializační hodnota`

10.5.4 Řízení zpracování technologických cyklů (ICYCOF, ICYCON)

Funkce

Pro účely časového řízení zpracovávání technologických cyklů slouží příkazy NC jazyka ICYCOF a ICYCON.

Pomocí příkazu ICYCOF se nastaví, že všechny bloky technologického cyklu se zpracují jen v jednom interpolačním taktu. Všechny akce, jejichž zpracování vyžaduje větší počet taktů, mají v případě příkazu ICYCOF za následek paralelní procesy zpracování.

Aplikace

V případě příkazu ICYCON mohou pohyby příkazových způsobovat, že se zpracování technologického cyklu zpozdí. Jestliže je to nežádoucí, potom je možné pomocí příkazu ICYCOF zpracovat všechny akce bez čekání v jednom interpolačním taktu.

Syntaxe

Pro cyklické zpracovávání technologických cyklů platí:

`ICYCON` Každý blok technologického cyklu za příkazem ICYCON se zpracovává v samostatném taktu IPO.

`ICYCOF` Všechny bloky technologického cyklu následující za příkazem ICYCOF se zpracují v jednom taktu IPO.

Poznámka

Oba příkazy NC jazyka ICYCON a ICYCOF se uplatňují jen v rámci stejné programové úrovně. Ve výrobním programu jsou oba příkazy jednoduše zpracovány bez jakékoli reakce.

Příklad pro režim zpracování ICYCOF

Programový kód	**Komentář**
`Takt IPO`	`; PROC TECHNOCYC`
`1.`	`; $R1=1`
`2.25`	`; POS[X]=100`
`26.`	`; ICYCOF`
`26.`	`; $R1=2`
`26.`	`; $R2=$R1+1`
`26.`	`; POS[X]=110`
`26.`	`; $R3=3`
`26.`	`; RET`

10.5.5 Kaskádové řazení technologických cyklů

Funkce

Může být postupně zpracováno až 8 technologických cyklů v řadě. Tímto způsobem lze v jedné synchronní akci naprogramovat větší počet technologických cyklů.

Syntaxe

```
ID=1 WHEN $AA_IW[X]>50 DO TEC1($R1) TEC2 TEC3(X)
```

Posloupnost při zpracování

Technologické cykly jsou zpracovávány v řadě po sobě (v kaskádě) zleva doprava v souladu s výše uvedeným příkladem programování. Jestliže by měl být cyklus zpracováván v režimu ICYCON, všechny následující kroky zpracování se tím zpozdí. Pokud se vyskytne alarm, všechny následující akce se přeruší.

10.5.6 Technologické cykly v blokových synchronních akcích

Funkce

Technologické cykly jsou možné také v synchronních akcích s blokovou platností.

Jestliže je doba zpracování technologického cyklu delší než doba zpracovávání odpovídajících bloků, potom se technologický cyklus při přechodu na další blok přeruší.

Poznámka

Technologický cyklus nebrání přechodu na další blok.

10.5.7 Řídící struktury (IF)

Funkce

Pro účely větvení v posloupnosti zpracovávaných příkazů technologického cyklu se mohou v synchronních akcích používat řídící struktury s příkazem IF.

Syntaxe

```
IF <podmínka>
$R1=1
[ELSE] nepovinné
$R1=0
ENDIF
```

10.5.8 Příkazy skoku (GOTO, GOTOF, GOTOB)

Funkce

V technologických cyklech jsou možné příkazy skoků GOTO, GOTOF, GOTOB. Uvedená návěští musí být v programu k dispozici, jinak se aktivuje alarm.

Poznámka

Návěští a čísla bloků smí být jen konstanty.

Syntaxe

Nepodmíněné skoky

`GOTO` Návěští, číslo bloku

`GOTOF` Návěští, číslo bloku

`GOTOB` Návěští, číslo bloku

Příkazy skoku a cíle skoku

`GOTO`	Skok napřed dozadu a potom dopředu
`GOTOF`	Skok směrem dozadu
`GOTOB`	Skok směrem dopředu
`Label:`	Návěští skoku
`Číslo bloku`	Cílem skoku je tento blok
`N100`	Číslem bloku je vedlejší blok
`:100`	Číslem bloku je hlavní blok

10.5.9 Blokování, uvolnění, reset (LOCK, UNLOCK, RESET)

Funkce

Zpracovávání technologického cyklu může být prostřednictvím jiné modální synchronní akce zablokováno, znovu odblokováno nebo resetováno.

Syntaxe

```
LOCK(<n1>,<n2>,...)
UNLOCK(<n1>,<n2>,...)
RESET(<n1>,<n2>,...)
```

Význam

`LOCK`	Příkaz pro zablokování synchronní akce Aktivní akce se přeruší.
`UNLOCK`	Příkaz pro odblokování synchronní akce
`RESET`	Příkaz pro reset technologického cyklu
`<n1>,<n2>,...`	Identifikační čísla synchronních akcí, příp. technologických cyklů, které mají být zablokovány, odblokovány nebo resetovány.

Blokování synchronních akcí

Modální synchronní akce s identifikačními čísly (ID) **<n> = 1 ... 64** mohou být zablokovány z PLC. Odpovídající podmínka se tím pádem už nevyhodnocuje a zpracovávání příslušné funkce je v NCK blokováno.

Pomocí signálu na rozhraní PLC je možné paušálně zablokovat všechny synchronní akce.

Poznámka

Naprogramovaná synchronní akce je standardně aktivní a může být proti přepsání/ zablokování chráněna pomocí strojního parametru.

Synchronní akce definované výrobcem stroje by neměly být koncovým zákazníkem nijak ovlivňovány.

Příklady

Příklad 1: Blokování synchronních akcí (LOCK)

Programový kód

```
N100 ID=1 WHENEVER $A_IN[1]==1 DO M130
...
N200 ID=2 WHENEVER $A_IN[2]==1 DO LOCK(1)
```

Příklad 2: Odblokování synchronních akcí (UNLOCK)

Programový kód

```
N100 ID=1 WHENEVER $A_IN[1]==1 DO M130
...
N200 ID=2 WHENEVER $A_IN[2]==1 DO LOCK(1)
...
N250 ID=3 WHENEVER $A_IN[3]==1 DO UNLOCK(1)
```

Příklad 3: Přerušení technologického cyklu (RESET)

Programový kód

```
N100 ID=1 WHENEVER $A_IN[1]==1 DO M130
...
N200 ID=2 WHENEVER $A_IN[2]==1 DO RESET(1)
```

10.6 Zrušení synchronní akce (CANCEL)

Funkce

Pomocí příkazu `CANCEL` z výrobního programu může být synchronní akce s modální nebo statickou platností přerušena (vymazána).

Pokud je synchronní akce přerušena, přičemž mezitím stále ještě probíhá pohyb polohovací osy, který tato akce spustila, pohyb polohovací osy se ukončí. Jestliže je to nežádoucí, může být pohyb osy zabrzděn s vymazáním zbytkové dráhy pro danou osu ještě před příkazem `CANCEL`.

Syntaxe

```
CANCEL(<n1>,<n2>,...)
```

Význam

`CANCEL`:	Příkaz pro vymazání naprogramované synchronní akce
`<n1>,<n2>,...`:	Identifikační čísla synchronních akcí, které mají být vymazány. **Upozornění:** Bez uvedení identifikačních čísel jsou vymazány **všechny** modální/ statické synchronní akce.

Příklady

Příklad 1: Přerušení synchronní akce

Programový kód	Komentář
`N100 ID=2 WHENEVER $A_IN[1]==1 DO M130`	
`...`	
`N200 CANCEL(2)`	`; Vymaže se modální synchronní akce č. 2.`

Příklad 2: Vymazání zbytkové dráhy před přerušením synchronní akce

Programový kód	Komentář
`N100 ID=17 EVERY $A_IN[3]==1 DO POS[X]=15 FA[X]=1500`	`; Spuštění pohybu polohovací osy.`
`...`	
`N190 WHEN ... DO DELDTG(X)`	`; Ukončení pohybu polohovací osy.`
`N200 CANCEL(17)`	`; Vymaže se modální synchronní akce č. 17.`

10.7 Chování řídícího systému v určitých provozních stavech

POWER ON

Při zapnutí systému (Power-On) nejsou v principu aktivní žádné synchronní akce. Statické synchronní akce mohou být aktivovány prostřednictvím asynchronního podprogramu (`ASUP`) spouštěného z PLC.

Změna provozního režimu

Synchronní akce aktivované pomocí klíčového slova `IDS` zůstávají aktivní, i když se změní provozní režim. Všechny zbývající synchronní akce se při změně provozního režimu deaktivují (např. nastavování polohy osy) a při nastavení na původní polohu a přepnutí zpět do automatického režimu se znovu aktivují.

RESET

Když je proveden RESET NC systému, všechny synchronní akce s blokovou i s modální platností se ukončí. Statické synchronní akce zůstávají aktivní. Z nich se mohou spouštět nové akce. Pokud je při resetu aktivní pohyb nějaké příkazové osy, přeruší se. Již uskutečněné synchronní akce typu WHEN se po resetu už dále nezpracovávají.

Chování po resetu		
Synchronní akce/ Technologický cyklus	**modální / bloková**	**statická (IDS)**
	Aktivní akce se přeruší, synchronní akce se vymaže.	Aktivní akce se přeruší, technologický cyklus se resetuje.
Osa/ polohovací vřeteno	Pohyb se přeruší.	Pohyb se přeruší.
Vřeteno s regulací otáček	$MA_SPIND_ACTIVE_AFTER_RESET==1: Vřeteno zůstává aktivní $MA_SPIND_ACTIVE_AFTER_RESET==0: Vřeteno je zastaveno.	
Vazba řídící hodnotou	$MC_RESET_MODE_MASK, Bit13 == 1: Vazba řídící hodnotou zůstává aktivní $MC_RESET_MODE_MASK, Bit13 == 0: Vazba řídící hodnotou se zruší.	
Měřicí operace	Měřicí operace spouštěné ze synchronních akcí se přeruší.	Měřicí operace spouštěné ze statických synchronních akcí se přeruší.

NC-Stop

Statické synchronní akce zůstávají ve stavu NC-Stop aktivní. Pohyby spouštěné ze statických synchronních akcí se nepřeruší. **Programově lokální** synchronní akce náležející k aktivnímu bloku zůstávají aktivní, z nich spouštěné pohyby jsou přerušeny.

Konec programu

Konec programu a synchronní akce se vzájemně neovlivňují. Probíhající synchronní akce se ukončí i po skončení programu. Synchronní akce aktivní v bloku s příkazem M30 zůstávají v bloku s příkazem M30 aktivní. Jestliže je to nežádoucí, musí být synchronní akce před koncem programu přerušena příkazem `CANCEL`.

Chování po skončení programu		
Synchronní akce/ Technologický cyklus	**modální / bloková → přeruší se**	**statická (IDS) → zůstane zachována**
Osa/ polohovací vřeteno	Příkaz M30 bude pozastaven, dokud se osa/ vřeteno nezastaví.	Pohyb probíhá dál.
Vřeteno s regulací otáček	Konec programu: $MA_SPIND_ACTIVE_AFTER_RESET==1: Vřeteno zůstává aktivní $MA_SPIND_ACTIVE_AFTER_RESET==0: Vřeteno je zastaveno Při změně provozního režimu zůstává vřeteno aktivní.	Vřeteno zůstává aktivní.
Vazba řídící hodnotou	$MC_RESET_MODE_MASK, Bit13 == 1: Vazba řídící hodnotou zůstává aktivní $MC_RESET_MODE_MASK, Bit13 == 0: Vazba řídící hodnotou se zruší.	Vazba spuštěná ze statické synchronní akce zůstává zachována.
Měřicí operace	Měřicí operace spouštěné ze synchronních akcí se přeruší.	Měřicí operace spouštěné ze statických synchronních akcí zůstávají aktivní.

Vyhledávání bloku

V průběhu vyhledávání bloku jsou synchronní akce soustředěny, a když je stisknuto tlačítko NC-Start, vyhodnotí se, které z příslušných akcí s v případě nutnosti spustí. Statické synchronní akce se uplatňují i v průběhu vyhledávání bloku. Pokud jsou při vyhledávání bloku nalezeny koeficienty polynomu naprogramované pomocí příkazu `FCTDEF`, jsou přímo aktivovány.

Přerušení zpracování programu prostřednictvím asynchronního podprogramu ASUP

Začátek podprogramu ASUP:

Modální a statické pohybové synchronní akce zůstávají zachovány a uplatňují se také v asynchronním podprogramu.

Konec podprogramu ASUP:

Jestliže asynchronní podprogram nepokračuje příkazem REPOS, uplatňují se modální a statické pohybové synchronní akce, které byly v asynchronním podprogramu změněny, i nadále v hlavním programu.

Obnovení původní polohy (REPOS)

Po obnovení původní polohy (REPOS) jsou synchronní akce, které byly v platnosti v přerušením bloku, znovu v platnosti. Modální synchronní akce, které byly v rámci asynchronního podprogramu změněny, nejsou po příkazu REPOS při zpracování zbývajících bloků v platnosti.

Koeficienty polynomu, které byly naprogramovány pomocí příkazu `FCTDEF`, nejsou asynchronními podprogramy a funkcí REPOS ovlivňovány. Nezávisle na tom, kde byly naprogramovány, jsou jak v asynchronním podprogramu, tak i v hlavním programu kdykoli použitelné i po zpracování příkazu REPOS.

Chování při alarmech

Když se aktivuje alarm se zastavením pohybů, jsou pohyby os a vřeten spuštěné prostřednictvím synchronních akcí zabrzděny. Všechny ostatní akce (jako např. nastavení výstupu) jsou dále provedeny.

Jestliže alarm spustí samotná synchronní akce, potom se zpracování přeruší a následující akce v rámci této synchronní akce se už neuskuteční. Pokud se jedná o synchronní akci s modální platností, v následujícím interpolačním taktu se už nebude zpracovávat. Alarm je tedy aktivován je jedenkrát. Všechny další synchronní akce se už dále nezpracovávají.

Alarmy, jejichž reakce na alarm způsobuje zastavení překladače, se uplatní až po zpracování dekódovaných bloků.

Pokud technologický cyklus spustí alarm se zastavením pohybů, není už tento technologický cyklus dále zpracováván.

Pohyb tam a zpět 11

11.1 Asynchronní oscilační pohyby (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB)

Funkce

Osa vykonávající oscilační pohyb se pohybuje mezi dvěma body obratu 1 a 2 se zadanou hodnotou posuvu, dokud se pohyb tam a zpět nevypne.

Ostatní osy mohou být v průběhu oscilačního pohybu libovolně interpolovány. Prostřednictvím pohybu po dráze nebo s polohovací osou je možné dosáhnout spojitého přísuvu. Přitom však neexistuje **žádná souvislost** mezi oscilačním pohybem a přísuvným pohybem.

Vlastnosti asynchronního oscilačního pohybu

- Asynchronní oscilační pohyb je vztažen na specifickou osu a je v platnosti i přes hranice bloků.
- Blokově synchronní zapínání oscilačního pohybu je zajišťováno pomocí výrobního programu.
- Společná interpolace většího počtu os a superpozice úseku pro oscilační pohyb nejsou možné.

Programování

Zapínání a ovlivňování asynchronního oscilačního pohybu u výrobního programu, které odpovídá zpracování NC programu, je možné prostřednictvím následujících příkazů.

Naprogramované hodnoty se synchronně s blokem v rámci zpracování hlavního programu zadávají do odpovídajících nastavovaných parametrů a zůstávají v platnosti, dokud nejsou změněny.

Syntaxe

```
OSP1[<osa>]=<hodnota> OSP2[<osa>]=<hodnota>
OST1[<osa>]=<hodnota> OST2[<osa>]=<hodnota>
FA[<osa>]=<hodnota>
OSCTRL[<osa>]=(<volba pro aktivování>,<volba pro deaktivování>)
OSNSC[<osa>]=<hodnota>
OSE[<osa>]=<hodnota>
OSB[<osa>]=<hodnota>
OS[<osa>]=1
OS[<osa>]=0
```

Význam

`<osa>` Posuv oscilační osy

`OS` Zapínání a vypínání oscilačního pohybu

Hodnota: 1 **Aktivování** oscilačního pohybu

0 **Deaktivování** oscilačního pohybu

`OSP1` Definice polohy bodu obratu 1

`OSP2` Definice polohy bodu obratu 2

Upozornění:
Jestliže je aktivní inkrementální zadávání rozměrů, vypočítá se poloha inkrementálně vůči odpovídajícímu bodu obratu naposled naprogramovanému v NC programu.

`OST1` Definice doby pozastavení v bodě obratu 1 v [s].

`OST2` Definice doby pozastavení v bodě obratu 2 v [s].

`<hodnota>`:
- -2 Interpolace bude pokračovat bez čekání na přesné najetí
- -1 Čekání na hrubé přesné najetí
- 0 Čekání na jemné přesné najetí
- >0 Čekání na jemné přesné najetí a následné setrvání po zadanou dobu pozastavení

  Upozornění:
  Jednotky pro dobu pozastavení jsou identické s dobou pozastavení naprogramovanou pomocí příkazu `G4`.

`FA` Definice rychlosti posuvu

Jako rychlost posuvu platí definovaná rychlost posuvu polohovací osy. Jestliže rychlost posuvu není definována, platí hodnota uložená ve strojním parametru.

`OSCTRL` Zadání možností pro aktivování a deaktivování

Hodnoty voleb 0 - 3 určují chování v bodech obratu při vypínání. Mohou být zvoleny varianty 0 - 3. Zbývající nastavení mohou být podle potřeby kombinovány se zvolenou variantou. Větší počet voleb může být uveden postupně pomocí znamének plus (+).

`<hodnota>`:

- 0 Při vypnutí oscilačního pohybu zastavit v následujícím bodě obratu (předdefinované nastavení).
  Upozornění:
  Hodnoty 1 a 2 jsou možné jen prostřednictvím resetu.
- 1 Při vypnutí oscilačního pohybu zastavit v bodě obratu 1.
- 2 Při vypnutí oscilačního pohybu zastavit v bodě obratu 2.
- 3 Při vypnutí oscilačního pohybu nenajíždět do žádného bodu obratu, jestliže nejsou naprogramovány žádné vyjiskřovací zdvihy.
- 4 Po vyjiskření najet na koncovou pozici.
- 8 Jestliže je oscilační pohyb přerušen prostřednictvím vymazání zbytkové dráhy, mají se potom zpracovat vyjiskřovací zdvihy a v případě potřeby se má najet na koncovou pozici.
- 16 Jestliže je oscilační pohyb přerušen prostřednictvím vymazání zbytkové dráhy, má se po vypnutí najet do odpovídajícího bodu obratu.
- 32 Změněná hodnota posuvu se uplatní teprve až od následujícího bodu obratu.
- 64 FA se rovná 0, FA = 0: Superpozice dráhy je aktivní.
  FA se nerovná 0, FA <> 0: Superpozice rychlosti je aktivní.
- 128 V případě kruhové osy DC (nejkratší dráha)
- 256 Vyjiskřovací zdvih se uskuteční jako dvojitý zdvih (standardní) 1 = vyjiskřovací zdvih se uskuteční jako jeden zdvih.
- 512 Najíždění napřed na počáteční pozici

`OSNSC` Stanovení počtu vyjiskřovacích zdvihů

`OSE` Definice koncové pozice (ve WCS), na kterou se má po vypnutí oscilačního pohybu najet.

Upozornění:
Jestliže byl naprogramován příkaz `OSE`, pro příkaz `OSCTRL` systém automaticky aktivuje volbu 4.

`OSB` Definice počáteční pozice (ve WCS), na kterou se má před zapnutím oscilačního pohybu najet.

Na počáteční pozici se najíždí před bodem obratu 1. Jestliže se počáteční pozice kryje s polohou bodu obratu 1, jako následující se najíždí na bod obratu 2. Při dosažení počáteční pozice se neuplatňuje žádný doba prodlevy, a to i když se počáteční pozice kryje s bodem obratu 1. Místo toho se čeká na jemné přesné najetí. Nastavená podmínka přesného najetí je dodržena.

Upozornění:
Aby se najelo na počáteční pozici, musí být v nastavovaném parametru SD43770 $SA_OSCILL_CTRL_MASK nastaven bit 9.

Příklady

Příklad 1: Osa vykonávající oscilační pohyb se má pohybovat mezi dvěma body obratu

Osa Z se má pohybovat tam a zpět mezi polohami 10 a 100. Na bod obratu 1 se má najíždět s jemným přesným najetím, na bod obratu 2 se má najíždět s hrubým přesným najetím. Posuv pro oscilační osu má činit 250. Na konci opracování se mají uskutečnit 3 vyjiskřovací zdvihy a potom má oscilační osa odjet na koncovou pozici 200. Posuv pro přísuvnou osu má být 1, konec přísuvu ve směru osy X má být dosažen v poloze 15.

Programový kód	**Komentář**
`WAITP(X,Y,Z)`	`; Výchozí nastavení.`
`G0 X100 Y100 Z100`	`; Přepnutí do režimu polohovací osy.`
`WAITP(X,Z)`	
`OSP1[Z]=10 OSP2[Z]=100`	`; Bod obratu 1, bod obratu 2.`
`OSE[Z]=200`	`; Koncová pozice.`
`OST1[Z]=0 OST2[Z]=-1`	`; Doba pozastavení v bodě U1: jemné přesné ; najetí` `Doba pozastavení v bodě U2: hrubé přesné najetí`
`FA[Z]=250 FA[X]=1`	`; Posuv oscilační osy, posuv přísuvné osy.`
`OSCTRL[Z]=(4,0)`	`; Volby pro aktivování.`
`OSNSC[Z]=3`	`; 3 vyjiskřovací zdvihy`
`OS[Z]=1`	`; Spuštění oscilačního pohybu.`
`WHEN $A_IN[3]==TRUE DO DELDTG(X)`	`; Vymazání zbytkové dráhy.`
`POS[X]=15`	`; Počáteční poloha osy X`
`POS[X]=50`	`Koncová polohy osy X.`
`OS[Z]=0`	`; Zastavení oscilačního pohybu.`
`M30`	

Poznámka

Posloupnost příkazů `OSP1[Z]=...` až `OSNCS[Z]=...` může být naprogramována také v jednom bloku.

Příklad 2: Oscilační pohyb s on-line změnou bodu obratu

Nastavované parametry, které jsou zapotřebí pro asynchronní oscilační pohyby, mohou být nastavovány ve výrobním programu.

Jestliže se do nastavovaných parametrů zapisují hodnoty přímo ve výrobním programu, je změna v platnosti už v okamžiku předběžného zpracování. Synchronního chování lze dosáhnout pomocí zastavení předběžného zpracování (STOPRE).

Programový kód	**Komentář**
`$SA_OSCILL_REVERSE_POS1[Z]=-10`	
`$SA_OSCILL_REVERSE_POS2[Z]=10`	
`G0 X0 Z0`	
`WAITP(Z)`	
`ID=1 WHENEVER $AA_IM[Z] < $$AA_OSCILL_REVERSE_POS1[Z] DO $AA_OVR[X]=0` `ID=2 WHENEVER $AA_IM[Z] < $$AA_OSCILL_REVERSE_POS2[Z] DO $AA_OVR[X]=0`	`; Jestliže dojde k překročení skutečné hodnoty oscilační osy v bodě obratu, přísuvná osa se zastaví.`
`OS[Z]=1 FA[X]=1000 POS[X]=40`	`; Zapnutí oscilačního pohybu.`
`OS[Z]=0`	`; Vypnutí oscilačního pohybu.`
`M30`	

Další informace

Osa vykonávající oscilační pohyb tam a zpět

Pro osu vykonávající pohyb tam a zpět platí:

- Pro oscilační pohyb je možné využít kteroukoli osu.
- Současně může být aktivních i více os vykonávajících oscilační pohyb (maximum: celkový počet polohovacích os).
- Pro oscilační osu je vždy aktivní lineární interpolace G1, a to nezávisle na G-příkazu, který momentálně platí v programu.

Oscilační osa může:

- být vstupní osou pro dynamickou transformaci
- být řídící osou pro osy gantry a vlečné osy
- se pohybovat:
  - bez omezení ryvu (BRISK)

    nebo
  - s omezením ryvu (SOFT)

    nebo
  - s nespojitou charakteristikou zrychlení (jako u polohovacích os)

Body obratu oscilačního pohybu

Při určování poloh pro pohyb tam a zpět je zapotřebí mít na paměti momentálně platná posunutí:

- Zadání absolutních hodnot

  `OSP1[Z]=<hodnota>`

  Poloha bodu obratu = součet posunutí + naprogramovaná hodnota

- Zadání relativních hodnot

  `OSP1[Z]=IC(<hodnota>)`

  Poloha bodu obratu = bod obratu 1 + naprogramovaná hodnota

Příklad:

Programový kód
`N10 OSP1[Z] = 100 OSP2[Z] = 110`
`...`
`...`
`N40 OSP1[Z] = IC(3)`

WAITP

Jestliže má oscilační pohyb vykonávat geometrická osa, musí být pro účely tohoto pohyb uvolněna příkazem `WAITP`.

Po ukončení oscilačního pohybu se pomocí příkazu `WAITP` osa vykonávající oscilační pohyb znovu zaregistruje jako polohovací osa a je možné ji znovu normálně používat.

Oscilační pohyby s pohybovými synchronními akcemi a doby pozastavení

Po uplynutí nastavené doby pozastavení se v případě oscilačního pohybu uskuteční interní přechod na další blok (což lze pozorovat podle nové zbytkové dráhy os). Při přechodu na další blok se kontroluje vypínací funkce. Přitom je podle určeného nastavení řídícího systému pro průběh pohybu (`OSCTRL`) stanovena vypínací funkce. *Toto časové chování může být ovlivněno pomocí korekce (override) posuvu.*

Podle okolností se potom uskuteční ještě jeden oscilační zdvih, než se spustí vyjiskřovací zdvihy nebo než se najede do koncové pozice. *Přitom vzniká dojem, jakoby se změnilo chování při vypínání. Ale není tomu tak.*

11.2 Oscilační pohyby řízené prostřednictvím synchronních akcí (OSCILL)

Funkce

U tohoto druhu oscilačních pohybů je přísuvný pohyb povolen jedině v bodech obratu, příp. v rámci definované oblasti okolo bodu obratu.

V závislosti na požadavcích mohou být oscilační pohyby v průběhu přísuvu

- uskutečňovány nebo
- pozastaveny, dokud není přísuv zcela dokončen.

Syntaxe

1. **Definice parametrů pro oscilační pohyby**
2. **Definice pohybových synchronních akcí**
3. **Přiřazení os, definice přísuvu**

Význam

`OSP1[<oscilační osa>]=`	Poloha bodu obratu 1
`OSP2[<oscilační osa>]=`	Poloha bodu obratu 2
`OST1[<oscilační osa>]=`	Doba prodlevy v bodě obratu 1 v sekundách
`OST2[<oscilační osa>]=`	Doba prodlevy v bodě obratu 2 v sekundách
`FA[<oscilační osa>]=`	Posuv oscilační osy
`OSCTRL[<oscilační osa>]=`	Možnosti aktivování/deaktivování
`OSNSC[<oscilační osa>]=`	Počet vyjiskřovacích zdvihů
`OSE[<oscilační osa>]=`	Koncová pozice
`WAITP(<oscilační osa>)`	Uvolnění osy pro oscilační pohyby

Přiřazení os, přísuv

```
OSCILL[<oscilační osa>]=(<přísuvná osa 1>,<přísuvná osa 2>,<přísuvná osa 3>)
POSP[<přísuvná osa>]=(<koncová pozice>,<dílčí délka>,<režim>)
```

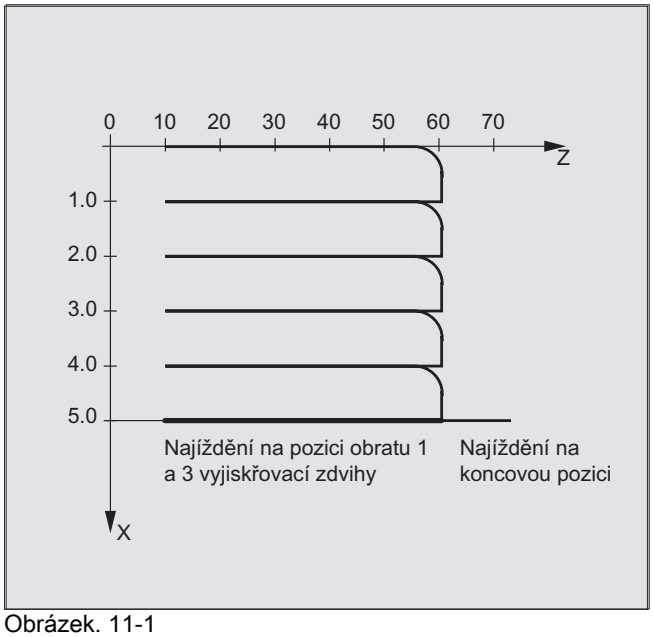
`OSCILL`:	Přiřazení přísuvných os k oscilační ose
`POSP`:	Definice celkového a dílčího přísuvu (viz kapitola správa souborů a programů)
`koncová pozice`:	Koncová pozice pro přísuvnou osu poté, co byly uskutečněny všechny dílčí přísuvy.
`dílčí délka`:	Velikost dílčího přísuvu v bodě obratu/oblasti obratu
`režim`:	Rozdělení celkového přísuvu do dílčích přísuvů = dva stejně velké zbytkové přísuvy (standardní nastavení), = všechny dílčí přísuvy jsou stejně veliké

Pohybové synchronní akce

`WHEN… … DO`	jestliže..., potom...
`WHENEVER … DO`	vždycky když..., potom...

Příklad

V bodě obratu 1 se nemá provádět žádný přísuv. U bodu obratu 2 se má uskutečnit přísuv již ve vzdálenosti ii2 před bodem obratu 2 a oscilační osa v bodě obratu nemá čekat na ukončení dílčího přísuvu. Oscilační osou je osa Z a přísuvnou osou je osa X.



Obrázek. 11-1

1. Parametry pro oscilační pohyby

Programový kód	**Komentář**
`DEF INT ii2`	`; Definice proměnné pro oblast obratu 2`
`OSP1[Z]=10 OSP2[Z]=60`	`; Definice bodu obratu 1 a 2`
`OST1[Z]=0 OST2[Z]=0`	`; Bod obratu 1: Jemné přesné najetí` `Bod obratu 2: Jemné přesné najetí`
`FA[Z]=150 FA[X]=0.5`	`; Posuv oscilační osy Z, posuv přísuvné osy X`
`OSCTRL[Z]=(2+8+16,1)`	`; Oscilační pohyb vypnout v bodě obratu 2, po vymazání zbytkové dráhy vyjiskření a najíždění do koncové polohy, po vymazání zbytkové dráhy najíždění do odpovídajícího bodu obratu`
`OSNC[Z]=3`	`; Vyjiskřovací zdvihy`
`OSE[Z]=70`	`; Koncová pozice = 70`
`ii2=2`	`; Nastavení oblasti obratu`
`WAITP(Z)`	`; Povolení oscilačního pohybu pro osu Z`

2. Pohybové synchronní akce

Programový kód	Komentář
WHENEVER $AA_IM[Z]<$SA_OSCILL_REVERSE_POS2[Z] DO -> $AA_OVR[X]=0 $AC_MARKER[0]=0	; Vždy když je aktuální pozice oscilační osy Z v MCS menší, než je začátek oblasti obratu 2, nastavit axiální korekci (override) přísuvné osy X na 0% a značku s indexem 0 na hodnotu 0.
WHENEVER $AA_IM[Z]>=$SA_OSCILL_REVERSE_POS2[Z] DO $AA_OVR[Z]=0	; Vždy když je aktuální pozice oscilační osy Z v MCS větší nebo rovny poloze bodu obratu 2, nastaví se axiální korekce (override) oscilační osy Z na 0%.
WHENEVER $AA_DTEPW[X]==0 DO $AC_MARKER[0]=1	; Vždy když je zbytková dráha dílčího přísuvu rovna 0, nastavit značku s indexem 0 na hodnotu 1.
WHENEVER $AC_MARKER[0]==1 DO $AA_OVR[X]=0 $AA_OVR[Z]=100	; Vždy když je značka s indexem 0 rovna 1, nastavit axiální korekci (override) přísuvné osy X na 0%. Tím se zabrání příliš brzkému přísuvu (oscilační osa Z ještě oblast obratu 2 znovu neopustila, přísuvná osa X je ale připravena pro nový přísuv). Nastavení axiální korekce (override) oscilační osy Z z 0% (akce 2. synchronní akce) znovu na 100% pro další posuv.

→ musí být naprogramováno v jednom bloku

3. Spuštění oscilačního pohybu

Programový kód	Komentář
OSCILL[Z]=(X) POSP[X]=(5,1,1)	; Spuštění os Oscilační osa Z je přiřazena ose X jako přísuvné ose. Osa X má najíždět až do koncové pozice 5 v krocích po 1.
M30	; Konec programu

Popis

1. **Definice parametrů oscilačního pohybu**
   Před pohybovým blokem, který obsahuje přiřazení přísuvné a oscilační osy, jakož i stanovení přísuvu, je zapotřebí definovat parametry pro oscilační pohyby (viz "Asynchronní oscilační pohyby").
2. **Definice pohybových synchronních akcí**
   Prostřednictvím podmínek synchronních akcí se uskutečňuje:
   Potlačení přísuvu, dokud se oscilační osa nachází uvnitř oblasti obratu (ii1, ii2) nebo v bodě obratu (U1, U2).
   Pozastaveníoscilačního pohybu v průběhu přísuvu v bodě obratu.
   Spuštěníoscilačního pohybu po skončení dílčího přísuvu.
   Stanovení **spuštění následujícího dílčího přísuvu**.
3. **Přiřazení oscilační a přísuvné osy** a definice **celkového a dílčího přísuvu**.

Definice parametrů oscilačního pohybu

Přiřazení oscilační a přísuvné osy: OSCILL

```
OSCILL[oscilační osa] = (přísuvná osa1, přísuvná osa2, přísuvná osa3)
```

Pomocí příkazu `OSCILL` se přiřazují osy a spouští se oscilační pohyb.

Maximálně mohou být oscilační ose přiřazeny 3 přísuvné osy.

Poznámka

Před spuštěním oscilačního pohybu musí být definovány synchronizačním podmínky pro chování os.

Definice přiřazení: POSP

```
POSP[přísuvná osa] = (koncová pozice, dílčí délka, režim)
```

Pomocí příkazu `POSP` je řídícímu systému sdělováno:

- Celkový přísuv (přes koncovou pozici)
- Velikost jednotlivých dílčích přísuvů v bodě obratu, příp. v oblasti obratu.
- Chování dílčího přísuvu při dosažení koncové pozice (pomocí parametru "režim").

Režim = 0	Posledním dvěma dílčím přísuvům je přiřazena taková délka, aby zbývající dráha do cílového bodu byla rozdělena na 2 stejně velké úseky (předdefinované nastavení).
Režim = 1	Všechny dílčí přísuvy jsou stejně veliké. Vypočítají se z celkového přísuvu.

Definice pohybových synchronních akcí

Pohybové synchronní akce popisované v následujících odstavcích se používají zcela obecně pro oscilační pohyby.

Představují vzorové postupy pro řešení jednotlivých požadavků, takže Vám mohou posloužit jako stavební kameny pro sestavování specifických uživatelských oscilačních pohybů.

Poznámka

V jednotlivých případech mohou být synchronizační podmínky naprogramovány i jinak.

Klíčová slova

`WHEN … DO …`	jestliže..., potom...
`WHENEVER … DO`	vždycky když..., potom...

Funkce

Pomocí následujících podrobně popisovaných prostředků NC jazyka můžete realizovat následující funkce:

1. Přísuv v bodě obratu.
2. Přísuv v oblasti obratu.
3. Přísuv v obou bodech obratu.
4. Pozastavení oscilačního pohybu v bodě obratu.
5. Opětovné spuštění oscilačního pohybu.
6. Dílčí přísuv nespouštět příliš brzy.

Pro všechny synchronní akce uváděné v těchto příkladech platí následující předpoklady:

- Bod obratu 1 < bod obratu 2
- Z = oscilační osa
- X = přísuvná osa

Poznámka

Pokud budete potřebovat podrobnější vysvětlení, viz kapitola "Pohybové synchronní akce".

Přiřazení oscilační a přísuvné osy a definice celkového a dílčího přísuvu

Přísuv v oblasti obratu

Přísuvný pohyb by měl začínat v rámci oblasti obratu, předtím než je dosaženo bodu obratu.

Tyto synchronní akce blokují přísuvný pohyb, dokud se oscilační osa nachází v oblasti obratu.

Za uvedených předpokladů (viz výše) z toho vyplývají následující příkazy:

Oblast obratu 1:

```
WHENEVER
$AA_IM[Z]>$SA_OSCILL
_RESERVE_POS1[Z]+ii1
DO $AA_OVR[X] = 0
```

Vždy když je aktuální pozice oscilační osy v MCS větší, než je začátek oblasti obratu 1, nastaví se axiální korekce (override) přísuvné osy na 0%.

Oblast obratu 2:

```
WHENEVER
$AA_IM[Z]<$SA_OSCILL
_RESERVE_POS2[Z]+ii2
DO $AA_OVR[X] = 0
```

Vždy když je aktuální pozice oscilační osy v MCS menší, než je začátek oblasti obratu 2, nastaví se axiální korekce (override) přísuvné osy na 0%.

Přísuv v bodě obratu

Dokud oscilační osa nedosáhla bodu obratu, neuskutečňuje se žádný pohyb v přísuvné ose.

Za uvedených předpokladů (viz výše) z toho vyplývají následující příkazy:

Oblast obratu 1:

```
WHENEVER
$AA_IM[Z]<>$SA_OSCIL
L_RESERVE_POS1[Z] DO
$AA_OVR[X] = 0 →
→ $AA_OVR[Z] = 100
```

Vždy když je aktuální pozice oscilační osy Z v MCS větší nebo menší, než je poloha bodu obratu 1, potom se nastaví axiální korekce (override) přísuvné osy X na 0% a axiální korekce (override) oscilační osy Z na hodnotu 100%.

Oblast obratu 2:

Pro bod obratu 2:

```
WHENEVER
$AA_IM[Z]<>$SA_OSCIL
L_RESERVE_POS2[Z] DO
$AA_OVR[X] = 0 →
→ $AA_OVR[Z] = 100
```

Vždy když je aktuální pozice oscilační osy Z v MCS větší nebo menší, než je poloha bodu obratu 2, potom se nastaví axiální korekce (override) přísuvné osy X na 0% a axiální korekce (override) oscilační osy Z na hodnotu 100%.

Pozastavení oscilačního pohybu v bodě obratu

Oscilační osa se v bodě obratu pozastaví a současně se spustí přísuvný pohyb. Oscilační pohyb bude pokračovat, až když je přísuvný pohyb zcela dokončen.

Současně se může tyto synchronní akce používat k tomu, aby se přísuvný pohyb spouštěl, jestliže byl zastaven předcházející synchronní akcí, která je stále ještě v platnosti.

Za uvedených předpokladů (viz výše) z toho vyplývají následující příkazy:

Oblast obratu 1:

```
WHENEVER
$SA_IM[Z]==$SA_OSCIL
L_RESERVE_POS1[Z] DO
$AA_OVR[X] = 0 →
→ $AA_OVR[Z] = 100
```

Vždy když je aktuální pozice oscilační osy v MCS rovna poloze bodu obratu 1, nastaví se axiální korekce (override) oscilační osy na 0% a axiální korekce (override) přísuvné osy na 100%.

Oblast obratu 2:

```
WHENEVER
$SA_IM[Z]==$SA_OSCIL
L_RESERVE_POS2[Z] DO
$AA_OVR[X] = 0 →
→ $AA_OVR[Z] = 100
```

Vždy když je aktuální pozice oscilační osy Z v MCS rovna poloze bodu obratu 2, nastaví se axiální korekce (override) oscilační osy X na 0% a axiální korekce (override) přísuvné osy na 100%.

On-line vyhodnocování bodu obratu

Jestliže se na pravé straně porovnávaného výrazu nachází proměnná hlavního zpracování označená znaky `$$`, jsou obě proměnné průběžně vyhodnocovány během taktu IPO a vzájemně porovnávány.

Poznámka

Pokud budete potřebovat další související informace, viz kapitola "Pohybové synchronní akce".

Opětovné spuštění oscilačního pohybu

Tyto synchronní akce se použije tehdy, když potřebujete, aby pohyb oscilační osy pokračoval, jakmile je dílčí přísuvný pohyb dokončen.

Za uvedených předpokladů (viz výše) z toho vyplývají následující příkazy:

```
WHENEVER
$AA_DTEPW[X]==0 DO
$AA_OVR[Z] = 100
```

Vždy když je zbytková dráha pro dílčí pohyb přísuvné osy X v MCS rovna nule, potom se nastaví axiální korekce (override) oscilační osy na 100%.

Následující dílčí přísuv

Po dokončení přísuvu je nutno zajistit, aby se následující dílčí přísuv nespustil příliš brzy.

Pro tento účel se používá kanálová značka (`$AC_MARKER[Index]`), která se na konci dílčího přísuvu (dílčí zbytková dráha ≡ 0) nastavuje a která se při opuštění oblasti obratu opět vymaže. Potom se pomocí synchronní akce zablokuje následující přísuvný pohyb.

Za uvedených předpokladů (viz výše) z toho vyplývají např. pro bod obratu 1 následující příkazy:

1. Nastavení značky:

```
WHENEVER
$AA_DTEPW[X]==0 DO
$AC_MARKER[1] = 1
```

Vždy když je zbytková dráha pro dílčí pohyb přísuvné osy X v MCS rovna nule, potom se značka s indexem 1 nastaví na 1.

2. Vymazání značky

```
WHENEVER $AA_IM[Z]<>
$SA_OSCILL_RESERVE_P
OS1[Z] DO
$AC_MARKER[1] = 0
```

Vždy když je aktuální pozice oscilační osy Z v MCS větší nebo menší, než je poloha bodu obratu 1, nastaví se značka 1 na 0.

3. Blokování přísuvu

```
WHENEVER
$AC_MARKER[1]==1 DO
$AA_OVR[X] = 0
```

Vždy když je značka 1 rovna, nastaví se axiální korekce (override) přísuvné osy X na 0%.

Lisování a prostřihování 12

12.1 Aktivování, deaktivování

12.1.1 Zapnutí nebo vypnutí lisování a prostřihování (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC)

Funkce

Aktivování/deaktivování lisování, příp. prostřihování

Lisování, příp. prostřihování se aktivují pomocí příkazů `PON` a `SON`. Příkazem `SPOF` se všechny specifické funkce pro lisování a prostřihování ukončí. Příkazy `PON` a `SON` s modální platností se vzájemně vylučují, tzn. příkaz `PON` deaktivuje příkaz `SON` a naopak.

Lisování/prostřihování s hlavičkou

Funkce pro lisování, příp. prostřihování se spouštějí také pomocí příkazů `SONS` a `PONS`.

Oproti příkazům `SON`/`PON`, u kterých se řízení zdvihu uskutečňuje na úrovni interpolace, probíhá u těchto funkcí řízení spouštění zdvihu na úrovni technických signálů servomechanismů. Díky tomu lze pracovat s vyšší frekvencí zdvihů a tedy i s vyšším výkonem při lisování.

V průběhu vyhodnocování signálů v hlavičce jsou blokovány všechny funkce, které mají za následek změnu polohy lisovacích nebo prostřihovacích os (např. pohyby ručním kolečkem, změny framu pomocí PLC, měřicí funkce).

Lisování se zpožděním

Funkce `PDELAYON` způsobuje zpožděné spuštění lisovacího zdvihu. Tento příkaz s modální platností má přípravnou funkci a zpravidla se nachází před příkazem `PON`. Po příkazu `PDELAYOF` probíhá normální další lisování.

Poznámka

Doba prodlevy se nastavuje pomocí nastavovaného parametru SD42400 $SC_PUNCH_DWELLTIME.

Zrychlení závisející na dráze

Pomocí příkazu `PUNCHACC` může být definována charakteristika zrychlení, která definuje různá zrychlení v závislosti na vzdálenosti díry.

Druhé rozhraní pro lisování

Stroje, které mají střídavě využívat druhé stanoviště s rozhraním (druhá lisovací jednotka nebo srovnatelné médium) mohou být přepínány na druhou dvojici rychlých digitálních vstupů a výstupů řídícího systému (V/V-pár). Obě stanoviště s rozhraním mohou využívat plné spektrum funkcí pro lisování/prostřihování. Pro přepínání mezi prvním a druhým stanovištěm s rozhraním slouží příkazy `SPIF1` a `SPIF2`.

Poznámka

Předpoklad: Prostřednictvím strojních parametrů musí být druhý pár V/V definován pro lisovací funkce (→ viz informace od výrobce stroje!).

Syntaxe

```
PON G... X... Y... Z...
SON G... X... Y... Z...
SONS G... X... Y... Z...
PONS G... X... Y... Z...
PDELAYON
PDELAYOF
PUNCHACC(<Smin>,<Amin>,<Smax>,<Amax>)
SPIF1/SPIF2
SPOF
```

Význam

PON	Aktivování lisování
SON	Aktivování prostřihování
PONS	Aktivování lisování s hlavičkou
SONS	Aktivování prostřihování s hlavičkou
SPOF	Deaktivování lisování/prostřihování
PDELAYON	Aktivování lisování se zpožděním
PDELAYOF	Deaktivování lisování se zpožděním
PUNCHACC	Aktivování zrychlení závisejícího na dráze Parametry: <Smin> Nejmenší vzdálenost od díry <Amin> Počáteční zrychlení Parametr <Amin> může být větší než <Amax>. <Smax> Největší vzdálenost od díry <Amax> Koncové zrychlení Parametr <Amax> může být větší než <Amin>.
SPIF1	Aktivování **prvního** rozhraní pro lisování Řízení zdvihu se uskutečňuje prostřednictvím první dvojice rychlých V/V.
SPIF2	Aktivování **druhého** rozhraní pro lisování Řízení zdvihu se uskutečňuje prostřednictvím druhé dvojice rychlých V/V. **Upozornění:** Po resetu nebo po náběhu řídícího systému je aktivní vždy první rozhraní. Jestliže se pro lisování používá jen jedno rozhraní, nemusí být naprogramováno.

Příklady

Příklad 1: Aktivování prostřihování

Programový kód	Komentář
...	
N70 X50 SPOF	; Najíždění na polohu bez spuštění lisování.
N80 X100 SON	; Aktivování prostřihování, spouštění zdvihu před pohybem (X=50) a na konci naprogramovaného pohybu (X=100).
...	

Příklad 2: Lisování se zpožděním

Programový kód	Komentář
...	
N170 PDELAYON X100 SPOF	; Najíždění na polohu bez spuštění lisování, aktivování zpožděného spuštění lisování.
N180 X800 PON	; Aktivování lisování. Po dosažení koncové pozice se uskuteční zpožděný lisovací zdvih.
N190 PDELAYOF X700	; Deaktivování lisování se zpožděním, normální spouštění lisování na konci naprogramovaného pohybu.
...	

Příklad 3: Lisování se dvěma stanovišti s rozhraním

Programový kód	Komentář
...	
N170 SPIF1 X100 PON	; Na konci bloku se uskutečňuje spuštění zdvihu na prvním rychlém výstupu. Na prvním rychlém vstupu je monitorován signál "Zdvih aktivní".
N180 X800 SPIF2	; Druhý zdvih se spouští na druhém rychlém výstupu. Na druhém rychlém vstupu je monitorován signál "Zdvih aktivní".
N190 SPIF1 X700	; Řízení zdvihu se pro všechny další zdvihy uskutečňuje na prvním rozhraní.
...	

Další informace

Lisování a prostřihování s hlavičkou (PONS/SONS)

Lisování a prostřihování s hlavičkou není možné provádět ve více kanálech současně. Příkazy `PONS`, příp. `SONS` mohou být aktivovány vždy jen v jednom kanálu.

Aktivování zrychlení závisejícího na dráze (PUNCHACC)

Příklad:
```
PUNCHACC(2,50,10,100)
```

Vzdálenost od díry menší než 2 mm:

Pohyb bude probíhat se zrychlením 50% maximálního zrychlení.

Vzdálenost od díry 2 mm až 10 mm:

Zrychlení se bude přímo úměrně vzdálenosti zvyšovat až na 100%.

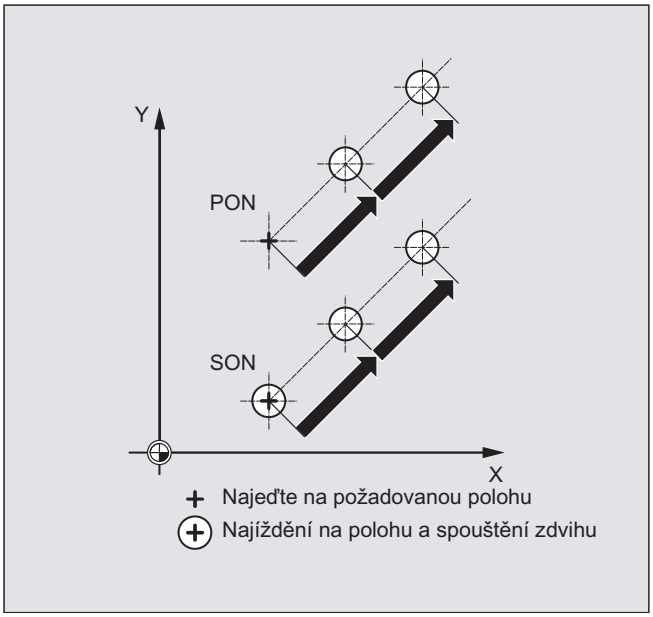
Vzdálenost od díry větší než 10 mm:

Pohyb se rychlením 100%.

Spouštění prvního zdvihu

Spouštění prvního zdvihu po aktivování funkce se uskutečňuje při prostřihování časově odlišně než při lisování.

- `PON/PONS`:
  - Všechny zdvihy - i zdvih prvního bloku po aktivování - probíhají na konci bloku.
- `SON/SONS`:
  - První zdvih po aktivování prostřihování se uskuteční už na začátku bloku.
  - Všechny ostatní zdvihy jsou však spouštěny až na konci bloku.



Lisování a prostřihování na místě

Ke spuštění zdvihu dojde jen tehdy, pokud blok obsahuje informace o pohybu lisovacích nebo prostřihovacích os (osy aktivní roviny).

Aby se přesto spustil zdvih na stejném místě, naprogramují se lisovací/prostřihovací osy s délkou dráhy 0.

Práce s otočnými nástroji

Poznámka

Aby byly otočné nástroje nastavovány tangenciálně vůči naprogramované dráze, používejte tangenciální řízení.

Použití příkazů M-funkcí

Pomocí techniky maker je stejně jako dříve možné používat speciální M-funkce namísto příkazů NC jazyka (kompatibilita). Vzhledem ke starším systémům přitom platí následující konvence:

```
M20, M23   ≙  SPOF
M22        ≙  SON
M25        ≙  PON
M26        ≙  PDELAYON
```

Příklad souboru makra:

Programový kód	Komentář
`DEFINE M25 AS PON`	`; Lisování aktivováno`
`DEFINE M125 AS PONS`	`; Aktivování lisování s hlavičkou`
`DEFINE M22 AS SON`	`; Prostřihování aktivováno`
`DEFINE M122 AS SONS`	`; Aktivování prostřihování s hlavičkou`
`DEFINE M26 AS PDELAYON`	`; Aktivování lisování se zpožděním`
`DEFINE M20 AS SPOF`	`; Lisování, prostřihování deaktivovat`
`DEFINE M23 AS SPOF`	`; Lisování, prostřihování deaktivovat`

Příklad programování:

Programový kód	Komentář
`...`	
`N100 X100 M20`	`; Najíždění na polohu bez spuštění lisování.`
`N110 X120 M22`	`; Aktivování prostřihování, spouštění zdvihu před a po pohybu`
`N120 X150 Y150 M25`	`; Aktivování lisování, spouštění zdvihu na konci pohybu.`
`...`	

12.2 Automatická příprava cesty

Funkce

Rozčlenění na dílčí úseky

Když je aktivní lisování, příp. prostřihování, zajišťují příkazy SPP a SPN rozčlenění celkové dráhy posuvu, která byla naprogramována pro dráhové osy, na daný počet stejně dlouhých dílčích úseků (ekvidistantní rozdělení dráhy). Interně odpovídá každému dílčímu úseku jeden blok.

Počet zdvihů

Při lisování se první zdvih uskuteční na konci prvního dílčího úseku, při prostřihování oproti tomu v počátečním bodě prvního dílčího úseku. V důsledku toho na celkové dráze platí následující počty:

Lisování: Počet zdvihů = počet dílčích úseků

Prostřihování: Počet zdvihů = počet dílčích úseků + 1

Pomocné funkce

Pomocné funkce jsou zpracovávány v prvním vygenerovaném úseku.

Syntaxe

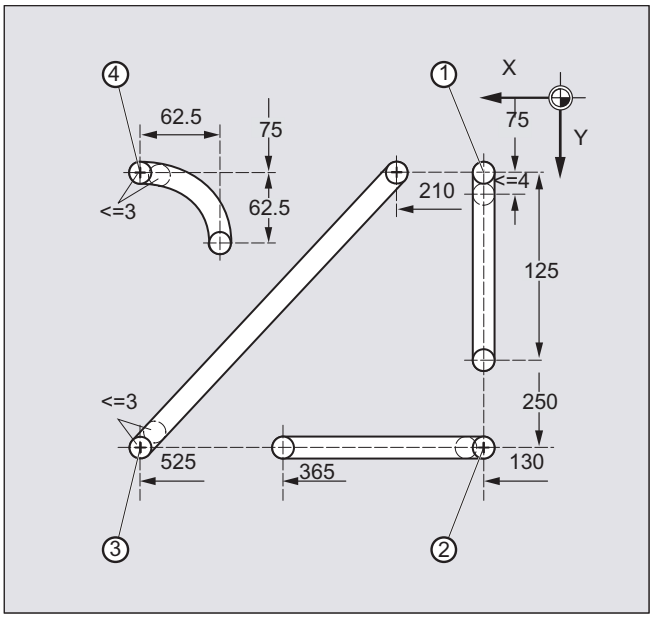
```
SPP=
SPN=
```

Význam

SPP	Velikost dílčího úseku (maximální vzdálenost zdvihů), modální platnost
SPN	Počet úseků na blok, bloková platnost

Příklad 1

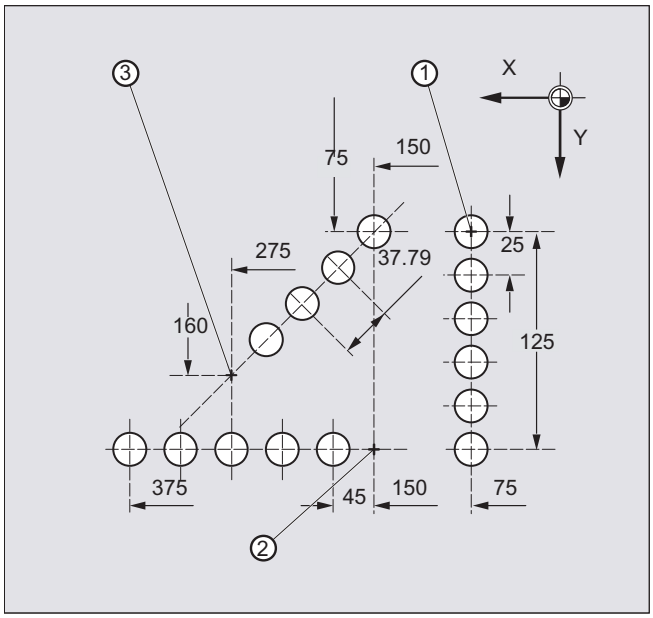
Naprogramované úseky pro prostřihování mají být automaticky rozděleny na stejně veliké úseky.



Programový kód	**Komentář**
N100 G90 X130 Y75 F60 SPOF	; Najíždění na polohu počátečního bodu 1
N110 G91 Y125 SPP=4 SON	; Aktivování prostřihování, maximální délka dílčího úseku pro automatické rozdělení dráhy: 4 mm
N120 G90 Y250 SPOF	; Vypnutí prostřihování, najíždění na pozici počátečního bodu 2
N130 X365 SON	; Aktivování prostřihování, maximální délka dílčího úseku pro automatické rozdělení dráhy: 4 mm
N140 X525 SPOF	; Vypnutí prostřihování, najíždění na pozici počátečního bodu 3
N150 X210 Y75 SPP=3 SON	; Aktivování prostřihování, maximální délka dílčího úseku pro automatické rozdělení dráhy: 3 mm
N160 X525 SPOF	; Vypnutí prostřihování, najíždění na pozici počátečního bodu 4
N170 G02 X-62.5 Y62.5 I J62.5 SPP=3 SON	; Aktivování prostřihování, maximální délka dílčího úseku pro automatické rozdělení dráhy: 3 mm
N180 G00 G90 Y300 SPOF	; Vypnutí prostřihování

Příklad 2

Pro jednotlivé řady děr má být provedeno automatické rozdělení dráhy na úseky. Pro rozdělení se vždy zadává maximální délka úseku dráhy (hodnota SPP).



Programový kód	**Komentář**
N100 G90 X75 Y75 F60 PON	; Najíždění na počáteční bod 1, lisování jednotlivé díry
N110 G91 Y125 SPP=25	; Maximální délka dílčího úseku pro automatické rozdělení dráhy: 25 mm
N120 G90 X150 SPOF	; Vypnutí lisování, najíždění na pozici počátečního bodu 2
N130 X375 SPP=45 PON	; Aktivování lisování, maximální délka dílčího úseku pro automatické rozdělení dráhy: 45 mm
N140 X275 Y160 SPOF	; Vypnutí lisování, najíždění na pozici počátečního bodu 3
N150 X150 Y75 SPP=40 PON	; Aktivování lisování, namísto naprogramované délky dílčího úseku 40 mm se ;použije vypočítaná délka dílčího úseku 37,79 mm.
N160 G00 Y300 SPOF	; Vypnutí lisování, najíždění na danou pozici

12.2.1 Rozdělení cesty u dráhových os

Délka dílčího úseku SPP

Pomocí příkazu `SPP` zadáváte maximální vzdálenost zdvihů a v důsledku toho maximální délku dílčího úseku, na které má být celková dráha rozdělena. Pro deaktivování této funkce slouží příkazy `SPOF` nebo `SPP=0`.

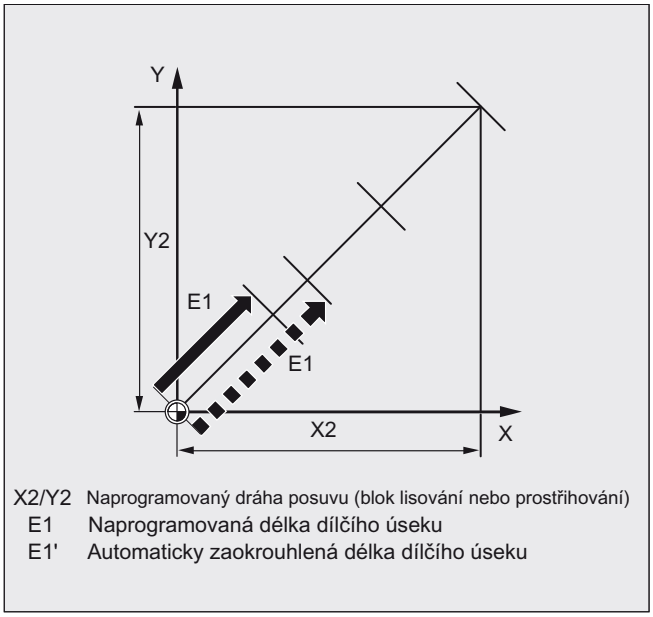
Příklad:

```
N10 SON X0 Y0
N20 SPP=2 X10
```

Celková dráha 10 mm se rozdělí na 5 dílčích úseků každý o délce 2 mm (`SPP=2`).

Poznámka

Rozdělení dráhy pomocí příkazu `SPP` je vždy ekvidistantní: všechny dílčí úseky jsou stejně dlouhé. To znamená, že naprogramovaná velikost dílčího úseku (hodnota `SPP`) platí jen tehdy, pokud je podíl celkové délky dráhy a hodnoty SPP celé číslo. Jestliže tomu tak není, interně se velikost dílčího úseku zmenší natolik, aby byl podíl celé číslo.



X2/Y2 Naprogramovaný dráha posuvu (blok lisování nebo prostřihování)
E1 Naprogramovaná délka dílčího úseku
E1' Automaticky zaokrouhlená délka dílčího úseku

Příklad:

```
N10 G1 G91 SON X10 Y10
N20 SPP=3.5 X15 Y15
```

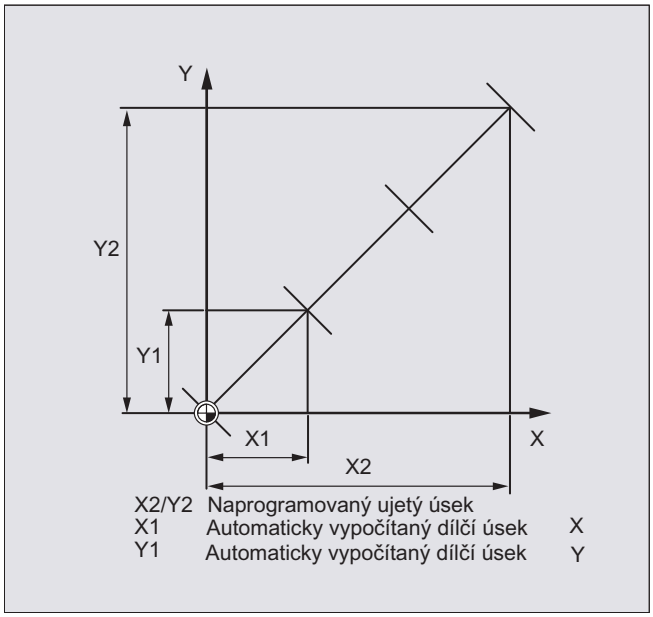
V případě délky celé dráhy 15 mm a délky dílčích úseků 3,5 mm vyplývá hodnota podílu, která není celočíselná (4,28). V důsledku toho systém hodnotu SPP zmenší, aby vyšel nejbližší možný celočíselný podíl. V tomto případě vyjde délka dílčího úseku 3 mm.

Počet dílčích úseků SPN

Pomocí příkazu SPN definujete počet dílčích úseků, které mají být z celkové délky dráhy vytvořeny. Délka dílčího úseku se vypočítá automaticky. Protože příkaz SPN má blokovou platnost, musí být předtím pomocí příkazů PON nebo SON aktivováno lisování nebo prostřihování.

Příkazy SPP a SPN ve stejném bloku

Jestliže v jednom bloku naprogramujete jak délku dílčího úseku (SPP), tak také počet dílčích úseků (SPN), potom bude v tomto bloku platit příkaz SPN a pro všechny následující bloky příkaz SPP. Jestliže byl příkaz SPP aktivován už před příkazem SPN, bude po bloku s příkazem SPN znovu aktivní.



Poznámka

Jestliže je lisování/prostřihování v zásadě v řídícím systému k dispozici, může být programování automatického rozdělení dráhy pomocí příkazů SPN, příp. SPP aktivováno i nezávisle na této technologii.

12.2.2 Rozdělení cesty u jednotlivých os

Jestliže jsou vedle dráhových os také samostatné osy definovány jako osy pro lisování/ prostřihování, je možné i na tyto osy aplikovat automatické rozdělení dráhy.

Chování samostatné osy v případě SPP

Naprogramovaná délka dílčího úseku (`SPP`) se v zásadě vztahuje na dráhové osy. Proto je v bloku, ve kterém kromě pohybu samostatnou osou a hodnoty příkazu SPP není naprogramována žádná dráhová osa, je hodnota SPP ignorována.

Jestliže jsou v bloku naprogramovány jak samostatné osy, tak i dráhové osy, řídí se chování samostatné osy podle nastavení odpovídajícího strojního parametru.

1. Standardní nastavení

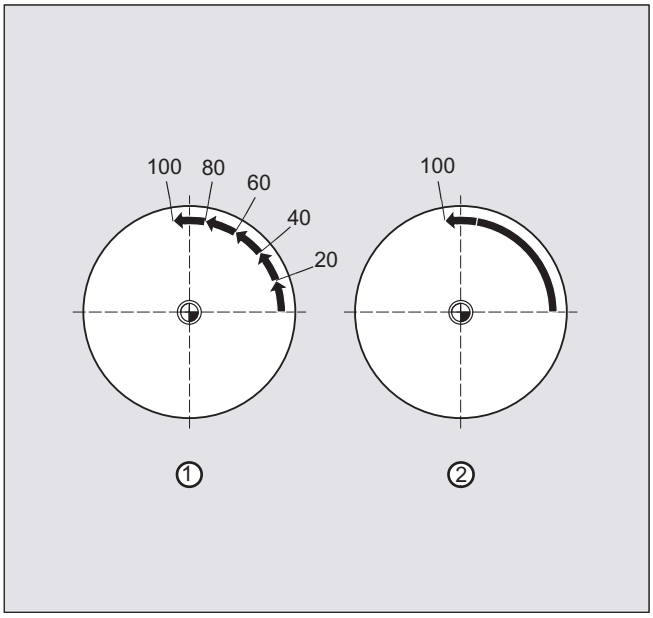
   Dráha samostatné osy je prostřednictvím pomocných bloků vygenerovaných příkazem `SPP` rozdělena na stejné úseky.

Příklad:

```
N10 G1 SON X10 A0
N20 SPP=3 X25 A100
```

Na základě zdvihové vzdálenosti 3 mm se v případě celkové délky posuvu osy X (dráhová osa) 15 mm vytvoří 5 bloků.

Osa A se tedy v každém bloku otočí o 20°.



1. Samostatná osa bez rozdělení dráhy

   Samostatná osa urazí svou celkovou dráhu v rámci prvního z vytvořených bloků.

2. Odlišné rozdělení dráhy

   Chování samostatné osy je závislé na interpolaci dráhových os.

- Kruhová interpolace: Rozdělení dráhy
- Lineární interpolace: Žádné rozdělení dráhy

Chování v případě SPN

Naprogramovaný počet dílčích úseků platí také tehdy, pokud dráhová osa není naprogramována současně.

Předpoklad: Samostatná osa je definována jako osa pro prostřihování/lisování.

Broušení 13

13.1 Specifické monitorování nástroje pro broušení ve výrobním programu (TMON, TMOF)

Funkce

Příkazem TMON můžete pro brusné nástroje (typ 400 – 499) v NC programu aktivovat monitorování geometrie a otáček. Monitorování zůstává aktivní, dokud není ve výrobním programu deaktivováno příkazem TMOF.

Poznámka

Věnujte prosím pozornost informacím od výrobce stroje!

Předpoklady

Specifické parametry brusných nástrojů $TC_TPG1 až $TC_TPG9 musí být nastaveny.

Syntaxe

```
TMON(<T-číslo>)
TMOF(<T-číslo>)
```

Význam

TMON	Příkaz pro **zapnutí** specifického monitorování brusných nástrojů
TMOF	Příkaz pro **vypnutí** specifického monitorování brusných nástrojů
<T-číslo>	Zadání T-čísla **Upozornění:** Je potřebné jen tehdy, pokud nástroj s tímto číslem není aktivní.
TMOF(0)	Deaktivování monitorování pro všechny nástroje

Další informace

Specifické parametry brusného nástroje

Parametry	Význam	Datový typ
$TC_TPG1	Číslo vřetena	INT
$TC_TPG2	Pravidlo pro zřetězení Parametry jsou automaticky udržovány identické pro levou a pravou stranu kotouče.	INT
$TC_TPG3	Minimální rádius kotouče	REAL
$TC_TPG4	Minimální šířka kotouče	REAL
$TC_TPG5	Momentální šířka kotouče	REAL
$TC_TPG6	Maximální otáčky	REAL
$TC_TPG7	Maximální obvodová rychlost	REAL
$TC_TPG8	Úhel šikmého kotouče	REAL
$TC_TPG9	Číslo parametru pro výpočet rádiusu	INT

Literatura:
Příručka Popis funkcí, Základní funkce; "Korekční parametry nástrojů (W1)

Aktivování monitorování nástroje prostřednictvím jeho zvolení

V závislosti na nastavení strojního parametru je možné pro brusné nástroje (typ 400 – 499) zapínat jejich monitorování implicitně spolu s jejich vyvoláváním.

V každém okamžiku může být pro každé vřeteno aktivní jen **jedno** monitorování.

Monitorování geometrie

Monitorovány jsou aktuální rádius kotouče a jeho aktuální šířka.

Sledování požadované hodnoty otáček, zda nedošlo k překročení maximální hodnoty, se provádí cyklicky a bere se přitom v úvahu korekce (override) vřetena.

Jako mezní hodnota otáček platí menší hodnota, která vyplyne při porovnávání maximálních otáček s otáčkami vypočítanými na základě maximální obvodové rychlosti kotouče a jeho aktuálního rádiusu.

Práce bez T-čísla a D-čísla

Na jeden strojní parametr mohou být nastaveny jedno standardní **T**-číslo a standardní **D**-číslo,

které pak už není nutné programovat a které je aktivní po zapnutí a po resetu.

Příklad: Práce se stále stejným kotoučem

Pomocí strojního parametru lze nastavit, aby při resetu zůstal aktivní nástroj zachován (viz "Volné zadávání D-čísel, číslo břitu (adresa CE) [Strana 439]").

Další funkce 14

14.1 Osové funkce (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL)

Funkce

Příkaz AXNAME se používá např. při sestavování všeobecně platných cyklů, pokud názvy os nejsou známy.

Příkaz AX se používá pro nepřímé programování geometrických a synchronních os. Identifikátor osy je přitom uložen v proměnné typu AXIS nebo je zjištěn pomocí příkazu, jako např. AXNAME nebo SPI.

Příkaz SPI se použije, jestliže se programují funkce osy pro vřeteno, např. pro synchronní vřeteno.

Příkaz AXTOSPI slouží v situacích, kdy potřebujete identifikátor osy převést na index vřetena (inverzní funkce k SPI).

Příkaz AXSTRING slouží v situacích, kdy potřebujete identifikátor osy (datový typ AXIS) převést na řetězec (inverzní funkce k AXNAME).

Příkaz ISAXIS se používá ve všeobecně platných cyklech, aby se zjistilo, že určitá geometrická osa je k dispozici a aby následující volání proměnné $P_AXNX nebylo přerušeno s chybou.

Funkce MODAXVAL se používá pro zjišťování polohy kruhových os typu Modulo po zpracování funkcí Modulo.

Syntaxe

```
AXNAME("řetězec")
AX[AXNAME("řetězec")]
SPI (n)
```

AXTOSPI(A) nebo AXTOSPI(B) nebo AXTOSPI(C)

```
AXSTRING(SPI(n))
ISAXIS(<číslo geometrické osy>)
<pozice modulo>=MODAXVAL(<osa>,<poloha osy>)
```

Význam

AXNAME	Převádí vstupní řetězec na identifikátor osy, vstupní řetězec musí obsahovat platný název osy.
AX	Proměnný identifikátor osy
SPI	Převádí číslo vřetena na identifikátor osy; předávaný parametr musí obsahovat platné číslo vřetena.
n	Číslo vřetena
AXTOSPI	Převádí identifikátor osy na index vřetena typu INTEGER. Příkaz AXTOSPI odpovídá inverzní funkci k SPI.
X, Y, Z	Identifikátor osy typu AXIS jako proměnná nebo jako konstanta.
AXSTRING	Na výstupu bude řetězec, jemuž je přiřazeno číslo vřetena.
ISAXIS	Zkontroluje, zda je uvedená geometrická osa k dispozici.
MODAXVAL	Zjišťuje polohu kruhových os typu Modulo po zpracování funkcí Modulo, tato hodnota odpovídá zbytku po celočíselném dělení vzhledem k rozsahu Modulo (v případě standardního nastavení je to 0° až 360°, pomocí strojních parametrů MD30340 MODULO_RANGE_START a MD30330 $MA_MODULO_RANGE mohou být začátek s velikost rozsahu Modulo změněny).

Poznámka

Rozšíření SPI

Osová funkce SPI(n) m§že být použita také pro čtení a zápis komponentů framu. Tímto způsobem je možné zapisovat údaje do framu, např. pomocí syntaxe `$P_PFRAME[SPI(1),TR]=2.22`.

Prostřednictvím doplňkového naprogramování pozice os pomocí adresy `AX[SPI(1)]=<pozice osy>` je možné osou pohybovat. Předpokladem pro tuto operaci ale je, že se vřeteno nachází v režimu polohování nebo v režimu osy.

Příklady

Příklad 1: AXNAME, AX, ISAXIS

```
Programový kód                                  Komentář
OVRA[AXNAME("Planachse")]=10                    ; Korekce (override) pro příčnou osu
AX[AXNAME("Planachse")]=50.2                    ; Koncová pozice pro příčnou osu
OVRA[SPI(1)]=70                                 ; Korekce (override) pro vřeteno 1
AX[SPI(1)]=180                                  ; Koncová pozice pro vřeteno 1
IF ISAXIS(1) == FALSE GOTOF WEITER              ; Je k dispozici abscisa?
AX[$P_AXN1]=100                                 ; Pohyb abscisy
WEITER:
```

Příklad 2: AXSTRING

Při programování příkazu AXSTRING[SPI(n)] se nezadává index osy pro osu, která je přiřazena vřetenu, jako číslo vřetena, nýbrž místo toho se zadává řetězec "Sn".

Programový kód	Komentář
AXSTRING[SPI(2)]	; Na výstupu bude řetězec "S2".

Příklad 3: MODAXVAL

Tímto způsobem se má zjistit poloha kruhové osy A typu Modulo po zpracování funkcí modulo.

Výchozí hodnota pro výpočet je poloha osy 372,55.

Rozsah Modulo nastavený v parametrech je 0 až 360 stupňů:

MD30340 MODULO_RANGE_START = 0

MD30330 $MA_MODULO_RANGE = 360

Programový kód	Komentář
R10=MODAXVAL(A,372.55)	; Vypočítaná pozice Modulo je R10 = 12,55.

Příklad 4: MODAXVAL

Jestliže se naprogramovaný identifikátor osy nevztahuje na kruhovou osu typu Modulo, potom bude převáděná hodnota (<poloha osy>) vrácena zpět nezměněná.

Programový kód	Komentář
R11=MODAXVAL(X,372.55)	; X je lineární osa; R11 = 372.55.

14.2 Přepínatelné geometrické osy (GEOAX)

Funkce

Pomocí funkce „Přepínatelné geometrické osy“ je možné výrobním programem změnit prostřednictvím strojních parametrů nastavené přiřazení geometrických os. Přitom může být libovolná geometrická osa nahrazena kanálovou osou definovanou jako pomocná synchronní osa.

Syntaxe

```
GEOAX(<n>,<kanálová osa>,<n>,<kanálová osa>,<n>,<kanálová osa>)
GEOAX()
```

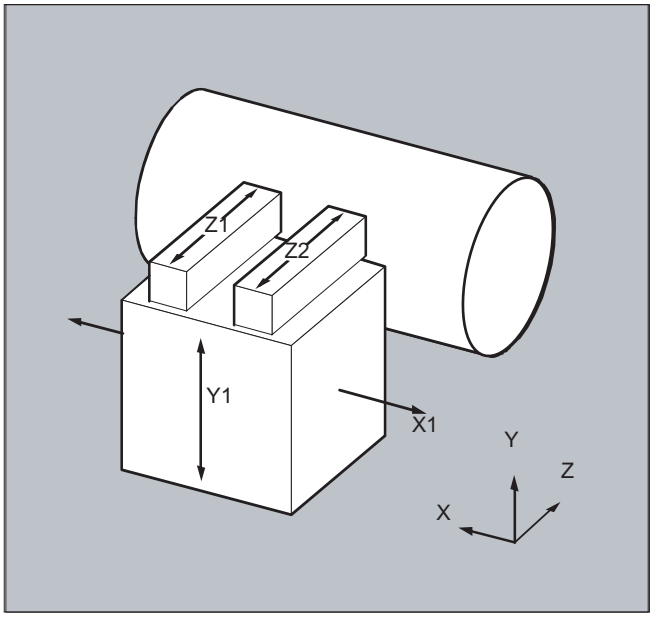
Význam

`GEOAX(...)`	Příkaz pro přepínání geometrických os **Upozornění:** Příkaz `GEOAX()` bez zadání parametrů vyvolá základní konfiguraci geometrických os.
`<n>`	Pomocí tohoto parametru se zadává číslo geometrické osy, která má být přiřazena následně uvedené kanálové ose. Rozsah hodnot: 1, 2 nebo 3 **Upozornění:** Pokud platí <n>=0, může být následně uvedená kanálová osa bez náhrady odstraněna ze svazku geometrických os.
`<kanálová osa>`	Prostřednictvím tohoto parametru se zadává název kanálové osy, která má být přijata do svazku geometrických os.

Příklady

Příklad 1: Střídavé přepínání dvou os na místě jedné geometrické osy

Nástrojovými saněmi je možné pohybovat pomocí kanálových os X1, Y1, Z1 a Z2.



Konfigurace geometrických os je taková, že po zapnutí systému se napřed jako platnou 3. geometrickou osou pod názvem geometrické osy "Z" stává osa Z1 a spolu s osami X1 a Y1 vytváří svazek geometrických os.

Ve výrobním programu se nyní mají osy Z1 a Z2 střídavě používat jako geometrická osa Z.

Programový kód	Komentář
...	
N100 GEOAX(3,Z2)	; Jako 3. geometrická osa (Z) je konfigurována kanálová osa Z2.
N110 G1 ...	
N120 GEOAX(3,Z1)	; Jako 3. geometrická osa (Z) je konfigurována kanálová osa Z1.
...	

Příklad 2: Přepínání geometrických os v případě 6 kanálových os

Stroj je vybaven 6 kanálovými osami s názvy XX, YY, ZZ, U, V, W.

Základní nastavení konfigurace geometrických os prostřednictvím strojních parametrů je:

Kanálová osa XX = 1. geometrická osa (osa X)

Kanálová osa YY = 2. geometrická osa (osa Y)

Kanálová osa ZZ = 3. geometrická osa (osa Z)

Programový kód	Komentář
N10 GEOAX()	; Základní konfigurace geometrických je v platnosti.
N20 G0 X0 Y0 Z0 U0 V0 W0	; Všechny osy rychlým posuvem na pozici 0.
N30 GEOAX(1,U,2,V,3,W)	; Přiřazení kanálové osy U první (X), osy V druhé (Y) a osy W třetí geometrické ose (Z).
N40 GEOAX(1,XX,3,ZZ)	; Přiřazení kanálové osy XX první (X) a osy ZZ třetí geometrické ose (Z). Kanálová osa V zůstává druhou geometrickou osou (Y).
N50 G17 G2 X20 I10 F1000	; Plná kružnice v rovině X/Y. Pohybují se kanálové osy XX a V.
N60 GEOAX(2,W)	; Kanálová osa W se stává druhou geometrickou osou (Y).
N80 G17 G2 X20 I10 F1000	; Plná kružnice v rovině X/Y. Pohybují se kanálové osy XX a W.
N90 GEOAX()	; Obnovení původního stavu.
N100 GEOAX(1,U,2,V,3,W)	; Přiřazení kanálové osy U první (X), osy V druhé (Y) a osy W třetí geometrické ose (Z).
N110 G1 X10 Y10 Z10 XX=25	; Kanálové osy U, V a W najíždějí na své pozice 10. Osa XX jako přísuvná osa najíždí na pozici 25.
N120 GEOAX(0,V)	; Osa V je vyjmuta ze svazku geometrických os. Osy U a W jsou nadále první (X) a třetí geometrickou osou (Z). Druhá geometrická osa (Y) zůstává neobsazena.
N130 GEOAX(1,U,2,V,3,W)	; Kanálová osa U zůstává přiřazena první (X), osa V druhé (Y) a osa W třetí geometrické ose (Z).
N140 GEOAX(3,V)	; Osa V se stává třetí geometrickou osou (Z), přičemž přepisuje osu W a tím pádem ji vyjímá ze svazku geometrických os. Druhá geometrická osa (Y) je stejně jako předtím neobsazena.

Poznámka

Konfigurace osy

Přiřazení mezi geometrickými osami, doplňkovými osami, kanálovými osami a osami stroje, stejně jako definice názvů os jednotlivých typů, jsou definovány prostřednictvím následujících strojních parametrů:

MD20050 $MC_AXCONF_GEOAX_ASIGN_TAB (přiřazení geometrických os kanálovým osám)

MD20060 $MC_AXCONF_GEOAX_NAME_TAB (název geometrické osy v kanálu)

MD20070 $MC_AXCONF_MACHAX_USED (číslo osy stroje je v kanálu platné)

MD20080 $MC_AXCONF_CHANAX_NAME_TAB (název kanálové osy v kanálu)

MD10000 $MN_AXCONF_MACHAX_NAME_TAB (název osy stroje)

MD35000 $MA_SPIND_ASSIGN_TO_MACHAX (přiřazení vřetena ose stroje)

Literatura:
Příručka Popis funkcí, Základní funkce; "Osy, souřadné systémy, framy" (K2)

Omezení

- Přepínání geometrických os není možné za následujících okolností:
  - je aktivní transformace
  - je aktivní splinová interpolace
  - je aktivní korekce rádiusu nástroje
  - je aktivní jemná korekce nástroje
- Jestliže mají geometrická osa a kanálová osa stejný název, výměna příslušné geometrické osy není možná.
- Žádná z os podílejících se na přepnutí se nesmí podílet na akci, které může trvat i přes hranice bloku, což se může stát např. u polohovacích os typu A nebo v případě vlečných os.
- Pomocí příkazu `GEOAX` mohou být nahrazovány jedině geometrické osy, které existovaly už při zapnutí (tedy žádné nově definované osy).
- Výměna os pomocí příkazu `GEOAX` v době, kdy probíhá příprava tabulky kontury (`CONTPRON`, `CONTDCON`), má za následek alarm.

Okrajové podmínky

Stav os po nahrazení

Osa, která byla ve svazku geometrických os nahrazena pomocí přepnutí, může být po přepnutí naprogramována jako doplňková osa pomocí svého kanálového názvu.

Framy, chráněné oblasti, ohraničení pracovního pole

Přepnutím geometrických os jsou všechny framy, chráněné oblasti a ohraničení pracovního pole vymazány.

Polární souřadnice

Výměna geometrických os pomocí příkazu `GEOAX` nastavuje analogicky se změnou roviny pomocí příkazů `G17-G19` modální polární souřadnice na hodnotu 0.

DRF, NPV

Případná posunutí pomocí ručního kolečka (DRF) nebo externí posunutí počátku (NPV) zůstávají i po přepnutí v platnosti.

Základní konfigurace geometrických os

Příkaz `GEOAX()` vyvolá základní konfiguraci svazku geometrických os.

Po náběhu systému (Power-On) a při přepnutí do provozního režimu "Najíždění na referenční bod" se automaticky nastavuje základní konfigurace.

Korekce délky nástroje

Aktivní korekce délky nástroje je v platnosti i po operaci přepnutí. Platí však pro geometrické osy, které byly nově připojeny, příp. které si vyměnili pozici, jako hodnota, na kterou se zatím nenajelo. Při prvním pohybovém příkazu pro tyto geometrické osy vznikne výsledná dráha posuvu, která je odpovídajícím způsobem složena z korekce délky nástroje a naprogramované dráhy posuvu.

Geometrické osy, které při přepnutí své pozice zůstaly ve svazku os zachovány, si ponechávají i svůj stav týkající se korekce délky nástroje.

Konfigurace geometrických os při aktivní transformaci

Konfigurace geometrických os (definována pomocí strojních parametrů), která platí v aktivní transformaci, nemůže být pomocí funkce "Přepínatelné geometrické osy" změněna.

Jestliže se vyskytne potřeba v souvislosti s transformacemi změnit konfiguraci geometrických os, je to možné uskutečnit jedině další transformací.

Konfigurace geometrických os změněná pomocí funkce `GEOAX` se aktivováním transformace vymaže.

Jestliže si nastavení ve strojních parametrech pro transformaci a pro přepínání geometrických os odporují, mají nastavení pro transformaci přednost.

Příklad:

Je aktivní transformace. Podle nastavení strojních parametrů má při resetu zůstat transformace zachována, současně však ale při resetu má být obnovena původní konfigurace geometrických os. V tomto případě zůstane zachována konfigurace geometrických os, která byla definována pomocí transformace.

14.3 Osový zásobník (AXCTSWE, AXCTSWED, AXCTSWEC)

Funkce

V případě kruhových taktových strojů a strojů s více vřeteny se pohybují osy nesoucí obrobek od jedné obráběcí jednotky ke druhé. Protože tyto obráběcí jednotky podléhají různým kanálům, musí být osy nesoucí obrobek v případě změny stanice / polohy dynamicky nově přiřazeny odpovídajícímu kanálu. Pro tento účel slouží osový zásobník.

V daném okamžiku je na lokální obráběcí jednotce aktivní vždy jen jedna osa/vřeteno pro upnutí obrobku. Osový zásobník soustřeďuje možnosti spojení se všemi upínacími osami/ vřeteny, z nichž je vždy právě jen jedna aktivována pro obráběcí jednotku.

Změna využitelné osy, která je definována pomocí osového zásobníku, se uskutečňuje posunutím položek v osovém zásobníku ("otočení osového zásobníku") o počet kroků zadaný pomocí nastavovaného parametru (počet slotů).

Uvolnění pro otáčení osového zásobníku může být ve výrobním programu nebo v synchronní akci naprogramováno pomocí příkazů `AXCTSWE` nebo `AXCTSWED`. Otáčení se uskuteční, jakmile je k dispozici uvolnění ze všech kanálů, které mají osy v daném osovém zásobníku.

Zrušení uvolnění pro otáčení osového zásobníku může být ve výrobním programu nebo v synchronní akci naprogramováno pomocí příkazu `AXCTSWEC` (informace k programování v synchronních akcích viz také "Zrušení uvolnění pro otáčení osového zásobníku (AXCTSWEC) [Strana 615]").

Syntaxe

```
AXCTSWE(<osový zásobník>)
AXCTSWED(<osový zásobník>)
AXCTSWEC(<osový zásobník>)
```

Význam

`AXCTSWE`: Požadavek na otočení osového zásobníku

Zpracovávání programu není příkazem `AXCTSWE` pozastaveno.

Jestliže jsou v řídícím systému odblokovány všechny kanály pro osy zásobníku, uskutečňuje se otáčení zásobníku s krokem, jehož velikost je pro daný zásobník uložena ve strojním parametru SD41700 $SN_AXCT_SWWIDTH[<číslo zásobníku>].

`AXCTSWED`: Požadavek na otáčení osového zásobníku, při kterém jako jediný působí aktivní kanál.

Upozornění
Varianta příkazu určená speciálně pro zjednodušení uvádění do provozu výrobního programu, příp. synchronní akce.

Upozornění
Chování vztahující se k dalším kanálům, které mají osy v příslušném osovém zásobníku, může být předem stanoveno pomocí následujícího strojního parametru:

MD12760 $MN_ AXCT_FUNCTION_MASK, Bit 0

`AXCTSWEC`: Zrušení uvolnění pro otáčení osového zásobníku

Upozornění
Uvolnění pro otáčení osového zásobníku může být zrušeno, pokud otáčení nebylo ještě zahájeno:

$AN_AXCTSWA[<osový zásobník>] == 0

Systémová proměnná viz "Osový zásobník (AXCTSWE, AXCTSWED, AXCTSWEC) [Strana 685]"

`<osový zásobník>`: Identifikátor osového zásobníku

Možné údaje jsou následující:

`CT<číslo zásobníku>`: Ke kombinaci písmen CT je připojeno číslo osového zásobníku.

Příklad: `CT3`

`<název zásobníku>`: Parametrem MD12750 $MN_AXCT_NAME_TAB nastavený individuální název osového zásobníku.

Příklad: `A_CONT3`

`<název osy>`: Název osy osového zásobníku, který je v příslušném kanálu známý.

Okrajové podmínky

Použití osy ze zásobníku před voláním funkce AXCTSWEC

Protože zpracovávání programu není příkazem AXCTSWE pozastaveno, je zapotřebí mít při programování synchronní akce DO AXCTSWEC na paměti následující:

Příklad:

Programový kód	Komentář
N10 AXCTSWE(CT3)	; Uvolnění otáčení osového zásobníku.
N20 AX_A10	; AX_A = osa ze zásobníku. ; Čeká se na konec otáčení osového zásobníku: $AN_AXCTSWA[CT3]==0
WHEN <podmínka> DO AXCTSWEC(AX_A)	; Odvolání uvolnění. **Nemá žádný efekt!**
N30 G4 F1	

Protože se po bloku N10 s uvolněním otáčení osového zásobníku používá v bloku N20 osa z tohoto osového zásobníku (AX_A) a protože toto použití má za následek, že se čeká na dokončení otáčení tohoto osového zásobníku, uskuteční se synchronní akce teprve společně s programovým blokem N30 ve zpracování hlavní větve programu a v důsledku toho nemá příkaz žádný efekt.

Pomoc:

Programový kód	Komentář
N11 AXCTSWE(CT3)	; Uvolnění otáčení osového zásobníku.
WHEN <podmínka> DO AXCTSWEC(AX_A)	; Odvolání uvolnění
N21 ...	; Zpracovatelný NC blok.
N31 AX_A10	; Čeká se na konec otáčení osového zásobníku: $AN_AXCTSWA[CT3]==0

UPOZORNĚNÍ

Bez zpracovatelného bloku N21 se synchronní akce uskuteční teprve po skončení otáčení osového zásobníku spolu s následujícím zpracovatelným programovým blokem N31 v hlavní větvi programu a byla by stejně jako ve výše uvedeném příkladu bez jakéhokoli efektu.

Viz také

Osový zásobník (AXCTSWE, AXCTSWED, AXCTSWEC) Osový zásobník (AXCTSWE, AXCTSWED, AXCTSWEC) [Strana 685]

Další informace

Osový zásobník

Prostřednictvím osového zásobníku mohou být přiřazovány:

- lokální adresy a/nebo
- Spřažené osy

Osové zásobníky se spřaženými osami jsou provozními prostředky, jež přesahují hranice NCU (globální v NCU), které mohou být řídícím systémem koordinovány. Osové zásobníky, v nichž jsou spravovány výlučně lokální osy, jsou možné.

Literatura:
Pokud budete potřebovat podrobné informace o konfigurace osových zásobníků, viz: Příručka Popis funkcí, Rozšiřovací funkce; B3: Další ovládací panely na více jednotkách NCU, decentralizované systémy

Kritéria uvolnění

AXCTSWE( )

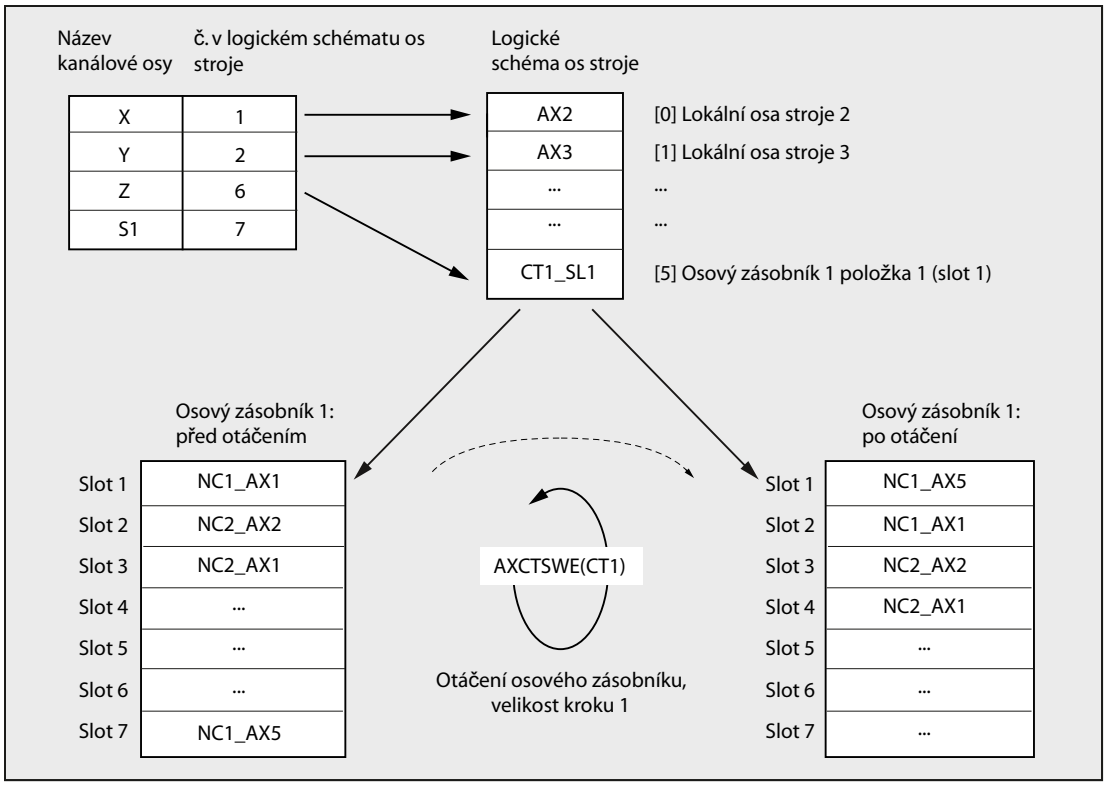
Pomocí příkazu `AXCTSWE` se uskutečňuje uvolnění otáčení osového zásobníku ve specifickém kanálu. Jakmile mají všechny kanály, které mají osy v zásobníku, uděleno uvolnění, otáčení se uskuteční.

Otáčení osového zásobníku probíhá s velikostí kroku uloženou v nastavovaném parametru:

SD41700 $SN_AXCT_SWWIDTH[<osový zásobník>]

Následující příklad ukazuje, jak operace probíhá.

Příklad:



Po otočení osového zásobníku o velikost kroku rovnající se 1 je kanálové ose Z namísto osy AX1 na NCU1 přiřazena osa AX5 na NCU1.

AXCTSWED( )

Příkaz `AXCTSWED` slouží pro zjednodušení uvádění do provozu výrobního programu, příp. synchronní akce. Otočení osového zásobníku se přitom uskuteční okamžitě při zpracování příkazu `AXCTSWED`. Uvolnění jiných kanálů, které mají osy v tomto osovém zásobníku, není zapotřebí. Aby se uvolněné otáčení osového zásobníku uskutečnilo, musí se určité kanály nacházet ve stavu RESET. To, které kanály mají být přitom zohledňovány, se nastavuje pomocí následujícího strojního parametru:

MD12760 $MN_ AXCT_FUNCTION_MASK		
Bit	**Hodnota**	**Význam**
0	0	Při přímém otáčení osového zásobníku (AXCTSWED) musí být ve stavu RESET všechny ostatní kanály.
	1	Při přímém otáčení osového zásobníku (AXCTSWED) musí být ve stavu RESET jen ty kanály, které mají interpolační oprávnění na osy v osovém zásobníku.

Otáčení osového zásobníku probíhá s velikostí kroku uloženou v nastavovaném parametru:

SD41700 $SN_AXCT_SWWIDTH[<osový zásobník>]

Platnost

Nové přiřazení os po otočení osového zásobníku se týká všech jednotek NCU, jejichž kanály kooperují prostřednictvím logického zobrazení os stroje s otočeným osovým zásobníkem.

Diagnostika

Aktuální stav osového zásobníku může být ve výrobních programech a v synchronních akcích načítán prostřednictvím následující systémové proměnné:

Systémové proměnné	Typ	Popis	
$AC_AXCTSWA[<osový zásobník>]	BOOL	Status osového zásobníku specifického kanálu	
		1	Kanál pro uvedený osový zásobník uvolnil otáčení tohoto zásobníku. Otáčení ještě nebylo uskutečněno.
		0	Otáčení osového zásobníku už bylo provedeno.
$AN_AXCTSWA[<osový zásobník>]	BOOL	Status osového zásobníku specifické NCU	
		1	Všechny kanály dané NCU mají otáčení osového zásobníku uvolněno. Otáčení momentálně probíhá.
		0	Všechny kanály dané NCU ještě nemají otáčení osového zásobníku uvolněno. Momentálně neprobíhá žádné otáčení.
$AN_AXCTSWE[<osový zásobník>]	INT	Status otáčení osového zásobníku specifického slotu Tato systémová proměnná poskytuje informace o stavu slotů osového zásobníku prostřednictvím jednotlivých **bitů**. Každý bit odpovídá jednomu slotu.	
		1	Slot je pro otáčení uvolněn.
		0	Slot **není** uvolněn pro otáčení.

Systémové proměnné	Typ	Popis	
$AN_AXCTAS[<osový zásobník>]	INT	Počet míst (slotů), o které byl osový zásobník momentálně pootočený. Inicializační hodnota po POWER ON: 0	
		Rozsah hodnot:	0 ... maximální počet obsazených míst v osovém zásobníku - 1

Zrušení uvolnění otáčení osového zásobníku (AXCTSWEC)

V případě potřeby může být uvolnění aktuálního kanálu pro otáčení osového zásobníku zrušeno:

- naprogramováním příkazu `AXCTSWEC` ve výrobním programu nebo
- prostřednictvím synchronní akce (viz "Zrušení uvolnění pro otáčení osového zásobníku (AXCTSWEC) [Strana 615]")

Aby byla funkce v platnosti, musí být splněny následující podmínky:

- Aktuální kanál musí být své uvolnění pro otáčení osového zásobníku již přiděleno, tzn. pro tento kanál je nastavena systémová proměnná $AC_AXCTSWA[<n>] = 1.
- Otáčení osového zásobníku nebylo dosud zahájeno, tzn. systémová proměnná $AN_AXCTSWA[<n>] = 0.

Po zpracování příkazu `AXCTSWEC` má systémová proměnná $AC_AXCTSWA[<n>] v tomto kanálu hodnotu "0".

Otočení osového zásobníku s implicitními příkazy GET/GETD

Při uvolnění otáčení osového zásobníku se všechny osy tohoto zásobníku příslušející ke kanálu přiřadí kanálu prostřednictvím příkazů GET, příp. GETD. Odevzdání os je opět povoleno až po otočení osového zásobníku.

Poznámka

Toto chování může být nastaveno pomocí strojního parametru. Věnujte prosím pozornost informacím od výrobce stroje!

Poznámka

Otočení osového zásobníku pomocí implicitních příkazů GET / GETD **nemůže** být použito pro osu nacházející se ve stavu "osy hlavního zpracování" (např. pro osu PLC), protože tato osa by potom musela tento stav kvůli otáčení osového zásobníku opustit.

14.4 Čekání na platnou pozici osy (WAITENC)

Funkce

Pomocí příkazu NC jazyka `WAITENC` je možné v NC programu počkat, dokud pro osy nastavené v konfiguraci pomocí parametru MD34800 $MA_WAIT_ENC_VALID = 1 nebude k dispozici synchronizovaná, příp. obnovená poloha osy.

V čekacím stavu může dojít k přerušení, např. v důsledku spuštění programu ASUP nebo přepnutím do provozního režimu JOG. Při pokračování programu bude čekací stav v případě potřeby opět obnoven.

Poznámka

Na uživatelském rozhraní se čekací stav zobrazuje prostřednictvím stavového hlášení o čekání "Čekání na měřicí systém".

Syntaxe

Příkaz `WAITENC` může být naprogramován v programové části libovolného NC programu.

Programování se musí provádět v samostatném bloku:

```
...
WAITENC
...
```

Příklad

Příkaz `WAITENC` se používá např. v uživatelském programu .../_N_CMA_DIR/_N_PROG_EVENT_SPF řízeném prostřednictvím událostí, jak ukazuje následující příklad použití.

Příklad použití:Zpětný pohyb nástroje po POWER-OFF s transformací orientace

Obrábění s orientací nástroje bylo v důsledku výpadku proudu přerušeno.
Při následujícím opětovném náběhu systému se vyvolává uživatelský program .../_N_CMA_DIR/_N_PROG_EVENT_SPF řízený na základě událostí.

V uživatelském programu řízeném na základě událostí se pomocí příkazu `WAITENC` čeká na synchronizovanou, příp. obnovenou polohu os, aby potom bylo možné vypočítat frame, který nasměruje WCS podle směru nástroje.

```
Programový kód                          Komentář
...
IF $P_PROG_EVENT == 4                   ; Náběh systému.
    IF $P_TRAFO <> 0                    ; Transformace byla deaktivována.
        WAITENC                         ; Čekání na platné pozice orientačních os.
        TOROTZ                          ; Otočení osy Z systému WCS ve směru osy nástroje.
    ENDIF
    M17
ENDIF
...
```

Potom může nástroj v provozním režimu JOG volně vyjet pomocí zpětného pohybu ve směru osy nástroje.

14.5 Kontrola existujícího rozsahu NC jazyka (STRINGIS)

Funkce

Pomocí funkce `STRINGIS(...)` je možné zkontrolovat, zda je zadaný řetězec k dispozici jako prvek aktuálního rozsahu programovacího jazyka NC systému.

Definice

```
INT STRINGIS(STRING <název>)
```

Syntaxe

```
STRINGIS(<název>)
```

Význam

`STRINGIS`:	Funkce s výstupní hodnotou
`<název>`:	Název zkoumaného prvku programovacího jazyka NC systému
Výsledná hodnota:	Výsledná hodnota má formát yxx (desetinný formát).

Prvky programovacího jazyka NC systému

Mohou být kontrolovány následující prvky programovacího jazyka NC systému:

- G-kódy všech existujících skupin G-funkcí, např. `G0`, `INVCW`, `POLY`, `ROT`, `KONT`, `SOFT`, `CUT2D`, `CDON`, `RMB`, `SPATH`
- DIN- nebo NC-adresy, jako např. `ADIS`, `RNDM`, `SPN`, `SR`, `MEAS`
- Funkce, jako např. `TANG(...)` nebo `GETMDACT`
- Procedury, např. `SBLOF`.
- Klíčová slova, např. `ACN`, `DEFINE` nebo `SETMS`
- Systémová data, např. strojní parametry `$M...` , nastavované parametry `$S...` nebo data volitelných doplňků `$O...`
- Systémové proměnné `$A...` , `$V...` , `$P...`
- Početní parametry `R...`
- Názvy aktivovaných cyklů
- Proměnné GUD a LUD
- Názvy maker
- Názvy návěští

Výsledná hodnota

Pro výslednou hodnotu jsou důležité pouze první tři desetinná místa. Výsledná hodnota má formát yxx, kde y = základní informace a xx = podrobné informace.

Výsledná hodnota	Význam	
000	Řetězec "název" není v předkládaném systému znám [1)]	
100	Řetězec "název" je prvek programovacího jazyka NC systému, momentálně ale nemůže být naprogramován (volitelný doplněk/funkce jsou neaktivní)	
2xx	Řetězec "název" je naprogramovatelný prvek programovacího jazyka NC systému (volitelný doplněk/ funkce jsou aktivní) Podrobné informace xx obsahují další údaje o druhu prvku:	
	xx	**Význam**
	01	Adresa podle normy DIN nebo NC adresa [2)]
	02	G-kód (např. `G04`, `INVCW`)
	03	Funkce s výstupní hodnotou
	04	Funkce bez výstupní hodnoty
	05	Klíčové slovo (např. `DEFINE`)
	06	Strojní parametry (`$M`...), nastavované parametry (`$S`...) nebo parametry volitelných doplňků (`$O`...)
	07	Systémové parametry, např. systémové proměnné (`$`...) nebo početní parametry (`R`...)
	08	Cyklus (cyklus musí být načten v NCK a program cyklu musí být aktivní [3)])
	09	Proměnná GUD (proměnné GUD musí být definovány v definičním souboru GUD a proměnné GUD musí být aktivovány)
	10	Název makra (makro musí být definováno v definičním souboru makra a makra musí být aktivována) [4)]
	11	Proměnná LUD aktuálního výrobního programu
	12	G-kód podle normy ISO (režim jazyka ISO musí být aktivní)
400	Řetězec "název" je NC-adresa, která nebyla rozpoznána jako xx = 01 nebo xx = 10 a není typu G nebo R	
y00	Není možné žádné specifické přiřazení	

1) V závislosti na řídícím systému je za určitých okolností známá pouze určitá podmnožina příkazů programovacího jazyka NC systému, např. v případě systému SINUMERIK 802D sl. U těchto řídících systémů je pro řetězce, které jsou v principu příkazy NC jazyka firmy Siemens, výsledkem hodnota 0. Toto chování může být pomocí strojního parametru MD10711 $MN_NC_LANGUAGE_CONFIGURATION změněno. Je-li nastaveno MD10711 = 1, potom bude pro příkazy NC jazyka firmy Siemens vždy výsledkem hodnota 100.

2) NC adresami jsou následující písmena: A, B, C, E, I, J, K, Q, U, V, W, X, Y, Z. Tyto NC adresy mohou být naprogramovány také s rozšířením adresy. Rozšíření adresy může být zadáno při kontrole pomocí funkce STRINGIS.
Příklad: `201 == STRINGIS("A1").`

Písmena: D, F, H, L, M, N, O, P, S, T jsou NC adresy nebo pomocné funkce, které jsou používány podle uživatelských definic. V jejich případě je výsledkem vždy hodnota 400. Příklad: 400 == STRINGIS("D"). Tyto NC adresy mohou být zadány při kontrole pomocí funkce STRINGIS, ale bez rozšíření adresy.
Příklad: `000 == STRINGIS("M02")`, ale `400 == STRINGIS("M").`

3) Názvy parametrů cyklů nemohou být pomocí funkce STRINGIS kontrolovány.
4) Adresa definovaná jako makro, např.. G, H, M, L, je jako makro identifikována.

Příklady

V následujících příkladech se předpokládá, že řetězec zadaný jako prvek jazyka NC systému, pokud není uvedeno jinak, může být v principu v NC systému naprogramován.

1. Řetězec "T" je definován jako pomocná funkce:
   ```
   400 == STRINGIS("T")
   000 == STRINGIS ("T3")
   ```
2. Řetězec "X" je definován jako osa:
   ```
   201 == STRINGIS("X")
   201 == STRINGIS("X1")
   ```
3. Řetězec "A2" je definován jako NC adresa s rozšířením:
   ```
   201 == STRINGIS("A")
   201 == STRINGIS("A2")
   ```
4. Řetězec "INVCW" je definován jako specifický G-kód:
   ```
   202 == STRINGIS("INVCW")
   ```
5. Řetězec "$MC_GCODE_RESET_VALUES" je definován jako strojní parametr:
   ```
   206 == STRINGIS("$MC_GCODE_RESET_VALUES")
   ```
6. Řetězec "GETMDACT" je funkce NC-jazyka:
   ```
   203 == STRINGIS("GETMDACT ")
   ```
7. Řetězec "DEFINE" je klíčové slovo:
   ```
   205 == STRINGIS("DEFINE")
   ```
8. Řetězec "$TC_DP3" je systémový parametr (složka délky nástroje):
   ```
   207 == STRINGIS("$TC_DP3")
   ```
9. Řetězec "$TC_TP4" je systémový parametr (velikost nástroje):
   ```
   207 == STRINGIS("$TC_TP4")
   ```
10. Řetězec "$TC_MPP4" je systémový parametr (stav místa v zásobníku):
    - Správa zásobníku nástrojů je aktivní. `207 == STRINGIS("$TC_MPP4") ;`
    - Správa zásobníku nástrojů není aktivní. `000 == STRINGIS("$TC_MPP4")`

    Viz také kapitola: Správa zásobníku nástrojů
11. Řetězec "MACHINERY_NAME" je definován jako proměnná typu GUD.
    ```
    209 == STRINGIS("MACHINERY_NAME")
    ```
12. Řetězec "LONGMACRO" je definován jako makro:
    ```
    210 == STRINGIS("LONGMACRO")
    ```
13. Řetězec "MYVAR" je definován jako proměnná typu LUD.
    ```
    211 == STRINGIS("MYVAR")
    ```
14. Řetězec "XYZ" není žádný příkaz, který by byl v NCK známý, není ani názvem proměnné GUD, makra nebo cyklu:
    ```
    000 == STRINGIS("XYZ")
    ```

Správa zásobníku nástrojů

Jestliže funkce "Správa zásobníku nástrojů" není aktivní, poskytuje funkce STRINGIS pro systémové parametry správy zásobníku nástrojů, nezávisle na strojním parametru, následující výsledky:

- MD10711 $MN_NC_LANGUAGE_CONFIGURATION

Vždy hodnota 000.

Režim ISO

Pokud je funkce "Režim ISO" aktivní:

- MD18800 $MN_MM_EXTERN_LANGUAGE (aktivování externích BC jazyků)
- MD10880 $MN_ MM_EXTERN_CNC_SYSTEM (řídící systémy, které se mají přizpůsobit)

kontroluje funkce STRINGIS zadaný řetězec napřed jako G-kód systému SINUMERIK. Pokud řetězec není G-kódem systému SINUMERIK, zkontroluje se potom, zda se nejedná o G-kód ISO.

Naprogramovaná přepnutí (`G290` (režim SINUMERIK), `G291` (režim ISO)) nemají na funkci STRINGIS žádný vliv.

Příklad

Strojní parametry, které jsou pro funkci STRINGIS(...) důležité, mají následující hodnoty:

- MD10711 $MN_NC_LANGUAGE_CONFIGURATION = 2 (jako známé jsou rozpoznány jen ty příkazy NC-jazyka, jejichž volitelné doplňky jsou aktivovány)
- MD19410 $ON_TRAFO_TYPE_MASK = 'H0' (volitelný doplněk: transformace)
- MD10700 $MN_PREPROCESSING_LEVEL='H43' (předběžné zpracování pro cykly je aktivní)

Program uvedený v následujícím příkladu se zpracuje bez chybového hlášení:

```
Programový kód                          Komentář
N1 R1=STRINGIS("TRACYL")                ;   R1 == 0, protože příkaz TRACYL je kvůli chybějícímu
                                        ;   volitelnému doplňku "Transformace" rozpoznán jako
                                        ;   "neznámý".
N2 IF STRINGIS("TRACYL") == 204         ;
N3  TRACYL(1,2,3)                       ;   N3 se bude přeskakovat
N4 ELSE
N5  G00                                 ;   a místo něj se bude zpracovávat N5
N6 ENDIF
N7 M30
```

14.6 Volání funkce ISVAR a čtení indexu pole strojních parametrů

Funkce

Příkaz ISVAR je funkce ve smyslu jazyka NC systému, která má tyto parametry:

- Hodnota funkce je typu BOOL
- Předávaný parametr je typu STRING

Výsledek funkce ISVAR je hodnota TRUE, pokud předávaný parametr obsahuje proměnnou, která je NC systému známá (strojní parametr, nastavovaný parametr, systémová proměnná, obecná proměnná, jako např. GUD).

Syntaxe

```
ISVAR(<identifikátor proměnné>)
ISVAR(<identifikátor>,[<hodnota>,<hodnota>])
```

Význam

<identifikátor proměnné>	Předávaný parametr typu STRING může být buď bezrozměrný, jednorozměrný nebo dvourozměrný.
<identifikátor>	Identifikátor obsahující proměnnou s nebo bez indexu pole, která je NC systému známá, jako např. strojní parametr, nastavovaný parametr, systémová proměnná nebo obecná proměnná. **Rozšíření:** U všeobecných a kanálových strojních parametrů se načte první prvek pole, i když index není uveden.
<hodnota>	Hodnota funkce je typu BOOL

Zkoušky

V závislosti na předávaném parametru jsou prováděny následující zkoušky:

- Je identifikátor k dispozici?
- Jedná se o jedno- nebo dvourozměrné pole?
- Je index pole povolen?

Jen když odpovědi na všechny tyto otázky jsou kladné, je výsledkem hodnota TRUE. Jestliže jen jedna ze zkoušek není splněna nebo pokud se vyskytne syntaktická chyba, je výsledkem hodnota FALSE. Osové proměnné jako index pro název osy jsou akceptovány, blíže se však už nezkoumají.

Rozšíření: Čtení pole strojních a nastavovaných parametrů bez indexu.

Jestliže v případě **všeobecných a kanálových** strojních parametrů není uveden index, alarm 12400 "Kanál % 1 Blok % 2 Pole % 3 Prvek není k dispozici" se už **nebude** aktivovat.

V případě **osových** strojních parametrů musí být i nadále naprogramován **nejméně jeden index osy**. Jinak se aktivuje alarm 12400.

Příklad: Volání funkce ISVAR

Programový kód	Komentář
DEF INT VAR1	
DEF BOOL IS_VAR=FALSE	; Předávaný parametr je všeobecná proměnná
N10 IS_VAR=ISVAR("VAR1")	; IS_VAR je v tomto případě TRUE
DEF REAL VARARRAY[10,10]	
DEF BOOL IS_VAR=FALSE	; různé syntaktické proměnné
N20 IS_VAR=ISVAR("VARARRAY[,]")	; IS_VAR je TRUE v případě dvourozměrného pole
N30 IS_VAR=ISVAR("VARARRAY")	; IS_VAR je TRUE, proměnná existuje
N40 IS_VAR=ISVAR("VARARRAY[8,11]")	; IS_VAR je FALSE, index pole je nepřípustný
N50 IS_VAR=ISVAR("VARARRAY[8,8")	; IS_VAR je FALSE, syntaktická chyba kvůli chybějícímu znaku "]"
N60 IS_VAR=ISVAR("VARARRAY[,8]")	; IS_VAR je TRUE, index pole je přípustný
N70 IS_VAR=ISVAR("VARARRAY[8,]")	; IS_VAR je TRUE
DEF BOOL IS_VAR=FALSE	; Předávaný parametr je strojní parametr
N100 IS_VAR=ISVAR("$MC_GCODE_RESET_VALUES[1]"	; IS_VAR je TRUE
DEF BOOL IS_VAR=FALSE	; Předávaný parametr je systémová proměnná
N10 IS_VAR=ISVAR("$P_EP")	; IS_VAR je v tomto případě TRUE
N10 IS_VAR=ISVAR("$P_EP[X]")	; IS_VAR je v tomto případě TRUE

Příklad: Čtení pole strojních parametrů s indexem a bez indexu.

První prvek se načte

```
R1=$MC_EXTERN_GCODE_RESET_VALUES
```

což odpovídá dřívějšímu

```
R1=$MC_EXTERN_GCODE_RESET_VALUES[0]
```

nebo se načte první prvek

```
R1=$MA_POSTCTRL_GAIN[X1]
```

což odpovídá dřívějšímu

```
R1=$MA_POSTCTRL_GAIN[0, X1]
```

Načítání prvního prvku je možné také v synchronních akcích:

```
WHEN TRUE DO $R1 = $MC_EXTERN_GCODE_RESET_VALUES
```

což odpovídá dřívějšímu

```
WHEN TRUE DO $R1 = $MC_EXTERN_GCODE_RESET_VALUES[0]
```

což se dříve **nepřečetlo** a aktivoval se alarm 12400.

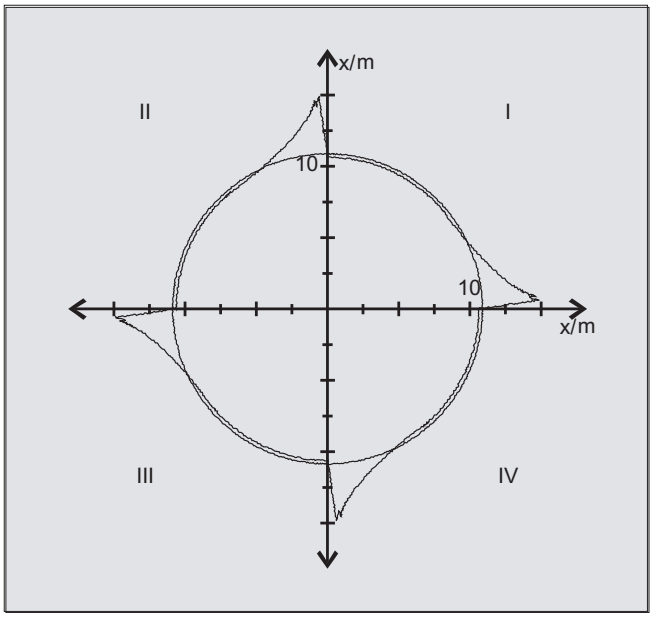
Alarm 12400 se i nadále aktivuje, je-li naprogramováno:

```
R1=$MA_POSTCTRL_GAIN
```

14.7 Zjišťování kompenzačních charakteristik (QECLRNON, QECLRNOF)

Funkce

Kompenzace chyby kvadrantu (QFK) zmenšuje chybu kontury, která vzniká při obrácení směru pohybu v důsledku mechanických nelinearit (např. tření, vůle) nebo torze. Optimální kompenzační parametry mohou být přizpůsobeny prostřednictvím neuronových sítí řídícího systému v průběhu fáze učení a tímto způsobem jsou kompenzační charakteristiky zjišťovány automaticky. Učení může probíhat pro až 4 osy současně.



Syntaxe

```
QECLRNON
QECLRNOF
```

Aktivování operace učení: QECLRNON

Vlastní operace učení se aktivuje v NC programu pomocí příkazu `QECLRNON` a zadání os:

```
QECLRNON (X1, Y1, Z1, Q)
```

Charakteristiky se změní, jen když je aktivován tento příkaz.

Ukončení učení: QECLRNOF

Poté, co se výukové pohyby požadovaných os ukončily, se operace učení pro všechny osy pomocí příkazu `QECLRNOF` vypne.

Význam

`QECLRNON (osa.1,…4)`	Aktivování funkce "Učení kompenzace chyby kvadrantu"
`QECLRNO`	Deaktivování funkce "Učení kompenzace chyby kvadrantu"
`QECLRN.SPF`	Výukový cyklus
`QECDAT.MPF`	Vzorový NC program pro dosazení hodnot pro systémové proměnné a pro nastavení parametrů výukového cyklu.
`QECTEST.MPF`	Vzorový program pro zkoušku kruhového tvaru.

Popis

Posuvové pohyby os, které jsou pro učení zapotřebí, jsou generovány pomocí NC programu. V tomto programu jsou uloženy pro učení potřebné pohyby ve formě výukového cyklu.

Úplně první operace učení

Pro účely první operace učení v rámci uvádění do provozu jsou vzorové NC programy pro učení s pohyby, které jsou pro fázi učení zapotřebí, jakož i s hodnotami dosazovanými systémovým proměnným pro QFK, obsaženy na disketě se základním programem pro PLC.

Dodatečné učení

Dodatečná optimalizace již naučených charakteristik je možná pomocí "dodatečného učení". Přitom se využívají starší data, která se už nacházejí v uživatelské paměti. Pro účely dodatečného učení přizpůsobte vzorové NC programy svým potřebám.

Parametry výukového cyklu (např. QECLRN.SPF) je zapotřebí v případě potřeby pro "dodatečné učení" upravit:

- Nastavte "Režim učení" = 1
- V případě potřeby zmenšete "Počet výukových cyklů"
- V případě potřeby aktivujte "Blokové učení" a definujte odpovídající hranice oblastí.

14.8 Interaktivní vyvolávání oken z výrobního programu (MMC)

Funkce

Pomocí příkazu MMC mohou být z výrobního programu zobrazována na HMI uživatelem definovaná dialogová okna (dialogové obrazovky).

Vzhled dialogového okna je definován čistě textovou konfigurací (soubor COM v adresáři cyklů), software systému HMI zůstává přitom nezměněn.

Uživatelem definovaná dialogová okna nemohou být vyvolána v různých kanálech současně.

Syntaxe

```
MMC(CYCLES,PICTURE_ON,T_SK.COM,BILD,MGUD.DEF,BILD_3.AWB,TEST_1,A1","
S")
```

Význam

MMC	Vyvolání interaktivního dialogového okna na HMI z výrobního programu.
CYCLES	Systémová oblast, ve které se v konfiguraci nastavený dialog s uživatelem uskutečňuje.
PICTURE_ON příp. PICTURE_OFF	Příkaz: Aktivování nebo deaktivování obrázku
T_SK.COM	Soubor COM: Název souboru dialogové obrazovky (uživatelské cykly) Zde je definován vzhled dialogové obrazovky. V dialogové obrazovce se mohou vypisovat uživatelské proměnné a/nebo textové komentáře.
BILD	Název dialogové obrazovky: Jednotlivé dialogové obrazovky se vybírají pomocí svých názvů.
MGUD.DEF	Definiční soubor uživatelských dat, do kterých je při čtení/zápisu proměnných zajištěn přístup.
BILD_3.AWB	Soubor grafiky
TEST_1	Doba zobrazování potvrzovací proměnné
A1	Textové proměnné...",
"S"	Režim potvrzování: synchronní, potvrzování pomocí programového tlačítka "OK".

Literatura

Pokud budete potřebovat podrobné pokyny pro programování příkazu MMC (včetně příkladů programování), viz Příručka pro uvádění do provozu.

14.9 Doba zpracování programu / počítadlo obrobků

14.9.1 Doba zpracování programu / počítadlo obrobků (přehled)

Za účelem podpory pracovníků obsluhy obráběcích strojů jsou k dispozici informace týkající doby zpracovávání programu a počtu kusů.

Tyto informace mohou být zpracovávány jako systémové proměnné v NC programu a/nebo v programu PLC. Současně jsou tyto informace k dispozici pro zobrazování na uživatelském rozhraní.

14.9.2 Doba zpracování programu

Funkce

Funkce "Doba zpracování programu" dává k dispozici interní časovač NC systému pro monitorování technologických procesů, jehož hodnotu je možné načítat ve výrobním programu a v synchronních akcích pomocí systémových proměnných kanálu a NC systému.

Spouštění měření doby zpracování ($AC_PROG_NET_TIME_TRIGGER) je jediná systémová proměnná této funkce, do které lze zapisovat, a slouží pro selektivní měření úseků programu, tzn. zapsáním hodnoty pro spuštění může být v NC programu měření času zastaveno a znovu spuštěno.

Systémové proměnné	Význam	Aktivita
V NC systému		
$AN_SETUP_TIME	Čas od posledního náběhu řídícího systému se standardními hodnotami ("studený start") v minutách Při každém náběhu řídícího systému se standardními hodnotami se automaticky nastaví na "0".	• vždy aktivní
$AN_POWERON_TIME	Čas od posledního normálního náběhu řídícího systému ("teplý start") v minutách Při každém normálním náběhu řídícího systému se automaticky nastaví na "0".	
V kanálu		
$AC_OPERATING_TIME	Celková doba, po kterou jsou zpracovávány NC-programy v automatickém provozním režimu v sekundách Hodnota se s každým náběhem řídícího systému se automaticky nastaví na "0".	• Aktivování prostřednictvím MD27860 • v provozním režimu "AUTO"
$AC_CYCLE_TIME	Doba zpracování zvoleného NC programu v sekundách Hodnota se s každým spuštěním nového NC programu automaticky nastaví na "0".	
$AC_CUTTING_TIME	Doba opracovávání v sekundách Měří se doba, po kterou se pohybují dráhové osy (minimálně jedna je aktivní) bez aktivního rychlého posuvu ve všech NC programech od stisknutí tlačítka NC-Start do konce programu/resetu NC systému. Měření této doby se přeruší také tehdy, když je aktivní doba prodlevy. Hodnota se při každém náběhu řídícího systému se standardními hodnotami se automaticky nastaví na "0".	

Systémové proměnné	Význam		Aktivita
$AC_ACT_PROG_NET_TIME	Aktuální čistá doba zpracování momentálně vybraného NC programu v sekundách S každým spuštěním nového NC programu automaticky nastaví na "0".		• vždy aktivní • v provozním režimu "AUTO"
$AC_OLD_PROG_NET_TIME	Čistá doba zpracování právě správně a pomocí příkazu `M30` ukončeného programu v sekundách		
$AC_OLD_PROG_NET_TIME_COUNT	Změny parametru $AC_OLD_PROG_NET_TIME Po náběhu systému je proměnná $AC_OLD_PROG_NET_TIME_COUNT nastavena na "0". Proměnná $AC_OLD_PROG_NET_TIME_COUNT se zvýší vždy, když řídící systém zapíše do proměnné $AC_OLD_PROG_NET_TIME novou hodnotu.		
$AC_PROG_NET_TIME_TRIGGER	Spouštění pro měření doby zpracování:		• v provozním režimu "AUTO"
	0	Neutrální stav Spouštění není aktivní.	
	1	Ukončit Ukončí měření a zkopíruje hodnotu z proměnné $AC_ACT_PROG_NET_TIME do $AC_OLD_PROG_NET_TIME. $AC_ACT_PROG_NET_TIME se znovu nastaví na "0" a potom se zpustí od začátku.	
	2	Start Spustí se měření a proměnná $AC_ACT_PROG_NET_TIME se přitom nastaví na "0". Proměnná $AC_OLD_PROG_NET_TIME se nemění.	
	3	Zastavit Zastaví měření: Proměnná $AC_OLD_PROG_NET_TIME se nemění a obsahuje hodnotu proměnné $AC_ACT_PROG_NET_TIME konstantní, dokud se nespustí další pokračování.	
	4	Pokračovat Pokračování měření, tzn. dříve zastavené měření bude pokračovat dál. Proměnná $AC_ACT_PROG_NET_TIME běží dál. Proměnná $AC_OLD_PROG_NET_TIME se nemění.	
Při náběhu systému (Power-On) jsou všechny systémové proměnné nastaveny na "0"!			
Literatura: Pokud budete potřebovat podrobný popis zde uváděných systémových proměnných, viz: Příručka Popis funkcí, Základní funkce; BAG, kanál, zpracování programu, chování při resetu (K1), kapitola: Doba zpracování programu			

Poznámka

Výrobce stroje

Spuštění aktivovatelného časovače se uskutečňuje pomocí strojního parametru MD27860 $MC_PROCESSTIMER_MODE.

Chování aktivního měření času v případě určitých funkcí (např. GOTOS, korekce (override) = 0%, aktivní posuv při zkušebním zpracování, zkouška programu, ASUP, PROG_EVENT, ...) se nastavuje v konfiguraci pomocí strojních parametrů MD27850 $MC_PROG_NET_TIMER_MODE a MD27860 $MC_PROCESSTIMER_MODE.

Literatura:
Příručka Popis funkcí, Základní funkce; BAG, kanál, zpracování programu, chování při resetu (K1), kapitola: Doba zpracování programu

Poznámka

Zbývající čas pro obrobek

Jestliže jsou jeden po druhém produkovány stejné obrobky, je možné na základě časových hodnot zjistit

- dobu opracovávání pro naposled vyrobený obrobek (viz $AC_OLD_PROG_NET_TIME)

a

- aktuální dobu opracovávání (viz $AC_ACT_PROG_NET_TIME)

a z nich vypočítat zbývající čas pro daný obrobek.

Zbývající čas se vedle aktuální doby opracovávání vypisuje na uživatelském rozhraní.

UPOZORNĚNÍ

Použití příkazu STOPRE

Systémové proměnné $AC_OLD_PROG_NET_TIME a $AC_OLD_PROG_NET_TIME_COUNT nezpůsobují žádné implicitní zastavení předběžného zpracování. Při použití ve výrobním programu nelze poznat, kdy hodnota systémové proměnné pochází z předchozího zpracování programu. Jestliže se ale příkazy pro spuštění měření doby zpracování ($AC_PROG_NET_TIME_TRIGGER) zapisují s velmi vysokou frekvencí a hodnota v proměnné $AC_OLD_PROG_NET_TIME se v důsledku toho mění velmi často, pak by měl být ve výrobním programu použit explicitní příkaz `STOPRE`.

Okrajové podmínky

- **Vyhledávání bloku**

  Při vyhledávání bloku se žádná doba zpracovávání programu nezjišťuje.

- **REPOS**

  Doba trvání operace REPOS se do aktuální doby zpracovávání programu ($AC_ACT_PROG_NET_TIME) započítává.

Příklady

Příklad 1: Měření doby zpracovávání programu "mySubProgrammA"

Programový kód

```
...
N50 DO $AC_PROG_NET_TIME_TRIGGER=2
N60 FOR ii= 0 TO 300
N70 mySubProgrammA
N80 DO $AC_PROG_NET_TIME_TRIGGER=1
N95 ENDFOR
N97 mySubProgrammB
N98 M30
```

Poté, co byl zpracován programový řádek `N80`, se v proměnné $AC_OLD_PROG_NET_TIME nachází čistá doba zpracování programu "mySubProgrammA".

Hodnota proměnné $AC_OLD_PROG_NET_TIME:

- zůstává zachována i za příkazem `M30`.
- po každém průchodu smyčkou se aktualizuje.

Příklad 2: Měření doby zpracovávání programů "mySubProgrammA" a "mySubProgrammC"

Programový kód

```
...
N10 DO $AC_PROG_NET_TIME_TRIGGER=2
N20 mySubProgrammA
N30 DO $AC_PROG_NET_TIME_TRIGGER=3
N40 mySubProgrammB
N50 DO $AC_PROG_NET_TIME_TRIGGER=4
N60 mySubProgrammC
N70 DO $AC_PROG_NET_TIME_TRIGGER=1
N80 mySubProgrammD
N90 M30
```

14.9.3 Počítadlo obrobků

Funkce

Funkce "Počítadlo obrobků" poskytuje k dispozici různé čítače, které lze použít zejména pro počítání obrobků v rámci řídícího systému.

Počítadla existují jako specifické kanálové proměnné s oprávněními pro čtení a pro zápis a s rozsahem hodnot od 0 do 999 999 999.

Systémové proměnné	Význam
$AC_REQUIRED_PARTS	Počet obrobků, které mají být vyrobeny (požadovaná hodnota obrobků) V tomto čítači může být definován počet obrobků, při jehož dosažení se počet aktuálních obrobků v proměnné ($AC_ACTUAL_PARTS) nastaví zpět na "0".
$AC_TOTAL_PARTS	Celkový počet vyrobených obrobků (celková skutečná hodnota) Tento čítač udává počet všech obrobků vyrobených od okamžiku spuštění. Hodnota se automaticky nastaví na "0" jen při náběhu řídícího systému se standardními hodnotami.
$AC_ACTUAL_PARTS	Počet vyrobených obrobků (skutečná hodnota počtu obrobků) V tomto čítači je zaznamenáván počet všech obrobků vyrobených od okamžiku spuštění. Jestliže je dosaženo požadované hodnoty počtu kusů ($AC_REQUIRED_PARTS), čítač se automaticky nastaví zpět na "0" (předpokladem je, že $AC_REQUIRED_PARTS > 0).
$AC_SPECIAL_PARTS	Počet uživatelem počítaných obrobků Tento čítač umožňuje uživateli počítání obrobků podle jeho vlastní definice. Je možno také definovat aktivování alarmu, jakmile je dosaženo požadovaného počtu obrobků ($AC_REQUIRED_PARTS). Vynulování čítače musí provádět sám uživatel.

Poznámka

Všechna počítadla obrobků se při náběhu řídícího systému se standardními hodnotami nastavují na "0" a mohou být nezávisle na svém aktivování čteny nebo lze do nich zapisovat.

Poznámka

Aktivování čítačů, okamžik vynulování a algoritmus počítání je možno ovlivňovat pomocí kanálových strojních parametrů.

Poznámka

Počítání obrobků s příkazem M-funkce definovaným uživatelem

Pomocí strojního parametru může být nastaveno, že počítané impulzy pro různá počítadla obrobků mohou být spouštěny nikoli pomocí příkazu pro konec programu `M2`/`M30`, ale pomocí uživatelem definované M-funkce.

Literatura

Pokud budete potřebovat další informace týkající se funkce "Počítadlo obrobků", viz:

- Příručka Popis funkcí, Základní funkce; BAG, kanál, zpracování programu, chování při resetu (K1), kapitola: Počítadlo obrobků

14.10 Výstup do externího zařízení/souboru (EXTOPEN, WRITE, EXTCLOSE)

Funkce

Pomocí této funkce je možné zapisovat data z výrobního programu na externí zařízení / do externího souboru, např. za účelem protokolování výrobních dat nebo ovládání doplňkových agregátů připojených k řídícímu systému.

Výstup do externího zařízení/souboru probíhá ve třech krocích:

1. Otevření externího zařízení/souboru

   Pomocí příkazu `EXTOPEN` se externí zařízení/souboru otevřou za účelem zápisu.

2. Zápis dat

   Výstupní data mohou být připravena pomocí funkcí NC jazyka pro práci s řetězci "(Operace s řetězci [Strana 75]"), např. `SPRINT`. Samotný zápis se uskutečňuje pomocí příkazu `WRITE`.

3. Zavření externího zařízení/souboru

   Pomocí příkazu `EXTCLOSE` nebo při dosažení konce programu (`M30`), jakož i při resetu kanálu, že v kanálu obsazený externí zařízení/soubor bude opět uvolněn.

Poznámka

Ve výrobním programu/kanálu může být používán i více než jedno externí zařízení/soubor.

Využitelnost

Funkce je k dispozici:

- jen ve výrobních programech (**nikoli** v synchronních akcích)
- paralelně ve všech kanálech NCK sloužících pro opracovávání pro všechna (konfigurovaná) výstupní zařízení, která jsou k dispozici

Pro každé výstupní zařízení může být při jeho otevírání zadáno, zda toto zařízení smí být používáno výlučně z jednoho konkrétního kanálu nebo zda může být používáno střídavě různými kanály, které na ně chtějí odeslat data (sdílený režim).

Syntaxe

```
DEF INT <chyba>
DEF STRING[<n>] <výstup>
…
EXTOPEN(<chyba>,"<ExtG>",<režim zpracování>,<režim využívání>,
<režim zápisu>)
…
<výstup>="Výstup dat"
WRITE(<chyba>,"<ExtG>",<výstup>)
…
EXTCLOSE(<chyba>,"<ExtG>")
```

Význam

EXTOPEN: Příkaz pro otevření externího zařízení/souboru

<chyba>: **Parametr 1:** Proměnná pro výslednou chybovou hodnotu
Na základě hodnoty chyby je možné v programu vyhodnocovat, jak operace skončila, a na základě toho potom odpovídajícím způsobem postupovat.

Typ: INT

Hodnoty:

0	žádná chyba
1	externí zařízení nebylo možné otevřít
2	externí zařízení není konfigurováno
3	pro externí zařízení je v konfiguraci zadána nesprávná cesta
4	pro externí zařízení nemáte přístupová oprávnění
5	Režim využívání: externímu zařízení je přiřazen "exkluzivní" přístup
6	Režim využívání: externímu zařízení je přiřazen "sdílený" přístup
7	Délka souboru je větší než kolik udává parametr LOCAL_DRIVE_MAX_FILESIZE
8	je překročen maximální počet externích zařízení
9	funkce pro LOCAL_DRIVE není aktivována
11	rozhraní RS-232 je již obsazeno funkcí Easy-Message (jen 828D)
12	Režim zápisu: zadání je v rozporu se souborem extdev.ini
16	naprogramována neplatná externí cesta
22	externí zařízení není připojeno

<ExtG>: **Parametr 2:** Symbolický identifikátor pro otevírané externí zařízení/soubor

Typ: STRING

Symbolický identifikátor se skládá z těchto částí:

1. název logického zařízení
2. v případě potřeby následovaný cestou k danému souboru (připojuje se pomocí znaku "/")

Jsou definovány následující **názvy logických zařízení**:

"LOCAL_DRIVE":	Lokální kompaktní Flash-karta (předdefinováno)
"CYC_DRIVE":	Rezervované udání jednotky používané v cyklech firmy SIEMENS (předdefinováno)

"/dev/ext/1",... "/dev/ext/9":	Síťové jednotky, které jsou k dispozici **Upozornění:** Je zapotřebí nastavit konfiguraci v souboru extdev.ini!
"/dev/cyc/1", "/dev/cyc/2":	Rezervované udání jednotky používané v cyklech firmy SIEMENS **Upozornění:** Je zapotřebí nastavit konfiguraci v souboru extdev.ini!
"/dev/v24":	Rozhraní RS-232 **Upozornění:** Je zapotřebí nastavit konfiguraci v souboru extdev.ini!

Cesta k souboru:

- Pro názvy "LOCAL_DRIVE" a "CYC_DRIVE" musí být uvedena cesta k souboru, např.:

  "LOCAL_DRIVE/my_dir/my_file.txt"

- Názvy logických zařízení "/dev/ext/1...9" a "/dev/cyc/1...2" mohou prostřednictvím konfigurace odkazovat:
  - na určitý soubor, potom smí být uváděny pouze názvy logických zařízení, např.:

    "/dev/ext/4"

  - nebo na adresář, potom ale musí být cesta k souboru zadána, např.:

    "/dev/ext/5/my_dir/my_file.txt"

- U názvu "/dev/v24" nesmí být žádná cesta k souboru připojena.

Upozornění:
Pro názvy logických zařízení "/dev/ext/1...9", "/dev/v24" a "/dev/cyc/1...2" nejsou velká a malá písmena rozlišována, u zadání cesty k souboru jsou velká a malá písmena rozlišována. Pro "LOCAL_DRIVE" a "CYC_DRIVE" jsou přípustná jen velká písmena.

<režim zpracování>: **Parametr 3:** Režim zpracování pro příkazy WRITE k tomuto zařízení/souboru

Typ: STRING

Hodnoty: "SYN":

Synchronní zápis

Zpracovávání programu je pozastaveno, dokud není operace zápisu dokončena.

Úspěšné ukončení synchronního zápisu může být zkontrolováno pomocí vyhodnocování chybové proměnné příkazu WRITE.

"ASYN":
Asynchronní zápis
Zpracovávání programu není příkazem WRITE přerušeno.
Upozornění:
Chybová proměnná příkazu WRITE nemá v tomto režimu žádnou vypovídací hodnotu a má vždy hodnotu 0 (žádná chyba). V tomto režimu si nemůžete být jisti, že byl příkaz WRITE úspěšně zpracován.

<režim využívání>: **Parametr 4:** Režim využívání pro toto zařízení/tento soubor

Typ: STRING

Hodnoty:
- "SHARED": Zařízení/soubor jsou přiřazovány ve sdíleném režimu. Zařízení může být používáno také jinými kanály, tzn. otevřenými rovněž v tomto režimu.
- "EXCL": Zařízení/soubor jsou používány výlučně v tomto kanálu a žádný další kanál nemůže toto zařízení používat také.

<režim zápisu>: **Parametr 5:** Režim zápisu pro příkazy WRITE k tomuto zařízení/souboru (volitelné)

Typ: STRING

Hodnoty:
- "APP": Připojit
  Obsah souboru zůstává zachován, zapisovaná data se vkládají na jeho konec.
- "OVR": Přepisování
  Obsah souboru se vymaže a následující operací zápisu se znovu vytvoří.

Upozornění:
Režim zápisu nastavený v konfiguraci v souboru extdev.ini **není možné** pomocí tohoto parametru přepsat. V případě konfliktu je volání funkce EXTOPEN potvrzeno s chybou.

WRITE: Příkaz pro zápis výstupních dat
Popis viz "Zápis do souboru (WRITE) [Strana 140]"!

EXTCLOSE: Příkaz pro zavření otevřeného externího zařízení/souboru

<chyba>: **Parametr 1:** Proměnná pro výslednou chybovou hodnotu

Typ: INT

Hodnoty:
- 0 žádná chyba
- 16 naprogramována neplatná externí cesta
- 21 chyba při zavírání externího zařízení

<ExtG>: **Parametr 2:** Symbolický identifikátor pro zavírané externí zařízení/soubor

Pokud budete potřebovat popis, viz příkaz EXTOPEN!

Upozornění:
Tento identifikátor musí být identický s identifikátorem použitým v příkazu EXTOPEN!

Příklad

Programový kód

```
N10    DEF INT RESULT
N20    DEF BOOL EXTDEVICE
N30    DEF STRING[80] AUSGABE
N40    DEF INT PHASE
N50    EXTOPEN(RESULT,"LOCAL_DRIVE/my_file.txt","SYN","SHARED")
N60    IF RESULT > 0
N70      MSG("Fehler bei EXTOPEN:" << RESULT)
N80    ELSE
N90      EXTDEVICE=TRUE
N100   ENDIF
…
N200   PHASE=4
N210   IF EXTDEVICE
N220     AUSGABE=SPRINT("Ende Phase: %D",PHASE)
N230     WRITE(RESULT,"LOCAL_DRIVE/my_file.txt",AUSGABE)
N240   ENDIF
…
```

Další informace

Vliv na režim řízení pohybu po dráze

Příkazy EXTOPEN, WRITE a EXTCLOSE spouštějí zastavení předběžného zpracování a v důsledku toho přerušují režim řízení pohybu po dráze.

Chování při vyhledávání bloku

V průběhu "Vyhledávání bloku s výpočtem" nemá funkce WRITE žádný výstup. Příkazy EXTOPEN a EXTCLOSE jsou však shromažďovány a -- poté, co bylo cíle vyhledávání dosaženo -- stisknutím tlačítka NC-Start se aktivuje příslušný stav. Následující příkazy WRITE takto naleznou přesně stejné prostředí jako při normálním zpracovávání programu.

Při vyhledávání bloku s výpočtem v režimu "testování programu" (SERUPRO) jsou příkazy EXTOPEN, WRITE a EXTCLOSE prováděny stejně jako při normálním zpracovávání programu.

Chování při resetu

Na konci výrobního programu a při resetu kanálu jsou všechna externí zařízení/soubory otevřené v tomto kanálu uzavřeny.

Dostupná externí zařízení

Jako externí zařízení/soubory mohou být k dispozici:

- Soubory na lokální kompaktní Flash-kartě

  Lokální kompaktní Flash-kartou je míněna paměť, na kterou se lze z HMI obracet prostřednictvím symbolického identifikátoru LOCAL_DRIVE. U systému SINUMERIK 840D sl se jedná o lokální diskovou jednotku, u systému SINUMERIK 828D je to uživatelská kompaktní Flash-karta.

  Poznámka

  Aby bylo možné používat výstup na zařízení LOCAL_DRIVE, je v systému SINUMERIK 840D sl zapotřebí, aby byl instalován volitelný doplněk "Dodatečných xxx MB uživatelské paměti HMI na CF kartě NCU". U systému SINUMERIK 828D musí být uživatelská kompaktní Flash-karta k dispozici, volitelný doplněk zde však není nutný.

- Soubory na síťové jednotce
- Rozhraní RS-232

  Poznámka

  Aby bylo možné pracovat s výstupem na rozhraní RS-232, je u systému SINUMERIK 840D sl zapotřebí volitelný modul NCU "Rozhraní RS-232". U systému SINUMERIK 828D se výstup uskutečňuje na integrované rozhraní RS-232 (předpoklad: MD51233 $MNS_ENABLE_GSM_MODEM = 0).

Konfigurace

Konfigurace externích zařízení, které se mají používat, se uskutečňuje v souboru /oem/sinumerik/nck/extdev.ini, příp. /user/sinumerik/nck/extdev.ini. Jsou-li k dispozici oba soubory, mají záznamy v uživatelské oblasti přednost. Soubor může být ošetřován v systémové oblasti "Uvádění do provozu", ve složce "Systémová data/CF-karta".

Poznámka

Aby bylo možné používat LOCAL_DRIVE a CYC_DRIVE, nejsou v souboru extdev.ini zapotřebí žádné konfigurační úpravy. Pokud je aktivován příslušný parametr, příp. pokud je uživatelská kompaktní Flash-karta k dispozici, jsou obě zařízení vždy k dispozici.

V části [ExternalDevices] souboru extdev.ini jsou definována/uvedena externí zařízení, na která se lze obracet. Jako zařízení mohou být uvedeny přístroje se sériovým portem (/dev/v24) a až devět souborů nebo adresářů (/dev/ext/1…9). Způsob zápisu odpovídá konvencím v Linuxu. Řádky, které začínají znakem ";", jsou komentáře a jsou přeskakovány.

S výjimkou /dev/v24 mohou být zařízení definována jako cesta do adresáře - kdy je na konci připojen znak "/" - nebo jako cesta k souboru - kdy je připojena úplná kvalifikovaná cesta zakončená názvem souboru (aniž by byl na konci znak "/"). U zařízení se zadanou cestou do adresáře musí být při použití ve výrobním programu uveden název souboru (cesta).

S výjimkou /dev/v24 se definice zařízení uskutečňuje pomocí třech údajů pro "server", "cestu" a volitelný "režim zápisu", které jsou odděleny čárkami.

U souborů, příp. adresářů (potom to platí pro všechny soubory v adresáři) může být zadáno, zda se má soubor po svém otevření přepisovat ("O" = Overwrite) nebo zda se mají výstupy do souboru připojovat na jeho konec ("A" = Append). Standardní nastavení je "A". Neexistující soubor/adresář se při otevírání nově založí.

Pro zařízení "Rozhraní RS-232" se zadávají pouze parametry pro přenosovou rychlost, datové bity, stop-bity, paritu, protokol a v případě potřeby konec a to v tomto pořadí.

Pro soubory, které jsou vytvářeny/ukládány na jednotce LOCAL_DRIVE, se prostřednictvím parametru LOCAL_DRIVE_MAX_FILESIZE zadává maximální velikost souboru v bytech, která pak platí jednotně pro všechny soubory. Velikost souboru se kontroluje při zpracovávání příkazu `EXTOPEN` v režimu Append. Pokud si přejete, je možné pomocí parametru LOCAL_DRIVE_FILE_MODE definovat režim zápisu ("O" = Overwrite, "A" = Append). Standardní nastavení je "A".

Poznámka

Pro konfigurační soubor extdev.ini je k v adresáři /siemens/sinumerik/nck k dispozici šablona pro zkopírování.

Poznámka

Změny provedené v souboru extdev.ini vstupují v platnost až po novém spuštění/náběhu NCK.

Poznámka

Zařízení připojené přes USB

V systému SINUMERIK 828D může být jako cílové zařízení definováno také "usb" (bez udání oddílu!), což znamená jednotku USB připojenou do konektoru na čelním panelu. K zařízení připojenému na USB je možné mít ve výrobním programu přístup pouze nepřímo prostřednictvím symbolického identifikátoru zařízení "`/dev/ext/x`".

V systému SINUMERIK 840D sl mohou být jako USB zařízení v konfiguraci definovány pouze staticky připojená rozhraní USB dané TCU. Konfigurace se uskutečňuje pomocí zápisu SERVER:/PATH, přičemž "server" je specifikován v tom smyslu, jak bylo popsáno výše, kdy SERVER označuje název TCU a /PATH rozhraní USB. Pro přístup k jednotlivým rozhraním USB jedné TCU jsou používány identifikátory "dev0-0", "dev0-1", "dev1-0". Zadání cesty vždy začíná "/Partition", přičemž tento parametr může být zadán prostřednictvím dvoumístného čísla oddílu nebo svého názvu oddílu. V případě potřeby může být rozšířen pomocí cesty k souboru až po požadovaný cíl, tedy např.:

/dev/ext/8 = "TCU4:/dev0-0, /01/, A"

/dev/ext/8 = "TCU4:/dev0-0, /01/mydir.dir/"

/dev/ext/8 = "TCU4:/dev0-0, /myfirstpartition/Mydir.dir/myfile.txt, O"

Příklady:

```
[ExternalDevices]
; řádek komentáře
; například pro RS-232
; /dev/v24 = "9600, 8, 1, none, rts [, etx]"
; příklady pro síťové jednotky
; /dev/ext/1 = "//[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE/, /, A"
; /dev/ext/2 = "//[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE, /myfile.txt, O"
; /dev/ext/3 = "//[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE, /mydir/, A"
; /dev/ext/4 = "SERVER:/dev0-0, /01/, A"
; …
; pouze SINUMERIK 828 (USB)
; /dev/ext/9 = "usb, / [ , O]"
; předdefinované nastavení: číslo oddílu = 1

; pouze SIEMENS
; /dev/cyc/1= "//[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE, /mydir/, A"
; /dev/cyc/2= "//[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE/mydir, /, A"

LOCAL_DRIVE_MAX_FILESIZE = 50000
LOCAL_DRIVE_FILE_MODE = "O"
```

Funkce parametru <režim zápisu> u příkazu EXTOPEN

V důsledku zadání režimu zápisu jak při konfiguraci v souboru extdev.ini, tak také při volání příkazu `EXTOPEN` se mohou vyskytnout konflikty s oprávněními, které při volání příkazu `EXTOPEN` mohou případně skončit s chybou.

Hodnota ze souboru extdev.ini	Hodnota parametru u příkazu EXTOPEN		
	"OVR"	"APP"	-
"O"	O	Chyba	O
"A"	Chyba	A	A
-	O	A	A
	Vysvětlení: O: V platnosti je režim "přepisování". A: V platnosti je režim "připojování". Chyba: Volání funkce EXTOPEN potvrzeno s chybou.		

LOCAL_DRIVE: atribut souboru

Soubory založené pomocí funkce `EXTOPEN` na jednotce LOCAL_DRIVE mají následující atributy:

- Vlastník: "user" Oprávnění k zápisu/čtení nastaveny
- Skupina: "operator" Oprávnění k zápisu/čtení nastaveny

Maximální počet otevřených externích zařízení

Prostřednictvím všech NC kanálů může být současně otevřeno maximálně 10 výstupních zařízení. Kromě toho existují ještě dvě položky rezervované pro cykly firmy Siemens.

K těmto zařízením může být současně aktivních maximálně 5 úloh.

14.11 Alarmy (SETAL)

Funkce

V NC programu mohou být aktivovány alarmy, které se pak vypisují na uživatelském rozhraní ve zvláštním poli. S alarmem je vždy spojena nějaká reakce řídícího systému závisející na kategorii alarmu.

Literatura:
Pokud budete potřebovat podrobnější informace o reakcích na alarm, nahlédněte do Příručky pro uvádění do provozu.

Syntaxe

```
SETAL(<číslo alarmu>[,<znakový řetězec>])
```

Význam

`SETAL`:	Klíčové slovo pro programování alarmu. Příkaz `SETAL` musí být naprogramován v samostatném NC-bloku.
`<číslo alarmu>`:	Proměnná typu INT. Obsahuje číslo alarmu. Platným rozsahem alarmových čísel je interval 60000 až 69999, z čehož 60000 až 64999 je vyhrazeno pro cykly Siemens, zatímco 65000 až 69999 je k dispozici uživateli.
`<Řetězec znaků>`:	Při programování alarmů uživatelských cyklů může být kromě alarmového čísla zadán ještě řetězec znaků obsahující až 4 parametry. V těchto parametrech mohou být definovány proměnné uživatelské texty. Budou Vám k dispozici ale i následující předdefinované parametry:

Parametry	Význam
%1	Číslo kanálu
%2	Číslo bloku, návěští
%3	Textový index pro alarmy cyklů
%4	Pomocný parametr alarmu

Poznámka

Texty alarmu musí být konfigurovány v uživatelském rozhraní.

Poznámka

Jestliže se má alarm vypisovat v aktivním jazyce uživatelského rozhraní, potřebuje uživatel informace o tom, který jazyk je momentálně na HMI nastaven. Tento údaj je možné ve výrobním programu a v synchronních akcích zjistit pomocí systémové proměnné $AN_LANGUAGE_ON_HMI (viz "Aktuální jazyk v HMI [Strana 898]").

Příklad

Programový kód	Komentář
...	
N100 SETAL (65000)	; Nastavení alarmu č. 65000.
...	

14.12 Rozšířené zastavování a odjíždění nezávislé na pohonu (ESR)

14.12.1 Konfigurace zastavování nezávislého na pohonu (ESRS)

Funkce

Pomocí funkce `ESRS(...)` se nastavuje konfigurace parametrů pohonu pro "zastavování" v rámci funkce ESR nezávislé na pohonu.

Syntaxe

```
ESRS(<osa_1>,<doba zastavení_1>[,...,<osa_n>,<doba zastavení_n>])
```

Význam

`ESRS(...)`: Funkce pro zápis parametrů pohonu pro funkci ESR "zastavení".

Funkce:

- musí se nacházet v samostatném bloku
- zastavuje předběžné zpracování
- nemůže se používat v synchronních akcích

`<osa_1>, ..., <osa_n>`: Osa, pro kterou má být konfigurováno zastavování nezávislé na pohonu.

V pohonu se pro tuto osu zapisuje parametr pohonu p0888 (konfigurace):

p0888 = 1

Typ:	AXIS
Rozsah hodnot:	Identifikátor kanálové osy

`<doba zastavení_1>, ..., <doba zastavení_n>`: Časový interval po který se pohon poté, co se vyskytla chyba, pohybuje konstantně s momentální požadovanou hodnotou otáček.

V pohonu se pro uvedenou osu zapisuje parametr pohonu p0892 (časový interval):

p0892 = <doba zastavení>

Jednotka:	s
Typ:	REAL
Rozsah hodnot:	0.00 - 20.00

V jednom volání funkce může být naprogramováno maximálně 5 os; n = 5

Literatura

Příručka Popis funkcí, Speciální funkce; Rozšířené zastavování a odjíždění (R3), ESR nezávislé na pohonu

14.12.2 Konfigurace odjíždění nezávislého na pohonu (ESRR)

Funkce

Pomocí funkce `ESRR(...)` se nastavuje konfigurace parametrů pohonu pro "odjíždění" v rámci funkce ESR nezávislé na pohonu.

Syntaxe

```
ESRR(<osa_1>,<návratová dráha_1>,<návratová
rychlost_1>[,...,<osa_n>,<návratová dráha_n>,<návratová
rychlost_n>])
```

Význam

`ESRR(...)`: Funkce pro zápis parametrů pohonu pro funkci ESR "odjíždění".

Funkce:

- musí se nacházet v samostatném bloku
- zastavuje předběžné zpracování
- nemůže se používat v synchronních akcích

`<osa_1>, ..., <osa_n>`: Osa, pro kterou má být konfigurováno odjíždění nezávislé na pohonu.

V pohonu se pro tuto osu zapisuje parametr pohonu p0888 (konfigurace):

p0888 = 2

Typ:	AXIS
Rozsah hodnot:	Identifikátor kanálové osy

`<návratová dráha_1>, ..., <návratová dráha_n>`: Návratová dráha se pro pohon přepočítává na návratové otáčky. Hodnota se pro uvedenou osu zapisuje do parametru pohonu p0893 (otáčky):

p0893 = (<návratová dráha_n> přepočítaná na návratové otáčky)

Jednotka:	mm/min, palce/min, stupně/min (v závislosti na jednotkách nastavených pro osu)
Typ:	REAL
Rozsah hodnot:	MIN - MAX

`<návratová rychlost_1>, ..., <návratová rychlost_n>:`	Návratová rychlost se pro pohon přepočítává na časový interval. Hodnota se pro uvedenou osu zapisuje do parametru pohonu p0892 (časový interval) [s]: p0892 = <návratová dráha_n> / <návratová rychlost_n>	
	Jednotka:	mm/min, palce/min, stupně/min (v závislosti na jednotkách nastavených pro osu)
	Typ:	REAL
	Rozsah hodnot:	0.00 - MAX

V jednom volání funkce může být naprogramováno maximálně 5 os; n = 5

Literatura

Příručka Popis funkcí, Speciální funkce; Rozšířené zastavování a odjíždění (R3), ESR nezávislé na pohonu

Vlastní programy pro oddělování třísky 15

15.1 Podporované funkce pro oddělování třísky

Funkce

Pro oddělování třísky jsou Vám nabízeny již hotové obráběcí cykly. Kromě toho máte možnost pomocí funkcí uvedených v následujících odstavcích sestavovat své vlastní programy pro obrábění:

- Sestavování kontury (CONTPRON)
- Sestavování kódované tabulky kontury (CONTPRON)
- Ukončení přípravy kontury (EXECUTE)
- Zjištění průsečíku mezi dvěma konturovými prvky (INTERSEC)

  (jen pro tabulky, které byly vytvořeny pomocí funkce CONTPRON.)
- Zpracovávání konturových prvků v tabulce blok po bloku (EXECTAB)

  (jen pro tabulky, které byly vytvořeny pomocí funkce CONTPRON.)
- Výpočet parametrů kruhu (CALCDAT)

Poznámka

Tyto funkce můžete používat nejen pro oddělování třísky, jsou použitelné univerzálně.

Předpoklady

Před vyvoláváním funkcí CONTPRON nebo CONTDCON musíte:

- Najet na počáteční bod, který umožňuje bezkolizní obrábění.
- Deaktivovat korekci rádiusu nástroje pomocí příkazu `G40`.

15.2 Sestavování kontury (CONTPRON)

Funkce

Pomocí příkazu `CONTPRON` se aktivuje příprava kontury. Následně vyvolávané NC-bloky nejsou zpracovávány, ale jsou rozděleny do jednotlivých pohybů a uloženy do tabulky kontury. Každému konturovému prvku odpovídá řádek ve dvourozměrném poli tabulky kontury. Systém uvádí počet zjištěných podříznutí.

Syntaxe

Aktivování přípravy kontury:
`CONTPRON(<tabulka kontury>,<druh opracování>,<podříznutí>, <směr opracování>)`

Deaktivování přípravy kontury a návrat zpět do normálního režimu zpracování:
`EXECUTE(<FEHLER>)`

Viz " Ukončení přípravy kontury (EXECUTE) "

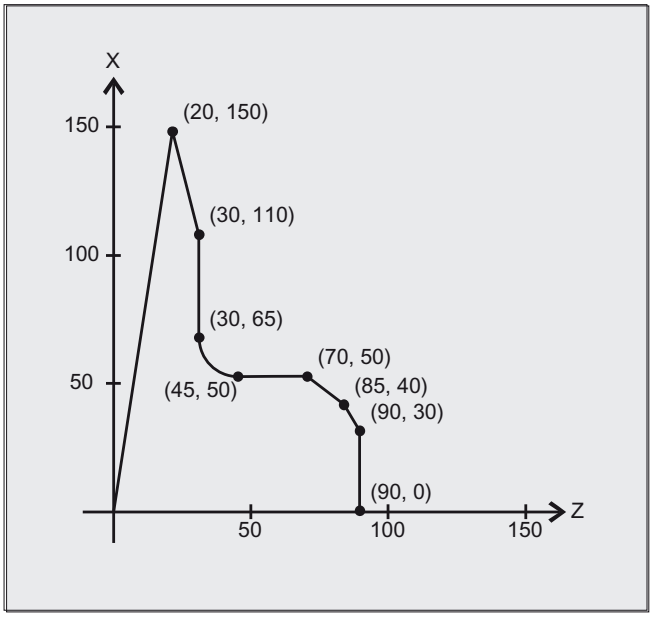
Význam

`CONTPRON`	Příkaz pro aktivování přípravy kontury za účelem sestavení tabulky kontury		
`<tabulka kontury>`	Název tabulky kontury		
`<druh opracování>`	Parametr pro stanovení druhu opracování		
	Typ:	CHAR	
	Hodnota:	`"G"`	podélné soustružení: vnitřní opracování
		`"L"`	podélné soustružení: vnější opracování
		`"N"`	příčné soustružení: vnitřní opracování
		`"P"`	příčné soustružení: vnější opracování
`<podříznutí>`	Varianta výsledku pro počet zjištěných prvků podříznutí		
	Typ:	INT	
`<směr opracování>`	Parametr pro stanovení směru opracování		
	Typ:	INT	
	Hodnota:	0	Ve směru přípravy kontury (standardní hodnota)
		1	V obou směrech vzhledem ke směru přípravy kontury

Příklad 1

Sestavení tabulky kontury s následujícími parametry:

- Název "KTAB"
- max. 30 konturových prvků (kruhy, přímky)
- Proměnná pro počet zjištěných prvků podříznutí
- Proměnní pro chybová hlášení



NC program:

Programový kód	**Komentář**
N10 DEF REAL KTAB[30,11]	; Tabulka kontury s názvem KTAB a s max. 30 konturovými prvky, hodnota parametru 11 (počet sloupců v tabulce) je pevná hodnota.
N20 DEF INT ANZHINT	; Proměnná pro počet prvků podříznutí s názvem ANZHINT.
N30 DEF INT FEHLER	; Proměnná pro zpětná chybová hlášení (0 = žádná chyba, 1 = chyba).
N40 G18	
N50 CONTPRON(KTAB,"G",ANZHINT)	; Aktivování přípravy kontury.
N60 G1 X150 Z20 N70 X110 Z30 N80 X50 RND=15 N90 Z70 N100 X40 Z85 N110 X30 Z90 N120 X0	; N60 až N120: popis kontury
N130 EXECUTE(FEHLER)	; Ukončení plnění tabulky kontury, přepnutí do normálního programového režimu.
N140 …	; Další zpracovávání tabulky.

Tabulka kontury KTAB:

Index Řádek	Sloupec									
(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
7	7	11	0	0	20	150	0	82.40535663	0	0
0	2	11	20	150	30	110	-1111	104.0362435	0	0
1	3	11	30	110	30	65	0	90	0	0
2	4	13	30	65	45	50	0	180	45	65
3	5	11	45	50	70	50	0	0	0	0
4	6	11	70	50	85	40	0	146.3099325	0	0
5	7	11	85	40	90	30	0	116.5650512	0	0
6	0	11	90	30	90	0	0	90	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

Vysvětlení obsahu sloupců

(0) Ukazatel na nejbližší konturový prvek (na číslo jeho řádku)

(1) Ukazatel na předcházející konturový prvek

(2) Kódování režimu kontury pro pohyb

Možné hodnoty pro X = abc

a = 10^2 G90 = 0 G91 = 1

b = 10^1 G70 = 0 G71 = 1

c = 10^0 G0 = 0 G1 = 1 G2 = 2 G3 = 3

(3), (4) Počáteční bod konturového prvku
(3) = abscisa, (4) = ordináta v aktuální rovině

(5), (6) Koncový bod konturového prvku
(5) = abscisa, (6) = ordináta v aktuální rovině

(7) Ukazatel max./min.: označuje lokální maxima a minima na kontuře

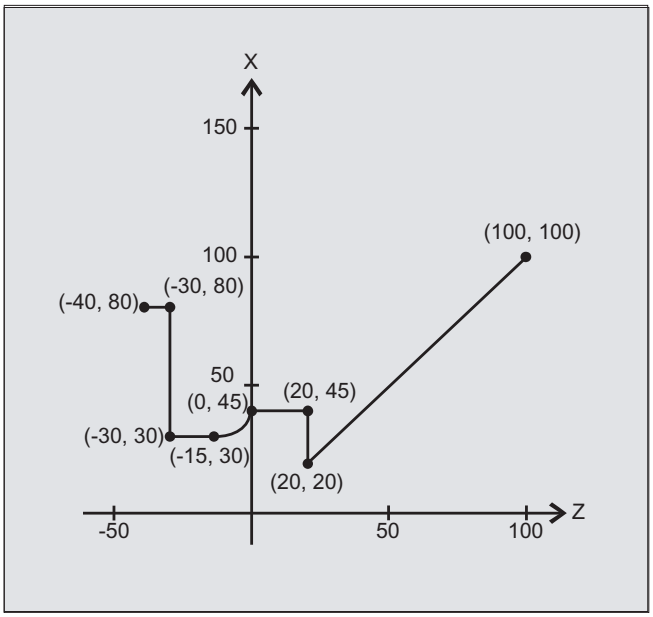
(8) Maximální hodnota mezi konturovým prvkem a abscisou)při podélném obrábění), příp. ordinátou (při příčném obrábění). Úhel je závislý na naprogramovaném druhu opracování.

(9), (10) Souřadnice středu konturového prvku, jestliže se jedná o kruhový blok.
(9) = abscisa, (10) = ordináta

Příklad 2

Sestavení tabulky kontury s následujícími parametry:

- Název KTAB
- max. 92 konturových prvků (kruhy, přímky)
- Provozní režim: podélné obrábění, vnější obrábění
- Příprava směrem dopředu a dozadu



NC program:

Programový kód	**Komentář**
N10 DEF REAL KTAB[92,11]	; Tabulka kontury s názvem KTAB a s max. 92 konturovými prvky, hodnota parametru 11 je pevná hodnota.
N20 DEF CHAR BT="L"	; Druh opracování CONTPRON: podélné obrábění, vnější obrábění
N30 DEF INT HE=0	; Počet prvků podříznutí = 0
N40 DEF INT MODE=1	; Příprava směrem dopředu a dozadu
N50 DEF INT ERR=0	; Chybové hlášení
...	
N100 G18 X100 Z100 F1000	
N105 CONTPRON(KTAB,BT,HE,MODE)	; Aktivování přípravy kontury.
N110 G1 G90 Z20 X20 N120 X45 N130 Z0 N140 G2 Z-15 X30 K=AC(-15) I=AC(45) N150 G1 Z-30 N160 X80 N170 Z-40	
N180 EXECUTE(ERR)	; Ukončení plnění tabulky kontury, přepnutí do normálního programového režimu.
...	

Tabulka kontury KTAB:

Po ukončení přípravy kontury je kontura k dispozici v obou směrech.

Index	Sloupec										
Řádek	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
0	6[1)]	7[2)]	11	100	100	20	20	0	45	0	0
1	0[3)]	2	11	20	20	20	45	-3	90	0	0
2	1	3	11	20	45	0	45	0	0	0	0
3	2	4	12	0	45	-15	30	5	90	-15	45
4	3	5	11	-15	30	-30	30	0	0	0	0
5	4	7	11	-30	30	-30	45	-1111	90	0	0
6	7	0[4)]	11	-30	80	-40	80	0	0	0	0
7	5	6	11	-30	45	-30	80	0	90	0	0
8	1[5)]	2[6)]	0	0	0	0	0	0	0	0	0
	...										
83	84	0[7)]	11	20	45	20	80	0	90	0	0
84	90	83	11	20	20	20	45	-1111	90	0	0
85	0[8)]	86	11	-40	80	-30	80	0	0	0	0
86	85	87	11	-30	80	-30	30	88	90	0	0
87	86	88	11	-30	30	-15	30	0	0	0	0
88	87	89	13	-15	30	0	45	-90	90	-15	45
89	88	90	11	0	45	20	45	0	0	0	0
90	89	84	11	20	45	20	20	84	90	0	0
91	83[9)]	85[10)]	11	20	20	100	100	0	45	0	0

Vysvětlení obsahu sloupců a poznámky k řádkům 0, 1, 6, 8, 83, 85 a 91

Pokud jde o obsah sloupců, platí vysvětlení uvedená u příkladu 1.

Vždy v řádku tabulky 0:

1) Předcházející blok: Řádek n obsahuje konec tabulky kontury směrem dopředu

2) Následující blok: Řádek n je koncem tabulky kontury směrem dopředu

Pokaždé v rámci prvku kontury směrem dopředu:

3) Předcházející blok: začátek kontury (směrem dopředu)

4) Následující blok: konec kontury (směrem dopředu)

Vždy na řádku konec tabulky kontury (směrem dopředu) + 1:

5) Předcházející blok: Počet podříznutí směrem dopředu

6) Následující blok: Počet podříznutí směrem dozadu

Pokaždé v rámci prvku kontury směrem dozadu:

7) Následující blok: konec kontury (směrem dozadu)

8) Předcházející blok: začátek kontury (směrem dozadu)

Vždy na posledním řádku tabulky:

9) Předcházející blok: Řádek n je počátek tabulky kontury (směrem dozadu)

10) Následující blok: Řádek n obsahuje počátek kontury (směrem dozadu)

Další informace

Povolené příkazy pohybu, souřadný systém

Pro programování kontury jsou přípustné následující příkazy G-funkcí:

- G-skupina 1: `G0`, `G1`, `G2`, `G3`

Kromě toho jsou k dispozici následující možnosti:

- Zaoblení a faseta
- Programování kruhových oblouků pomocí příkazů `CIP` a `CT`

Funkce pro spliny, polynomy a závity mají za následek chybu.

Změny souřadného systému prostřednictvím aktivování framu jsou v úseku mezi příkazy `CONTPRON` a `EXECUTE` nepřípustné. Totéž platí i v případě přepnutí mezi funkcemi `G70` a `G71`, příp. `G700` a `G710`.

Výměna geometrických os pomocí příkazu `GEOAX` v době, kdy probíhá příprava tabulky kontury má za následek alarm.

Prvky podříznutí

Popis kontury jednotlivých prvků podříznutí se může dle Vašeho přání uskutečnit buď v podprogramu nebo v jednotlivých blocích.

Oddělování třísky nezávisle na naprogramovaném směru kontury

Příprava kontury pomocí funkce `CONTPRON` byla rozšířena tak, aby po svém vyvolání byla tabulka kontury k dispozici nezávisle na směru, ve kterém byla naprogramována.

15.3 Sestavování kódované tabulky kontury (CONTPRON)

Funkce

Při přípravě kontury aktivované příkazem `CONTDCON` jsou následně vyvolávané NC-bloky ukládány kódovaně do tabulky se 6 sloupci, což je výhodné z hlediska využití paměti. Každému konturovému prvku odpovídá řádek v tabulce kontury. Ze znalosti níže uvedených pravidel pro kódování můžete např. z řádků tabulky sestavit program v kódu DIN pro cyklus. V řádku tabulky s číslem 0 jsou uložena data výchozího bodu.

Syntaxe

Aktivování přípravy kontury:
`CONTDCON(<tabulka kontury>,<směr opracování>)`

Deaktivování přípravy kontury a návrat zpět do normálního režimu zpracování:
`EXECUTE(<FEHLER>)`

Viz " Ukončení přípravy kontury (EXECUTE) "

Význam

`CONTDCON`	Příkaz pro aktivování přípravy kontury za účelem sestavení kódované tabulky kontury		
`<tabulka kontury>`	Název tabulky kontury		
`<směr opracování>`	Parametr pro stanovení směru opracování		
	Typ:	INT	
	Hodnota:	0	Příprava kontury podle posloupnosti konturových bloků (standardní hodnota)
		1	nepřípustné

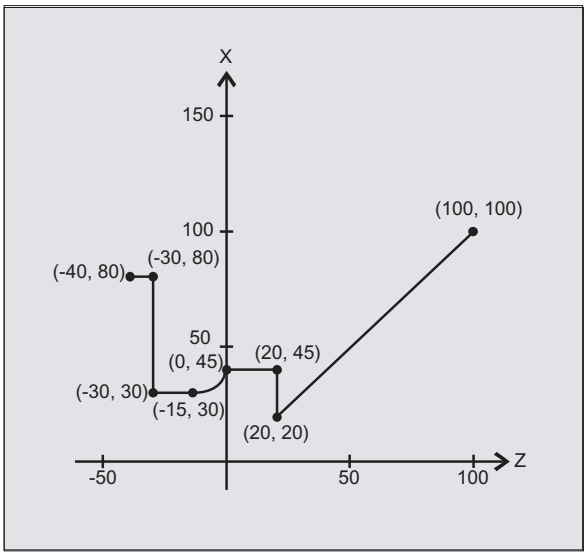
Poznámka

Spektrum G-kódů, které jsou přípustné pro funkci `CONTDCON` v úseku programu, který se má uložit do tabulky, je širší než v případě příkazu `CONTPRON`. Kromě toho jsou s každým úsekem kontury ukládány také posuvy a typ posuvu.

Příklad

Sestavení tabulky kontury s následujícími parametry:

- Název "KTAB"
- Konturové prvky (kruhy, přímky)
- Provozní režim: Soustružení
- Směr obrábění: Směrem dopředu



NC program:

Programový kód	**Komentář**
N10 DEF REAL KTAB[9,6]	; Tabulka kontury s názvem KTAB a s 9 řádky, které umožňují 8 konturových bloků. Hodnota parametru 6 (počet sloupců v tabulce) je pevně danou veličinou.
N20 DEF INT MODE = 0	; Proměnná pro stanovení směru opracování Standardní hodnota 0: jen v naprogramovaném směru kontury.
N30 DEF INT ERROR = 0	; Proměnná pro chybové hlášení
...	
N100 G18 G64 G90 G94 G710	
N101 G1 Z100 X100 F1000	
N105 CONTDCON (KTAB, MODE)	; Vyvolání přípravy tabulky (parametr MODE lze vypustit).
N110 G1 Z20 X20 F200 N120 G9 X45 F300 N130 Z0 F400	; Popis kontury
N140 G2 Z-15 X30 K=AC(-15) I=AC(45)F100	
N150 G64 Z-30 F600 N160 X80 F700 N170 Z-40 F800	
N180 EXECUTE(ERROR)	; Ukončení plnění tabulky kontury, přepnutí do normálního programového režimu.
...	

Tabulka kontury KTAB:

	Index sloupce					
	0	1	2	3	4	5
Index řádku	Režim kontury	Koncový bod abscisa	Koncový bod ordináta	Střed abscisa	Střed ordináta	Posuv
0	30	100	100	0	0	7
1	11031	20	20	0	0	200
2	111031	20	45	0	0	300
3	11031	0	45	0	0	400
4	11032	-15	30	-15	45	100
5	11031	-30	30	0	0	600
6	11031	-30	80	0	0	700
7	11031	-40	80	0	0	800
8	0	0	0	0	0	0

Vysvětlení obsahu sloupců

Řádek 0: Kódování pro **počáteční bod**

Sloupec 0: 10^0 (místo jednotek): G0 = 0
10^1 (místo desítek): G70 = 0, G71 = 1, G700 = 2, G710 = 3
Sloupec 1: Počáteční bod, abscisa
Sloupec 2: Počáteční bod, ordináta
Sloupec 3-4: 0
Sloupec 5: Index řádku posledního úseku kontury v tabulce

Řádky 1-n: Záznamy **úseků kontury**

Sloupec 0: 10^0 (místo jednotek): G0 = 0, G1 = 1, G2 = 2, G3 = 3
10^1 (místo desítek): G70 = 0, G71 = 1, G700 = 2, G710 = 3
10^2 (místo stovek): G90 = 0, G91 = 1
10^3 (místo tisíců): G93 = 0, G94 = 1, G95 = 2, G96 = 3
10^4 (místo desítek tisíců): G60 = 0, G44 = 1, G641 = 2, G642 = 3
10^5 (místo stovek tisíců): G9 = 1
Sloupec 1: Koncový bod, abscisa
Sloupec 2: Koncový bod, ordináta
Sloupec 3: Střed kruhu na abscise při kruhové interpolaci
Sloupec 4: Střed kruhu na ordinátě při kruhové interpolaci
Sloupec 5: Posuv

Další informace

Povolené příkazy pohybu, souřadný systém

Pro programování kontury jsou přípustné následující G-skupiny a následující příkazy G-funkcí:

G-skupina 1:	G0, G1, G2, G3
G-skupina 10:	G60, G64, G641, G642
G-skupina 11:	G9
G-skupina 13:	G70, G71, G700, G710
G-skupina 14:	G90, G91
G-skupina 15:	G93, G94, G95, G96, G961

Kromě toho jsou k dispozici následující možnosti:

- Zaoblení a faseta
- Programování kruhových oblouků pomocí příkazů CIP a CT

Funkce pro spliny, polynomy a závity mají za následek chybu.

Změny souřadného systému prostřednictvím aktivování framu jsou v úseku mezi příkazy CONTDCON a EXECUTE nepřípustné. Totéž platí i v případě přepnutí mezi funkcemi G70 a G71, příp. G700 a G710.

Výměna geometrických os pomocí příkazu GEOAX v době, kdy probíhá příprava tabulky kontury má za následek alarm.

Směr obrábění

Při opracovávání pomocí tabulky kontury sestavené pomocí funkce CONTDCON se smí postupovat ve směru naprogramování kontury.

15.4 Zjištění průsečíku mezi dvěma konturovými prvky (INTERSEC)

Funkce

Funkce `INTERSEC` zjišťuje dvou normovaných konturových prvků z tabulek kontury sestavených pomocí funkce `CONTPRON`.

Syntaxe

```
<Status>=INTERSEC(<tabulka kontury_1>[<prvek kontury_1>],
<tabulka kontury_2>[<prvek kontury_2>],<průsečík>,<druh opracování>)
```

Význam

`INTERSEC`	Klíčové slovo pro výpočet průsečíku dvou konturových prvků z tabulek kontury vytvořených pomocí funkce `CONTPRON`.		
`<Status>`	Proměnná pro stav průsečíku		
	Typ:	BOOL	
	Hodnota:	TRUE	Průsečík nalezen
		FALSE	Žádný průsečík nenalezen
`<tabulka kontury_1>`	Název první tabulky kontury		
`<prvek kontury_1>`	Číslo konturového prvku z první tabulky		
`<tabulka kontury_2>`	Název druhé tabulky kontury		
`<prvek kontury_2>`	Číslo konturového prvku ze druhé tabulky		
`<průsečík>`	Souřadnice průsečíku v aktivní rovině (`G17` / `G18` / `G19`)		
	Typ:	REAL	
`<druh opracování>`	Parametr pro stanovení druhu opracování		
	Typ:	INT	
	Hodnota:	0	Výpočet průsečíku v aktivní rovině dané parametrem 2 (standardní hodnota)
		1	Výpočet průsečíku nezávisle na předávané rovině

Poznámka

Mějte prosím na paměti, že proměnné musí být před svým použitím definovány.

Předávání kontur vyžaduje, aby byly dodrženy hodnoty definované příkazem CONTPRON.

Parametry	Význam
2	Kódování režimu kontury pro pohyb
3	Počáteční bod kontury, abscisa
4	Počáteční bod kontury, ordináta
5	Koncový bod kontury, abscisa
6	Koncový bod kontury, ordináta
9	Souřadnice středu pro abscisu (jen v případě kruhové kontury)
10	Souřadnice středu pro ordinátu (jen v případě kruhové kontury)

Příklad

Zjištění průsečíku konturového prvku 3 z tabulky TABNAME1 a konturového prvku 7 z tabulky TABNAME2. Souřadnice průsečíku v aktivní rovině jsou uloženy v proměnné ISCOORD (1. prvek = abscisa, 2. prvek = ordináta). Jestliže neexistuje žádný průsečík, uskuteční se skok na návěští KEINSCH (žádný průsečík nenalezen).

```
Programový kód                                                    Komentář
DEF REAL TABNAME1[12,11]                                          ; Tabulka kontury 1
DEF REAL TABNAME2[10,11]                                          ; Tabulka kontury 2
DEF REAL ISCOORD[2]                                               ; Proměnná pro souřadnice
                                                                    průsečíku.
DEF BOOL ISPOINT                                                  ; Proměnná pro stav průsečíku.
DEF INT MODE                                                      ; Proměnná pro druh obrábění.
…
MODE=1                                                            ; Výpočet nezávisle na aktivní
                                                                    rovině.
N10 ISPOINT=INTERSEC(TABNAME1[3],TABNAME2[7],ISCOORD,MODE)        ; Volání průsečíku konturových
                                                                    prvků.
N20 IF ISPOINT==FALSE GOTOF KEINSCH                               ; Skok na návěští KEINSCH.
…
```

15.5 Pohyb po konturových prvcích v tabulce blok po bloku (EXECTAB)

Funkce

Pomocí příkazu `EXECTAB` můžete uskutečnit pohyb blok po bloku po konturových prvcích tabulky, která byla vytvořena například příkazem `CONTPRON`.

Syntaxe

```
EXECTAB(<tabulka kontury>[<konturový prvek>])
```

Význam

`EXECTAB`	Příkaz pro aktivování pohybu po konturových prvcích
`<tabulka kontury>`	Název tabulky kontury
`<konturový prvek>`	Číslo konturového prvku

Příklad

Má být uskutečněn pohyb blok po bloku po konturových prvcích 0 až 2 z tabulky KTAB.

Programový kód	**Komentář**
`N10 EXECTAB(KTAB[0])`	`; Pohyb po prvku 0 z tabulky kontury KTAB.`
`N20 EXECTAB(KTAB[1])`	`; Pohyb po prvku 1 z tabulky kontury KTAB.`
`N30 EXECTAB(KTAB[2])`	`; Pohyb po prvku 2 z tabulky kontury KTAB.`

15.6 Výpočet parametrů kruhu (CALCDAT)

Funkce

Pomocí příkazu `CALCDAT` můžete ze tří nebo čtyř známých bodů na kružnici vypočítat její rádius a souřadnice jejího středu. Zadávané body musí být různé. V případě 4 bodů, které neleží na kružnici úplně přesně, se pro střed a rádius kružnice vypočítá střední hodnota.

Syntaxe

```
<Status>=CALCDAT(<body kružnice>[<počet>,<druh>],<počet>,<výsledek>)
```

Význam

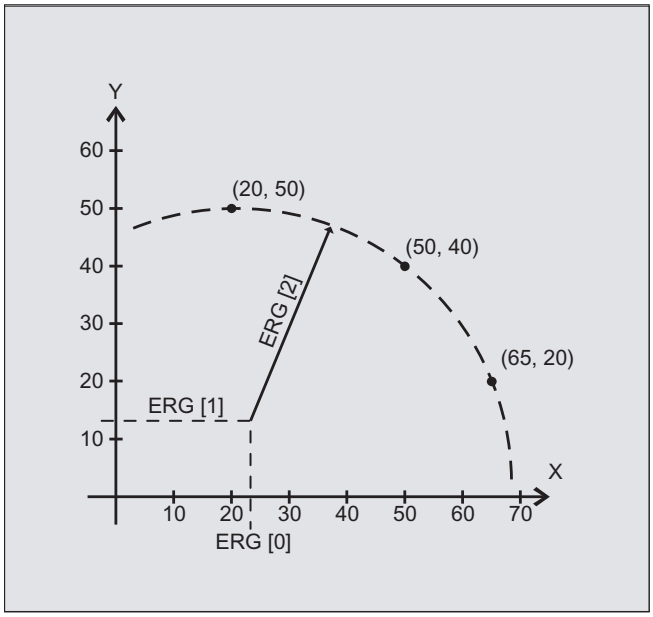
`CALCDAT`	Příkaz pro výpočet rádiusu a souřadnic středu kružnice na základě 3 nebo 4 bodů.		
`<Status>`	Proměnná pro stav výpočtu kružnice		
	Typ:	BOOL	
	Hodnota:	TRUE	Zadané body leží na kružnici.
		FALSE	Zadané body **neleží** na kružnici.
`<body kružnice>[]`	Proměnná pro zadání bodů na kružnici s následujícími parametry:		
	`<počet>`	Počet bodů na kružnici (3 nebo 4)	
	`<druh>`	Druh zadání souřadnic, např. 2 pro souřadnice 2 bodů	
`<počet>`	Parametr pro zadání počtu bodů použitých pro výpočet (3 nebo 4)		
`<výsledek>[3]`	Proměnná pro výsledek		
	Zadání souřadnic středu kružnice a rádiusu		
	0	Souřadnice středu kružnice: hodnota abscisy	
	1	Souřadnice středu kružnice: hodnota ordináty	
	2	Rádius	

Poznámka

Mějte prosím na paměti, že proměnné musí být před svým použitím definovány.

Příklad

U tří bodů se má zjistit, zda leží na kruhovém oblouku.



Programový kód	**Komentář**
`N10 DEF REAL PKT[3,2]=(20,50,50,40,65,20)`	`; Proměnná pro zadání bodů na kružnici`
`N20 DEF REAL ERG[3]`	`; Proměnná pro výsledek`
`N30 DEF BOOL STATUS`	`; Proměnná pro status`
`N40 STATUS=CALCDAT(PKT,3,ERG)`	`; Vyvolání zjištěných parametrů kružnice.`
`N50 IF STATUS == FALSE GOTOF ERROR`	`; Skok v případě chyby.`

15.7 Ukončení přípravy kontury (EXECUTE)

Funkce

Pomocí příkazu `EXECUTE` se příprava kontury ukončí a současně se přepne zpátky do normálního režimu zpracovávání programu.

Syntaxe

```
EXECUTE(<FEHLER>)
```

Význam

`EXECUTE`	Příkaz pro ukončení přípravy kontury
`<FEHLER>`	Proměnná pro zpětné chybové hlášení Typ: INT Hodnota proměnné ukazuje, zda bylo možné zpracovat konturu bezchybně. 0 Chyba 1 žádná chyba

Příklad

Programový kód

```
...
N30 CONTPRON(...)
N40 G1 X... Z...
...
N100 EXECUTE(...)
...
```

Programování externích cyklů 16

16.1 Technologické cykly

16.1.1 Úvod

Obsah

V této kapitole jsou dokumentovány technologické cykly od verze 2.6 pro sestavování externích NC programů.

Struktura

Tato dokumentace má následující strukturu:

- **Programování**

  Název cyklu a posloupnost vyvolávání předávaných parametrů

- **Parametry**

  Tabulka s vysvětleními jednotlivých parametrů

Popis parametrů

V tabulce se nacházejí názvy interně používaných parametrů a vysvětlivky týkající se významu a možného rozsahu hodnot. Kromě toho jsou vysvětlovány závislosti mezi jednotlivými parametry. Sloupec s odkazem na parametr na obrazovce má sloužit k tomu, aby při zpětném překladu externě generovaného volání cyklu bylo možné v řídícím systému znovu najít naprogramované hodnoty.

Parametry "jen pro pracovní plochu"

V tabulce jsou některé parametry označeny příznakem "jen pro pracovní plochu". Tyto parametry nemají pro fungování cyklu žádný význam. Jsou zapotřebí jen k tomu, aby volání cyklu bylo možné dokonale zpětně přeložit. Pokud nejsou naprogramovány, může být cyklus i přesto zpětně přeložen, pole jsou ale potom odpovídajícím způsobem barevně označena a musí být vyplněna na obrazovce.

Parametry "rezervováno"

Pro parametry, které jsou popsány jako "rezervované", musí být naprogramována hodnota 0 nebo musí zůstat prázdné, aby přiřazení následujících parametrů volání odpovídalo interním parametrům cyklu. Výjimka: U parametrů typu STRING hodnota "" nebo prázdný znak.

Kompatibilita

Technologické cykly od verze 2.6 představují další krok ve vývoji modulu cyklů pro systémy SINUMERIK 840D sl až GIV 1.5 (cykly do verze 7.5). NC programy s voláními cyklů těchto dřívějších verzí programového vybavení jsou i nadále spustitelné.

Většina cyklů byla rozšířena o nové předávané parametry nebo rozsah hodnot u již existujících parametrů byl rozšířen, aby bylo možné naprogramovat nové funkce (jako např. často používaný parametr _VARI pro způsob opracování.

Pojem kompatibilita ukazuje v této dokumentaci vstupní hodnoty, které nebyly naprogramovány dříve. Pokud jsou hodnoty předávány tímto způsobem, je cyklus zpracován po funkční stránce stejným způsobem jako do verze 7.5.

Opakování cyklu na polohovacím vzoru

Vrtací a frézovací cykly mohou být opakovány na polohovacím vzoru (modální volání). Před cyklem je potom na tomtéž řádku zapotřebí napsat příkaz `MCALL`, např. `MCALL CYCLE83(...)`.

Poznámka

Pokud jsou určité předávané parametry (např. `_VARI`, `_GMODE`, `_DMODE`, `_AMODE`) naprogramovány nepřímo jako parametry, při zpětném překladu se otevře vstupní obrazovka, tu však není možné uložit, protože pro některá pole s volitelnou hodnotou neexistuje žádné jednoznačné přiřazení.

16.1.2 Vrtání, navrtávání středicích důlků - CYCLE81

Programování

```
CYCLE81(REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL _DTB,
INT _GMODE, INT _DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	RP	RTP	Návratová rovina (abs)
2	Z0	RFP	Vztažný bod (abs)
3	SC	_SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
4	Z1/?	_DP	Vrtaná hloubka (abs)/průměr středicího důlku (abs), viz _GMODE
5	Z1	-DPR	Vrtaná hloubka (ink)
6	DT	_DTB	Doba prodlevy na konečné vrtané hloubce, viz _AMODE
7		_GMODE	Geometrický režim (vyhodnocování naprogramované geometrické hodnoty)
			JEDNOTKY: rezervováno
			DESÍTKY: Navrtávání středicího důlku je vztaženo na hloubku/průměr
			0 = kompatibilita, hloubka
			1 = průměr
8		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
9		_AMODE	Alternativní režim
			JEDNOTKY: Vrtaná hloubka Z1 (abs/ink)
			0 = kompatibilita, podle naprogramování DP/DPR
			1 = inkrementálně
			2 = absolutně
			DESÍTKY: Doba prodlevy na konečné vrtané hloubce DT v sekundách/otáčkách
			0 = kompatibilita, vyplývá ze znaménka DTB (> 0 sekundy nebo < 0 otáčky)
			1 = v sekundách
			2 = v otáčkách

16.1.3 Vrtání, čelní zahlubování - CYCLE82

Programování

```
CYCLE82(REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL DTB,
INT _GMODE, INT _DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	RP	RTP	Návratová rovina (abs)
2	Z0	RFP	Vztažný bod (abs)
3	SC	SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
4	Z1	DP	Vrtaná hloubka (abs), viz _AMODE
5	Z1	DPR	Vrtaná hloubka (ink), viz _AMODE
6	DT	DTB	Doba prodlevy na konečné vrtané hloubce, viz _AMODE
7		_GMODE	Geometrický režim (vyhodnocování naprogramované geometrické hodnoty)
			JEDNOTKY: rezervováno
			DESÍTKY: Vrtaná hloubka vztažena na špičku/stopku
			0 = kompatibilita, špička
			1 = stopka
8		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
9		_AMODE	Alternativní režim
			JEDNOTKY: Vrtaná hloubka Z1 (abs/ink)
			0 = kompatibilita, podle naprogramování DP/DPR
			1 = inkrementálně
			2 = absolutně
			DESÍTKY: Doba prodlevy DT na konečné vrtané hloubce v sekundách/otáčkách
			0 = kompatibilita, vyplývá ze znaménka DT (> 0 sekundy / < 0 otáčky)
			1 = v sekundách
			2 = v otáčkách

16.1.4 Vystružování - CYCLE85

Programování

```
CYCLE85(REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL DTB,
REAL FFR, REAL RFF, INT _GMODE, INT _DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	RP	RTP	Návratová rovina (abs)
2	Z0	RFP	Vztažný bod (abs)
3	SC	SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
4	Z1	DP	Vrtaná hloubka (abs), viz _AMODE
5	Z1	DPR	Vrtaná hloubka (ink), viz _AMODE
6	DT	DTB	Doba prodlevy na konečné vrtané hloubce, viz _AMODE
7	F	FFR	Posuv
8	FR	RFF	Posuv při zpětném pohybu
9		_GMODE	rezervováno
10		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
11		_AMODE	Alternativní režim (vrtání)
			JEDNOTKY: Vrtaná hloubka Z1 (abs/ink)
			0 = kompatibilita, podle naprogramování DP/DPR
			1 = inkrementálně
			2 = absolutně
			DESÍTKY: Doba prodlevy DT na konečné vrtané hloubce v sekundách/otáčkách
			0 = kompatibilita, vyplývá ze znaménka DT (> 0 sekundy nebo < 0 otáčky)
			1 = v sekundách
			2 = v otáčkách

16.1.5 Vrtání hlubokých děr - CYCLE83

Programování

```
CYCLE83(REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL FDEP,
REAL FDPR, REAL _DAM, REAL DTB, REAL DTS, REAL FRF, INT VARI, INT
_AXN, REAL _MDEP, REAL _VRT, REAL _DTD, REAL _DIS1, INT _GMODE, INT
_DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	RP	RTP	Návratová rovina (abs)
2	Z0	RFP	Vztažný bod (abs)
3	SC	SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
4	Z1	DP	Konečná vrtaná hloubka (abs), viz _AMODE
5	Z1	DPR	Konečná vrtaná hloubka (ink), viz _AMODE
6	D	FDEP	1. vrtaná hloubka (abs), viz _AMODE
7	D	FDPR	1. vrtaná hloubka (ink), viz _AMODE
8	DF	_DAM	Absolutní hodnota/procentuální změna pro každý další přísuv (degrese/procentuální změna), viz _AMODE
9	DTB	DTB	Doba prodlevy na vrtané hloubce, viz _AMODE
10	DTS	DTS	Doba prodlevy v počátečním bodě (jen při odstraňování třísek), viz _AMODE
11	FD1	FRF	Procentuální hodnota pro posuv při prvním přísuvu, viz _AMODE
12		VARI	Způsob opracování
			JEDNOTKY: Ulamování třísky/odstraňování třísek
			0 = ulamování třísky 1 = odstraňování třísek
13		_AXN	Osa nástroje: 0 = 3. Geometrická osa 1 = 1. Geometrická osa 2 = 2. Geometrická osa > 2 = 3. Geometrická osa
14	V1	_MDEP	minimální přísuv (jen když je degrese v procentech)
15	V2	_VRT	Vzdálenost zpětného pohybu po každém obrobení (jen při ulamování třísky) > 0 = proměnná vzdálenost zpětného pohybu 0 = standardní hodnota 1 mm
16	DT	_DTD	Doba prodlevy na konečné vrtané hloubce, viz _AMODE
17	V3	_DIS1	Chráněná vzdálenost (jen při odstraňování třísek), viz _AMODE
18		_GMODE	Geometrický režim (vyhodnocování naprogramované geometrické hodnoty)
			JEDNOTKY: rezervováno
			DESÍTKY: Vrtaná hloubka vztažena na špičku/stopku
			0 = špička 1 = stopka

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
19		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu 1 = G17 (aktivní jen v cyklu) 2 = G18 (aktivní jen v cyklu) 3 = G19 (aktivní jen v cyklu)
20		_AMODE	Alternativní režim
			JEDNOTKY: Vrtaná hloubka = konečná vrtaná hloubka Z1 (abs/ink)
			0 = kompatibilita, podle naprogramování DP/DPR 1 = inkrementálně 2 = absolutně
			DESÍTKY: Doba prodlevy na vrtané hloubce DTB v sekundách/otáčkách
			0 = kompatibilita, vyplývá ze znaménka DTB (> 0 sekundy nebo < 0 otáčky) 1 = v sekundách 2 = v otáčkách
			STOVKY: Doba prodlevy v počátečním bodě před DTS v sekundách/otáčkách
			0 = kompatibilita, vyplývá ze znaménka DTS (> 0 sekundy nebo < 0 otáčky) 1 = v sekundách 2 = v otáčkách
			TISÍCE: Doba prodlevy na konečné vrtané hloubce DT v sekundách/otáčkách
			0 = kompatibilita, vyplývá ze znaménka _DTD (> 0 sekundy nebo < 0 otáčky) 1 = v sekundách 2 = v otáčkách
			DESÍTKY TISÍC: 1. Vrtaná hloubka D (abs/ink)
			0 = kompatibilita, podle naprogramování FDEP/FDPR 1 = inkrementálně 2 = absolutně
			STOVKY TISÍC: Absolutní hodnota/procentuální hodnota DAM pro každý další přísuv (degrese)
			0 = kompatibilita, vyplývá ze znaménka DAM (> 0 absolutní hodnota nebo < 0 faktor 0.001 až 1.0) 1 = absolutní hodnota 2 = procentuální změna (0,001 až 100%)
			JEDNOTKY MILIONŮ: Chráněná vzdálenost V3 automaticky/manuálně
			0 = kompatibilita, vyplývá ze znaménka _DIS1 (= 0 automaticky nebo > 0 manuálně) 1 = automaticky (vypočítává se v cyklu) 2 = manuálně (naprogramovaná hodnota)
			DESÍTKY MILIONŮ: Faktor posuvu pro první přísuv FRF jako faktor/procentuální změna
			0 = kompatibilita, jako faktor (0.001 až 1.0, FRF = 0 znamená 100%) 1 = procentuální změna (0,001 až 999,999 %)

16.1.6 Vyvrtávání - CYCLE86

Programování

```
CYCLE86(REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL DTB,
INT SDIR, REAL RPA, REAL RPO, REAL RPAP, REAL POSS, INT _GMODE, INT
_DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	RP	RTP	Návratová rovina (abs)
2	Z0	RFP	Vztažný bod (abs)
3	SC	SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
4	Z1	DP	Vrtaná hloubka (abs), viz _AMODE
5	Z1	DPR	Vrtaná hloubka (ink), viz _AMODE
6	DT	DTB	Doba prodlevy na konečné vrtané hloubce, viz _AMODE
7	DIR	SDIR	Otáčení vřetena 3 = M3 4 = M4
8	DX	RPA	Vzdálenost pro pozvednutí ve směru X
9	DY	RPO	Vzdálenost pro pozvednutí ve směru Y
10	DZ	RPAP	Vzdálenost pro pozvednutí ve směru Z
11	SPOS	POSS	Poloha vřetena pro pozvednutí (pro orientované zastavení vřetena, ve stupních)
12		_GMODE	Geometrický režim
			JEDNOTKY: Způsob pozvednutí nástroje
			0 = pozvednutí, kompatibilita 1 = nepozvednout
13		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu 1 = G17 (aktivní jen v cyklu) 2 = G18 (aktivní jen v cyklu) 3 = G19 (aktivní jen v cyklu)
14		_AMODE	Alternativní režim
			JEDNOTKY: Vrtaná hloubka Z1 (abs/ink)
			0 = kompatibilita, podle naprogramování DP/DPR 1 = inkrementálně 2 = absolutně
			DESÍTKY: Doba prodlevy na konečné vrtané hloubce DT v sekundách/otáčkách
			0 = kompatibilita, vyplývá ze znaménka DT (> 0 sekundy nebo < 0 otáčky) 1 = v sekundách 2 = v otáčkách

16.1.7 Vrtání závitů bez vyrovnávací hlavičky - CYCLE84

Programování

```
CYCLE84(REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL DTB,
INT SDAC, REAL MPIT, REAL PIT, REAL POSS, REAL SST, REAL SST1, INT
_AXN, INT _PITA, INT _TECHNO, INT _VARI, REAL _DAM, REAL _VRT,
STRING[15] _PITM, STRING[5] _PTAB, STRING[20] _PTABA, INT _GMODE, INT
_DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení	
1	RP	RTP	Návratová rovina (abs)	
2	Z0	RFP	Vztažný bod (abs)	
3	SC	SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)	
4	Z1	DP	Vrtaná hloubka = konečná vrtaná hloubka (abs), viz _AMODE	
5	Z1	DPR	Vrtaná hloubka = konečná vrtaná hloubka (ink), viz _AMODE	
6	DT	DTB	Doba prodlevy na vrtané hloubce v sekundách	
7	SDE	SDAC	Směr otáčení po skončení cyklu	
8		MPIT	Velikost závitu, pouze pro "ISO metrický" (stoupání se vypočítává interně v průběhu zpracovávání programu)	
9	P	PIT	Stoupání závitu jako hodnota, rozměrové jednotky viz _PITA	
10	αS[1)]	POSS	Poloha pro orientované zastavení vřetena	
11	S	SST	Otáčky vřetena pro vrtání závitu	
12	SR	SST1	Otáčky vřetena pro zpětný pohyb	
13		_AXN	Osa vrtání: 0 = 3. Geometrická osa 1 = 1. Geometrická osa 2 = 2. Geometrická osa ≥ 3 = 3. Geometrická osa	
14		_PITA	Rozměrové jednotky pro stoupání závitu 0 = stoupání v mm 1 = stoupání v mm 2 = stoupání v TPI 3 = stoupání v palcích 4 = MODUL	(vyhodnocování PIT a MPIT) - vyhodnocování PIT/MPIT - vyhodnocování PIT - vyhodnocování PIT (počet chodů závitu na palec) - vyhodnocování PIT - vyhodnocování PIT

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
15		_TECHNO	Technologie[1)]
			JEDNOTKY: Režim přesného najetí
			0 = je aktivní chování přesného najetí, jaké bylo před voláním cyklu
			1 = přesné najetí G601
			2 = přesné najetí G602
			3 = přesné najetí G603
			DESÍTKY: Dopředná regulace
			0 = s/bez dopředné regulace, jak bylo aktivní před voláním cyklu
			1 = s dopřednou regulací FFWON
			2 = bez dopředné regulace FFWOF
			STOVKY: Zrychlení
			0 = SOFT/BRISK/DRIVE, jak bylo aktivní před voláním cyklu
			1 = s omezením ryvu, SOFT
			2 = bez omezení ryvu, BRISK
			3 = snížené zrychlení, DRIVE
			TISÍCE: Režim vřetena při MCALL
			0 = s příkazem MCALL se opět aktivuje režim vřetena
			1 = s příkazem MCALL zůstat v režimu polohové regulace
16		_VARI	Způsob opracování:
			JEDNOTKY:
			0 = 1 řez
			1 = ulamování třísky (vrtání závitů v hlubokých dírách)
			2 = odstraňování třísek (vrtání závitů v hlubokých dírách)
			TISÍCE: Režim ISO/SIEMENS pro vstupní obrazovku není relevantní
			1 = vyvolávání na základě kompatibility s ISO
			0 = vyvolávání na základě kontextu firmy SIEMENS
17	D	_DAM	Maximální přísuv do hloubky (jen při odstraňování třísky/ulamování třísky)
18	V2	_VRT	Vzdálenost zpětného pohybu po každém obrobení (jen při ulamování třísky), viz _AMODE
19		_PITM	Řetězec jako ukazatel pro zadávání stoupání závitu[2)]
20		_PTAB	Řetězec pro tabulku závitů ("", "ISO", "BSW", "BSP", "UNC")[2)]
21		_PTABA	Řetězec pro volbu v tabulce závitů (např. "M 10", "M 12", ...)[2)]
22		_GMODE	Geometrický režim (vyhodnocování naprogramované geometrické hodnoty)
			JEDNOTKY: rezervováno
			DESÍTKY: rezervováno

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
23		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
			DESÍTKY: rezervováno
			STOVKY: rezervováno
			TISÍCE: Režim kompatibility (jen pro vstupní obrazovku pro zpětný překlad), pokud je MD 52216 Bit0 = 1[1)]
			0 = technologické parametry se vypisují (kompatibilita): parametr TECHNO je v platnosti
			1 = technologické parametry se nevypisují: platí technologie, jaká byla naprogramována před voláním cyklu
24		_AMODE	Alternativní režim
			JEDNOTKY: Vrtaná hloubka = konečná vrtaná hloubka Z1 (abs/ink)
			0 = kompatibilita, podle naprogramování DP/DPR
			1 = inkrementálně
			2 = absolutně
			DESÍTKY: rezervováno
			STOVKY: rezervováno
			TISÍCE: směr otáčení závitu pravý/levý
			0 = kompatibilita, vyplývá ze znaménka parametru PIT/MPIT
			1 = pravý
			2 = levý
			DESÍTKY TISÍC: rezervováno
			STOVKY TISÍC: rezervováno
			JEDNOTKY MILIONŮ: Vzdálenost zpětného pohybu po každém obrobení V2 manuálně/ automaticky
			0 = kompatibilita, podle naprogramování parametru _VRT (> 0 proměnná hodnota nebo ≤ 0 standardní hodnota 1 mm/0.0394 palce)
			1 = automaticky (standardní hodnota 1mm/0.0394 palce)
			2 = manuálně (jak bylo naprogramováno pro V2)

[1)] V závislosti na nastavovaném parametru SD52216 $MCS_FUNCTION_MASK_DRILL může být zobrazování polí technologie blokováno.

[2)] Parametry 19, 20 a 21 se používají pouze při volbě závitu v tabulce závitů ve vstupní obrazovce. Přístup k tabulkám závitů prostřednictvím definice cyklů v době, kdy jsou cykly zpracovávány, není možný.

16.1.8 Vrtání závitů s vyrovnávací hlavičkou - CYCLE840

Programování

```
CYCLE840(REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL DTB,
INT SDR, INT SDAC, INT ENC, REAL MPIT, REAL PIT, INT _AXN, INT _PITA,
INT _TECHNO, STRING[15] _PITM, STRING[5] _PTAB, STRING[20] _PTABA,
INT _GMODE, INT _DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení	
1	RP	RTP	Návratová rovina (abs)	
2	Z0	RFP	Vztažný bod (abs)	
3	SC	SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)	
4	Z1	DP	Vrtaná hloubka (abs), viz _AMODE	
5	Z1	DPR	Vrtaná hloubka (ink), viz _AMODE	
6	DT	DTB	Doba prodlevy na vrtané hloubce/na bezpečnostní vzdálenosti po zpětném pohybu v sekundách, viz ENC	
7		SDR	Směr otáčení pro zpětný pohyb	
8	SDE	SDAC	Směr otáčení po skončení cyklu	
9		ENC	Vrtání závitu s vřetenovým snímačem (G33)/vrtání závitu bez vřetenového snímače (G63)	
			0 = s vřetenovým snímačem	- stoupání z MPIT/PIT - bez DT
			20 = s vřetenovým snímačem	- stoupání z MPIT/PIT - s DT po zpětném pohybu na bezpečnostní vzdálenost
			11 = bez vřetenového snímače	- stoupání z MPIT/PIT - s DT na vrtané hloubce
			1 = bez vřetenového snímače	- Stoupání z naprogramovaného posuvu - s DT na vrtané hloubce (posuv = otáčky stoupání)
10		MPIT	Velikost závitu, pouze pro "ISO metrický" (stoupání se vypočítává interně v průběhu zpracovávání programu) Rozsah hodnot: 3 až 48 (pro M3 až M48), alternativa k PIT	
11		PIT	Stoupání závitu jako hodnota, rozměrové jednotky viz _PITA Rozsah hodnot: > 0, alternativa k MPIT	
12		_AXN	Osa vrtání: 0 = 3. Geometrická osa 1 = 1. Geometrická osa 2 = 2. Geometrická osa ≥ 3 = 3. Geometrická osa	
13		_PITA	Rozměrové jednotky pro stoupání závitu (vyhodnocování PIT a MPIT)	
			0 = stoupání v mm	- vyhodnocování PIT/MPIT
			1 = stoupání v mm	- vyhodnocování PIT
			2 = stoupání v TPI	- vyhodnocování PIT (počet chodů závitu na palec)
			3 = stoupání v palcích	- vyhodnocování PIT
			4 = MODUL	- vyhodnocování PIT

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
14		_TECHNO	Technologie[1)]
			JEDNOTKY: Režim přesného najetí
			0 = je aktivní přesné najetí, jaké bylo před voláním cyklu
			1 = přesné najetí G601
			2 = přesné najetí G602
			3 = přesné najetí G603
			DESÍTKY: Dopředná regulace
			0 = s/bez dopředné regulace, jak bylo aktivní před voláním cyklu
			1 = s dopřednou regulací FFWON
			2 = bez dopředné regulace FFWOF
15		_PITM	Řetězec jako ukazatel pro zadávání stoupání závitu[2)]
16		_PTAB	Řetězec pro tabulku závitů ("", "ISO", "BSW", "BSP", "UNC")[2)]
17		_PTABA	Řetězec pro volbu v tabulce závitů (např. "M 10", "M 12", ...)[2)]
18		_GMODE	rezervováno
19		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
			DESÍTKY: rezervováno
			STOVKY: rezervováno
			TISÍCE: Režim kompatibility (jen pro vstupní obrazovku pro zpětný překlad), pokud je MD 52216 Bit0 = 1[1)]
			0 = technologické parametry se vypisují (kompatibilita): parametr TECHNO je v platnosti
			1 = technologické parametry se nevypisují: platí technologie, jaká byla naprogramována před voláním cyklu
20		_AMODE	Alternativní režim
			JEDNOTKY: Vrtaná hloubka Z1 (abs/ink)
			0 = kompatibilita, podle naprogramování DP/DPR
			1 = inkrementálně
			2 = absolutně

[1)] V závislosti na nastavovaném parametru SD52216 MCS_FUNCTION_MASK_DRILL může být zobrazování polí technologie blokováno.

[2)] Parametry 15, 16 a 17 se používají pouze při volbě závitu v tabulce závitů ve vstupní obrazovce. Přístup k tabulkám závitů prostřednictvím definice cyklů v době, kdy jsou cykly zpracovávány, není možný!

16.1.9 Frézování vrtaných závitů - CYCLE78

Programování

```
CYCLE78(REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _ADPR, REAL
_FDPR, REAL _LDPR, REAL _DIAM, REAL _PIT, INT _PITA, REAL _DAM, REAL
_MDEP, INT _VARI, INT _CDIR, REAL _GE, REAL _FFD, REAL _FRDP, REAL
_FFR, REAL _FFP2, INT _FFA, STRING[15] _PITM, STRING[20] _PTAB,
STRING[20] _PTABA, INT _GMODE, INT _DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	RP	_RTP	Návratová rovina (abs)
2	Z0	_RFP	Vztažný bod osy nástroje (abs)
3	SC	_SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
4	Z1	_DP	Konečná vrtaná hloubka (abs/ink), viz _AMODE
5		_ADPR	Navrtávaná hloubka se sníženou hodnotou posuvu při vrtání (ink), v platnosti podle místa desítek tisíc parametru VARI
6	D	_FDPR	Maximální přísuv do hloubky (ink) D ≥ Z1 ⇒ jeden přísuv až na konečnou vrtanou hloubku D < Z1 ⇒ cyklus pro vrtání hlubokých děr s větším počtem přísuvů a s odstraňováním třísek
7	ZR	_LDPR	Zbytková vrtaná hloubka při provrtávání (ink), s posuvem FR
8	?	_DIAM	Jmenovitý průměr závitu
9	P	_PIT	Stoupání závitu jako číselná hodnota
10		_PITA	Vyhodnocování stoupání závitu P 1 = stoupání v mm/ot 2 = stoupání v chodech/palec 3 = stoupání v palcích/ot 4 = stoupání jako MODUL
11	DF	_DAM	Absolutní hodnota/procentuální změna pro každý další přísuv (degrese), viz _AMODE
12	V1	_MDEP	minimální přísuv (ink), jen když je používána degrese
13		_VARI	Způsob opracování
			JEDNOTKY: rezervováno
			DESÍTKY:
			0 = žádné odstraňování třísek před frézováním závitu (uplatňuje se pouze na konečné vrtané hloubce) 1 = odstraňování třísek před frézováním závitu (uplatňuje se pouze na konečné vrtané hloubce)
			STOVKY:
			0 = pravý závit 1 = levý závit
			TISÍCE:
			0 = žádná zbytková vrtaná hloubka s posuvem při vrtání FR 1 = zbytková vrtaná hloubka s posuvem při vrtání FR
			DESÍTKY TISÍC:
			0 = žádné navrtávání se sníženou hodnotou posuvu 1 = navrtávání se sníženou hodnotou posuvu Posuv při navrtávání = 0,3 F1, pokud je F1 < 0.15 mm/ot Posuv při navrtávání = 0,1 mm/ot, pokud je F1 ≥ 0.15 mm/ot

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
14		_CDIR	Směr frézování 0 = sousledné frézování 1 = nesousledné frézování 4 = nesousledné + sousledné (kombinace obrábění nahrubo + načisto)
15	Z2	_GE	Vzdálenost zpětného pohybu před frézováním závitu (ink)
16	F1	_FFD	Posuv při vrtání (mm/min, příp. palce/min nebo mm/ot)
17	FR	_FRDP	Posuv při vrtání pro zbytkovou vrtanou hloubku (mm/min nebo mm/ot)
18	F2	-FFR	Posuv pro frézování závitu (mm/min nebo mm/zub)
19	FS	_FFP2	Posuv při obrábění načisto pro CDIR = 4 (mm/min nebo mm/zub)
20		_FFA	Vyhodnocování posuvu
			JEDNOTKY: posuv při vrtání F1
			DESÍTKY: posuv při vrtání pro zbytkovou vrtanou hloubku FR
			STOVKY: Posuv pro frézování závitu F2
			TISÍCE: Posuv při obrábění načisto FS
21		_PITM	Řetězec jako ukazatel pro zadávání stoupání závitu (jen pro plochu)[1)]
22		_PTAB	Řetězec pro tabulku závitů ("", "ISO", "BSW", "BSP", "UNC") (jen pro pracovní plochu)[1)]
23		_PTABA	Řetězec pro volbu v tabulce závitů (např. "M 10", "M 12", ...) (jen pro pracovní plochu) [1)]
24		_GMODE	Geometrický režim, rezervováno
25		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu 1 = G17 (aktivní jen v cyklu) 2 = G18 (aktivní jen v cyklu) 3 = G19 (aktivní jen v cyklu)
26		_AMODE	Alternativní režim
			JEDNOTKY: Vrtaná hloubka = konečná vrtaná hloubka Z1 (abs/ink)
			0 = absolutně 1 = inkrementálně
			DESÍTKY: Absolutní hodnota/procentuální hodnota DF pro každý další přísuv (degrese)
			0 = absolutní hodnota 1 = procentuální změna (0,001 až 100%)

Poznámka

1) Parametry 21, 22 a 23 se používají pouze při volbě závitu v tabulce závitů ve vstupní obrazovce. Přístup k tabulkám závitů prostřednictvím definice cyklů v době, kdy jsou cykly zpracovávány, není možný.

16.1.10 Libovolné pozice - CYCLE802

Programování

```
CYCLE802(INT _XA, INT _YA, REAL _X0, REAL _Y0, REAL _X1, REAL _Y1,
REAL _X2, REAL _Y2, REAL _X3, REAL _Y3, REAL _X4, REAL _Y4, REAL _X5,
REAL _Y5, REAL _X6, REAL _Y6, REAL _X7, REAL _Y7, REAL _X8, REAL _Y8,
INT _VARI, INT _UMODE, INT _DMODE)
```

Parametr

Č.	Obrazovka parametrů	Parametr interní	Vysvětlení
1		_XA	Alternativa pro všechny pozice na ose X (9-místné desítkové číslo) Číslo místa: 876543210 (místo odpovídá poloze vrtané díry Xn) Hodnota místa: 1 = absolutní (1. naprogramovaná pozice vždy absolutně!) 2 = inkrementálně
2		_YA	Alternativa pro všechny pozice na ose Y (9-místné desítkové číslo) Číslo místa: 876543210 (místo odpovídá poloze vrtané díry Yn) Hodnota místa: 1 = Údaj polohy (abs) 2 = Údaj polohy (ink)
3	X0	_X0	1. poloha X
4	Y0	_Y0	1. poloha Y
5	X1	_X1	2. poloha X
6	Y1	_Y1	2. poloha Y
7	X2	_X2	3. poloha X
8	Y2	_Y2	3. poloha Y
9	X3	_X3	4. poloha X
10	Y3	_Y3	4. poloha Y
11	X4	_X4	5. poloha X
12	Y4	_Y4	5. poloha Y
13	X5	_X5	6. poloha X
14	Y5	_Y5	6. poloha Y
15	X6	_X6	7. poloha X
16	Y6	_Y6	7. poloha Y
17	X7	_X7	8. poloha X
18	Y7	_Y7	8. poloha Y
19	X8	_X8	9. poloha X
20	Y8	_Y8	9. poloha Y
21		_VARI	rezervováno

Č.	Obrazovka parametrů	Parametr interní	Vysvětlení
22		_UMODE	rezervováno
23		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)

Poznámka

Nepotřebné pozice dané parametry X1/Y1 až X8/Y8 mohou být vypuštěny.

Alternativní hodnoty pro parametry _XA a _YA je však zapotřebí úplně zadat pro všech 9 pozic.

16.1.11 Řada děr - HOLES1

Programování

```
HOLES1(REAL SPCA, REAL SPCO, REAL STA1, REAL FDIS, REAL DBH, INT NUM,
INT _VARI, INT _UMODE, STRING[200] _HIDE, INT _NSP, INT _DMODE)
```

Parametr

Č.	Obrazovka parametrů	Parametr interní	Vysvětlení
1	X0	SPCA	Vztažný bod na 1. ose pro řadu děr (abs)
2	Y0	SPCO	Vztažný bod na 2. ose pro řadu děr (abs)
3	α0	STA1	Základní úhel otočení (úhel vzhledem k 1. ose)
4	L0	FDIS	Vzdálenost první díry od vztažného bodu
5	L	DBH	Vzdálenost mezi dírami
6	N	NUM	Počet děr
7		_VARI	rezervováno
8		_UMODE	rezervováno
9		_HIDE	Přeskakované pozice • max. 198 znaků • Údaj pořadového čísla pozice, např. "1,3" (pozice 1 a 3 se nebudou uskutečňovat)
10		_NSP	rezervováno
11		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)

16.1.12 Mřížka nebo obdélník - CYCLE801

Programování

```
CYCLE801(REAL _SPCA, REAL _SPCO, REAL _STA, REAL _DIS1, REAL _DIS2,
INT _NUM1, INT _NUM2, INT _VARI, INT _UMODE, REAL _ANG1, REAL _ANG2,
STRING[200] _HIDE, INT _NSP, INT _DMODE)
```

Parametr

Č.	Obrazovka parametrů	Parametr interní	Vysvětlení
1	X0	_SPCA	Vztažný bod pro polohovací vzor (mřížka/obdélník) na 1. ose (abs)
2	Y0	_SPCO	Vztažný bod pro polohovací vzor (mřížka/obdélník) na 2. ose (abs)
3	α0	_STA	Základní úhel otočení (úhel vzhledem k 1. ose) < 0 = Opisování ve směru hodinových ručiček > 0 = Opisování proti směru hodinových ručiček
4	L1	_DIS1	Vzdálenost sloupců (vzdálenost mezi pozicemi na 1. ose, zadává se bez znaménka)
5	L2	_DIS2	Vzdálenost řádků (vzdálenost mezi pozicemi na 2. ose, zadává se bez znaménka)
6	N1	_NUM1	Počet sloupců
7	N2	_NUM2	Počet řádků
8		_VARI	Způsob opracování
			JEDNOTKY: Polohovací vzor
			0 = mřížka 1 = obdélník
			DESÍTKY: rezervováno
			STOVKY: rezervováno
9		_UMODE	rezervováno
10	αX	_ANG1	Úhel skosení vzhledem k 1. ose (zešikmení řádků vzhledem k 1. ose) < 0 = měření ve směru hodinových ručiček (0 až -90 stupňů) > 0 = měření proti směru hodinových ručiček (0 až 90 stupňů)
11	αY	_ANG2	Úhel skosení vzhledem k 2. ose (zešikmení sloupců vzhledem ke 2. ose) < 0 = měření ve směru hodinových ručiček (0 až -90 stupňů) > 0 = měření proti směru hodinových ručiček (0 až 90 stupňů)
12		_HIDE	Přeskakované pozice • max. 198 znaků • Údaj pořadového čísla pozice, např. "1,3" (pozice 1 a 3 se nebudou uskutečňovat)
13		_NSP	rezervováno
14		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu 1 = G17 (aktivní jen v cyklu) 2 = G18 (aktivní jen v cyklu) 3 = G19 (aktivní jen v cyklu)

16.1.13 Díry uspořádané na kruhovém oblouku - HOLES2

Programování

```
HOLES2(REAL CPA, REAL CPO, REAL RAD, REAL STA1, REAL INDA, INT NUM,
INT _VARI, INT _UMODE, STRING[200] _HIDE, INT _NSP, INT _DMODE)
```

Parametr

Č.	Obrazovka parametrů	Parametr interní	Vysvětlení
1	X0	CPA	Střed kruhu na 1. ose pro díry na kruhovém oblouku (abs)
2	Y0	CPO	Střed kruhu na 2. ose pro díry na kruhovém oblouku (abs)
3	R	RAD	Rádius kruhového oblouku
4	α0	STA1	Počáteční úhel
5	α1	INDA	Úhlový krok, kterým jsou drážky od sebe vzdáleny (jen u kruhového oblouku) < 0 = ve směru hodinových ručiček > 0 = proti směru hodinových ručiček
6	N	NUM	Počet pozic
7		_VARI	Způsob opracování
			JEDNOTKY: rezervováno
			DESÍTKY: Způsob najíždění na pozici
			0 = najíždění na pozici po lineární dráze 1 = najíždění na pozici po kruhové dráze
			STOVKY: : rezervováno
			TISÍCE: Kruhový vzor
			0 = režim kompatibility, pokud je INDA = 0, pak celá kružnice, INDA <> 0, potom kruhový oblouk 1 = ceká kružnice 2 = kruhový oblouk
8		_UMODE	rezervováno
9		_HIDE	Přeskakované pozice • max. 198 znaků • Údaj pořadového čísla pozice, např. "1,3" (pozice 1 a 3 se nebudou uskutečňovat)
10		_NSP	rezervováno
11		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu 1 = G17 (aktivní jen v cyklu) 2 = G18 (aktivní jen v cyklu) 3 = G19 (aktivní jen v cyklu)

16.1.14 Rovinné frézování - CYCLE61

Programování

```
CYCLE61(REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _PA, REAL
_PO, REAL _LENG, REAL _WID, REAL _MID, REAL _MIDA, REAL _FALD, REAL
_FFP1, INT _VARI, INT _LIM, INT _DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	RP	_RTP	Návratová rovina (abs)
2	Z0	_RFP	Vztažný bod osy nástroje, výška surového obrobku (abs)
3	SC	_SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
4	Z1	_DP	Výška hotového obrobku (abs/ink), viz _AMODE
5	X0	_PA	Rohový bod 1 na 1. ose (abs)
6	Y0	_PO	Rohový bod 1 na 2. ose (abs)
7	X1	_LENG	Rohový bod 2 na 1. ose (abs/ink), viz _AMODE
8	Y1	_WID	Rohový bod 2 na 2. ose (abs/ink), viz _AMODE
9	DZ	_MID	Maximální přísuv do hloubky
10	DXY	_MIDA	Maximální přísuv v rovině (jednotky viz _AMODE)
11	UZ	_FALD	Přídavek rozměru pro opracování načisto na dně
12	F	_FFP1	Pracovní posuv
13		_VARI	Způsob opracování
			JEDNOTKY: Opracování
			1 = Obrábění nahrubo
			2 = Obrábění načisto
			DESÍTKY: Směr obrábění
			1 = rovnoběžně s 1. osou, jeden směr
			2 = rovnoběžně s 2. osou, jeden směr
			3 = rovnoběžně s 1. osou, střídání směrů
			4 = rovnoběžně s 2. osou, střídání směrů

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
14		_LIM	Ohraničení
			JEDNOTKY: Ohraničení 1. osy ve směru mínus
			0 = ne
			1 = ano
			DESÍTKY: Ohraničení 1. osy ve směru plus
			0 = ne
			1 = ano
			STOVKY: Ohraničení 2. osy ve směru mínus
			0 = ne
			1 = ano
			TISÍCE: Ohraničení 2. osy ve směru plus
			0 = ne
			1 = ano
15		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
16		_AMODE	Alternativní režim
			JEDNOTKY: Konečná hloubka (_DP)
			0 = absolutně
			1 = inkrementálně
			DESÍTKY: Jednotky pro přísuv v rovině (_MIDA)
			0 = mm
			1 = procentuální hodnota z průměru nástroje
			STOVKY: rezervováno
			TISÍCE: Délka plochy
			0 = inkrementálně
			1 = absolutně
			DESÍTKY TISÍC: Šířka plochy
			0 = inkrementálně
			1 = absolutně

16.1.15 Frézování pravoúhlé kapsy - POCKET3

Programování

```
POCKET3(REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _LENG, REAL
_WID, REAL _CRAD, REAL _PA, REAL _PO, REAL _STA, REAL _MID, REAL
_FAL, REAL _FALD, REAL _FFP1, REAL _FFD, INT _CDIR, INT _VARI, REAL
_MIDA, REAL _AP1, REAL _AP2, REAL _AD, REAL _RAD1, REAL _DP1, INT
_UMODE, REAL _FS, REAL _ZFS, INT _GMODE, INT _DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	RP	_RTP	Návratová rovina (abs)
2	Z0	_RFP	Vztažný bod osy nástroje (abs)
3	SC	_SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
4	Z1	_DP	Hloubka kapsy (abs/ink), viz _AMODE
5	L	_LENG	Délka kapsy (ink, je potřeba zadávat se znaménkem)
6	W	_WID	Šířka kapsy (ink, je potřeba zadávat se znaménkem)
7	RP	_CRAD	Rohový rádius kapsy
8	X0	_PA	Vztažný bod, 1. osa (abs)
9	YO	_PO	Vztažný bod, 2. osa (abs)
10	α0	_STA	Úhel otočení, úhel mezi podélnou osou (L) a 1. osou
11	DZ	_MID	Maximální přísuv do hloubky
12	UXY	_FAL	Přídavek rozměru pro opracování načisto v rovině
13	UZ	_FALD	Přídavek rozměru pro opracování načisto na dně
14	F	_FFP1	Posuv v rovině
15	FZ	_FFD	Rychlost přísuvu do hloubky
16		_CDIR	Směr frézování: 0 = sousledné frézování 1 = nesousledné frézování
17		_VARI	Způsob opracování
			JEDNOTKY:
			1 = Obrábění nahrubo 2 = Obrábění načisto 4 = Obrábění okraje načisto 5 = Srážení hran
			DESÍTKY:
			0 = předvrtáno, přísuv s G0 1 = kolmo, přísuv s G1 2 = po spirále 3 = oscilačním pohybem v podélné ose kapsy
			STOVKY: rezervováno

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
18	DXY	_MIDA	Maximální přísuv v rovině, jednotky viz _AMODE
19	L1	_AP1	Délka částečně obrobené kapsy (ink)
20	W1	_AP2	Šířka částečně obrobené kapsy (ink)
21	AZ	_AD	Hloubka částečně obrobené kapsy (ink)
22	ER	_RAD1	Rádius spirální dráhy při zajíždění po šroubovici
	EW		Maximální úhel zajíždění pro oscilační pohyb
23	EP	_DP1	Stoupání spirály při zajíždění po spirální dráze
24		_UMODE	rezervováno
25	FS	_FS	Šířka fasety (ink)
26	ZFS	_ZFS	Hloubka zajíždění (špička nástroje) při srážení hran (abs/ink), viz _AMODE
27		_GMODE	Geometrický režim
			JEDNOTKY: rezervováno
			DESÍTKY: rezervováno
			STOVKY: Volba opracování/pouze výpočet počátečního bodu
			0 = režim kompatibility
			1 = normální opracování
			TISÍCE: Zadávání rozměrů pomocí středu/rohu
			0 = režim kompatibility
			1 = kótování pomocí středu
			2 = kótování pomocí rohového bodu, poloha kapsy +LENG/+WID
			3 = kótování pomocí rohového bodu, poloha kapsy -LENG/+WID
			4 = kótování pomocí rohového bodu, poloha kapsy +LENG/-WID
			5 = kótování pomocí rohového bodu, poloha kapsy -LENG/-WID
			DESÍTKY TISÍC: Kompletní opracování/dopracování
			0 = režim kompatibility (_AP1, _AP2 a _AD jsou zpracovány jako dříve)
			1 = kompletní opracování
			2 = dopracování
28		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
			DESÍTKY: Druh posuvu: G-skupina (G94/G95) pro posuv v rovině a posuv do hloubky
			0 = režim kompatibility
			1 = G-kód stejný jako před voláním cyklu. G94/G95 pro posuv v rovině a posuv do hloubky zůstávají stejné.

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
29		_AMODE	Alternativní režim
			JEDNOTKY: Hloubka kapsy (Z1)
			0 = absolutně (režim kompatibility)
			1 = inkrementálně
			DESÍTKY: Jednotky pro přísuv v rovině (DXY)
			0 = mm
			1 = procentuální hodnota z průměru nástroje
			STOVKY: Hloubka zajíždění při srážení hran (ZFS)
			0 = absolutně
			1 = inkrementálně

16.1.16 Frézování kruhové kapsy - POCKET4

Programování

```
POCKET4(REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _CDIAM, REAL
_PA, REAL _PO, REAL _MID, REAL _FAL, REAL _FALD, REAL _FFP1, REAL
_FFD, INT _CDIR, INT _VARI, REAL _MIDA, REAL _AP1, REAL _AD, REAL
_RAD1, REAL _DP1, INT _UMODE, REAL _FS, REAL _ZFS, INT _GMODE, INT
_DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	RP	_RTP	Návratová rovina (abs)
2	Z0	_RFP	Vztažný bod osy nástroje (abs)
3	SC	_SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
4	Z1	_DP	Hloubka kapsy (abs/ink), viz _AMODE
5	∅	_DIAM	Průměr kapsy nebo rádius kapsy, viz _DMODE
6	X0	_PA	Vztažný bod, 1. osa (abs)
7	Y0	_PO	Vztažný bod, 2. osa (abs)
8	DZ	_MID	maximální přísuv do hloubly, viz _VARI = po rovinách maximální stoupání spirály, viz _VARI = po šroubovici
9	UXY	_FAL	Přídavek rozměru pro opracování načisto v rovině
10	UZ	_FALD	Přídavek rozměru pro opracování načisto na dně
11	F	_FFP1	Posuv pro opracovávání plochy
12	FZ	_FFD	Rychlost přísuvu do hloubky
13		_CDIR	Směr frézování 0 = sousledné frézování 1 = nesousledné frézování
14		_VARI	Způsob opracování
			JEDNOTKY:
			1 = Obrábění nahrubo 2 = Obrábění načisto 4 = Obrábění okraje načisto 5 = Srážení hran
			DESÍTKY: Způsob přísuvu (obrábění nahrubo a obrábění načisto)
			0 = předvrtáno, přísuv s G0 (kapsa je předem opracována) 1 = kolmo, přísuv s G1 2 = po spirále
			STOVKY: rezervováno
			TISÍCE:
			0 = po rovinách 1 = po spirále

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
15	DXY	_MIDA	Maximální přísuv v rovině, viz _AMODE, 0 = 0,8 průměr nástroje
16	∅	_AP1	Průměr/rádius částečně obrobené kapsy (ink)
17	AZ	_AD	Hloubka částečně obrobené kapsy (ink)
18	ER	_RAD1	Rádius spirální dráhy při zajíždění po šroubovici
19	EP	_DP1	Stoupání spirály při zajíždění po spirální dráze
20		_UMODE	rezervováno
21	FS	_FS	Šířka fasety (ink)
22	ZFS	_ZFS	Hloubka zajíždění (špička nástroje) při srážení hran (abs/ink), viz _AMODE
23		_GMODE	Geometrický režim
			JEDNOTKY: rezervováno
			DESÍTKY: rezervováno
			STOVKY: Opracování/výpočet počátečního bodu
			0 = režim kompatibility 1 = normální opracování
			TISÍCE: rezervováno
			DESÍTKY TISÍC: Kompletní opracování/dopracování
			0 = režim kompatibility (_AP1 a _AD jsou zpracovány jako dříve) 1 = kompletní opracování 2 = dopracování
24		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu 1 = G17 (aktivní jen v cyklu) 2 = G18 (aktivní jen v cyklu) 3 = G19 (aktivní jen v cyklu)
			DESÍTKY: Druh posuvu: G-skupina (G94/G95) pro posuv v rovině a posuv do hloubky
			0 = režim kompatibility 1 = G-kód stejný jako před voláním cyklu. G94/G95 pro posuv v rovině a posuv do hloubky zůstávají stejné.
			STOVKY:
			0 = režim kompatibility (_CDIAM/_AP1 se zadávají jako rádius) 1 = _CDIAM/_AP1 se zadávají jako průměr
25		_AMODE	Alternativní režim
			JEDNOTKY: Hloubka kapsy (Z1)
			0 = absolutně (režim kompatibility) 1 = inkrementálně
			DESÍTKY: Jednotky pro přísuv do šířky (DXY)
			0 = mm 1 = procentuální hodnota z průměru nástroje
			STOVKY: Hloubka zajíždění při srážení hran (ZFS)
			0 = absolutně 1 = inkrementálně

16.1.17 Frézování pravoúhlého čepu - CYCLE76

Programování

```
CYCLE76(REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _DPR, REAL
_LENG, REAL _WID, REAL _CRAD, REAL _PA, REAL _PO, REAL _STA, REAL
_MID, REAL _FAL, REAL _FALD, REAL _FFP1, REAL _FFD, INT _CDIR, INT
_VARI, REAL _AP1, REAL _AP2, REAL _FS, REAL _ZFS, INT _GMODE, INT
_DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	RP	_RTP	Návratová rovina (abs)
2	Z0	_RFP	Vztažný bod osy nástroje (abs)
3	SC	_SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
4	Z1	_DP	Hloubka čepu (abs)
5		_DPR	Hloubka čepu (ink), vztažená na Z0 (zadává se bez znaménka)
6	L	_LENG	Délka čepu, viz _GMODE (zadává se bez znaménka)
7	W	_WID	Šířka čepu, viz _GMODE (zadává se bez znaménka)
8	R	_CRAD	Rádius v rohu čepu (zadává se bez znaménka)
9	X0	_PA	Vztažný bod čepu, 1. osa v rovině (abs)
10	Y0	_PO	Vztažný bod čepu, 2. osa v rovině (abs)
11	α0	_STA	Úhel otočení, úhel mezi podélnou osou (L) a 1. osou v rovině
12	DZ	_MID	Maximální přísuv do hloubky (ink, zadává se bez znaménka)
13	UXY	_FAL	Přídavek rozměru pro opracování načisto v rovině (ink), přídavek rozměru na okraji kontury
14	UZ	_FALD	Přídavek rozměru pro obrábění načisto do hloubky (ink), přídavek rozměru na dně (zadává se bez znaménka)
15	FX	_FFP1	Posuv na kontuře
16	FZ	_FFD	Rychlost přísuvu do hloubky
17		_CDIR	Směr frézování (zadává se bez znaménka)
			JEDNOTKY:
			0 = sousledné frézování
			1 = nesousledné frézování
18		_VARI	Opracování
			JEDNOTKY:
			1 = Obrábění nahrubo
			2 = Obrábění načisto
			5 = Srážení hran
19	L1	_AP1	Délka surového čepu
20	W1	_AP2	Šířka surového čepu
21	FS	_FS	Šířka fasety (ink)
22	ZFS	_ZFS	Hloubka zajíždění (špička nástroje) při srážení hran (abs, ink), viz _AMODE

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
23		_GMODE	Režim pro vyhodnocování naprogramované geometrické hodnoty
			JEDNOTKY: rezervováno
			DESÍTKY: rezervováno
			STOVKY: Volba opracování nebo pouze výpočet počátečního bodu
			0 = režim kompatibility
			1 = normální opracování
			TISÍCE: Kótování čepu pomocí středu nebo rohu
			0 = režim kompatibility
			1 = kótování pomocí středu
			2 = kótování pomocí rohového bodu, čep +L +W
			3 = kótování pomocí rohového bodu, čep -L +W
			4 = kótování pomocí rohového bodu, čep +L -W
			5 = kótování pomocí rohového bodu, čep -L -W
			DESÍTKY TISÍC: Kompletní opracování nebo dopracování
			0 = režim kompatibility
			1 = kompletní opracování
			2 = dopracování
24		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
25		_AMODE	Alternativní režim
			JEDNOTKY: Konečná hloubka Z1 (abs/ink)
			0 = kompatibilita
			1 = Z1 (ink)
			2 = Z1 (abs)
			DESÍTKY: rezervováno
			STOVKY: Hloubka zajíždění při srážení hran ZFS
			0 = ZFS (abs)
			1 = ZFS (ink)

16.1.18 Frézování kruhového čepu - CYCLE77

Programování

```
CYCLE77(REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _DPR, REAL
_CDIAM, REAL _PA, REAL _PO, REAL _MID, REAL _FAL, REAL _FALD, REAL
_FFP1, REAL _FFD, INT _CDIR, INT _VARI, REAL _AP1, REAL _FS, REAL
_ZFS, INT _GMODE, INT _DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	RP	_RTP	Návratová rovina (abs)
2	Z0	_RFP	Vztažný bod osy nástroje (abs)
3	SC	_SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
4	Z1	_DP	Hloubka čepu (abs)
5		_DPR	Hloubka čepu (ink), vztažená na Z0 (zadává se bez znaménka)
6	∅	_CDIAM	Průměr čepu (zadává se bez znaménka)
7	X0	_PA	Vztažný bod čepu, 1. osa v rovině (abs)
8	Y0	_PO	Vztažný bod čepu, 2. osa v rovině (abs)
9	DZ	_MID	Maximální přísuv do hloubky (ink, zadává se bez znaménka)
10	UXY	_FAL	Přídavek rozměru pro opracování načisto v rovině (ink), přídavek rozměru na okraji kontury
11	UZ	_FALD	Přídavek rozměru pro obrábění načisto do hloubky (ink), přídavek rozměru na dně (zadává se bez znaménka)
12	FX	_FFP1	Posuv na kontuře
13	FZ	_FFD	Rychlost přísuvu do hloubky
14		_CDIR	Směr frézování (zadává se bez znaménka)
			JEDNOTKY:
			0 = sousledné frézování
			1 = nesousledné frézování
15		_VARI	Opracování
			JEDNOTKY:
			1 = Obrábění nahrubo až na přídavek rozměru pro opracování načisto
			2 = Obrábění načisto (přídavek rozměru X/Y/Z = 0)
			5 = Srážení hran
16	∅ 1	_AP1	Průměr surového čepu
17	FS	_FS	Šířka fasety (ink)
18	ZFS	_ZFS	Hloubka zajíždění (špička nástroje) při srážení hran (abs/ink), viz _AMODE

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
19		_GMODE	Režim pro vyhodnocování naprogramované geometrické hodnoty
			JEDNOTKY: rezervováno
			DESÍTKY: rezervováno
			STOVKY: Volba opracování/pouze výpočet počátečního bodu
			0 = režim kompatibility
			1 = normální opracování
			TISÍCE: rezervováno
			DESÍTKY TISÍC: Kompletní opracování/dopracování
			0 = režim kompatibility (_AP1 je zpracováno jako dříve)
			1 = kompletní opracování
			2 = dopracování
20		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
21		_AMODE	Alternativní režim
			JEDNOTKY: Konečná hloubka Z1 (abs/ink)
			0 = kompatibilita
			1 = Z1 (ink)
			2 = Z1 (abs)
			DESÍTKY: rezervováno
			STOVKY: Hloubka zajíždění při srážení hran ZFS
			0 = ZFS (abs)
			1 = ZFS (ink)

16.1.19 Mnohohran - CYCLE79

Programování

```
CYCLE79(REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, INT _NUM, REAL
_SWL, REAL _PA, REAL _PO, REAL _STA, REAL _RC, REAL _AP1, REAL _MIDA,
REAL _MID, REAL _FAL, REAL _FALD, REAL _FFP1, INT _CDIR, INT _VARI,
REAL _FS, REAL _ZFS, INT _GMODE, INT _DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	RP	_RTP	Návratová rovina (abs)
2	Z0	_RFP	Vztažný bod osy nástroje (abs)
3	SC	_SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
4	Z1	_DP	Hloubka mnohohranu (abs/ink), viz _AMODE
5	N	_NUM	Počet hran (1...n)
6	SW / L	_SWL	Velikost klíče nebo délka hrany (v závislosti na _VARI) ("SW" v případě velikosti klíče, "L" v případě délky hrany) Velikost klíče jen u sudého počtu hran a jednohranu
7	X0	_PA	Vztažný bod čepu, 1. osa (abs)
8	Y0	_PO	Vztažný bod čepu, 2. osa (abs)
9	α0	_STA	Úhel otočení středu hrany vzhledem k 1. ose (osa X)
10	R1/FS1	_RC	Rohové zaoblení v případě _NUM > 2 (rádius/faseta, viz _AMODE) (ink, zadává se bez znaménka) ("R1" v případě rádiusu, "FS1" v případě fasety)
11	∅	_AP1	Průměr surového čepu
12	DXY	_MIDA	Maximální přísuv do šířky (jednotky viz _AMODE)
13	DZ	_MID	Maximální přísuv do hloubky
14	UXY	_FAL	Přídavek rozměru pro opracování načisto v rovině
15	UZ	_FALD	Přídavek rozměru pro opracování načisto na dně
16	F	_FFP1	Pracovní posuv
17		_CDIR	Směr frézování
			0 = sousledné frézování
			1 = nesousledné frézování

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
18		_VARI	Způsob opracování
			JEDNOTKY: Opracování
			1 = Obrábění nahrubo
			2 = Obrábění načisto
			3 = Obrábění okraje načisto
			5 = Srážení hran
			DESÍTKY: Velikost klíče nebo délka hrany
			0 = velikost klíče
			1 = délka hrany
19	FS	_FS	Šířka fasety (ink)
20	ZFS	_ZFS	Hloubka zajíždění (špička nástroje) při srážení hran (abs/ink), viz _AMODE
21		_GMODE	Geometrický režim
			JEDNOTKY: rezervováno
			DESÍTKY: rezervováno
			STOVKY: Opracování/výpočet počátečního bodu
			1 = normální opracování
22		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
23		_AMODE	Alternativní režim
			JEDNOTKY: Konečná hloubka (_DP)
			0 = absolutně
			1 = inkrementálně
			DESÍTKY: Jednotky pro přísuv v rovině (_MIDA)
			0 = mm
			1 = procentuální hodnota z průměru nástroje
			STOVKY: Hloubka zajíždění při srážení hran (_ZFS)
			0 = absolutně
			1 = inkrementálně
			TISÍCE: Rohové zaoblení (_RC)
			0 = rádius
			1 = faseta

16.1.20 Podélná drážka - SLOT1

Programování

```
SLOT1 (REAL RTP, REAL RFP, REAL SDIS, REAL _DP, REAL _DPR, INT NUM,
REAL LENG, REAL WID, REAL _CPA, REAL _CPO, REAL RAD, REAL STA1, REAL
INDA, REAL FFD, REAL FFP1, REAL _MID, INT CDIR, REAL _FAL, INT VARI,
REAL _MIDF, REAL FFP2, REAL SSF, REAL _FALD, REAL _STA2, REAL _DP1,
INT _UMODE, REAL _FS, REAL _ZFS, INT _GMODE, INT _DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	RP	RTP	Návratová rovina (abs)
2	Z0	RFP	Vztažný bod osy nástroje (abs)
3	SC	SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
4	Z1	_DP	Hloubka drážky (abs)
5		_DPR	Hloubka drážky (ink), vztažená na Z0 (zadává se bez znaménka)
6		NUM	Počet drážek = 1
7	L	LENG	Délka drážky
8	W	WID	Šířka drážky
9	X0	_CPA	Vztažný bod, 1. osa roviny
10	Y0	_CPO	Vztažný bod, 2. osa roviny
11		_RAD	rezervováno
12	α	STA1	Úhel otočení
13		INDA	rezervováno
14	FZ	FFD	Rychlost přísuvu do hloubky
15	F	FFP1	Posuv
16	DZ	_MID	Maximální přísuv do hloubky
17		CDIR	Směr frézování 0 = sousledné frézování 1 = nesousledné frézování
18	UXY	_FAL	Přídavek rozměru pro opracování načisto v rovině nebo na okrajích drážky

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
19		VARI	Způsob opracování
			JEDNOTKY:
			0 = rezervováno
			1 = Obrábění nahrubo
			2 = Obrábění načisto
			4 = Obrábění načisto okraje (opracování jen okraje)
			5 = Srážení hran
			DESÍTKY: Najíždění
			0 = předvrtáno, přísuv s G0 (drážka je předem opracována)
			1 = kolmo, přísuv s G1
			2 = po spirále
			3 = oscilačním pohybem
			STOVKY: rezervováno
20	DZF	MIDF	rezervováno
21	FF	FFP2	rezervováno
22	SF	SSF	rezervováno
23	UZ	_FALD	Přídavek rozměru pro opracování načisto na dně
24	ER	_STA2	Rádius spirální dráhy při zajíždění po šroubovici
	EW		Maximální úhel zajíždění pro oscilační pohyb
25	EP	_DP1	Hloubka zajíždění na jednu otáčku pro spirální pohyb
26		_UMODE	rezervováno
27	FS	_FS	Šířka fasety (ink) u srážení hran
28	ZFS	_ZFS	Hloubka zajíždění (špička nástroje) při srážení hran (abs/ink), viz _AMODE
29		_GMODE	Geometrický režim
			JEDNOTKY: rezervováno
			DESÍTKY: rezervováno
			STOVKY: Volba opracování nebo pouze výpočet počátečního bodu
			1 = normální opracování
			TISÍCE: Kótování vztažného bodu, poloha drážky
			0 = střed
			1 = vlevo uvnitř +L
			2 = vpravo uvnitř -L
			3 = levý okraj +L
			4 = pravý okraj -L

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
30		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
			DESÍTKY: rezervováno
			STOVKY: rezervováno
			TISÍCE: Identifikace verze SW
			1 = rozšíření funkcí SLOT1
31		_AMODE	Alternativní režim
			JEDNOTKY: Konečná hloubka Z1 (abs/ink)
			0 = kompatibilita
			1 = Z1 (ink)
			2 = Z1 (abs)
			DESÍTKY: rezervováno
			STOVKY: Hloubka zajíždění při srážení hran ZFS
			0 = ZFS (abs)
			1 = ZFS (ink)

Poznámka

Cyklus je oproti předcházejícím verzím SW vybaven novými funkcemi. To má za následek, že určité parametry se na vstupní obrazovce už nevypisují (NUM, RAD, INDA). Větší počet drážek na polohovacím vzoru je možné naprogramovat pomocí příkazu "MCALL" a voláním požadovaného polohovacího vzoru, např. HOLES2.

16.1.21 Kruhová drážka - SLOT2

Programování

```
SLOT2(REAL RTP, REAL RFP, REAL SDIS, REAL _DP, REAL _DPR, INT NUM,
REAL AFSL, REAL WID, REAL _CPA, REAL _CPO, REAL RAD, REAL STA1, REAL
INDA, REAL FFD, REAL FFP1, REAL _MID, INT CDIR, REAL _FAL, INT VARI,
REAL _MIDF, REAL FFP2, REAL SSF, REAL _FFCP, INT _UMODE, REAL _FS,
REAL _ZFS, INT _GMODE, INT _DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	RP	RTP	Návratová rovina (abs)
2	Z0	RFP	Vztažný bod osy nástroje (abs)
3	SC	SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
4	Z1	_DP	Hloubka drážky (abs)
5		_DPR	Hloubka drážky (ink), vztažená na Z0 (zadává se bez znaménka)
6	N	NUM	Počet drážek
7	α1	AFSL	Úhel kruhové výseče dané drážky
8	W	WID	Šířka drážky
9	X0	_CPA	Vztažný bod = střed kruhového oblouku, 1. osa roviny
10	Y0	_CPO	Vztažný bod = střed kruhového oblouku, 2. osa roviny
11	R	RAD	Rádius kruhového oblouku
12	α0	STA1	Počáteční úhel
13	α2	INDA	Úhlový krok
14	FZ	FFD	Rychlost přísuvu do hloubky
15	F	FFP1	Posuv
16	DZ	_MID	Maximální přísuv do hloubky
17		CDIR	Směr frézování 0 = sousledné frézování 1 = nesousledné frézování
18	UXY	_FAL	Přídavek rozměru pro opracování načisto v rovině nebo na okrajích drážky
19		VARI	Způsob opracování
			JEDNOTKY:
			0 = kompletní opracování 1 = Obrábění nahrubo 2 = Obrábění načisto 3 = Obrábění okraje načisto 5 = Srážení hran
			DESÍTKY:
			0 = najíždění na pomocné polohy po přímkách s G0 1 = najíždění na pomocné polohy po kruhové dráze
			STOVKY: rezervováno
			TISÍCE:
			0 = režim kompatibility, pokud je INDA = 0, pak celá kružnice, INDA <> 0, potom kruhový oblouk 1 = ceká kružnice 2 = kruhový oblouk

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
20	DZF	_MIDF	rezervováno
21		FFP2	rezervováno
22		SSF	rezervováno
23	FF	_FFCP	rezervováno
24		_UMODE	rezervováno
25	FS	_FS	Šířka fasety (ink)
26	ZFS	_ZFS	Hloubka zajíždění (špička nástroje) při srážení hran (abs/ink), viz _AMODE
27		_GMODE	Geometrický režim
			JEDNOTKY: rezervováno
			DESÍTKY: rezervováno
			STOVKY: Volba opracování nebo pouze výpočet počátečního bodu
			0 = režim kompatibility
			1 = normální opracování
28		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
			DESÍTKY: rezervováno
			STOVKY: rezervováno
			TISÍCE: Identifikace verze SW
			1 = funkce SLOT2 od SW verze 2.5
29		_AMODE	Alternativní režim
			JEDNOTKY: Konečná hloubka Z1 (abs/ink)
			0 = kompatibilita
			1 = Z1 (ink)
			2 = Z1 (abs)
			DESÍTKY: rezervováno
			STOVKY: Hloubka zajíždění při srážení hran ZFS
			0 = ZFS (abs)
			1 = ZFS (ink)

16.1.22 Frézování otevřené drážky - CYCLE899

Programování

```
CYCLE899(REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _LENG, REAL
_WID, REAL _PA, REAL _PO, REAL _STA, REAL _MID, REAL _MIDA, REAL
_FAL, REAL _FALD, REAL _FFP1, INT _CDIR, INT _VARI, INT _GMODE, INT
_DMODE, INT _AMODE, INT _UMODE, REAL _FS, REAL _ZFS)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	RP	_RTP	Návratová rovina (abs)
2	Z0	_RFP	Vztažný bod osy nástroje (abs)
3	SC	_SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
4	Z1	_DP	Hloubka drážky (abs/ink), viz _AMODE
5	L	_LENG	Délka drážky (ink)
6	W	_WID	Šířka drážky (ink)
7	X0	_PA	Vztažný bod/počáteční pozice, 1. osa (abs)
8	Y0	_PO	Vztažný bod/počáteční pozice, 2. osa (abs)
9	α0	_STA	Úhel otočení vzhledem k 1. ose
10	DZ	_MID	Maximální přísuvná hloubka (ink), pouze u trochoidálního frézování
11	DXY	_MIDA	Maximální přísuv v rovině, viz _AMODE
12	UXY	_FAL	Přídavek rozměru pro opracování načisto v rovině
13	UZ	_FALD	Přídavek rozměru pro opracování načisto na dně
14	F	_FFP1	Posuv
15		_CDIR	Směr frézování
			JEDNOTKY:
			0 = sousledné frézování
			1 = nesousledné frézování
			4 = střídavě

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
16		_VARI	Opracování
			JEDNOTKY:
			1 = Obrábění nahrubo
			2 = Obrábění načisto
			3 = Obrábění dna načisto
			4 = Obrábění okraje načisto
			5 = předběžné obrábění načisto
			6 = Srážení hran
			DESÍTKY: rezervováno
			STOVKY: rezervováno
			TISÍCE:
			1 = trochoidální frézování
			2 = zapichovací frézování
17		_GMODE	Vyhodnocování geometrických hodnot
			JEDNOTKY: rezervováno
			DESÍTKY: rezervováno
			STOVKY: Volba opracování/pouze výpočet počátečního bodu
			1 = normální opracování
			TISÍCE: Zadávání rozměrů pomocí středu / hrany
			0 = kótování pomocí středu
			1 = kótování pomocí hrany "vlevo" ("-" směr 1. osy)
			2 = kótování pomocí hrany "vpravo" ("+" směr 1. osy)
18		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
19		_AMODE	Alternativní režim
			JEDNOTKY: Hloubka drážky Z1
			0 = absolutně
			1 = inkrementálně
			DESÍTKY: Jednotky pro přísuv v rovině (_MIDA) DXY
			0 = mm
			1 = procentuální hodnota z průměru nástroje
			STOVKY: Hloubka zajíždění při srážení hran ZFS
			0 = absolutně
			1 = inkrementálně
20		_UMODE	rezervováno
21	FS	_FS	Šířka fasety (ink)
22	ZFS	_ZFS	Hloubka zajíždění (špička nástroje) při srážení hran (abs/ink), viz _AMODE

16.1.23 Podlouhlá díra - LONGHOLE

Programování

```
LONGHOLE(REAL RTP,REAL RFP,REAL SDIS,REAL _DP,REAL _DPR,
INT NUM,REAL LENG,REAL _CPA,REAL _CPO,REAL RAD,REAL STA1,
REAL INDA,REAL FFD,REAL FFP1,REAL MID,INT _VARI,INT _UMODE,
INT _GMODE,INT _DMODE,INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Parametr interní	Vysvětlení
1	RP	RTP	Návratová rovina (abs)
2	Z0	_RFP	Vztažný bod osy nástroje (abs)
3	SC	SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
4	Z1	_DP	Hloubka podlouhlé díry (abs)
5		_DPR	Hloubka podlouhlé díry (ink), vztažená na Z0 (zadává se bez znaménka)
6		NUM	Počet podlouhlých děr = 1
7	L	LENG	Délka podlouhlé díry
8	X0	_CPA	Vztažný bod, 1. osa roviny
9	Y0	_CPO	Vztažný bod, 2. osa roviny
10		RAD	rezervováno
11	α0	STA1	Úhel otočení
12		INDA	rezervováno
13	FZ	FFD	Rychlost přísuvu do hloubky
14	F	FFP1	Posuv
15	DZ	MID	Maximální přísuv do hloubky
16		_VARI	Způsob opracování
			JEDNOTKY: Druh přísuvu
			1 = kolmo s G1
			3 = oscilačním pohybem
			STOVKY: rezervováno
17		_UMODE	rezervováno

Č.	Obrazovka parametrů	Parametr interní	Vysvětlení
18		_GMODE	Geometrický režim
			JEDNOTKY: rezervováno
			DESÍTKY: rezervováno
			STOVKY: Volba opracování nebo pouze výpočet počátečního bodu
			0 = režim kompatibility
			1 = normální opracování
			TISÍCE: Kótování vztažného bodu, poloha drážky
			0 = střed
			1 = vlevo uvnitř +L
			2 = vpravo uvnitř -L
			3 = levý okraj +L
			4 = pravý okraj -L
19		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
			DESÍTKY: Druh posuvu: G-skupina (G94/G95) pro posuv v rovině a posuv do hloubky
			0 = režim kompatibility
			1 = G-kód stejný jako před voláním cyklu. G94/G95 pro posuv v rovině a posuv do hloubky zůstávají stejné.
			STOVKY: rezervováno
			TISÍCE: Identifikace verze SW
			1 = rozšíření funkce LONGHOLE (kótování vztažného bodu)
20		_AMODE	Alternativní režim
			JEDNOTKY: Konečná hloubka Z1 (abs/ink)
			0 = kompatibilita
			1 = Z1 (ink)
			2 = Z1 (abs)

Poznámka

Cyklus je oproti předcházejícím verzím SW vybaven novými funkcemi. To má za následek, že určité parametry se na vstupní obrazovce už nevypisují (NUM, RAD, INDA). Větší počet drážek na polohovacím vzoru je možné naprogramovat pomocí příkazu "MCALL" a voláním požadovaného polohovacího vzoru, např. HOLES2.

16.1.24 Frézování závitu - CYCLE70

Programování

```
CYCLE70(REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _DIATH, REAL
_H1, REAL _FAL, REAL _PIT, INT _NT, REAL _MID, REAL _FFR, INT _TYPTH,
REAL _PA, REAL _PO, REAL _NSP, INT _VARI, INT _PITA, STRING[15]
_PITM, STRING[20] _PTAB, STRING[20] _PTABA, INT _GMODE, INT _DMODE,
INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	RP	_RTP	Návratová rovina (abs)
2	Z0	_RFP	Vztažný bod osy nástroje (abs)
3	SC	_SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
4	Z1	_DP	Délka závitu (abs, ink), viz _AMODE Zohledňovat vyjíždění na dně vyvrtané díry (minimálně polovina stoupání)
5	∅	_DIATH	Jmenovitý průměr závitu
6	H1	_H1	Hloubka závitu
7	U	_FAL	Přídavek rozměru pro opracování načisto
8	P	_PIT	Stoupání závitu (možnosti na výběr _PITA: mm, palce, MODUL, chodů/palec)
9	NT	_NT	Počet zubů na řezné destičce Délka nástroje je vždy vztažena na spodní zub!
10	DXY	_MID	Maximální přísuv za řez _MID > _H1: vše s jedním řezem
11	F	_FFR	Posuv při frézování
12		_TYPTH	Typ závitu 0 = vnitřní závit 1 = vnější závit
13	X0	_PA	Střed kružnice, 1. osa (abs)
14	Y0	_PO	Střed kružnice, 2. osa (abs)
15	αS	_NSP	Počáteční úhel (vícechodý závit)
16		_VARI	Způsob opracování
			JEDNOTKY:
			1 = Obrábění nahrubo
			2 = Obrábění načisto
			DESÍTKY:
			1 = obrábění shora dolů
			2 = obrábění zdola nahoru
			STOVKY:
			0 = pravý závit
			1 = levý závit

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
17		_PITA	Vyhodnocování stoupání závitu 0 = režim kompatibility 1 = stoupání v mm 2 = stoupání v chodech na palec (TPI) 3 = stoupání v palcích 4 = stoupání jako MODUL
18		_PITM	Řetězec jako ukazatel pro zadávání stoupání závitu (jen pro plochu)
19		_RTAB	Řetězec pro tabulku závitů ("", "ISO", "BSW", "BSP", "UNC") (jen pro pracovní plochu)
20		_PTABA	Řetězec pro volbu v tabulce závitů (např. "M 10", "M 12", ...) (jen pro pracovní plochu)
21		_GMODE	Geometrický režim
			JEDNOTKY: rezervováno
			DESÍTKY: rezervováno
			STOVKY: Opracování/výpočet počátečního bodu
			0 = režim kompatibility 1 = normální opracování
22		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu 1 = G17 (aktivní jen v cyklu) 2 = G18 (aktivní jen v cyklu) 3 = G19 (aktivní jen v cyklu)
23		_AMODE	Alternativní režim
			JEDNOTKY: Délka závitu (_DP)
			0 = absolutně 1 = inkrementálně

16.1.25 Cyklus pro gravírování - CYCLE60

Programování

```
CYCLE60(STRING[200] _TEXT, REAL _RTP, REAL _RFP, REAL _SDIS, REAL
_DP, REAL _DPR, REAL _PA, REAL _PO, REAL _STA, REAL _CP1, REAL _CP2,
REAL _WID, REAL _DF, REAL _FFD, REAL _FFP1, INT _VARI, INT _CODEP,
INT _UMODE, INT _GMODE, INT _DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1		_TEXT	Text, který má být vygravírován (maximálně 100 znaků)
2	RP	_RTP	Návratová rovina (abs)
3	Z0	_RFP	Vztažný bod osy nástroje (abs)
4	SC	_SDIS	Bezpečnostní vzdálenost (přičítá se k referenční rovině, zadává se bez znaménka)
5	Z1	_DP	Hloubka (abs), viz _AMODE
6	Z1	_DPR	Hloubka (ink), viz _AMODE
7	X0	_PA	Vztažný bod, 1. osa v rovině (abs) - pravoúhlé souřadnice, viz _VARI
	R		Vztažný bod, délka (rádius) - polární souřadnice, viz _VARI
8	Y0	_PO	Vztažný bod, 2. osa v rovině (abs) - pravoúhlé souřadnice, viz _VARI
	α0		Vztažný bod, úhel vztažený na 1. osu - polární souřadnice, viz _VARI
9	α1	_STA	Směr textu, úhel textové čáry vztažený na 1. osu, viz _VARI
10	XM	_CP1	Střed textového kruhového oblouku, 1. osa v rovině (abs) - pravoúhlé souřadnice, viz _VARI
	LM		Střed textového kruhového oblouku, délka (rádius) vztažená na počátek WCS - polární souřadnice, viz _VARI
11	YM	_CP2	: Střed textového kruhového oblouku, 2. osa v rovině (abs) - pravoúhlé souřadnice, viz _VARI
	αM		Střed textového kruhového oblouku, úhel vztažený na 1. osu - polární souřadnice, viz _VARI
12	W	_WID	Výška znaků (zadává se bez znaménka)
13	DX1 DX2	_DF	Vzdálenost mezi znaky / celková šířka, viz _VARI
	α2		Úhel kruhové výseče, viz _VARI
14	FZ	_FFD	Posuv při přísuvu do hloubky, viz _DMODE
15	F	_FFP1	Posuv pro opracovávání plochy

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
16		_VARI	Opracování (vyrovnání a vztažný bod gravírovaného textu)
			JEDNOTKY: Vztažný bod
			0: pravoúhlý
			1: polární
			DESÍTKY: Vyrovnání textu
			0: Text na přímce
			1: Text shora na kruhovém oblouku
			2: Text zdola na kruhovém oblouku
			STOVKY: rezervováno
			TISÍCE: : Vztažný bod textu ve vodorovném směru
			0: vlevo
			1: na střed
			2: vpravo
			DESÍTKY TISÍC: Vztažný bod textu ve svislém směru
			0: dole
			1: na střed
			2: nahoře
			STOVKY TISÍC: Délka textu
			0: Vzdálenost mezi znaky
			1: Celková šířka textu (jen u textu na přímce)
			2: Úhel kruhové výseče (jen u textu na kruhovém oblouku)
			MILIONY: Střed kruhu
			0: pravoúhlé (kartézské) souřadnice
			1: polární
			DESÍTKY MILIONŮ: Zrcadlově převrácené písmo
			0 = kompatibilita
			1 = zrcadlově převrácené písmo aktivováno
			2 = zrcadlově převrácené písmo deaktivováno
17		_CODEP	Číslo kódové stránky pro písmo (v současnosti pouze 1252)
18		_UMODE	rezervováno
19		_GMODE	Režim pro vyhodnocování naprogramované geometrické hodnoty
			JEDNOTKY: rezervováno
			DESÍTKY: rezervováno
			STOVKY: Volba opracování/pouze výpočet počátečního bodu
			0 = režim kompatibility
			1 = normální opracování

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
20		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17
			2 = G18
			3 = G19
			DESÍTKY: Druh posuvu: G-skupina (G94/G95) pro posuv v rovině a posuv do hloubky
			0 = režim kompatibility
			1 = G-kód stejný jako před voláním cyklu. G94/G95 pro posuv v rovině a posuv do hloubky zůstávají stejné.
21		_AMODE	Alternativní režim
			JEDNOTKY: Konečná hloubka (_DP,_DPR)
			0 = kompatibilita
			1 = inkrementálně (_DPR)
			2 = absolutně (_DP)

16.1.26 Volání kontury - CYCLE62

Programování

```
CYCLE62(STRING[140] _KNAME, INT _TYPE, STRING[32] _LAB1, STRING[32]
_LAB2)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	PRG/ CON	_KNAME	Název kontury nebo podprogramu, nemusí být naprogramován, když je _TYPE = 2
2		_TYPE	Stanovení zadání kontury 0 = podprogram 1 = název kontury 2 = návěští 3 = Návěští v podprogramu
3	LAB1	_LAB1	Návěští 1, začátek kontury
4	LAB2	_LAB2	Návěští 2, konec kontury

16.1.27 Frézování po dráze - CYCLE72

Programování

```
CYCLE72(STRING[141] _KNAME, REAL _RTP, REAL _RFP, REAL _SDIS, REAL
_DP, REAL _MID, REAL _FAL, REAL _FALD, REAL _FFP1, REAL _FFD, INT
_VARI, INT _RL, INT _AS1, REAL __LP1, REAL _FF3, INT _AS2, REAL _LP2,
INT _UMODE, REAL _FS, REAL _ZFS, INT _GMODE, INT _DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1		_KNAME	Název podprogramu kontury
2	RP	_RTP	Návratová rovina (abs)
3	Z0	_RFP	Vztažný bod osy nástroje (abs)
4	SC	_SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
5	Z1	_DP	Koncový bod, konečná hloubka (abs/ink), viz _AMODE
6	DZ	_MID	Maximální přísuv do hloubky (ink, zadává se bez znaménka)
7	UXY	_FAL	Přídavek rozměru pro opracování načisto v rovině (ink), přídavek rozměru na okraji kontury
8	UZ	_FALD	Přídavek rozměru pro obrábění načisto do hloubky (ink), přídavek rozměru na dně (zadává se bez znaménka)
9	FX	_FFP1	Posuv na kontuře
10	FZ	_FFD	Posuv pro přísuv do hloubky (nebo přísuv v prostoru)
11		_VARI	Způsob opracování
			JEDNOTKY: Opracování
			1 = Obrábění nahrubo
			2 = Obrábění načisto
			5 = Srážení hran
			DESÍTKY:
			0 = Pomocné úseky dráhy s G0
			1 = Pomocné úseky dráhy s G1
			STOVKY:
			0 = Zpětný pohyb na konci kontury na vztažný bod
			1 = Zpětný pohyb na konci kontury na vztažný bod + _SDIS
			2 = Zpětný pohyb na konci kontury o _SDIS
			3 = žádný zpětný pohyb na konci kontury, na následující počáteční bod se najíždí s posuvem po kontuře
			TISÍCE: rezervováno
			DESÍTKY TISÍC:
			0 = konturu opracovat směrem dopředu
			1 = konturu opracovat směrem pozpátku
			Omezení při pohybu směrem pozpátku • max. 170 konturových prvků (včetně faset nebo zaoblení) • jsou vyhodnocovány pouze hodnoty v rovině (X/Y) a hodnota F

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
12		_RL	Směr obrábění
			40 = objíždění kontury středem nástroje (G40, najíždění a odjíždění: pouze po přímce nebo kolmo)
			41 = vlevo od kontury (G41, najíždění a odjíždění: pouze po přímce nebo po kruhové dráze)
			42 = vpravo od kontury (G42, najíždění a odjíždění: pouze po přímce nebo po kruhové dráze)
13		_AS1	Najížděcí pohyb na konturu
			JEDNOTKY:
			1 = přímka
			2 = čtvrtkruh
			3 = půlkruh
			4 = najíždění a odjíždění kolmo
			DESÍTKY:
			0 = poslední pohyb, v rovině.
			1 = poslední pohyb, v prostoru
14	L1	_LP1	Najížděcí dráha nebo najížděcí rádius (ink, zadává se bez znaménka)
15	FZ	_FF3	Posuv pro pomocné úseky dráhy (G94/G95, jako na kontuře)
16		_AS2	Odjížděcí pohyb od kontury (nikoli při najíždění/odjíždění kolmo)
			JEDNOTKY:
			1 = přímka
			2 = čtvrtkruh
			3 = půlkruh
			DESÍTKY:
			0 = poslední pohyb, v rovině.
			1 = poslední pohyb, v prostoru
17	L2	_LP2	Odjížděcí dráha nebo odjížděcí rádius (ink, zadává se bez znaménka)
18		_UMODE	rezervováno
19	FS	_FS	Šířka fasety (ink)
20	ZFS	_ZFS	Hloubka zajíždění (špička nástroje) při srážení hran (abs/ink), viz _AMODE
21		_GMODE	Režim pro vyhodnocování naprogramované geometrické hodnoty
			JEDNOTKY: rezervováno
			DESÍTKY: rezervováno
			STOVKY: Volba opracování/pouze výpočet počátečního bodu
			0 = režim kompatibility
			1 = normální opracování

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
22		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
			DESÍTKY: Druh posuvu: G-skupina (G94/G95) pro posuv v rovině a posuv do hloubky
			0 = režim kompatibility
			1 = G-kód stejný jako před voláním cyklu. G94/G95 pro posuv v rovině a posuv do hloubky zůstávají stejné.
			TISÍCE:
			0 = režim kompatibility: název kontury je uložen v _KNAME
			1 = název kontury je naprogramován v cyklu CYCLE62 a předává se v parametru _SC_CONT_NAME
23		_AMODE	Alternativní režim
			JEDNOTKY: Koncový bod Z1 (_DP)
			0 = absolutně (režim kompatibility)
			1 = inkrementálně
			DESÍTKY: Jednotky pro přísuv v rovině
			0 = mm, palce
			1 = rezervováno
			STOVKY: Hloubka zajíždění při srážení hran (_ZFS)
			0 = absolutně
			1 = inkrementálně

Poznámka

Pokud jsou následující předávané parametry naprogramovány nepřímo (jako parametry), zpětný překlad vstupní obrazovky nebude možný:

_VARI, _RL, _AS1, _AS2, _UMODE, _GMODE, _DMODE. _AMODE

16.1.28 Předvrtání konturové kapsy - CYCLE64

Programování

```
CYCLE64(STRING[100] _PRG, INT _VARI, REAL _RP, REAL _Z0, REAL _SC,
REAL _Z1, REAL _F, REAL _DXY, REAL _UXY, REAL _UZ, INT _CDIR,
STRING[20] _TR, INT _DR, INT _UMODE, INT _GMODE, INT _DMODE, INT
_AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	PRG	_PRG	Název programu pro vrtání/navrtávání středicích důlků
2		_VARI	Způsob opracování
			JEDNOTKY: rezervováno
			DESÍTKY: rezervováno
			STOVKY: rezervováno
			TISÍCE: Způsob pozvednutí nástroje
			0 = pozvednout na návratovou rovinu
			1 = pozvednout na vztažný bod + bezpečnostní vzdálenost
3	RP	_RP	Návratová rovina (abs)
4	Z0	_Z0	Vztažný bod (abs)
5	SC	_SC	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
6	Z1	_Z1	Hloubka vrtání/navrtávání středicích důlků, (viz JEDNOTKY parametru _AMODE)
7	F	_F	Posuv pro vrtání/navrtávání středicích důlků
8	DXY	_DXY	Přísuv v rovině - jednotky (viz DESÍTKY parametru _AMODE)
9	UXY	_UXY	Přídavek rozměru pro opracování načisto v rovině
10	UZ	_UZ	Přídavek rozměru pro opracování načisto na dně
11		_CDIR	Směr frézování
			0 = sousledné frézování
			1 = nesousledné frézování
12	TR	_TR	Název referenčního nástroje
13	DR	_DR	D-číslo referenčního nástroje
14		_UMODE	rezervováno
15		_GMODE	Režim pro vyhodnocování naprogramované geometrické hodnoty
			JEDNOTKY: rezervováno
			DESÍTKY: rezervováno
			STOVKY: Volba opracování/pouze výpočet počátečního bodu
			0 = normální opracování (není zapotřebí žádný režim kompatibility)
			1 = normální opracování
			2 = rezervováno

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
25		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
			DESÍTKY: technologický režim
			1 = předvrtání
			2 = navrtávání středicích důlků
26		_AMODE	Alternativní režim
			JEDNOTKY: Hloubka vrtání/navrtávání středicích důlků
			0 = absolutně (režim kompatibility)
			1 = inkrementálně
			DESÍTKY: : Jednotky pro přísuv v rovině (_DXY)
			0 = mm
			1 = procentuální hodnota z průměru nástroje

16.1.29 Frézování konturové kapsy - CYCLE63

Programování

```
CYCLE63(STRING[100] _PRG, INT _VARI, REAL _RP, REAL _Z0, REAL _SC,
REAL _Z1, REAL _F, REAL _FZ, REAL _DXY, REAL _DZ, REAL _UXY, REAL
_UZ, INT _CDIR, REAL _XS, REAL _YS, REAL _ER, REAL _EP, REAL _EW,
REAL _FS, REAL _ZFS, STRING[20] _TR, INT _DR, INT _UMODE, INT _GMODE,
INT _DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	PRG	_PRG	Název programu pro odstraňování materiálu
2		_VARI	Způsob opracování
			JEDNOTKY: Technologie opracování
			1 = Obrábění nahrubo 3 = Obrábění dna načisto 4 = Obrábění okraje načisto 5 = Srážení hran
			DESÍTKY: Způsob přísuvu
			0 = zajíždění středem 1 = zajíždění po spirále 2 = zajíždění oscilačním pohybem
			STOVKY: rezervováno
			TISÍCE: Způsob pozvednutí nástroje
			0 = pozvednout na návratovou rovinu 1 = pozvednout na vztažný bod + bezpečnostní vzdálenost
			DESÍTKY TISÍC: Počáteční bod při obrábění nahrubo a načisto na dně
			0 = automaticky 1 = manuálně
3	RP	_RP	Návratová rovina (abs)
4	Z0	_Z0	Vztažný bod osy nástroje (abs)
5	SC	_SC	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
6	Z1	_Z1	Konečná hloubka, (viz JEDNOTKY parametru _AMODE)
7	F	_F	Posuv v rovině při obrábění nahrubo/načisto
8	FZ	_FZ	Rychlost přísuvu do hloubky
9	DXY	_DXY	Přísuv v rovině - jednotky (viz DESÍTKY parametru _AMODE)
10	DZ	_DZ	Přísuv do hloubky
11	UXY	_UXY	Přídavek rozměru pro opracování načisto v rovině
12	UZ	_UZ	Přídavek rozměru pro opracování načisto na dně
13		_CDIR	Směr frézování 0 = sousledné frézování 1 = nesousledné frézování
14	XS	_XS	Souřadnice X počátečního bodu, absolutně

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
15	YS	_YS	Souřadnice Y počátečního bodu, absolutně
16	ER	_ER	Zajíždění po spirále: Rádius
17	EP	_EP	Zajíždění po spirále: Stoupání
18	EW	_EW	Zajíždění oscilačním pohybem: Maximální úhel při zajíždění
19	FS	_FS	Šířka fasety (ink) u srážení hran
20	ZFS	_ZFS	Hloubka zajíždění špičky nástroje při srážení hran (viz STOVKY parametru _AMODE)
21	TR	_TR	Název referenčního nástroje při odstraňování zbytkového materiálu
22	DR	_DR	D-číslo referenčního nástroje při odstraňování zbytkového materiálu
23		_UMODE	rezervováno
24		_GMODE	Režim pro vyhodnocování naprogramované geometrické hodnoty
			JEDNOTKY: rezervováno
			DESÍTKY: rezervováno
			STOVKY: Volba opracování/pouze výpočet počátečního bodu
			0 = normální opracování (není zapotřebí žádný režim kompatibility)
			1 = normální opracování
			2 = rezervováno
25		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
			DESÍTKY: rezervováno
			STOVKY: technologický režim
			1 = kapsa
			2 = čep
			TISÍCE: Odstraňování zbytkového materiálu
			0 = ne
			1 = ano
26		_AMODE	Alternativní režim
			JEDNOTKY: Koncová hloubka Z1
			0 = absolutně (režim kompatibility)
			1 = inkrementálně
			DESÍTKY: Jednotky pro přísuv v rovině (_DXY)
			0 = mm
			1 = procentuální hodnota z průměru nástroje
			STOVKY: Hloubka zajíždění při srážení hran (_ZFS)
			0 = absolutně
			1 = inkrementálně

16.1.30 Oddělování třísky - CYCLE951

Programování

```
CYCLE951(REAL _SPD, REAL _SPL, REAL _EPD, REAL _EPL, REAL _ZPD, REAL
_ZPL, INT _LAGE, REAL _MID, REAL _FALX, REAL _FALZ, INT _VARI, REAL
_RF1, REAL _RF2, REAL _RF3, REAL _SDIS, REAL _FF1, INT _NR, INT
_DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	X0	_SPD	Souřadnice vztažného bodu, (abs, vždycky průměr)
2	Z0	_SPL	Vztažný bod (abs)
3	X1	_EPD	Koncový bod
4	Z1	_EPL	Koncový bod
5	XM α1 α2	_ZPD	Vnitřní bod, viz _DMODE (DESÍTKY)
6	ZM α1 α2	_ZPL	Vnitřní bod, viz _DMODE (DESÍTKY)
7	Poloha	_LAGE	Poloha pro obráběného rohu 0 = vnější/vzadu 1 = vnější/vpředu 2 = vnitřní/vzadu 3 = vnitřní/vpředu
8	D	_MID	Maximální přísuv do hloubky při zajíždění nástroje
9	UX	_FALX	Přídavek rozměru pro opracování načisto ve směru X
10	UZ	_FALZ	Přídavek rozměru pro opracování načisto ve směru Z

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
11		_VARI	Způsob opracování
			JEDNOTKY: Směr oddělování třísky (podélné nebo příčné) v souřadném systému
			1 = podélné
			2 = příčné
			DESÍTKY:
			1 = Obrábění nahrubo až na přídavek rozměru pro opracování načisto
			2 = Obrábění načisto
			STOVKY:
			0 = s vyjížděním podél kontury, bez růžků zbytkového materiálu
			1 = bez vyjíždění podél kontury
			TISÍCE:
			0 = s rádiusem/fasetou v rohu 2
			1 = se zápichem v rohu 2
			DESÍTKY TISÍC:
			0 = po dokončení opracování zůstat stát
			1 = najíždět zpět na počáteční pozici
12	R1/FS1	_RF1	Rádius zaoblení nebo šířka fasety 1, viz _AMODE (DESÍTKY TISÍC)
13	R2/FS2	_RF2	Rádius zaoblení nebo šířka fasety 2, viz _AMODE (STOVKY TISÍC)
14	R3/FS3	_RF3	Rádius zaoblení nebo šířka fasety 3, viz _AMODE (MILIONY)
15	SC	_SDIS	Bezpečnostní vzdálenost
16	F	_FF1	Posuv pro obrábění nahrubo/načisto
17		_NR	Identifikace způsobu oddělování třísky (odpovídá programovým tlačítkům ve svislém pruhu na obrazovce)
			0 = oddělování třísky 1, roh 90 stupňů bez faset/zaoblení
			1 = oddělování třísky 2, roh 90 stupňů s fasetami/zaobleními
			2 = oddělování třísky 3, libovolný roh s fasetami/zaobleními
18		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
			DESÍTKY: Způsob zadávání parametrů _ZPD/_ZPL
			0 = Xm/Zm
			1 = Xm/α1
			2 = Xm/α2
			3 = α1/Zm
			4 = α2/Zm
			5 = α1/α2

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
21		_AMODE	Alternativní režim
			JEDNOTKY: Vnitřní bod, souřadnice X
			0 = absolutně, hodnota příčné osy se zadává jako průměr
			1 = inkrementálně, hodnota příčné osy se zadává jako rádius
			DESÍTKY: Vnitřní bod, souřadnice Z
			0 = absolutně
			1 = inkrementálně
			STOVKY: Koncový bod, souřadnice X
			0 = absolutně, hodnota příčné osy se zadává jako průměr
			1 = inkrementálně, hodnota příčné osy se zadává jako rádius
			TISÍCE: koncový bod v ose Z
			0 = absolutně
			1 = inkrementálně
			DESÍTKY TISÍC: Rádius / faseta 1
			0 = rádius
			1 = faseta
			STOVKY TISÍC: Rádius / faseta 2
			0 = rádius
			1 = faseta
			JEDNOTKY MILIONŮ: Rádius / faseta 3
			0 = rádius
			1 = faseta

16.1.31 Zápich - CYCLE930

Programování

```
CYCLE930(REAL _SPD, REAL _SPL, REAL _WIDG, REAL _WIDG2, REAL _DIAG,
REAL _DIAG2, REAL _STA, REAL _ANG1, REAL _ANG2, REAL _RCO1, REAL
_RCI1, REAL _RCI2, REAL _RCO2, REAL _FAL, REAL _IDEP1, REAL _SDIS,
INT _VARI, INT _DN, INT _NUM, REAL _DBH, REAL _FF1, INT _NR, REAL
_FALX, REAL _FALZ, INT _DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	X0	_SPD	Souřadnice vztažného bodu v příčné ose (vždycky průměr)
2	Z0	_SPL	Souřadnice vztažného bodu v podélné ose
3	B1	_WIDG	Šířka zápichu dole
4	B2	_WIDG2	Šířka zápichu nahoře (jen pro pracovní plochu)
5	T1	_DIAG	Hloubka zápichu ve vztažném bodě, v případě absolutních rozměrů a podélného obrábění = průměr, jinak inkrementální rozměr
6	T2	_DIAG2	Hloubka zápichu naproti vztažného bodu (jen pro pracovní plochu), v případě absolutních rozměrů a podélného obrábění = průměr, jinak inkrementální rozměr
7	α0	_STA	Úhel šikmé plochy (-180 ≤ _STA ≤ 180)
8	α1	_ANG1	Úhel stěny zápichu 1 (0 ≤ _ANG1 < 90) na straně zápichu určeného vztažným bodem
9	α2	_ANG2	Úhel stěny zápichu 2 (0 ≤ _ANG2 < 90) naproti vztažného bodu
10	R1/FS1	_RCO1	Rádius zaoblení nebo šířka fasety 1, vně u vztažného bodu
11	R2/FS2	_RCI1	Rádius zaoblení nebo šířka fasety 2, uvnitř u vztažného bodu
12	R3/FS3	_RCI2	Rádius zaoblení nebo šířka fasety 3, uvnitř naproti vztažného bodu
13	R4/FS4	_RCO2	Rádius zaoblení nebo šířka fasety 4, vně naproti vztažného bodu
14	U	_FAL	Přídavek rozměru pro opracování načisto ve směru X a Z, viz _VARI (DESÍTKY TISÍC) (zadává se bez znaménka)
15	D	_IDEP1	Maximální přísuv do hloubky při zajíždění nástroje (zadává se bez znaménka) 0 = 1. průchod nástroje přímo na celou hloubku > 0 = 1. průchod nástroje _IDEP1, 2. průchod nástroje 2 · _IDEP1 atd.
16	SC	_SDIS	Bezpečnostní vzdálenost (zadává se bez znaménka)

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
17		_VARI	Způsob opracování
			JEDNOTKY: rezervováno
			DESÍTKY: Technologie opracování
			1 = Obrábění nahrubo
			2 = Obrábění načisto
			3 = Obrábění nahrubo a obrábění načisto
			STOVKY: Poloha v podélném/příčném směru, vnější/vnitřní +Z/-Z, příp +X/-X
			1 = podélná/vnější +Z
			2 = příčná/vnitřní -X
			3 = podélná/vnitřní +Z
			4 = příčná/vnější +X
			5 = podélná/vnější -Z
			6 = příčná/vnější -X
			7 = podélná/vnitřní -Z
			8 = příčná/vnější +X
			TISÍCE: Poloha vztažného bodu
			0 = vztažný bod nahoře
			1 = vztažný bod dole
			DESÍTKY TISÍC: Definice platí jako přídavek rozměru pro opracování načisto
			0 = Přídavek rozměru pro opracování načisto U rovnoběžně s konturou
			1 = Přídavek rozměru pro opracování načisto UX a UZ odděleně
18		_DN	D-číslo pro 2. břit nástroje
			> 0 = D-číslo pro korekci nástroje pro 2. břit zapichovacího nože
			0 = žádný 2. břit není naprogramován
19	N	_NUM	Počet zápichů (0 = 1 zápich)
20	DP	_DBH	Vzdálenost zápichů (je zapotřebí jen tehdy, pokud je _NUM > 1)
21	F	_FF1	Posuv
22		_NR	Identifikace tvaru zápichu (odpovídá programovým tlačítkům ve svislém pruhu obrazovky)
			0 = 90° stěny zápichů bez faset/zaoblení
			1 = šikmé stěny s fasetami/zaobleními (bez α0)
			2 = jako 1, ale na kuželu (s α0)
23	UX	_FALX	Přídavek rozměru pro opracování načisto ve směru osy X, viz _VARI (DESÍTKY TISÍC) (zadává se bez znaménka)
24	UZ	_FALZ	Přídavek rozměru pro opracování načisto ve směru osy Z, viz _VARI (DESÍTKY TISÍC) (zadává se bez znaménka)
25		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
26		_AMODE	Alternativní režim
			JEDNOTKY: Zadávání rozměrů hloubky (jen pro pracovní plochu)
			0 = u vztažného bodu
			1 = naproti vztažného bodu
			DESÍTKY: Hloubka
			0 = absolutně
			1 = inkrementálně
			STOVKY: Zadávání rozměrů šířky (jen pro pracovní plochu)
			0 = na vnějším průměru (nahoře)
			1 = na vnitřním průměru (dole)
			TISÍCE: Rádius / faseta 1 (_RCO1)
			0 = rádius
			1 = faseta
			DESÍTKY TISÍC: Rádius / faseta 2 (_RCI1)
			0 = rádius
			1 = faseta
			STOVKY TISÍC: Rádius / faseta 3 (_RCI2)
			0 = rádius
			1 = faseta
			MILIONY: Rádius / faseta 4 (_RCO2)
			0 = rádius
			1 = faseta

16.1.32 Tvary odlehčovacího zápichu - CYCLE940

Pomocí cyklu CYCLE940 mohou být naprogramovány různé odlehčovací zápichy. Pokud jde o dosazování parametrů, jsou mezi nimi značné rozdíly.

Přidané sloupce v tabulce ukazují, které parametry jsou u jednotlivých tvarů zápichu zapotřebí. Odpovídají programovým tlačítkům ve svislém pruhu na obrazovce cyklu.

- E: Odlehčovací zápich:tvar E
- F: Odlehčovací zápich:tvar F
- A-D: Závitový zápich podle DIN (tvary A - D)
- T: Závitový zápich (volná definice tvaru)

Programování

```
CYCLE940(REAL _SPD, REAL _SPL, CHAR _FORM, INT _LAGE, REAL _SDIS,
REAL _FFP,INT _VARI, REAL _EPD, REAL _EPL, REAL _R1, REAL _R2, REAL
_STA, REAL _VRT, REAL _MID, REAL _FAL, REAL _FALX, REAL _FALZ, INT
_PITI, STRING[5] _PTAB, STRING[20] _PTABA, INT _DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Progr. u tvaru				Vysvětlení
			E	F	A-D	T	
1	X0	_SPD	x	x	x	x	Souřadnice vztažného bodu v příčné ose (vždycky průměr)
2	Z0	_SPL	x	x	x	x	Vztažný bod, podélná osa (abs)
3	FORM	_FORM	x	x	x	x	Tvar zápichu (velká písmena, např. "T") Volba, ze které tabulky se mají brát hodnoty parametrů zápichu A = vnější, podle DIN 76, A = normální B = vnější, podle DIN 76, B = krátký C = vnitřní, podle DIN 76, C = normální D = vnitřní, podle DIN 76, D = krátký E = podle DIN 509 F = podle DIN 509 T = volný tvar
4	LAGE	_LAGE	x	x	x	x	Poloha zápichu (rovnoběžně s osou Z) 0 = vnější +Z: ____| 1 = vnější -Z: |____/ 2 = vnitřní +Z: /-----| 3 = vnitřní -Z: |-----\
5	SC	_SDIS	x	x	x	x	Bezpečnostní vzdálenost (ink)
6	F	_FFP	x	x	x	x	Pracovní posuv (mm/ot)

Č.	Obrazovka parametrů	Interní parametr	Progr. u tvaru				Vysvětlení			
7		_VARI	-	-	x	x	Způsob opracování			
							JEDNOTKY: Opracování			
							1 = Obrábění nahrubo			
							2 = Obrábění načisto			
							3 = Obrábění nahrubo + obrábění načisto			
							DESÍTKY: Strategie obrábění			
							0 = rovnoběžně s konturou			
							1 = podélné			
			Zápichy tvarů E a F jsou vždy opracovávány v jednom tahu jako při obrábění načisto							
8	X1	_EPD	x	x	-	-	Přídavek rozměru pro opracování načisto ve směru X (abs/ink), viz _AMODE			
			-	-	-	x	Hloubka zápichu (abs/ink), viz _AMODE			
9	Z1	_EPL	-	x	-	-	Přídavek rozměru pro opracování načisto ve směru Z			
			-	-	-	x	Šířka zápichu (abs/ink), viz _AMODE			
10	R1	_R1	-	-	-	x	Rádius zaoblení na stěnách			
11	R2	_R2	-	-	-	x	Rádius zaoblení na rozích			
12	α	_STA	-	-	x	x	Úhel zajíždění do obrobku			
13	VX	_VRT	x	x	-	-	Příčný pohyb ve směru X (abs/ink), viz _AMODE			
			-	-	x	x	Příčný pohyb ve směru X při obrábění načisto (abs/ink), viz _AMODE			
14	D	_MID	-	-	x	x	Přísuv do hloubky			
15	U	_FAL	-	-	x	x	Přídavek rozměru pro obrábění načisto rovnoběžně s konturou, viz _AMODE			
16	UX	_FALX	-	-	x	x	Přídavek rozměru pro obrábění načisto ve směru X			
17	UZ	_FALZ	-	-	x	x	Přídavek rozměru pro obrábění načisto ve směru Z			
18	P	_PITI	-	-	x	-	Volba stoupání, tvar A - D, odpovídá M1 ... M68			
							0 = 0.20	6 = 0.50	12 = 1.25	18 = 3.50
							1 = 0.25	7 = 0.60	13 = 1.50	19 = 4.00
							2 = 0.30	8 = 0.70	14 = 1.75	20 = 4.50
							3 = 0.35	9 = 0.75	15 = 2.00	21 = 5.00
							4 = 0.40	10 = 0.80	16 = 2.50	22 = 5.50
							5 = 0.45	11 = 1.00	17 = 3.00	23 = 6.00
			x	x	-	-	Volba rádiusu/hloubky, tvar E, F			
							0 = 0.6 · 0.3	4 = 2.5 · 0.4	8 = 0.1 · 0.1	
							1 = 1.0 · 0.4	5 = 4.0 · 0.5	9 = 0.2 ··0.1	
							2 = 1.0 · 0.2	6 = 0.4 · 0.2		
							3 = 1.6 ··0.3	7 = 0.6 · 0.2		
19		_PTAB					Řetězec pro tabulku závitů ("", "ISO", "BSW", "BSP", "UNC") (jen pro pracovní plochu)			
20		_PTABA					Řetězec pro volbu v tabulce závitů (např. "M 10", "M 12", ...) (jen pro pracovní plochu)			

Č.	Obrazovka parametrů	Interní parametr	Progr. u tvaru				Vysvětlení
21		_DMODE					Režim zobrazování
			x	x	x	x	JEDNOTKY: Rovina obrábění G17/G18/G19
							0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
							1 = G17 (aktivní jen v cyklu)
							2 = G18 (aktivní jen v cyklu)
							3 = G19 (aktivní jen v cyklu)
22		_AMODE					Alternativní režim
			x	x	-	x	JEDNOTKY: Parametr _EPD, přídavek rozměru ve směru X nebo hloubka odlehčovacího zápichu
							0 = absolutně (vždy průměr)
							1 = inkrementálně
			x	x	-	x	DESÍTKY: Parametr _EPL, přídavek rozměru ve směru Z nebo šířka odlehčovacího zápichu
							0 = absolutně
							1 = inkrementálně
			x	x	x	x	STOVKY: Parametr _VRT, příčný pohyb ve směru X
							0 = absolutně (vždy průměr)
							1 = inkrementálně
			-	-	x	x	TISÍCE: Přídavek rozměru pro opracování načisto
							0 = Přídavek rozměru pro opracování načisto rovnoběžně s konturou (_FAL)
							= Přídavek rozměru pro opracování načisto odděleně (_FALX/ _FALZ)

16.1.33 Soustružení závitu - CYCLE99

Programování

```
CYCLE99(REAL _SPL, REAL _SPD, REAL _FPL, REAL _FPD, REAL _APP, REAL
_ROP, REAL _TDEP, REAL _FAL, REAL _IANG, REAL _NSP, INT _NRC, INT
_NID, REAL _PIT, INT _VARI, INT _NUMTH, REAL _SDIS, REAL _MID, REAL
_GDEP, REAL _PIT1, REAL _FDEP, INT _GST, INT _GUD, REAL _IFLANK, INT
_PITA, STRING[15] _PITM, STRING[20] _PTAB, STRING[20] _PTABA, INT
_DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	Z0	_SPL	Vztažný bod (abs)
2	X0	_SPD	Souřadnice vztažného bodu, (abs, vždycky průměr)
3	Z1	_FPL	Koncový bod, viz parametr _AMODE (JEDNOTKY)
4	X1	_FPD	Koncový bod, viz parametr _AMODE (DESÍTKY)
5	LW/LW2	_APP	Předsunutý náběh závitu, viz parametr _AMODE (STOVKY) nebo Náběh závitu = výběh závitu, viz parametr _AMODE (STOVKY)
6	LR	_ROP	Výběh závitu
7	H1	_TDEP	Hloubka závitu
8	U	_FAL	Přídavek rozměru pro opracování načisto ve směrech X a Z
9	DP	-_IANG	Šikmý přísuv jako vzdálenost nebo úhel, viz _AMODE (TISÍCE)
	αP		> 0 = přísuv po kladné stěně < 0 = přísuv po záporné stěně 0 = přísuv středem
10	α0	_NSP	Úhlové posunutí počátečního bodu (uplatňuje se jen u nastavení "jednochodý")
11	ND	_NRC	Počet průchodů nástroje při obrábění nahrubo, ve spojení s _VARI (DESÍTKY TISÍC)
12	NN	_NID	Počet průchodů nástroje naprázdno
13	P	_PIT	Stoupání závitu jako hodnota, viz _PITA

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
14		_VARI	Způsob opracování
			JEDNOTKY: Technologie
			1 = vnější závit s lineárním přísuvem
			2 = vnitřní závit s lineárním přísuvem
			3 = vnější závit s degresivním přísuvem, průřez třísky zůstává konstantní
			4 = vnitřní závit s degresivním přísuvem, průřez třísky zůstává konstantní
			DESÍTKY: rezervováno
			STOVKY: Způsob přísuvu
			1 = přísuv na jedné straně
			2 = přísuv střídavě na obou stranách
			TISÍCE: rezervováno
			DESÍTKY TISÍC: Alternativní přísuv do hloubky
			0 = zadání počtu průchodů nástroje při hrubování (_NRC)
			1 = předdefinovaná hodnota pro 1. přísuv (_MID)
			STOVKY TISÍC: Způsob opracování
			1 = Obrábění nahrubo
			2 = Obrábění načisto
			3 = Obrábění nahrubo a obrábění načisto
			JEDNOTKY MILIONŮ: Posloupnost opracování u vícechodých závitů
			0 = vzestupné pořadí chodů
			1 = obrácené pořadí chodů
15	N	_NUMTH	Počet chodů závitu
16	VR	_SDIS	Návratová vzdálenost, ink
17	D1	_MID	První přísuvná hloubka, viz _VARI (DESÍTKY TISÍC)
18	DA	_GDEP	Střídavá hloubka chodů závitu
			0 = na střídavou hloubku chodů závitu nebrat ohled
			> 0 = na střídavou hloubku chodů závitu brát ohled
19	G	_PIT1	Změna stoupání na otáčku
			0 = stoupání závitu je konstantní (G33)
			> 0 = stoupání závitu se zvětšuje (G34)
			< 0 = stoupání závitu se zmenšuje (G35)
20		_FDEP	Hloubka zajíždění nástroje (zadává se bez znaménka)
21	N1	_GST	Počáteční chod N1 = 1...N, viz _AMODE (STOVKY TISÍC)
22		_GUD	rezervováno
23		_IFLANK	Šikmý přísuv jako šířka (jen pro pracovní plochu)
24		_PITA	Rozměrové jednotky pro stoupání závitu (vyhodnocování PIT a/nebo MPIT)
			0 = stoupání v mm - vyhodnocování MPIT/PIT
			1 = stoupání v mm - vyhodnocování PIT
			2 = stoupání v TPI - vyhodnocování PIT (počet chodů závitu na palec)
			3 = stoupání v palcích - vyhodnocování PIT
			4 = MODUL - vyhodnocování PIT

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
25		_PITM	Řetězec jako ukazatel pro zadávání stoupání závitu (jen pro plochu)[1)]
26		_PTAB	Řetězec pro tabulku závitů (jen pro pracovní plochu)[1)]
27		_PTABA	Řetězec pro volbu v tabulce závitů (jen pro pracovní plochu)[1)]
28		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
			DESÍTKY: Druh závitu
			0 = podélný závit
			1 = příčný závit
			2 = kuželový závit
29		_AMODE	Alternativní režim
			JEDNOTKY: Délka závitu ve směru Z
			0 = absolutně
			1 = inkrementálně
			DESÍTKY: Délka závitu ve směru X
			0 = absolutně, hodnota příčné osy se zadává jako průměr
			1 = inkrementálně, hodnota příčné osy se zadává jako rádius
			2 = α
			STOVKY: Vyhodnocování náběhu _APP
			0 = předsunutý náběh závitu _APP
			1 = náběh závitu = výběh závitu, _APP = _ROP
			2 = zadání náběhu závitu _APP = _APP
			TISÍCE: Volba šikého přísuvu jako úhlu nebo šířky
			0 = úhel přísuvu _IANG
			1 = šířka přísuvu _IFLANK
			DESÍTKY TISÍC: jednochový/vícechodý
			0 = jednochový (s úhlovým posunutím počátečního bodu _NSP)
			1 = vícechodý
			STOVKY TISÍC: Počáteční chod _GST
			0 = kompletní opracování
			1 = zahájit opracování od tohoto chodu
			2 = opracovat jen tento chod

Poznámka

1) Parametry _PITM, _PTAB a _PTABA se používají pouze při volbě závitu v tabulce závitů ve vstupní obrazovce.

Přístup k tabulkám závitů prostřednictvím definice cyklů v době, kdy jsou cykly zpracovávány, není možný.

16.1.34 Řetězec závitů - CYCLE98

Programování

```
CYCLE98(REAL _PO1, REAL _DM1, REAL _PO2, REAL _DM2, REAL _PO3, REAL
_DM3, REAL _PO4, REAL _DM4, REAL APP, REAL ROP, REAL TDEP, REAL FAL,
REAL _IANG, REAL NSP, INT NRC, INT NID, REAL _PP1, REAL _PP2, REAL
_PP3, INT _VARI, INT _NUMTH, REAL _VRT, REAL _MID, REAL _GDEP, REAL
_IFLANK, INT _PITA, STRING[15] _PITM1, STRING[15] _PITM2, STRING[15]
_PITM3, INT _DMODE,INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	Z0	_PO1	Souřadnice Z vztažného bodu (abs)
2	X1	_DM1	Souřadnice X vztažného bodu, (abs), jako průměr
3	Z1	_PO2	Vnitřní bod 1, osa Z (abs/ink), viz _AMODE (JEDNOTKY)
4	X1	_DM2	Vnitřní bod 1, osa X (abs/ink), viz _AMODE (DESÍTKY) nebo
	X1α		Úhel šikmého závitu 1 (-90° až 90°) abs vždy jako průměr, ink vždy jako rádius
5	Z2	_PO3	Vnitřní bod 2, osa Z (abs/ink), viz _AMODE (STOVKY)
6	X2	_DM3	Vnitřní bod 2, osa X (abs/ink), viz _AMODE (TISÍCE) nebo
	X2α		Úhel šikmého závitu 2 (-90° až 90°) abs vždy jako průměr, ink vždy jako rádius
7	Z3	_PO4	Koncový bod, osa Z (abs/ink), viz _AMODE (DESÍTKY TISÍC)
8	X3	_DM4	Koncový bod, osa X (abs/ink), viz _AMODE (STOVKY TISÍC) nebo
	X3α		Úhel šikmého závitu 3 (-90° až 90°) abs vždy jako průměr, ink vždy jako rádius
9	LW	APP	Předsunutý náběh závitu (ink, zadává se bez znaménka)
10	LR	ROP	Výběh závitu (ink, zadává se bez znaménka)
11	H1	TDEP	Hloubka závitu (ink, zadává se bez znaménka)
12	U	FAL	Přídavek rozměru pro opracování načisto ve směrech X a Z
13	DP	_IANG	Šikmý přísuv jako vzdálenost nebo úhel, viz _AMODE (MILIONY)
	αP		Šikmý přísuv se uplatňuje v souladu s nastavením parametru _VARI (STOVKY) Definice pro parametr _VARI (STOVKY) = 0 - režim kompatibility: > 0 = přísuv po jedné stěně 0 = přísuv kolmo do závitu < 0 = přísuv střídavě po obou stěnách Definice pro parametr _VARI (STOVKY) <> 0: > 0 = přísuv po kladné stěně 0 = přísuv středem < 0 = přísuv po záporné stěně
14	α0	NSP	Úhlové posunutí počátečního bodu pro 1. chod závitu
15		NRC	Počet průchodů nástroje při obrábění nahrubo, viz _VARI (DESÍTKY TISÍC)

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
16	NN	NID	Počet průchodů nástroje naprázdno
17	P0	_PP1	Stoupání 1. úseku závitu, viz _PITA
18	P1	_PP2	Stoupání 2. úseku závitu, viz _PITA
19	P2	_PP3	Stoupání 3. úseku závitu, viz _PITA
20		_VARI	Opracování
			JEDNOTKY: Technologie
			1 = vnější závit s lineárním přísuvem 2 = vnitřní závit s lineárním přísuvem 3 = vnější závit s degresivním přísuvem, průřez třísky zůstává konstantní 4 = vnitřní závit s degresivním přísuvem, průřez třísky zůstává konstantní
			DESÍTKY: rezervováno
			STOVKY: Způsob přísuvu
			0 = režim kompatibility pro _IANG 1 = přísuv na jedné straně 2 = přísuv střídavě na obou stranách
			TISÍCE: rezervováno
			DESÍTKY TISÍC: Alternativní přísuv do hloubky
			0 = kompatibilita, zadání počtu průchodů nástroje při hrubování (_NRC) 1 = předdefinovaná hodnota pro 1. přísuv (_MID)
			STOVKY TISÍC: Způsob opracování
			0 = kompatibilita, obrábění nahrubo a obrábění načisto 1 = Obrábění nahrubo 2 = Obrábění načisto 3 = Obrábění nahrubo a obrábění načisto
			JEDNOTKY MILIONŮ: Posloupnost opracování u vícechodých závitů
			0 = vzestupné pořadí chodů 1 = obrácené pořadí chodů
21	N	_NUMTH	Počet chodů závitu
22		_VRT	Návratová vzdálenost (ink) 0 = interně se bude používat pozvednutí o 1 mm nezávisle na systému jednotek (palce/mm) > 0 = velikost pozvednutí
23	D1	_MID	První přísuv, viz _VARI (DESÍTKY TISÍC)
24	DA	_GDEP	Střídavá hloubka chodů závitu (uplatňuje se jen u nastavení "vícechodý") 0 = na střídavou hloubku chodů závitu nebrat ohled > 0 = na střídavou hloubku chodů závitu brát ohled
25		_IFLANK	Šikmý přísuv jako šířka (jen pro pracovní plochu)
26		_PITA	Vyhodnocování stoupání závitu 0 = režim kompatibility pro stoupání závitu, vyhodnocování _PP1 až _PP3 jako dříve v souladu s aktivním systémem měřicích jednotek (metrický/palce) 1 = stoupání v mm 2 = stoupání v TPI (počet chodů závitu na palec) 3 = stoupání v palcích 4 = MODUL

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
27		_PITM1	Řetězec jako ukazatel pro zadávání stoupání závitu (jen pro plochu)
28		_PITM2	Řetězec jako ukazatel pro zadávání stoupání závitu (jen pro plochu)
29		_PITM3	Řetězec jako ukazatel pro zadávání stoupání závitu (jen pro plochu)
30		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
31		_AMODE	Alternativní režim
			JEDNOTKY: 1. Vnitřní bod, souřadnice Z (Z1)
			0 = absolutně
			1 = inkrementálně
			DESÍTKY: 1. Vnitřní bod, souřadnice X (X1)
			0 = absolutně
			1 = inkrementálně
			2 = α
			STOVKY: 2. Vnitřní bod, souřadnice Z (Z2)
			0 = absolutně
			1 = inkrementálně
			TISÍCE: 2. Vnitřní bod, souřadnice X (X2)
			0 = absolutně
			1 = inkrementálně
			2 = α
			DESÍTKY TISÍC: koncový bod v ose Z (Z3)
			0 = absolutně
			1 = inkrementálně
			STOVKY TISÍC: koncový bod, souřadnice X (X3)
			0 = absolutně
			1 = inkrementálně
			2 = α
			JEDNOTKY MILIONŮ: Volba šikého přísuvu jako úhlu nebo šířky
			0 = úhel přísuvu _IANG
			1 = šířka přísuvu _IFLANK
			DESÍTKY MILIONŮ: jednochový/vícechodý
			0 = režim kompatibility (počáteční úhel _NSP se vyhodnocuje)
			1 = jednochový (s úhlovým posunutím počátečního bodu _NSP)
			2 = vícechodý

16.1.35 Upichování - CYCLE92

Programování

```
CYCLE92(REAL _SPD, REAL _SPL, REAL _DIAG1, REAL _DIAG2, REAL _RC,
REAL _SDIS, REAL _SV1, REAL _SV2, INT _SDAC, REAL _FF1, REAL _FF2,
REAL _SS2, REAL _DIAGM, INT _VARI, INT _DN, INT _DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	X0	_SPD	Souřadnice vztažného bodu, (abs, vždycky průměr)
2	Y0	_SPL	Vztažný bod (abs)
3	X1	_DIAG1	Hloubka pro snížení otáček, viz parametr _AMODE (JEDNOTKY)
4	X2	_DIAG2	Konečná hloubka, viz parametr _AMODE (DESÍTKY)
5	R/FS	_RC	Rádius zaoblení nebo šířka fasety, viz _AMODE (TISÍCE)
6	SC	_SDIS	Bezpečnostní vzdálenost (přičítá se ke vztažnému bodu, zadává se bez znaménka)
7	S	_SV1	Konstantní otáčky vřetena, viz _AMODE (DESÍTKY TISÍC)
	V		konstantní řezná rychlost
8	SV	_SV2	Maximální otáčky při konstantní řezné rychlosti
9	DIR	_SDAC	Otáčení vřetena 3 = pro M3 4 = pro M4
10	F	_FF1	Posuv až do hloubky pro snížení otáček
11	FR	_FF2	Snížená hodnota posuvu až do konečné hloubky
12	SR	_SS2	Snížená hodnota otáček až do konečné hloubky
13	XM	_DIAGM	Hloubka pro vyjíždění podavače součástek (abs, vždycky průměr)
14		_VARI	Způsob opracování
			JEDNOTKY: Návratová dráha
			0 = zpětný pohyb na _SPD+_SDIS 1 = žádný zpětný pohyb na konci
			DESÍTKY: Podavač obrobků
			0 = ne, nevyvolávat žádný M-příkaz 1 = ano, vyvolání funkce CUST_TECHCYC(101) - vyjíždění zásuvky, CUST_TECHCYC(102) - zasunutí zásuvky
15		_DN	D-číslo pro 2. břit nástroje, pokud není naprogramováno ⇒ D+1
20		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu 1 = G17 (aktivní jen v cyklu) 2 = G18 (aktivní jen v cyklu) 3 = G19 (aktivní jen v cyklu)

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
21		_AMODE	Alternativní režim
			JEDNOTKY: Hloubka pro snížení otáček (_DIAG1)
			0 = absolutně, hodnota příčné osy se zadává jako průměr
			1 = inkrementálně, hodnota příčné osy se zadává jako rádius
			DESÍTKY: Konečná hloubka (_DIAG2)
			0 = absolutně, hodnota příčné osy se zadává jako průměr
			1 = inkrementálně, hodnota příčné osy se zadává jako rádius
			STOVKY: rezervováno
			TISÍCE: Rádius / faseta (_RC)
			0 = rádius
			1 = faseta
			DESÍTKY TISÍC: Otáčky vřetena/řezná rychlost (_SV1)
			0 = konstantní otáčky vřetena
			1 = konstantní řezná rychlost

16.1.36 Zápichy na kontuře - CYCLE952

Programování

```
CYCLE952(STRING[100] _PRG, STRING[100] _CON, STRING[100] _CONR, INT
_VARI, REAL _F, REAL _FR, REAL _RP, REAL _D, REAL _DX, REAL _DZ, REAL
_UX, REAL _UZ, REAL _U, REAL _U1, INT _BL, REAL _XD, REAL _ZD, REAL
_XA, REAL _ZA, REAL _XB, REAL _ZB, REAL _XDA, REAL _XDB, INT _N, REAL
_DP, REAL _DI, REAL _SC, INT _DN, INT _GMODE, INT _DMODE, INT _AMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	PRG	_PRG	Název programu pro oddělování třísky
2	CON	_CON	Název programu, z něhož se načítá aktualizovaná kontura surového obrobku (při obrábění zbytkového materiálu)
3	CONR	_CONR	Název programu, ve kterém je popsána aktualizovaná kontura surového obrobku (viz _AMODE, DESÍTKY TISÍC)
4		_VARI	Způsob opracování
			JEDNOTKY: Způsob opracování
			1 = podélné 2 = příčné 3 = rovnoběžně s konturou
			DESÍTKY: Technologie opracování (viz parametr _GMODE (STOVKY))
			1 = Obrábění nahrubo 2 = Obrábění načisto 3 = kompletní opracování
			STOVKY: Směr obrábění
			1 = směr obrábění X - 2 = směr obrábění X + 3 = směr obrábění Z - 4 = směr obrábění Z +
			TISÍCE: Směr přísuvu
			1 = vnější X - 2 = vnitřní X + 3 = čelní strana Z - 4 = zadní strana Z +
			DESÍTKY TISÍC: Definice platí jako přídavek rozměru pro opracování načisto
			0 = Přídavek rozměru pro opracování načisto UX a UZ odděleně 1 = Přídavek rozměru pro opracování načisto U rovnoběžně s konturou
			STOVKY TISÍC: Vyjíždění podél kontury
			0 = kompatibilita, automatické vyjíždění podél kontury 1 = s vyjížděním podél kontury 2 = bez vyjíždění podél kontury 3 = automatické vyjíždění podél kontury
			JEDNOTKY MILIONŮ: Podříznutí
			0 = místo se nevyhodnocuje při zapichování, obrábění zbytkového materiálu a soustružení zápichů, při obrábění zbytkového materiálu 1 = podříznutí opracovat 2 = podříznutí neopracovat
			DESÍTKY MILIONŮ: Před/za osou otáčení
			0 = obrábění před středem otáčení 1 = rezervováno

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
5	F	_F	Posuv pro obrábění nahrubo/načisto
	FZ		Posuv ve směru abscisy při soustružení zápichů
6	FR	_FR	Posuv pro zajíždění nástroje při obrábění podříznutí nahrubo
	FX		Posuv ve směru ordináty při soustružení zápichů
7	RP	_RP	Návratová rovina při vnitřním obrábění (abs, vždycky průměr)
8	D	_D	Přísuv při obrábění nahrubo, (viz JEDNOTKY parametru _AMODE)
9	DX	_DX	Přísuv ve směru X (viz JEDNOTKY parametru _AMODE)
10	DZ	_DZ	Přísuv ve směru Z (viz JEDNOTKY parametru _AMODE)
11	UX	_UX	Přídavek rozměru pro opracování načisto ve směru X (viz _VARI (DESÍTKY TISÍC))
12	UZ	_UZ	Přídavek rozměru pro opracování načisto ve směru Z (viz _VARI (DESÍTKY TISÍC))
13	U	_U	Přídavek rozměru pro opracování načisto rovnoběžně s konturou (viz _VARI (DESÍTKY TISÍC))
14	U1	_U1	Dodatečný přídavek rozměru pro opracování načisto (viz _AMODE (TISÍCE))
15	BL	_BL	Definice surového obrobku 1 = válec s přídavkem rozměru 2 = přídavek rozměru na kontuře hotového obrobku 3 = kontura surového obrobku je zadána
16	XD	_XD	Definice surového obrobku ve směru X (viz _AMODE (STOVKY TISÍC))
17	ZD	_ZD	Definice surového obrobku ve směru Z (viz _AMODE (MILIONY))
18	XA	_XA	Hranice 1 ve směru X (abs, vždycky průměr)
19	ZA	_ZA	Hranice 1 ve směru Z (abs)
20	XB	_XB	Hranice 2 ve směru X (viz _AMODE (DESÍTKY MILIONŮ))
21	ZB	_ZB	Hranice 2 ve směru Z (viz _AMODE (STOVKY MILIONŮ))
22	XDA	_XDA	Hranice zápichu 1 při zapichování na čelní straně (abs, vždy průměr)
23	XDB	_XDB	Hranice zápichu 2 při zapichování na čelní straně (abs, vždy průměr)
24	N	_N	Počet zápichů
25	DP	_DP	Vzdálenost mezi zápichy Podélný zápich: rovnoběžně s osou Z Příčný zápich: rovnoběžně s osou X
26	DI	_DI	Vzdálenost pro přerušení posuvu 0 = žádné přerušení 0 > s přerušením
27	SC	_SC	Bezpečnostní vzdálenost pro objíždění překážky, inkrementálně
28	D2	_DN	D-číslo pro 2. břit nástroje, pokud není naprogramováno ⇒ D+1

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
29		_GMODE	Geometrický režim (vyhodnocování naprogramované geometrické hodnoty)
			JEDNOTKY: rezervováno
			DESÍTKY: rezervováno
			STOVKY: Volba opracování/pouze výpočet počátečního bodu
			0 = normální opracování (není zapotřebí žádný režim kompatibility)
			1 = normální opracování
			2 = výpočet počáteční polohy - žádné obrábění (jen pro vyvolání ze systémů ShopMill/ ShopTurn)
			TISÍCE: Ohraničení
			0 = ne
			1 = ano
			DESÍTKY TISÍC: Zadat hranici 1 ve směru X
			0 = ne
			1 = ano
			STOVKY TISÍC: Zadat hranici 2 ve směru X
			0 = ne
			1 = ano
			JEDNOTKY MILIONŮ: Zadat hranici 1 ve směru Z
			0 = ne
			1 = ano
			DESÍTKY MILIONŮ: Zadat hranici 2 ve směru Z
			0 = ne
			1 = ano
30		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)
			DESÍTKY: technologický režim
			1 = oddělování třísky od kontury
			2 = obrábění zápichů na kontuře
			3 = soustružení zápichů
			STOVKY: Odstraňování zbytkového materiálu
			0 = ne
			1 = ano

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
31		_AMODE	Alternativní režim
			JEDNOTKY: Volba přísuvu
			0 = přísuv DX a DZ při druhu obrábění rovnoběžně s konturou
			1 = přísuv D
			DESÍTKY: Strategie pro přísuv
			0 = proměnná hloubka řezu (90 ... 100 %)
			1 = konstantní hloubka řezu
			STOVKY: Rozdělení řezů
			0 = rovnoměrně
			1 = optimalizace na hranách
			TISÍCE: Volba přídavku rozměru na kontuře U1, dvojí obrábění načisto
			0 = ne
			1 = ano
			DESÍTKY TISÍC: Aktualizace volby surového obrobku
			0 = ne
			1 = ano
			STOVKY TISÍC: Volba přídavku rozměru na surovém obrobku XD
			0 = absolutně, hodnota příčné osy se zadává jako průměr
			1 = inkrementálně, hodnota příčné osy se zadává jako rádius
			JEDNOTKY MILIONŮ: Volba přídavku rozměru na surovém obrobku ZD
			0 = absolutně
			1 = inkrementálně
			DESÍTKY MILIONŮ: Volba hranice 2 XB
			0 = absolutně, hodnota příčné osy se zadává jako průměr
			1 = inkrementálně, hodnota příčné osy se zadává jako rádius
			STOVKY MILIONŮ: Volba hranice 2 ZB
			0 = absolutně
			1 = inkrementálně

16.1.37 Naklápění - CYCLE800

Programování

```
CYCLE800(INT _FR, STRING[32] _TC, INT _ST, INT _MODE, REAL _X0, REAL
_Y0, REAL _Z0, REAL _A, REAL _B, REAL _C, REAL _X1, REAL _Y1, REAL
_Z1, INT _DIR, REAL _FR_I , INT _DMODE)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1		_FR	Režim volného vyjíždění:
			0 = žádné volné vyjíždění 1 = volné vyjíždění strojní osy Z 2 = volné vyjíždění strojní osy Z a potom XY 3 = rezervováno 4 = maximální volné vyjíždění ve směru nástroje 5 = inkrementální volné vyjíždění ve směru nástroje
2		_TC	Název datového bloku naklápění: "" (žádný název), pokud existuje jen 1 datový blok naklápění "0" deaktivování datového bloku naklápění (vymazání framu naklápění)
3		_ST	Status transformací
			JEDNOTKY:
			0 = nová, naklopená rovina se vymaže a znovu se vypočítá s aktuálními parametry 1 = aditivní, naklopená rovina se přičte k aktivní naklopené rovině
			DESÍTKY: Špička nástroje na jednom místě ano/ne (aktivní jen tehdy, je-li instalována funkce IBN SCHWENKEN)
			0 = špička nástroje nezůstává na jednom místě 1 = špička nástroje zůstává na jednom místě (TRAORI)
			STOVKY: Nastavování polohy/orientace (funkce se zobrazuje ve vstupní obrazovce NAKLÁPĚNÍ)
			0 = nástroj se nenastavuje 1 = nástroj se nastavuje (zejména fréza) 2 = nastavit orientaci soustružnického nástroje (pokud je ve funkci IBN SCHWENKEN instalována kinematika osy B pro technologii soustružení) 3 = nastavit orientaci frézovacího nástroje (pokud je ve funkci IBN SCHWENKEN instalována kinematika osy B pro technologii soustružení) 9 = rezervováno
			TISÍCE: Interní parametr naklápění v režimu JOG
			DESÍTKY TISÍC: Viz parametr pro směr _DIR
			0 = naklápění "ano" 1 = naklápění "ne", směr "mínus" [3)] 2 = naklápění "ne", směr "plus" [3)]
			STOVKY TISÍC: Viz parametr pro směr _DIR
			0 = kompatibilita 1 = volba směru "mínus" optimalizována[4)] 2 = volba směru "plus" optimalizována[4)]

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
4		_MODE 5)	Režim naklápění: Vyhodnocování úhlu naklápění a posloupnosti naklápění (bitové kódování!)
			Bit: 7 6
			0 0: úhel naklopení po osách → viz parametry _A, _B, _C
			0 1: prostorový úhel → viz parametry _A, _B 1)
			1 0: projekční úhel → viz parametry _A, _B, _C 1)
			1 1: režim naklápění: kruhové osy přímo → viz parametry _A, _B 1)
			Bit: 5 4 3 2 1 0 (v případě prostorového úhlu nemají žádný význam!)
			x x x x 0 1 1. otáčení _A okolo osy X
			x x x x 1 0 1. otáčení _A okolo osy Y
			x x x x 1 1 1. otáčení _A okolo osy Z
			x x 0 1 x x 2. otáčení _B okolo osy X
			x x 1 0 x x 2. otáčení _B okolo osy Y
			x x 1 1 x x 2. otáčení _B okolo osy Z
			0 1 x x x x 3. otáčení _C okolo osy X
			1 0 x x x x 3. otáčení _C okolo osy Y
			1 1 x x x x 3. otáčení _C okolo osy Z
5	X0	_X0	Souřadnice X vztažného bodu před otáčením
6	Y0	_Y0	Souřadnice Y vztažného bodu před otáčením
7	Z0	_Z0	Souřadnice Z vztažného bodu před otáčením
8	X(A)	_A	1. Otáčení podle nastavení v parametru _MODE
9	Y(B)	_B	2. Otáčení podle nastavení v parametru _MODE
10	Z(C)	_C	3. Otáčení podle nastavení v parametru _MODE
11	X1	_X1	Souřadnice X vztažného bodu po otáčení
12	Y1	_Y1	Souřadnice Y vztažného bodu po otáčení
13	Z1	_Z1	Souřadnice Z vztažného bodu po otáčení
14	- nebo +	_DIR	Spustit pohyb kruhových os pracovním posuvem (předdefinovaná hodnota = -1!):
			-1 = najíždět na menší hodnotu kruhové osy 1 nebo 2 2)
			+1 = najíždět na větší hodnotu kruhové osy 1 nebo 2 2)
			0 = žádné naklápění (jen vypočítat frame naklápění) 1) 3)
15	FR	_FR_I	Hodnota (ink): inkrementální volné vyjíždění ve směru nástroje
16		_DMODE	Režim zobrazování
			JEDNOTKY: Rovina obrábění G17/G18/G19
			0 = kompatibilita, zůstává aktivní rovina nastavená před voláním cyklu
			1 = G17 (aktivní jen v cyklu)
			2 = G18 (aktivní jen v cyklu)
			3 = G19 (aktivní jen v cyklu)

Poznámka

Pokud jsou následující předávané parametry naprogramovány nepřímo (jako parametry), zpětný překlad vstupní obrazovky nebude možný: _FR, _ST, _TC, _MODE, _DIR

1) Volba je možná, pokud je instalována funkce IBN SCHWENKEN.

2) Volba je možná, pokud je ve funkci IBN SCHWENKEN nastaven vztažný směr kruhové osy 1 nebo 2.

žádné pole pro volbu vztažného směru: není

3) Zobrazování volby naklápění "ne" může být blokováno (SD 55221 Bit 0).

Naklápění "ne" ve směnu "mínus" odpovídá _DIR = 0 a _ST DESÍTKY TISÍC = 1

Naklápění "ne" ve směnu "plus" odpovídá _DIR = 0 a _ST DESÍTKY TISÍC = 2

4) Volba směru kruhové osy 1 nebo 2 se uskutečňuje také tehdy, když se kruhová osa se vztažným směrem nachází v poloze pólu (hodnota polohy se rovná nule).

5 Příklad kódování: otáčení po osách, posloupnost otáčení ZYX

binárně: 00011011; decimálně: 27

Identifikátory os XYZ odpovídají geometrickým osám NC kanálu. Otáčení okolo os XYZ se smí uskutečňovat jednotlivě. Např. posloupnost otáčení ZXZ je ve volání cyklu CYCLE800 nepřípustná.

16.1.38 Vysokorychlostní obrábění - CYCLE832

Programování

```
CYCLE832(_TOL, _TOLM, _V832)
```

Poznámka

Cyklus CYCLE832 nezbavuje výrobce stroje povinnosti uskutečnit nezbytné optimalizační úkoly při uvádění stroje do provozu. To se týká optimalizace os podílejících se na obrábění a nastavení parametrů NCU (dopředná regulace, omezení ryvu atd.).

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení
1	TOL	_TOL	Tolerance
2		_TOLM	Technologie
			JEDNOTKY:
			0 = deaktivováno
			1 = obrábění načisto (Finish)
			2 = předběžné obrábění načisto (Semifinish)
			3 = obrábění nahrubo (Rough)
3		_V832	Verze cyklu CYCLE832
			JEDNOTKY:
			0 = do SW 7.5
			1 = od HMI sl SW 2.6

Poznámka

Při deaktivování cyklu CYCLE832 musí být parametr Tolerance předáván s nulovou hodnotou.

Příklad: `CYCLE832(0,0,1)`

Syntaxe `CYCLE832()` je pro deaktivování cyklu CYCLE832 přípustná také.

16.1.39 Vysokorychlostní obrábění (HSC) - CYCLE_HSC

Programování

```
CYCLE_HSC(_Mode, _TOL, _RTOL)
```

Parametr

Č.	Obrazovka parametrů	Interní parametr	Vysvětlení	
1		_MODE	Druh obrábění (technologie)	
			Parametr "Druh obrábění" se předává do cyklu CYCLE_HSC prostým textem jako řetězec (jsou přípustná malá a velká písmena).	
			"FINISH" =	Obrábění načisto
			"SEMIFINISH" =	Předběžné obrábění načisto
			"ROUGH" =	Obrábění nahrubo
			"OFF" nebo "DESELECT" =	Deaktivování cyklu CYCLE_HSC
2		_TOL	Tolerance	
			Parametry "Tolerance" odpovídá toleranci geometrických os.	
3		_RTOL	Tolerance kruhových os	
			Tolerance kruhových je předávána pomocí příkazu OTOL = _RTOL. Parametr OTOL se uplatňuje pouze při aktivní transformaci orientace (TRAORI).	

Poznámka

Cyklus CYCLE_HSC interně vyvolává cyklus CYCLE832 (High Speed Settings).

Poznámka

Při deaktivování cyklu CYCLE_HSC se nemusí předávat žádné další parametry.

Příklad: `CYCLE_HSC("OFF")`

Příklad

```
Programový kód                    Komentář
G710
TRAORI
CYCLE_HSC("FINISH",0.01,3)        ; Druh opracování FINISH s tolerancí geometrických
                                    os 0,01 mm a s tolerancí kruhových os 3 stupně.
```

Tabulky

17

17.1 Příkazy

Legenda:

1) Platnost příkazu:
   - m modální
   - s bloková

2) Odkaz na dokument, který obsahuje podrobný popis příkazu:
   - *PGs/* Příručka programování, Základy
   - *PGAs/* Příručka programování, Pro pokročilé
   - *BNMs/* Příručka programování, Měřicí cykly
   - *BHDs/* Návod k obsluze, Soustružení
   - *BHFs/* Návod k obsluze, Frézování
   - *FB1 ( )* Příručka Popis funkcí, Základní funkce (s alfanumerickou zkratkou popisu příslušné funkce v závorkách)
   - *FB2 ( )* Příručka Popis funkcí, Rozšiřovací funkce (s alfanumerickou zkratkou popisu příslušné funkce v závorkách)
   - *FB3 ( )* Příručka Popis funkcí, Speciální funkce (s alfanumerickou zkratkou popisu příslušné funkce v závorkách)
   - *FBSI/s/* Příručka Popis funkcí, Safety Integrated
   - *FBSY* Příručka Popis funkcí, Synchronní akce
   - *FBW* Příručka Popis funkcí, Správa nástrojů

3) Standardní nastavení na začátku programu (stav systému při dodávce řídícího systému, pokud není naprogramováno nic jiného).

Příkaz	Význam	W 1)	Popis viz 2)
:	Číslo hlavního NC-bloku, ukončení značky skoku, operátor zřetězení		*PGAs/* Matematické funkce [Strana 64]
*	Operátor pro násobení		*PGAs/* Matematické funkce [Strana 64]
+	Operátor pro sečítání		*PGAs/* Matematické funkce [Strana 64]
-	Operátor pro odečítání		*PGAs/* Matematické funkce [Strana 64]
<	Porovnávací operátor, je menší než		*PGAs/* Matematické funkce [Strana 64]
<<	Operátor zřetězení pro řetězce		*PGAs/* Matematické funkce [Strana 64]

Příkaz	Význam	W [1]	Popis viz [2]
<=	Porovnávací operátor, je menší nebo rovno než		*PGAs*/ Matematické funkce [Strana 64]
=	Operátor přiřazení		*PGAs*/ Matematické funkce [Strana 64]
>=	Porovnávací operátor, je větší nebo rovno než		*PGAs*/ Matematické funkce [Strana 64]
/	Operátor pro dělení		*PGAs*/ Matematické funkce [Strana 64]
/0 … … /7	Blok se bude přeskakovat (1. přeskakovaná úroveň) Blok se bude přeskakovat (8. přeskakovaná úroveň)		*PGs*/
A	Název osy	m/s	*PGAs*/ Programování orientace nástroje (A..., B..., C..., LEAD, TILT) [Strana 333]
A2	Orientace nástroje: Úhel RPY nebo Eulerův úhel	s	*PGAs*/ Programování orientace nástroje (A..., B..., C..., LEAD, TILT) [Strana 333]
A3	Orientace nástroje: Složky vektoru normály směru/plochy	s	*PGAs*/ Programování orientace nástroje (A..., B..., C..., LEAD, TILT) [Strana 333]
A4	Orientace nástroje: Normálový vektor plochy pro začátek bloku	s	*PGAs*/ Frézování na čelní ploše (3D frézování A4, B4, C4, A5, B5, C5) [Strana 340]
A5	Orientace nástroje: Normálový vektor plochy pro konec bloku	s	*PGAs*/ Frézování na čelní ploše (3D frézování A4, B4, C4, A5, B5, C5) [Strana 340]
ABS	Absolutní hodnota		*PGAs*/ Matematické funkce [Strana 64]
AC	Absolutní zadání rozměru pro souřadnice/polohy	s	*PGs*/
ACC	Ovlivňování aktuálního zrychlení osy	m	*PGs*/
ACCLIMA	Ovlivňování aktuálního maximálního zrychlení osy	m	*PGs*/
ACN	Zadání absolutního rozměru pro kruhovou osu, na pozici se najíždí v záporném směru	s	*PGs*/
ACOS	Arkus kosinus (trigonometrická funkce)		*PGAs*/ Matematické funkce [Strana 64]
ACP	Zadání absolutního rozměru pro kruhovou osu, na pozici se najíždí v kladném směru	s	*PGs*/

Příkaz	Význam	W [1]	Popis viz [2]
ACTBLOCNO	Výstup aktuální čísla alarmového bloku, a to i když je aktivní příkaz "Potlačit výpis aktuálního bloku" (DISPLOF)!		*PGAsl* Potlačení vypisování aktuálního bloku (DISPLOF, DISPLON, ACTBLOCNO) [Strana 177]
ADDFRAME	Započítání a případné aktivování změřeného framu		*PGAsl, FB1(K2)* Výpočet framu na základě 3 změřených bodů v prostoru (MEAFRAME) [Strana 308]
ADIS	Vzdálenost zaoblení pro funkce pohybu po dráze G1, G2, G3, …	m	*PGsl*
ADISPOS	Vzdálenost zaoblení pro rychlý posuv G0	m	*PGsl*
ADISPOSA	Velikost tolerančního okna pro IPOBRKA	m	*PGAsl* Programovatelné kritérium konce pohybu (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) [Strana 284]
ALF	Úhel rychlého zvedání	m	*PGAsl* Rychlé pozvednutí od kontury (SETINT LIFTFAST, ALF) [Strana 126]
AMIRROR	Programovatelné zrcadlové převrácení	s	*PGsl*
AND	Logické A		*PGAsl* Porovnávací a logické operace [Strana 67]
ANG	Úhel konturové křivky	s	*PGsl*
AP	Polární úhel	m/s	*PGsl*
APR	Načtení / výpis přístupové ochrany		*PGAsl* Atribut: Přístupová oprávnění (APR, APW, APRP, APWP, APRB, APWB) [Strana 41]
APRB	Načtení přístupové ochrany, BTSS		*PGAsl* Atribut: Přístupová oprávnění (APR, APW, APRP, APWP, APRB, APWB) [Strana 41]
APRP	Načtení přístupové ochrany, výrobní program		*PGAsl* Atribut: Přístupová oprávnění (APR, APW, APRP, APWP, APRB, APWB) [Strana 41]
APW	Zápis přístupové ochrany		*PGAsl* Atribut: Přístupová oprávnění (APR, APW, APRP, APWP, APRB, APWB) [Strana 41]
APWB	Zápis přístupové ochrany, BTSS		*PGAsl* Atribut: Přístupová oprávnění (APR, APW, APRP, APWP, APRB, APWB) [Strana 41]
APWP	Zápis přístupové ochrany, výrobní program		*PGAsl* Atribut: Přístupová oprávnění (APR, APW, APRP, APWP, APRB, APWB) [Strana 41]

Příkaz	Význam	W [1]	Popis viz [2]
APX	Definice ochrany proti přístupu pro provedení uvedeného prvku jazyka		*PGAs*/ Opětovná definice systémových proměnných, uživatelských proměnných a příkazů NC jazyka (REDEF) [Strana 31]
AR	Úhel kruhové výseče	m/s	*PGs*/
AROT	Programovatelné otočení	s	*PGs*/
AROTS	Programovatelné otáčení framu o prostorový úhel	s	*PGs*/
AS	Definice makra		*PGAs*/ Technika maker (DEFINE ... AS) [Strana 213]
ASCALE	Programovatelná změna měřítka	s	*PGs*/
ASIN	Matematická funkce, arkus sinus		*PGAs*/ Matematické funkce [Strana 64]
ASPLINE	Akimovy spliny	m	*PGAs*/ Splinová interpolace (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [Strana 244]
ATAN2	Arkus tangens2		*PGAs*/ Matematické funkce [Strana 64]
ATOL	Tolerance pro specifickou osu pro funkce kompresoru, vyhlazení orientace a druhy zaoblování konturových přechodů		*PGAs*/ Programovatelná tolerance kontury/orientace (CTOL, OTOL, ATOL) [Strana 498]
ATRANS	Aditivní programovatelné posunutí	s	*PGs*/
AX	Proměnný identifikátor osy	m/s	*PGAs*/ Osové funkce (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) [Strana 677]
AXCTSWE	Otočení osového zásobníku		*PGAs*/ Osový zásobník (AXCTSWE, AXCTSWED, AXCTSWEC) [Strana 685]
AXCTSWEC	Uvolnění pro reset otočení osového zásobníku		*PGAs*/ Osový zásobník (AXCTSWE, AXCTSWED, AXCTSWEC) [Strana 685]
AXCTSWED	Otočení osového zásobníku (varianta příkazu pro uvádění do provozu!)		*PGAs*/ Osový zásobník (AXCTSWE, AXCTSWED, AXCTSWEC) [Strana 685]
AXIS	Identifikátor osy, adresa osy		*PGAs*/ Definice uživatelských proměnných (DEF) [Strana 25]
AXNAME	Konvertovaný vstupní řetězec v identifikátoru osy		*PGAs*/ Osové funkce (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) [Strana 677]

Příkaz	Význam	W [1]	Popis viz [2]
AXSTRING	Konvertovaný řetězec čísla vřetena		*PGAsl* Osové funkce (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) [Strana 677]
AXTOCHAN	Vyžádání osy pro určitý kanál Je možné z NC programu a ze synchronních akcí.		*PGAsl* Předávání osy jinému kanálu (AXTOCHAN) [Strana 137]
AXTOSPI	Konvertovaný identifikátor osy v indexu vřetena		*PGAsl* Osové funkce (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) [Strana 677]
B	Název osy	m/s	*PGAsl* Programování orientace nástroje (A..., B..., C..., LEAD, TILT) [Strana 333]
B2	Orientace nástroje: Úhel RPY nebo Eulerův úhel	s	*PGAsl* Programování orientace nástroje (A..., B..., C..., LEAD, TILT) [Strana 333]
B3	Orientace nástroje: Složky vektoru normály směru/plochy	s	*PGAsl* Programování orientace nástroje (A..., B..., C..., LEAD, TILT) [Strana 333]
B4	Orientace nástroje: Normálový vektor plochy pro začátek bloku	s	*PGAsl* Frézování na čelní ploše (3D frézování A4, B4, C4, A5, B5, C5) [Strana 340]
B5	Orientace nástroje: Normálový vektor plochy pro konec bloku	s	*PGAsl* Frézování na čelní ploše (3D frézování A4, B4, C4, A5, B5, C5) [Strana 340]
B_AND	Bitové logické A		*PGAsl* Porovnávací a logické operace [Strana 67]
B_OR	Bitové logické NEBO		*PGAsl* Porovnávací a logické operace [Strana 67]
B_NOT	Bitová negace		*PGAsl* Porovnávací a logické operace [Strana 67]
B_XOR	Bitové logické XOR		*PGAsl* Porovnávací a logické operace [Strana 67]
BAUTO	Definice prvního splinového úseku prostřednictvím následujících tří bodů	m	*PGAsl* Splinová interpolace (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [Strana 244]
BLOCK	Spolu s klíčovým slovem TO definuje úsek programu v nepřímo volaném podprogramu, který se má zpracovat.		*PGAsl* Nepřímé volání podprogramu se zadáním úseku programu, který á být zpracován (CALL BLOCK ... TO ...) [Strana 199]
BLSYNC	Zpracování rutiny přerušení má být zahájeno teprve s následující změnou bloku.		*PGAsl* Přiřazování a spouštění rutin přerušení (SETINT, PRIO, BLSYNC) [Strana 122]

Příkaz	Význam	W [1]	Popis viz [2]
BNAT [3]	Přirozený přechod na první splinový blok	m	*/PGAs/* Splinová interpolace (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [Strana 244]
BOOL	Datový typ: Hodnoty TRUE/FALSE, příp. 1/0		*/PGAs/* Definice uživatelských proměnných (DEF) [Strana 25]
BOUND	Zkontroluje, zda hodnota leží v rámci definovaného rozsahu hodnot. V případě rovnosti se vrací zpět zkoušená hodnota.		*/PGAs/* Minimum, maximum a rozsah proměnných (MINVAL, MAXVAL, BOUND) [Strana 71]
BRISK [3]	Skokové změny zrychlení po dráze	m	*/PGs/*
BRISKA	Aktivování skokových změn zrychlení při pohybu po dráze pro naprogramované osy		*/PGs/*
BSPLINE	B-Spline	m	*/PGAs/* Splinová interpolace (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [Strana 244]
BTAN	Tangenciální přechod na první splinový blok	m	*/PGAs/* Splinová interpolace (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [Strana 244]
C	Název osy	m/s	*/PGAs/* Programování orientace nástroje (A..., B..., C..., LEAD, TILT) [Strana 333]
C2	Orientace nástroje: Úhel RPY nebo Eulerův úhel	s	*/PGAs/* Programování orientace nástroje (A..., B..., C..., LEAD, TILT) [Strana 333]
C3	Orientace nástroje: Složky vektoru normály směru/plochy	s	*/PGAs/* Programování orientace nástroje (A..., B..., C..., LEAD, TILT) [Strana 333]
C4	Orientace nástroje: Normálový vektor plochy pro začátek bloku	s	*/PGAs/* Frézování na čelní ploše (3D frézování A4, B4, C4, A5, B5, C5) [Strana 340]
C5	Orientace nástroje: Normálový vektor plochy pro konec bloku	s	*/PGAs/* Frézování na čelní ploše (3D frézování A4, B4, C4, A5, B5, C5) [Strana 340]
CAC	Absolutní najíždění na pozici		*/PGAs/* Najíždění na kódované pozice (CAC, CIC, CDC, CACP, CACN) [Strana 243]
CACN	Na hodnotu uloženou v tabulce se bude najíždět v záporném směru a hodnota bude považována za absolutní.		*/PGAs/* Najíždění na kódované pozice (CAC, CIC, CDC, CACP, CACN) [Strana 243]
CACP	Na hodnotu uloženou v tabulce se bude najíždět v kladném směru a hodnota bude považována za absolutní.		*/PGAs/* Najíždění na kódované pozice (CAC, CIC, CDC, CACP, CACN) [Strana 243]

Příkaz	Význam	W [1]	Popis viz [2]
CALCDAT	Tato funkce vypočítá rádius a střed kruhu ne základě 3 nebo 4 bodů.		*PGAsl* Výpočet parametrů kruhu (CALCDAT) [Strana 737]
CALCPOSI	Kontrola narušení chráněné oblasti, ohraničení pracovního pole a softwarové mezní hodnoty		*PGAsl* Kontrola narušení chráněné oblasti, ohraničení pracovního pole a softwarových koncových spínačů (CALCPOSI) [Strana 235]
CALL	Nepřímé volání podprogramu		*PGAsl* Nepřímé volání podprogramu (CALL) [Strana 198]
CALLPATH	Programovatelné vyhledávání cesty při volání podprogramu		*PGAsl* Rozšíření cesty pro vyhledávání při volání podprogramu (CALLPATH) [Strana 203]
CANCEL	Přerušení modální synchronní akce		*PGAsl* Zrušení synchronní akce (CANCEL) [Strana 644]
CASE	Podmíněné větvení programu		*PGAsl* Větvení programu (CASE ... OF ... DEFAULT ...) [Strana 97]
CDC	Přímé najíždění na pozici		*PGAsl* Najíždění na kódované pozice (CAC, CIC, CDC, CACP, CACN) [Strana 243]
CDOF [3]	Vypnutí monitorování kolizí	m	*PGsl*
CDOF2	Vypnutí monitorování kolizí, při 3D obvodovém frézování	m	*PGsl*
CDON	Zapnutí monitorování kolizí	m	*PGsl*
CFC [3]	Konstantní posuv na kontuře	m	*PGsl*
CFIN	Konstantní posuv, jen na vnitřních zakřiveních, ne na vnějších zakřiveních	m	*PGsl*
CFINE	Přiřazení jemného posunutí proměnné typu FRAME		*PGAsl* Hrubé a jemné posunutí (CFINE, CTRANS) [Strana 303]
CFTCP	Konstantní posuv na vztažném bodu břitu nástroje, dráha středu nástroje	m	*PGsl*
CHAN	Specifikace rozsahu platnosti dat		*PGAsl* Definice uživatelských proměnných (DEF) [Strana 25]
CHANDATA	Nastavení čísla kanálu pro přístup ke kanálovým datům		*PGAsl* Pracovní paměť (CHANDATA, COMPLETE, INITIAL) [Strana 222]
CHAR	Datový typ: Znaky ASCII		*PGAsl* Definice uživatelských proměnných (DEF) [Strana 25]
CHECKSUM	Vytvoří kontrolní součet ve formátu STRING pro dané pole s pevně danou délkou		*PGAsl* Výpočet kontrolního součtu pole (CHECKSUM) [Strana 156]

Příkaz	Význam	W [1]	Popis viz [2]
CHF	Faseta; hodnota = délka fasety	s	*PGs/*
CHKDM	Zkouška jednoznačnosti v rámci zásobníku		*FBW*
CHKDNO	Kontrola jednoznačnosti D-čísel		*PGAs/* Volné zadávání D-čísel: Kontrola D-čísla (CHKDNO) [Strana 439]
CHR	Faseta; Hodnota = délka fasety ve směru pohybu		*PGs/*
CIC	Inkrementální najíždění na pozici		*PGAs/* Najíždění na kódované pozice (CAC, CIC, CDC, CACP, CACN) [Strana 243]
CIP	Kruhová interpolace přes vnitřní bod	m	*PGs/*
CLEARM	Vymazání jedné/více značek pro přiřazení kanálu		*PGAs/* Koordinace programů (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) [Strana 115]
CLRINT	Deaktivování přerušení		*PGAs/* Vymazání přiřazení rutiny přerušení (CLRINT) [Strana 125]
CMIRROR	Zrcadlové převrácení souřadné osy.		*PGAs/* Matematické funkce [Strana 64]
COARSEA	Konec pohybu při dosažení hrubého okna přesného najetí	m	*PGAs/* Programovatelné kritérium konce pohybu (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) [Strana 284]
COMPCAD	Aktivování kompresoru: Optimalizovaná jakost povrchu u programů typu CAD.	m	*PGAs/* Komprese NC-bloků (COMPON, COMPCURV, COMPCAD, COMPOF) [Strana 258]
COMPCURV	Aktivování kompresoru: polynomy s konst. zakřivením	m	*PGAs/* Komprese NC-bloků (COMPON, COMPCURV, COMPCAD, COMPOF) [Strana 258]
COMPLETE	Řídící příkaz pro odesílání a příjem dat		PGAsl Pracovní paměť (CHANDATA, COMPLETE, INITIAL) [Strana 222]
COMPOF [3]	Deaktivování kompresoru	m	*PGAs/* Komprese NC-bloků (COMPON, COMPCURV, COMPCAD, COMPOF) [Strana 258]
COMPON	Aktivování kompresoru		*PGAs/* Komprese NC-bloků (COMPON, COMPCURV, COMPCAD, COMPOF) [Strana 258]
CONTDCON	Aktivování dekódování kontury v tabulkové formě		*PGAs/* Sestavování kódované tabulky kontury (CONTPRON) [Strana 730]

Příkaz	Význam	W [1]	Popis viz [2]
CONTPRON	Aktivování referenční přípravy		*PGAsl* Sestavování kontury (CONTPRON) [Strana 724]
CORROF	Všechny aktivní superponované pohyby jsou deaktivovány.		*PGsl*
COS	Kosinus (trigonometrická funkce)		*PGAsl* Matematické funkce [Strana 64]
COUPDEF	Definice vazby ELG / vazby synchron. vřeten		*PGAsl* Synchronní vřeteno: Programové příkazy (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) [Strana 542]
COUPDEL	Zrušení vazby ELG		*PGAsl* Synchronní vřeteno: Programové příkazy (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) [Strana 542]
COUPOF	Aktivování vazby ELG / dvojice synchron. vřeten		*PGAsl* Synchronní vřeteno: Programové příkazy (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) [Strana 542]
COUPOFS	Vypnutí vazby ELG / dvojice synchronních vřeten se zastavením vlečného vřetena		*PGAsl* Synchronní vřeteno: Programové příkazy (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) [Strana 542]
COUPON	Aktivování vazby ELG / dvojice synchron. vřeten		*PGAsl* Synchronní vřeteno: Programové příkazy (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) [Strana 542]
COUPONC	Zapnutí vazby ELG / dvojice synchronních vřeten, převzít předešlé naprogramování		*PGAsl* Synchronní vřeteno: Programové příkazy (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) [Strana 542]
COUPRES	Reset vazby ELG		*PGAsl* Synchronní vřeteno: Programové příkazy (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) [Strana 542]
CP	Pohyb po dráze	m	*PGAsl* Posuv PTP v kartézských souřadnicích [Strana 386]
CPRECOF [3]	Vypnutí programovatelné přesnosti kontury	m	*PGsl*
CPRECON	Zapnutí programovatelné přesnosti kontury	m	*PGsl*
CPROT	Zapnutí/vypnutí chráněné oblasti pro specifický kanál		*PGAsl* Aktivování/deaktivování chráněné oblasti (CPROT, NPROT) [Strana 231]
CPROTDEF	Definice chráněné oblasti pro specifický kanál		*PGAsl* Stanovení chráněné oblasti (CPROTDEF, NPROTDEF) [Strana 227]

Příkaz	Význam	W [1]	Popis viz [2]
CR	Rádius kruhu	s	*PGs/*
CROT	Otočení aktuálního souřadného systému		*PGAs/* Matematické funkce [Strana 64]
CROTS	Programovatelné otočení framu o prostorový úhel (otáčení v uvedených osách)	s	*PGs/*
CRPL	Otočení framu v libovolné rovině		*FB1(K2)*
CSCALE	Faktor změny měřítka pro více os		*PGAs/* Matematické funkce [Strana 64]
CSPLINE	Kubické spliny	m	*PGAs/* Splinová interpolace (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [Strana 244]
CT	Kruh s tangenciálním přechodem	m	*PGs/*
CTAB	Zjistit polohu vlečné osy z tabulky křivek na základě polohy řídící osy		*PGAs/* Čtení hodnot z tabulky křivky (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [Strana 521]
CTABDEF	Zapnutí definice tabulek		*PGAs/* Definice tabulek křivek (CTABDEF, CATBEND) [Strana 510]
CTABDEL	Vymazání tabulky křivek		*PGAs/* Mazání tabulek křivek (CTABDEL) [Strana 517]
CTABEND	Vypnutí definice tabulek		*PGAs/* Definice tabulek křivek (CTABDEF, CATBEND) [Strana 510]
CTABEXISTS	Kontrola tabulky křivek s číslem n		*PGAs/* Kontrola existence tabulky křivky (CTABEXISTS) [Strana 516]
CTABFNO	Počet ještě možných tabulek křivek v paměti		*PGAs/* Tabulky křivek: Kontrola využití systémových zdrojů (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [Strana 526]
CTABFPOL	Počet ještě možných polynomů v paměti		*PGAs/* Tabulky křivek: Kontrola využití systémových zdrojů (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [Strana 526]
CTABFSEG	Počet ještě možných křivkových segmentů v paměti		*PGAs/* Tabulky křivek: Kontrola využití systémových zdrojů (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [Strana 526]

Příkaz	Význam	W [1]	Popis viz [2]
CTABID	Zjištění čísla tabulky n-té tabulky křivek		*PGAs/* Tabulky křivek: Zjišťování vlastností tabulek (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD) [Strana 519]
CTABINV	Zjistit polohu řídící osy z tabulky křivek na základě polohy vlečné osy		*PGAs/* Čtení hodnot z tabulky křivky (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [Strana 521]
CTABISLOCK	Deaktivování zablokování tabulky křivek s číslem n		*PGAs/* Tabulky křivek: Zjišťování vlastností tabulek (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD) [Strana 519]
CTABLOCK	Zablokování mazání a přepisování		*PGAs/* Blokování tabulek křivek proti mazání a přepisování (CTABLOCK, CTABUNLOCK) [Strana 518]
CTABMEMTYP	Zjištění paměti, ve které je uložena tabulka křivek s číslem n.		*PGAs/* Tabulky křivek: Zjišťování vlastností tabulek (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD) [Strana 519]
CTABMPOL	Maximální možný počet polynomů v paměti		*PGAs/* Tabulky křivek: Kontrola využití systémových zdrojů (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [Strana 526]
CTABMSEG	Maximální možný počet křivkových segmentů v paměti		*PGAs/* Tabulky křivek: Kontrola využití systémových zdrojů (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [Strana 526]
CTABNO	Počet definovaných křivkových tabulek v paměti SRAM nebo DRAM		*FB3(M3)*
CTABNOMEM	Počet definovaných křivkových tabulek v paměti SRAM nebo DRAM		*PGAs/* Tabulky křivek: Kontrola využití systémových zdrojů (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [Strana 526]
CTABPERIOD	Zjištění periodicity tabulky křivek s číslem n		*PGAs/* Tabulky křivek: Zjišťování vlastností tabulek (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD) [Strana 519]
CTABPOL	Počet již používaných polynomů v paměti		*PGAs/* Tabulky křivek: Kontrola využití systémových zdrojů (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [Strana 526]

Příkaz	Význam	W [1]	Popis viz [2]
CTABPOLID	Počet křivkových polynomů používaných tabulkou křivek s číslem n		*PGAs*/ Tabulky křivek: Kontrola využití systémových zdrojů (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [Strana 526]
CTABSEG	Počet již používaných křivkových segmentů v paměti		*PGAs*/ Tabulky křivek: Kontrola využití systémových zdrojů (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [Strana 526]
CTABSEGID	Počet křivkových segmentů používaných tabulkou křivek s číslem n		*PGAs*/ Tabulky křivek: Kontrola využití systémových zdrojů (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [Strana 526]
CTABSEV	Zjištění koncové hodnoty vlečné osy segmentu křivkové tabulky		*PGAs*/ Čtení hodnot z tabulky křivky (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [Strana 521]
CTABSSV	Zjištění počáteční hodnoty vlečné osy segmentu křivkové tabulky		*PGAs*/ Čtení hodnot z tabulky křivky (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [Strana 521]
CTABTEP	Zjištění hodnoty řídící osy na konci křivkové tabulky		*PGAs*/ Čtení hodnot z tabulky křivky (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [Strana 521]
CTABTEV	Zjištění hodnoty vlečné osy na konci křivkové tabulky		*PGAs*/ Čtení hodnot z tabulky křivky (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [Strana 521]
CTABTMAX	Zjištění maximální hodnoty vlečné osy křivkové tabulky		*PGAs*/ Čtení hodnot z tabulky křivky (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [Strana 521]
CTABTMIN	Zjištění minimální hodnoty vlečné osy křivkové tabulky		*PGAs*/ Čtení hodnot z tabulky křivky (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [Strana 521]
CTABTSP	Zjištění hodnoty řídící osy na začátku křivkové tabulky		*PGAs*/ Čtení hodnot z tabulky křivky (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [Strana 521]
CTABTSV	Zjištění hodnoty vlečné osy na začátku křivkové tabulky		*PGAs*/ Čtení hodnot z tabulky křivky (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [Strana 521]

Příkaz	Význam	W [1]	Popis viz [2]
CTABUNLOCK	Odstranění blokování mazání a přepisování		*PGAs/* Blokování tabulek křivek proti mazání a přepisování (CTABLOCK, CTABUNLOCK) [Strana 518]
CTOL	Tolerance kontury pro funkce kompresoru, vyhlazení orientace a druhy zaoblování konturových přechodů		*PGAs/* Programovatelná tolerance kontury/orientace (CTOL, OTOL, ATOL) [Strana 498]
CTRANS	Posunutí počátku pro více os.		*PGAs/* Hrubé a jemné posunutí (CFINE, CTRANS) [Strana 303]
CUT2D [3]	2D korekce nástroje	m	*PGs/*
CUT2DF	2D korekce nástroje. Korekce nástroje se uplatňuje vzhledem k aktuálnímu framu (šikmá rovina).	m	*PGs/*
CUT3DC	3D korekce nástroje, obvodové frézování	m	*PGAs/* Aktivování 3D korekcí nástroje (CUT3DC..., CUT3DF...) [Strana 419]
CUT3DCC	3D korekce nástroje, obvodové frézování s omezujícími plochami	m	*PGAs/* 3D korekce nástroje: Zohlednění hraniční plochy (CUT3DCC, CUT3DCCD) [Strana 429]
CUT3DCCD	3D korekce nástroje, obvodové frézování s omezujícími plochami s diferenčním nástrojem	m	*PGAs/* 3D korekce nástroje: Zohlednění hraniční plochy (CUT3DCC, CUT3DCCD) [Strana 429]
CUT3DF	3D korekce nástroje, frézování na čelní ploše	m	*PGAs/* Aktivování 3D korekcí nástroje (CUT3DC..., CUT3DF...) [Strana 419]
CUT3DFF	3D korekce nástroje, čelní frézování s konstantní orientací nástroje v závislosti na aktivním framu	m	*PGAs/* Aktivování 3D korekcí nástroje (CUT3DC..., CUT3DF...) [Strana 419]
CUT3DFS	3D korekce nástroje, čelní frézování s konstantní orientací nástroje nezávisle na aktivním framu	m	*PGAs/* Aktivování 3D korekcí nástroje (CUT3DC..., CUT3DF...) [Strana 419]
CUTCONOF [3]	Deaktivování konstantní korekce rádiusu	m	*PGs/*
CUTCONON	Aktivování konstantní korekce rádiusu	m	*PGs/*
CUTMOD	Aktivování funkce "Modifikace korekčních parametrů u točivých nástrojů"		*PGAs/* Editace parametrů břitu u otočných nástrojů (CUTMOD) [Strana 455]
CYCLE60	Technologický cyklus: Gravírovací cyklus		*PGAs/* Cyklus pro gravírování - CYCLE60 [Strana 785]
CYCLE61	Technologický cyklus: Rovinné frézování		*PGAs/* Rovinné frézování - CYCLE61 [Strana 761]
CYCLE62	Technologický cyklus: Volání kontury		*PGAs/* Volání kontury - CYCLE62 [Strana 788]

Příkaz	Význam	W [1]	Popis viz [2]
CYCLE63	Technologický cyklus: Frézování konturové kapsy		*PGAs*/ Frézování konturové kapsy - CYCLE63 [Strana 794]
CYCLE64	Technologický cyklus: Předvrtání konturové kapsy		*PGAs*/ Předvrtání konturové kapsy - CYCLE64 [Strana 792]
CYCLE70	Technologický cyklus: Frézování závitu		*PGAs*/ Frézování závitu - CYCLE70 [Strana 783]
CYCLE72	Technologický cyklus: Frézování po dráze		*PGAs*/ Frézování po dráze - CYCLE72 [Strana 789]
CYCLE76	Technologický cyklus: Frézování pravoúhlého čepu		*PGAs*/ Frézování pravoúhlého čepu - CYCLE76 [Strana 768]
CYCLE77	Technologický cyklus: Frézování kruhového čepu		*PGAs*/ Frézování kruhového čepu - CYCLE77 [Strana 770]
CYCLE78	Technologický cyklus: Vrtání a frézování závitu		*PGAs*/ Frézování vrtaných závitů - CYCLE78 [Strana 754]
CYCLE79	Technologický cyklus: Mnohohran		*PGAs*/ Mnohohran - CYCLE79 [Strana 772]
CYCLE81	Technologický cyklus: Vrtání, navrtávání středicích důlků		*PGAs*/ Vrtání, navrtávání středicích důlků - CYCLE81 [Strana 743]
CYCLE82	Technologický cyklus: Vrtání, čelní zahlubování		*PGAs*/ Vrtání, čelní zahlubování - CYCLE82 [Strana 744]
CYCLE83	Technologický cyklus: Vrtání hlubokých děr		*PGAs*/ Vrtání hlubokých děr - CYCLE83 [Strana 746]
CYCLE84	Technologický cyklus: Vrtání závitů bez vyrovnávací hlavičky		*PGAs*/ Vrtání závitů bez vyrovnávací hlavičky - CYCLE84 [Strana 749]
CYCLE85	Technologický cyklus: Vystružování		*PGAs*/ Vystružování - CYCLE85 [Strana 745]
CYCLE86	Technologický cyklus: Vyvrtávání		*PGAs*/ Vyvrtávání - CYCLE86 [Strana 748]
CYCLE92	Technologický cyklus: Upichování		*PGAs*/ Upichování - CYCLE92 [Strana 811]
CYCLE98	Technologický cyklus: Řetězec závitů		*PGAs*/ Řetězec závitů - CYCLE98 [Strana 808]
CYCLE99	Technologický cyklus: Soustružení závitů		*PGAs*/ Soustružení závitu - CYCLE99 [Strana 805]
CYCLE800	Technologický cyklus: Naklápění		*PGAs*/ Naklápění - CYCLE800 [Strana 817]
CYCLE801	Technologický cyklus: Mřížka nebo obdélník		*PGAs*/ Mřížka nebo obdélník - CYCLE801 [Strana 759]
CYCLE802	Technologický cyklus: Libovolné polohy		*PGAs*/ Libovolné pozice - CYCLE802 [Strana 756]

Příkaz	Význam	W [1]	Popis viz [2]
CYCLE832	Technologický cyklus: Parametry pro vysokorychlostní obrábění		*PGAsl* Vysokorychlostní obrábění - CYCLE832 [Strana 820]
CYCLE840	Technologický cyklus: Vrtání závitů s vyrovnávací hlavičkou		*PGAsl* Vrtání závitů s vyrovnávací hlavičkou - CYCLE840 [Strana 752]
CYCLE899	Technologický cyklus: Frézování otevřené drážky		*PGAsl* Frézování otevřené drážky - CYCLE899 [Strana 779]
CYCLE930	Technologický cyklus: Zápich		*PGAsl* Zápich - CYCLE930 [Strana 799]
CYCLE940	Technologický cyklus: Odlehčovací zápich určitého tvaru		*PGAsl* Tvary odlehčovacího zápichu - CYCLE940 [Strana 802]
CYCLE951	Technologický cyklus: Oddělování třísky		*PGAsl* Oddělování třísky - CYCLE951 [Strana 796]
CYCLE952	Technologický cyklus: Zápichy na kontuře		*PGAsl* Zápichy na kontuře - CYCLE952 [Strana 813]
CYCLE_HSC	Technologický cyklus: Vysokorychlostní obrábění		*PGAsl* Vysokorychlostní obrábění (HSC) - CYCLE_HSC [Strana 821]

Příkaz	Význam	W [1]	Popis viz [2]
D	Číslo korekčních parametrů nástroje		*PGsl*
D0	Je-li použito D0, jsou korekční parametry daného nástroje deaktivovány.		*PGsl*
DAC	Absolutní blokové programování průměrů pro specifickou osu	s	*PGsl*
DC	Údaj absolutního rozměru pro kruhové osy, na pozici se najíždí přímo	s	*PGsl*
DEF	Definice proměnných		*PGAsl* Definice uživatelských proměnných (DEF) [Strana 25]
DEFINE	Klíčové slovo pro definice maker		*PGAsl* Technika maker (DEFINE ... AS) [Strana 213]
DEFAULT	Větev v příkazu větvení CASE		*PGAsl* Větvení programu (CASE ... OF ... DEFAULT ...) [Strana 97]
DELAYFSTON	Definice začátku oblasti zastavení-zpoždění	m	*PGAsl* Větvení programu (CASE ... OF ... DEFAULT ...) [Strana 97]
DELAYFSTOF	Definice konce oblasti zastavení-zpoždění	m	*PGAsl* Úseky programu s podmíněným přerušením (DELAYFSTON, DELAYFSTOF) [Strana 476]

Příkaz	Význam	W [1]	Popis viz [2]
DELDL	Vymazání aditivních korekcí		*PGAs*/ Vymazání aditivních korekcí (DELDL) [Strana 405]
DELDTG	Vymazání zbytkové dráhy		*PGAs*/ Vymazání zbytkové dráhy (DELDTG) [Strana 587]
DELETE	Vymazání zadaného souboru. Název souboru může být zadán pomocí cesty a identifikace souboru.		*PGAs*/ Vymazání souboru (DELETE) [Strana 146]
DELTOOLENV	Vymazání datových bloků pro popis nástrojového prostředí		*FB1(W1)*
DIACYCOFA	Modální programování průměrů pro specifickou osu: V cyklech deaktivováno	m	*FB1(P1)*
DIAM90	Programování průměrů pro G90, programování rádiusů pro G91	m	*PGAs*/
DIAM90A	Modální programování průměrů pro G90 a AC, programování rádiusů pro G91 a IC pro specifickou osu	m	*PGs*/
DIAMCHAN	Převzetí všech os ze strojních parametrů týkajících se funkcí os v kanálovém stavu programování průměrů		*PGs*/
DIAMCHANA	Převzetí programování průměrů ze specifického kanálu		*PGs*/
DIAMCYCOF	Programování průměrů pro specifický kanál V cyklech deaktivováno	m	*FB1(P1)*
DIAMOF [3]	Vypnutí programování průměrů vypnuto Základní nastavení viz údaje od výrobce stroje	m	*PGs*/
DIAMOFA	Vypnutí modálního programování průměrů pro specifickou osu Základní nastavení viz údaje od výrobce stroje	m	*PGs*/
DIAMON	Vypnutí programování průměrů Zapnuto	m	*PGs*/
DIAMONA	Zapnutí modálního programování průměrů pro specifickou osu Odblokování viz údaje od výrobce stroje	m	*PGs*/
DIC	Relativní blokové programování průměrů pro specifickou osu	s	*PGs*/
DILF	Návratová dráha (délka)	m	*PGs*/
DISABLE	Vypnutí přerušení		*PGAs*/ Deaktivování/opětovné aktivování přiřazení rutiny přerušení (DISABLE, ENABLE) [Strana 124]
DISC	Převýšení přechodové kružnice – korekce rádiusu nástroje	m	*PGs*/
DISCL	Vzdálenost koncového bodu rychlého přísuvného pohybu od roviny obrábění		*PGs*/

Příkaz	Význam	W [1]	Popis viz [2]
DISPLOF	Potlačení vypisování aktuálního bloku		*PGAsl* Potlačení vypisování aktuálního bloku (DISPLOF, DISPLON, ACTBLOCNO) [Strana 177]
DISPLON	Odstranění potlačení vypisování aktuálního bloku		*PGAsl* Potlačení vypisování aktuálního bloku (DISPLOF, DISPLON, ACTBLOCNO) [Strana 177]
DISPR	Rozdíl dráhy pro zpětné polohování	s	*PGAsl* Opětovné najíždění na konturu (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [Strana 484]
DISR	Vzdálenost pro zpětné polohování	s	*PGAsl* Opětovné najíždění na konturu (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [Strana 484]
DITE	Výběr závitu	m	*PGsl*
DITS	Náběh závitu	m	*PGsl*
DIV	Celočíselné dělení		*PGAsl* Matematické funkce [Strana 64]
DL	Aktivování lokálně závislé aditivní korekce nástroje (DL, součtová a seřizovací korekce)	m	*PGAsl* Aktivování aditivních korekcí (DL) [Strana 402]
DO	Klíčové slovo pro synchronní akci, ke spuštění dojde, když je podmínka splněna.		*PGAsl* Akce (DO) [Strana 563]
DRFOF	Vypnutí posunutí ručním kolečkem (DRF)	m	*PGsl*
DRIVE	Zrychlení po dráze závislé na rychlosti	m	*PGsl*
DRIVEA	Zapnutí lomené charakteristiky zrychlení pro naprogramované osy		*PGsl*
DYNFINISH	Dynamika pro jemné obrábění načisto	m	*PGsl*
DYNNORM	Normální dynamika	m	*PGsl*
DYNPOS	Dynamika pro režim polohování, vrtání závitů	m	*PGsl*
DYNROUGH	Dynamika pro obrábění nahrubo	m	*PGsl*
DYNSEMIFIN	Dynamika pro obrábění načisto	m	*PGsl*
DZERO	Označení všech D-čísel dané jednotky TO jako neplatné		*PGAsl* Volné zadávání D-čísel: Nastavení D-čísla jako neplatného (DZERO) [Strana 442]

Příkaz	Význam	W [1]	Popis viz [2]
EAUTO	Definice posledního splinového úseku prostřednictvím posledních 3 bodů	m	*PGAs/* Splinová interpolace (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [Strana 244]
EGDEF	Definice elektronické převodovky		*PGAs/* Definice elektronické převodovky (EGDEF) [Strana 534]
EGDEL	Vymazání definice vazby pro vlečnou osu		*PGAs/* Vymazání definice elektronické převodovky (EGDEL) [Strana 540]
EGOFC	Kontinuální vypínání elektronické převodovky		*PGAs/* Vypnutí elektronické převodovky (EGOFS, EGOFC) [Strana 539]
EGOFS	Selektivní vypnutí elektronické převodovky		*PGAs/* Vypnutí elektronické převodovky (EGOFS, EGOFC) [Strana 539]
EGON	Zapnutí elektronické převodovky		*PGAs/* Vypnutí elektronické převodovky (EGOFS, EGOFC) [Strana 539]
EGONSYN	Zapnutí elektronické převodovky		*PGAs/* Vypnutí elektronické převodovky (EGOFS, EGOFC) [Strana 539]
EGONSYNE	Zapnutí elektronické převodovky, spolu se zadáním režimu najíždění		*PGAs/* Vypnutí elektronické převodovky (EGOFS, EGOFC) [Strana 539]
ELSE	Větvení programu, pokud podmínka IF není splněna		*PGAs/* Programová smyčka s alternativou (IF, ELSE, ENDIF) [Strana 107]
ENABLE	Zapnutí přerušení		*PGAs/* Deaktivování/opětovné aktivování přiřazení rutiny přerušení (DISABLE, ENABLE) [Strana 124]
ENAT [3]	Přirozený přechod na následující blok posuvu	m	*PGAs/* Splinová interpolace (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [Strana 244]
ENDFOR	Koncový řádek smyčky FOR se zadaným počtem průchodů		*PGAs/* Smyčka s počítadlem (FOR ... TO ..., ENDFOR) [Strana 110]
ENDIF	Koncový řádek větvení IF		*PGAs/* Programová smyčka s alternativou (IF, ELSE, ENDIF) [Strana 107]
ENDLABEL	Koncová značka pro opakování výrobního programu pomocí příkazu REPEAT		*PGAs/, FB1(K1)* Opakování části programu (REPEAT, REPEATB, ENDLABEL, P) [Strana 99]

Příkaz	Význam	W [1]	Popis viz [2]
ENDLOOP	Koncový řádek nekonečné programové smyčky LOOP		*PGAsl* Nekonečná programová smyčka (LOOP, ENDLOOP) [Strana 109]
ENDPROC	Koncový řádek programu, na jehož prvním řádku je příkaz PROC.		
ENDWHILE	Koncový řádek smyčky WHILE		*PGAsl* Programová smyčka s podmínkou na začátku smyčky (WHILE, ENDWHILE) [Strana 112]
ESRR	Nastavení parametrů pro odjíždění nezávislé na pohonu (ESR)		*PGAsl* Konfigurace odjíždění nezávislého na pohonu (ESRR) [Strana 720]
ESRS	Nastavení parametrů pro zastavení nezávislé na pohonu (ESR)		*PGAsl* Konfigurace zastavování nezávislého na pohonu (ESRS) [Strana 719]
ETAN	Tangenciální křivkový přechod na následující blok posuvu na začátku splinu	m	*PGAsl* Splinová interpolace (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [Strana 244]
EVERY	Synchronní akce se uskuteční v případě změny stavu podmínky z FALSE na TRUE		*PGAsl* Cyklická kontrola podmínky (WHEN, WHENEVER, FROM, EVERY) [Strana 561]
EX	Klíčové slovo pro přiřazení hodnoty v exponenciálním způsobu zápisu		*PGAsl* Předdefinované uživatelské proměnné: Početní parametr (R) [Strana 21]
EXECSTRING	Předání proměnné typu String se zpracovatelným řádkem výrobního programu		*PGAsl* Nepřímé programování řádků výrobního programu (EXECSTRING) [Strana 63]
EXECTAB	Spuštění zpracování prvku z tabulky pohybů		*PGAsl* Nepřímé programování řádků výrobního programu (EXECSTRING) [Strana 63]
EXECUTE	Zapnutí zpracovávání programu		*PGAsl* Ukončení přípravy kontury (EXECUTE) [Strana 739]
EXP	Exponenciální funkce ex		*PGAsl* Matematické funkce [Strana 64]
EXTCALL	Zpracovávání externího podprogramu		*PGAsl* Zpracovávání externího podprogramu (EXTCALL) [Strana 205]
EXTCLOSE	Zavření externího přístroje/souboru otevřeného pro zápis		*PGAsl* Výstup do externího zařízení/souboru (EXTOPEN, WRITE, EXTCLOSE) [Strana 708]
EXTERN	Identifikace podprogramu s předáváním parametrů		*PGAsl* Volání podprogramu bez předávání parametrů [Strana 190]
EXTOPEN	Otevření externího přístroje/souboru pro daný kanál pro zápis		*PGAsl* Výstup do externího zařízení/souboru (EXTOPEN, WRITE, EXTCLOSE) [Strana 708]

Příkaz	Význam	W [1]	Popis viz [2]
F	Hodnota posuvu (ve spojení s G4 se pomocí F programuje také doba prodlevy)		*PGs*/
FA	Posuv osy	m	*PGs*/
FAD	Posuv pro přísuv při měkkém najíždění a odjíždění		*PGs*/
FALSE	Logická konstanta: FALSE		*PGAs*/ Definice uživatelských proměnných (DEF) [Strana 25]
FB	Blokový posuv		*PGs*/
FCTDEF	Definice polynomické funkce		*PGAs*/ On-line korekce nástroje (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) [Strana 414]
FCUB	Posuv proměnný podle kubického splinu	m	*PGAs*/ Průběh charakteristiky posuvu (FNORM, FLIN, FCUB, FPO) [Strana 468]
FD	Posuv po dráze pro korekci pomocí ručního kolečka	s	*PGs*/
FDA	Posuv osy při korekci ručním kolečkem	s	*PGs*/
FENDNORM	Vypnutí zpožďování v rozích	m	*PGAs*/ Snížení posuvu se zpožděním v rozích (FENDNORM, G62, G621) [Strana 283]
FFWOF [3]	Vypnutí dopředné regulace	m	*PGs*/
FFWON	Zapnutí dopředné regulace	m	*PGs*/
FGREF	Vztažný rádius u kruhových os nebo vztažný faktor dráhy u orientovaných os (vektorová interpolace)	m	*PGs*/
FGROUP	Definice os s posuvem po dráze		*PGs*/
FI	Parametr pro přístup k datům framu: jemné posunutí		*PGAs*/ Načítání a editace složek framu (TR, FI, RT, SC, MI) [Strana 299]
FIFOCTRL	Ovládání zásobníku dopředné regulace	m	*PGAs*/ Zpracování programu s pamětí předběžného zpracování (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE) [Strana 473]
FILEDATE	Zjištění data posledního přístupu za účelem zápisu do daného souboru		*PGAs*/ Zjišťování informací o souboru (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) [Strana 153]

Příkaz	Význam	W [1]	Popis viz [2]
FILEINFO	Zjištění parametrů FILEDATE, FILESIZE, FILESTAT a FILETIME najednou		*PGAs* Zjišťování informací o souboru (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) [Strana 153]
FILESIZE	Zjištění aktuální velikosti souboru		*PGAs* Zjišťování informací o souboru (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) [Strana 153]
FILESTAT	Zjištění stavových informací o souboru, jako jsou oprávnění pro čtení, zápis, spouštění, vypisování a mazání (rwxsd)		*PGAs* Zjišťování informací o souboru (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) [Strana 153]
FILETIME	Zjištění času posledního přístupu za účelem zápisu do daného souboru		*PGAs* Zjišťování informací o souboru (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) [Strana 153]
FINEA	Konec pohybu při dosažení jemného okna přesného najetí	m	*PGAs* Programovatelné kritérium konce pohybu (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) [Strana 284]
FL	Mezní hodnota rychlosti pro synchronní osy	m	*PGs*
FLIN	Lineárně proměnný posuv	m	*PGAs* Průběh charakteristiky posuvu (FNORM, FLIN, FCUB, FPO) [Strana 468]
FMA	Větší počet axiálních posuvů	m	*PGs*
FNORM [3]	Normální posuv podle DIN 66025	m	*PGAs* Průběh charakteristiky posuvu (FNORM, FLIN, FCUB, FPO) [Strana 468]
FOCOF	Vypnutí najíždění s omezeným momentem/silou	m	*PGAs* Najíždění na pevný doraz (FXS, FXST, FXSW, FOCON, FOCOF) [Strana 628]
FOCON	Zapnutí najíždění s omezeným momentem/silou	m	*PGAs* Najíždění na pevný doraz (FXS, FXST, FXSW, FOCON, FOCOF) [Strana 628]
FOR	Smyčka s pevně daným počtem průchodů		*PGAs* Smyčka s počítadlem (FOR ... TO ..., ENDFOR) [Strana 110]
FP	Pevný bod: Číslo pevného bodu, na který se má najíždět	s	*PGs*
FPO	Průběh posuvu naprogramovaný pomocí polynomu		*PGAs* Průběh charakteristiky posuvu (FNORM, FLIN, FCUB, FPO) [Strana 468]
FPR	Označení kruhové osy		*PGs*
FPRAOF	Deaktivování otáčkového posuvu		*PGs*

Příkaz	Význam	W [1]	Popis viz [2]
FPRAON	Aktivování otáčkového posuvu		*PGs/*
FRAME	Datový typ určený pro definici souřadných systémů		*PGAs/* Definice nového framu (DEF FRAME) [Strana 302]
FRC	Posuv pro rádius a fasetu	s	*PGs/*
FRCM	Modální posuv pro rádius a fasetu	m	*PGs/*
FROM	Akce se uskuteční, jestliže je daná podmínka jednorázově splněna a pokud je aktivní synchronní akce.		*PGAs/* Cyklická kontrola podmínky (WHEN, WHENEVER, FROM, EVERY) [Strana 561]
FTOC	Změna jemné korekce nástroje		*PGAs/* On-line korekce nástroje (FTOC) [Strana 598]
FTOCOF [3]	Vypnutí on-line působící jemné korekce nástroje	m	*PGAs/* On-line korekce nástroje (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) [Strana 414]
FTOCON	Zapnutí on-line působící jemné korekce nástroje	m	*PGAs/* On-line korekce nástroje (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) [Strana 414]
FXS	Najíždění na pevný doraz	m	*PGs/*
FXST	Mezní hodnota momentu pro najíždění na pevný doraz:	m	*PGs/*
FXSW	Monitorovací okno pro najíždění na pevný doraz		*PGs/*
FZ	Posuv na zub	m	*PGs/*

Příkaz	Význam	W [1]	Popis viz [2]
G0	Lineární interpolace rychlým posuvem (pohyb rychlým posuvem)	m	*PGs/*
G1 [3]	Lineární interpolace s pracovním posuvem (přímková interpolace)	m	*PGs/*
G2	Kruhová interpolace ve směru hodinových ručiček	m	*PGs/*
G3	Kruhová interpolace proti směru hodinových ručiček	m	*PGs/*
G4	Doba prodlevy určená časově	s	*PGs/*
G5	Šikmé zapichovací broušení	s	*PGAs/* Šikmá osa (TRAANG) [Strana 381]
G7	Vyrovnávací pohyb při šikmém zapichovacím broušení	s	*PGAs/* Šikmá osa (TRAANG) [Strana 381]

Příkaz	Význam	W [1]	Popis viz [2]
G9	Přesné najetí – snižování rychlosti	s	*PGsl*
G17 [3]	Volba pracovní roviny X/Y	m	*PGsl*
G18	Volba pracovní roviny Z/X	m	*PGsl*
G19	Volba pracovní roviny Y/Z	m	*PGsl*
G25	Dolní ohraničení pracovního pole	s	*PGsl*
G26	Horní ohraničení pracovního pole	s	*PGsl*
G33	Řezání závitů s konstantním stoupáním	m	*PGsl*
G34	Řezání závitu s lineárně narůstajícím stoupáním	m	*PGsl*
G35	Řezání závitu s lineárně klesajícím stoupáním	m	*PGsl*
G40 [3]	Vypnutí korekce rádiusu nástroje	m	*PGsl*
G41	Korekce rádiusu nástroje vlevo od kontury	m	*PGsl*
G42	Korekce rádiusu nástroje vpravo od kontury	m	*PGsl*
G53	Potlačení aktuálního posunutí počátku (blokové)	s	*PGsl*
G54	1. nastavitelné posunutí počátku	m	*PGsl*
G55	2. nastavitelné posunutí počátku	m	*PGsl*
G56	3. nastavitelné posunutí počátku	m	*PGsl*
G57	4. nastavitelné posunutí počátku	m	*PGsl*
G58 (840D sl)	Osové programovatelné posunutí počátku - absolutní, hrubé posunutí	s	*PGsl*
G58 (828D)	5. nastavitelné posunutí počátku	m	*PGsl*
G59 (840D sl)	Osové programovatelné posunutí počátku - aditivní, jemné posunutí	s	*PGsl*
G59 (828D)	6. nastavitelné posunutí počátku	m	*PGsl*

Příkaz	Význam	W [1]	Popis viz [2]
G60 [3]	Přesné najetí – snižování rychlosti	m	*/PGs/*
G62	Zpoždění na vnitřních rozích při aktivní korekci rádiusu nástroje (G41, G42)	m	*/PGAs/* Snížení posuvu se zpožděním v rozích (FENDNORM, G62, G621) [Strana 283]
G63	Vrtání závitů s vyrovnávací hlavičkou	s	*/PGs/*
G64	Režim řízení pohybu po dráze	m	*/PGs/*
G70	Rozměrové údaje pro geometrické parametry v palcích (délky)	m	*/PGs/*
G71 [3]	Rozměrové údaje pro geometrické parametry v metrických jednotkách (délky)	m	*/PGs/*
G74	Najíždění na referenční bod	s	*/PGs/*
G75	Najíždění na pevný bod	s	*/PGs/*
G90 [3]	Zadávání absolutních rozměrů	m/s	*/PGs/*
G91	Zadávání inkrementálních rozměrů	m/s	*/PGs/*
G93	Časově reciproční posuv v jednotkách 1/min	m	*/PGs/*
G94 [3]	Lineární posuv F v mm/min, v palcích/min nebo stupních/min	m	*/PGs/*
G95	Otáčkový posuv F v mm/ot nebo palcích/ot	m	*/PGs/*
G96	Zapnutí konstantní řezné rychlosti (jako u G95)	m	*/PGs/*
G97	Vypnutí konstantní řezné rychlosti (jako u G95)	m	*/PGs/*
G110	Programování pólu vztažené na naposled naprogramovanou požadovanou pozici	s	*/PGs/*
G111	Programování pólu vzhledem k počátku aktuální souřadné soustavy obrobku	s	*/PGs/*
G112	Programování pólu vzhledem k poslednímu platnému pólu	s	*/PGs/*
G140 [3]	Směr najíždění WAB definován příkazy G41/G42	m	*/PGs/*
G141	Směr najíždění WAB vlevo od kontury	m	*/PGs/*

Příkaz	Význam	W [1]	Popis viz [2]
G142	Směr najíždění WAB vpravo od kontury	m	*PGsl*
G143	Směr najíždění WAB v závislosti na tečně	m	*PGsl*
G147	Měkké najíždění po přímce	s	*PGsl*
G148	Měkké odjíždění po přímce	s	*PGsl*
G153	Potlačení aktuálního framu včetně základního framu	s	*PGsl*
G247	Měkké najíždění po čtvrtkruhu	s	*PGsl*
G248	Měkké odjíždění po čtvrtkruhu	s	*PGsl*
G290	Aktivování přepnutí na režim SINUMERIK	m	*FBW*
G291	Aktivování přepnutí na režim ISO2/3	m	*FBW*
G331	Vrtání závitů bez vyrovnávací hlavičky, kladné stoupání, směr otáčení vpravo	m	*PGsl*
G332	Vrtání závitů bez vyrovnávací hlavičky, záporné stoupání, směr otáčení vlevo	m	*PGsl*
G340 [3]	Prostorový najížděcí blok (hloubka a v rovině stejné – spirála)	m	*PGsl*
G341	Napřed přísuv v kolmé ose (z), pak najíždění v rovině	m	*PGsl*
G347	Měkké najíždění po půlkruhu	s	*PGsl*
G348	Měkké odjíždění po půlkruhu	s	*PGsl*
G450 [3]	Přechodový kruh	m	*PGsl*
G451	Průsečík ekvidistantních drah	m	*PGsl*
G460 [3]	Aktivování protikolizního monitorování pro blok najíždění a odjíždění	m	*PGsl*
G461	Vložení kruhu do bloku korekce rádiusu nástroje	m	*PGsl*
G462	Vložení přímky do bloku korekce rádiusu nástroje	m	*PGsl*
G500 [3]	Deaktivování všech nastavitelných framů, základní frame je aktivní	m	*PGsl*

Příkaz	Význam	W [1]	Popis viz [2]
G505 ... G599	5 ... 99. nastavitelné posunutí počátku	m	*PGs/*
G601 [3]	Přechod na další blok při jemném přesném najetí	m	*PGs/*
G602	Přechod na další blok při hrubém přesném najetí	m	*PGs/*
G603	Přechod na další blok při konci bloku IPO	m	*PGs/*
G621	Snížení rychlosti na všech rozích	m	*PGAs/* Snížení posuvu se zpožděním v rozích (FENDNORM, G62, G621) [Strana 283]
G641	Režim řízení pohybu po dráze s přechodovými zaobleními podle kritéria dráhy (= programovatelná vzdálenost zaoblení)	m	*PGs/*
G642	Režim řízení pohybu po dráze s přechodovými zaobleními při dodržení definovaných tolerancí	m	*PGs/*
G643	Režim řízení pohybu po dráze s přechodovými zaobleními při dodržení definovaných tolerancí (uvnitř bloku)	m	*PGs/*
G644	Režim řízení pohybu po dráze s přechodovými zaobleními s maximální možnou dynamikou	m	*PGs/*
G645	Režim řízení pohybu po dráze s přechodovými zaobleními v rozích a s tangenciálními přechody mezi bloky při dodržení definovaných tolerancí	m	*PGs/*
G700	Rozměrové údaje pro geometrické a technologické parametry v palcích (délky, posuv)	m	*PGs/*
G710 [3]	Rozměrové údaje pro geometrické a technologické parametry v metrických jednotkách (délky, posuv)	m	*PGs/*
G751	Najíždění na pevný bod přes pomocný bod	s	PGsl
G810 [3], ..., G819	G-skupina vyhrazená pro uživatele OEM		*PGAs/* Speciální funkce pro uživatele OEM (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829) [Strana 282]
G820 [3], ..., G829	G-skupina vyhrazená pro uživatele OEM		*PGAs/* Speciální funkce pro uživatele OEM (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829) [Strana 282]
G931	Zadávání posuvu dobou pohybu	m	
G942	Zmrazení lineárního posuvu a konstantní řezné rychlosti nebo otáček vřetena	m	

Příkaz	Význam	W [1]	Popis viz [2]
G952	Zmrazení otáčkového posuvu a konstantní řezné rychlosti nebo otáček vřetena	m	
G961	Konstantní řezná rychlost a lineární posuv	m	*PGsl*
G962	Lineární posuv nebo otáčkový posuv a konstantní řezná rychlost	m	*PGsl*
G971	Zmražení otáček vřetena a lineární posuv	m	*PGsl*
G972	Zmrazení lineárního nebo otáčkového posuvu a konstantních otáček vřetena	m	*PGsl*
G973	Otáčkový posuv bez omezení otáček vřetena	m	*PGsl*
GEOAX	Nové přiřazení geometrických os 1 - 3 kanálovým osám		*PGAsl* Přepínatelné geometrické osy (GEOAX) [Strana 680]
GET	Výměna uvolněné osy mezi kanály		*PGAsl* Výměna osy, výměna vřetena (RELEASE, GET, GETD) [Strana 132]
GETACTT	Stanovení aktivního nástroje ze skupiny stejnojmenných nástrojů.		*FBW*
GETACTTD	Stanovení odpovídajícího T-čísla k absolutnímu D-číslu		*PGAsl* Volné zadávání D-čísel: Zjišťování T-čísla k zadanému D-číslu (GETACTTD) [Strana 441]
GETD	Výměna uvolněné osy mezi kanály		*PGAsl* Výměna osy, výměna vřetena (RELEASE, GET, GETD) [Strana 132]
GETDNO	Zjištění D-čísla břitu (CE) daného nástroje (T)		*PGAsl* Volné zadávání D-čísel: Přejmenovávání D-čísel (GETDNO, SETDNO) [Strana 440]
GETEXET	Načtení T-čísla vyměňovaného nástroje		*FBW*
GETFREELOC	Vyhledání volného místa v zásobníku pro zadaný nástroj		*FBW*
GETSELT	Zjištění předem zvoleného T-čísla		*FBW*
GETT	Přiřazení T-čísla názvu nástroje		*FBW*
GETTCOR	Načtení délek nástroje, příp. složek délky nástroje		*FB1(W1)*
GETTENV	Načtení T-čísla, D-čísla a DL-čísla		*FB1(W1)*
GOTO	Příkaz skoku napřed dopředu a potom zpátky (napřed ve směru konce programu a potom směrem k začátku programu)		*PGAsl* Programové skoky na návěští skoků (GOTOB, GOTOF, GOTO, GOTOC) [Strana 94]
GOTOB	Příkaz skoku směrem dozadu (směrem k začátku programu)		*PGAsl* Programové skoky na návěští skoků (GOTOB, GOTOF, GOTO, GOTOC) [Strana 94]

Příkaz	Význam	W [1]	Popis viz [2]
GOTOC	Stejné jako GOTO, ale s potlačením alarmu 14080 „Cíl skoku nenalezen“.		*PGAs/* Programové skoky na návěští skoků (GOTOB, GOTOF, GOTO, GOTOC) [Strana 94]
GOTOF	Příkaz skoku směrem dopředu (směrem ke konci programu)		*PGAs/* Programové skoky na návěští skoků (GOTOB, GOTOF, GOTO, GOTOC) [Strana 94]
GOTOS	Skok zpátky na začátek programu		*PGAs/* Skok zpátky na začátek programu (GOTOS) [Strana 93]
GP	Klíčové slovo pro nepřímé programování atributů polohy		*PGAs/* Nepřímé programování atributů polohy (GP) [Strana 60]
GWPSOF	Deaktivování konstantní obvodové rychlosti brusného kotouče	s	*PGs/*
GWPSON	Aktivování konstantní obvodové rychlosti brusného kotouče	s	*PGs/*
H...	Výstup pomocných funkcí do PLC		*PGsl/FB1(H2)*
HOLES1	Technologický cyklus: Řada děr		*PGAs/* Řada děr - HOLES1 [Strana 758]
HOLES2	Technologický cyklus: Díry na kruhovém oblouku		*PGAs/* Díry uspořádané na kruhovém oblouku - HOLES2 [Strana 760]
I	Interpolační parametr	s	*PGs/*
I1	Souřadnice vnitřního bodu	s	*PGs/*
IC	Zadávání inkrementálních rozměrů	s	*PGs/*
ICYCOF	Všechny bloky technologického cyklu za příkazem ICYCOF zpracovávat v taktu IPO		*PGAs/* Řízení zpracování technologických cyklů (ICYCOF, ICYCON) [Strana 639]
ICYCON	Každý blok technologického cyklu za příkazem ICYCON zpracovávat v samostatném taktu IPO		*PGAs/* Řízení zpracování technologických cyklů (ICYCOF, ICYCON) [Strana 639]
ID	Identifikace pro modální synchronní akce	m	*PGAs/* Oblast platnosti a posloupnost při zpracovávání (ID, IDS) [Strana 559]
IDS	Identifikace pro modální statické synchronní akce		*PGAs/* Oblast platnosti a posloupnost při zpracovávání (ID, IDS) [Strana 559]
IF	Úvodní příkaz pro podmíněné skoky ve výrobním programu / technologickém cyklu		*PGAs/* Programová smyčka s alternativou (IF, ELSE, ENDIF) [Strana 107]

Příkaz	Význam	W [1]	Popis viz [2]
INDEX	Stanovení indexu znaku ve vstupním řetězci		*PGAs/* Vyhledávání znaků/řetězců v řetězci (INDEX, RINDEX, MINDEX, MATCH) [Strana 81]
INIPO	Inicializace proměnných při zapnutí systému (Power-On)		*PGAs/* Definice uživatelských proměnných (DEF) [Strana 25]
INIRE	Inicializace proměnných při resetu		*PGAs/* Definice uživatelských proměnných (DEF) [Strana 25]
INICF	Inicializace proměnných při aktivování nové konfigurace (NewConfig)		*PGAs/* Definice uživatelských proměnných (DEF) [Strana 25]
INIT	Aktivování určitého NC programu pro zpracovávání v určitém kanálu.		*PGAs/* Koordinace programů (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) [Strana 115]
INITIAL	Vytvoření souboru INI pro všechny oblasti		*PGAs/* Pracovní paměť (CHANDATA, COMPLETE, INITIAL) [Strana 222]
INT	Datový typ: Celočíselné hodnoty se znaménkem		*PGAs/* Definice uživatelských proměnných (DEF) [Strana 25]
INTERSEC	Výpočet průsečíku mezi dvěma konturovými prvky		*PGAs/* Zjištění průsečíku mezi dvěma konturovými prvky (INTERSEC) [Strana 734]
INVCCW	Pohyb po evolventě, proti směru hodinových ručiček	m	*PGs/*
INVCW	Pohyb po evolventě, ve směru hodinových ručiček	m	*PGs/*
INVFRAME	Výpočet inverzního framu z daného framu		*FB1(K2)*
IP	Proměnný interpolační parametr		*PGAs/* Nepřímé programování [Strana 56]
IPOBRKA	Kritérium pro řízení pohybu od počátečního bodu skokové změny brzdné charakteristiky	m	*PGAs/* Programovatelné kritérium konce pohybu (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) [Strana 284]
IPOENDA	Konec pohybu při dosažení "IPO Stop"	m	*PGAs/* Programovatelné kritérium konce pohybu (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) [Strana 284]
IPTRLOCK	Napevno nastavit začátek úseku programu, který může být prohledáván, na následující blok s funkcemi stroje	m	*PGAs/* Zabránění pozice programu pro SERUPRO (IPTRLOCK, IPTRUNLOCK) [Strana 481]
IPTRUNLOCK	Nastavení konce úseku programu, který může být prohledáván, na aktuální blok v okamžiku přerušení	m	*PGAs/* Zabránění pozice programu pro SERUPRO (IPTRLOCK, IPTRUNLOCK) [Strana 481]
ISAXIS	Kontrola, zda geometrická osa 1 zadaná jako parametr existuje		*PGAs/* Osové funkce (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) [Strana 677]

Příkaz	Význam	W [1]	Popis viz [2]
ISD	Hloubka zajíždění nástroje	m	*PGAs*/ Aktivování 3D korekcí nástroje (CUT3DC, CUT3DF, CUT3DFS, CUT3DFF, ISD) [Strana 419]
ISFILE	Kontrola, zda daný soubor v uživatelské paměti NCK je k dispozici		*PGAs*/ Kontrola existence souboru (ISFILE) [Strana 151]
ISNUMBER	Kontrola, zda je možno převést vstupní řetězec na číslo.		*PGAs*/ Převod z datového typu STRING (NUMBER, ISNUMBER, AXNAME) [Strana 77]
ISOCALL	Nepřímé volání programu naprogramovaného v jazyce ISO		*PGAs*/ Nepřímé volání programu naprogramovaného v jazyce ISO (ISOCALL) [Strana 200]
ISVAR	Kontrola, zda předávaný parametr obsahuje proměnnou, která je NC systému známá.		*PGAs*/ Volání funkce ISVAR a čtení indexu pole strojních parametrů [Strana 697]
J	Interpolační parametr	s	*PGs*/
J1	Souřadnice vnitřního bodu	s	*PGs*/
JERKA	Aktivování chování při zrychlení, které je nastaveno prostřednictvím MD, pro naprogramované osy		
JERKLIM	Snížení nebo zvýšení maximálního osového ryvu	m	*PGAs*/ Procentuální korekce ryvu (JERKLIM) [Strana 493]
JERKLIMA	Snížení nebo zvýšení maximálního osového ryvu	m	*PGs*/
K	Interpolační parametr	s	*PGs*/
K1	Souřadnice vnitřního bodu	s	*PGs*/
KONT	Objíždění kontury s korekcí nástroje	m	*PGs*/
KONTC	Najíždění/odjíždění se spojitým polynomickým zakřivením	m	*PGs*/
KONTT	Najíždění/odjíždění se spojitým tangenciálním polynomem	m	*PGs*/
L	Číslo podprogramu	s	*PGAs*/ Volání podprogramu bez předávání parametrů [Strana 190]
LEAD	Předozadní úhel 1. Orientace nástroje 2. Polynom pro řízení orientace	m	*PGAs*/ Programování orientace nástroje (A..., B..., C..., LEAD, TILT) [Strana 333]
LEADOF	Deaktivování vazby pomocí řídící hodnoty		*PGAs*/ Osová vazba řídící hodnotou (LEADON, LEADOF) [Strana 528]

Příkaz	Význam	W [1]	Popis viz [2]
LEADON	Aktivování vazby pomocí řídící hodnoty		*PGAs/* Osová vazba řídící hodnotou (LEADON, LEADOF) [Strana 528]
LENTOAX	Vyvolání informací o přiřazení délek nástroje L1, L2 a L3 aktivního nástroje abscise, ordinátě a aplikátě.		*FB1(W1)*
LFOF [3]	Deaktivování rychlého zpětného pohybu při řezání závitů	m	*PGs/*
LFON	Aktivování rychlého zpětného pohybu při řezání závitů	m	*PGs/*
LFPOS	Návrat osy stanovené příkazem POLFMASK nebo POLFLIN na absolutní pozici naprogramovanou pomocí příkazu POLF.	m	*PGs/*
LFTXT	Rovina zpětného pohybu při rychlém pozvednutí nástroje se bude určovat na základě tečny k dráze a momentálního směru nástroje.	m	*PGs/*
LFWP	Rovina zpětného pohybu při rychlém pozvednutí nástroje je určena prostřednictvím aktuální pracovní roviny (G17/G18/G19).	m	*PGs/*
LIFTFAST	Rychlé pozvednutí		*PGs/* Rychlé pozvednutí od kontury (SETINT LIFTFAST, ALF) [Strana 126]
LIMS	Omezení otáček u příkazů G96/G961 a G97	m	*PGs/*
LLI	Spodní mezní hodnota pro proměnné		*PGAs/* Atribut: Mezní hodnoty (LLI, ULI) [Strana 37]
LN	Přirozený logaritmus		*PGAs/* Matematické funkce [Strana 64]
LOCK	Zablokování synchronní akce s identifikací ID (zastavení technologického cyklu)		*PGAs/* Blokování, uvolnění, reset (LOCK, UNLOCK, RESET) [Strana 642]
LONGHOLE	Technologický cyklus: Podlouhlá díra		*PGAs/* Podlouhlá díra - LONGHOLE [Strana 781]
LOOP	Úvodní příkaz nekonečné smyčky		*PGAs/* Nekonečná programová smyčka (LOOP, ENDLOOP) [Strana 109]

Příkaz	Význam	W [1]	Popis viz [2]
M0	Programovatelné zastavení		*PGs/*
M1	Volitelné zastavení		*PGs/*

Příkaz	Význam	W [1]	Popis viz [2]
M2	Konec hlavního programu s návratem na začátek programu		*/PGs/*
M3	Vřeteno se otáčí vpravo		*/PGs/*
M4	Vřeteno se otáčí vlevo		*/PGs/*
M5	Zastavení vřetena		*/PGs/*
M6	Výměna nástroje		*/PGs/*
M17	Konec podprogramu		*/PGs/*
M19	Nastavení pozice vřetena na polohu zadanou v SD 43240		*/PGs/*
M30	Konec programu, stejné jako M2		*/PGs/*
M40	Automatické přepínání stupňů převodovky		*/PGs/*
M41 ... M45	Stupeň převodovky 1 ... 5		*/PGs/*
M70	Přechod do osového režimu		*/PGs/*
MASLDEF	Definice spojení os typu Master/Slave		*/PGAs/* Spojení master/slave (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) [Strana 553]
MASLDEL	Zrušení spojení os typu Master/Slave a vymazání definice tohoto spojení		*/PGAs/* Spojení master/slave (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) [Strana 553]
MASLOF	Deaktivování dočasného spojení		*/PGAs/* Spojení master/slave (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) [Strana 553]
MASLOFS	Deaktivování dočasného spojení s automatickým zastavením osy typu Slave		*/PGAs/* Spojení master/slave (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) [Strana 553]
MASLON	Aktivování dočasného spojení		*/PGAs/* Spojení master/slave (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) [Strana 553]
MATCH	Vyhledávání řetězce v rámci jiného řetězce		*/PGAs/* Vyhledávání znaků/řetězců v řetězci (INDEX, RINDEX, MINDEX, MATCH) [Strana 81]
MAXVAL	Větší z hodnot ve dvou proměnných (aritmetická funkce)		*/PGAs/* Minimum, maximum a rozsah proměnných (MINVAL, MAXVAL, BOUND) [Strana 71]

Příkaz	Význam	W [1]	Popis viz [2]
MCALL	Modální volání podprogramu		*PGAsl* Modální volání podprogramu (MCALL) [Strana 196]
MEAC	Kontinuální měření bez mazání zbytkové dráhy	s	*PGAsl* Rozšířené měřicí funkce (MEASA, MEAWA, MEAC) (volitelný doplněk) [Strana 273]
MEAFRAME	Výpočet framu na základě změřených bodů		*PGAsl* Výpočet framu na základě 3 změřených bodů v prostoru (MEAFRAME) [Strana 308]
MEAS	Měření se spínací sondou	s	*PGAsl* Měření se spínací sondou (MEAS, MEAW) [Strana 270]
MEASA	Měření s vymazáním zbytkové dráhy	s	*PGAsl* Rozšířené měřicí funkce (MEASA, MEAWA, MEAC) (volitelný doplněk) [Strana 273]
MEASURE	Metoda výpočtu pro měření obrobku a nástroje		*FB2(M5)* Měření se spínací sondou (MEAS, MEAW) [Strana 270]
MEAW	Měření se spínací sondou bez vymazání zbytkové dráhy	s	*PGAsl* Měření se spínací sondou (MEAS, MEAW) [Strana 270]
MEAWA	Měření bez mazání zbytkové dráhy	s	*PGAsl* Rozšířené měřicí funkce (MEASA, MEAWA, MEAC) (volitelný doplněk) [Strana 273]
MI	Přístup k datům framu: Zrcadlové převrácení		*PGAsl* Načítání a editace složek framu (TR, FI, RT, SC, MI) [Strana 299]
MINDEX	Stanovení indexu znaku ve vstupním řetězci		*PGAsl* Vyhledávání znaků/řetězců v řetězci (INDEX, RINDEX, MINDEX, MATCH) [Strana 81]
MINVAL	Menší z hodnot ve dvou proměnných (aritmetická funkce)		*PGAsl* Minimum, maximum a rozsah proměnných (MINVAL, MAXVAL, BOUND) [Strana 71]
MIRROR	Programovatelné zrcadlové převrácení	s	*PGAsl*
MMC	Vyvolání interaktivního dialogového okna na HMI z výrobního programu		*PGAsl* Interaktivní vyvolávání oken z výrobního programu (MMC) [Strana 701]
MOD	Dělení MODULO		*PGAsl* Matematické funkce [Strana 64]
MODAXVAL	Zjištění pozice modulo kruhové osy modulo		*PGAsl* Osové funkce (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) [Strana 677]
MOV	Spuštění polohovací osy		*PGAsl* Zastavení/spuštění osy (MOV) [Strana 607]

Příkaz	Význam	W [1]	Popis viz [2]
MSG	Programovatelná hlášení	m	*PGs/*
MVTOOL	Příkaz jazyka pro pohyb nástroje		*FBW*
N	Číslo vedlejšího bloku v NC programu		*PGs/*
NCK	Specifikace rozsahu platnosti dat		*PGAs/* Definice uživatelských proměnných (DEF) [Strana 25]
NEWCONF	Převzetí změněných strojních parametrů (odpovídá příkazu "Nastavit strojní parametr jako platný")		*PGAs/* Aktivování strojních parametrů (NEWCONF) [Strana 139]
NEWT	Založení nového nástroje		*PGAs/* Načítání a editace složek framu (TR, FI, RT, SC, MI) [Strana 299]
NORM [3]	Normální nastavení v počátečním a koncovém bodě při korekci nástroje	m	*PGs/*
NOT	Logické NE (negace)		*PGAs/* Porovnávací a logické operace [Strana 67]
NPROT	Zapnutí/vypnutí chráněné oblasti pro specifický stroj		*PGAs/* Aktivování/deaktivování chráněné oblasti (CPROT, NPROT) [Strana 231]
NPROTDEF	Definice chráněné oblasti pro specifický stroj		*PGAs/* Stanovení chráněné oblasti (CPROTDEF, NPROTDEF) [Strana 227]
NUMBER	Převedení vstupního řetězce na číslo		*PGAs/* Převod z datového typu STRING (NUMBER, ISNUMBER, AXNAME) [Strana 77]
OEMIPO1	Interpolace OEM 1	m	*PGAs/* Speciální funkce pro uživatele OEM (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829) [Strana 282]
OEMIPO2	Interpolace OEM 2	m	*PGAs/* Speciální funkce pro uživatele OEM (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829) [Strana 282]
OF	Klíčové slovo v příkazu větvení CASE		*PGAs/* Větvení programu (CASE ... OF ... DEFAULT ...) [Strana 97]
OFFN	Přídavek rozměru pro naprogramovanou konturu	m	*PGs/*
OMA1	Adresa OEM 1	m	
OMA2	Adresa OEM 2	m	
OMA3	Adresa OEM 3	m	
OMA4	Adresa OEM 4	m	
OMA5	Adresa OEM 5	m	

Příkaz	Význam	W [1]	Popis viz [2]
OR	Logický operátor, spojení typu NEBO		*PGAsl* Porovnávací a logické operace [Strana 67]
ORIAXES	Lineární interpolace os stroje nebo orientačních os	m	*PGAsl* Programování orientačních os (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) [Strana 344]
ORIAXPOS	Úhel orientace pomocí virtuálních orientačních os s polohováním kruhové osy	m	
ORIC [3]	Změny orientace na vnějších rozích jsou superponovány s vkládaným kruhovým blokem	m	*PGAsl* Orientace nástroje (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) [Strana 433]
ORICONCCW	Interpolace po ploše pláště kužele proti směru hodinových ručiček.	m	*PGAsl/FB3(F3)* Programování orientace podél kuželové plášťové plochy (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) [Strana 346]
ORICONCW	Interpolace po ploše pláště kužele ve směru hodinových ručiček.	m	*PGAsl/FB3(F4)* Programování orientace podél kuželové plášťové plochy (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) [Strana 346]
ORICONIO	Interpolace po ploše pláště kužele s udáním pomocné meziorientace	m	*PGAsl/FB3(F4)* Programování orientace podél kuželové plášťové plochy (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) [Strana 346]
ORICONTO	Interpolace po ploše pláště kuželu s tangenciálním přechodem. (zadání koncové orientace)	m	*PGAsl/FB3(F5)* Programování orientace podél kuželové plášťové plochy (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) [Strana 346]
ORICURVE	Interpolace orientace s udáním pohybu dvou kontaktních bodů nástroje	m	*PGAsl/FB3(F6)* Zadání orientace pomocí druhého styčného bodu (ORICURVE, PO[XH]=, PO[YH]=, PO[ZH]=) [Strana 350]
ORID	Změna orientace se bude provádět před kruhovým blokem.	m	*PGAsl* Orientace nástroje (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) [Strana 433]
ORIEULER	Úhel orientace pomocí Eulerova úhlu	m	*PGAsl* Programování orientačních os (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) [Strana 344]
ORIMKS	Orientace nástroje v souřadném systému stroje	m	*PGAsl* Vztah orientačních os (ORIWKS, ORIMKS) [Strana 342]
ORIPATH	Orientace nástroje vztažená na dráhu	m	*PGAsl* Otáčení orientace nástroje vzhledem k dráze (ORIPATH, ORIPATHS, úhel otočení) [Strana 359]

Příkaz	Význam	W [1]	Popis viz [2]
ORIPATHS	Orientace nástroje vztažená na dráhu, zlom v průběhu orientace se vyhladí	m	*PGAs*/ Otáčení orientace nástroje vzhledem k dráze (ORIPATH, ORIPATHS, úhel otočení) [Strana 359]
ORIPLANE	Interpolace v rovině (odpovídá ORIVECT) kruhová interpolace s velkým rádiusem	m	*PGAs*/ Programování orientace podél kuželové plášťové plochy (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) [Strana 346]
ORIRESET	Základní poloha orientace nástroje s až 3 orientačními osami		*PGAs*/ Varianty programování orientace a základního nastavení (ORIRESET) [Strana 332]
ORIROTA	Úhel otočení vztažen ke směru otáčení zadanému absolutně	m	*PGAs*/ Otáčení orientace nástroje (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) [Strana 354]
ORIROTC	Tangenciální vektor otočení k tečně dráhy	m	*PGAs*/ Otáčení orientace nástroje (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) [Strana 354]
ORIROTR	Úhel otočení relativně vůči rovině mezi počáteční a koncovou orientací	m	*PGAs*/ Otáčení orientace nástroje (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) [Strana 354]
ORIROTT	Úhel otočení relativně vůči změně vektoru orientace	m	*PGAs*/ Otáčení orientace nástroje (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) [Strana 354]
ORIRPY	Úhel orientace prostřednictvím úhlu RPY (XYZ)	m	*PGAs*/ Programování orientačních os (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) [Strana 344]
ORIRPY2	Úhel orientace pomocí úhlu RPY (ZYX)	m	*PGAs*/ Programování orientačních os (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) [Strana 344]
ORIS	Změna orientace	m	*PGAs*/ Orientace nástroje (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) [Strana 433]
ORISOF [3]	Vypnutí vyhlazování charakteristiky orientace	m	*PGAs*/ Zapnutí vyhlazování charakteristiky orientace (ORISON, ORISOF) [Strana 367]
ORISON	Zapnutí vyhlazování charakteristiky orientace	m	*PGAs*/ Zapnutí vyhlazování charakteristiky orientace (ORISON, ORISOF) [Strana 367]
ORIVECT	Interpolace pomocí největší kružnice koule (identická s ORIPLANE)	m	*PGAs*/ Programování orientačních os (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) [Strana 344]
ORIVIRT1	Úhel orientace pomocí virtuálních orientačních os (definice 1)	m	*PGAs*/ Programování orientačních os (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) [Strana 344]

Příkaz	Význam	W [1]	Popis viz [2]
ORIVIRT2	Úhel orientace pomocí virtuálních orientačních os (definice 1)	m	*PGAsl* Programování orientačních os (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) [Strana 344]
ORIWKS [3]	Orientace nástroje v souřadném systému obrobku	m	*PGAsl* Vztah orientačních os (ORIWKS, ORIMKS) [Strana 342]
OS	Oscilační pohyb ZAP/VYP		*PGAsl* Asynchronní oscilační pohyby (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) [Strana 649]
OSB	Oscilace: Počáteční bod	m	*FB2(P5)*
OSC	Konstantní vyhlazení orientace nástroje	m	*PGAsl* Orientace nástroje (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) [Strana 433]
OSCILL	Osa: 1 - 3 přísuvná osa	m	*PGAsl* Oscilační pohyby řízené prostřednictvím synchronních akcí (OSCILL) [Strana 655]
OSCTRL	Možnosti oscilačního pohybu	m	*PGAsl* Asynchronní oscilační pohyby (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) [Strana 649]
OSD	Vyhlazování orientace nástroje zadáním délky zaoblení pomocí parametru SD	m	*PGAsl* Orientace nástroje (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) [Strana 433]
OSE	Koncový bod oscilačního pohybu	m	*PGAsl* Asynchronní oscilační pohyby (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) [Strana 649]
OSNSC	Oscilace: Počet vyjiskření	m	*PGAsl* Asynchronní oscilační pohyby (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) [Strana 649]
OSOF [3]	Vypnutí vyhlazování orientace nástroje	m	*PGAsl* Asynchronní oscilační pohyby (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) [Strana 649]
OSP1	Oscilace: levý bod obratu	m	*PGAsl* Asynchronní oscilační pohyby (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) [Strana 649]
OSP2	Oscilace: pravý bod obratu	m	*PGAsl* Asynchronní oscilační pohyby (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) [Strana 649]

Příkaz	Význam	W [1]	Popis viz [2]
OSS	Vyhlazení orientace nástroje na konci bloku	m	*PGAs*/ Orientace nástroje (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) [Strana 433]
OSSE	Vyhlazení orientace nástroje na počátku a na konci bloku	m	*PGAs*/ Orientace nástroje (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) [Strana 433]
OST	Vyhlazování orientace nástroje zadáním úhlové tolerance ve stupních pomocí parametru SD (maximální odchylka od naprogramovaného průběhu orientace)	m	*PGAs*/ Orientace nástroje (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) [Strana 433]
OST1	Oscilace: stop v levém bodu obratu	m	*PGAs*/ Asynchronní oscilační pohyby (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) [Strana 649]
OST2	Oscilace: stop v pravém bodu obratu	m	*PGAs*/ Asynchronní oscilační pohyby (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) [Strana 649]
OTOL	Tolerance orientace pro funkce kompresoru, vyhlazení orientace a druhy zaoblování konturových přechodů		*PGAs*/ Programovatelná tolerance kontury/orientace (CTOL, OTOL, ATOL) [Strana 498]
OVR	Korekce otáček	m	*PGAs*/
OVRA	Korekce otáček pro osu	m	*PGAs*/
OVRRAP	Korekce rychlého posuvu	m	*PGAs*/
P	Počet průchodů podprogramem		*PGAs*/ Počet opakování programu (P) [Strana 194]
PAROT	Srovnání souřadného systému obrobku podle obrobku	m	*PGs*/
PAROTOF	Vypnutí otáčení framu vztahující se na obrobek	m	*PGs*/
PCALL	Volání podprogramu s absolutním udáním cesty a předáváním parametrů		*PGAs*/ Volání podprogramu s udáním cesty a parametrů (PCALL) [Strana 202]
PDELAYOF	Deaktivování zpoždění při lisování	m	*PGAs*/ Zapnutí nebo vypnutí lisování a prostřihování (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) [Strana 663]
PDELAYON [3]	Aktivování zpoždění při lisování	m	*PGAs*/ Zapnutí nebo vypnutí lisování a prostřihování (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) [Strana 663]

Příkaz	Význam	W [1]	Popis viz [2]
PHU	Fyzikální jednotka proměnné		*PGAs/* Definice uživatelských proměnných (DEF) [Strana 25]
PL	1. B-Spline: Vzdálenost uzlů 2. Polynomická interpolace: Délka intervalu parametru při polynomické interpolaci	s	*PGAs/* 1. Splinová interpolace (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [Strana 244] 2. Polynomická interpolace (POLY, POLYPATH, PO, PL) [Strana 261]
PM	za minutu		*PGs/*
PO	Koeficient polynomu při polynomické interpolaci	s	*PGAs/* Polynomická interpolace (POLY, POLYPATH, PO, PL) [Strana 261]
POCKET3	Technologický cyklus: Frézování pravoúhlé kapsy		*PGAs/* Frézování pravoúhlé kapsy - POCKET3 [Strana 763]
POCKET4	Technologický cyklus: Frézování kruhové kapsy		*PGAs/* Frézování kruhové kapsy - POCKET4 [Strana 766]
POLF	Poloha pro zpětný pohyb LIFTFAST	m	*PGs//PGAs/*
POLFA	Poloha pro zpětný pohyb jednotlivými osami se spouštěním pomocí parametru $AA_ESR_TRIGGER	m	*PGs/*
POLFMASK	Uvolnění os pro zpětný pohyb bez souvislostí mezi jednotlivými osami	m	*PGs/*
POLFMLIN	Uvolnění os pro zpětný pohyb s lineární souvislostí mezi jednotlivými osami	m	*PGs/*
POLY	Polynomická interpolace	m	*PGAs/* Polynomická interpolace (POLY, POLYPATH, PO, PL) [Strana 261]
POLYPATH	Polynomická interpolace může být vybrána pro skupiny os AXIS nebo VECT	m	*PGAs/* Polynomická interpolace (POLY, POLYPATH, PO, PL) [Strana 261]
PON	Zapnutí lisování	m	*PGAs/* Zapnutí nebo vypnutí lisování a prostřihování (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) [Strana 663]
PONS	Zapnutí lisování v taktu IPO	m	*PGAs/* Zapnutí nebo vypnutí lisování a prostřihování (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) [Strana 663]
POS	Polohování osy		*PGs/*
POSA	Polohování osy přes hranici bloku		*PGs/*
POSM	Polohování zásobníku		*FBW*

Příkaz	Význam	W [1]	Popis viz [2]
POSP	Polohování v úsecích (oscilace)		*PGs/*
POSRANGE	Zjištění, zda se požadovaná poloha osy, jejíž interpolace právě probíhá, nachází v okně okolo předem zadané referenční pozice.		*PGAs/* Pozice v předdefinované referenční oblasti (POSRANGE) [Strana 606]
POT	Druhá mocnina (aritmetická funkce)		*PGAs/* Matematické funkce [Strana 64]
PR	na otáčku		*PGs/*
PREPRO	Označení podprogramů s přípravou		*PGAs/* Označení podprogramů s přípravou (PREPRO) [Strana 180]
PRESETON	Dosazení skutečné hodnoty pro naprogramované osy		*PGAs/* Předvolba posunutí (PRESETON) [Strana 306]
PRIO	Klíčové slovo pro nastavení priority při spravování přerušení		*PGAs/* Přiřazování a spouštění rutin přerušení (SETINT, PRIO, BLSYNC) [Strana 122]
PROC	První příkaz programu		*PGAs/* Volání podprogramu s udáním cesty a parametrů (PCALL) [Strana 202]
PTP	Pohyb od bodu k bodu	m	*PGAs/* Posuv PTP v kartézských souřadnicích [Strana 386]
PTPG0	Pohyb od bodu k bodu jen při G0, jinak CP	m	*PGAs/* PTP u příkazu TRANSMIT [Strana 391]
PUNCHACC	Zrychlení při prostřihování závislé na dráze		*PGAs/* Zapnutí nebo vypnutí lisování a prostřihování (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) [Strana 663]
PUTFTOC	Jemná korekce nástroje pro paralelní orovnávání		*PGAs/* On-line korekce nástroje (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) [Strana 414]
PUTFTOCF	Jemná korekce nástroje v závislosti na funkci určené příkazem FCTDEF pro paralelní orovnávání		*PGAs/* On-line korekce nástroje (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) [Strana 414]
PW	B-Spline, váha uzlového bodu	s	*PGAs/* Splinová interpolace (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [Strana 244]
QECLRNOF	Deaktivování učení kompenzace chyby kvadrantu		*PGAs/* Zjišťování kompenzačních charakteristik (QECLRNON, QECLRNOF) [Strana 699]
QECLRNON	Aktivování učení kompenzace chyby kvadrantu		*PGAs/* Zjišťování kompenzačních charakteristik (QECLRNON, QECLRNOF) [Strana 699]

Příkaz	Význam	W [1]	Popis viz [2]
QU	Výstup rychlých doplňkových (pomocných) funkcí		*PGs*/
R...	Početní parametr, také jako nastavitelný adresový identifikátor a s numerickým rozšířením		*PGAs*/ Předdefinované uživatelské proměnné: Početní parametr (R) [Strana 21]
RAC	Absolutní blokové programování rádiusů pro specifickou osu	s	*PGs*/
RDISABLE	Zablokování načítání		*PGAs*/ Aktivování blokování načítání (RDISABLE) [Strana 585]
READ	Načtení jednoho nebo více řádků ze zadaného souboru a uložení načtených informací do pole		*PGAs*/ Čtení řádků v souboru (READ) [Strana 148]
REAL	Datový typ: Proměnná s plovoucí řádovou čárkou a se znaménkem (reálná čísla)		*PGAs*/ Definice uživatelských proměnných (DEF) [Strana 25]
REDEF	Nastavení skupin uživatelů, u kterých se zobrazují strojní parametry, prvky NC jazyka a systémové proměnné		*PGAs*/ Opětovná definice systémových proměnných, uživatelských proměnných a příkazů NC jazyka (REDEF) [Strana 31]
RELEASE	Odblokování os stroje za účelem výměny os		*PGAs*/ Výměna osy, výměna vřetena (RELEASE, GET, GETD) [Strana 132]
REP	Klíčové slovo pro inicializaci všech prvků pole se stejnou hodnotou		*PGAs*/ Definice a inicializace proměnných typu pole (DEF, SET, REP) [Strana 47]
REPEAT	Opakování programové smyčky		*PGAs*/ Opakování části programu (REPEAT, REPEATB, ENDLABEL, P) [Strana 99]
REPEATB	Opakování programového řádku		*PGAs*/ Opakování části programu (REPEAT, REPEATB, ENDLABEL, P) [Strana 99]
REPOSA	Zpětné najíždění na konturu lineárně všemi osami	s	*PGAs*/ Opětovné najíždění na konturu (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [Strana 484]
REPOSH	Najetí zpět na konturu po půlkruhu	s	*PGAs*/ Opětovné najíždění na konturu (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [Strana 484]
REPOSHA	Najetí zpět na konturu všemi osami po půlkruhu; geometrické osy po půlkruhu	s	*PGAs*/ Opětovné najíždění na konturu (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [Strana 484]

Příkaz	Význam	W [1]	Popis viz [2]
REPOSL	Najíždění na konturu po čtvrtkruhu	s	*PGAs*/ Opětovné najíždění na konturu (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [Strana 484]
REPOSQ	Najetí zpět na konturu po čtvrtkruhu	s	*PGAs*/ Opětovné najíždění na konturu (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [Strana 484]
REPOSQA	Najetí zpět na konturu všemi osami po čtvrtkruhu; geometrické osy po čtvrtkruhu	s	*PGAs*/ Opětovné najíždění na konturu (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [Strana 484]
RESET	Reset technologického cyklu		*PGAs*/ Blokování, uvolnění, reset (LOCK, UNLOCK, RESET) [Strana 642]
RESETMON	Příkaz jazyka pro aktivování požadované hodnoty		*FBW*
RET	Konec podprogramu		*PGAs*/ Návrat z podprogramu s nastavitelnými parametry (RET ...) [Strana 183]
RIC	Relativní blokové programování rádiusů pro specifickou osu	s	*PGs*/ Výměna osy, výměna vřetena (RELEASE, GET, GETD) [Strana 132]
RINDEX	Stanovení indexu znaku ve vstupním řetězci		*PGAs*/ Vyhledávání znaků/řetězců v řetězci (INDEX, RINDEX, MINDEX, MATCH) [Strana 81]
RMB	Zpětné najíždění na začátek bloku	m	*PGAs*/ Opětovné najíždění na konturu (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [Strana 484]
RME	Zpětné najíždění na konec bloku	m	*PGAs*/ Opětovné najíždění na konturu (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [Strana 484]
RMI [3]	Zpětné najíždění na místo přerušení	m	*PGAs*/ Opětovné najíždění na konturu (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [Strana 484]
RMN	Zpětné najíždění na nejbližší blok s bodem dráhy	m	*PGAs*/ Opětovné najíždění na konturu (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [Strana 484]
RND	Zaoblení rohů kontury	s	*PGs*/
RNDM	Modální zaoblení	m	*PGs*/

Příkaz	Význam	W [1]	Popis viz [2]
ROT	Programovatelné otočení	s	*PGsl*
ROTS	Programovatelná otáčení framu o prostorový úhel	s	*PGsl*
ROUND	Zaokrouhlení desetinných míst		*PGAsl* Matematické funkce [Strana 64]
ROUNDUP	Zaokrouhlování vstupní hodnoty		*PGAsl* Zaokrouhlení (ROUNDUP) [Strana 158]
RP	Polární rádius	m/s	*PGsl*
RPL	Rotace v rovině	s	*PGsl*
RT	Parametr pro přístup k datům framu: otočení		*PGAsl* Načítání a editace složek framu (TR, FI, RT, SC, MI) [Strana 299]
RTLIOF	G0 bez lineární interpolace (interpolace jednotlivých os)	m	*PGsl*
RTLION	G0 s lineární interpolací	m	*PGsl*

Příkaz	Význam	W [1]	Popis viz [2]
S	Otáčky vřetena (u G4, G96/G961 jiný význam)	m/s	*PGsl*
SAVE	Atribut pro záchranu informací při vyvolávání podprogramů		*PGAsl* Ukládání modálních G-funkcí (SAVE) [Strana 170]
SBLOF	Potlačení zpracovávání blok po bloku		*PGAsl* Potlačení zpracovávání blok po bloku (SBLOF, SBLON) [Strana 171]
SBLON	Odstranění blokování zpracovávání blok po bloku		*PGAsl* Potlačení zpracovávání blok po bloku (SBLOF, SBLON) [Strana 171]
SC	Parametr pro přístup k datům framu: změna měřítka		*PGAsl* Načítání a editace složek framu (TR, FI, RT, SC, MI) [Strana 299]
SCALE	Programovatelná změna měřítka	s	*PGsl*
SCC	Selektivní přiřazení příčné osy příkazu G96/G961/G962 Identifikátorem osy mohou být geometrická, kanálová nebo strojní osa.		*PGsl*
SCPARA	Programování bloku parametrů servomechanismu		*PGAsl* Programovatelný blok parametrů servomechnismu (SCPARA) [Strana 287]

Příkaz	Význam	W [1]	Popis viz [2]
SD	Stupeň splinu	s	*PGAsl* Splinová interpolace (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [Strana 244]
SEFORM	Strukturovací příkaz ve Stepeditoru, aby z něj bylo možné vygenerovat výpis kroků pro HMI Advanced.		*PGAsl* Strukturovací příkaz ve Stepeditoru (SEFORM) [Strana 225]
SET	Klíčové slovo pro inicializaci všech prvků pole se zadanými hodnotami		*PGAsl* Definice a inicializace proměnných typu pole (DEF, SET, REP) [Strana 47]
SETAL	Aktivování alarmu		*PGAsl* Alarmy (SETAL) [Strana 717]
SETDNO	Přiřazení D-čísla břitu (CE) daného nástroje (T)		*PGAsl* Volné zadávání D-čísel: Přejmenovávání D-čísel (GETDNO, SETDNO) [Strana 440]
SETINT	Stanovení, která rutina přerušení má být aktivována, když se aktivuje daný vstup NCK.		*PGAsl* Přiřazování a spouštění rutin přerušení (SETINT, PRIO, BLSYNC) [Strana 122]
SETM	Nastavování značek ve vlastním kanálu		PGAsl Koordinace programů (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) [Strana 115]
SETMS	Zpětné přepnutí na řídící vřeteno určené ve strojním parametru		
SETMS (n)	Vřeteno n má platit jako řídící vřeteno		*PGsl*
SETMTH	Definice čísla držáku hlavního nástroje		*FBW*
SETPIECE	Nastavení počtu kusů pro všechny nástroje, které jsou přiřazeny danému vřetenu.		*FBW*
SETTA	Aktivování nástroje ze skupiny opotřebení		*FBW*
SETTCOR	Změna komponent nástroje; při této změně jsou zohledňovány všechny okrajové podmínky		*FB1(W1)*
SETTIA	Deaktivování nástroje ze skupiny opotřebení		*FBW*
SF	Úhlové posunutí počátečního bodu pro řezání závitů	m	*PGsl*
SIN	Sinus (trigonometrická funkce)		*PGAsl* Matematické funkce [Strana 64]
SIRELAY	Aktivování bezpečnostních funkcí stanovených pomocí parametrů SIRELILN, SIRELOUT, SIRELTIME		*FBSIsl*
SIRELIN	Inicializace vstupních veličin funkčního modulu		*FBSIsl*

Příkaz	Význam	W [1]	Popis viz [2]
SIRELOUT	Inicializace výstupních veličin funkčního modulu		*FBSIs*
SIRELTIME	Inicializace časovače funkčního modulu		*FBSIs*
SLOT1	Technologický cyklus: Podélná drážka		*PGAs* Podélná drážka - SLOT1 [Strana 774]
SLOT2	Technologický cyklus: Kruhová drážka		*PGAs* Kruhová drážka - SLOT2 [Strana 777]
SOFT	Zrychlení po dráze s omezením ryvu	m	*PGs*
SOFTA	Aktivování změn zrychlení pro naprogramované osy s omezením trhavých pohybů		*PGs*
SON	Aktivování prostřihování	m	*PGAs* Zapnutí nebo vypnutí lisování a prostřihování (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) [Strana 663]
SONS	Zapnutí prostřihování v taktu IPO	m	*PGAs* Zapnutí nebo vypnutí lisování a prostřihování (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) [Strana 663]
SPATH [3]	Referenční dráha pro osy v FGROUP je délka oblouku	m	*PGAs* Nastavitelný vztah k dráze (SPATH, UPATH) [Strana 267]
SPCOF	Přepnutí řídícího vřetena nebo vřetena (n) z režimu polohové regulace do režimu regulace otáček	m	*PGs*
SPCON	Přepnutí řídícího vřetena nebo vřetena (n) z režimu regulace otáček do režimu regulace polohy	m	*PGAs*
SPI	Převedení čísla vřetena v identifikátoru osy		*PGAs* Osové funkce (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) [Strana 677]
SPIF1 [3]	Rychlé vstupy/výstupy NCK pro lisování/ prostřihování, byte 1	m	*FB2(N4)*
SPIF2	Rychlé vstupy/výstupy NCK pro lisování/ prostřihování, byte 2	m	*FB2(N4)*
SPLINEPATH	Definice pásma hodnot pro spliny		*PGAs* Splinový svazek (SPLINEPATH) [Strana 256]
SPN	Počet úseků na blok	s	*PGAs* Automatická příprava cesty [Strana 668]
SPOF [3]	Vypnutí zdvihu, vypnutí vystřihování/lisování	m	*PGAs* Zapnutí nebo vypnutí lisování a prostřihování (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) [Strana 663]

Příkaz	Význam	W [1]	Popis viz [2]
SPOS	Poloha vřetena	m	*PGs/*
SPOSA	Polohování vřetena přes hranice bloků	m	*PGs/*
SPP	Délka úseku	m	*PGAs/* Automatická příprava cesty [Strana 668]
SPRINT	Vrací zpět vstupní řetězec formátovaný		*PGAs/* Formátování řetězce (SPRINT) [Strana 84]
SQRT	Druhá odmocnina (aritmetická funkce) (square root)		*PGAs/* Matematické funkce [Strana 64]
SR	Zpětná dráha oscilačního pohybu pro synchronní akci	s	*PGs/*
SRA	Zpětná dráha oscilačního pohybu osy při externím vstupu pro synchronní akci	m	*PGs/*
ST	Doba vyjiskřování s oscilačním pohybem pro synchronní akci	s	*PGs/*
STA	Doba vyjiskřování s oscilačním pohybem osy pro synchronní akci	m	*PGs/*
START	Spuštění zvoleného programu současně ve více kanálech z momentálně zpracovávaného programu		*PGAs/* Koordinace programů (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) [Strana 115]
STARTFIFO [3]	Zpracovávat; souběžně s plněním paměti preprocesoru	m	*PGAs/* Zpracování programu s pamětí předběžného zpracování (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE) [Strana 473]
STAT	Poloha kloubu	s	*PGAs/* Posuv PTP v kartézských souřadnicích [Strana 386]
STOLF	Toleranční faktor G0	m	*PGAs/* Tolerance u pohybů s funkcí G0 (STOLF) [Strana 502]
STOPFIFO	Zastavení zpracování; plnění paměti preprocesoru, dokud není zjištěn příkaz STARTFIFO, naplnění paměti preprocesoru nebo konec programu	m	*PGAs/* Zpracování programu s pamětí předběžného zpracování (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE) [Strana 473]
STOPRE	Zastavení předběžného zpracování, dokud nejsou zpracovány všechny připravené bloky z hlavního zpracování programu.		*PGAs/* Zpracování programu s pamětí předběžného zpracování (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE) [Strana 473]
STOPREOF	Odblokování zastavení předběžného zpracování		*PGAs/* Zrušení zastavení předběžného zpracování (STOPREOF) [Strana 586]
STRING	Datový typ: Řetězec znaků		*PGAs/* Definice uživatelských proměnných (DEF) [Strana 25]

Příkaz	Význam	W [1]	Popis viz [2]
STRINGFELD	Vybírání jednotlivých znaků z naprogramovaného řetězcového pole		*PGAsl* Vybírání jednotlivých znaků (STRINGVAR, STRINGFELD) [Strana 83]
STRINGIS	Kontrola existujícího rozsahu jazyka NC systému a speciálně u tohoto příkazu odpovídajících názvů NC-cyklů, uživatelských proměnných, maker a názvů návěští, zda existují, zda jsou platné, definované nebo aktivní.		*PGAsl* Kontrola existujícího rozsahu NC jazyka (STRINGIS) [Strana 693]
STRINGVAR	Vybírání jednotlivých znaků z naprogramovaného řetězce		*PGAsl* Vybírání jednotlivých znaků (STRINGVAR, STRINGFELD) [Strana 83]
STRLEN	Zjištění délky řetězce		*PGAsl* Zjištění délky řetězce (STRLEN) [Strana 80]
SUBSTR	Stanovení indexu znaku ve vstupním řetězci		*PGAsl* Vybírání dílčího řetězce (SUBSTR) [Strana 82]
SUPA	Potlačení aktuálního posunutí počátku, včetně programovatelných posunutí, systémových framů, posunutí ručním kolečkem (DRF), externích posunutí a superponovaných pohybů	s	*PGsl*
SVC	Řezná rychlost nástroje	m	*PGsl*
SYNFCT	Vyhodnocování polynomu v závislosti na podmínce v pohybové synchronní akci		*PGAsl* Synchronní funkce (SYNFCT) [Strana 592]
SYNR	Načtení proměnné se uskutečňuje synchronně, tzn. v okamžiku zpracovávání		*PGAsl* Definice uživatelských proměnných (DEF) [Strana 25]
SYNRW	Načtení a zápis do proměnné se uskutečňuje synchronně, tzn. v okamžiku zpracovávání		*PGAsl* Definice uživatelských proměnných (DEF) [Strana 25]
SYNW	Zápis do proměnné se uskutečňuje synchronně, tzn. v okamžiku zpracovávání		*PGAsl* Definice uživatelských proměnných (DEF) [Strana 25]
T	Vyvolání nástroje (výměna jen tehdy, je-li nastaveno strojním parametrem, jinak je třeba příkaz M6)		*PGsl*
TAN	Tangens (trigonometrická funkce)		*PGAsl* Matematické funkce [Strana 64]
TANG	Definice vazby mezi osami na principu tangenciálního vlečení		*PGAsl* Tangenciální řízení (TANG, TANGON, TANGOF, TLIFT, TANGDEL) [Strana 461]
TANGDEL	Vymazání definice vazby mezi osami na principu tangenciálního vlečení		*PGAsl* Tangenciální řízení (TANG, TANGON, TANGOF, TLIFT, TANGDEL) [Strana 461]

Příkaz	Význam	W [1]	Popis viz [2]
TANGOF	Vypnutí tangenciálního vlečení		*PGAs*/ Tangenciální řízení (TANG, TANGON, TANGOF, TLIFT, TANGDEL) [Strana 461]
TANGON	Zapnutí tangenciálního vlečení		*PGAs*/ Tangenciální řízení (TANG, TANGON, TANGOF, TLIFT, TANGDEL) [Strana 461]
TCA (828D: _TCA)	Volba nástroje / výměna nástroje nezávisle na jeho stavu		*FBW*
TCARR	Vyžádání držáku nástroje s číslem [m]		*PGAs*/ Délková korekce nástroje pro orientovatelný držák nástroje (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) [Strana 449]
TCI	Výměna nástroje ze schránky do zásobníku		*FBW*
TCOABS [3]	Stanovení délkových složek nástroje z aktuální orientace nástroje	m	*PGAs*/ Délková korekce nástroje pro orientovatelný držák nástroje (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) [Strana 449]
TCOFR	Stanovení složek délky nástroje z orientace aktivního framu	m	*PGAs*/ Délková korekce nástroje pro orientovatelný držák nástroje (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) [Strana 449]
TCOFRX	Stanovení orientace nástroje aktivního framu při volbě tohoto nástroje, nástroj nasměrovaný v ose X	m	*PGAs*/ Délková korekce nástroje pro orientovatelný držák nástroje (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) [Strana 449]
TCOFRY	Stanovení orientace nástroje aktivního framu při volbě tohoto nástroje, nástroj nasměrovaný v ose Y	m	*PGAs*/ Délková korekce nástroje pro orientovatelný držák nástroje (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) [Strana 449]
TCOFRZ	Stanovení orientace nástroje aktivního framu při volbě tohoto nástroje, nástroj nasměrovaný v ose Z	m	*PGAs*/ Délková korekce nástroje pro orientovatelný držák nástroje (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) [Strana 449]
THETA	Úhel otočení	s	*PGAs*/ Otáčení orientace nástroje (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) [Strana 354]
TILT	Úhel bočního naklonění	m	*PGAs*/ Programování orientace nástroje (A..., B..., C..., LEAD, TILT) [Strana 333]
TLIFT	V případě tangenciálního řízení vložení pomocného bloku v rozích kontury		*PGAs*/ Tangenciální řízení (TANG, TANGON, TANGOF, TLIFT, TANGDEL) [Strana 461]
TMOF	Deaktivování monitorování nástroje		*PGAs*/ Specifické monitorování nástroje pro broušení ve výrobním programu (TMON, TMOF) [Strana 675]

Příkaz	Význam	W [1]	Popis viz [2]
TMON	Aktivování monitorování nástroje		*PGAs*/ Specifické monitorování nástroje pro broušení ve výrobním programu (TMON, TMOF) [Strana 675]
TO	Příkaz pro zadání koncové hodnoty ve smyčce FOR s počítadlem		*PGAs*/ Smyčka s počítadlem (FOR ... TO ..., ENDFOR) [Strana 110]
TOFF	Offset délky nástroje ve směru délkové složky nástroje, která se uplatňuje rovnoběžně s geometrickou osou zadanou v indexu.	m	*PGs*/
TOFFL	Offset délky nástroje ve směru délkové složky nástroje L1, L2, příp. L3.	m	*PGs*/
TOFFOF	Vynulování on-line korekce délky nástroje		*PGAs*/ On-line korekce délky nástroje (TOFFON, TOFFOF) [Strana 452]
TOFFON	Aktivování on-line korekce délky nástroje		*PGAs*/ On-line korekce délky nástroje (TOFFON, TOFFOF) [Strana 452]
TOFFR	Offset rádiusu nástroje	m	*PGs*/
TOFRAME	Nastavení osy Z systému WCS prostřednictvím otáčení framu rovnoběžně s orientací nástroje	m	*PGs*/
TOFRAMEX	Nastavení osy X systému WCS prostřednictvím otáčení framu rovnoběžně s orientací nástroje	m	*PGs*/
TOFRAMEY	Nastavení osy Y systému WCS prostřednictvím otáčení framu rovnoběžně s orientací nástroje	m	*PGs*/
TOFRAMEZ	Stejné jako příkaz TOFRAME	m	*PGs*/
TOLOWER	Přeměna všech písmen v řetězci na malá písmena		*PGAs*/ Převádění na malá/velká písmena (TOLOWER, TOUPPER) [Strana 79]
TOOLENV	Uložení do paměti všech aktuálních stavů, které jsou důležité pro vyhodnocování parametrů nástroje uchovávaných v paměti.		*FB1(W1)*
TOROT	Nastavení osy Z systému WCS prostřednictvím otáčení framu rovnoběžně s orientací nástroje	m	*PGs*/
TOROTOF	Zrušení otočení framu ve směru nástroje	m	*PGs*/
TOROTX	Nastavení osy X systému WCS prostřednictvím otáčení framu rovnoběžně s orientací nástroje	m	*PGs*/

Příkaz	Význam	W [1]	Popis viz [2]
TOROTY	Nastavení osy Y systému WCS prostřednictvím otáčení framu rovnoběžně s orientací nástroje	m	*/PGs/*
TOROTZ	Stejné jako příkaz TOROT	m	*/PGs/*
TOUPPER	Přeměna všech písmen v řetězci na velká písmena		*/PGAs/* Převádění na malá/velká písmena (TOLOWER, TOUPPER) [Strana 79]
TOWBCS	Hodnoty opotřebení v základním souřadném systému (BCS)	m	*/PGAs/* Souřadný systém aktivního obrábění (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) [Strana 410]
TOWKCS	Hodnoty opotřebení v souřadném systému hlavy nástroje při kinetické transformaci (liší se od MCS otočením nástroje)	m	*/PGAs/* Souřadný systém aktivního obrábění (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) [Strana 410]
TOWMCS	Hodnoty opotřebení v souřadném systému stroje (MCS)	m	*/PGAs/* Souřadný systém aktivního obrábění (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) [Strana 410]
TOWSTD	Základní nastavení pro korekce hodnoty ve směru délky nástroje	m	*/PGAs/* Souřadný systém aktivního obrábění (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) [Strana 410]
TOWTCS	Hodnoty opotřebení v souřadném systému nástroje (vztažný bod držáku nástroje T na nástrojovém sklíčidle)	m	*/PGAs/* Souřadný systém aktivního obrábění (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) [Strana 410]
TOWWCS	Hodnoty opotřebení v souřadném systému obrobku (WCS)	m	*/PGAs/* Souřadný systém aktivního obrábění (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) [Strana 410]
TR	Složka posunutí proměnné typu FRAME		*/PGAs/* Načítání a editace složek framu (TR, FI, RT, SC, MI) [Strana 299]
TRAANG	Transformace šikmé osy		*/PGAs/* Šikmá osa (TRAANG) [Strana 381]
TRACON	Kaskádová transformace		*/PGAs/* Zřetězené transformace (TRACON, TRAFOOF) [Strana 397]
TRACYL	Válec: Transformace plášťové plochy		*/PGAs/* Transformace válcového pláště (TRACYL) [Strana 373]
TRAFOOF	Vypnutí aktivních transformací v kanálu		*/PGAs/* Zřetězené transformace (TRACON, TRAFOOF) [Strana 397]

Příkaz	Význam	W [1]	Popis viz [2]
TRAILOF	Vypnutí asynchronního vlečení		*PGAs*/ Vlečení (TRAILON, TRAILOF) [Strana 505]
TRAILON	Zapnutí asynchronního vlečení		*PGAs*/ Vlečení (TRAILON, TRAILOF) [Strana 505]
TRANS	Programovatelné posunutí	s	*PGs*/
TRANSMIT	Polární transformace (obrábění šikmých ploch)		*PGAs*/ Frézovací práce na rotačních součástech (TRANSMIT) [Strana 369]
TRAORI	4-, 5-osá transformace, generická transformace		*PGAs*/ Transformace ve třech, čtyřech a pěti osách (TRAORI) [Strana 330]
TRUE	Logická konstanta: TRUE		*PGAs*/ Definice uživatelských proměnných (DEF) [Strana 25]
TRUNC	Odříznutí míst za desetinnou čárkou		*PGAs*/ Korekce přesnosti při příkazech porovnávání (TRUNC) [Strana 69]
TU	Úhel osy	s	*PGAs*/ Posuv PTP v kartézských souřadnicích [Strana 386]
TURN	Počet závitů u spirály	s	*PGs*/
ULI	Horní mezní hodnota pro proměnné		*PGAs*/ Atribut: Mezní hodnoty (LLI, ULI) [Strana 37]
UNLOCK	Odblokování synchronní akce s identifikací ID (pokračování technologického cyklu)		*PGAs*/ Blokování, uvolnění, reset (LOCK, UNLOCK, RESET) [Strana 642]
UNTIL	Podmínka pro ukončení smyčky REPEAT		*PGAs*/ Programová smyčka s podmínkou na začátku smyčky (WHILE, ENDWHILE) [Strana 112]
UPATH	Referenční dráha pro osy v FGROUP je křivkový parametr	m	*PGAs*/ Nastavitelný vztah k dráze (SPATH, UPATH) [Strana 267]
VAR	Klíčové slovo: Druh předávání parametrů		*PGAs*/ Volání podprogramu s předáváním parametrů (EXTERN) [Strana 192]
VELOLIM	Omezení maximální rychlosti osy	m	*PGAs*/ Procentuální korekce rychlosti (VELOLIM) [Strana 494]
VELOLIMA	Snížení nebo zvýšení maximální rychlosti vlečné osy	m	*PGs*/
WAITC	Čekání, až bude pro osu/vřeteno splněno kritérium pro změnu bloku vazby.		*PGAs*/ Koordinace programů (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) [Strana 115]

Příkaz	Význam	W [1]	Popis viz [2]
WAITE	Čekání na konec programu v jiném kanálu		*PGAs*/ Koordinace programů (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) [Strana 115]
WAITENC	Čekání na synchronizované, příp. restaurované polohy os		*PGAs*/ Čekání na platnou pozici osy (WAITENC) [Strana 691]
WAITM	Čekání na značku v uvedeném kanálu; konec předešlého bloku s přesným najetím.		*PGAs*/ Koordinace programů (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) [Strana 115]
WAITMC	Čekání na značku v uvedeném kanálu; přesné najetí jen tehdy, pokud ostatní kanály značky ještě nedosáhly		*PGAs*/ Koordinace programů (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) [Strana 115]
WAITP	Čekání na konec posuvu polohovací osy		*PGs*/
WAITS	Čekání na dosažení pozice vřetena		*PGs*/
WALCS0	Deaktivování ohraničení pracovního pole ve WCS	m	*PGs*/
WALCS1	Skupina 1 ohraničení pracovního pole ve WCS aktivní	m	*PGs*/
WALCS2	Skupina 2 ohraničení pracovního pole ve WCS aktivní	m	*PGs*/
WALCS3	Skupina 3 ohraničení pracovního pole ve WCS aktivní	m	*PGs*/
WALCS4	Skupina 4 ohraničení pracovního pole ve WCS aktivní	m	*PGs*/
WALCS5	Skupina 5 ohraničení pracovního pole ve WCS aktivní	m	*PGs*/
WALCS6	Skupina 6 ohraničení pracovního pole ve WCS aktivní	m	*PGs*/
WALCS7	Skupina 7 ohraničení pracovního pole ve WCS aktivní	m	*PGs*/
WALCS8	Skupina 8 ohraničení pracovního pole ve WCS aktivní	m	*PGs*/
WALCS9	Skupina 9 ohraničení pracovního pole ve WCS aktivní	m	*PGs*/
WALCS10	Skupina 10 ohraničení pracovního pole ve WCS aktivní	m	*PGs*/
WALIMOF	Vypnutí ohraničení pracovního pole BCS	m	*PGs*/
WALIMON [3]	Zapnutí ohraničení pracovního pole BCS	m	*PGs*/

Příkaz	Význam	W [1]	Popis viz [2]
WHEN	Pokud je podmínka splněna, bude se daná akce cyklicky provádět.		*PGAs*/ Cyklická kontrola podmínky (WHEN, WHENEVER, FROM, EVERY) [Strana 561]
WHENEVER	Pokud je podmínka jedenkrát splněna, daná akce se jedenkrát uskuteční.		*PGAs*/ Cyklická kontrola podmínky (WHEN, WHENEVER, FROM, EVERY) [Strana 561]
WHILE	Začátek programové smyčky WHILE		*PGAs*/ Programová smyčka s podmínkou na začátku smyčky (WHILE, ENDWHILE) [Strana 112]
WRITE	Zápis textu do systému souborů. Vložení bloku na konec zadaného souboru.		*PGAs*/ Zápis do souboru (WRITE) [Strana 140]
WRTPR	Aktivování zpoždění úlohy obrábění, aniž by však došlo k přerušení režimu řízení pohybu po dráze.		*PGAs*/
X	Název osy	m/s	*PGs*/
XOR	Logické XOR		*PGAs*/ Porovnávací a logické operace [Strana 67]
Y	Název osy	m/s	*PGs*/
Z	Název osy	m/s	*PGs*/

17.2 Příkazy: Použitelnost u systému SINUMERIK 828D

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
:	●	●	●	●	●	●
*	●	●	●	●	●	●
+	●	●	●	●	●	●
-	●	●	●	●	●	●
<	●	●	●	●	●	●
<<	●	●	●	●	●	●
<=	●	●	●	●	●	●
=	●	●	●	●	●	●
>=	●	●	●	●	●	●
/	●	●	●	●	●	●
/0 … … /7	● ○	● ○	● ○	● ○	● ○	● ○
A	●	●	●	●	●	●
A2	-	-	-	-	-	-
A3	-	-	-	-	-	-
A4	-	-	-	-	-	-
A5	-	-	-	-	-	-
ABS	●	●	●	●	●	●
AC	●	●	●	●	●	●
ACC	●	●	●	●	●	●
ACCLIMA	●	●	●	●	●	●
ACN	●	●	●	●	●	●
ACOS	●	●	●	●	●	●
ACP	●	●	●	●	●	●
ACTBLOCNO	●	●	●	●	●	●
ADDFRAME	●	●	●	●	●	●
ADIS	●	●	●	●	●	●
ADISPOS	●	●	●	●	●	●
ADISPOSA	●	●	●	●	●	●
ALF	●	●	●	●	●	●
AMIRROR	●	●	●	●	●	●
AND	●	●	●	●	●	●
ANG	●	●	●	●	●	●
AP	●	●	●	●	●	●
APR	●	●	●	●	●	●
APRB	●	●	●	●	●	●

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
APRP	●	●	●	●	●	●
APW	●	●	●	●	●	●
APWB	●	●	●	●	●	●
APWP	●	●	●	●	●	●
APX	●	●	●	●	●	●
AR	●	●	●	●	●	●
AROT	●	●	●	●	●	●
AROTS	●	●	●	●	●	●
AS	●	●	●	●	●	●
ASCALE	●	●	●	●	●	●
ASIN	●	●	●	●	●	●
ASPLINE	-	○	-	○	-	○
ATAN2	●	●	●	●	●	●
ATOL	-	●	-	●	-	●
ATRANS	●	●	●	●	●	●
AX	●	●	●	●	●	●
AXCTSWE	-	-	-	-	-	-
AXCTSWEC	-	-	-	-	-	-
AXCTSWED	-	-	-	-	-	-
AXIS	●	●	●	●	●	●
AXNAME	●	●	●	●	●	●
AXSTRING	●	●	●	●	●	●
AXTOCHAN	●	●	●	●	●	●
AXTOSPI	●	●	●	●	●	●
B	●	●	●	●	●	●
B2	-	-	-	-	-	-
B3	-	-	-	-	-	-
B4	-	-	-	-	-	-
B5	-	-	-	-	-	-
B_AND	●	●	●	●	●	●
B_OR	●	●	●	●	●	●
B_NOT	●	●	●	●	●	●
B_XOR	●	●	●	●	●	●
BAUTO	-	○	-	○	-	○
BLOCK	●	●	●	●	●	●
BLSYNC	●	●	●	●	●	●
BNAT	-	○	-	○	-	○
BOOL	●	●	●	●	●	●
BOUND	●	●	●	●	●	●

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
BRISK	●	●	●	●	●	●
BRISKA	●	●	●	●	●	●
BSPLINE	-	○	-	○	-	○
BTAN	-	○	-	○	-	○
C	●	●	●	●	●	●
C2	-	-	-	-	-	-
C3	-	-	-	-	-	-
C4	-	-	-	-	-	-
C5	-	-	-	-	-	-
CAC	●	●	●	●	●	●
CACN	●	●	●	●	●	●
CACP	●	●	●	●	●	●
CALCDAT	●	●	●	●	●	●
CALCPOSI	●	●	●	●	●	●
CALL	●	●	●	●	●	●
CALLPATH	●	●	●	●	●	●
CANCEL	●	●	●	●	●	●
CASE	●	●	●	●	●	●
CDC	●	●	●	●	●	●
CDOF	●	●	●	●	●	●
CDOF2	●	●	●	●	●	●
CDON	●	●	●	●	●	●
CFC	●	●	●	●	●	●
CFIN	●	●	●	●	●	●
CFINE	●	●	●	●	●	●
CFTCP	●	●	●	●	●	●
CHAN	●	●	●	●	●	●
CHANDATA	●	●	●	●	●	●
CHAR	●	●	●	●	●	●
CHECKSUM	●	●	●	●	●	●
CHF	●	●	●	●	●	●
CHKDM	●	●	●	●	●	●
CHKDNO	●	●	●	●	●	●
CHR	●	●	●	●	●	●
CIC	●	●	●	●	●	●
CIP	●	●	●	●	●	●
CLEARM	-	-	-	-	-	-
CLRINT	●	●	●	●	●	●
CMIRROR	●	●	●	●	●	●

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
COARSEA	●	●	●	●	●	●
COMPCAD	-	○	-	○	-	○
COMPCURV	-	○	-	○	-	○
COMPLETE	●	●	●	●	●	●
COMPOF	-	○	-	○	-	○
COMPON	-	○	-	○	-	○
CONTDCON	●	●	●	●	●	●
CONTPRON	●	●	●	●	●	●
CORROF	●	●	●	●	●	●
COS	●	●	●	●	●	●
COUPDEF	○	-	○	-	○	-
COUPDEL	○	-	○	-	○	-
COUPOF	○	-	○	-	○	-
COUPOFS	○	-	○	-	○	-
COUPON	○	-	○	-	○	-
COUPONC	○	-	○	-	○	-
COUPRES	○	-	○	-	○	-
CP	●	●	●	●	●	●
CPRECOF	●	●	●	●	●	●
CPRECON	●	●	●	●	●	●
CPROT	●	●	●	●	●	●
CPROTDEF	●	●	●	●	●	●
CR	●	●	●	●	●	●
CROT	●	●	●	●	●	●
CROTS	●	●	●	●	●	●
CRPL	●	●	●	●	●	●
CSCALE	●	●	●	●	●	●
CSPLINE	-	○	-	○	-	○
CT	●	●	●	●	●	●
CTAB	-	-	-	-	-	-
CTABDEF	-	-	-	-	-	-
CTABDEL	-	-	-	-	-	-
CTABEND	-	-	-	-	-	-
CTABEXISTS	-	-	-	-	-	-
CTABFNO	-	-	-	-	-	-
CTABFPOL	-	-	-	-	-	-
CTABFSEG	-	-	-	-	-	-
CTABID	-	-	-	-	-	-
CTABINV	-	-	-	-	-	-

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
CTABISLOCK	-	-	-	-	-	-
CTABLOCK	-	-	-	-	-	-
CTABMEMTYP	-	-	-	-	-	-
CTABMPOL	-	-	-	-	-	-
CTABMSEG	-	-	-	-	-	-
CTABNO	-	-	-	-	-	-
CTABNOMEM	-	-	-	-	-	-
CTABPERIOD	-	-	-	-	-	-
CTABPOL	-	-	-	-	-	-
CTABPOLID	-	-	-	-	-	-
CTABSEG	-	-	-	-	-	-
CTABSEGID	-	-	-	-	-	-
CTABSEV	-	-	-	-	-	-
CTABSSV	-	-	-	-	-	-
CTABTEP	-	-	-	-	-	-
CTABTEV	-	-	-	-	-	-
CTABTMAX	-	-	-	-	-	-
CTABTMIN	-	-	-	-	-	-
CTABTSP	-	-	-	-	-	-
CTABTSV	-	-	-	-	-	-
CTABUNLOCK	-	-	-	-	-	-
CTOL	-	○	-	○	-	○
CTRANS	●	●	●	●	●	●
CUT2D	●	●	●	●	●	●
CUT2DF	●	●	●	●	●	●
CUT3DC	-	-	-	-	-	-
CUT3DCC	-	-	-	-	-	-
CUT3DCCD	-	-	-	-	-	-
CUT3DF	-	-	-	-	-	-
CUT3DFF	-	-	-	-	-	-
CUT3DFS	-	-	-	-	-	-
CUTCONOF	●	●	●	●	●	●
CUTCONON	●	●	●	●	●	●
CUTMOD	●	●	●	●	●	●
CYCLE...	●	●	●	●	●	●
D	●	●	●	●	●	●
D0	●	●	●	●	●	●
DAC	●	●	●	●	●	●
DC	●	●	●	●	●	●

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
DEF	●	●	●	●	●	●
DEFINE	●	●	●	●	●	●
DEFAULT	●	●	●	●	●	●
DELAYFSTON	●	●	●	●	●	●
DELAYFSTOF	●	●	●	●	●	●
DELDL	●	●	●	●	●	●
DELDTG	●	●	●	●	●	●
DELETE	●	●	●	●	●	●
DELTOOLENV	●	●	●	●	●	●
DIACYCOFA	●	●	●	●	●	●
DIAM90	●	●	●	●	●	●
DIAM90A	●	●	●	●	●	●
DIAMCHAN	●	●	●	●	●	●
DIAMCHANA	●	●	●	●	●	●
DIAMCYCOF	●	●	●	●	●	●
DIAMOF	●	●	●	●	●	●
DIAMOFA	●	●	●	●	●	●
DIAMON	●	●	●	●	●	●
DIAMONA	●	●	●	●	●	●
DIC	●	●	●	●	●	●
DILF	●	●	●	●	●	●
DISABLE	●	●	●	●	●	●
DISC	●	●	●	●	●	●
DISCL	●	●	●	●	●	●
DISPLOF	●	●	●	●	●	●
DISPLON	●	●	●	●	●	●
DISPR	●	●	●	●	●	●
DISR	●	●	●	●	●	●
DITE	●	●	●	●	●	●
DITS	●	●	●	●	●	●
DIV	●	●	●	●	●	●
DL	-	-	-	-	-	-
DO	●	●	●	●	●	●
DRFOF	●	●	●	●	●	●
DRIVE	●	●	●	●	●	●
DRIVEA	●	●	●	●	●	●
DYNFINISH	●	●	●	●	●	●
DYNNORM	●	●	●	●	●	●
DYNPOS	●	●	●	●	●	●

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
DYNROUGH	●	●	●	●	●	●
DYNSEMIFIN	●	●	●	●	●	●
DZERO	●	●	●	●	●	●
EAUTO	-	○	-	○	-	○
EGDEF	-	-	-	-	-	-
EGDEL	-	-	-	-	-	-
EGOFC	-	-	-	-	-	-
EGOFS	-	-	-	-	-	-
EGON	-	-	-	-	-	-
EGONSYN	-	-	-	-	-	-
EGONSYNE	-	-	-	-	-	-
ELSE	●	●	●	●	●	●
ENABLE	●	●	●	●	●	●
ENAT	-	○	-	○	-	○
ENDFOR	●	●	●	●	●	●
ENDIF	●	●	●	●	●	●
ENDLABEL	●	●	●	●	●	●
ENDLOOP	●	●	●	●	●	●
ENDPROC	●	●	●	●	●	●
ENDWHILE	●	●	●	●	●	●
ESRR	●	●	●	●	●	●
ESRS	●	●	●	●	●	●
ETAN	-	○	-	○	-	○
EVERY	●	●	●	●	●	●
EX	●	●	●	●	●	●
EXECSTRING	●	●	●	●	●	●
EXECTAB	●	●	●	●	●	●
EXECUTE	●	●	●	●	●	●
EXP	●	●	●	●	●	●
EXTCALL	●	●	●	●	●	●
EXTCLOSE	●	●	●	●	●	●
EXTERN	●	●	●	●	●	●
EXTOPEN	●	●	●	●	●	●
F	●	●	●	●	●	●
FA	●	●	●	●	●	●
FAD	●	●	●	●	●	●
FALSE	●	●	●	●	●	●
FB	●	●	●	●	●	●
FCTDEF	-	-	-	-	-	-

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
FCUB	●	●	●	●	●	●
FD	●	●	●	●	●	●
FDA	●	●	●	●	●	●
FENDNORM	●	●	●	●	●	●
FFWOF	●	●	●	●	●	●
FFWON	●	●	●	●	●	●
FGREF	●	●	●	●	●	●
FGROUP	●	●	●	●	●	●
FI	●	●	●	●	●	●
FIFOCTRL	●	●	●	●	●	●
FILEDATE	●	●	●	●	●	●
FILEINFO	●	●	●	●	●	●
FILESIZE	●	●	●	●	●	●
FILESTAT	●	●	●	●	●	●
FILETIME	●	●	●	●	●	●
FINEA	●	●	●	●	●	●
FL	●	●	●	●	●	●
FLIN	●	●	●	●	●	●
FMA	-	-	-	-	-	-
FNORM	●	●	●	●	●	●
FOCOF	○	-	○	-	○	-
FOCON	○	-	○	-	○	-
FOR	●	●	●	●	●	●
FP	●	●	●	●	●	●
FPO	-	-	-	-	-	-
FPR	●	●	●	●	●	●
FPRAOF	●	●	●	●	●	●
FPRAON	●	●	●	●	●	●
FRAME	●	●	●	●	●	●
FRC	●	●	●	●	●	●
FRCM	●	●	●	●	●	●
FROM	●	●	●	●	●	●
FTOC	●	●	●	●	●	●
FTOCOF	●	●	●	●	●	●
FTOCON	●	●	●	●	●	●
FXS	●	●	●	●	●	●
FXST	●	●	●	●	●	●
FXSW	●	●	●	●	●	●
FZ	●	●	●	●	●	●

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
G0	●	●	●	●	●	●
G1	●	●	●	●	●	●
G2	●	●	●	●	●	●
G3	●	●	●	●	●	●
G4	●	●	●	●	●	●
G5	●	●	●	●	●	●
G7	●	●	●	●	●	●
G9	●	●	●	●	●	●
G17	●	●	●	●	●	●
G18	●	●	●	●	●	●
G19	●	●	●	●	●	●
G25	●	●	●	●	●	●
G26	●	●	●	●	●	●
G33	●	●	●	●	●	●
G34	●	●	●	●	●	●
G35	●	●	●	●	●	●
G40	●	●	●	●	●	●
G41	●	●	●	●	●	●
G42	●	●	●	●	●	●
G53	●	●	●	●	●	●
G54	●	●	●	●	●	●
G55	●	●	●	●	●	●
G56	●	●	●	●	●	●
G57	●	●	●	●	●	●
G58	●	●	●	●	●	●
G59	●	●	●	●	●	●
G60	●	●	●	●	●	●
G62	●	●	●	●	●	●
G63	●	●	●	●	●	●
G64	●	●	●	●	●	●
G70	●	●	●	●	●	●
G71	●	●	●	●	●	●
G74	●	●	●	●	●	●
G75	●	●	●	●	●	●
G90	●	●	●	●	●	●
G91	●	●	●	●	●	●
G93	●	●	●	●	●	●
G94	●	●	●	●	●	●
G95	●	●	●	●	●	●

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
G96	●	●	●	●	●	●
G97	●	●	●	●	●	●
G110	●	●	●	●	●	●
G111	●	●	●	●	●	●
G112	●	●	●	●	●	●
G140	●	●	●	●	●	●
G141	●	●	●	●	●	●
G142	●	●	●	●	●	●
G143	●	●	●	●	●	●
G147	●	●	●	●	●	●
G148	●	●	●	●	●	●
G153	●	●	●	●	●	●
G247	●	●	●	●	●	●
G248	●	●	●	●	●	●
G290	●	●	●	●	●	●
G291	●	●	●	●	●	●
G331	●	●	●	●	●	●
G332	●	●	●	●	●	●
G340	●	●	●	●	●	●
G341	●	●	●	●	●	●
G347	●	●	●	●	●	●
G348	●	●	●	●	●	●
G450	●	●	●	●	●	●
G451	●	●	●	●	●	●
G460	●	●	●	●	●	●
G461	●	●	●	●	●	●
G462	●	●	●	●	●	●
G500	●	●	●	●	●	●
G505 ... G599	●	●	●	●	●	●
G601	●	●	●	●	●	●
G602	●	●	●	●	●	●
G603	●	●	●	●	●	●
G621	●	●	●	●	●	●
G641	●	●	●	●	●	●
G642	●	●	●	●	●	●
G643	●	●	●	●	●	●
G644	●	●	●	●	●	●
G645	●	●	●	●	●	●
G700	●	●	●	●	●	●

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
G710	●	●	●	●	●	●
G751	●	●	●	●	●	●
G810 ... G819	-	-	-	-	-	-
G820 ... G829	-	-	-	-	-	-
G931	●	●	●	●	●	●
G942	●	●	●	●	●	●
G952	●	●	●	●	●	●
G961	●	●	●	●	●	●
G962	●	●	●	●	●	●
G971	●	●	●	●	●	●
G972	●	●	●	●	●	●
G973	●	●	●	●	●	●
GEOAX	●	●	●	●	●	●
GET	●	●	●	●	●	●
GETACTT	●	●	●	●	●	●
GETACTTD	●	●	●	●	●	●
GETD	●	●	●	●	●	●
GETDNO	●	●	●	●	●	●
GETEXET	●	●	●	●	●	●
GETFREELOC	●	●	●	●	●	●
GETSELT	●	●	●	●	●	●
GETT	●	●	●	●	●	●
GETTCOR	●	●	●	●	●	●
GETTENV	●	●	●	●	●	●
GOTO	●	●	●	●	●	●
GOTOB	●	●	●	●	●	●
GOTOC	●	●	●	●	●	●
GOTOF	●	●	●	●	●	●
GOTOS	●	●	●	●	●	●
GP	●	●	●	●	●	●
GWPSOF	●	●	●	●	●	●
GWPSON	●	●	●	●	●	●
H...	●	●	●	●	●	●
HOLES1	●	●	●	●	●	●
HOLES2	●	●	●	●	●	●
I	●	●	●	●	●	●
I1	●	●	●	●	●	●
IC	●	●	●	●	●	●
ICYCOF	●	●	●	●	●	●

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
ICYCON	●	●	●	●	●	●
ID	●	●	●	●	●	●
IDS	●	●	●	●	●	●
IF	●	●	●	●	●	●
INDEX	●	●	●	●	●	●
INIPO	●	●	●	●	●	●
INIRE	●	●	●	●	●	●
INICF	●	●	●	●	●	●
INIT	-	-	-	-	-	-
INITIAL	●	●	●	●	●	●
INT	●	●	●	●	●	●
INTERSEC	●	●	●	●	●	●
INVCCW	-	-	-	-	-	-
INVCW	-	-	-	-	-	-
INVFRAME	●	●	●	●	●	●
IP	●	●	●	●	●	●
IPOBRKA	●	●	●	●	●	●
IPOENDA	●	●	●	●	●	●
IPTRLOCK	●	●	●	●	●	●
IPTRUNLOCK	●	●	●	●	●	●
ISAXIS	●	●	●	●	●	●
ISD	-	-	-	-	-	-
ISFILE	●	●	●	●	●	●
ISNUMBER	●	●	●	●	●	●
ISOCALL	●	●	●	●	●	●
ISVAR	●	●	●	●	●	●
J	●	●	●	●	●	●
J1	●	●	●	●	●	●
JERKA	●	●	●	●	●	●
JERKLIM	●	●	●	●	●	●
JERKLIMA	●	●	●	●	●	●
K	●	●	●	●	●	●
K1	●	●	●	●	●	●
KONT	●	●	●	●	●	●
KONTC	●	●	●	●	●	●
KONTT	●	●	●	●	●	●
L	●	●	●	●	●	●

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
LEAD Orientace nástroje Polynom orientace	- -	- -	- -	- -	- -	- -
LEADOF	-	-	-	-	-	-
LEADON	-	-	-	-	-	-
LENTOAX	●	●	●	●	●	●
LFOF	●	●	●	●	●	●
LFON	●	●	●	●	●	●
LFPOS	●	●	●	●	●	●
LFTXT	●	●	●	●	●	●
LFWP	●	●	●	●	●	●
LIFTFAST	●	●	●	●	●	●
LIMS	●	●	●	●	●	●
LLI	●	●	●	●	●	●
LN	●	●	●	●	●	●
LOCK	●	●	●	●	●	●
LONGHOLE	-	-	-	-	-	-
LOOP	●	●	●	●	●	●
M0	●	●	●	●	●	●
M1	●	●	●	●	●	●
M2	●	●	●	●	●	●
M3	●	●	●	●	●	●
M4	●	●	●	●	●	●
M5	●	●	●	●	●	●
M6	●	●	●	●	●	●
M17	●	●	●	●	●	●
M19	●	●	●	●	●	●
M30	●	●	●	●	●	●
M40	●	●	●	●	●	●
M41 ... M45	●	●	●	●	●	●
M70	●	●	●	●	●	●
MASLDEF	●	●	●	●	●	●
MASLDEL	●	●	●	●	●	●
MASLOF	●	●	●	●	●	●
MASLOFS	●	●	●	●	●	●
MASLON	●	●	●	●	●	●
MATCH	●	●	●	●	●	●
MAXVAL	●	●	●	●	●	●
MCALL	●	●	●	●	●	●

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
MEAC	-	-	-	-	-	-
MEAFRAME	●	●	●	●	●	●
MEAS	●	●	●	●	●	●
MEASA	-	-	-	-	-	-
MEASURE	●	●	●	●	●	●
MEAW	●	●	●	●	●	●
MEAWA	-	-	-	-	-	-
MI	●	●	●	●	●	●
MINDEX	●	●	●	●	●	●
MINVAL	●	●	●	●	●	●
MIRROR	●	●	●	●	●	●
MMC	●	●	●	●	●	●
MOD	●	●	●	●	●	●
MODAXVAL	●	●	●	●	●	●
MOV	●	●	●	●	●	●
MSG	●	●	●	●	●	●
MVTOOL	●	●	●	●	●	●
N	●	●	●	●	●	●
NCK	●	●	●	●	●	●
NEWCONF	●	●	●	●	●	●
NEWT	●	●	●	●	●	●
NORM	●	●	●	●	●	●
NOT	●	●	●	●	●	●
NPROT	●	●	●	●	●	●
NPROTDEF	●	●	●	●	●	●
NUMBER	●	●	●	●	●	●
OEMIPO1	-	-	-	-	-	-
OEMIPO2	-	-	-	-	-	-
OF	●	●	●	●	●	●
OFFN	●	●	●	●	●	●
OMA1	-	-	-	-	-	-
OMA2	-	-	-	-	-	-
OMA3	-	-	-	-	-	-
OMA4	-	-	-	-	-	-
OMA5	-	-	-	-	-	-
OR	●	●	●	●	●	●
ORIAXES	-	-	-	-	-	-
ORIAXPOS	-	-	-	-	-	-
ORIC	-	-	-	-	-	-

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
ORICONCCW	-	-	-	-	-	-
ORICONCW	-	-	-	-	-	-
ORICONIO	-	-	-	-	-	-
ORICONTO	-	-	-	-	-	-
ORICURVE	-	-	-	-	-	-
ORID	-	-	-	-	-	-
ORIEULER	-	-	-	-	-	-
ORIMKS	-	-	-	-	-	-
ORIPATH	-	-	-	-	-	-
ORIPATHS	-	-	-	-	-	-
ORIPLANE	-	-	-	-	-	-
ORIRESET	-	-	-	-	-	-
ORIROTA	-	-	-	-	-	-
ORIROTC	-	-	-	-	-	-
ORIROTR	-	-	-	-	-	-
ORIROTT	-	-	-	-	-	-
ORIRPY	-	-	-	-	-	-
ORIRPY2	-	-	-	-	-	-
ORIS	-	-	-	-	-	-
ORISOF	-	-	-	-	-	-
ORISON	-	-	-	-	-	-
ORIVECT	-	-	-	-	-	-
ORIVIRT1	-	-	-	-	-	-
ORIVIRT2	-	-	-	-	-	-
ORIWKS	-	-	-	-	-	-
OS	-	-	-	-	-	-
OSB	-	-	-	-	-	-
OSC	-	-	-	-	-	-
OSCILL	-	-	-	-	-	-
OSCTRL	-	-	-	-	-	-
OSD	-	-	-	-	-	-
OSE	-	-	-	-	-	-
OSNSC	-	-	-	-	-	-
OSOF	-	-	-	-	-	-
OSP1	-	-	-	-	-	-
OSP2	-	-	-	-	-	-
OSS	-	-	-	-	-	-
OSSE	-	-	-	-	-	-
OST	-	-	-	-	-	-

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
OST1	-	-	-	-	-	-
OST2	-	-	-	-	-	-
OTOL	-	●	-	●	-	●
OVR	●	●	●	●	●	●
OVRA	●	●	●	●	●	●
OVRRAP	●	●	●	●	●	●
P	●	●	●	●	●	●
PAROT	●	●	●	●	●	●
PAROTOF	●	●	●	●	●	●
PCALL	●	●	●	●	●	●
PDELAYOF	-	-	-	-	-	-
PDELAYON	-	-	-	-	-	-
PHU	●	●	●	●	●	●
PL	- -	○ -	- -	○ -	- -	○ -
PM	●	●	●	●	●	●
PO	-	-	-	-	-	-
POCKET3	●	●	●	●	●	●
POCKET4	●	●	●	●	●	●
POLF	●	●	●	●	●	●
POLFA	●	●	●	●	●	●
POLFMASK	●	●	●	●	●	●
POLFMLIN	●	●	●	●	●	●
POLY	-	-	-	-	-	-
POLYPATH	-	-	-	-	-	-
PON	-	-	-	-	-	-
PONS	-	-	-	-	-	-
POS	●	●	●	●	●	●
POSA	●	●	●	●	●	●
POSM	●	●	●	●	●	●
POSP	●	●	●	●	●	●
POSRANGE	●	●	●	●	●	●
POT	●	●	●	●	●	●
PR	●	●	●	●	●	●
PREPRO	●	●	●	●	●	●
PRESETON	●	●	●	●	●	●
PRIO	●	●	●	●	●	●
PROC	●	●	●	●	●	●

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
PTP	●	●	●	●	●	●
PTPG0	●	●	●	●	●	●
PUNCHACC	-	-	-	-	-	-
PUTFTOC	●	●	●	●	●	●
PUTFTOCF	●	●	●	●	●	●
PW	-	○	-	○	-	○
QECLRNOF	●	●	●	●	●	●
QECLRNON	●	●	●	●	●	●
QU	●	●	●	●	●	●
R...	●	●	●	●	●	●
RAC	●	●	●	●	●	●
RDISABLE	●	●	●	●	●	●
READ	●	●	●	●	●	●
REAL	●	●	●	●	●	●
REDEF	●	●	●	●	●	●
RELEASE	●	●	●	●	●	●
REP	●	●	●	●	●	●
REPEAT	●	●	●	●	●	●
REPEATB	●	●	●	●	●	●
REPOSA	●	●	●	●	●	●
REPOSH	●	●	●	●	●	●
REPOSHA	●	●	●	●	●	●
REPOSL	●	●	●	●	●	●
REPOSQ	●	●	●	●	●	●
REPOSQA	●	●	●	●	●	●
RESET	●	●	●	●	●	●
RESETMON	●	●	●	●	●	●
RET	●	●	●	●	●	●
RIC	●	●	●	●	●	●
RINDEX	●	●	●	●	●	●
RMB	●	●	●	●	●	●
RME	●	●	●	●	●	●
RMI	●	●	●	●	●	●
RMN	●	●	●	●	●	●
RND	●	●	●	●	●	●
RNDM	●	●	●	●	●	●
ROT	●	●	●	●	●	●
ROTS	●	●	●	●	●	●
ROUND	●	●	●	●	●	●

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
ROUNDUP	●	●	●	●	●	●
RP	●	●	●	●	●	●
RPL	●	●	●	●	●	●
RT	●	●	●	●	●	●
RTLIOF	●	●	●	●	●	●
RTLION	●	●	●	●	●	●
S	●	●	●	●	●	●
SAVE	●	●	●	●	●	●
SBLOF	●	●	●	●	●	●
SBLON	●	●	●	●	●	●
SC	●	●	●	●	●	●
SCALE	●	●	●	●	●	●
SCC	●	●	●	●	●	●
SCPARA	●	●	●	●	●	●
SD	-	○	-	○	-	○
SEFORM	●	●	●	●	●	●
SET	●	●	●	●	●	●
SETAL	●	●	●	●	●	●
SETDNO	●	●	●	●	●	●
SETINT	●	●	●	●	●	●
SETM	-	-	-	-	-	-
SETMS	●	●	●	●	●	●
SETMS (n)	●	●	●	●	●	●
SETMTH	●	●	●	●	●	●
SETPIECE	●	●	●	●	●	●
SETTA	●	●	●	●	●	●
SETTCOR	●	●	●	●	●	●
SETTIA	●	●	●	●	●	●
SF	●	●	●	●	●	●
SIN	●	●	●	●	●	●
SIRELAY	-	-	-	-	-	-
SIRELIN	-	-	-	-	-	-
SIRELOUT	-	-	-	-	-	-
SIRELTIME	-	-	-	-	-	-
SLOT1	●	●	●	●	●	●
SLOT2	●	●	●	●	●	●
SOFT	●	●	●	●	●	●
SOFTA	●	●	●	●	●	●
SON	-	-	-	-	-	-

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
SONS	-	-	-	-	-	-
SPATH	●	●	●	●	●	●
SPCOF	●	●	●	●	●	●
SPCON	●	●	●	●	●	●
SPI	●	●	●	●	●	●
SPIF1	-	-	-	-	-	-
SPIF2	-	-	-	-	-	-
SPLINEPATH	-	○	-	○	-	○
SPN	-	-	-	-	-	-
SPOF	-	-	-	-	-	-
SPOS	●	●	●	●	●	●
SPOSA	●	●	●	●	●	●
SPP	-	-	-	-	-	-
SPRINT	●	●	●	●	●	●
SQRT	●	●	●	●	●	●
SR	-	-	-	-	-	-
SRA	-	-	-	-	-	-
ST	-	-	-	-	-	-
STA	-	-	-	-	-	-
START	-	-	-	-	-	-
STARTFIFO	●	●	●	●	●	●
STAT	●	●	●	●	●	●
STOLF	-	-	-	-	-	-
STOPFIFO	●	●	●	●	●	●
STOPRE	●	●	●	●	●	●
STOPREOF	●	●	●	●	●	●
STRING	●	●	●	●	●	●
STRINGFELD	●	●	●	●	●	●
STRINGIS	●	●	●	●	●	●
STRINGVAR	-	-	-	-	-	-
STRLEN	●	●	●	●	●	●
SUBSTR	●	●	●	●	●	●
SUPA	●	●	●	●	●	●
SVC	●	●	●	●	●	●
SYNFCT	●	●	●	●	●	●
SYNR	●	●	●	●	●	●
SYNRW	●	●	●	●	●	●
SYNW	●	●	●	●	●	●
T	●	●	●	●	●	●

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
TAN	●	●	●	●	●	●
TANG	-	-	-	-	-	-
TANGDEL	-	-	-	-	-	-
TANGOF	-	-	-	-	-	-
TANGON	-	-	-	-	-	-
TCA (828D: _TCA)	●	●	●	●	●	●
TCARR	-	●	-	●	-	●
TCI	●	●	●	●	●	●
TCOABS	-	●	-	●	-	●
TCOFR	-	●	-	●	-	●
TCOFRX	-	●	-	●	-	●
TCOFRY	-	●	-	●	-	●
TCOFRZ	-	●	-	●	-	●
THETA	-	-	-	-	-	-
TILT	-	-	-	-	-	-
TLIFT	-	-	-	-	-	-
TMOF	●	●	●	●	●	●
TMON	●	●	●	●	●	●
TO	●	●	●	●	●	●
TOFF	●	●	●	●	●	●
TOFFL	●	●	●	●	●	●
TOFFOF	●	●	●	●	●	●
TOFFON	●	●	●	●	●	●
TOFFR	●	●	●	●	●	●
TOFRAME	●	●	●	●	●	●
TOFRAMEX	●	●	●	●	●	●
TOFRAMEY	●	●	●	●	●	●
TOFRAMEZ	●	●	●	●	●	●
TOLOWER	●	●	●	●	●	●
TOOLENV	●	●	●	●	●	●
TOROT	●	●	●	●	●	●
TOROTOF	●	●	●	●	●	●
TOROTX	●	●	●	●	●	●
TOROTY	●	●	●	●	●	●
TOROTZ	●	●	●	●	●	●
TOUPPER	●	●	●	●	●	●
TOWBCS	-	●	-	●	-	●
TOWKCS	-	●	-	●	-	●

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
TOWMCS	-	●	-	●	-	●
TOWSTD	-	●	-	●	-	●
TOWTCS	-	●	-	●	-	●
TOWWCS	-	●	-	●	-	●
TR	●	●	●	●	●	●
TRAANG	-	-	-	-	○	-
TRACON	-	-	-	-	○	-
TRACYL	○	○	○	○	○	○
TRAFOOF	●	●	●	●	●	●
TRAILOF	●	●	●	●	●	●
TRAILON	●	●	●	●	●	●
TRANS	●	●	●	●	●	●
TRANSMIT	○	○	○	○	○	○
TRAORI	-	●	-	●	-	●
TRUE	●	●	●	●	●	●
TRUNC	●	●	●	●	●	●
TU	●	●	●	●	●	●
TURN	●	●	●	●	●	●
ULI	●	●	●	●	●	●
UNLOCK	●	●	●	●	●	●
UNTIL	●	●	●	●	●	●
UPATH	●	●	●	●	●	●
VAR	●	●	●	●	●	●
VELOLIM	●	●	●	●	●	●
VELOLIMA	●	●	●	●	●	●
WAITC	-	-	-	-	○	-
WAITE	-	-	-	-	-	-
WAITENC	-	-	-	-	-	-
WAITM	-	-	-	-	-	-
WAITMC	-	-	-	-	-	-
WAITP	●	●	●	●	●	●
WAITS	●	●	●	●	●	●
WALCS0	●	●	●	●	●	●
WALCS1	●	●	●	●	●	●
WALCS2	●	●	●	●	●	●
WALCS3	●	●	●	●	●	●
WALCS4	●	●	●	●	●	●
WALCS5	●	●	●	●	●	●
WALCS6	●	●	●	●	●	●

Příkaz	Varianta řídícího systému 828D					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Soustružení	Frézování	Soustružení	Frézování
WALCS7	●	●	●	●	●	●
WALCS8	●	●	●	●	●	●
WALCS9	●	●	●	●	●	●
WALCS10	●	●	●	●	●	●
WALIMOF	●	●	●	●	●	●
WALIMON	●	●	●	●	●	●
WHEN	●	●	●	●	●	●
WHENEVER	●	●	●	●	●	●
WHILE	●	●	●	●	●	●
WRITE	●	●	●	●	●	●
WRTPR	●	●	●	●	●	●
X	●	●	●	●	●	●
XOR	●	●	●	●	●	●
Y	●	●	●	●	●	●
Z	●	●	●	●	●	●

- ● Standardní
- ○ Volitelný doplněk
- \- Není k dispozici

17.3 Aktuální jazyk v HMI

Následující tabulka obsahuje všechny jazyky, které jsou na uživatelském rozhraní k dispozici.

Momentálně nastavený jazyk je možné ve výrobním programu a v synchronních akcích zjistit pomocí následující systémové proměnné:

$AN_LANGUAGE_ON_HMI = <hodnota>

<hodnota>	Jazyk	Zkratka jazyka
1	Němčina (Německo)	DEU
2	Francouzština	FRA
3	Angličtina (Spojené království)	ENG
4	Španělština	ESP
6	Italština	ITA
7	Holandština	NLD
8	Čínština (zjednodušená)	CHS
9	Švédština	SVE
18	Maďarština	HUN
19	Finština	FIN
28	Čeština	CSY
50	Portugalština (Brazílie)	PTB
53	Polština	PLK
55	Dánština	DAN
57	Ruština	RUS
68	Slovenština	SKY
72	Rumunština	ROM
80	Čínština (tradiční)	CHT
85	Korejština	KOR
87	Japonština	JPN
89	Turečtina	TRK

Poznámka

Aktualizace proměnné $AN_LANGUAGE_ON_HMI se uskutečňuje:

- po náběhu systému
- po resetu NCK a/nebo PLC
- po přepnutí na jinou jednotku NCK v rámci M2N
- po přepnutí jazyka na HMI

Přílohy

A

A.1 Seznam zkratek

A	Výstup
AS	Automatizační systém
ASCII	American Standard Code for Information Interchange: Americká norma pro kódy při výměně informací.
ASIC	Application Specific Integrated Circuit: Integrovaný obvod pro specifickou aplikaci
ASUP	Asynchronní podprogram
AV	Pro pokročilé
AWL	Seznam příkazů
BA	Provozní režim
BAG	Skupiny provozních režimů
BB	Prov. připravenost
BCD	Binary Coded Decimals: Desítková čísla vyjádřená v binárním kódu.
BCS	Základní souřadný systém
BHG	Jednotka manuálního ovládání
BIN	Binární data (**bin**ární soubory)
BIOS	Basic Input Output System
BOF	Uživatelské rozhraní
BT	Ovládací panel
BTSS	Rozhraní ovládacího panelu
BuB, B&B	Obsluha a monitorování
CAD	Computer-Aided Design (konstrukce s podporou počítače)
CAM	Výroba s podporou počítače
CNC	Computerized Numerical Control: Numerické řízení s počítačovou podporou
COM	Komunikace
CP	Komunikační procesor
CPU	Central Processing Unit: Centrální procesorová jednotka
CR	Carriage Return
CRT	Cathode Ray Tube: obrazovka
CSB	Central Service Board: modul PLC
CTS	Clear To Send: Hlášení o připravenosti k odesílání u sériových datových rozhraní.
CUTCOM	Cutter radius compensation: Korekce rádiusu nástroje
DAU	Digitálně-analogový převodník
DB	Datový modul v PLC
DBB	Byte datového modulu v PLC
DBW	Slovo datového modulu v PLC
DBX	Bit datového modulu v PLC

DC	Direct Control: Pohyb kruhové osy po nejkratší dráze na absolutní pozici v rámci jedné otáčky.
DCD	Carrier Detect
DDE	Dynamic Data Exchange (dynamická výměna dat)
DEE	Datové zařízení
DIN	Deutsche Industrie Norm (Německá průmyslová norma)
DIO	Data Input/Output: Obrazovka s informacemi o přenosu dat
DIR	Directory: Adresář
DLL	Dynamic Link Library (dynamická knihovna)
DOE	Zařízení pro přenos dat
DOS	Diskový operační systém
DPM	Paměť se dvěma porty
DPR	Paměť RAM se dvěma porty
DRAM	Dynamic Random Access Memory (dynamická paměť RAM)
DRF	Differential Resolver Function: Funkce diferenčního otočného snímače (ruční kolečko)
DRY	Dry Run: Posuv při zkušebním zpracování
DSB	Decoding Single Block: dekódování blok po bloku
DW	Datové slovo
E	Vstup
E/A	Vstupy / výstupy
ENC	Encoder: snímač skutečné polohy
EPROM	Erasable Programmable Read Only Memory (Mazatelná elektricky programovatelná paměť jen pro čtení)
ERROR	Chyba z tiskárny
FB	Funkční modul
FBS	Plochá obrazovka
FC	Function Call: Modul funkcí v PLC
FDB	Databáze produktu
FDD	Disketová jednotka
FEPROM	Flash-EPROM: paměť s možností čtení a zápisu
FIFO	First In First Out: Paměť, která pracuje bez zadávání adres. Data, která jsou do ní uložená, jsou čtena ve stejné posloupnosti, v jaké byla uložena.
FIPO	Jemný interpolátor
FM	Funkční modul
FPU	Floating Point Unit: jednotka pracující v plovoucí řádové čárce
FRA	Modul framu
FRAME	Datový blok (frame)
FRK	Korekce rádiusu frézy
FST	Feed Stop: Zastavení posuvu
FUP	Funkční schéma (metoda programování pro PLC)
GP	Základní program
GUD	Global User Data: Globální uživatelská data

HD	Hard Disk: Pevný disk
HEX	Zkratka pro hexadecimální formát
HiFu	Pomocná funkce
HMI	Human Machine Interface: Uživatelské rozhraní systému SINUMERIK pro obsluhu, programování a simulaci.
HMS	Měřicí systém s vysokým rozlišením
HSA	Pohon hlavního vřetena
HW	Hardware
IBN	Uvádění do provozu
IF	Uvolnění impulzu pro modul pohonu
IK (GD)	Implicitní komunikace (globální data)
IKA	Interpolative Compensation: Interpolační kompenzace
IM	Interface-Modul: modul rozhraní
IMR	Interface-Modul Receive: modul rozhraní pro přijímací režim
IMS	Interface-Modul Send: modul rozhraní pro režim odesílání
INC	Increment: Velikost kroku
INI	Initializing Data: Inicializační data
IPO	Interpolátor
ISA	Mezinárodní standardní architektura
ISO	International Standard Organization (mezinárodní organizace pro normy)
JOG	Jogging: Seřizovací režim
K-Bus	Komunikační sběrnice
K1 .. K4	Kanál 1 až kanál 4
KD	Rotace souřadného systému
Kód EIA	Speciální formát děrné pásky, počet děr na znak je vždy lichý.
Kód ISO	Speciální formát děrné pásky, počet děr na znak je vždy sudý.
KOP	Kontaktní schéma (metoda programování pro PLC)
$K_{Ü}$	Převodový poměr
K_v	Zesílení smyčky
LCD	Liquid-Crystal Display: displej z tekutých krystalů
LED	Light-Emitting Diode: světelná dioda
LF	Line Feed (konec řádku)
LMS	Systém pro měření polohy
LR	Regulátor polohy
LUD	Lokální uživatelská data
MB	Megabyte
MCS	Souřadný systém stroje
MD	Strojní parametry
MDA	Manual Data Automatic: Manuální zadávání
MK	Měřicí obvod
MLFB	Strojově čitelné označení produktu
MPF	Main Program File: výrobní program pro NC systém (hlavní program)

MPI	Multi Port Interface: Rozhraní s více porty
MS-	Microsoft (výrobce softwaru)
MSTT	Řídící panel stroje
NC	Numerical Control: Numerický řídící systém
NCK	Numerical Control Kernel: Jádro numerického řídícího systému pro přípravu bloků, řízením posuvů atd.
NCU	Numerical Control Unit: Hardwarová jednotka NCK
NRK	Název operačního systému v NCK
NST	Signál rozhraní
NURBS	Neuniformní racionální B-spliny
NV	Posunutí počátku
OB	Organizační modul v PLC
OEM	Original Equipment Manufacturer (původní výrobce zařízení)
OP	Operation Panel: Zařízení pro obsluhu systému
OPI	Operation Panel Interface: Rozhraní ovládacího panelu
OPT	Options: volitelné doplňky
OSI	Open Systems Interconnection: Spojení otevřených systémů – norma pro komunikaci mezi počítači
P-Bus	Sběrnice pro periferie
PC	Osobní počítač
PCIN	Název programového vybavení pro výměnu dat s řídícím systémem
PCMCIA	Personal Computer Memory Card International Association: (Mezinárodní asociace pro paměťové karty do osobních počítačů) – normy pro paměťové karty
PCU	Jednotka PC: skříň PC (výpočetní jednotka)
PG	Programovací přístroj
PLC	Programmable Logic Control: Programovatelné logické řízení
POS	polohovací
RAM	Random Access Memory (dynamická paměť RAM): programová paměť, kterou lze číst a do níž lze zapisovat
REF	Funkce najíždění na referenční bod
REPOS	Funkce najíždění na původní polohu
RISC	Reduced Instruction Set Computer: Typ procesor s malým instrukčním souborem a rychlým zpracováním příkazů.
ROV	Rapid Override: korekce rychlého posuvu
RPA	R-Parameter Active: Paměťová oblast v NCK pro čísla R-parametrů R-NCK.
RPY	Roll Pitch Yaw: Způsob otáčení souřadného systému
RS-232	Sériové rozhraní (definice signálových vedení pro výměnu dat mezi DTE a DCE)
RTS	Request To Send: Požadavek na odeslání, řídící signál sériového rozhraní pro přenos dat.
SBL	Single Block: Zpracování blok po bloku
SD	Nastavovaný parametr
SDB	Systémový datový modul
SEA	Setting Data Active: Identifikace (datový typ) pro nastavované parametry

SFB	Systémový funkční modul
SFC	Systémové volání funkce
SK	Programové tlačítko
SKP	Skip: Přeskočení bloku
SM	Krokový motor
SPF	Sub Program File: Podprogram
SPS	Řídící systém s programovatelnou pamětí
SRAM	Statická paměť (se zálohovaným napájením)
SRK	Korekce rádiusu břitu
SSFK	Korekce chyby stoupání vřetena
SSI	Serial Synchron Interface: Synchronní sériové rozhraní
SW	Software
SYF	System Files: Systémové soubory
TEA	Testing Data Active: Identifikace strojních parametrů
TO	Tool Offset: Korekční parametry nástroje
TOA	Tool Offset Active: Označení (typ souboru) pro korekční parametry nástroje
TRANSMIT	Transform Milling into Turning: Přepočítávání souřadného systému na soustruzích pro frézovací obrábění.
UFR	User Frame: Posunutí počátku
UP	Podprogram
VSA	Pohon posuvu
WCS	Souřadný systém obrobku
WKZ	Nástroj
WLK	Korekce délky nástroje
WOP	Dílensky orientované programování
WPD	Work Piece Directory: Adresář obrobku
WRK	Korekce rádiusu nástroje
WZK	Korekční parametry nástroje
WZW	Výměna nástroje
ZOA	Zero Offset Active: Identifikace (datový typ) dat posunutí počátku
µC	Mikrokontrolér

A.2 Přehled dokumentace

Přehled dokumentace systému SINUMERIK 840D sl

Všeobecná dokumentace

- SINUMERIK 840D sl — Reklamní brožura
- SINUMERIK 840D sl 840Di sl 802D sl — Katalog NC 61
- SINAMICS S120 — Katalog PM 21 SIMOTION, SINAMICS S120 a motory pro výrobní stroje

Uživatelská dokumentace

- SINUMERIK 840D sl — Návod k obsluze
  – Universal
  - Soustružení
  – Frézování
  – HMI-Advanced
- SINUMERIK 840D sl 828D — Návod k obsluze
  - Soustružení
  - Frézování
- SINUMERIK 840D sl 828D — Příručka programování
  – Základy
  – Pro pokročilé
  – Měřicí cykly
- SINUMERIK 840D sl 840Di sl 828D 802D sl — Příručka programování
  – Soustružení ISO
  – Frézování ISO
- SINUMERIK 840D sl S120 — Příručka pro diagnostiku

Dokumentace výrobce / servisní dokumentace

- SINUMERIK 840D sl — Příručka k přístroji
  – NCU
  – Ovládací prvky a zapojení do sítě
- SINUMERIK 840D sl S120 — Systémová příručka
  Příručka pro konfiguraci stroje
- SINUMERIK 840D sl 828D — Systémová příručka
  Ctrl Energie
- SINUMERIK 840D sl — Příručka pro uvádění do provozu
  – CNC: NCK, PLC, Pohon
  – Základní programové vybavení a progr. vybav. pro obsluhu systému
  – Základní programové vybavení a HMI-Advanced
- SINUMERIK 840D sl S120 — Popis seznamů
  – Díl 1
  – Díl 2
  – Podrobný popis strojních parametrů
  – Systémové proměnné

Dokumentace výrobce / servisní dokumentace

- SINUMERIK 840D sl 828D — Popis funkcí
  – Základní funkce
  – Rozšiřovací funkce
  – Speciální funkce
  – Synchronní akce
  – Správa nástrojů
- SINUMERIK 840D sl 840Di sl 802D sl 828D — Popis funkcí
  Dialekty ISO
- SINAMICS S120 — Popis funkcí
  Funkce pro pohony
- SINUMERIK 840D sl — Popis funkcí
  Safety Integrated
- SINUMERIK — Příručka pro nastavení konfigurace
  Konstrukční směrnice pro EMC

Informace / Školení

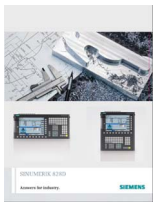
- SINUMERIK — Úvod pro začátečníky
  Frézování a soustružení
- SINUMERIK — Podklady pro školení
  – Jednoduché frézování se systémem ShopMill
  – Jednoduché soustružení se systémem ShopTurn
- SINUMERIK — Příručky
  Výroba nástrojů a Forem

Elektronická dokumentace

-  — DOConCD
  DOConWEB
-  — My Documentation Manager

Přehled dokumentace SINUMERIK 828D

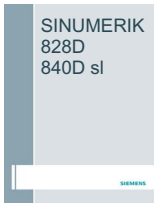
Všeobecná dokumentace



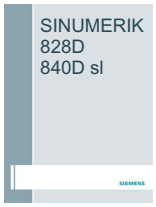
Reklamní brožura

Směrnice pro EMC

Uživatelská dokumentace

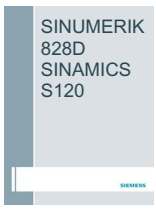


Návod k obsluze
- Soustružení
- Frézování



Příručka programování
- Základy
- Pro pokročilé
- Easy Screen

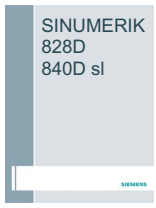
Příručka programování
- Soustružení ISO
- Frézování ISO



Příručka pro diagnostiku

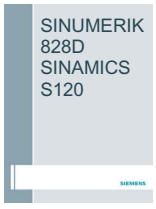
Dokumentace výrobce / servisní dokumentace

Příručka k přístroji
Příručka pro uvádění do provozu
Servisní příručka



Popis funkcí
- Základní funkce
- Rozšiřovací funkce

Popis funkcí
Dialekty ISO



Popis seznamů
- Parametry stroje a signály rozhraní
- Podrobný popis parametrů

Elektronická dokumentace

DOConCD
DOConWEB

Průmyslová zóna

Glosář

Absolutní rozměry

Udání cíle pohybu osy prostřednictvím údaje, který je vztažen na počátek momentálně platného souřadného systému. Viz → Inkrementální rozměr

Adresa

Adresa je identifikátor pro určitý operand nebo rozsah operandů, např. vstup, výstup atd.

Adresa osy

Viz → Identifikátor osy

Alarmy

Všechna → hlášení a alarmy se vypisují prostým textem na ovládacím panelu spolu s datem, časem a odpovídajícím symbolem pro kritérium vymazání. Vypisování se uskutečňuje odděleně pro alarmy a hlášení.

1. Alarmy a hlášení ve výrobním programu

   Alarmy a hlášení se mohou přenášet ke zobrazení prostým textem přímo z výrobního programu.

2. Alarmy a hlášení z PLC

   Alarmy a hlášení stroje se mohou přenášet ke zobrazení prostým textem z programu PLC. Za tím účelem nejsou zapotřebí žádné doplňkové funkční moduly.

Archivace

Odesílání dat a/nebo adresářů do **externího** paměťového zařízení.

Asynchronní podprogram

Výrobní program, který může být spuštěn asynchronně, tedy nezávisle na aktuálním stavu jiného programu signálem přerušení (např. signál "rychlejší vstup NC systému").

Automatický režim

Provozní režim řídícího systému (režim zpracovávání posloupnosti bloků podle DIN): Provozní režim NC-systémů, ve kterém je zvolen → výrobní program a ten je kontinuálně zpracováván.

Baudrate

Rychlost přenosu dat (bitů/s).

Bezpečnostní funkce

Řídící systém obsahuje neustále aktivní kontroly, které se snaží rozpoznat poruchy v → CNC, v → PLC a na stroji dostatečně včas, aby byly z větší části vyloučeny poškození obrobku, nástroje nebo stroje. V případě poruchy se operace obrábění přeruší a pohony se vypnou, příčina poruchy se uloží do paměti a aktivuje se alarm. Současně se sdělí do PLC, že se spustil alarm CNC.

Blok výrobního programu

Část → výrobního programu, která je vymezena znaky Line Feed. Jsou rozlišovány → hlavní bloky a → vedlejší bloky.

Celkový reset

V případě celkového resetu jsou z → CPU vymazány následující paměti:

- → Pracovní paměť
- oblasti pro čtení a zápis → paměti pro načítání
- → Systémová paměť
- → Zálohovaná paměť

CNC

Viz → NC

COM

Součást řídícího systému NC pro uskutečňování a koordinaci komunikace.

CPU

Central Processor Unit, → Centrální procesorová jednotka

C-Spline

C-spline je nejznámějším a nejčastěji používaným splinem. Přechody mezi uzlovými body mají spojitou tečnu a zakřivení. Používají se polynomy 3. stupně.

Cykly

Cykly jsou chráněné podprogramy pro uskutečňování opakovaně se vyskytujících obráběcích procesů na → obrobcích.

Časově reciproční posuv

U systému SINUMERIK 840D může být namísto rychlosti posuvu pro pohyb osy naprogramován čas, za jaký se má úsek dráhy v bloku urazit (G93).

Datové slovo

Datová jednotka o velikosti dva byty v → datovém modulu.

Datový modul

1. Datová jednotka → PLC, ke které mají přístup programy → HIGHSTEP.
2. Datová jednotka → NC systému: Datové moduly obsahují definice pro globální uživatelská data. Data mohou být při své definici přímo inicializována.

Definice proměnných

Definice proměnné zahrnuje stanovení datového typu a názvu proměnné. Pomocí názvu proměnné je přístup k hodnotě proměnné.

Diagnostika

1. Systémová oblast řídícího systému
2. Řídící systém obsahuje jak program pro diagnostiku sebe sama, tak také zkušební nástroje pro servis: Stavové, alarmové a servisní obrazovky.

Dráhová osa

Dráhové osy jsou všechny osy podílející se na obrábění v → kanálu, které jsou → interpolátorem ovládány tak, aby byly současně spouštěny, urychlovány, zastavovány a naváděny do koncového bodu.

DRF

Differential Resolver Function: Funkce NC systému, která ve spojení s elektronickým ručním kolečkem vytváří v režimu "Auto" inkrementální posunutí počátku.

Dynamická funkce předběžného zpracování

Nepřesnosti → kontury způsobované vlečnou chybou se dají téměř eliminovat dynamickou funkcí předběžného zpracování, která je závislá na zrychlení. Díky tomu se dosahuje i při vysokých → rychlostech pohybu po dráze vynikající přesnosti opracování. Předběžné zpracování může být pro jednotlivé osy ve → výrobním programu aktivováno a deaktivováno.

Editor

Editor umožňuje sestavování, upravování, doplňování, kompresi a vkládání programů/textů/programových bloků.

Externí posunutí počátku

Posunutí počátku specifikované → PLC.

Frame

Frame představuje matematický předpis, který převádí jeden kartézský souřadný systém do jiného kartézského souřadného systému. Frame obsahuje tyto komponenty: → posunutí počátku, → otočení, → změna měřítka, → zrcadlové převrácení.

Geometrická osa

Geometrické osy slouží pro popis 2- nebo 3-rozměrných oblastí v souřadném systému obrobku.

Geometrie

Popis → obrobku v → souřadném systému obrobku.

HIGHSTEP

Shrnutí programovacích možností pro → PLC systému AS300/AS400.

Hlášení

Všechna hlášení naprogramovaná v programu pro výrobu součásti a systémem rozpoznané → alarmy se vypisují na řídícím panelu stroje srozumitelným textem doplněným o udání data a času a o příslušný symbol pro kritérium vymazání. Vypisování se uskutečňuje odděleně pro alarmy a hlášení.

Hlavní blok

Blok začínající znakem ":", který obsahuje všechny příkazy, které jsou zapotřebí pro spuštění pracovního postupu ve → výrobním programu.

Hlavní program

Označení hlavní program pochází ještě z dob, kdy byly výrobní programy pevně rozděleny na hlavní programy a → podprogramy. Toto pevné rozdělení v dnešním jazyku systému SINUMERIK už neexistuje. V principu může být kterýkoli výrobní program v kanálu zvolen a spuštěn. To se potom uskutečňuje na → programové úrovni 0 (úroveň hlavního programu). V hlavním programu mohou být jako podprogramy vyvolávány další výrobní programy nebo → cykly.

Hodnota kompenzace

Rozdíl mezi polohou osy zjištěnou měřicím snímačem a požadovanou naprogramovanou polohou osy.

Chráněný prostor

Trojrozměrný prostor v rámci → pracovního prostoru, do kterého se špička nástroje nesmí dostat.

Identifikátor

Slova podle normy DIN 66025 jsou doplňována identifikátory (názvy) pro proměnné (početní proměnné, systémové proměnné, uživatelské proměnné), pro podprogramy, pro klíčová slova a slova s více adresovými písmeny. Tato doplnění mají při sestavování bloku stejný význam jako slova. Identifikátor musí být jednoznačný. Stejný identifikátor se nesmí používat pro různé objekty.

Identifikátor osy

Podle normy DIN 66217 pro pravoúhlý pravotočivý → souřadný systém jsou osy označeny X, Y, Z.

→ Kruhové osy otáčející se okolo os X, Y, Z mají identifikátory A, B, C. Doplňkové osy mohou být souběžně s již uvedenými označeny dalšími adresovými písmeny.

Interpolace spliny

Pomocí splinové interpolace je řídící systém schopen pouze na základě několika předem zadaných opěrných bodů vytvořit požadovanou konturu s hladkým křivkovým průběhem.

Interpolační kompenzace

Prostřednictvím interpolační kompenzace mohou být kompenzovány chyby vřetena (SSFK) a chyby měřicího systému (MSFK) (**S**pindel**s**teigungs**f**ehler a **M**ess**s**ystem**f**ehler **K**ompensation) podmíněné obráběcím postupem.

Interpolátor

Logická jednotka systému → NCK, která po zadání cílové pozice ve výrobním programu stanoví pomocné hodnoty pro jednotlivé osy odpovídající pohybu, který je potřeba uskutečnit.

Jednotka TOA

Každá → oblast TOA může obsahovat větší počet jednotek TOA. Počet možných jednotek TOA je omezen maximálním možným počtem aktivních → kanálů. Jednotka TOA obsahuje právě jeden datový modul nástrojů a jeden datový modul zásobníku. Kromě toho může obsahovat ještě i jeden datový modul držáku nástroje (volitelné).

JOG

Provozní režim řídícího systému (seřizování): V provozním režimu JOG je možné provádět seřizování stroje. Jednotlivými osami a vřeteny je možné pohybovat pomocí směrových tlačítek v tipovacím režimu. Dalšími funkcemi v provozním režimu JOG jsou → najíždění na referenční bod, → Repos a → Preset (dosazení skutečné hodnoty).

Kanál

Kanál se vyznačuje tím, že může zpracovávat → výrobní program nezávisle na jiných kanálech. Kanál řídí výlučně osy a vřetena, která mu byla přiřazena. Programové postupy různých kanálů mohou být prostřednictvím → synchronizace koordinovány.

Kanál pro zpracování

Prostřednictvím kanálové struktury mohou být zkráceny jalové časy, neboť pohybové operace mohou probíhat paralelně, např.posuv podavače souběžně s obráběním. Na kanál CNC je přitom možno pohlížet jako na samostatný CNC řídící systém s dekódováním, přípravou bloků a interpolací.

Klíč programátora

Znaky a posloupnosti znaků, které v programovacím jazyku pro → výrobní programy mají pevně definovaný význam.

Klíčová slova

Slova s pevně definovaným způsobem zápisu, která mají v programovacím jazyku pro výrobní program definovaný význam.

Kompenzace chyby kvadrantu

Chyby kontury na přechodech mezi kvadranty, které vznikají v důsledku měnících se podmínek tření na vodicích drahách, mohou být do značné míry odstraněny kompenzací chyby kvadrantu. Dosazení parametrů pro kompenzaci chyby kvadrantu se provádí pomocí zkoušky kruhového tvaru.

Kompenzace chyby stoupání vřetena

Vyrovnávání mechanické nepřesnosti vřetena podílejícího se na posuvu prováděné řídícím systémem na základě změřených hodnot odchylek.

Kompenzace vůle

Vyrovnávání mechanických vůlí stroje, např. na valivých ložiscích při změně směru. Pro každou osu se může kompenzace vůle zadávat odděleně.

Kompenzační osa

Osa, jejíž požadovaná a skutečná hodnota byly modifikovány hodnotou kompenzace.

Kompenzační tabulka

Tabulka uzlových bodů. Jsou zde uvedeny kompenzační hodnoty kompenzační osy pro zvolené pozice základní osy.

Kontrola kontury

Jako měřítko pro zachování kontury se sleduje, zda vlečná chyba leží v rámci definovaného tolerančního pásma. Nepřípustně vysoká vlečná chyba může mít např. za následek přetížení pohonu. V takovém případě se aktivuje alarm a osy se zastaví.

Kontura

Obrys → obrobku

Kontura hotového obrobku

Kontura nahotovo obrobeného obrobku. Viz → Surový obrobek.

Kontura obrobku

Požadovaná kontura vyráběného/obráběného → obrobku.

Korekce rádiusu břitu

Při programování kontury se vychází z toho, že nástroj je špičatý. Jelikož toto v praxi není realizovatelné, zadává se do řídícího systému rádius zakřivení použitého nástroje, který se potom bere v úvahu. Při vedení nástroje podél kontury se střed zakřivení pohybuje ve stále stejné vzdálenosti rovnající se rádiusu zakřivení.

Korekce rádiusu nástroje

Abyste mohli požadovanou → konturu obrobku přímo naprogramovat, musí řídící systém pohybovat nástrojem po ekvidistantní dráze vzhledem ke kontuře, přičemž musí znát přesný rádius použitého nástroje (`G41`/`G42`).

Korekční parametry nástroje

Zohledňování rozměrů nástroje při výpočtu dráhy.

Kostra

Za kostru se považuje celek složený ze všech vzájemně spojených neaktivních dílů výrobního prostředku, kde se ani v případě poruchy nemůže vyskytnout nebezpečné dotykové napětí.

Kruhová interpolace

→ Nástroj se má pohybovat po kruhové dráze mezi pevně zvolenými body kontury s uvedeným posuvem a přitom opracovávat obrobek.

Kruhová osa

Kruhová osa uskutečňuje otočení obrobku nebo nástroje do předem definované úhlové polohy.

KÜ

Převodový poměr

KV

Faktor zesílení smyčky, regulační charakteristika regulačního obvodu.

Lineární osa

Lineární osa je osa, která oproti kruhové ose opisuje přímku.

Look Ahead

Pomocí funkce **Look Ahead** řídící systém vyhodnocuje několik bloků dopředu (tento počet lze nastavit pomocí parametru), čímž se dosahuje optimální rychlosti při zpracování.

MDA

Provozní režim řídícího systému: Manual Data Automatic. V provozním režimu MDA mohou být jednotlivé bloky programu nebo jejich posloupnosti zadávány bez vztahu na hlavní program nebo podprogram a potom mohou být stisknutím tlačítka NC-Start ihned uskutečňovány.

Měřicí jednotky palce nebo metrické

V programu pro obrábění můžete pozice a hodnoty stoupání programovat v palcích. Nezávisle na programovatelných měřicích jednotkách (`G70`/`G71`) se řídící systém převede na základní systém.

Měřicí systém využívající palce

Měřicí systém, který vzdálenosti udává v „palcích“ a jejich zlomcích.

Metrický systém měřicích jednotek

Normovaný systém využívající jednotky: pro délky např. mm (milimetr), m (metr).

Mez přesného najetí

Pokud všechny dráhové osy dosáhnou své meze přesného najetí, řídící systém se chová, jako by bylo cílového bodu přesně dosaženo. Uskuteční se přechod na další blok → výrobního programu.

Mezní hodnota otáček

Maximální/minimální otáčky (vřetena): Zadáním strojních parametrů, parametrů → PLC, nebo → nastavovaných parametrů mohou být maximální otáčky vřetena omezeny.

Modul

Pojmem "moduly" jsou označovány všechny soubory, které jsou zapotřebí pro vytváření a zpracovávání programů.

Najíždění na pevný bod

Obráběcí stroje mohou definovaným způsobem najíždět na pevné body, jako je např. bod pro výměnu nástroje, zakládací bod, bod pro výměnu palety atd. Souřadnice těchto bodů jsou uloženy v řídícím systému. Pokud je to možné, řídící systém pohybuje příslušnými osami → rychlým posuvem.

Nastavované parametry

Parametry, které definovaným způsobem systémovým programovým vybavením zprostředkovávají řídícímu systému NC vlastnosti obráběcího stroje.

Nástroj

Pracovní součást na obráběcím stroji, která způsobuje obrábění, např. soustružnický nůž, fréza, vrták, laserový paprsek …).

Název osy

Viz → Identifikátor osy

NC

Numerical Control: Řídící systém zahrnující všechny komponenty pro ovládání obráběcího stroje: → NCK, → PLC, → HMI, → COM.

Poznámka

Pro řídící systémy SINUMERIK 840D by bylo přesnější označení CNC: Computerized Numerical Control.

NCK

Numerical Control Kernel: Součást řídícího systému, která zpracovává → výrobní program a v zásadě koordinuje pohybové operace obráběcího stroje.

NRK

Numerický robotický kernel (operační systém → NCK).

NURBS

V rámci řídícího systému prováděné interní vedení pohybu a dráhová interpolace se uskutečňují na základě NURBS (**N**e**U**niformní **R**acionální **B**-**S**pliny). Díky tomu je v řídícím systému SINUMERIK 840D k dispozici jednotné chování pro všechny interpolace.

Oblast TOA

V oblasti TOA jsou soustředěna všechna data nástrojů a zásobníků. Pokud jde o dosažitelnost těchto dat, standardně se oblast kryje s oblastí → kanálu. Pomocí strojních parametrů je však možné nastavit, že několik kanálů → jednotku TOA sdílí, takže tyto kanály potom mohou mít k dispozici společná data nástrojů.

Obrábění šikmých ploch

Vrtání a frézování na plochách obrobku, které neleží v souřadných rovinách stroje, se mohou pohodlně uskutečňovat s podporou funkce „obrábění šikmých ploch“.

Obrobek

Součást, která má být vyráběna nebo opracovávána obráběcím strojem.

OEM

Pro výrobce stroje, který si přeje v řídícím systému instalovat své vlastní uživatelské rozhraní nebo specifické technologické funkce, existuje prostor pro individuální řešení (aplikace OEM) pro SINUMERIK 840D.

Ohraničení pracovního pole

Navíc kromě koncových spínačů může být rozsah pohybu os dále omezen pomocí ohraničení pracovního pole. Pro každou osu může existovat dvojice hodnot, která chráněný pracovní prostor popisuje.

Orientované zastavení vřetena

Zastavení vřetena obrobku v předem definované úhlové poloze, např. aby bylo možné uskutečnit další obrábění na určitém místě.

Orientovaný návrat nástroje

RETTOOL: Při přerušení obrábění (např. při zlomení nástroje) je možné nástroj pomocí programového příkazu stáhnout zpět s předem definovanou orientací.

Osa C

Osa, okolo které se uskutečňuje řízený otočný pohyb a polohování s nástrojovým vřetenem.

Osy

Osy CNC jsou v závislosti na spektru svých funkcí rozděleny do těchto kategorií:

- Osy: interpolační dráhové osy
- Pomocné osy: Přísuvné a polohovací osy bez interpolace a se specifickým osovým posuvem. Pomocné osy se nepodílejí na vlastním obrábění, např. jsou to podavače nástroje, zásobník nástrojů atd.

Osy stroje

Fyzicky existující osy v obráběcím stroji.

Otočení

Složka → framu, která definuje otočení souřadného systému o určitý úhel.

Override

Manuální, příp. programovatelná možnost zásahu, která obsluhujícímu pracovníkovi umožňuje změnit naprogramované posuvy nebo otáčky, aby je bylo možné přizpůsobit určitému obrobku či materiálu.

Override posuvu

Naprogramovaná rychlost je nahrazena aktuálním nastavením rychlosti uskutečněným pomocí → řídícího panelu stroje nebo na → PLC (0-200%). Rychlost posuvu může být dodatečně měněna v programu pro opracování součásti prostřednictvím programovatelného procentuálního faktoru (1 – 200 %).

Paměť korekcí

Datová oblast řídícího systému, ve které jsou uloženy korekční parametry nástroje.

Paměť pro načítání

Paměť pro načítání se u CPU 314 systému → SPS rovná → pracovní paměti.

Periferní modul

Periferní moduly vytvářejí spojení mezi CPU a procesem.

Jedná se o následující:

- → Moduly digitálních vstupů/výstupů
- → Moduly analogových vstupů/výstupů
- → Simulační moduly

Pevný bod stroje

Bod jednoznačně definovaný obráběcím strojem, např. referenční bod stroje.

PLC

Programmable **L**ogic **C**ontrol: → Řídící systém s programovatelnou pamětí. Komponenty řídícího → NC systému: Přizpůsobení řídícího systému pro řídící logiku obráběcího stroje.

Počátek souřadného systému obrobku

Počátek (nula) → souřadného systému obrobku tvoří výchozí bod této soustavy. Je definován vzdáleností od počátku → souřadné soustavy stroje.

Počátek souřadného systému stroje

Pevný bod obráběcího stroje, na který jsou vztaženy všechny (odvozené) měřicí systémy.

Podprogram

Označení podprogram pochází ještě z dob, kdy byly výrobní programy pevně rozděleny na → hlavní programy a podprogramy. Toto pevné rozdělení v dnešním jazyku systému SINUMERIK už neexistuje. V principu může být každý výrobní program nebo každý → cyklus vyvolán jako podprogram v rámci nějakého jiného výrobního programu. To se potom uskutečňuje na → programové úrovni (x+1) (úroveň podprogramu (x+1)).

Pohon

Pohon je tou jednotkou CNC systému, která na základě dat z NC systému uskutečňuje regulaci otáček a momentu.

Polární souřadnice

Souřadný systém, ve kterém je poloha bodu v rovině dána vzdáleností od počátku a úhlem, který svírá vektor rádiusu s definovanou osou.

Polohovací osa

Osa, která provádí pomocné pohyby na obráběcím stroji (např. zásobník nástrojů, přeprava palet). Polohovací osy jsou osy, které nejsou interpolovány spolu s → dráhovými osami.

Polynomická interpolace

Pomocí polynomické interpolace mohou být konstruovány křivky rozmanitých průběhů, jako jsou **přímka, parabola, mocninná funkce** atd. (SINUMETIK 840D).

Pomocné bloky

Pracovní posuvy s aktivovanou → korekcí nástroje (`G41`/`G42`) smí být přerušeny omezeným počtem pomocných bloků (bloků bez pohybu os v rovině korekce), přičemž korekce nástroje se ještě dá správně vypočítat. Přípustný počet pomocných bloků, které je řídící systém schopen dopředu načíst, je nastavitelný pomocí systémového parametru.

Pomocné funkce

Prostřednictvím pomocných funkcí mohou být ve → výrobních programech předávány → parametry do → PLC, které tam potom spouští výrobcem stroje definovanou reakci.

Posunutí počátku

Udání nového vztažného bodu pro souřadný systém, které je vztaženo na již existující počátek a frame.

1. Nastavitelné

   SINUMERIK 840D: K dispozici je určitý v konfiguraci definovaný počet nastavitelných posunutí počátku pro každou CNC osu. Alternativně lze používat posunutí aktivovaná pomocí G-funkcí.

2. Externí

   Navíc na všechna posunutí, jež definují polohu souřadného systému obrobku, může být aplikována korekce externím posunutím počátku pomocí ručního kolečka (posunutí DRF) nebo z PLC.

3. Programovatelná

   Pomocí příkazu `TRANS` lze naprogramovat posunutí pro všechny dráhové a polohovací osy.

Posuv po dráze

Posuv po dráze se vztahuje na → dráhové osy. Představuje geometrický součet posuvů → geometrických os, které se na něm podílejí.

Pracovní paměť

Pracovní paměť je paměť typu RAM v → CPU, do níž má přístup procesor během zpracování uživatelského programu.

Pracovní prostor

Trojrozměrný prostor, v němž se na základě konstrukce obráběcího stroje může pohybovat špička nástroje. Viz také → Chráněný prostor

Program pro přenos dat PCIN

PCIN je pomocný program pro odesílání a přijímání uživatelských dat CNC přes sériové rozhraní, jako jsou např. výrobní programy, korekční parametry nástroje atd. Program PCIN se může spouštět pod MS-DOSem na standardních průmyslových PC.

Programová paměť PLC

SINUMERIK 840D: V uživatelské paměti PLC jsou společně uloženy uživatelský program PLC a uživatelská data a základní program PLC.

Programová úroveň

Výrobní program spuštěný v kanálu je zpracováván jako → hlavní program na programové úrovni 0 (úroveň hlavního programu). Každý výrobní program, který je vyvoláván v hlavním programu, je zpracováván jako → podprogram na své vlastní programové úrovni 1 ... n.

Programování PLC

PLC se programuje pomocí softwaru **STEP 7**. Programovací software STEP 7 je založen na standardním operačním systému **Windows** a obsahuje funkce systému STEP 5 s nově vyvinutými rutinami.

Programovatelné framy

Pomocí programovatelných → framů je možné dynamicky v průběhu zpracovávání výrobního programu definovat nové počátky souřadného systému. Je třeba rozlišovat mezi absolutní definicí na základě nového framu a aditivní definicí vycházející z již existujícího počátečního bodu.

Programovatelné ohraničení pracovního pole

Ohraničení pracovního prostoru pro pohyby nástroje na prostor vymezený programovými mezemi.

Programové tlačítko

Tlačítko, jehož popis je reprezentován políčkem na obrazovce. Toto tlačítko se dynamicky přizpůsobuje aktuální situaci obsluhy systému. Volně obsaditelným funkčním tlačítkům jsou programovým vybavením přiřazovány definované funkce.

Programový modul

Programové moduly obsahují hlavní programy a podprogramy → výrobního programu.

Provozní režim

Pojem označující způsob fungování řídícího systému SINUMERIK. Jsou definovány provozní režimy → Jog, → MDA, → Auto.

Předběžná koincidence

K přechodu na další blok dochází už tehdy, když se pohyb po dráze dostane do blízkosti koncového bodu, takže vzdálenost od něj je menší než předem zadaná hodnota delta.

Přepínač na klíč

Přepínač na klíč na ovládacím panelu stroje má 4 polohy, které jsou obsazeny funkcemi operačního systému řídícího systému. K přepínači na klíč patří tři různě barevné klíče, které je možné vytáhnout v dále uvedených polohách.

Přesné najetí

Při programovatelném příkazu přesného najetí se na pozici uvedenou v bloku najíždí přesně a v případě potřeby velmi pomalu. Pro zkrácení doby přibližování jsou pro rychlý a pracovní posuv definovány → meze přesného najetí.

Přímková interpolace

Nástroj se pohybuje po přímkách k cílovému bodu a přitom opracovává obrobek.

Referenční bod

Bod obráběcího stroje, na který je vztažen měřicí systém os stroje.

Režim řízení pohybu po dráze

Cílem řízení pohybu po dráze je zabránit velkým brzděním → dráhových os na hranicích bloků ve výrobním programu a přecházet do následujícího bloku pokud možno se stejnou rychlostí pohybu po dráze.

Rozsah posuvu

Maximální přípustný rozsah pohybu u lineárních os je ± 9 dekád. Absolutní hodnota závisí na zvolené jemnosti zadávané hodnoty a polohové regulace a na systému jednotek (palce nebo metrický systém).

R-Parametry

Početní parametry, mohou být programátorem → výrobního programu použity pro libovolné účely v programu nebo mohou být zjišťovány jejich hodnoty.

Rutina přerušení

Rutiny přerušení jsou speciální → podprogramy, které se mohou spouštět v důsledku určité události (externí signál) z technologického procesu. Právě zpracovávaný výrobní program se přeruší a pozice os, na které k přerušení došlo, se automaticky uloží.

Rychlé digitální vstupy/ výstupy

Pomocí digitálních vstupů se mohou spouštět např. rychlé programové CNC rutiny (rutiny přerušení). Pomocí digitálních CNC výstupů se mohou spouštět rychlé programem řízené spínací funkce (SINUMERIK 840D).

Rychlé pozvednutí od kontury

Vyskytne-li se přerušení, může být pomocí programu CNC spouštěn pohyb, který umožňuje rychlé pozvednutí nástroje od právě obráběné kontury obrobku. Kromě toho lze v parametrech nastavit úhel zpětného pohybu a délku této dráhy. Po rychlém pozvednutí se může spouštět navíc i rutina přerušení (SINUMERIK 840D).

Rychlost pohybu po dráze

Maximální naprogramovatelná dráhová rychlost závisí na jemnosti zadávané hodnoty. Například při rozlišení 0,1 mm činí maximální programovatelná dráhová rychlost 1000 m/min.

Rychlý posuv

Nejvyšší rychlost pohybu osy. Použije se např. tehdy, je-li potřeba nástrojem v klidu najet na → konturu obrobku nebo od kontury obrobku odjet. Rychlost rychlého posuvu je nastavena prostřednictvím strojního parametru specificky pro daný stroj.

Řetězové kótování

Též inkrementální rozměr: Stanovení cíle pohybu osy pomocí dráhy a směru, které je potřeba urazit, vztažené na již dosažený bod. Viz → Absolutní rozměr.

Řídící osa

Řídící osa je osa → gantry, která je z pohledu obsluhujícího pracovníka a programátora k dispozici a která může být v důsledku toho odpovídajícím způsobem ovlivňována stejně jako normální osa NC systému.

Řídící panel stroje

Řídící panel obráběcího stroje s ovládacími prvky, jako jsou tlačítka, otočné přepínače atd. a s jednoduchými signalizačními prvky, jako jsou světelné diody. Slouží k bezprostřednímu ovlivňování obráběcího stroje pomocí PLC.

Řídící systém s programovatelnou pamětí

Paměťové programovatelné řídící systémy (SPS) jsou elektronické řídící systémy, jejichž funkce je uložena ve formě programu v paměťovém zařízení. Konstrukce a zapojení zařízení tedy nezávisí na funkci řídícího systému. Paměťové programovatelné řídící systémy mají konstrukci počítače: skládají se z CPU (centrální modul) s pamětí, modulů vstupů/výstupů a interního sběrnicového systému. Periferie a programovací jazyk jsou podřízeny potřebám řízení.

Řízení podle rychlosti

Aby při pracovních posuvech o velmi krátké vzdálenosti na blok bylo možné dosáhnout přijatelné rychlosti pohybu, je možné aktivovat vyhodnocování průběhu rychlosti na několik bloků dopředu (funkce → Look Ahead).

Sériové rozhraní RS-232

Pro vstup/výstup dat je na jednotce PCU20 jedno sériové rozhraní RS-232, na jednotce PCU 50/70 jsou k dispozici dvě rozhraní RS-232. Pomocí těchto rozhraní můžete načítat, odesílat a zálohovat výrobní programy, jakož i data výrobce a uživatelská data.

Síť

Síť je spojení několika systémů S7-300 a dalších koncových zařízení, např. PG, pomocí → spojovacího kabelu. Prostřednictvím sítě se uskutečňuje výměna dat mezi připojenými zařízeními.

Skupiny provozních režimů

Osy a vřetena, které k sobě po technologické stránce patří, mohou být soustředěny do skupiny provozních režimů (BAG). Osy/vřetena v jedné skupině provozních režimů mohou být řízeny prostřednictvím jednoho nebo více → kanálů. Kanálům ve skupině provozních režimů je vždy přiřazen stejný → provozní režim.

Softwarový koncový spínač

Softwarový koncový spínač omezuje rozsah pohybu osy a zabraňuje najíždění saní na hardwarový koncový spínač. Pro každou osu je možné zadat 2 páry hodnot, které pak mohou být odděleně aktivovány pomocí → PLC.

Souřadný systém

Viz → souřadný systém stroje, → souřadný systém obrobku

Souřadný systém obrobku

Souřadný systém obrobku je svým → počátkem (nulou) vztažen na obrobek. Při programování v souřadném systému obrobku jsou rozměry a směry vztaženy na tento systém.

Souřadný systém stroje

Souřadný systém, který je vztažen na osy obráběcího stroje.

Spirální interpolace

Spirální interpolace se hodí obzvláště pro jednoduchou výrobu vnějších a vnitřních závitů s tvarovými frézami a pro frézování mazacích drážek.

Spirála se přitom skládá ze dvou pohybů:

- Kruhový pohyb v rovině
- Lineární pohyb kolmo na tuto rovinu

Spojovací kabel

Spojovací kabely jsou dodávaná nebo uživatelem vyrobená dvoudrátová spojení s konektory na obou koncích. Tyto spojovací kabely propojují → CPU pomocí → vícebodového rozhraní (MPI) s → PG nebo s jinou CPU.

Správa výrobních programů

Správa výrobních programů může být organizována podle → obrobků. Počet programů a dat, která lze spravovat, je dána velikostí uživatelské paměti. Každý soubor (program a data) může být opatřen názvem skládajícím se z maximálně 24 alfanumerických znaků.

Standardní cykly

Pro často se opakující obráběcí operace jsou k dispozici standardní cykly:

- pro technologie vrtání/frézování
- pro technologii soustružení

V systémové oblasti „Program“ pod menu „Podpora cyklů“ se nachází seznam cyklů, které jsou Vám k dispozici. Po aktivování požadovaného obráběcího cyklu se srozumitelným textem vypíší potřebné parametry, jimž je potřeba přiřadit odpovídající hodnoty.

Surový obrobek

Součást, na které má být zahájeno opracovávání obrobku.

Synchronizace

Příkazy na určitých místech ve → výrobním programu pro koordinaci operací v různých → kanálech.

Synchronizovaná osa

Synchronizovaná osa je osa → gantry, jejíž požadovaná poloha je neustále odvozena od pracovních posuvových pohybů → vodicí osy a která se proto pohybuje synchronizovaně. Z pohledu obsluhujícího pracovníka a programátora "není" synchronizovaná osa "k dispozici".

Synchronizované osy

Synchronní osy potřebují pro provedení svého pohybu stejný čas, jaký potřebuje geometrická osa pro svůj pohyb po dráze.

Synchronní akce

1. Výstup pomocné funkce

   Při opracovávání obrobku se mohou předávat z CNC programu do PLC technologické funkce (→ pomocné funkce). Pomocí těchto pomocných funkcí jsou např. řízeny pomocná zařízení obráběcího stroje, jako jsou pinola, podavač, upínací sklíčidlo atd.

2. Rychlé výstupy pomocných funkcí

   Pro časově kritické spínací funkce mohou být minimalizovány potvrzovací časy (→ pomocné funkce). Zbytečné body pozastavení jsou z obráběcího procesu odstraněny.

Systémová paměť

Systémová paměť je paměť v NCU, do které se ukládají následující data:

- Data, která potřebuje řídící systém
- Operandy časovačů, počítadel a ukazatelů

Systémové proměnné

Proměnná, která existuje bez přičinění programátora výrobního programu. Je definována svým datovým typem a názvem proměnné, který začíná znakem **$**. Viz také → Uživatelská proměnná.

Technika maker

Shrnutí většího počtu příkazů do jednoho identifikátoru. Tento identifikátor v programu reprezentuje tento daný počet soustředěných příkazů.

Textový editor

Viz → Editor

Transformace

Aditivní nebo absolutní posunutí počátku v jedné ose.

Uživatelem definované proměnné

Uživatel může pro libovolné využití ve → výrobním programu nebo v datovém modulu (globální uživatelská data) definovat uživatelské proměnné. Definice obsahuje udání datového typu a název proměnné. Viz také → Systémová proměnná.

Uživatelská paměť

Všechny programy a data, jako jsou výrobní programy, podprogramy, komentáře, korekční parametry nástroje, posunutí počátku/framy, jakož i kanálová a programová uživatelská data, mohou být společně uloženy v uživatelské paměti CNC systému.

Uživatelské rozhraní

Pracovní plocha je zobrazovací médium CNC řídícího systému představovaná displejem. Zobrazuje se s programovými tlačítky ve vodorovném a svislém pruhu.

Uživatelský program

Uživatelské programy pro automatizační systémy S7-300 jsou vytvářeny v programovacím jazyku STEP 7. Uživatelský program je strukturovaný a skládá se z jednotlivých modulů.

Základní typy modulů jsou:

- Modul kódů

  Tyto moduly obsahují příkazy jazyka STEP 7.

- Datový modul

  Tyto moduly obsahují konstanty a proměnné pro programy v jazyce STEP 7.

Vedlejší blok

Blok začínající „N“ a obsahující informace pro krok pracovního postupu, např. udání polohy.

Velikost kroku

Udání délky posuvu pomocí počtu inkrementů (velikost kroku). Počet inkrementů může být uložen jako nastavovaný parametr, příp. může být zvolen pomocí tlačítek s odpovídajícím popisem 10, 100, 1000, 10000.

Vrtání závitů bez vyrovnávací hlavičky

Pomocí této funkce můžete vyrábět závity bez vyrovnávací hlavičky. Díky interpolačnímu chování vřetena, které je řízeno jako kruhová osa a osa vrtání, jsou závity odříznuty přesně na koncové vrtané hloubce, např. závity ve slepých dírách (předpoklad: osový režim vřetena).

Vyhledávání bloku

Při testování výrobního programu nebo po přerušení jeho zpracování je možné pomocí této funkce vyhledat libovolné místo ve výrobním programu, od kterého se má zpracování znovu spustit nebo odkud má pokračovat.

Výrobní program

Posloupnost příkazů pro NC řídící systém, který zabezpečí celkové opracování určitého obrobku. Také uskutečnění určitého opracování na zadaném → surovém obrobku.

Vyšší programovací jazyk CNC

Vyšší programovací jazyk nabízí: → uživatelem definované proměnné, → systémové proměnné, → makra.

WinSCP

WinSCP je Open Source Program pro Windows sloužící pro přenášení souborů, který je volně k dispozici.

Základní osa

Osa, na kterou jsou vztaženy požadovaná nebo skutečná hodnota za účelem výpočtu hodnoty kompenzace.

Základní souřadný systém

Kartézský souřadný systém, který se prostřednictvím transformace zobrazuje na souřadný systém stroje.

Ve → výrobním programu programátor používá názvy os základního souřadného systému. Pokud není aktivní žádná → transformace, existuje paralelně k → souřadnému systému stroje. Liší se od něho v → identifikátorech os.

Zakřivení

Zakřivení k kontury je inverzní hodnota rádiusu r oskulační kružnice v daném bodě kontury (k = 1/r).

Záložní baterie

Záložní baterie zaručuje, že → uživatelský program v → CPU je chráněn proti výpadku napájení a že definované datové oblasti a značky, časy a čísla zůstanou nezměněny.

Zaokrouhlovací osa

Zaokrouhlovací osa uskutečňuje otočení obrobku nebo nástroje do odpovídající úhlové polohy v dělicí mřížce. Při dosažení mřížky je zaokrouhlovací osa "na svém místě".

Zavádění

Načítání systémových programů po zapnutí.

Změna měřítka

Komponent → framu, který způsobuje změnu měřítka pro určitou osu.

Zrcadlové převrácení

Při zrcadlovém převrácení jsou znaménka hodnot souřadnic osy vztahující se k dané kontuře vyměněna. Současně je možné zrcadlově převrátit i několik os.

Zrychlení s omezením ryvu

Aby bylo dosaženo optimálního průběhu zrychlení stroje a aby se současně šetřila jeho mechanika, je možné ve výrobním programu přepínat mezi skokovým a spojitým (bez trhnutí) zrychlováním.

Rejstřík

Symbols

Numerics

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z