SIEMENS

SINUMERIK

SINUMERIK 840D sl/828D
Notions complémentaires

Manuel de programmation

Avant-propos

Programmation CN flexible 1

Gestion des fichiers et programmes 2

Zones de protection 3

Instructions de déplacement spéciales 4

Transformations de coordonnées (FRAMES) 5

Transformations 6

Corrections d'outils 7

Modes de déplacement 8

couplages d'axes 9

Actions synchrones au déplacement 10

Oscillation 11

Poinçonnage et grignotage 12

Rectification 13

Autres fonctions 14

Programmes de chariotage personnalisés 15

Tableaux 16

Annexes A

Valable pour

Commande
SINUMERIK 840D sl / 840DE sl
SINUMERIK 828D

Logiciel	Version
Logiciel système NCU	2.6 SP1

03/2010
6FC5398-2BP20-1DA0

Mentions légales

Signalétique d'avertissement

Ce manuel donne des consignes que vous devez respecter pour votre propre sécurité et pour éviter des dommages matériels. Les avertissements servant à votre sécurité personnelle sont accompagnés d'un triangle de danger, les avertissements concernant uniquement des dommages matériels sont dépourvus de ce triangle. Les avertissements sont représentés ci-après par ordre décroissant de niveau de risque.

⚠DANGER
signifie que la non-application des mesures de sécurité appropriées **entraîne** la mort ou des blessures graves.

⚠ATTENTION
signifie que la non-application des mesures de sécurité appropriées **peut entraîner** la mort ou des blessures graves.

⚠PRUDENCE
accompagné d'un triangle de danger, signifie que la non-application des mesures de sécurité appropriées peut entraîner des blessures légères.

PRUDENCE
non accompagné d'un triangle de danger, signifie que la non-application des mesures de sécurité appropriées peut entraîner un dommage matériel.

IMPORTANT
signifie que le non-respect de l'avertissement correspondant peut entraîner l'apparition d'un événement ou d'un état indésirable.

En présence de plusieurs niveaux de risque, c'est toujours l'avertissement correspondant au niveau le plus élevé qui est reproduit. Si un avertissement avec triangle de danger prévient des risques de dommages corporels, le même avertissement peut aussi contenir un avis de mise en garde contre des dommages matériels.

Personnes qualifiées

L'appareil/le système décrit dans cette documentation ne doit être manipulé que par du **personnel qualifié** pour chaque tâche spécifique. La documentation relative à cette tâche doit être observée, en particulier les consignes de sécurité et avertissements. Les personnes qualifiées sont, en raison de leur formation et de leur expérience, en mesure de reconnaître les risques liés au maniement de ce produit / système et de les éviter.

Utilisation des produits Siemens conforme à leur destination

Tenez compte des points suivants:

⚠ATTENTION
Les produits Siemens ne doivent être utilisés que pour les cas d'application prévus dans le catalogue et dans la documentation technique correspondante. S'ils sont utilisés en liaison avec des produits et composants d'autres marques, ceux-ci doivent être recommandés ou agréés par Siemens. Le fonctionnement correct et sûr des produits suppose un transport, un entreposage, une mise en place, un montage, une mise en service, une utilisation et une maintenance dans les règles de l'art. Il faut respecter les conditions d'environnement admissibles ainsi que les indications dans les documentations afférentes.

Marques de fabrique

Toutes les désignations repérées par ® sont des marques déposées de Siemens AG. Les autres désignations dans ce document peuvent être des marques dont l'utilisation par des tiers à leurs propres fins peut enfreindre les droits de leurs propriétaires respectifs.

Exclusion de responsabilité

Nous avons vérifié la conformité du contenu du présent document avec le matériel et le logiciel qui y sont décrits. Ne pouvant toutefois exclure toute divergence, nous ne pouvons pas nous porter garants de la conformité intégrale. Si l'usage de ce manuel devait révéler des erreurs, nous en tiendrons compte et apporterons les corrections nécessaires dès la prochaine édition.

Siemens AG
Industry Sector
Postfach 48 48
90026 NÜRNBERG
ALLEMAGNE

Numéro de référence du document: 6FC5398-2BP20-1DA0
℗

Avant-propos

Documentation SINUMERIK

La documentation SINUMERIK comporte 3 catégories :

- Documentation générale
- Documentation utilisateur
- Documentation constructeur / S.A.V.

Le lien http://www.siemens.com/motioncontrol/docu fournit des informations sur les thèmes suivants :

- Ordering documentation
  Vous trouverez ici un aperçu actuel de la documentation disponible.
- Download documentation
  Liens permettant de télécharger des fichiers à partir de Service & Support
- (Online) research in the documentation
  Informations sur DOConCD et accès direct aux documents dans DOConWEB.
- Pour établir de la documentation individuellement sur la base des contenus Siemens à l'aide de My Documentation Manager (MDM) (Compiling documentation individually), voir http://www.siemens.com/mdm.

  My Documentation Manager propose des fonctionnalités permettant de créer votre propre documentation machine.
- Formation (Training) et foire aux questions (FAQ)

  Des informations sur les offres de formation et les FAQ (foires aux questions) sont accessibles à partir du menu de navigation.

Groupe cible

Le présent manuel s'adresse aux :

- programmeurs
- ingénieurs de projet

Utilité

Le Manuel de programmation permet au groupe cible de créer, d'écrire, de tester des programmes et des interfaces logicielles et de supprimer des erreurs.

Version standard

Le présent manuel de programmation décrit les fonctionnalités de la version standard. Les options complémentaires ou modifications apportées par le constructeur de la machine-outil sont documentées par celui-ci.

La commande numérique peut posséder des fonctions qui dépassent le cadre de la présente description. Le client ne peut toutefois pas faire valoir de droit en liaison avec ces fonctions, que ce soit dans le cas de matériels neufs ou dans le cadre d'interventions du service après-vente.

Pour des raisons de clarté, la présente documentation ne contient pas toutes les informations de détail relatives à toutes les variantes du produit ; elle ne peut pas non plus tenir compte de tous les cas d'installation, d'exploitation et de maintenance.

Assistance technique

Pour toutes vos questions techniques, adressez-vous au service d'assistance téléphonique :

	Europe / Afrique
Téléphone	+49 180 5050 - 222
Télécopie	+49 180 5050 - 223
0,14 €/minute depuis le réseau téléphonique fixe allemand, les tarifs de téléphonie mobile peuvent être différents.	
Internet	http://www.siemens.com/automation/support-request

	Amérique
Téléphone	+1 423 262 2522
Télécopie	+1 423 262 2200
Courrier électronique	mailto:techsupport.sea@siemens.com

	Asie / Pacifique
Téléphone	+86 1064 757575
Télécopie	+86 1064 747474
Courrier électronique	mailto:support.asia.automation@siemens.com

Remarque

Vous trouverez les numéros de téléphone pour une assistance technique dans les différents pays sur le site Internet :
http://www.automation.siemens.com/partner

Questions concernant la documentation

Pour toute autre demande (suggestion, correction) concernant la documentation, envoyez une télécopie ou un courriel aux adresses suivantes :

Fax : +49 9131- 98 2176

Courriel : Courriel : docu.motioncontrol@siemens.com

Vous trouverez en annexe un formulaire de réponse par télécopie.

Adresse Internet pour SINUMERIK

http://www.siemens.com/sinumerik

Manuels de programmation "Notions de base" et "Notions complémentaires"

Les descriptions relatives à la programmation CN sont réparties sur deux manuels :

1. Notions de base

   Le manuel de programmation "Notions de base" s'adresse aux techniciens utilisant la machine-outil et suppose d'avoir la connaissance préalable des opérations de perçage, de fraisage et de tournage. Les instructions et commandes, conformes à la norme DIN 66025, sont illustrées par des exemples de programmation simples.

2. Notions complémentaires

   Le manuel de programmation "Notions complémentaires" s'adresse aux technologues connaissant toutes les possibilités de programmation. Grâce à un langage de programmation spécifique, la commande SINUMERIK permet d'élaborer un programme pièce complexe (par exemple : surfaces gauches, coordination entre canaux...) et dispense le technologue d'une programmation fastidieuse.

Disponibilité des éléments de langage

Tous les éléments de langage CN décrits dans le présent manuel sont disponibles pour SINUMERIK 840D sl. La disponibilité pour SINUMERIK 828D est indiquée dans la colonne "828D" de "Liste des instructions (Page 719)".

Table des matières

Avant-propos **3**

1 Programmation CN flexible **15**

1.1 Variables 15
1.1.1 Informations générales sur les variables 15
1.1.2 Variables système 16
1.1.3 Variables utilisateur prédéfinies : Paramètre de calcul (R) 18
1.1.4 Variables utilisateur prédéfinies : variables Link 20
1.1.5 Définition de variables utilisateur (DEF) 22
1.1.6 Redéfinition de variables système, variables utilisateur et instruction de langage CN (REDEF) 28
1.1.7 Attribut : Valeur d'initialisation 31
1.1.8 Attribut : valeurs limites (LLI, ULI) 34
1.1.9 Attribut : unité physique (PHU) 35
1.1.10 Attribut : droits d'accès (APR, APW, APRP, APWP, APRB, APWB) 38
1.1.11 Vue d'ensemble des attributs définissables et redéfinissables 43
1.1.12 Définition et initialisation de variables de tableau (DEF, SET, REP) 44
1.1.13 Définition et initialisation de variables de tableau (DEF, SET, REP) : Informations complémentaires 49
1.1.14 Types de données 52
1.2 Programmation indirecte 53
1.2.1 Programmation indirecte d'adresses 53
1.2.2 Programmation indirecte de codes G 56
1.2.3 Programmation indirecte d'attributs de position (PB) 57
1.2.4 Programmation indirecte de lignes de programme pièce (EXECSTRING) 60
1.3 Fonctions de calcul 61
1.4 Opérations relationnelles et opérations logiques 64
1.5 Correction de la précision pour les erreurs relationnelles (TRUNC) 66
1.6 Minimum, maximum et plage de variables (MINVAL, MAXVAL, BOUND) 68
1.7 Priorité des opérations 70
1.8 Conversions de types possibles 71
1.9 Opérations sur les chaînes de caractères 72
1.9.1 Conversion de type en type STRING (AXSTRING) 73
1.9.2 Conversion de type à partir de STRING (NUMBER, ISNUMBER, AXNAME) 74
1.9.3 Concaténation de chaînes de caractères (<<) 75
1.9.4 Conversion en minuscules/majuscules (TOLOWER, TOUPPER) 77
1.9.5 Déterminer la longueur d'une chaîne de caractères (STRLEN) 78
1.9.6 Recherche de caractères/chaînes de caractères dans une chaîne (INDEX, RINDEX, MINDEX, MATCH) 78
1.9.7 Sélection d'une chaîne de caractères partielle (SUBSTR) 80
1.9.8 Sélection d'un caractère individuel (STRINGVAR, STRINGFELD) 80
1.10 Sauts de programme 82
1.10.1 Retour au début du programme (GOTOS) 82
1.10.2 Sauts de programme sur repères de saut (IF, GOTOB, GOTOF, GOTO, GOTOC) 83
1.10.3 Saut de programme (CASE ... OF ... DEFAULT ...) 86
1.11 Répétition de sections de programme (REPEAT, REPEATB, ENDLABEL, P) 88

1.12 Structures de contrôle ... 95
1.12.1 Boucle de programme avec alternative (IF, ELSE, ENDIF) ... 96
1.12.2 Boucle de programme sans fin (LOOP, ENDLOOP) ... 98
1.12.3 Boucle de comptage (FOR ... TO ..., ENDFOR) ... 99
1.12.4 Boucle de programme avec condition en début de boucle (WHILE, ENDWHILE) ... 100
1.12.5 Boucle de programme avec condition en fin de boucle (REPEAT, UNTIL) ... 101
1.12.6 Exemple de programme avec des structures de contrôle imbriquées ... 102
1.13 Coordination de programmes (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) ... 103
1.14 Routine d'interruption (ASUP) ... 109
1.14.1 Fonction d'une routine d'interruption ... 109
1.14.2 Création d'une routine d'interruption ... 110
1.14.3 Affectation et démarrage d'une routine d'interruption (SETINT, PRIO, BLSYNC) ... 111
1.14.4 Désactivation/activation de l'affectation d'une routine d'interruption (DISABLE, ENABLE) ... 113
1.14.5 Suppression d'une affectation de routine d'interruption (CLRINT) ... 114
1.14.6 Retrait rapide du contour (SETINT LIFTFAST, ALF) ... 115
1.14.7 Sens de retrait rapide du contour ... 117
1.14.8 Séquence de déplacement avec des routines d'interruption ... 120
1.15 Permutation d'axe, permutation de broche (RELEASE, GET, GETD) ... 121
1.16 Transmettre un axe à un autre canal (AXTOCHAN) ... 126
1.17 Activation des paramètres machine (NEWCONF) ... 128
1.18 Ecriture d'un fichier (WRITE) ... 129
1.19 Suppression d'un fichier (DELETE) ... 132
1.20 Lecture des lignes d'un fichier (READ) ... 134
1.21 Vérification de l'existence d'un fichier (ISFILE) ... 137
1.22 Lecture des informations fichier (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) ... 139
1.23 Calcul du total de contrôle d'un tableau (CHECKSUM) ... 142
1.24 Arrondissement (ROUNDUP) ... 144
1.25 Technique des sous-programmes ... 145
1.25.1 Généralités ... 145
1.25.1.1 Sous-programme ... 145
1.25.1.2 Nom de sous-programme ... 146
1.25.1.3 Imbrication de sous-programmes ... 147
1.25.1.4 Chemin de recherche ... 148
1.25.1.5 Paramètres formels et paramètres effectifs ... 149
1.25.1.6 Transfert de paramètres ... 150
1.25.2 Définition d'un sous-programme ... 152
1.25.2.1 Sous-programme sans transfert de paramètres ... 152
1.25.2.2 Sous-programme avec transfert de paramètres Call-by-Value (PROC) ... 153
1.25.2.3 Sous-programme avec transfert de paramètres Call-by-Reference (PROC, VAR) ... 155
1.25.2.4 Sauvegarde des fonctions G modales (SAVE) ... 157
1.25.2.5 Inhibition de l'exécution bloc par bloc (SBLOF, SBLON) ... 158
1.25.2.6 Inhibition de l'affichage du bloc courant (DISPLOF, DISPLON, ACTBLOCNO) ... 164
1.25.2.7 Identifier les sous-programmes avec prétraitement (PREPRO) ... 168
1.25.2.8 Retour de sous-programme M17 ... 169
1.25.2.9 Retour de sous-programme RET ... 170
1.25.2.10 Retour paramétrable dans les sous-programmes (RET ...) ... 171
1.25.3 Appel d'un sous-programme ... 178
1.25.3.1 Appel de sous-programme sans transfert de paramètres ... 178
1.25.3.2 Appel d'un sous-programme avec transfert de paramètres (EXTERN) ... 180
1.25.3.3 Nombre de répétitions de programme (P) ... 183

1.25.3.4 Appel modal d'un sous-programme (MCALL) 185
1.25.3.5 Appel indirect de sous-programme (CALL) 187
1.25.3.6 Appel indirect d'un sous-programme avec indication de la section de programme à exécuter (CALL BLOCK ... TO ...) 188
1.25.3.7 Appel indirect d'un programme programmé en langage ISO (ISOCALL) 190
1.25.3.8 Appel de sous-programme avec indication de chemin et paramètres (PCALL) 191
1.25.3.9 Extension du chemin de recherche pour l'appel de sous-programmes (CALLPATH) 192
1.25.3.10 Exécution de sous-programme externe (EXTCALL) 193
1.25.4 Cycles 197
1.25.4.1 Cycles : paramétrage de cycles utilisateur 197
1.26 Macroprogrammation (DEFINE ... AS) 201

2 Gestion des fichiers et programmes 205
2.1 Mémoire de programmes 205
2.2 Mémoire de travail (CHANDATA, COMPLETE, INITIAL) 210
2.3 Instruction structurelle dans l'éditeur Step (SEFORM) 213

3 Zones de protection 215
3.1 Définition des zones de protection (CPROTDEF, NPROTDEF) 215
3.2 Activation/désactivation des zones de protection (CPROT, NPROT) 219
3.3 Contrôle des violations de zone de protection, des limitations de la zone de travail et des fins de course logiciels (CALCPOSI) 223

4 Instructions de déplacement spéciales 231
4.1 Accostage de positions codées (CAC, CIC, CDC, CACP, CACN) 231
4.2 Interpolation de type spline (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) 233
4.3 Groupe spline (SPLINEPATH) 245
4.4 Compactage de bloc CN (COMPON, COMPCURV, COMPCAD, COMPOF) 247
4.5 Interpolation polynomiale (POLY, POLYPATH, PO, PL) 250
4.6 Référence réglable de trajectoire (SPATH, UPATH) 256
4.7 Mesure avec palpeur à déclenchement (MEAS, MEAW) 259
4.8 Fonction de mesure étendue (MEASA, MEAWA, MEAC) (option) 262
4.9 Fonctions spéciales pour l'utilisateur OEM (OEMIPO1, OEMIPO2, G810 à G829) 272
4.10 Réduction de l'avance avec décélération aux angles (FENDNORM, G62, G621) 273
4.11 Critère programmable de fin de déplacement (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) 274
4.12 Jeu de paramètres servo programmable (SCPARA) 278

5 Transformations de coordonnées (FRAMES) 279
5.1 Transformation de coordonnées par variables frames 279
5.1.1 Variables frames prédéfinies ($P_BFRAME, $P_IFRAME, $P_PFRAME, $P_ACTFRAME) 281
5.2 Affectation de valeurs à des variables frames / frames 286
5.2.1 Affectation de valeurs directes (valeur d'axe, angle, échelle) 286
5.2.2 Lecture et modification de composantes de frames (TR, FI, RT, SC, MI) 289
5.2.3 Combinaison de frames complets 290
5.2.4 Définition de nouveaux frames (DEF FRAME) 292
5.3 Décalage grossier et décalage fin (CFINE, CTRANS) 293
5.4 Décalage d'origine externe 295
5.5 Décalage Preset (PRESETON) 296

5.6 Calcul d'un frame à partir de 3 points mesurés dans l'espace (MEAFRAME) ... 298
5.7 Frames à définition globale pour NCU ... 302
5.7.1 Frames spécifiques à un canal ($P_CHBFR, $P_UBFR) ... 303
5.7.2 Frames actifs dans un canal ... 304

6 Transformations ... 309

6.1 Programmation générale des types de transformation ... 309
6.1.1 Mouvements d'orientation lors des transformations ... 312
6.1.2 Aperçu de la transformation d'orientation TRAORI ... 315
6.2 Transformation trois, quatre et cinq axes (TRAORI) ... 317
6.2.1 Corrélations générales de tête de fraisage de type cardan ... 317
6.2.2 Transformation trois, quatre et cinq axes (TRAORI) ... 320
6.2.3 Variantes de programmation d'orientation et position initiale (ORIRESET) ... 321
6.2.4 Programmation de l'orientation de l'outil (A..., B..., C..., LEAD, TILT) ... 323
6.2.5 Fraisage en bout (fraisage 3D A4, B4, C4, A5, B5, C5) ... 329
6.2.6 Référence des axes d'orientation (ORIWKS, ORIMKS) ... 331
6.2.7 Programmation des axes d'orientation (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) ... 333
6.2.8 Programmation de l'orientation le long d'une enveloppe de cône (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) ... 335
6.2.9 Définition de l'orientation de deux points de contact (ORICURVE, PO[XH]=, PO[YH]=, PO[ZH]=) ... 339
6.3 Polynômes d'orientation (PO[angles], PO[coordonnée]) ... 341
6.4 Rotations de l'orientation de l'outil (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) ... 343
6.5 Orientations par rapport à la trajectoire ... 346
6.5.1 Types d'orientation par rapport à la trajectoire ... 346
6.5.2 Rotation de l'orientation de l'outil relative à la trajectoire (ORIPATH, ORIPATHS, angle de rotation) ... 348
6.5.3 Interpolation relative à la trajectoire de la rotation de l'outil (ORIROTC, THETA) ... 349
6.5.4 Lissage du tracé d'orientation (ORIPATHS A8=, B8=, C8=) ... 352
6.6 Compactage de l'orientation (COMPON, COMPCURV, COMPCAD) ... 353
6.7 Lissage du tracé d'orientation (ORISON, ORISOF) ... 357
6.8 Transformation cinématique ... 359
6.8.1 Opérations de fraisage sur des pièces de tournage (TRANSMIT) ... 359
6.8.2 Transformation de surface latérale de cylindre (TRACYL) ... 362
6.8.3 Axe oblique (TRAANG) ... 371
6.8.4 Programmation d'un axe oblique (G05, G07) ... 374
6.9 Déplacement PTP cartésien ... 376
6.9.1 PTP avec TRANSMIT ... 381
6.10 Conditions marginales pour l'activation d'une transformation ... 385
6.11 Désactivation d'une transformation (TRAFOOF) ... 386
6.12 Transformations concaténées (TRACON, TRAFOOF) ... 387

7 Corrections d'outils ... 389

7.1 Mémoire de correcteurs ... 389
7.2 Corrections additives ... 392
7.2.1 Sélection des corrections additives (DL) ... 392
7.2.2 Définition des valeurs d'usure et de réglage ($TC_SCPxy[t,d], $TC_ECPxy[t,d]) ... 394
7.2.3 Effacer les corrections additives (DELDL) ... 395
7.3 Correcteur d'outil : Interventions spéciales ... 396
7.3.1 Application de la fonction miroir aux longueurs d'outil ... 398
7.3.2 Exploitation du signe de l'usure ... 399

7.3.3 Système de coordonnées de l'usinage actif (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) 400
7.3.4 Longueurs d'outil et changement de plan 403
7.4 Correction d'outil en ligne (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) 404
7.5 Activation des corrections d'outil 3D (CUT3DC..., CUT3DF...) 409
7.5.1 Activation des corrections d'outil 3D (CUT3DC, CUT3DF, CUT3DFS, CUT3DFF, ISD) 409
7.5.2 Correction d'outil 3D : fraisage périphérique, fraisage en bout 411
7.5.3 Correction d'outil 3D : formes et données d'outil pour le fraisage en bout 413
7.5.4 Correction d'outil 3D : Correction sur la trajectoire, courbure de la trajectoire, profondeur de pénétration (CUT3DC, ISD) 415
7.5.5 Correction d'outil 3D : angles rentrants/angles saillants et méthode du point d'intersection (G450/G451) 417
7.5.6 Correction d'outil 3D : fraisage périphérique 3D avec surfaces de délimitation 419
7.5.7 Correction d'outil 3D : prise en compte d'une surface de délimitation (CUT3DCC, CUT3DCCD) 419
7.6 Orientation de l'outil (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) 423
7.7 Numéros D libres, numéro de tranchant 429
7.7.1 Numéros D libres, numéro de tranchant (adresse CE) 429
7.7.2 Libre affectation des numéros D : contrôle des numéros D (CHKDNO) 430
7.7.3 Libre affectation des numéros D : modification des numéros D (GETDNO, SETDNO) 431
7.7.4 Libre affectation des numéros D : détermination du numéro T correspondant au numéro D prescrit (GETACTTD) 432
7.7.5 Libre affectation des numéros D : numéros D déclarés non valides (DZERO) 433
7.8 Cinématique d'un organe porte-outil 434
7.9 Correction de longueur d'outil pour organes porte-outils orientables (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) 439
7.10 Correction de longueur d'outil en ligne (TOFFON, TOFFOF) 443
7.11 Modification des données de tranchant des outils orientables (CUTMOD) 446

8 Modes de déplacement 453

8.1 Positionnement tangentiel (TANG, TANGON, TANGOF, TLIFT, TANGDEL) 453
8.2 Variation de l'avance (FNORM, FLIN, FCUB, FPO) 460
8.3 Exécution du programme avec une mémoire tampon (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE) 465
8.4 Sections de programme interruptibles sous conditions (DELAYFSTON, DELAYFSTOF) 468
8.5 Interdire des positions de programme pour SERUPRO (IPTRLOCK, IPTRUNLOCK) 473
8.6 Réaccostage du contour (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) 476
8.7 Correction du pilotage des déplacements 486
8.7.1 Correction de l'à-coup en pourcentage (JERKLIM) 486
8.7.2 Correction de la vitesse en pourcentage (VELOLIM) 487
8.7.3 Exemple de programme pour JERKLIM et VELOLIM 490
8.8 Tolérance de contour/orientation programmable (CTOL, OTOL, ATOL) 491
8.9 Tolérance pour déplacements G0 (STOLF) 495

9 couplages d'axes 497

9.1 Déplacements conjugués (TRAILON, TRAILOF) 497
9.2 Tables de courbes (CTAB) 501
9.2.1 Définition de tables de courbes (CTABDEF, CATBEND) 502
9.2.2 Vérification de l'existence d'une table de courbes (CTABEXISTS) 508
9.2.3 Suppression d'une table de courbes (CTABDEL) 508

9.2.4 Verrouillage de tables de courbes contre la suppression et l'écrasement (CTABLOCK, CTABUNLOCK) ... 510
9.2.5 Tables de courbes : détermination des propriétés de la table (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD) ... 511
9.2.6 Lecture des valeurs de la table de courbes (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) ... 513
9.2.7 Tables de courbes : Vérification de l'utilisation des ressources (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) ... 518
9.3 Couplage de deux axes par valeur pilote (LEADON, LEADOF) ... 520
9.4 Réducteur électronique (EG) ... 526
9.4.1 Définition d'un réducteur électronique (EGDEF) ... 526
9.4.2 Activation du réducteur électronique (EGON, EGONSYN, EGONSYNE) ... 528
9.4.3 Désactivation du réducteur électronique (EGOFS, EGOFC) ... 531
9.4.4 Suppression de la définition d'un réducteur électronique (EGDEL) ... 532
9.4.5 Avance par tour (G95) / réducteur électronique (FPR) ... 533
9.5 Broche synchrone ... 534
9.5.1 Broche synchrone : Programmation (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) ... 535
9.6 Couplage maître/esclave (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) ... 546
10 Actions synchrones au déplacement ... 551
10.1 Notions de base ... 551
10.1.1 Domaine de validité et ordre d'exécution (ID, IDS) ... 553
10.1.2 Contrôle cyclique de la condition (WHEN, WHENEVER, FROM, EVERY) ... 555
10.1.3 Actions (DO) ... 557
10.2 Opérateurs pour les conditions et actions ... 558
10.3 Variables d'exécution pour actions synchrones ... 560
10.3.1 Variables système ... 560
10.3.2 Conversion de type implicite ... 562
10.3.3 Variables GUD ... 563
10.3.4 Descripteur d'axe par défaut (NO_AXIS) ... 565
10.3.5 Marque d'action synchrone ($AC_MARKER[n]) ... 566
10.3.6 Paramètre d'action synchrone ($AC_PARAM[n]) ... 566
10.3.7 Paramètre de calcul ($R[n]) ... 567
10.3.8 Lecture et écriture des paramètres machine CN et des données de réglage CN ... 568
10.3.9 Variables de temporisation ($AC_Timer[n]) ... 570
10.3.10 Variables FIFO ($AC_FIFO1[n] ... $AC_FIFO10[n]) ... 571
10.3.11 Renseignement sur les types de bloc dans l'interpolateur ($AC_BLOCKTYPE, $AC_BLOCKTYPEINFO, $AC_SPLITBLOCK) ... 573
10.4 Actions dans des actions synchrones ... 576
10.4.1 Actions possibles dans les actions synchrones ... 576
10.4.2 Sortie de fonctions auxiliaires ... 579
10.4.3 Activation du blocage de la lecture (RDISABLE) ... 580
10.4.4 Annulation de l'arrêt du prétraitement des blocs (STOPREOF) ... 581
10.4.5 Effacement de la distance restant à parcourir (DELDTG) ... 582
10.4.6 Définition d'un polynôme (FCTDEF) ... 584
10.4.7 Fonction synchrone (SYNFCT) ... 586
10.4.8 Régulation d'écartement avec correction limitée ($AA_OFF_MODE) ... 590
10.4.9 Correction d'outil en ligne (FTOC) ... 593
10.4.10 Correction en ligne de la longueur d'outil ($AA_TOFF) ... 596
10.4.11 Déplacements de positionnement ... 598
10.4.12 Positionnement d'un axe (POS) ... 599
10.4.13 Position dans la zone de référence prescrite (POSRANGE) ... 601

10.4.14 Démarrer/arrêter un axe (MOV) 602
10.4.15 Permutation d'axe (RELEASE, GET) 603
10.4.16 Avance axiale (FA) 607
10.4.17 Fins de course logiciels 607
10.4.18 Coordination d'axe 608
10.4.19 Préréglage des mémoires de valeurs réelles (PRESETON) 609
10.4.20 Déplacement de broches 610
10.4.21 Déplacements conjugués (TRAILON, TRAILOF) 611
10.4.22 Couplage par valeur pilote (LEADON, LEADOF) 613
10.4.23 Mesure (MEAWA, MEAC) 616
10.4.24 Initialisation de variables de tableau (SET, REP) 617
10.4.25 Activer/effacer des marques d'attente (SETM, CLEARM) 618
10.4.26 Réaction aux erreurs (SETAL) 619
10.4.27 Accostage d'une butée (FXS, FXST, FXSW, FOCON, FOCOF) 620
10.4.28 Détermination de l'angle de la tangente à la trajectoire dans les actions synchrones 623
10.4.29 Détermination de la correction de vitesse courante 623
10.4.30 Traitement de l'utilisation par rapport à la durée des actions synchrones 624
10.5 Cycles technologiques 626
10.5.1 Variable de contexte ($P_TECCYCLE) 629
10.5.2 Paramètres Call-by-Value 630
10.5.3 Initialisation de paramètres par défaut 630
10.5.4 Pilotage du traitement de cycles technologiques (ICYCOF, ICYCON) 631
10.5.5 Cascadages de cycles technologiques 632
10.5.6 Cycles technologiques en actions synchrones non modales 632
10.5.7 Structures de contrôle (IF) 632
10.5.8 Instructions de saut (GOTO, GOTOF, GOTOB) 633
10.5.9 Bloquer, débloquer, remettre à zéro (LOCK, UNLOCK, RESET) 634
10.6 Effacer une action synchrone (CANCEL) 636
10.7 Comportement de la commande dans des états de fonctionnement donnés 637

11 Oscillation 641

11.1 Oscillation asynchrone (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) 641
11.2 Oscillation commandée par des actions synchrones (OSCILL) 647

12 Poinçonnage et grignotage 655

12.1 Activation, désactivation 655
12.1.1 Activation ou désactivation du poinçonnage et du grignotage (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) 655
12.2 Préparation automatique du déplacement 660
12.2.1 Segmentation du déplacement dans le cas d'axes à interpolation 663
12.2.2 Segmentation du déplacement dans le cas d'axes individuels 665

13 Rectification 667

13.1 Surveillance d'outil spécifique à la rectification dans le programme pièce (TMON, TMOF) 667

14 Autres fonctions 669

14.1 Fonctions d'axe (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) 669
14.2 Axes géométriques permutables (GEOAX) 672
14.3 Conteneur d'axes (AXCTSWE, AXCTSWED) 677
14.4 Attente de la position d'axe définitive (WAITENC) 681
14.5 Contrôler le langage CN existant (STRINGIS) 683
14.6 Lecture de l'appel de la fonction ISVAR et des paramètres machine d'indice de tableau 687
14.7 Apprentissage de compensations (QECLRNON, QECLRNOF) 689

14.8 Appel interactif des fenêtres depuis le programme pièce (MMC) ... 691
14.9 Temps d'exécution du programme / Compteur de pièces ... 692
14.9.1 Temps d'exécution du programme / Compteur de pièces (vue d'ensemble) ... 692
14.9.2 Temps d'exécution de programme ... 692
14.9.3 Compteurs de pièces ... 697
14.10 Alarmes (SETAL) ... 699
15 Programmes de chariotage personnalisés ... 701
15.1 Fonctions additionnelles pour le chariotage ... 701
15.2 Création d'une table de contour (CONTPRON) ... 702
15.3 Création d'une table de contour codée (CONTDCON) ... 708
15.4 Détermination de l'intersection entre deux éléments de contour (INTERSEC) ... 712
15.5 Exécution des éléments de contour d'une table bloc par bloc (EXECTAB) ... 714
15.6 Calcul de données de cercles (CALCDAT) ... 715
15.7 Désactivation de la préparation du contour (EXECUTE) ... 717
16 Tableaux ... 719
16.1 Liste des instructions ... 719
A Annexes ... 791
A.1 Liste des abréviations ... 791
A.2 Remarques sur la documentation ... 797
A.3 Vue d'ensemble de la documentation ... 799
Glossaire ... 801
Indice ... 823

Programmation CN flexible 1

1.1 Variables

1.1.1 Informations générales sur les variables

L'utilisation de variables, en particulier en association avec des fonctions de calcul et des structures de contrôle, permet de réaliser des programmes pièce et des cycles de façon extrêmement flexible. A cet effet, le système met à disposition trois différents types de variables :

- Variables système

  Il s'agit de variables définies dans le système et mises à disposition de l'utilisateur avec une signification précise prédéfinie. Elles peuvent également être lues et écrites par le logiciel système. Exemple : Paramètres machine

  La signification d'une variable système est prédéfinie dans une large mesure de manière précise par le système, dans les propriétés. L'utilisateur a cependant la possibilité, dans une faible mesure, d'adapter les propriétés en les redéfinissant. Voir ""

- Variables utilisateur

  Il s'agit de variables sont la signification n'est pas connue par le système et qui ne sont pas non plus exploitées par ce dernier. Leur signification est exclusivement définie par l'utilisateur.

  Les variables utilisateur sont réparties en :

  – Variables utilisateur prédéfinies

    Il s'agit de variables déjà définies dans le système, dont l'utilisateur n'a plus qu'à définir le nombre aux moyen de paramètres machine spécifiques. L'utilisateur a la possibilité d'adapter dans une large mesure les propriétés de ces variables. Voir "".

  – Variables définies par l'utilisateur

    Il s'agit de variables définies exclusivement par l'utilisateur et que le système ne créée que lors de l'exécution. Leur nombre, type de données, visibilité ainsi que toutes les autres propriétés sont définies exclusivement par l'utilisateur.

    Voir ""

1.1.2 Variables système

Il s'agit de variables prédéfinies dans le système, qui permettent d'accéder, dans les programmes pièce et les cycles, aux paramétrages actuel de la commande, tel qu'états de la machine, de la commande et du processus.

Variables de prétraitement

Il s'agit de variables système qui sont lues et écrites dans le contexte du prétraitement, autrement dit au moment de l'interprétation du bloc de programme pièce dans lequel la variable système est programmée. Les variables de prétraitement ne déclenchent pas d'arrêt de prétraitement.

Variables d'exécution

Il s'agit de variables système qui sont lues ou écrites dans le contexte de l'exécution, autrement dit au moment de l'exécution du bloc de programme pièce dans lequel la variable système est programmée. Les variables d'exécution sont des :

- variables système pouvant être programmées dans des actions de synchronisation (lecture/écriture),
- variables système pouvant être programmées dans le programme pièce et capables de déclencher un arrêt de prétraitement (lecture/écriture),
- variables système pouvant être programmées dans le programme pièce, et dont la valeur est déterminée durant le prétraitement, mais n'est écrite que lors de l'exécution (synchrones à l'exécution : écriture seule)

Systématique de préfixe

Pour l'identification spécifique de variables système, leur nom est normalement précédé d'un préfixe formé par le caractère $, suivi d'une ou de deux lettres et d'un trait de soulignement :

$ + 1. Lettre	Signification : Type de paramètre
Variables système qui sont lues et écrites durant le prétraitement des blocs	
$M	Paramètres machine[1)]
$S	Données de réglage, zones de protection [1)]
$T	Paramètres de gestion des outils
$P	Valeurs programmées
$C	Variables des cycles enveloppes ISO
$O	Données optionnelles
R	Paramètres R (paramètres de calcul) [2)]
Variables système qui sont lues / écrites durant l'exécution des blocs	
$$M	Paramètres machine[1)]
$$S	Données de réglage [1)]
$A	Données d'exécution courantes
$V	Données servo

$ + 1. Lettre	Signification : Type de paramètre
$R	Paramètres R (paramètres de calcul) 2)
1) En cas d'utilisation de paramètres machine et de donnée de réglage comme variables d'exécution dans le programme pièce / cycle, le préfixe est formé d'un caractère $. En cas d'utilisation comme variable d'exécution dans les actions synchrones, le préfixe est formé de deux caractères $. 2) En cas d'utilisation d'un paramètre R comme variable d'exécution dans le programme pièce / cycle, aucun préfixe n'est spécifié, p. ex. R10. En cas d'utilisation comme variable d'exécution dans une action synchrone, le préfixe est formé d'un caractère $, p. ex. $R10.	

2. Lettre	Signification : Visibilité
N	Variable globale NCK (**N**CK)
C	Variable spécifique au canal (**C**hannel)
A	Variable spécifique à l'axe (**A**xis)

Conditions marginales

Exception dans la systématique des préfixes

Les variables système suivantes divergent de la systématique des préfixes indiqués ci-dessus :

- $TC_... : Le deuxième caractère C ne désigne pas une variable spécifique au canal, mais une variable système spécifique au porte-outils (TC = Tool Carrier)
- $P_ ... : Variables système spécifiques à un canal

Utilisation de paramètres machine et de données de réglage dans des actions synchrones

En cas d'utilisation des paramètres machine et des données de réglage dans des actions synchrones, le préfixe permet de déterminer si la lecture/écriture du paramètre machine ou de la donnée de réglage s'effectue de façon synchrone au prétraitement ou à l'exécution.

Si le paramètre reste inchangé pendant l'exécution, la lecture peut être effectuée de façon synchrone au prétraitement. Le préfixe du paramètre machine ou de la donnée de réglage s'écrit de ce fait avec un caractère $:

Code de programme

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

Si le paramètre est modifié pendant l'exécution, la lecture /écriture doit être effectuée de façon synchrone à l'exécution. Le préfixe du paramètre machine ou de la donnée de réglage s'écrit de ce fait avec deux caractères $:

Code de programme

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

Remarque

Ecriture de paramètres machine

Lors de l'écriture d'un paramètre machine ou d'une donnée de réglage, il faut s'assurer que le niveau d'accès actif pour l'exécution du programme pièce / cycle autorise la protection en écriture et que la prise d'effet du paramètre soit "IMMEDIATE".

Bibliographie

Pour obtenir la liste des propriétés de toutes les variables système, référez-vous aux :

/PGA1/ Tables de paramètres Variables système

1.1.3 Variables utilisateur prédéfinies : Paramètre de calcul (R)

Fonction

Les paramètres de calcul ou paramètres R correspondent à une variable utilisateur prédéfinie désigné par R et définie comme champ de type de données `REAL`. Pour des raisons historiques, outre la notation des paramètres R avec un indice de tableau, p. ex. `R[10]`, la notation sans indice de tableau, p. ex. `R10` est également autorisée.

En cas d'utilisation dans des actions synchrones, la lettre $ doit figurer en tête, p. ex. `$R10`.

Syntaxe

En cas d'utilisation comme variable de prétraitement :

```
R<n>
R[<expression>]
```

En cas d'utilisation comme variable d'exécution :

```
$R<n>
$R[<expression>]
```

Signification

`R` : Descripteur en cas d'utilisation comme variable de prétraitement, p. ex. dans le programme pièce

`$R` : Descripteur en cas d'utilisation comme variable d'exécution, p. ex. dans des actions synchrones

Type : REAL

Plage de valeurs : En notation non exponentielle :

± (0.000 0001 ... 9999 9999)

Nota :
Le nombre maximum de décimales autorisées est de 8

En notation exponentielle :

± ($1*10^{-300}$ ... $1*10^{+300}$)

Nota :

- Notation : <mantisse>EX<exposant>, p. ex. 8.2EX-3
- Le nombre maximum de caractères autorisés, y compris le signe et le point décimal, est de 10.

`<n>` : Numéro du paramètre R

Type : INT

Plage de valeurs : 0 - MAX_INDEX

Nota
MAX_INDEX résulte du nombre de paramètres R paramétrés :
MAX_INDEX = (MD28050 $MN_MM_NUM_R_PARAM) - 1

`<expression>` : Indice de tableau

L'indice de tableau peut être une expression quelconque, à condition que le résultat de cette expression puisse être converti dans le type de données `INT` (`INT`, `REAL`, `BOOL`, `CHAR`)

Exemple

Affectations de paramètres R et utilisation de paramètres R dans des fonctions mathématiques :

```
Code de programme          Commentaire
R0=3.5678                  ;   Affectation dans le prétraitement
R[1]=-37.3                 ;   Affectation dans le prétraitement
R3=-7                      ;   Affectation dans le prétraitement
$R4=-0.1EX-5               ;   Affectation dans l'exécution : R4 = -0.1 * 10^-5
$R[6]=1.874EX8             ;   Affectation dans l'exécution : R6 = 1.874 * 10^8
R7=SIN(25.3)               ;   Affectation dans le prétraitement

R[R2]=R10                  ;   Adressage indirect au moyen d'un paramètre R
R[(R1+R2)*R3]=5            ;   Adressage indirect au moyen d'une expression
                               mathématique
```

```
Code de programme              Commentaire

X=(R1+R2)                  ;   Déplacer l'axe X à la position résultant de la
                               somme de R1 et de R2
Z=SQRT(R1*R1+R2*R2)        ;   Déplacer l'axe Z à la position racine carrée (R1^2
                               + R2^2)
```

1.1.4 Variables utilisateur prédéfinies : variables Link

Fonction

Dans le cadre de la fonction "Link NCU", les variables Link permettent de réaliser l'échange cyclique de données entre des NCU mises en réseau. Elle autorisent un accès spécifique au format de données à la mémoire de variables Link. La taille et la structure de données de la mémoire de variables Link sont définies de manière spécifique à l'installation par l'utilisateur / le constructeur de la machine-outil.

Les variables Link sont des variables utilisateur globales du système, dont l'écriture et la lecture peuvent être effectuées dans des programmes pièce et des cycles à partir de toutes les NCU du groupe Link, à condition que la communication Link soit configurée. Contrairement aux variables utilisateur globales (GUD), l'utilisation des variables Link est également possible dans des actions synchrones.

Dans le cas d'installations ne disposant pas de Link NCU actif , les variables Link peuvent être utilisées localement dans la commande en plus des variables utilisateur globales (GUD), comme variables utilisateur globales supplémentaires.

Syntaxe

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

Signification

`$A_DLB` :	Variable Link pour le format de données BYTE (1 octet)	
	Type de données :	UINT
	Plage de valeurs :	0 ... 255
`$A_DLW` :	Variable Link pour le format de données WORD (2 octets)	
	Type de données :	INT
	Plage de valeurs :	-32768 ... 32767
`$A_DLD` :	Variable Link pour le format de données DWORD (4 octets)	
	Type de données :	INT
	Plage de valeurs :	-2147483648 ... 2147483647
`$A_DLR` :	Variable Link pour le format de données REAL (8 octets)	
	Type de données :	REAL
	Plage de valeurs :	±(2,2*10^{-308} … 1,8*10^{+308})

`<indice>` : Indice d'adresse en octets, calculé à partir du début de la mémoire de variables Link

Type de données : INT

Plage de valeurs : 0 - MAX_INDEX

Nota

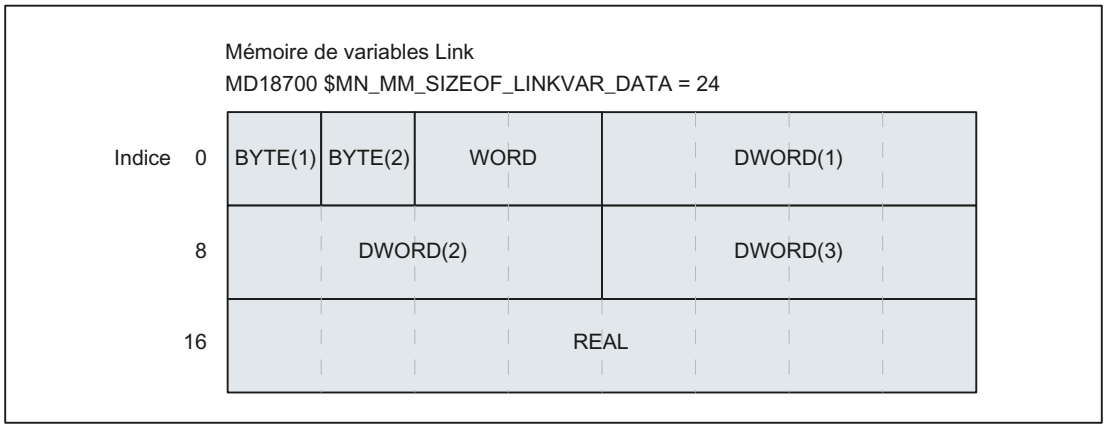
- MAX_INDEX résulte de la taille paramétrée pour la mémoire de variables Link : MAX_INDEX = (MD18700 $MN_MM_SIZEOF_LINKVAR_DATA) - 1
- Seule est autorisée la programmation d'indices pour lesquels les octets adressés dans la mémoire de variables Link correspondent à une limite de format de données ⇒
  Indice = n * octets, avec n = 0, 1, 2, ...
  - `$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, ...

Exemple

L'installation d'automatisation ne contient que 2 NCU (NCU1 et NCU2). L'axe machine AX2 déplacé comme axe Link par NCU2 est raccordé à NCU1.

NCU1 inscrit de façon cyclique la valeur réelle du courant ($VA_CURR) de l'axe AX2 dans la mémoire de variables Link. NCU2 lit de façon cyclique la valeur réelle du courant transmise par la communication Link et affiche l'alarme 61000 en cas de dépassement de la valeur limite.

La structure de données dans la mémoire des variables Link est représentée dans la figure suivante. La valeur réelle du courant est transmise au moyen de la valeur REAL.



NCU1
Durant la période d'appel de l'interpolateur pendant une action synchrone statique, NCU1 écrit de façon cyclique la valeur réelle du courant de l'axe AX2 dans la mémoire des variables Link, au moyen de la variable Link $A_DLR[16].

Code de programme

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

NCU2
Durant la période d'appel de l'interpolateur pendant une action synchrone statique, NCU2 lit de façon cyclique la valeur réelle du courant de l'axe AX2 dans la mémoire des variables Link, au moyen de la variable Link $A_DLR[16]. Si la valeur réelle du courant est supérieure à 23.0 A, l'alarme 61000 s'affiche.

Code de programme

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

1.1.5 Définition de variables utilisateur (DEF)

Fonction

L'instruction `DEF` permet à l'utilisateur de définir ses propres variables et leur affecter des valeurs. Pour les différencier des variables système, celles-ci sont désignées comme variables définies par l'utilisateur ou variables utilisateur (User Data).

Les catégories suivantes de variables utilisateurs sont disponibles en fonction du domaine de validité, autrement dit du domaine dans lequel la variable est visible :

- Variables utilisateur locales (LUD)

  Les variables utilisateur locales (LUD) sont des variables définies dans un programme pièce qui n'est pas le programme principal au moment de l'exécution. Elles sont déclarées à l'appel du programme pièce et supprimées à la fin du programme pièce ou du Reset CN. L'accès aux variables LUD peut uniquement être réalisé dans le programme pièce dans lequel ces variables sont définies.

- Variables utilisateur globales d'un programme (PUD)

  Les variables utilisateur globales d'un programme (PUD) sont des variables définies dans un programme pièce utilisé comme programme principal. Elles sont déclarées au lancement du programme pièce et supprimées à la fin du programme pièce ou du Reset CN. L'accès aux variables PUD peut être réalisé dans le programme principal et dans tous ses sous-programmes.

- Variables utilisateur globales (GUD)

  Les variables utilisateur globales (GUD) sont des variables globales d'une CN ou d'un canal, définies dans un bloc de données (SGUD, MGUD, UGUD, GUD4 ... GUD9) et conservées même après un Power On. L'accès aux variables GUD peut être réalisé dans tous les programmes pièce.

Les variables utilisateur doivent avoir été définies avant leur utilisation (lecture / écriture). Il convient de respecter les règles suivantes :

- Les variables GUD doivent être définies dans un fichier de définition, par ex. _N_DEF_DIR/_M_SGUD_DEF.
- Les variables PUD et LUD doivent être définies dans la section de définition d'un programme pièce.
- La définition des données doit être effectuée dans un bloc spécifique.
- Un seul type de données peut être utilisé par définition de données.
- Plusieurs variables du même type de données peuvent être définies par définition de données.

Syntaxe

```
DEF <domaine> <type> <arrêt prétraitement blocs> <instant
d'initialisation> <unité physique> <valeurs limites> <droits
d'accès> <nom>[<valeur_1>,<valeur_2>,<valeur_3>]=<valeur
d'initialisation>
```

Signification

`DEF` :	Instruction de définition de variables utilisateur GUD, PUD, LUD	
`<domaine>` :	Domaine de validité, uniquement significatif pour GUD :	
	`NCK` :	Variable utilisateur globale dans la CN
	`CHAN` :	Variable utilisateur globale dans le canal
`<type>` :	Type de données :	
	`INT` :	valeur entière avec signe
	`REAL` :	Nombre réel (LONG REAL selon IEEE)
	`BOOL` :	Valeur booléenne TRUE (1) / FALSE (0)
	`CHAR` :	Caractère ASCII
	`STRING[<Longueur maximale>]` :	Chaîne de caractères de longueur définie
	`AXIS` :	Descripteur d'axe/de broche
	`FRAME` :	Indications géométriques pour une transformation statique de coordonnées
	voir ""	
`<arrêt prétraitement blocs>` :	Arrêt de prétraitement des blocs, uniquement significatif pour GUD (facultatif)	
	`SYNR` :	Arrêt du prétraitement à la lecture
	`SYNW` :	Arrêt du prétraitement à l'écriture
	`SYNRW` :	Arrêt du prétraitement à la lecture/écriture
`<instant d'initialisation>` :	Instant auquel la variable est réinitialisée (facultatif)	
	`INIPO` :	Power On
	`INIRE` :	Fin du programme principal, Reset CN ou Power On

	INICF :	NewConfig ou fin du programme principal, Reset CN ou Power On
	PRLOC :	Fin du programme principal, Reset CN après une modification locale ou Power On
	voir ""	
<unité physique> :	Unité physique (facultative)	
	PHU <unité> :	
	voir ""	
<valeurs limites> :	Valeur limite inférieure et supérieure (facultative)	
	LLI <valeur limite> :	Valeur limite inférieure (lower limit)
	ULI <valeur limite> :	Valeur limite supérieure (upper limit)
	voir ""	
<droits d'accès> :	Droits d'accès en lecture / écriture aux variables GUD au moyen du programme pièce ou de BTSS (facultatif)	
	APRP <niveau de protection> :	Lecture : Programme pièce
	APWP <niveau de protection> :	Ecriture : programme pièce
	APRB <niveau de protection> :	Lecture : OPI
	APWB <niveau de protection> :	Ecriture : OPI
	Niveau de protection	Plage de valeurs : 0 ... 7
	voir ""	
<Nom> :	Nom de la variable	
	Nota	
	• 31 caractères au maximum • Les deux premiers caractères doivent être une lettre et/ou un caractère de soulignement. • Le caractère "$" est réservé aux variables système et ne doit pas être utilisé.	
[<valeur_1>, <valeur_2>, <valeur_3>] :	Indication de la dimension des tableaux pour des variables de tableaux comportant 1 à 3 dimensions au maximum (facultative)	
<valeur d'initialisation> :	Valeur d'initialisation (facultative)	
	voir ""	
	Pour l'initialisation de variables de tableaux :	
	voir ""	

Exemples

Exemple 1 : définitions de variables utilisateur dans le bloc de données pour constructeur de la machine-outil

Code de programme

```
%_N_MGUD_DEF                                          ; Bloc GUD : constructeur de la machine-outil
$PATH=/_N_DEF_DIR
DEF CHAN REAL PHU 24 LLI 0 ULI 10 STROM_1, STROM_2
;  Description
;  Définition de deux variables GUD : STROM_1, STROM_2
;  Domaine de validité : pour l'ensemble du canal
;  Type de données : REAL
;  Arrêt prétraitement blocs : non programmé => valeur par défaut = pas d'arrêt prétraitement blocs
;  Unité physique : 24 = [A]
;  Valeurs limites : inférieure = 0.0, supérieure = 10.0
;  Droits d'accès : non programmés => valeur par défaut = 7 = position 0 du commutateur à clé
;  Valeur d'initialisation : non programmée => valeur par défaut = 0.0

DEF NCK REAL PHU 13 LLI 10 APWP 3 APRP 3 APWB 0 APRB 2 ZEIT_1=12, ZEIT_2=45
;  Description
;  Définition de deux variables GUD : ZEIT_1, ZEIT_2
;  Domaine de validité : pour l'ensemble du NCK
;  Type de données : REAL
;  Arrêt prétraitement blocs : non programmé => valeur par défaut = pas d'arrêt prétraitement blocs
;  Unité physique : 13 = [s]
;  Valeurs limites : inférieure = 10.0, supérieure = non programmée => limite de la plage de
   définition
;  Droits d'accès :
;     Programme pièce : écriture/lecture = 3 = utilisateur final
;     OPI : écriture = 0 = Siemens, lecture = 3 = utilisateur final
;  Valeur d'initialisation : ZEIT_1 = 12.0, ZEIT_2 = 45.0

DEF NCK APWP 3 APRP 3 APWB 0 APRB 3 STRING[5] GUD5_NAME = "COUNTER"
;  Description
;  Définition d'une variable GUD : GUD5_NAME
;  Domaine de validité : pour l'ensemble du NCK
;  Type de données : STRING, 5 caractères au maximum
;  Arrêt prétraitement blocs : non programmé => valeur par défaut = pas d'arrêt prétraitement blocs
;  Unité physique : non programmée => valeur par défaut = 0 = pas d'unité physique
;  Valeurs limites : non programmées => limites de la plage de définition : inférieure = 0, supérieure
   = 255
;  Droits d'accès :
;     Programme pièce : écriture/lecture = 3 = utilisateur final
;     OPI : écriture = 0 = Siemens, lecture = 3 = utilisateur final
;  Valeur d'initialisation : "COUNTER"
M30
```

Exemple 2 : variables utilisateur globales et locales du programme (PUD / LUD)

Code de programme	Commentaire
PROC MAIN	; Programme principal
DEF INT VAR1	; Définition PUD
...	
SUB2	; appel de sous-programme
...	
M30	

Code de programme	Commentaire
PROC SUB2	; Sous-programme SUB2
DEF INT VAR2	; Définition LUD
...	
IF (VAR1==1)	; Lecture PUD
VAR1=VAR1+1	; Lecture et écriture PUD
VAR2=1	; Ecriture LUD
ENDIF	
SUB3	; Appel de sous-programme
...	
M17	

Code de programme	Commentaire
PROC SUB3	; Sous-programme SUB3
...	
IF (VAR1==1)	; Lecture PUD
VAR1=VAR1+1	; Lecture et écriture PUD
VAR2=1	; Erreur : LUD de SUB2 non connu
ENDIF	
...	
M17	

Exemple 3 : définition et utilisation de variables utilisateur du type de données AXIS

Code de programme	Commentaire
DEF AXIS ABSCISSE	; 1. Axe géométrique
DEF AXIS BROCHE	; Broche
...	
IF ISAXIS(1) == FALSE GOTOF SUITE	
ABSCISSE = $P_AXN1	
SUITE :	
...	
BROCHE=(S1)	1. Broche
OVRA[BROCHE]=80	; Correction de vitesse de rotation de broche = 80 %
BROCHE=(S3)	3. Broche

Contraintes

Variables utilisateur globales (GUD)

Les paramètres machines suivants doivent être pris en compte dans le cadre de la définition de variables utilisateur globales (GUD) :

N°	Descripteur : $MN_	Signification
11140	GUD_AREA_ SAVE_TAB	Sauvegarde supplémentaire pour blocs GUD
18118 [1)]	MM_NUM_GUD_MODULES	Nombre de fichiers GUD dans le système de fichiers actif
18120 [1)]	MM_NUM_GUD_NAMES_NCK	Nombre de noms GUD globaux
18130 [1)]	MM_NUM_GUD_NAMES_CHAN	Nombre de noms GUD spécifiques à un axe
18140 [1)]	MM_NUM_GUD_NAMES_AXIS	Nombre de noms GUD spécifiques à un axe
18150 [1)]	MM_GUD_VALUES_MEM	Espace mémoire pour valeurs GUD globales
18660 [1)]	MM_NUM_SYNACT_GUD_REAL	Nombre de variables GUD configurables du type de données REAL
18661 [1)]	MM_NUM_SYNACT_GUD_INT	Nombre de variables GUD configurables du type de données INT
18662 [1)]	MM_NUM_SYNACT_GUD_BOOL	Nombre de variables GUD configurables du type de données BOOL
18663 [1)]	MM_NUM_SYNACT_GUD_AXIS	Nombre de variables GUD configurables du type de données AXIS
18664 [1)]	MM_NUM_SYNACT_GUD_CHAR	Nombre de variables GUD configurables du type de données CHAR
18665 [1)]	MM_NUM_SYNACT_GUD_STRING	Nombre de variables GUD configurables du type de données STRING

[1)] Non disponible pour SINUMERIK 828D.

Variables utilisateur globales d'un programme (PUD)

IMPORTANT

Visibilité de variables utilisateur locales d'un programme (PUD)

Les variables utilisateur locales définies dans le programme principal (PUD) sont uniquement également visibles dans les sous-programmes, si le paramètre machine suivant est mis à 1 :

MD11120 $MN_LUD_EXTENDED_SCOPE = 1

Avec MD11120 = 0, les variables utilisateur locales définies dans le programme principal sont uniquement visibles dans le programme principal.

Utilisation dans tous les canaux d'une variable utilisateur globale NCK du type de données AXIS

Une variable utilisateur globale NCK du type de données `AXIS`, initialisée au moyen d'un descripteur lors de la définition dans le bloc de données, peut uniquement être utilisée dans différents canaux de la CN, si l'axe possède le même numéro de canal dans l'ensemble des canaux.

Si cela n'est pas le cas, la variable doit être chargée au début du programme pièce, ou la fonction AXNAME(...) (voir "") doit être utilisée, comme dans l'exemple suivant.

Code de programme	Commentaire
`DEF NCK STRING[5] ACHSE="X"`	`; Définition dans le bloc de données`
`N100 AX[AXNAME(ACHSE)]=111 G00`	`; Utilisation dans le programme pièce`

1.1.6 Redéfinition de variables système, variables utilisateur et instruction de langage CN (REDEF)

Fonction

L'instruction `REDEF` permet de modifier les attributs de variables système, variables utilisateur et instructions de langage CN. La condition de base nécessaire pour une redéfinition est qu'elle soit exécutée après la définition correspondante.

Lors d'une redéfinition, il n'est pas possible de modifier simultanément plusieurs attributs. Une instruction `REDEF` spécifique doit être programmée pour chaque attribut à modifier.

Dans le cas où plusieurs modifications d'attributs concurrentes sont programmées, la modification active est toujours la dernière.

Attributs redéfinissables

Voir "Vue d'ensemble des attributs définissables et redéfinissables (Page 43)"

Variables utilisateur locales (PUD / LUD)

Aucune redéfinition n'est autorisée pour des variables utilisateur locales (PUD / LUD).

Syntaxe

```
REDEF <nom> <arrêt prétraitement blocs>
REDEF <nom> <unité physique>
REDEF <nom> <valeurs limites>
REDEF <nom> <droits d'accès>
REDEF <nom> <instant d'initialisation>
REDEF <nom> <instant d'initialisation> <valeur d'initialisation>
```

Signification

`REDEF` : Instruction de redéfinition d'un attribut spécifique de variables système, variables utilisateur et instruction de langage CN

`<Nom>` : Nom d'une variable déjà définie ou d'une instruction de langage CN

`<arrêt prétraitement blocs>` : Arrêt du prétraitement des blocs

`SYNR` : Arrêt du prétraitement à la lecture

SYNW : Arrêt du prétraitement à l'écriture

SYNRW : Arrêt du prétraitement à la lecture/écriture

<unité physique> : Unité physique

PHU <unité> :

voir "Attribut : unité physique (PHU) (Page 35)"

Nota

Non redéfinissable pour :

- Variables système
- Données utilisateur globales (GUD)
- Types de données : BOOL, AXIS, STRING, FRAME

<valeurs limites> : Valeur limite inférieure et/ou supérieure

LLI <valeur limite> : Valeur limite inférieure (lower limit)

ULI <valeur limite> : Valeur limite supérieure (upper limit)

voir "Attribut : valeurs limites (LLI, ULI) (Page 34)"

Nota

Non redéfinissable pour :

- Variables système
- Données utilisateur globales (GUD)
- Types de données : BOOL, AXIS, STRING, FRAME

<droits d'accès> : Droits d'accès en lecture / écriture au moyen du programme pièce ou de BTSS

APX <niveau de protection> : Exécuter : Elément de langage CN

APRP <niveau de protection> : Lecture : Programme pièce

APWP <niveau de protection> : Ecriture : Programme pièce

APRB <niveau de protection> : Lecture : BTSS

APWB <niveau de protection> : Ecriture : BTSS

Niveau de protection : Plage de valeurs : 0 ... 7

voir "Attribut : droits d'accès (APR, APW, APRP, APWP, APRB, APWB) (Page 38)"

<instant d'initialisation> : Instant de réinitialisation de la variable

INIPO : PowerOn

INIRE : Fin du programme principal, Reset CN ou PowerOn

INICF : NewConfig ou fin du programme principal, Reset CN ou PowerOn

PRLOC : Fin du programme principal, Reset CN après une modification locale ou un PowerOn

voir "Attribut : Valeur d'initialisation (Page 31)"

<valeur d'initialisation> :	Valeur d'initialisation Pour redéfinir la valeur d'initialisation, il faut également toujours spécifier un instant d'initialisation (voir <instant d'initialisation>). voir "Attribut : Valeur d'initialisation (Page 31)" Pour l'initialisation de variables de tableaux : voir "" **Nota** Non redéfinissable pour : • variables système, à l'exception des données de réglage

Exemple

Redéfinitions de la variable système $TC_DPC1 dans le bloc de données pour le constructeur de la machine-outil

Code de programme

```
%_N_MGUD_DEF                                          ; Bloc GUD : constructeur de la machine-outil
$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)
;  Description
;  Droit d'accès en écriture : BTSS = niveau de protection 2, programme pièce = niveau de protection 3
;  Nota
;  En cas d'utilisation de fichiers ACCESS, la redéfinition des droits d'accès doit être transposée de
;  _N_MGUD_DEF en _N_MACCESS_DEF
;  Unité physique = [ % ]
;  Valeurs limites : inférieure = 0, supérieure = 200
;  La variable de tableau est initialisée avec les quatre valeurs lors d'un PowerOn
M30
```

Conditions

Granularité

Une redéfinition concerne toujours la variable complète, caractérisée de façon univoque par son nom. Il n'est pas possible, p. ex. pour des variables de tableau, d'attribuer différentes valeurs d'attribut à des éléments individuels de tableau.

1.1.7 Attribut : Valeur d'initialisation

Définition (DEF) de variables utilisateur

Lors de la définition, une valeur d'initialisation peut être prédéfinie pour les variables utilisateur suivantes :

- Variables utilisateur globales (GUD)
- Variables utilisateur globales d'un programme (PUD)
- Variables utilisateur locales (LUD)

Redéfinition (REDEF) de variables système et de variables utilisateur

Lors de la redéfinition, une valeur d'initialisation peut être prédéfinie pour les variables suivantes :

- Données système
  - Données de réglage
- Données utilisateur
  - Paramètre R
  - Variable à action synchrone ($AC_MARKER, $AC_PARAM, $AC_TIMER)
  - Variable GUD à action synchrone (SYG_xy[ ], avec x=R, I, B, A, C, S et y=S, M, U, 4, ..., 9)
  - Paramètre EPS
  - Données outil OEM
  - Données magasin OEM
  - Variables utilisateur globales (GUD)

Instant de réinitialisation

Lors de la redéfinition, il est possible de spécifier l'instant auquel la variable est réinitialisée, c'est-à-dire doit reprendre sa valeur d'initialisation :

- INIPO (Power On)

  La variable est réinitialisée lors d'un PowerOn.

- INIRE (Reset)

  La variable est réinitialisée lors d'un Reset CN, d'un Reset GMF, à la fin du programme pièce (M02 / M30) ou lors d'un PowerOn.

- INICF (NewConfig)

  La variable est réinitialisée lors de la requête NewConf via IHM, via une instruction du programme pièce NEWCONFIG ou via un Reset CN, un Reset GMF, ou à la fin du programme pièce (M02 / M30) ou lors d'un PowerOn.

- PRLOC (modification locale du programme)

  La variable est uniquement réinitialisée en cas de Reset CN, Reset GMF, à la fin du programme pièce (M02 / M30) si elle a été modifiée dans le cadre du programme pièce actuel.

L'attribut PRLOC ne doit être utilisé que dans le contexte de données de réglage programmables (voir le tableau suivant).

Tableau 1- 1 Données de réglage programmables

Numéro	Descripteur	Fonction G 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
43780	$SA_OSCILL_IS_ACTIVE	OS
43790	$SA_OSCILL_START_POS	OSB
1) Cette fonction G sollicite la donnée de réglage		

Contraintes

Valeur d'initialisation : variables utilisateur globales (GUD)

- Dans le cas de la variable utilisateur globale (GUD) avec le domaine de validité NCK, seul l'instant d'initialisation INIPO (Power On) peut être prédéfini.
- Dans le cas de variables utilisateur globales (GUD) avec le domaine de validité CHAN, il est possible de prédéfinir comme instant d'initialisation, outre INIPO (Power On), également INIRE (Reset) ou INICF (NewConfig).
- Dans le cas de variables utilisateur globales (GUD) avec le domaine de validité CHAN et l'instant d'initialisation INIRE (Reset) ou INICF (NewConfig), les variables sont seulement réinitialisées pour Reset CN, Reset GMF et NewConfig dans les canaux dans lesquels les événements spécifiés ont été déclenchés.

Valeur d'initialisation : type de données FRAME

Pour les variables du type de données FRAME, la spécification d'une valeur d'initialisation n'est pas autorisée. Les variables du type de données FRAME sont implicitement toujours initialisées avec le frame par défaut.

Valeur d'initialisation : type de données CHAR

Pour les variables du type de données CHAR, il est possible de programmer à la place du code ASCII (0...255), le caractère ASCII correspondant représenté entre guillemets, p. ex. "A"

Valeur d'initialisation : type de données STRING

Pour les variables du type de données STRING, la chaîne de caractères doit figurer entre guillemets, p. ex. : ...= "MACHINE_1"

Valeur d'initialisation : type de données AXIS

Pour les variables du type de données AXIS, le descripteur d'axe doit figurer entre guillemets pour la notation étendue d'adresse, p. ex. : ...=(X3)

Valeur d'initialisation : variable système

Pour les variables système, il n'est pas possible de prédéfinir par redéfinition des valeurs d'initialisation spécifiques à l'utilisateur. Les valeurs d'initialisation des variables système sont prédéfinies par le système. Il est par contre possible de modifier par redéfinition l'instant (INIRE, INICF) auquel la variable système est réinitialisée.

Valeur d'initialisation implicite : type de données AXIS

Pour les variables du type de données AXIS, la valeur d'initialisation implicite suivante est utilisée :

- Données système : "premier axe géométrique"
- GUD d'action synchrone (désignation : SYG_A*), PUD, LUD :
  descripteur d'axe à partir du paramètre machine : MD20082
  $MC_AXCONF_CHANAX_DEFAULT_NAME

Valeur d'initialisation implicite : données d'outil et de magasin

Pour les données d'outil et de magasin, il est possible de prédéfinir des valeurs d'initialisation à partir du paramètre machine suivant : MD17520
$MN_TOOL_DEFAULT_DATA_MASK

IMPORTANT
Synchronisme La synchronisation d'événements déclenchant la réinitialisation d'une variable globale avec lecture de cette variable à un autre endroit relève totalement de la responsabilité de l'utilisateur / du constructeur de la machine-outil.

1.1.8 Attribut : valeurs limites (LLI, ULI)

Une valeur limite supérieure et inférieure de la plage de définition peut uniquement être prédéfinie pour les types de données suivants :

- INT
- REAL
- CHAR

Définition (DEF) de variables utilisateur : valeurs limites et valeurs d'initialisation implicites

Si aucune valeur d'initialisation explicite n'est définie lors de la déclaration d'une variable utilisateur de l'un des types de données précités, la valeur d'initialisation implicite du type de données est affectée à la variable :

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

Si la valeur d'initialisation implicite se situe en dehors de la plage de définition déterminée par les valeurs limites programmées, la variable est initialisée avec la valeur limite la plus proche de la valeur d'initialisation implicite :

- Valeur d'initialisation implicite < valeur limite inférieure (LLI) ⇒ valeur d'initialisation = valeur limite inférieure
- Valeur d'initialisation implicite > valeur limite supérieure (ULI) ⇒ valeur d'initialisation = valeur limite supérieure

Exemples :

```
Code de programme          Commentaire
DEF REAL GUD1              ;  Valeur limite inférieure = limite de la plage de
                           ;  définition
                           ;  Valeur limite supérieure = limite de la plage de
                           ;  définition
                              Aucune valeur d'initialisation programmée
                              => valeur d'initialisation implicite = 0.0
DEF REAL LLI 5.0 GUD2      ;  Valeur limite inférieure = 5.0
                           ;  Valeur limite supérieure = limite de la plage de
                           ;  définition
                              => Valeur d'initialisation = 5.0
DEF REAL ULI -5 GUD3       ;  Valeur limite inférieure = limite de la plage de
                           ;  définition
                           ;  Valeur limite supérieure = -5.0
                              => Valeur d'initialisation = -5.0
```

Redéfinition (REDEF) de variables utilisateur : valeurs limites et valeurs réelles actuelles

Si lors de la redéfinition des valeurs limites d'une variable utilisateur, celles-ci sont modifiées de sorte à ce que la valeur réelle actuelle se situe en dehors de la nouvelle plage de définition, une alarme est émise et les valeurs limites ne sont pas appliquées.

Remarque

Redéfinition (REDEF) de variables utilisateur

Lors de la redéfinition des valeurs limites d'une variable utilisateur, il faut veiller à la modification cohérente des valeurs suivantes :

- Valeurs limites
- Valeur réelle
- Valeur d'initialisation lors de la redéfinition et de la réinitialisation automatique sur la base de INIPO, INIRE ou INICF

1.1.9 Attribut : unité physique (PHU)

Une unité physique peut uniquement être prédéfinie pour des variables possédant l'un des types de données suivants :

- INT
- REAL

Unités physiques programmables (PHU)

L'unité physique est spécifiée en tant que nombre à virgule fixe : `PHU <unité>`

Les unités physiques suivantes peuvent être programmées :

<Unité>	Signification	Unité physique
0	Pas d'unité physique	-
1	Position linéaire ou angulaire 1)2)	[mm], [pouce], [degré]
2	Position linéaire2)	[mm], [pouce]
3	Position angulaire	[degré]
4	Vitesse linéaire ou angulaire 1)2)	[mm/min], [pouce/min], [tr/min]
5	Vitesse linéaire2)	[mm/min]
6	Vitesse angulaire	[tr/min]
7	Accélération linéaire ou angulaire 1)2)	[m/s^2], [$pouce/s^2$], [tr/s^2]
8	Accélération linéaire2)	[m/s^2], [$pouce/s^2$]
9	Accélération angulaire	[tr/s^2]
10	A-coup linéaire ou angulaire 1)2)	[m/s^3], [$pouce/s^3$], [tr/s^3]
11	À-coup linéaire2)	[m/s^3], [$pouce/s^3$]
12	À-coup angulaire	[tr/s^3]
13	Temporisation	[s]
14	Gain du régulateur de position	[16.667/s]

<Unité>	Signification	Unité physique
15	Avance par tour 2)	[mm/tr], [pouce/tr]
16	Compensation de température 1)2)	[mm], [pouce]
18	Force	[N]
19	Masse	[kg]
20	Moment d'inertie 3)	[kgm^2]
21	Pour cent	[%]
22	Fréquence	[Hz]
23	Tension	[V]
24	Courant	[A]
25	Température	[°C]
26	Angle	[degré]
27	Gain	[1000/min]
28	Position linéaire ou angulaire 3)	[mm], [pouce], [degré]
29	Vitesse de coupe 2)	[m/min], [pied/min]
30	Vitesse périphérique 2)	[m/s], [pied/s]
31	Résistance	[Ohm]
32	Inductance	[mH]
33	Couple 3)	[Nm]
34	Constante du couple 3)	[Nm/A]
35	Gain du régulateur de courant	[V/A]
36	Gain de vitesse de rotation 3)	[Nm/(rad*s)]
37	Vitesse de rotation	[tr/min]
42	Puissance	[kW]
43	Courant faible	[µA]
46	Couple faible 3)	[µNm]
48	Pour mille	-
49	-	[Hz/s]
65	Débit	[l/min]
66	pression	[bar]
67	Volumes 3)	[cm^3]
68	Gain du système3)	[mm/(V*min)]
69	Gain du système pour régulateur de couple	[N/V]
155	Pas de vis 3)	[mm/tr], [pouce/tr]
156	Modification du pas de vis 3)	[mm/tr / tr], [pouce/tr / tr]
1) L'unité physique dépend du type d'axe : axe linéaire ou rotatif		

<Unité>	Signification	Unité physique
2) Commutation du système de coordonnées G70/G71(inch/métrique) Après une commutation du système de base (MD10240 $MN_SCALING_SYSTEM_IS_METRIC) avec G70/G71, **aucune** conversion des valeurs (valeur réelle, valeur par défaut et valeurs limites) n'est effectuée lors d'accès d'écriture/de lecture à des variables système et utilisateur relatives à des longueurs G700/G710(inch/métrique) Après une commutation du système de base (MD10240 $MN_SCALING_SYSTEM_IS_METRIC) avec G700/G710, **une** conversion des valeurs (valeur réelle, valeur par défaut et valeurs limites) est effectuée lors d'accès d'écriture/de lecture à des variables système et utilisateur relatives à des longueurs		
3) La variable n'est **pas** convertie automatiquement dans le système de coordonnées actuel de la CN (en pouce/métrique). La conversion relève de l'entière responsabilité de l'utilisateur / constructeur de la machine.		

Remarque

Dépassement de niveaux dû aux conversions de format

Le format de stockage interne de l'ensemble des variables utilisateur (GUD / PUD / LUD) exprimées en unités physiques de longueur est métrique. Une utilisation excessive de telles variables lors de l'exécution de la NCK, p. ex. durant des actions synchrones, risquerait de provoquer un dépassement de la durée de calcul du niveau de l'interpolateur en cas de changement de système de coordonnées, ce qui conduirait à l'émission de l'alarme 4240.

IMPORTANT

Compatibilité d'unités

La compatibilité des unités concernées n'est pas vérifiée lors de l'utilisation de variables (affectation, comparaison, calcul, etc.). Une éventuelle conversion requise relève exclusivement de la responsabilité de l'utilisateur / constructeur de la machine.

1.1.10 Attribut : droits d'accès (APR, APW, APRP, APWP, APRB, APWB)

Les niveaux de protection suivants, à indiquer lors de la programmation, correspondent aux différents droits d'accès :

Droit d'accès	Niveau de protection
Mot de passe système	0
Mot de passe constructeur	1
Mot de passe maintenance	2
Mot de passe utilisateur final	3
Commutateur à clé, position 3	4
Commutateur à clé, position 2	5
Commutateur à clé, position 1	6
Commutateur à clé, position 0	7

Définition (DEF) de variables utilisateur

Il est possible de définir des droits d'accès (APR... / APW...) pour les variables suivantes :

- Données utilisateur globales (GUD)

Redéfinition (REDEF) de variables système et de variables utilisateur

Il est possible de redéfinir des droits d'accès (APR... / APW...) pour les variables suivantes :

- Données système
  - Paramètres machine
  - Données de réglage
  - FRAME
  - Données process
  - Compensation d'erreur de pas de vis de transmission (EEC)
  - Compensation de la flexion (CEC)
  - Compensation des défauts aux transitions entre quadrants (QEC)
  - Données de magasin
  - Données d'outil
  - Zones de protection
  - Organes porte-outil orientables
  - Chaînes cinématiques
  - Zones de protection 3D
  - Limitation de la zone de travail
  - Données outil ISO

- Données utilisateur
  - Paramètre R
  - Variable à action synchrone ($AC_MARKER, $AC_PARAM, $AC_TIMER)
  - Variable GUD à action synchrone (SYG_xy[ ], avec x=R, I, B, A, C, S et y=S, M, U, 4, ..., 9)
  - Paramètre EPS
  - Données outil OEM
  - Données magasin OEM
  - Variables utilisateur globales (GUD)

Remarque

Lors de la redéfinition, vous pouvez attribuer le droit d'accès à une variable quelconque comprise entre le niveau de protection le plus faible, à savoir 7, et son propre niveau de protection, p. ex. 1 (constructeur de la machine).

Redéfinition (REDEF) d'instructions en language de programmation CN

Il est possible de redéfinir le droit d'accès ou d'exécution (APX) pour les instructions en language de programmation CN suivantes :

- Fonctions G / fonctions préparatoires

  Bibliographie :

  /PG/ Manuel de programmation Notions de base ; chapitre : Fonctions G / fonctions préparatoires

- Fonctions prédéfinies

  Bibliographie :

  /PG/ Manuel de programmation Notions de base ; chapitre : Fonctions prédéfinies

- Appels de sous-programmes prédéfinis

  Bibliographie :

  /PG/ Manuel de programmation Notions de base ; chapitre : Appels de sous-programmes prédéfinis

- Instruction DO pour les actions synchrones
- Descripteurs de cycles dans un programme

  Le cycle doit être enregistré dans un répertoire de cycles et contenir une instruction PROC.

Droits d'accès concernant les programmes pièce et les cycles (APRP, APWP)

Les différents droits d'accès dans un programme pièce ou dans un cycle sont les suivants :

- APRP 0 / APWP 0
  - L'exécution du programme pièce requiert le mot de passe système.
  - Le cycle doit être enregistré dans le répertoire _N_CST_DIR (système)
  - Pour le répertoire _N_CST_DIR, le droit d'exécution doit être défini sur Système dans le paramètre machine MD11160 $MN_ACCESS_EXEC_CST
- APRP 1 / APWP 1 ou APRP 2 / APWP 2
  - Lors de l'exécution du programme pièce, le mot de passe du constructeur de la machine ou du service de maintenance doit être défini.
  - Le cycle doit être enregistré dans le répertoire _N_CMA_DIR (constructeur de la machine) ou _N_CST_DIR.
  - Pour les répertoires _N_CMA_DIR ou _N_CST_DIR, les droits d'exécution doivent être définis au moins pour le constructeur de la machine dans les paramètres machine MD11161 $MN_ACCESS_EXEC_CMA ou MD11160 $MN_ACCESS_EXEC_CST.
- APRP 3 / APWP 3
  - Lors de l'exécution du programme pièce, le mot de passe de l'utilisateur final doit être défini
  - Le cycle doit être enregistré dans le répertoire _N_CUS_DIR (utilisateur), _N_CMA_DIR ou _N_CST_DIR
  - Pour les répertoires _N_CUS_DIR, _N_CMA_DIR ou _N_CST_DIR, les droits d'exécution doivent être définis au moins pour l'utilisateur final dans les paramètres machine MD11162 $MN_ACCESS_EXEC_CUS, MD11161 $MN_ACCESS_EXEC_CMA ou MD11160 $MN_ACCESS_EXEC_CST.
- APRP 4...7 / APWP 4...7
  - L'exécution du programme pièce requiert la position 3 ... 0 du commutateur à clé
  - Le cycle doit être enregistré dans le répertoire _N_CUS_DIR, _N_CMA_DIR ou _N_CST_DIR
  - Pour les répertoires _N_CUS_DIR, _N_CMA_DIR ou _N_CST_DIR, les droits d'exécution doivent être définis au moins pour la position correspondante du commutateur à clé dans les paramètres machine MD11162 $MN_ACCESS_EXEC_CUS, MD11161 $MN_ACCESS_EXEC_CMA ou MD11160 $MN_ACCESS_EXEC_CST.

Droits d'accès concernant BTSS (APRB, APWB)

Les droits d'accès (APRB, APWB) restreignent l'accès via BTSS aux variables système et variables utilisateur de la même manière pour tous les composants système (IHM, AP, ordinateurs externes, services EPS, etc.).

Remarque

Droits d'accès IHM locaux

Lors de la modification des droits d'accès de données système, vous devez veillez à ce qu'elle s'effectue de façon cohérente avec les droits d'accès définis par les mécanismes IHM.

Attributs d'accès APR / APW

Pour des raisons de compatibilité, les attributs APR et APW sont représentés de manière implicite sur les attributs APRP / APRB et APWP / APWB :

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

Paramétrage des droits d'accès au moyen de fichiers ACCESS

Lorsque vous utilisez des fichiers ACCES pour l'attribution de droits d'accès, la programmation d'une redéfinition de ces droits pour les données système, données utilisateur et instructions en language de programmation CN est seulement encore possible dans ces fichiers ACCESS. Les données utilisateur globales (GUD) constituent une exception. Si une redéfinition des droits d'accès s'avère nécessaire pour celles-ci, elle doit toujours être programmée dans les fichiers de définition correspondants.

Pour une protection d'accès continue, les paramètres machine doivent être adaptés de façon cohérente pour les droits d'exécution et le droit d'accès des répertoires correspondants.

Il en résulte la marche à suivre de principe suivante :

- Création des fichiers de définition requis :
  - _N_DEF_DIR/_N_SACCESS_DEF
  - _N_DEF_DIR/_N_MACCESS_DEF
  - _N_DEF_DIR/_N_UACCESS_DEF
- Paramétrage du droit d'accès pour les fichiers de définition à la valeur requise pour la redéfinition :
  - MD11170 $MN_ACCESS_WRITE_SACCESS
  - MD11171 $MN_ACCESS_WRITE_MACCESS
  - MD11172 $MN_ACCESS_WRITE_UACCESS

- L'accès depuis des cycles à des éléments protégés nécessite l'adaptation des droits d'exécution et des droits d'écriture aux répertoires _N_CST_DIR, _N_CMA_DIR et _N_CST_DIR de ces cycles :

  Droits d'exécution

  – MD11160 $MN_ACCESS_EXEC_CST
  – MD11161 $MN_ACCESS_EXEC_CMA
  – MD11162 $MN_ACCESS_EXEC_CUS

  Droits d'écriture

  – MD11165 $MN_ACCESS_WRITE_CST
  – MD11166 $MN_ACCESS_WRITE_CMA
  – MD11167 MN_ACCESS_WRITE_CUS

  Le droit d'exécution doit être défini au moins au même niveau de protection que le niveau de protection le plus élevé de l'élément utilisé.

  Le droit d'écriture doit être défini au moins au même niveau de protection que le droit d'exécution.

- Les droits d'écriture des répertoires des cycles pour l'IHM locale doivent être définis au même niveau de protection que les répertoires des cycles de la CN locale.

  Bibliographie

  /BAD/ Manuel d'utilisation HMI-Advanced,

  Chapitre : Groupe fonctionnel Services > Gestion des données > Modification des propriétés

Appels de sous-programmes dans des fichiers ACCESS

Pour une meilleure structuration de la protection d'accès, il est possible d'appeler également des sous-programmes (extension SPF ou MPF) dans les fichiers ACCESS. Les droits d'accès du fichier ACCESS appelant sont alors attribués à ces sous-programmes.

Remarque

Seuls les droits d'accès peuvent être redéfinis dans les fichiers ACCESS. La programmation ou la redéfinition de tous les autres attributs doit toujours encore s'effectuer dans les fichiers de définition correspondants.

1.1.11 Vue d'ensemble des attributs définissables et redéfinissables

Les tableaux suivants indiquent pour quels types de données il est possible de définir (DEF) et/ou redéfinir (REDEF) les différents attributs.

Données système

Type de paramètre	Valeur initiale	Valeurs limites	Unité physique	Droits d'accès
Paramètres machine	---	---	---	REDEF
Données de réglage	REDEF	---	---	REDEF
Données FRAME	---	---	---	REDEF
Données process	---	---	---	REDEF
Compensation d'erreur de pas de vis de transmission (EEC)	---	---	---	REDEF
Compensation de la flexion (CEC)	---	---	---	REDEF
Compensation des défauts aux transitions entre quadrants (QEC)	---	---	---	REDEF
Données de magasin	---	---	---	REDEF
Données d'outil	---	---	---	REDEF
Zones de protection	---	---	---	REDEF
Organes porte-outil orientables	---	---	---	REDEF
Chaînes cinématiques	---	---	---	REDEF
Zones de protection 3D	---	---	---	REDEF
Limitation de la zone de travail	---	---	---	REDEF
Données outil ISO	---	---	---	REDEF

Données utilisateur

Type de paramètre	Valeur initiale	Valeurs limites	Unité physique	Droits d'accès
Paramètre R	REDEF	REDEF	REDEF	REDEF
Variable à action synchrone ($AC_...)	REDEF	REDEF	REDEF	REDEF
GUD d'action synchrone (SYG_...)	REDEF	REDEF	REDEF	REDEF
Paramètre EPS	REDEF	REDEF	REDEF	REDEF
Données outil OEM	REDEF	REDEF	REDEF	REDEF
Données magasin OEM	REDEF	REDEF	REDEF	REDEF
Variables utilisateur globales (GUD)	DEF / REDEF	DEF	DEF	DEF / REDEF
Variables utilisateur locales (PUD / LUD)	DEF	DEF	DEF	---

1.1.12 Définition et initialisation de variables de tableau (DEF, SET, REP)

Fonction

Une variable utilisateur peut être définie comme tableau (Array) de 1 à 3 dimensions au maximum :

- 1 dimension : `DEF <type de données> <nom de variable>[<n>]`
- 2 dimensions : `DEF <type de données> <nom de variable>[<n>,<m>]`
- 3 dimensions : `DEF <type de données> <nom de variable>[<n>,<m>,<o>]`

Remarque

Les variables utilisateur du type de données STRING peuvent être définies comme tableau à 2 dimensions au maximum.

Types de données

Les variables utilisateur peuvent être définie comme tableaux pour les types de données suivants : BOOL, CHAR, INT, REAL, STRING, AXIS, FRAME

Affectation de valeur à des éléments de tableau

Les affectations de valeurs à des éléments de tableau peuvent être réalisées aux instants suivants :

- Lors de la définition du tableau (valeurs d'initialisation)
- Pendant l'exécution du programme

Les affectations de valeurs peuvent être réalisées par :

- Indication explicite d'un élément de tableau
- Indication explicite d'un élément de tableau comme élément de départ et indication d'une liste de valeurs (SET)
- Indication explicite d'un élément de tableau comme élément de départ et indication d'une valeur ainsi que de sa fréquence d'itération (REP)

Remarque

Il n'est pas possible d'affecter des valeurs d'initialisation à des variables utilisateur du type de données FRAME.

Syntaxe (DEF)

```
DEF <type de données> <nom de variable>[<n>,<m>,<o>]
DEF  STRING[<longueur de chaîne>] <nom de variable>[<n>,<m>]
```

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

Utilisation d'une liste de valeurs :

- Lors de la définition :

```
DEF <type de données> <nom de variable>[<n>,<m>,<o>] =
SET(<valeur1>,<valeur2>,...)
```

Equivaut à :

```
DEF <type de données> <nom de variable>[<n>,<m>,<o>]= (<valeur1>,<valeur2>,...)
```

Remarque

Lors de l'initialisation au moyen d'une liste de valeurs, l'indication de SET est facultative.

- Lors d'une affectation de valeur :

```
<Nom de variable>[<n>,<m>,<o>]=SET(<VALEUR1>,<valeur2>,...)
```

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

Utilisation d'une valeur avec itération

- Lors de la définition :

```
DEF <type de données> <nom de variable>[<n>,<m>,<o>]=REP(<Valeur>)
```

```
DEF <type de données> <nom de variable>[<n>,<m>,<o>] = REP(<valeur>,<nombre
d'éléments de tableau>)
```

- Lors d'une affectation de valeur :

```
<Nom de variable>[<n>,<m>,<o>]=REP(<Valeur>)
<Nom de variable>[<n>,<m>,<o>] = REP(<valeur>,<nombre d'éléments de tableau>)
```

Signification

DEF :	Instruction de définition des variables
<type de données> :	Type de données de la variable Plage de valeurs : • Pour les variables système : BOOL, CHAR, INT, REAL, STRING, AXIS • Pour les variables GUD et LUD : BOOL, CHAR, INT, REAL, STRING, AXIS, FRAME
<longueur de chaîne> :	Nombre maximal de caractères du type de données STRING
<nom de variable> :	Nom de variable
[<n>,<m>,<o>] :	Dimensions de tableaux ou indices de tableaux
<n> :	Dimension ou indice pour un tableau à 1 dimension Type : INT (également AXIS pour les variables système)

	Plage de valeurs : Dimension de tableau max. : 65535 Indice de tableau : 0 ≤ n ≤ 65534
`<m>` :	Dimension ou indice pour un tableau à 2 dimension
	Type : INT (également AXIS pour les variables système)
	Plage de valeurs : Dimension de tableau max. : 65535 Indice de tableau : 0 ≤ m ≤ 65534
`<o>` :	Dimension ou indice pour un tableau à 3 dimension
	Type : INT (également AXIS pour les variables système)
	Plage de valeurs : Dimension de tableau max. : 65535 Indice de tableau : 0 ≤ o ≤ 65534
`SET` :	Affectation de valeur au moyen de la liste de valeurs indiquée
`(<valeur1>,<valeur2>,...)` :	Liste de valeurs
`REP` :	Affectation de valeur au moyen de la `<valeur>` indiquée
`<valeur>` :	Valeur à écrire dans les éléments de tableau lors de l'initialisation avec `REP`
`<nombre d'éléments de tableau>` :	Nombre d'éléments de tableau à décrire avec la `<valeur>` indiquée. Pour les autres éléments de tableau, s'applique selon l'instant : • Initialisation lors d la définition du tableau : → Les éléments de tableau restant sont décrits avec des zéros • Affectation pendant l'exécution du programme : → Les valeurs courantes des éléments de tableau demeurent inchangées. Si le paramètre n'est pas programmé, tous les éléments de tableau sont décrits avec `<valeur>`. Si le paramètre est égal à zéro, s'applique selon l'instant : • Initialisation lors d la définition du tableau : → Tous les éléments sont mis à zéro par le système. • Affectation pendant l'exécution du programme : → Les valeurs courantes des éléments de tableau demeurent inchangées.

Indice de tableau

L'ordre implicite des éléments de tableau, p. ex. lors d'une affectation de valeur via SET ou REP est déterminé par itération de l'indice de tableau de droite à gauche.

Exemple : Initialisation d'un tableau à 3 dimensions comportant 24 éléments de tableau :

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

Correspond à :

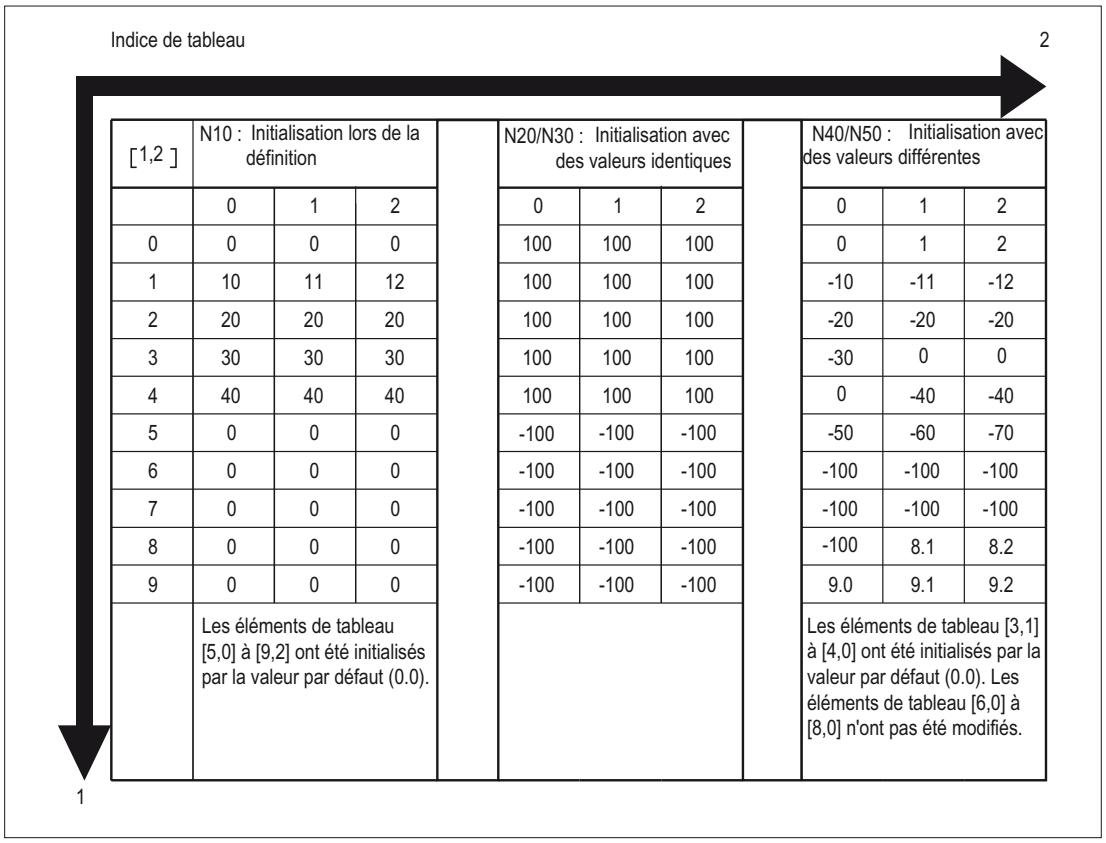
```
FOR n=0 TO 1
  FOR m=0 TO 2
     FOR o=0 TO 3
       FELD[n,m,o] = 1
     ENDFOR
  ENDFOR
ENDFOR
```

Exemple : Initialisation de tableaux de variables complets

Pour l'affectation courante, voir la figure.

Code de programme

```
N10 DEF REAL TABLEAU1[10,3]=SET(0,0,0,10,11,12,20,20,20,30,30,30,40,40,40,)
N20 TABLEAU1[0,0]=REP(100)
N30 TABLEAU1[5,0]=REP(-100)
N40 TABLEAU1[0,0]=SET(0,1,2,-10,-11,-12,-20,-20,-20,-30, , , ,-40,-40,-50,-60,-70)
N50 TABLEAU1[8,1]=SET(8.1,8.2,9.0,9.1,9.2)
```



[1,2]	N10 : Initialisation lors de la définition			N20/N30 : Initialisation avec des valeurs identiques			N40/N50 : Initialisation avec des valeurs différentes		
	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
	Les éléments de tableau [5,0] à [9,2] ont été initialisés par la valeur par défaut (0.0).						Les éléments de tableau [3,1] à [4,0] ont été initialisés par la valeur par défaut (0.0). Les éléments de tableau [6,0] à [8,0] n'ont pas été modifiés.		

1.1.13 Définition et initialisation de variables de tableau (DEF, SET, REP) : Informations complémentaires

Informations complémentaires (SET)

Initialisation lors de la définition

- Le nombre d'éléments de tableau correspondant aux éléments programmés dans la liste de valeurs sont initialisés avec les valeurs de la liste, à commencer par le 1er élément de tableau.
- La valeur 0 est affectée aux éléments de tableau sans valeur explicite indiquée dans la liste de valeurs (lacunes dans la liste de valeurs).
- Des lacunes dans la liste de valeurs ne sont pas admises pour les variables du type de données AXIS.
- Si la liste contient davantage de valeurs que d'éléments de tableau ne sont définis, une alarme est émise.

Affectation de valeur pendant l'exécution du programme

Les règles décrites ci-dessus dans le cadre de la définition s'appliquent à l'affectation de valeur pendant l'exécution du programme. Il existe en outre les possibilités suivantes :

- Il est également possible d'utiliser des expressions comme éléments dans la liste de valeurs.
- L'affectation de valeur débute par l'indice de tableau programmé. De cette façon, on peut affecter des valeurs à des tableaux partiels.

Exemple :

Code de programme	Commentaire
`DEF INT TABLEAU[5,5]`	`; Définition de tableaux`
`TABLEAU[0,0]=SET(1,2,3,4,5)`	`; Affectation de valeur aux 5 premiers éléments de tableau [0,0] - [0,4]`
`FELD[0,0]=SET(1,2, , ,5)`	`; Affectation de valeur avec lacune aux 5 premiers éléments de tableau [0,0] - [0,4], éléments de tableau [0,2] et [0,3] = 0`
`TABLEAU[2,3]=SET(VARIABLE,4*5.6)`	`; Affectation de valeur avec une variable et une expression comme indice de tableau [2,3] : [2,3] = VARIABLE [2,4] = 4 * 5.6 = 22.4`

Informations complémentaires (REP)

Initialisation lors de la définition

- L'ensemble des éléments de tableau ou le nombre d'entre eux spécifié de manière facultative sont initialisés avec la valeur indiquée (constante).
- Les variables du type de données FRAME ne peuvent pas être initialisées.

Exemple :

Code de programme	Commentaire
`DEF REAL varName[10]=REP(3.5,4)`	`; Initialiser la définition de tableau et les éléments de tableau [0] à [3] avec la valeur 3,5`

Affectation de valeur pendant l'exécution du programme

Les règles décrites ci-dessus dans le cadre de la définition s'appliquent à l'affectation de valeur pendant l'exécution du programme. Il existe en outre les possibilités suivantes :

- Il est également possible d'utiliser des expressions comme éléments dans la liste de valeurs.
- L'affectation de valeur débute par l'indice de tableau programmé. De cette façon, on peut affecter des valeurs à des tableaux partiels.

Exemples :

Code de programme	Commentaire
`DEF REAL varName[10]`	`; Définition de tableaux`
`varName[5]=REP(4.5,3)`	`; Eléments de tableau [5] à [7] = 4,5`
`R10=REP(2.4,3)`	`; Paramètre R, R10 à R12 = 2,4`
`DEF FRAME FRM[10]`	`; Définition de tableaux`
`FRM[5] = REP(CTRANS (X,5))`	`; Eléments de tableau [5] à [9] = CTRANS(X,5)`

Informations complémentaires (généralités)

Affectations de valeurs à des paramètres machine axiaux

Les paramètres machine axiaux possèdent par principe un indice de tableau du type de données AXIS. Lors d'affectations de valeurs à un paramètre machine axial avec SET ou REP, cet indice de tableau est ignoré ou pas exécuté.

Exemple : Affectation de valeur au paramètre machine MD36200 $MA_AX_VELO_LIMIT

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

Correspond à :

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

IMPORTANT
Affectations de valeurs à des paramètres machine axiaux
Lors d'affectations de valeurs à un paramètre machine axial avecSET ou REP, l'indice de tableau du type de données AXIS est ignoré ou pas exécuté.

Capacité mémoire requise

type de données	Capacité de mémoire requise par élément
BOOL	1 octet
CHAR	1 octet
INT	4 octets
REAL	8 octets
STRING	(longueur de chaîne de caractères + 1) octets
FRAME	~ 400 octets, en fonction du nombre d'axes
AXIS	4 octets

1.1.14 Types de données

Les types de données suivants sont disponibles dans la CN :

type de données	Signification	Plage de valeurs
INT	Valeur entière avec signe	-2147483648 ... +2147483647
REAL	Nombre réel (LONG REAL selon IEEE)	±(~2,2*10^{-308} … ~1,8*10^{+308})
BOOL	Valeur booléenne TRUE (1) et FALSE (0)	1, 0
CHAR	Caractère ASCII	Code ASCII 0 ... 255
STRING	Chaîne de caractères de longueur définie	au maximum 200 caractères (caractères spéciaux exclus)
AXIS	Descripteur d'axe/de broche	Descripteur d'axe de canal
FRAME	Indications géométriques pour une transformation statique de coordonnées (décalage, rotation, mise à l'échelle, fonction miroir)	---

Conversion implicite du type de données

Les conversions suivantes de types de données sont possibles et réalisées de façon implicite pour les affectations et transmissions de paramètres :

de ↓/ vers →	REAL	INT	BOOL
REAL	x	o	&
INT	x	x	&
BOOL	x	x	x

x : Réalisable sans restrictions

o : Risque de perte de données par écrasement de la plage de valeurs ⇒ alarme ;
arrondissement : valeur après la virgule ≥ 0,5 ⇒ arrondissement à la valeur supérieure, valeur après la virgule < 0,5 ⇒ arrondissement à la valeur inférieure

&: valeur ≠ 0 ⇒ TRUE, valeur == 0 ⇒ FALSE

1.2 Programmation indirecte

1.2.1 Programmation indirecte d'adresses

Fonction

Dans le cas de la programmation indirecte d'adresses, l'indice d'adresse est remplacé par une variable de type approprié.

Remarque

La programmation indirecte d'adresses est impossible pour :

- N (numéro de bloc)
- L (sous-programme)
- Adresses réglables
  (par exemple, X[1] au lieu de X1 n'est pas autorisé)

Syntaxe

```
<ADRESSE>[<indice>]
```

Signification

`<ADRESSE>[...]` :	Adresse fixe avec extension (indice)
`<indice>` :	Variable telle que n° de broche, axe, etc.

Exemples

Exemple 1 : Programmation indirecte d'un numéro de broche

Programmation directe :

Code de programme	Commentaire
`S1=300`	`; Vitesse de rotation de 300 tr/min pour la broche avec le numéro 1`

Programmation indirecte :

Code de programme	Commentaire
`DEF INT NUMBRO=1`	`; Définition des variables de type INT et affectation de valeur`
`S[SPINU]=300`	`; Vitesse de rotation de 300 tr/min pour la broche dont le numéro est consigné dans la variable SPINU (dans cet exemple, la broche avec le numéro 1)`

Exemple 2 : Programmation indirecte d'un axe

Programmation directe :

Code de programme	Commentaire
FA[U]=300	; Avance de 300 pour l'axe "U"

Programmation indirecte :

Code de programme	Commentaire
DEF AXIS AXVAR2=U	; Définition d'une variable de type AXIS et affectation de valeur
FA[AXVAR2]=300	; Avance de 300 pour l'axe dont le nom d'adresse est consigné dans la variable AXVAR2

Exemple 3 : Programmation indirecte d'un axe

Programmation directe :

Programmation	Commentaire
$AA_MM[X]	; Lecture de la valeur de mesure du palpeur (SCM) de l'axe "X"

Programmation indirecte :

Code de programme	Commentaire
DEF AXIS AXVAR3=X	; Définition d'une variable de type AXIS et affectation de valeur
$AA_MM[AXVAR3]	; Lecture de la valeur de mesure du palpeur (SCM) de l'axe dont le nom est consigné dans la variable AXVAR3

Exemple 4 : Programmation indirecte d'un axe

Programmation directe :

Code de programme
X1=100 X2=200

Programmation indirecte :

Code de programme	Commentaire
DEF AXIS AXVAR1 AXVAR2	; Définition de deux variables de type AXIS
AXVAR1=(X1) AXVAR2=(X2)	; Affectation des noms d'axe
AX[AXVAR1]=100 AX[AXVAR2]=200	; Déplacement des axes dont les noms d'adresse sont consignés dans les variables AXVAR1 et AXVAR2

Exemple 5 : Programmation indirecte d'un axe

Programmation directe :

```
Code de programme
G2 X100 I20
```

Programmation indirecte :

```
Code de programme                    Commentaire
DEF AXIS AXVAR1=X                    ; Définition d'une variable de type AXIS et
                                       affectation de valeur
G2 X100 IP[AXVAR1]=20                ; Programmation indirecte du centre de l'axe dont
                                       le nom d'adresse est consigné dans la variable
                                       AXVAR1
```

Exemple 6 : Programmation indirecte d'éléments de tableau

Programmation directe :

```
Code de programme                    Commentaire
DEF INT TABLEAU1[4,5]                ; Définition du tableau 1
```

Programmation indirecte :

```
Code de programme                    Commentaire
DEFINE DIM1 AS 4                     ; Les dimensions de tableau sont à indiquer comme
                                       valeurs fixes.
DEFINE DIM2 AS 5
DEF INT TABLEAU[DIM1,DIM2]
TABLEAU[DIM1-1,DIM2-1]=5
```

Exemple 7 : Appel indirect de sous-programme

```
Code de programme                    Commentaire
CALL "L" << R10                      ; Appel du programme dont le numéro figure dans R10
                                       (concaténation de chaînes de caractères)
```

1.2.2 Programmation indirecte de codes G

Fonction

La programmation indirecte des codes G permet une programmation efficace des cycles.

Syntaxe

```
G[<groupe>]=<numéro>
```

Signification

`G[...]` :	Instruction G avec extension (indice)
`<groupe>` :	Paramètre d'indice : groupe de fonctions G
	Type : INT
`<numéro>` :	Variable du numéro de code G
	Type : INT ou REAL

Remarque

En règle générale, seuls des codes G non déterminants pour la syntaxe peuvent être programmés indirectement.

Parmi les codes G déterminants pour la syntaxe, seuls ceux du groupe de fonctions G 1 sont possibles.
Les codes G déterminants pour la syntaxe des groupes de fonction 2, 3 et 4 ne sont pas possibles.

Remarque

La programmation indirecte des codes G n'admet pas les fonctions arithmétiques. Le calcul du numéro de code G doit s'effectuer dans une ligne distincte du programme pièce, avant sa programmation indirecte.

Exemples

Exemple 1 : Décalage d'origine réglable (groupe de fonctions G "8")

```
Code de programme                          Commentaire
N1010 DEF INT INT_VAR
N1020 INT_VAR=2
...
N1090 G[8]=INT_VAR G1 X0 Y0                ; G54
N1100 INT_VAR=INT_VAR+1                    ; Calcul du code G
N1110 G[8]=INT_VAR G1 X0 Y0                ; G55
```

Exemple 2 : Sélection du plan (groupe de fonctions G "6")

Code de programme	Commentaire
N2010 R10=$P_GG[6]	; Lecture de la fonction G active du groupe de fonctions G "6"
...	
N2090 G[6]=R10	

Bibliographie

Pour des informations sur les groupes de fonctions G, voir :
Manuel de programmation Notions de base, chapitre "Groupes de fonctions G"

1.2.3 Programmation indirecte d'attributs de position (PB)

Fonction

Les attributs de position tels que la programmation relative ou absolue de la position d'axe peuvent être programmés indirectement en tant que variables, en liaison avec le mot clé GP.

Application

La programmation indirecte d'attributs de position s'utilise dans les **cycles de substitution**, car elle présente alors les avantages suivants par rapport à la programmation des attributs de position comme mot clé (IC, AC, etc.)

Avec la programmation indirecte comme variables, **aucune** instruction CASE n'est nécessaire pour accéder à tous les attributs de position possibles.

Syntaxe

```
<INSTRUCTION DE POSITIONNEMENT>[<axe/broche>]=
GP(<position>,<attribut de position>)
<Axe/broche>=GP(<position>,<attribut de position>)
```

Signification

<INSTRUCTION DE POSITIONNEMENT>[] :	Les instructions de positionnement suivantes peuvent être programmées avec le mot clé GP : POS, POSA, SPOS, SPOSA Autres possibilités : • Tous les descripteurs d'axe/broche présents dans le canal : <Axe/broche> • Descripteur d'axe/broche variable AX
<axe/broche> :	Axe/broche à positionner
GP() :	Mot clé pour le positionnement
<position> :	Paramètre 1 Position d'axe/broche comme constante ou variable
<attribut de position> :	Paramètre 2 Attribut de position (par exemple le mode d'accostage de la position) comme variable (par exemple $P_SUB_SPOSMODE) ou comme mot clé (IC, AC, etc.)

Les valeurs fournies par les variables ont la signification suivante :

Valeur	Signification	Admis pour :
0	Aucune modification de l'attribut de position	
1	AC	POS, POSA, SPOS, SPOSA, AX, adresse d'axe
2	IC	POS, POSA, SPOS, SPOSA, AX, adresse d'axe
3	DC	POS, POSA, SPOS, SPOSA, AX, adresse d'axe
4	ACP	POS, POSA, SPOS, SPOSA, AX, adresse d'axe
5	ACN	POS, POSA, SPOS, SPOSA, AX, adresse d'axe
6	OC	-
7	PC	-
8	DAC	POS, POSA, AX, adresse d'axe
9	DIC	POS, POSA, AX, adresse d'axe
10	RAC	POS, POSA, AX, adresse d'axe
11	RIC	POS, POSA, AX, adresse d'axe
12	CAC	POS, POSA
13	CIC	POS, POSA
14	CDC	POS, POSA
15	CACP	POS, POSA
16	CACN	POS, POSA

Exemple

En cas de couplage actif de broches synchrones entre la broche pilote S1 et la broche asservie S2, l'instruction `SPOS` appelle le cycle de substitution suivant pour le positionnement des broches dans le programme principal.

Le positionnement est effectué par l'instruction de `N2230` :
`SPOS[1]=GP($P_SUB_SPOSIT,$P_SUB_SPOSMODE)`
`SPOS[2]=GP($P_SUB_SPOSIT,$P_SUB_SPOSMODE)`

La position à accoster est lue dans la variable système $P_SUB_SPOSIT et le mode d'accostage de la position dans la variable système $P_SUB_SPOSMODE.

```
Code de programme                                   Commentaire
N1000 PROC LANG_SUB DISPLOF SBLOF
...
N2100 IF($P_SUB_AXFCT==2)
N2110                                               ; Substitution de l'instruction SPOS / SPOSA / M19
                                                      en cas de couplage actif de broches synchrones
N2185 DELAYFSTON                                    ; Début de la plage d'arrêt temporisé
N2190 COUPOF(S2,S1)                                 ; Désactivation du couplage de broches synchrones
N2200                                               ; Positionnement de la broche pilote et de la
                                                      broche asservie
N2210 IF($P_SUB_SPOS==TRUE) OR
($P_SUB_SPOSA==TRUE)
N2220                                               ; Positionnement de broche avec SPOS :
N2230   SPOS[1]=GP($P_SUB_SPOSIT,$P_SUB_SPOSMODE)
        SPOS[2]=GP($P_SUB_SPOSIT,$P_SUB_SPOSMODE)
N2250 ELSE
N2260                                               ; Positionnement de broche avec M19 :
N2270 M1=19 M2=19                                   ; Positionnement de la broche pilote et de la
                                                      broche asservie
N2280 ENDIF
N2285 DELAYFSTOF                                    ; Fin de la plage d'arrêt temporisé
N2290 COUPON(S2,S1)                                 ; Activation du couplage de broches synchrones
N2410 ELSE
N2420                                               ; Interrogation de l'existence d'autres
                                                      substitutions
...
N3300 ENDIF
...
N9999 RET
```

Conditions marginales

- Dans les synchronisations, la programmation indirecte des attributs de position est impossible.

Bibliographie

Description fonctionnelle Fonctions de base ; GMF, Canal, Mode de programme, Comportement au reset (K1),
chapitre : Substitution de fonctions CN par des sous-programmes

1.2.4 Programmation indirecte de lignes de programme pièce (EXECSTRING)

Fonction

Une variable chaîne de caractères créée au préalable peut être exécutée en tant que ligne de programme pièce au moyen de l'instruction EXECSTRING.

Syntaxe

EXECSTRING est programmée dans une ligne spécifique du programme pièce :
EXECSTRING (<variable de type STRING>)

Signification

EXECSTRING :	Instruction pour l'exécution d'une variable chaîne de caractères en tant que ligne de programme pièce
<variable de type STRING> :	Variable du type STRING dans laquelle figure la ligne de programme pièce à exécuter proprement dite

Remarque

Tous les éléments susceptibles d'être programmés dans une section d'un programme pièce peuvent être transmis avec EXECSTRING. Les instructions PROC et DEF et, d'une manière générale, l'utilisation dans des fichiers INI et DEF en sont exclues.

Exemple

```
Code de programme                        Commentaire
N100 DEF STRING[100] BLOCK               ; Définition des variables chaînes de caractères
                                           pour la prise en compte des lignes de programme
                                           pièce à exécuter.
N110 DEF STRING[10] MFCT1="M7"
...
N200 EXECSTRING(MFCT1 << "M4711")        ; Exécution de la ligne "M7 M4711" du programme
                                           pièce.
...
N300 R10=1
N310 BLOC="M3"
N320 IF(R10)
N330 BLOCK = BLOCK << MFCT1
N340 ENDIF
N350 EXECSTRING(BLOCK)                   ; Exécution de la ligne "M3 M7" du programme
                                           pièce.
```

1.3 Fonctions de calcul

Fonction

Les fonctions de calcul sont applicables en priorité aux paramètres R et aux variables (ou constantes et fonctions) de type REAL. Les types INT et CHAR sont également admis.

Opérateur arithmétique / fonction de calcul	Signification
+	Addition
-	Soustraction
*	Multiplication
/	Division **Attention** : (type INT)/(type INT)=(type REAL) ; Exemple : 3/4 = 0.75
DIV	Division, pour types de variable INT et REAL **Attention** : (type INT)DIV(type INT)=(type INT) ; Exemple : 3 DIV 4 = 0
MOD	La division modulo (uniquement pour le type INT) fournit le reste d'une division INT. Exemple : 3 MOD 4 = 3
:	Opérateur de concaténation (pour variables de type FRAME)
Sin()	Sinus
COS()	Cosinus
TAN()	Tangente
ASIN()	Arc sinus
ACOS()	Arc cosinus
ATAN2(,)	Arc tangente 2
SQRT( )	Racine carrée
ABS( )	Valeur absolue
POT( )	2. Puissance (carré)
TRUNC( )	Partie entière Précision réglable avec TRUNC( ) dans les instructions relationnelles (voir "Correction de la précision pour les erreurs relationnelles (TRUNC) (Page 66)")
ROUND()	arrondir à un nombre entier
LN( )	Logarithme naturel
EXP( )	Fonction exponentielle
MINVAL()	valeur la plus petite de deux variables (voir "Minimum, maximum et plage de variables (MINVAL, MAXVAL, BOUND) (Page 68)")

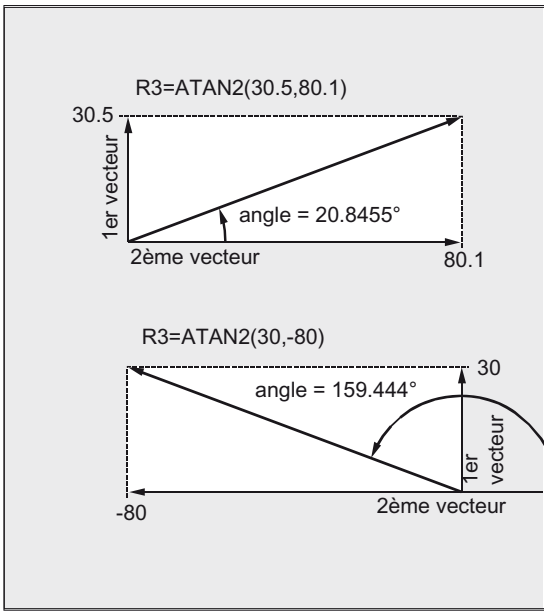
`MAXVAL()`	valeur la plus grande de deux variables (voir "Minimum, maximum et plage de variables (MINVAL, MAXVAL, BOUND) (Page 68)")
`BOUND()`	valeur de variable se trouvant dans la plage de valeurs définie (voir "Minimum, maximum et plage de variables (MINVAL, MAXVAL, BOUND) (Page 68)")
`CTRANS()`	Décalage
`CROT ()`	Rotation
`CSCALE ()`	Changement d'échelle
`CMIRROR ()`	Fonction miroir

Programmation

La notation mathématique usuelle est valable pour les fonctions de calcul. Les priorités de traitement sont fixées par des parenthèses. Les fonctions trigonométriques et leurs inverses emploient la notation en degrés (angle droit = 90°).

Exemples

Exemple 1 : ATAN2



A partir de deux vecteurs orthogonaux, la fonction de calcul ATAN2 détermine l'angle du vecteur somme.

Le résultat s'inscrit dans la plage des quatre quadrants (-180° < 0 < +180°).

La base de la référence angulaire est toujours représentée par la 2ème valeur dans le sens positif.

Exemple 2 : Initialisation de tableaux de variables complets

Code de programme	**Commentaire**
R1=R1+1	; Nouveau R1 = ancien R1 +1
R1=R2+R3 R4=R5-R6 R7=R8*R9	
R10=R11/R12 R13=SIN(25.3)	
R14=R1*R2+R3	; La multiplication est prioritaire.
R14=(R1+R2)*R3	; Les parenthèses sont traitées en priorité.
R15=SQRT(POT(R1)+POT(R2))	; Les parenthèses internes sont levées d'abord. R15 = racine carrée de (R1+R2)
RESFRAME=FRAME1:FRAME2 FRAME3=CTRANS(…):CROT(…)	; L'opérateur de concaténation combine des frames en un frame résultant ou assigne des valeurs aux composantes de frame.

1.4 Opérations relationnelles et opérations logiques

Fonction

Les opérations relationnelles peuvent servir par exemple à formuler une condition de saut dans le programme. Elles s'appliquent également aux expressions complexes.

Les opérations relationnelles sont applicables aux variables de type CHAR, INT, REAL et BOOL. Avec le type CHAR, la valeur de code est comparée.
Pour les types STRING, AXIS et FRAME, sont possibles : == et <> qui peuvent aussi être employés pour les opérations de type STRING, y compris dans des actions synchrones.

Le résultat d'opérations relationnelles est toujours du type BOOL.

Les opérateurs logiques servent à la combinaison de valeurs booléennes.

Les opérations logiques s'appliquent exclusivement aux variables du type BOOL. Avec la conversion de type interne, ils sont également applicables aux types de données CHAR, INT et REAL.

Avec les opérations logiques (booléennes) sont valables, pour les types de données BOOL, CHAR, INT et REAL :

- 0 correspond à : FALSE
- valeur différente de 0 correspond à : TRUE

Opérateurs logiques sur bits

Les variables des types CHAR et INT permettent également d'effectuer des opérations logiques sur bits. Le cas échéant, il est procédé à une conversion automatique de type.

Programmation

Opérateur de comparaison	Signification
==	égal à
<>	différent de
>	supérieur à
<	inférieur à
>=	supérieur ou égal à
<=	inférieur ou égal à

Opérateur logique	Signification
AND	ET
OR	OU
NOT	Négation
XOR	OU exclusif

Opérateur logique sur bits	Signification
B_AND	ET sur bits
B_OR	OU sur bits
B_NOT	Négation sur bits
B_XOR	OU exclusif sur bits

Remarque

Des parenthèses placées dans les expressions arithmétiques permettent de fixer un ordre des opérations différent des règles normales de priorité.

Remarque

Entre les opérandes booléens et les opérateurs doivent figurer des espacements.

Remarque

L'opérateur B_NOT ne se rapporte qu'à un seul opérande qui suit l'opérateur.

Exemples

Exemple 1 : opérateurs relationnels

```
IF R10>=100 GOTOF DESTINATION
```

ou

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

Le résultat de la comparaison R10>=100 est d'abord mémorisé dans R11.

Exemple 2 : Opérateurs logiques

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

ou

```
IF NOT R10 GOTOB DEPART
```

L'opérateur NOT ne se rapporte qu'à un seul opérande.

Exemple 3 : Opérateurs logiques sur bits

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

1.5 Correction de la précision pour les erreurs relationnelles (TRUNC)

Fonction

L'instruction TRUNC ampute l'opérande de la multiplication par un facteur de précision.

Précision réglable dans les instructions relationnelles

En interne, les données de type REAL du programme pièce sont représentées par 64 bits en format IEEE. De ce fait, les nombres décimaux ne peuvent pas être représentés de manière précise et leur comparaison avec des valeurs calculées de façon idéale peut conduire à des résultats inattendus.

Egalité relative

Afin que les imprécisions causées par le format de représentation ne faussent pas le flux du programme, les instructions relationnelles ne vérifient pas l'égalité absolue, mais l'égalité relative.

Syntaxe

Correction de la précision pour les erreurs relationnelles

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

Signification

`TRUNC` : troncature des décimales

Prise en compte d'une égalité relative de 10^{-12} pour

- Egalité : (==)
- Différence : (<>)
- Supérieur ou égal : (>=)
- Inférieur ou égal : (<=)
- Supérieur/inférieur : (><) avec égalité absolue
- Supérieur : (>)
- Inférieur : (<)

Compatibilité

Pour des raisons de compatibilité, le contrôle d'égalité relative de (>) et (<) peut être désactivé par définition du paramètre machine MD10280 $MN_ PROG_FUNCTION_MASK Bit0 = 1.

Remarque

Pour les raisons qui ont été citées, les comparaisons avec des données du type REAL connaissent d'une manière générale une certaine imprécision. Lorsque les écarts ne sont pas acceptables, vous devez effectuer des calculs de type INTEGER en multipliant les opérandes par un facteur de précision, puis en les amputant avec TRUNC.

Actions synchrones

Ce qui a été décrit pour les instructions relationnelles est aussi valable pour les actions synchrones.

Exemples

Exemple 1 : considérations relatives à la précision

```
Code de programme                                   Commentaire
N40 R1=61.01 R2=61.02 R3=0.01                       ;  Affectation des valeurs initiales
N41 IF ABS(R2-R1) > R3 GOTOF ERREUR                 ;  Le saut serait exécuté jusqu'ici.
N42 M30                                             ;  Fin du programme
N43 ERREUR   : SETAL(66000)                         ;
R1=61.01 R2=61.02 R3=0.01                           ;  Affectation des valeurs initiales
R11=TRUNC(R1*1000) R12=TRUNC(R2*1000)               ;  Correction de la précision
R13=TRUNC(R3*1000)
IF ABS(R12-R11) > R13 GOTOF ERREUR                  ;  Le saut n'est plus exécuté.
M30                                                 ;  Fin du programme
ERREUR   : SETAL(66000)                             ;
```

Exemple 2 : formation et analyse du quotient de deux opérandes

```
Code de programme                                   Commentaire
R1=61.01 R2=61.02 R3=0.01                           ;  Affectation des valeurs initiales
IF ABS((R2-R1)/R3)-1) > 10EX-5 GOTOF ERREUR         ;  Le saut n'est pas exécuté.
M30                                                 ;  Fin du programme
ERREUR   : SETAL(66000)                             ;
```

1.6 Minimum, maximum et plage de variables (MINVAL, MAXVAL, BOUND)

Fonction

Les instructions MINVAL et MAXVAL permettent de comparer entre elles les valeurs de deux variables. Le résultat fourni en retour la valeur la plus petite (pour MINVAL) ou la valeur la plus grande (pour MAXVAL).

L'instruction BOUND permet de vérifier si la valeur d'une variable à tester se trouve dans une plage de valeurs définie.

Syntaxe

```
<valeur la plus petite>=MINVAL(<variable1>,<variable2>)
<valeur la plus grande>=MAXVAL(<variable1>,<variable2>)
<valeur en retour>=<BOUND>(<minimum>,<maximum>,<variable à tester>)
```

Signification

MINVAL :	Détermine la **plus petite** parmi deux variables (<variable1>, <variable2>)
<valeur la plus petite> :	Variable résultante pour l'instruction MINVAL Est réglée à la valeur la plus petite de la variable.
MAXVAL :	Détermine la **plus grande** parmi deux variables (<variable1>, <variable2>)
<valeur la plus grande> :	Variable résultante pour l'instruction MAXVAL Est réglée à la valeur la plus grande de la variable.
BOUND :	Vérifie si une variable (<variable à tester>) se trouve dans la plage de valeurs définie.
<minimum> :	Variable définissant la valeur minimale de la plage de valeurs
<maximum> :	Variable définissant la valeur maximale de la plage de valeurs
<valeur en retour> :	Variable résultante pour l'instruction BOUND Si la valeur de la variable à tester se trouve dans la plage de valeurs définie, la variable résultante est réglée à la valeur de la variable à tester. Si la valeur de la variable à tester est supérieure à la valeur maximale, la variable résultante est réglée à la valeur maximale de la plage de définition. Si la valeur de la variable à tester est inférieure à la valeur minimale, la variable résultante est réglée à la valeur minimale de la plage de définition.

Remarque

MINVAL, MAXVAL et BOUND peuvent également être programmées dans actions synchrones.

Remarque

Comportement en cas d'égalité

En cas d'égalité de MINVAL/MAXVAL, cette valeur identique est fournie. Avec BOUND, la valeur de la variable à tester est de nouveau fournie.

Exemple

```
Code de programme                                   Commentaire
DEF REAL rVar1=10.5, rVar2=33.7, rVar3, rVar4, rVar5, rValMin, rValMax, rRetVar
rValMin=MINVAL(rVar1,rVar2)                         ; rValMin est réglée à la valeur 10.5.
rValMax=MAXVAL(rVar1,rVar2)                         ; rValMax est réglée à la valeur 33.7.
rVar3=19.7
rRetVar=BOUND(rVar1,rVar2,rVar3)                    ; rVar3 se trouve dans les limites, rRetVar est réglée
                                                      à 19.7.
rVar3=1.8
rRetVar=BOUND(rVar1,rVar2,rVar3)                    ; rVar3 se situe en-deçà de la limite minimale, rRetVar
                                                      est réglée à 10.5.

rVar3=45.2
rRetVar=BOUND(rVar1,rVar2,rVar3)                    ; rVar3 se situe au-dessus de la limite maximale,
                                                      rRetVar est réglée à 33.7.
```

1.7 Priorité des opérations

Fonction

A chaque opérateur est affectée une priorité. Lors du traitement d'une expression, les opérateurs au degré de priorité le plus élevé sont appliqués en premier lieu. En cas d'opérateurs d'égale priorité, le traitement s'opère de gauche à droite.

Des parenthèses placées dans les expressions arithmétiques permettent de fixer un ordre des opérations différent des règles normales de priorité.

Ordre des opérateurs

De la priorité la plus élevée à la moins élevée

1.	NOT, B_NOT	Négation, négation sur bits
2.	*, /, DIV, MOD	Multiplication, division
3.	+, –	Addition, soustraction
4.	B_AND	ET sur bits
5.	B_XOR	OU exclusif sur bits
6.	B_OR	OU sur bits
7.	AND	ET
8.	XOR	OU exclusif
9.	OR	OU
10.	<<	Concaténation de chaînes de caractères, type de résultat STRING
11.	==, <>, >, <, >=, <=	Opérateurs relationnels

Remarque

L'opérateur de concaténation ":" pour frames ne doit pas figurer avec d'autres opérateurs dans une même expression. Cet opérateur ne nécessite donc pas d'ordre de priorité.

Exemple d'instruction If

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

1.8 Conversions de types possibles

Fonction

Conversion de type lors d'une affectation

La valeur numérique constante, la variable ou l'expression assignée à une variable doit être compatible avec le type de cette variable. Si cette condition n'est pas remplie, le type est automatiquement converti au moment de l'affectation.

Conversions de type possibles

en	REAL	INT	BOOL	CHAR	STRING	AXIS	FRAME
de							
REAL	oui	oui*	oui[1])	oui*	–	–	–
INT	oui	oui	oui[1])	oui[2])	–	–	–
BOOL	oui	oui	oui	oui	oui	–	–
CHAR	oui	oui	oui[1])	oui	oui	–	–
STRING	–	–	oui[4])	oui[3])	oui	–	–
AXIS	–	–	–	–	–	oui	–
FRAME	–	–	–	–	–	–	oui

Observations

* En cas de conversion du type REAL en type INT, une partie décimale >= 0,5 est arrondie par excès ; dans le cas contraire, elle est arrondie par défaut (voir fonction ROUND)

1) La valeur <> 0 équivaut à TRUE, la valeur == 0 équivaut à FALSE

2) Si la valeur se situe dans la plage numérique admise

3) Dans le cas d'un seul caractère

4) Longueur de chaîne de caractères 0 = >FALSE, sinon TRUE

Remarque

Un message d'erreur est émis lorsqu'une valeur de conversion est supérieure à la plage cible.

Lorsque des types différents figurent dans une même expression, leur adaptation s'effectue automatiquement. Des conversions de type sont aussi possibles dans des actions synchrones, voir le chapitre Actions synchrones au déplacement, conversion de type implicite".

1.9 Opérations sur les chaînes de caractères

Opérations sur les chaînes de caractères

Outre les opérations classiques "Affectation" et "Comparaison", les opérations suivantes sont possibles sur les chaînes de caractères :

- Conversion de type en type STRING (AXSTRING)
- Conversion de type à partir du type STRING (NUMBER, ISNUMBER, AXNAME)
- Concaténation de chaînes de caractères (<<)
- Conversion en minuscules/majuscules (TOLOWER, TOUPPER)
- Déterminer la longueur d'une chaîne de caractères (STRLEN)
- Recherche de caractères/chaînes de caractères dans une chaîne (INDEX, RINDEX, MINDEX, MATCH)
- Sélection d'une chaîne de caractères partielle (SUBSTR)
- Sélection d'un caractère individuel (STRINGVAR, STRINGFELD)

Signification particulière du caractère 0

En interne, le caractère 0 est interprété comme identification de fin d'une chaîne de caractères. Le remplacement d'un caractère par 0 raccourcit donc la chaîne de caractères.

Exemple :

```
Code de programme                          Commentaire
DEF STRING[20] STRG="axe . à l'arrêt"
STRG[6]="X"
MSG(STRG)                                  ; Fournit le message "Axe X à
                                             l'arrêt".
STRG[6]=0
MSG(STRG)                                  ; Fournit le message "Axe".
```

1.9.1 Conversion de type en type STRING (AXSTRING)

Fonction

La fonction "Conversion de type en type STRING" permet d'utiliser des variables de types différents comme éléments d'un message (MSG).

Lors de l'utilisation de l'opérateur **<<**, la fonction est exécutée implicitement pour les données de type INT, REAL, CHAR et BOOL (voir "Concaténation de chaînes de caractères (**<<**) (Page 75)").

Une valeur INT est convertie sous la forme normale lisible. Les valeurs REAL sont données avec un nombre de décimales pouvant aller jusqu'à 10.

L'instruction `AXSTRING` convertit des variables de type AXIS en type STRING.

Syntaxe

```
<STRING_ERG> = << <type_quelconque>
<STRING_ERG> = AXSTRING(<descripteur d'axe>)
```

Signification

`<STRING_ERG>` :	Variable du résultat de la conversion de type Type : STRING
`<Type_quelconque>` :	Types de variables INT, REAL, CHAR, STRING et BOOL
`AXSTRING` :	L'instruction `AXSTRING` fournit le descripteur d'axe indiqué comme chaîne de caractères.
`<descripteur d'axe>` :	Variable du descripteur d'axe Type : AXIS

Remarque

Les variables FRAME ne peuvent pas être converties.

Exemples

Exemple 1 :

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

Exemple 2 : AXSTRING

Code de programme	Commentaire
`DEF STRING[32] RES_STRING`	
`RES_STRING=AXSTRING(X)`	`; RES_STRING == "X"`

1.9.2 Conversion de type à partir de STRING (NUMBER, ISNUMBER, AXNAME)

Fonction

Avec l'instruction NUMBER, la conversion se fait du type STRING en type REAL. L'instruction ISNUMBER s'utilise pour vérifier si la conversion est possible.

L'instruction AXNAME convertit une chaîne de caractères de type STRING en type AXIS.

Syntaxe

```
<RES_REAL>=NUMBER("<chaîne de caractères>")
<RES_BOOL>=ISNUMBER("<chaîne de caractères>")
<RES_AXIS>=AXNAME("<chaîne de caractères>")
```

Signification

NUMBER : L'instruction NUMBER donne en retour le nombre représenté par la <chaîne de caractères> comme valeur REAL.

<chaîne> : Variable de type STRING à convertir

<REAL_ERG> : Variable du résultat de la conversion de type avec NUMBER

Type : REAL

ISNUMBER : L'instruction ISNUMBER permet de contrôler si la <chaîne de caractères> est convertible en un nombre valable.

<BOOL_ERG> : Variable du résultat de l'interrogation avec ISNUMBER

Type : BOOL

Valeur : TRUE — ISNUMBER fournit la valeur TRUE quand la <chaîne de caractères> représente un nombre REAL valide selon les règles du langage.

FALSE — Si ISNUMBER indique la valeur FALSE, une alarme est émise lors de l'appel de NUMBER avec la même <chaîne de caractères>.

AXNAME : L'instruction AXNAME transforme la <chaîne de caractères> indiquée en un descripteur d'axe.

Remarque :
Une alarme est émise si la <chaîne de caractères> ne peut pas être affectée à un descripteur d'axe existant.

<AXIS_ERG> : Variable du résultat de la conversion de type avec AXNAME

Type : AXIS

Exemple

Code de programme	Commentaire
DEF BOOL BOOL_ERG	
DEF REAL REAL_ERG	
DEF AXIS AXIS_ERG	
RES_BOOL=ISNUMBER("1234.9876Ex-7")	; BOOL_ERG == TRUE
RES_BOOL=ISNUMBER("1234XYZ")	; BOOL_ERG == FALSE
RES_REAL=NUMBER("1234.9876Ex-7")	; REAL_ERG == 1234.9876Ex-7
RES_AXIS=AXNAME("X")	; AXIS_ERG == X

1.9.3 Concaténation de chaînes de caractères (<<)

Fonction

La fonction "Concaténation de chaînes de caractères" permet de composer une chaîne de caractères à partir d'éléments individuels.

La concaténation s'effectue avec l'opérateur "**<<**". Cet opérateur possède, pour toutes les combinaisons des types de base, CHAR, BOOL, INT, REAL et STRING, comme type de destination STRING. Une conversion éventuellement indispensable est entreprise selon les règles établies.

Syntaxe

```
<Type_quelconque> << <type_quelconque>
```

Signification

<Type_quelconque> :	Variable de type CHAR, BOOL, INT, REAL ou STRING
<< :	Opérateur pour la concaténation de variables (<type_quelconque>) en chaîne de caractères composée (type STRING). Cet opérateur est également disponible seul en variante dite "unaire". permettant d'effectuer une conversion de type explicite en type STRING (à l'exception des types FRAME et AXIS) : << <type_quelconque>

Il est ainsi possible, par exemple, de composer un message ou une commande à partir de listes de textes et d'insérer des paramètres (nom de bloc, etc.) :

```
MSG(STRG_TAB[LOAD_IDX]<<NOM_BLOC)
```

PRUDENCE

Les résultats intermédiaires de la concaténation de la chaîne de caractères ne doivent pas dépasser la longueur de chaîne maximale.

Remarque

L'utilisation des types FRAME et AXIS est impossible avec l'opérateur "**<<**".

Exemples

Exemple 1 : Concaténation de chaînes de caractères

```
Code de programme                                    Commentaire
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<<"/valeur:"<<VALUE)                ; Affichage :
                                                       "Index:2/valeur:9.654"
NO_MSG:
```

Exemple 2 : Conversion explicite de type avec <<

```
Code de programme          Commentaire
DEF REAL VALUE=3.5
<<VALUE                    ; La variable indiquée de type REAL est convertie en type
                             STRING.
```

1.9.4 Conversion en minuscules/majuscules (TOLOWER, TOUPPER)

Fonction

La fonction "Conversion en minuscules/majuscules" permet d'harmoniser l'écriture au sein d'une chaîne de caractères.

Syntaxe

```
<RES_STRING>=TOUPPER("<chaîne de caractères>")
<RES_STRING>=TOLOWER("<chaîne de caractères>")
```

Signification

`TOUPPER` :	L'instruction `TOUPPER` convertit toutes les lettres d'une chaîne de caractères en **majuscules**.
`TOLOWER` :	L'instruction `TOLOWER` convertit toutes les lettres d'une chaîne de caractères en **minuscules**.
`<chaîne>` :	Chaîne de caractères à convertir Type : STRING
`<STRING_ERG>` :	Variable du résultat de la conversion Type : STRING

Exemple

Etant donné qu'il est possible de rattacher à l'interface utilisateur des indications introduites par l'utilisateur, cette fonctionnalité permet d'obtenir une présentation uniformisée en lettres minuscules ou majuscules:

Code de programme

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

1.9.5 Déterminer la longueur d'une chaîne de caractères (STRLEN)

Fonction

L'instruction STRLEN permet de déterminer la longueur d'une chaîne de caractères.

Syntaxe

```
<RES_INT>=STRLEN("<chaîne de caractères>")
```

Signification

STRLEN :	L'instruction STRLEN détermine la longueur de la chaîne de caractères indiquée. Le résultat est le nombre de caractères comptés à partir du début de la chaîne et qui ne sont pas des caractères 0.
<chaîne> :	Chaîne de caractères dont la longueur doit être déterminée Type : STRING
<INT_ERG> :	Variable du résultat de la détermination Type : INT

Exemple

En liaison avec l'accès à des caractères individuels, la fonction permet de déterminer la fin d'une chaîne de caractères :

Code de programme

```
IF (STRLEN(NOM_BLOC)>10) GOTOF ERREUR
```

1.9.6 Recherche de caractères/chaînes de caractères dans une chaîne (INDEX, RINDEX, MINDEX, MATCH)

Fonction

Cette fonctionnalité permet de rechercher des caractères isolés ou une suite de caractères dans une chaîne de caractères. Les résultats signalent à quel endroit de la chaîne de caractères se trouve le caractère / la suite de caractères qui fait l'objet de la recherche.

Syntaxe

`INT_ERG=INDEX(STRING,CHAR)` ; **Type de résultat : INT**

`INT_ERG=RINDEX(STRING,CHAR)` ; **Type de résultat : INT**

`INT_ERG=MINDEX(STRING,STRING)` ; **Type de résultat : INT**

`INT_ERG=MATCH(STRING,STRING)` ; **Type de résultat : INT**

Sémantique

Fonctions de recherche : elles fournissent la position trouvée dans la chaîne de caractères (premier paramètre). Si le caractère ou la chaîne de caractères est introuvable, le résultat est -1, la position du premier caractère étant 0.

Signification

`INDEX` :	Recherche du caractère (depuis le début) indiqué comme deuxième paramètre dans le premier paramètre.
`RINDEX` :	Recherche du caractère (depuis la fin) indiqué comme deuxième paramètre dans le premier paramètre.
`MINDEX` :	Correspond à la fonction INDEX, sauf qu'elle fournit une liste de caractères (sous forme de chaîne) avec l'indice du premier caractère trouvé.
`MATCH` :	Recherche d'une chaîne de caractères dans une chaîne de caractères.

Il est ainsi possible de décomposer les chaînes de caractères suivant des critères définis tels que les espaces et les caractères de séparation des chemins ("/").

Exemple

Décomposition d'une entrée en noms de chemin et de bloc

```
Code de programme                                          Commentaire
DEF INT CHEMIDX, PROGIDX
DEF STRING[26] ENTREE
DEF INT LISTIDX
ENTREE = "/_N_MPF_DIR/_N_EXECUTE_MPF"
LISTIDX = MINDEX (ENTREE, "M,N,O,P") + 1                   ; Le résultat de LISTIDX est la valeur
                                                             3, "N" étant le premier caractère du
                                                             paramètre ENTREE dans la liste,
                                                             depuis le début.
CHEMIDX = INDEX (ENTREE, "/") +1                           ; On a donc : CHEMIDX = 1
PROGIDX = RINDEX (ENTREE, "/") +1                          ; On a donc : PROGIDX = 12
                                                             Avec la fonction SUBSTR présentée au
                                                             chapitre suivant, il est possible de
                                                             décomposer la variable ENTREE en
                                                             "chemin" et en "bloc" :
VARIABLE = SUBSTR (ENTREE, CHEMIDX, PROGIDX-CHEMIDX-1)     ; Fournit alors "_N_MPF_DIR".
VARIABLE = SUBSTR (ENTREE, PROGIDX)                        ; Fournit alors "_N_EXECUTE_MPF".
```

1.9.7 Sélection d'une chaîne de caractères partielle (SUBSTR)

Fonction

Cette fonctionnalité permet d'extraire une chaîne partielle de caractères d'une chaîne de caractères. Pour ce faire, on précise l'indice du premier caractère et, le cas échéant, la longueur désirée. Si l'information de longueur n'est pas précisée, c'est toute la chaîne de caractères restante qui est prise en compte.

Syntaxe

`STRING_ERG = SUBSTR (STRING,INT)` ; **Type de résultat : INT**

`STRING_ERG = SUBSTR(STRING,INT, INT)` ; **Type de résultat : INT**

Sémantique

Dans le premier cas, le résultat correspond à la chaîne de caractères partielle allant de la position définie par le deuxième paramètre jusqu'à la fin de la chaîne.

Dans le deuxième cas, le résultat est limité à la longueur maximale définie par le troisième paramètre.

Si la position de début se situe après la fin de la chaîne de caractères, le résultat est une chaîne vide (" ").

Si la position de début ou la longueur sont négatives, une alarme est déclenchée.

Exemple

Code de programme	Commentaire
`DEF STRING[29] RES`	
`RES = SUBSTR ("ACQUITTEMENT:10 à 99", 10, 2)`	`; On a donc : RES == "10"`

1.9.8 Sélection d'un caractère individuel (STRINGVAR, STRINGFELD)

Fonction

Cette fonctionnalité permet de sélectionner des caractères individuels dans une chaîne. Cela est valable pour l'accès en lecture aussi bien que pour l'accès en écriture.

Syntaxe

CHAR_ERG = STRINGVAR [IDX] ; **Type de résultat : CHAR**

CHAR_ERG = STRINGFELD [IDX_FELD, IDX_CHAR] ; **Type de résultat : CHAR**

Sémantique

Le caractère situé à l'emplacement spécifié dans la chaîne de caractères est lu/écrit. Si l'indication de position est négative ou plus grande que la chaîne de caractères, une alarme est déclenchée.

Exemple de messages :

Installation d'un descripteur d'axe dans une chaîne de caractères préétablie.

```
Code de programme                                          Commentaire
DEF STRING [50] MESSAGE = "Axe n a atteint sa position"
MESSAGE [6] = "X"
MSG (MESSAGE)                                              ; Fournit le message "Axe X a
                                                             atteint sa position".
```

Paramètres

L'accès à un caractère individuel est possible uniquement dans les variables définies par l'utilisateur (données LUD, GUD et PUD).

De plus, dans le cas de l'appel d'un sous-programme, ce mode d'accès n'est possible que pour des paramètres de type "Call-By-Value".

Exemples

Exemple 1 : accès d'un caractère individuel à un paramètre système, paramètre machine, …

```
Code de programme                                  Commentaire
DEF STRING [50] STRG
DEF CHAR ACQUITTEMENT
…
STRG = $P_MMCA
ACQUITTEMENT = STRG [0]                            ;  Exploitation des composantes de
                                                      l'acquittement
```

Exemple 2 : accès d'un caractère individuel dans un paramètre de type "Call-By-Reference"

```
Code de programme                                  Commentaire
DEF STRING [50] STRG
DEF CHAR CHR1
EXTERN UP_CALL (VAR CHAR1)                         ;  Paramètre de type "Call-By-
                                                      Reference" !
…
CHR1 = STRG [5]
UP_CALL (CHR1)                                     ;  Call-By-Reference
STRG [5] = CHR1
```

1.10 Sauts de programme

1.10.1 Retour au début du programme (GOTOS)

Fonction

Pour une répétition de programme, l'instruction GOTOS permet de retourner au début d'un programme principal ou d'un sous-programme.

Pour le retour au début du programme, les paramètres machine permettent de régler :

- la mise à "0" du temps d'exécution du programme,
- l'incrémentation du comptage de pièces de la valeur "1".

Syntaxe

```
GOTOS
```

Signification

GOTOS : Instruction de saut avec destination au début du programme.

L'exécution est commandée par le signal d'interface CN/AP :

DB21, ... DBX384.0 (commander un saut de programme)

Valeur :	Signification :
0	Sans retour au début du programme. Après GOTOS, l'exécution du programme reprend au bloc suivant du programme pièce.
1	Retour au début du programme et répétition du programme pièce.

Conditions marginales

- GOTOS déclenche un STOPRE interne (arrêt du prétraitement).
- Dans le cas d'un programme pièce avec des définitions de données (variables LUD), GOTOS exécute un retour au premier bloc de programme qui suit la section de définition, autrement dit l'exécution des définitions de données n'est pas répétée. Les variables définies gardent donc la valeur atteinte dans le bloc GOTOS et ne sont pas remises aux valeurs par défaut programmées dans la section de définition.
- L'instruction GOTOS n'est pas disponible dans les actions synchrones et les cycles technologiques.

Exemple

Code de programme	Commentaire
N10 ...	; Début du programme.
...	
N90 GOTOS	; Saut au début du programme.
...	

1.10.2 Sauts de programme sur repères de saut (IF, GOTOB, GOTOF, GOTO, GOTOC)

Fonction

Des repères de saut (étiquettes) à rejoindre au sein d'un programme avec les instructions GOTOF, GOTOB, GOTO et GOTOC peuvent être définis dans ce programme. L'exécution du programme reprend alors à l'instruction qui suit immédiatement le repère de saut. Il est donc possible de réaliser des sauts à l'intérieur du programme.

Outre les repères de saut, les destinations de saut peuvent être également des numéros de bloc principal ou auxiliaire.

Si une condition de saut (IF ...) est formulée avant l'instruction de saut, le saut de programme n'est exécuté que si cette condition est remplie.

Syntaxe

```
GOTOB <destination de saut>
IF <condition de saut> = TRUE GOTOB <destination de saut>
GOTOF <destination de saut>
IF <condition de saut> = TRUE GOTOF <destination de saut>
GOTO <destination de saut>
IF <condition de saut> = TRUE GOTO <destination de saut>
GOTOC <destination de saut>
IF <condition de saut> = TRUE GOTOC <destination de saut>
```

Signification

GOTOB :	Instruction de saut avec destination de saut en direction du début du programme.
GOTOF :	Instruction de saut avec destination de saut en direction de la fin du programme.
GOTO :	Instruction de saut avec recherche de destination de saut. La destination est d'abord recherchée en direction de la fin du programme, puis en direction du début du programme.
GOTOC :	Même effet que GOTO, sauf que l'alarme 14080 "Destination de saut pas trouvée" est inhibée. Cela signifie que l'exécution du programme n'est pas annulée en cas de recherche de destination de saut sans résultat, mais qu'elle reprend à la ligne de programme qui suit l'instruction GOTOC.

`<destination de saut>` :	Paramètre de la destination de saut Les indications possibles sont :	
	<repère de saut> :	La destination de saut correspond au repère de saut défini dans le programme avec un nom défini par l'utilisateur : <`repère de saut> :`
	<numéro de bloc> :	La destination de saut est un numéro de bloc principal ou auxiliaire (par exemple : `200`, `N300`)
	Variable de type STRING :	Destination de saut variable. La variable correspond à un repère de saut ou à un numéro de bloc.
`IF` :	Mot clé pour formuler la condition de saut. Pour la condition de saut, vous pouvez utiliser toutes les opérations relationnelles et logiques (résultat : TRUE ou FALSE). Le saut est exécuté si le résultat de l'opération est TRUE.	

Remarque

Repères de saut (étiquettes)

Les repères de saut figurent toujours au début d'un bloc. S'il existe un numéro de programme, le repère de saut figure immédiatement après le numéro de bloc.

Les règles suivantes s'appliquent à la définition des repères de saut :

- Nombre de caractères :
  - Minimum 2
  - Maximum 32
- Caractères autorisés :
  - Caractères alphabétiques
  - Chiffres
  - Traits de soulignement
- Les deux premiers caractères doivent être obligatoirement des lettres ou des traits de soulignement.
- Le nom d'un repère de saut est suivi de deux points (":").

Conditions marginales

- La destination du saut doit être un bloc avec un repère de saut ou un numéro de bloc figurant **dans** le programme.
- Une instruction de saut sans condition de saut doit être programmée dans un bloc spécifique. Cette restriction ne s'applique pas aux instructions de saut avec des conditions de saut. Dans ce cas, plusieurs instructions de saut peuvent être formulées dans un bloc.
- Dans le cas de programmes comportant des instructions de saut sans condition de saut, l'instruction de fin de programme `M2`/`M30` peut figurer avant la fin du programme.

Exemples

Exemple 1 : Sauts sur repères de saut

Code de programme	Commentaire
N10 …	
N20 GOTOF Label_1	; Saut en direction de la fin du programme sur le repère de saut "Label_1".
N30 …	
N40 Label_0: R1=R2+R3	; Définition du repère de saut "Label_0".
N50 …	
N60 Label_1:	; Définition du repère de saut "Label_1".
N70 …	
N80 GOTOB Label_0	; Saut en direction du début du programme sur le repère de saut "Label_0".
N90 …	

Exemple 2 : Saut indirect sur numéro de bloc

Code de programme	Commentaire
N5 R10=100	
N10 GOTOF "N"<<R10	; Saut sur le bloc dont le numéro de bloc est indiqué dans R10.
...	
N90 ...	
N100 ...	; Destination de saut
N110 ...	
...	

Exemple 3 : Saut sur destination de saut variable

Code de programme	Commentaire
DEF STRING[20] DESTINATION	
DESTINATION = "repère2"	
GOTOF DESTINATION	; Saut en direction de la fin du programme sur la destination de saut variable DESTINATION.
Repère1: T="foret1"	
...	
Repère2: T="foret2"	; Destination de saut
...	

Exemple 4 : Saut avec condition de saut

Code de programme	Commentaire
N40 R1=30 R2=60 R3=10 R4=11 R5=50 R6=20	; Affectation des valeurs initiales.
N41 LA1: G0 X=R2*COS(R1)+R5 Y=R2*SIN(R1)+R6	; Définition du repère de saut LA1.
N42 R1=R1+R3 R4=R4-1	
N43 IF R4>0 GOTOB LA1	; Si la condition de saut est remplie, saut en direction du début du programme sur le repère de saut LA1.
N44 M30	; Fin du programme

1.10.3 Saut de programme (CASE ... OF ... DEFAULT ...)

Fonction

La fonction CASE permet de vérifier la valeur actuelle (type : INT) d'une variable ou d'une fonction de calcul et de sauter à différents endroits dans le programme en fonction du résultat obtenu.

Syntaxe

```
CASE(<expression>) OF <constante_1> GOTOF <destination de saut_1>
<constante_2> GOTOF <destination de saut_2> ... DEFAULT GOTOF
<destination de saut_n>
```

Signification

CASE :	Instruction de saut
<expression> :	Variable ou fonction de calcul
OF :	Mot clé pour formuler les sauts de programme conditionnels.
<constante_1> :	Première valeur constante indiquée pour la variable ou la fonction de calcul Type : INT
<constante_2> :	Deuxième valeur constante indiquée pour la variable ou la fonction de calcul Type : INT
DEFAULT :	Au cas où la variable ou la fonction de calcul ne prend aucune des valeurs constantes indiquées, la destination du saut peut être définie par l'instruction DEFAULT. **Remarque :** Au cas où l'instruction DEFAULT n'est pas programmée, le bloc qui suit l'instruction CASE devient destination du saut.

GOTOF :	Instruction de saut avec destination de saut en direction de la fin du programme. Au lieu de GOTOF, il est également possible de programmer toutes les autres instructions GOTO (voir chapitre "Sauts de programme sur repères de saut").
<destination de saut_1> :	Destination de saut si la valeur de la variable ou de la fonction de calcul correspond à la première constante indiquée. La destination de saut peut être indiquée de la manière suivante : **<repère de saut>** : La destination de saut correspond au repère de saut défini dans le programme avec un nom défini par l'utilisateur : <repère de saut> : **<numéro de bloc>** : La destination de saut est un numéro de bloc principal ou auxiliaire (par exemple : 200, N300) Variable de type STRING : Destination de saut variable. La variable correspond à un repère de saut ou à un numéro de bloc.
<destination de saut_2> :	Destination de saut si la valeur de la variable ou de la fonction de calcul correspond à la deuxième constante indiquée.
<destination de saut_n> :	Destination de saut si la valeur de la variable ou de la fonction de calcul ne correspond à aucune des valeurs constantes indiquées.

Exemple

Code de programme

```
...
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
...
```

L'instruction CASE de N30 définit les possibilités de saut de programme suivantes :

1. Si la valeur de la fonction de calcul VAR1+VAR2-VAR3 = 7, la destination de saut correspond au bloc avec la définition du repère de saut "Label_1" (→ N40).
2. Si la valeur de la fonction de calcul VAR1+VAR2-VAR3 = 9, la destination de saut correspond au bloc avec la définition du repère de saut "Label_2" (→ N50).
3. Si la valeur de la fonction de calcul VAR1+VAR2-VAR3 n'est ni égale à 7 ni à 9, la destination de saut correspond au bloc avec la définition du repère de saut "Label_3" (→ N60).

1.11 Répétition de sections de programme (REPEAT, REPEATB, ENDLABEL, P)

Fonction

Cette fonction permet la répétition de sections de programme dans un ordre quelconque.

Les lignes ou sections de programme à répéter sont repérées par des repères de saut (étiquettes).

Remarque

Repères de saut (étiquettes)

Les repères de saut figurent toujours au début d'un bloc. S'il existe un numéro de programme, le repère de saut figure immédiatement après le numéro de bloc.

Les règles suivantes s'appliquent à la définition des repères de saut :

- Nombre de caractères :
  - Minimum 2
  - Maximum 32
- Caractères autorisés :
  - Caractères alphabétiques
  - Chiffres
  - Traits de soulignement
- Les deux premiers caractères doivent être obligatoirement des lettres ou des traits de soulignement.
- Le nom d'un repère de saut est suivi de deux points (":").

Syntaxe

1. Répétition d'une ligne de programme individuelle :

```
<repère de saut> : ...
...
REPEATB <repère de saut> P=<n>
...
```

2. Répétition d'une section de programme entre le repère de saut et l'instruction REPEAT :

```
<repère de saut> : ...
...
REPEAT <repère de saut> P=<n>
...
```

3. Répétition d'une section entre deux repères de saut :

```
<repère de saut de début> : ...
...
<repère de saut de fin> : ...
...
REPEAT <repère de saut de début> <repère de saut de fin> P=<n>
...
```

Remarque

Il n'est pas possible de placer l'instruction `REPEAT` entre les deux repères de saut. Si le `<repère de saut de début>` est placé devant l'instruction `REPEAT` et si le `<repère de saut de fin>` n'est pas atteint avant l'instruction `REPEAT`, la répétition a lieu entre le `<repère de saut de début>` et l'instruction `REPEAT`.

4. Répétition d'une section entre le repère de saut et ENDLABEL :

```
<repère de saut> : ...
...
ETIQUETTE_FIN: ...
...
REPEAT <repère de saut> P=<n>
...
```

Remarque

Il n'est pas possible de placer l'instruction `REPEAT` entre le `<repère de saut>` et `ENDLABEL`. Si le `<repère de saut>` est placé devant l'instruction `REPEAT` et si le `ENDLABEL` n'est pas atteint avant l'instruction `REPEAT`, la répétition a lieu entre le `<repère de saut>` et l'instruction `REPEAT`.

Signification

REPEATB :	Instruction de répétition d'une ligne de programme
REPEAT :	Instruction de répétition d'une section de programme
<repère de saut> :	Le <repère de saut> repère : • la ligne de programme à répéter (avec REPEATB) ou • le début de la section de programme à répéter (avec REPEAT) La ligne de programme repérée par le <repère de saut> peut se trouver avant ou après l'instruction REPEAT/REPEATB. La recherche commence en direction du début du programme. Si le repère de saut n'est pas trouvée dans cette direction, la recherche est s'effectue en direction de la fin du programme. **Exception :** si la section de programme doit être répétée entre le repère de saut et l'instruction REPEAT (voir le point 2 sous Syntaxe), la ligne de programme repérée par le <repère de saut> doit se trouver **devant** l'instruction REPEAT, car dans ce cas, la rechercher s'effectue **uniquement** en direction du début du programme. Si la ligne contenant le <repère de saut> contient d'autres instructions, celles-ci sont exécutées une nouvelle fois à chaque répétition.
ENDLABEL :	Mot-clé qui repère la fin d'une section de programme à répéter Si la ligne contenant ENDLABEL contient d'autres instructions, celles-ci sont exécutées une nouvelle fois à chaque répétition. ENDLABEL peut être utilisée plusieurs fois dans le programme.
P :	Adresse pour l'indication du nombre de répétitions
<n> :	Nombre de répétitions de la section de programme Type : INT La section de programme à répéter est répétée <n> fois. Après la dernière répétition, l'exécution du programme reprend à la ligne suivant celle qui contient REPEAT/REPEATB. **Nota :** si P=<n> n'est pas indiqué, la section de programme a répéter est répétée exactement une fois.

Exemples

Exemple 1 : répéter des lignes de programme individuelles

```
Code de programme                    Commentaire
N10 POSITION1: X10 Y20
N20 POSITION2: CYCLE(0,,9,8)         ; Cycle de positions
N30 ...
N40 REPEATB POSITION1 P=5            ; Exécuter le bloc N10 cinq fois.
N50 REPEATB POSITION2                ; Exécuter le bloc N20 une fois.
N60 ...
N70 M30
```

Exemple 2 : répéter la section de programme entre le repère de saut et l'instruction REPEAT

```
Code de programme          Commentaire
N5 R10=15
N10 DEBUT : R10=R10+1      ; Largeur
N20 Z=10-R10
N30 G1 X=R10 F200
N40 Y=R10
N50 X=-R10
N60 Y=-R10
N70 Z=10+R10
N80 REPEAT DEBUT P=4       ; Exécuter la section N10 à N70 quatre fois.
N90 Z10
N100 M30
```

Exemple 3 : répéter la section entre deux repères de saut

```
Code de programme              Commentaire
N5 R10=15
N10 DEBUT : R10=R10+1          ; Largeur
N20 Z=10-R10
N30 G1 X=R10 F200
N40 Y=R10
N50 X=-R10
N60 Y=-R10
N70 FIN : Z=10
N80 Z10
N90 CYCLE(10,20,30)
N100 REPEAT DEBUT FIN P=3      ; Exécuter la section N10 à N70 trois fois.
N110 Z10
N120 M30
```

Exemple 4 : répéter la section entre le repère de saut et ENDLABEL

```
Code de programme          Commentaire
N10 G1 F300 Z-10
N20 DEBUT1:
N30 X10
N40 Y10
N50 DEBUT2:
N60 X20
N70 Y30
N80 ETIQUETTE_FIN: Z10
N90 X0 Y0 Z0
N100 Z-10
N110 DEBUT3: X20
N120 Y30
N130 REPEAT DEBUT3 P=3      ; Exécuter la section N110 à N120 trois fois.
N140 REPEAT DEBUT2 P=2      ; Exécuter la section N50 à N80 deux fois.
N150 M100
N160 REPEAT DEBUT1 P=2      ; Exécuter la section N20 à N80 deux fois.
N170 Z10
N180 X0 Y0
N190 M30
```

Exemple 5 : réaliser un fraisage, des trous taraudés avec différentes technologies

```
Code de programme          Commentaire
N10 FORET_CENTRER()        ; Mettre en place le foret à centrer.
N20 POS_1:                 ; Réseau de positions 1
N30 X1 Y1
N40 X2
N50 Y2
N60 X3 Y3
N70 ENDLABEL :
N80 POS_2 :                ; Réseau de positions 2
N90 X10 Y5
N100 X9 Y-5
N110 X3 Y3
N120 ENDLABEL :
N130 FORET()               ; Mettre en place le foret et le cycle de perçage.
N140 TARAUD(6)             ; Mettre en place le taraud M6 et le cycle de taraudage.
N150 REPEAT POS_1          ; Répéter la section de programme une fois à partir de
                             POS_1 jusqu'à ENDLABEL.
N160 FORET()               ; Mettre en place le foret et le cycle de perçage.
```

```
Code de programme          Commentaire
N170 TARAUD(8)             ; Mettre en place le taraud M8 et le cycle de taraudage.
N180 REPEAT POS_2          ; Répéter la section de programme une fois à partir de
                             POS_2 jusqu'à ENDLABEL.
N190 M30
```

Informations complémentaires

- Des imbrications sont autorisées dans une section de programme à répéter. Chaque appel correspond à un niveau de sous-programme.
- Si M17 ou RET sont programmés pendant l'exécution de la répétition d'une section de programme, celle-ci est annulée. L'exécution du programme reprend au bloc suivant la ligne REPEAT.
- Dans l'affichage du programme en cours, la répétition de la section de programme est visualisée en tant que niveau spécifique de sous-programme.
- Si un abandon est déclenché pendant l'exécution de la section de programme, l'exécution du programme reprend après le bloc comportant l'instruction REPEAT.

  Exemple :

```
Code de programme              Commentaire
N5 R10=15
N10 DEBUT : R10=R10+1          ; Largeur
N20 Z=10-R10
N30 G1 X=R10 F200
N40 Y=R10                      ; Abandon
N50 X=-R10
N60 Y=-R10
N70 FIN : Z10
N80 Z10
N90 CYCLE(10,20,30)
N100 REPEAT DEBUT FIN P=3
N120 Z10                       ; Poursuivre l'exécution du programme.
N130 M30
```

- Structures de contrôle et répétition de sections de programme peuvent être combinées. Il y a cependant lieu d'éviter les chevauchements. Il est recommandé que la répétition d'une section de programme se trouve à l'intérieur d'une branche de structure de contrôle ou d'une structure de contrôle se trouvant à l'intérieur d'une répétition de section de programme.
- En cas de programmation mixte de sauts et d'une répétition de section de programme, les blocs sont exécutés de façon purement séquentielle. En cas de saut à partir d'une boucle de répétition de section de programme par exemple, l'exécution se poursuit jusqu'à ce que la fin de la section de programme à répéter soit trouvée.

Exemple :

```
Code de programme
N10 G1 F300 Z-10
N20 DEBUT1:
N30 X=10
N40 Y=10
N50 GOTOF DEBUT2
N60 ENDLABEL:
N70 DEBUT2:
N80 X20
N90 Y30
N100 ENDLABEL: Z10
N110 X0 Y0 Z0
N120 Z-10
N130 REPEAT DEBUT1 P=2
N140 Z10
N150 X0 Y0
N160 M30
```

Remarque

Il est recommandé que l'instruction `REPEAT` se trouve après les blocs de déplacement.

1.12 Structures de contrôle

Fonction

En version standard, la commande traite les blocs CN dans l'ordre de leur programmation.

Il est possible de varier cet ordre en programmant des blocs de programme et des boucles de programme alternatifs. La programmation de ces structures de contrôle s'effectue avec les éléments de structures de contrôle (mots clés) `IF...ELSE`, `LOOP`, `FOR`, `WHILE` et `REPEAT`.

⚠ PRUDENCE

Les structures de contrôle ne sont possibles qu'à l'intérieur de la partie instructions d'un programme. Les définitions dans l'en-tête de programme ne peuvent pas être exécutées de façon conditionnelle ou répétitive.

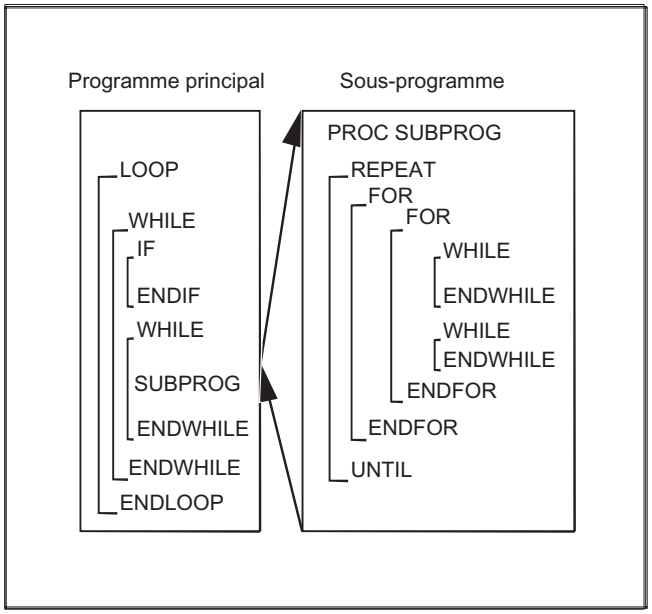
Les mots-clés pour les structures de contrôle, tout comme les destinations de sauts, ne doivent pas être utilisés dans des macros. Ceci ne fait pas l'objet d'une vérification lors de la définition des macros.

Prise d'effet

Les structures de contrôle sont valables localement dans le programme.

Niveaux d'imbrication

Une imbrication de 16 structures de contrôle au maximum est possible à l'intérieur de chaque niveau de sous-programme.



Durée d'exécution des programmes

En mode interpréteur standard, l'utilisation de sauts permet d'obtenir une exécution du programme plus rapide qu'avec les structures de contrôle.

Dans des cycles précompilés, il n'y a pas de différence entre les sauts dans le programme et les structures de contrôle.

Conditions marginales

- Les blocs comportant des éléments de structures de contrôle ne peuvent pas être déclarés optionnels.
- Les repères de saut (étiquettes) ne sont pas autorisés dans les blocs contenant des éléments de structures de contrôle.
- Les structures de contrôle sont exécutées de façon interprétative. En cas d'identification d'une fin de boucle, le début de la boucle est recherché compte tenu des structures de contrôle ainsi trouvées. Par conséquent, en mode interpréteur, la structure d'un programme n'est pas complètement testée.
- En principe, il est recommandé de ne pas utiliser en commun des structures de contrôle et des sauts dans le programme.
- Si des cycles subissent un prétraitement, l'imbrication correcte des structures de contrôle est vérifiée.

1.12.1 Boucle de programme avec alternative (IF, ELSE, ENDIF)

Fonction

Une construction avec `IF` et `ELSE` s'utilise lorsque la boucle de programme doit contenir un bloc de programme alternatif. Si la condition `IF` est remplie, le bloc de programme qui suit `IF` est exécuté. Si la condition `IF` n'est **pas** remplie, le bloc de programme alternatif qui suit `ELSE` est exécuté.

Remarque

Si une alternative n'est pas nécessaire, il est également possible de programmer une boucle IF sans instruction `ELSE` ni bloc de programme placé après `ELSE`.

Syntaxe

```
IF <condition>
...
ELSE
...
ENDIF
```

Signification

`IF` :	Début de la boucle IF.
`ELSE` :	Début du bloc de programme alternatif.
`ENDIF` :	Fin de la boucle IF provoquant un retour au début de la boucle.
`<condition >` :	Condition qui détermine le bloc de programme à exécuter.

Exemple

Sous-programme de changement d'outil

```
Code de programme                                   Commentaire
PROC L6                                             ; Routine de changement d'outil
N500 DEF INT TNR_COURANT                            ; Variable du numéro T actif
N510 DEF INT TNR_PRESELECTION                       ; Variable du numéro T
                                                      présélectionné
                                                    ; Détermination de l'outil
                                                      courant
N520 STOPRE
N530 IF $P_ISTEST                                   ; En mode de test de programme
                                                      ...
N540 TNR_COURANT = $P_TOOLNO                        ; ... lecture de l'outil
                                                      "courant" du contexte du
                                                      programme.
N550 ELSE                                           ; Sinon ...
N560 TNR_COURANT = $TC_MPP6[9998,1]                 ; ... lecture de l'outil de la
                                                      broche.
N570 ENDIF
N580 GETSELT(TNR_PRESELECTION)                      ; Lecture du numéro T de l'outil
                                                      présélectionné sur la broche.
N590 IF TNR_COURANT <> TNR_PRESELECTION             ; Si l'outil présélectionné
                                                      n'est pas encore l'outil
                                                      courant, ...
N600 G0 G40 G60 G90 SUPA X450 Y300 Z300 D0           ; ... accostage de la position
                                                      de changement d'outil ...
N610 M206                                           ; ... et exécution du changement
                                                      d'outil.
N620 ENDIF
N630 M17
```

1.12.2 Boucle de programme sans fin (LOOP, ENDLOOP)

Fonction

La boucle infinie trouve son application dans des programmes infinis. Un retour au début de boucle a toujours lieu en fin de boucle.

Syntaxe

```
LOOP
...
ENDLOOP
```

Signification

`LOOP` : Début de la boucle sans fin.

`ENDLOOP` : Fin de la boucle provoquant un retour au début de la boucle.

Exemple

Code de programme

```
...
LOOP
MSG ("Pas de changement d'outil actif")
M0
STOPRE
ENDLOOP
...
```

1.12.3 Boucle de comptage (FOR ... TO ..., ENDFOR)

Fonction

La boucle de comptage est utilisée lorsqu'une section de programme doit être répétée un certain nombre de fois.

Syntaxe

```
FOR <variable> = <valeur initiale> TO <valeur finale>
...
ENDFOR
```

Signification

`FOR` :	Début de la boucle de comptage.
`ENDFOR` :	Fin de la boucle provoquant un retour au début de la boucle tant que la valeur finale du comptage n'est pas atteinte.
`<variable>` :	Variable de comptage incrémentée de la valeur "1" à chaque exécution, de la valeur initiale à la valeur finale. type INT ou REAL **Remarque :** Le type REAL est pris en compte par exemple lorsque le paramètre R est programmé pour une boucle de comptage. Si la variable de comptage est de type REAL, sa valeur est arrondie à un nombre entier.
`<valeur initiale>` :	Valeur initiale du comptage Condition : La valeur initiale doit être inférieure à la valeur finale.
`<valeur finale>` :	Valeur finale du comptage

Exemples

Exemple 1 : Variable INT ou paramètre R en tant que variable de comptage

Variable INT en tant que variable de comptage :

```
Code de programme                    Commentaire
DEF INT iVARIABLE1
R10=R12-R20*R1 R11=6
FOR iVARIABLE1 = R10 TO R11          ; Variable de comptage = variable INT
 R20=R21*R22+R33
ENDFOR
M30
```

Paramètre R en tant que variable de comptage :

```
Code de programme                    Commentaire
R11=6
FOR R10=R12-R20*R1 TO R11            ; Variable comptage = paramètre R (variable REAL)
 R20=R21*R22+R33
ENDFOR
M30
```

Exemple 2 : Fabrication d'un nombre de pièces défini

```
Code de programme                    Commentaire
DEF INT SERIE                        ; Définition de la variable INT avec le nom
                                       "SERIE".
FOR SERIE = 0 TO 100                 ; Début de la boucle de comptage. La variable
                                       "SERIE" est incrémentée de la valeur initiale
                                       "0" à la valeur finale "100".
G01 …
ENDFOR                               ; Fin de la boucle de comptage.
M30
```

1.12.4 Boucle de programme avec condition en début de boucle (WHILE, ENDWHILE)

Fonction

Dans une boucle WHILE, la condition figure au début de la boucle. La boucle WHILE est exécutée tant que la condition est remplie.

Syntaxe

```
WHILE <condition>
...
ENDWHILE
```

Signification

`WHILE` :	Début de la boucle de programme.
`ENDWHILE` :	Fin de la boucle provoquant un retour au début de la boucle.
`<condition >` :	Condition à remplir pour l'exécution de la boucle WHILE.

Exemple

Code de programme	Commentaire
...	
WHILE $AA_IW[AXE DE PERÇAGE] > -10	; Appel de la boucle WHILE sous la condition suivante : la consigne SCP courante de l'axe de perçage doit être supérieure à -10.
G1 G91 F250 AX[AXE DE PERÇAGE] = -1	
ENDWHILE	
...	

1.12.5 Boucle de programme avec condition en fin de boucle (REPEAT, UNTIL)

Fonction

Dans une boucle REPEAT, la condition figure à la fin de la boucle. La boucle REPEAT est exécutée une fois, puis répétée jusqu'à ce que la condition soit remplie.

Syntaxe

```
REPEAT
...
UNTIL <condition>
```

Signification

`REPEAT` :	Début de la boucle de programme.
`UNTIL` :	Fin de la boucle provoquant un retour au début de la boucle.
`<condition >` :	Condition à remplir pour que la boucle REPEAT ne soit plus exécutée.

Exemple

Code de programme	Commentaire
...	
REPEAT	; Appel de la boucle REPEAT.
...	
UNTIL ...	; Vérifie si la condition est remplie.
...	

1.12.6 Exemple de programme avec des structures de contrôle imbriquées

Code de programme	**Commentaire**
LOOP	
IF NOT $P_SEARCH	; Pas de recherche de bloc
G01 G90 X0 Z10 F1000	
WHILE $AA_IM[X] <= 100	
G1 G91 X10 F500	; Réseau de trous
Z-F100	
Z5	
ENDWHILE	
Z10	
ELSE	
MSG("pas de perçage pendant la recherche de bloc")	
ENDIF	
$A_OUT[1]=1	; Plaque à percer suivante
G4 F2	
ENDLOOP	
M30	

1.13 Coordination de programmes (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM)

Fonction

Canaux

Un canal est susceptible d'exécuter son propre programme, indépendamment des autres canaux. C'est par ce moyen que les axes et broches qui lui sont affectés temporairement, sont contrôlables par programme.

A la mise en service, on peut créer deux canaux ou plus pour la commande.

Coordination de programmes

Si plusieurs canaux sont partie prenante dans la réalisation d'une pièce, une synchronisation du déroulement des programmes peut alors devenir nécessaire.

Pour réaliser cette synchronisation de programmes, il existe des instructions spéciales (ordres) qui doivent figurer dans un bloc spécifique.

Remarque

La coordination de programme est également possible dans un canal séparé.

instructions de coordination de programmes

- Indication du chemin absolu

	Le chemin absolu est formé selon les règles suivantes :
INIT (n,"/_HUGO_DIR/_N_name_MPF") ou	- répertoire courant/_N_nom_MPF "répertoire courant" est le répertoire pièce sélectionné ou le répertoire standard /_N_MPF_DIR.
INIT (n,"/_N_MPF_DIR/_N_nom_MPF")	- Sélection d'un programme pour exécution dans un canal déterminé : n : numéro du canal, valeur selon la configuration de la commande - Nom complet du programme
Exemple : INIT(2,"/_N_WKS_DIR/_DRESSAGE_MPF") G01F0.1 START INIT (2,"/_N_WKS_DIR/_N_SOUS_1_SPF")	**Jusqu'à la version 3 du logiciel :** Entre une instruction **init** (sans synchronisation) et un **Départ programme** doit figurer au moins un bloc exécutable. En cas d'appel de sous-programme, "_SPF" doit figurer dans l'indication de chemin.

- Indication du chemin relatif

Exemple :

Pour l'indication du chemin relatif, les règles à suivre sont les mêmes que pour les appels de sous-programmes.

```
INIT(2,"DRESSAGE")
INIT(3,"SOUS_1_SPF")
```

En cas d'appel de sous-programme, "_SPF" doit figurer dans le nom du sous-programme.

Paramètres

Les variables à la disposition commune des canaux (variables globales spécifiques à NCK) peuvent être utilisées pour l'échange de données entre programmes. Sinon l'élaboration des programmes sera entreprise séparément pour chaque canal.

`INIT(n, indication du chemin, mode d'acquittement)`	Instruction sur le traitement dans un canal. Sélection d'un programme donné avec indication du chemin absolue ou relative.
`START (n, n)`	Démarrage des programmes sélectionnés dans les autres canaux. n,n : Enumération des numéros de canal : valeur selon la configuration de la commande
`WAITM (N°_marque, n, n, ...)`	Définition de la marque "N°_marque" dans le propre canal. Fin du bloc précédent avec arrêt précis. Attente des marques avec le même "N°_marque" dans les canaux indiqués "n" { (il n'est pas nécessaire que le propre canal soit indiqué). La marque est effacée après la synchronisation. Jusqu'à 10 marques maxi peuvent être définies dans un même canal.
`WAITMC (N°_marque, n, n,`	Définition de la marque "N°_marque" dans le propre canal. L'arrêt précis est enclenché uniquement si les autres canaux n'ont pas encore atteint la marque. Attente de la marque avec le même "N°_marque" dans les canaux indiqués "n" (il n'est pas nécessaire que le propre canal soit indiqué). Dès que la marque "N°_marque" est atteinte dans les canaux indiqués, poursuite de l'usinage sans mettre fin à l'arrêt précis.

`WAITE (n, n, ...)`	Attente de la fin de programme dans les canaux indiqués (ne pas indiquer le propre canal). Exemple : programmation d'un arrêt temporisé après l'instruction START.
	`N30 START(2)`
	`N31 G4 F0.01`
	`N40 WAITE(2)`

`SETM (N°_marque, N°_marque,`	Définition des marques "N°_marque" dans le

	propre canal, sans effet sur l'usinage en cours. SETM() conserve sa validité au-delà de RESET et de Départ programme.
CLEARM (N°_marque, N°_marque,	Effacement des marques "N°_marque" dans le propre canal, sans effet sur l'usinage en cours. Toutes les marques dans le canal peuvent être effacées avec CLEARM(). CLEARM (0) efface la marque "0". CLEARM() conserve sa validité au-delà de RESET et de Départ programme.
n	Numéro ou nom de canal correspondant

Remarque

Toutes les instructions ci-dessus doivent figurer dans des blocs qui leur sont propres.

Le nombre de marques dépend de l'UC en place.

Numéros de canaux

Pour les canaux à coordonner, jusqu'à 10 canaux peuvent être indiqués par leur numéro (valeur entière).

Noms des canaux

Les noms de canaux doivent être convertis en numéros par des variables (voir le chapitre "Variables et paramètres de calcul") ou, à la place de numéros de canaux, les noms de canaux (descripteur ou mot clé) définis par $MC_CHAN_NAME peuvent également être programmés. Les noms définis doivent être conformes aux conventions linguistiques CN (autrement dit, les deux premiers caractères sont des lettres ou des caractères de soulignement).

⚠ PRUDENCE

Protéger l'affectation des numéros pour éviter toute modification accidentelle.

Ces noms ne doivent pas préalablement exister avec une autre signification dans la CN, par exemple comme mot-clé, comme commande CN, comme nom d'axe, etc.

SETM() et CLEARM()

`SETM()` et `CLEARM()` peuvent également être programmés à partir d'une action synchrone. Voir chapitre "Activer/effacer la marque d'attente : SETM CLEARM"

Exemple

Au canal avec le nom "MACHINE", on affecte le numéro de canal 1,

au canal avec le nom "CHARGEUR", on affecte le numéro de canal 2 :

```
DEF INT MACHINE=1, CHARGEUR=2
```

Les variables reçoivent les mêmes noms que les canaux.

L'instruction `START` s'appelle donc :

```
START(MACHINE)
```

Exemple de coordination de programmes

Canal 1 :

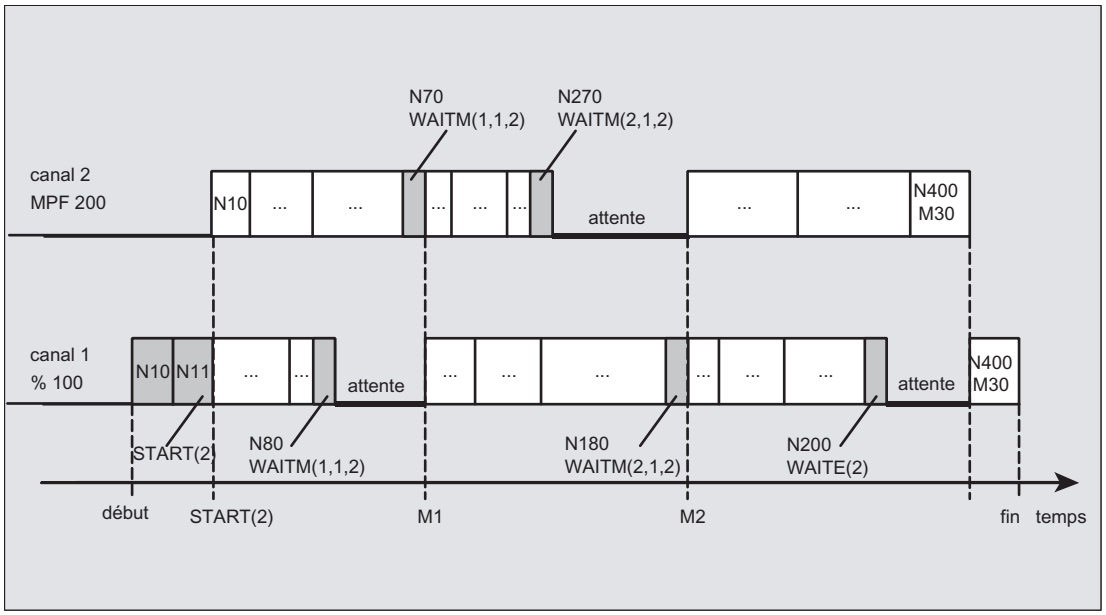
_N_MPF100_MPF

```
Code de programme          Commentaire
N10 INIT(2,"MPF200")
N11 START (2)              ; Exécution dans le canal 2
...
N80 WAITM(1,1,2)           ; Attente de la marque WAIT 1 dans le canal 1 et
                             dans le canal 2 et poursuite de l'exécution dans le
                             canal 1
...
N180 WAITM(2,1,2)          ; Attente de la marque WAIT 2 dans le canal 1 et
                             dans le canal 2 et poursuite de l'exécution dans le
                             canal 1
...
N200 WAITE(2)              ; Attente de la fin de programme dans le canal 2
N201 M30                   ; Fin de programme dans le canal 1, fin globale
...
```

Canal 2 :

_N_MPF200_MPF

```
Code de programme      Commentaire
;$PATH=/_N_MPF_DIR
                       ; Exécution dans le canal 2
N70 WAITM(1,1,2)       ; Attente de la marque WAIT 1 dans le canal 1 et dans le
                         canal 2 et poursuite de l'exécution dans le canal 1
...
N270 WAITM(2,1,2)      ; Attente de la marque WAIT 2 dans le canal 1 et dans le
                         canal 2 et poursuite de l'exécution dans le canal 2
...
N400 M30               ; Fin de programme dans le canal 2
```



Exemple : Programme à partir d'une pièce

```
Code de programme
N10 INIT(2,"/_N_WKS_DIR/_N_ARBRE1_WPD/_N_CHARIOTAGE1_MPF")
```

Exemple : Commande INIT avec indication de chemin relative

Dans le canal 1, le programme `/_N_MPF_DIR/_N_MAIN_MPF` est sélectionné

```
Code de programme          Commentaire
N10 INIT(2,"MYPROG")       ; Sélectionner le programme /_N_MPF_DIR/_N_MYPROG_MPF
                             dans le canal 2
```

Exemple : Nom et de numéro de canal avec une variable entière

```
$MC_CHAN_NAME[0]= "CHAN_X" ; nom du 1er canal
$MC_CHAN_NAME[1]= "CHAN_Y" ; nom du 2e canal
```

Code de programme	Commentaire
START(1, 2)	; Exécuter le démarrage dans le 1er et le 2e canal

De façon analogue, programmation avec les descripteurs de canaux :

Code de programme	Commentaire
START(CHAN_X, CHAN_Y)	; Exécuter le démarrage dans le 1er et le 2e canal ; Du fait du paramètre machine $MC_CHAN_NAME, les descripteurs canal_X et canal_Y représentent les numéros de canaux 1 et 2 en interne. Par conséquent, ils exécutent également un démarrage dans le 1er et le 2e canal.

Programmation avec une variable entière :

Code de programme	Commentaire
DEF INT chanNo1, chanNo2)	; Définir le numéro de canal
chanNo1=CHAN_X chanNo2=CHAN_Y	
START(chanNo1, chanNo2)	

1.14 Routine d'interruption (ASUP)

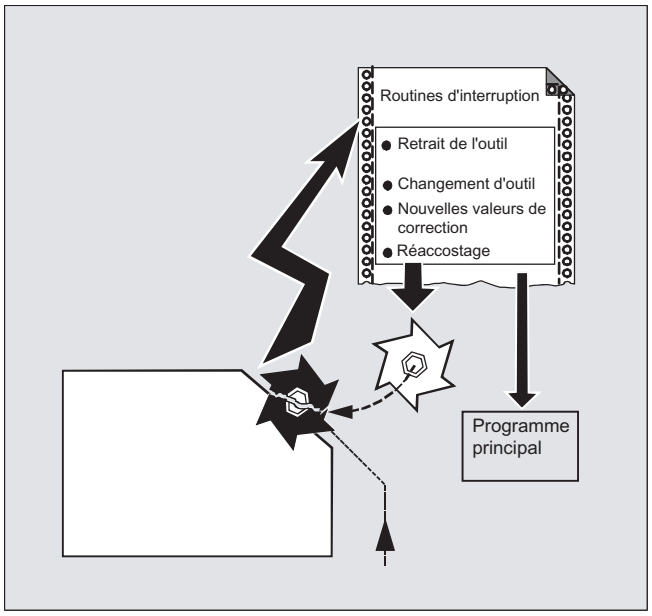
1.14.1 Fonction d'une routine d'interruption

Remarque

Les termes "sous-programme asynchrone (ASUP)" et "routine d'interruption" utilisés en alternance dans la description suivante désignent la même fonctionnalité.

Fonction

La fonction d'une routine d'interruption est explicitée à l'aide d'un exemple typique :



L'outil se brise pendant l'usinage. Ceci déclenche un signal qui arrête la phase d'usinage en cours et lance simultanément un sous-programme appelé "routine d'interruption". Ce sous-programme contient toutes les instructions qui doivent être exécutées dans ce cas.

Lorsque l'exécution du sous-programme est terminée (et la machine à nouveau prête), la commande retourne dans le programme principal et poursuit l'usinage, selon l'instruction `REPOS`, à l'endroit où il avait été interrompu (voir "Réaccostage du contour (Page 476)").

⚠ PRUDENCE

Si vous ne programmez pas d'instruction `REPOS` dans le sous-programme, il y a positionnement au point final du bloc qui suit le bloc interrompu.

Bibliographie

Description fonctionnelle Fonctions de base ; GMF, Canal, Mode de programme, Comportement au reset (K1), chapitre : "Sous-programmes asynchrones (ASUP), routines d'interruption"

1.14.2 Création d'une routine d'interruption

Créer une routine d'interruption en tant que sous-programme

La routine d'interruption est définie comme un sous-programme.

Exemple :

```
Code de programme  Commentaire
PROC RETR_Z        ; Nom de programme "RETR_Z"
N10 ...            ; Puis suivent les blocs CN.
...
N50 M17            ; Pour terminer, fin du programme et retour au programme
                     principal.
```

Sauvegarde des fonctions G modales (SAVE)

Lors de la définition, la routine d'interruption peut être accompagnée de l'attribut SAVE.

L'attribut SAVE entraîne la sauvegarde des fonctions G modales qui sont actives avant l'appel de la routine d'interruption, et leur réactivation à la fin de la routine d'interruption (voir "Sous-programmes avec mécanisme de sauvegarde (SAVE) (Page 157)").

Il est ainsi possible de reprendre l'usinage au point d'interruption après l'exécution de la routine d'interruption.

Exemple :

```
Code de programme
PROC RETR_Z SAVE
N10 ...
...
N50 M17
```

Affectation d'autres routines d'interruption (SETINT)

A l'intérieur de la routine d'interruption, il est possible de programmer des instructions SETINT (voir "Affectation et démarrage d'une routine d'interruption (SETINT) (Page 111)") pour activer d'autres routines d'interruption. Le démarrage de ces dernières n'a lieu que lorsque l'entrée correspondante commute.

Bibliographie

Pour plus d'informations sur la création de sous-programmes, consultez le chapitre "Sous-programmes, macro-instructions".

1.14.3 Affectation et démarrage d'une routine d'interruption (SETINT, PRIO, BLSYNC)

Fonction

La commande dispose de signaux (entrées 1…8) qui déclenchent l'interruption du programme courant et qui peuvent démarrer la routine d'interruption correspondante.

Dans le programme pièce, l'affectation d'une entrée au démarrage d'un programme s'effectue avec l'instruction SETINT.

Si le programme pièce contient plusieurs instructions SETINT et que, par conséquent, plusieurs signaux peuvent se présenter simultanément, les routines d'interruption correspondantes doivent avoir des valeurs de priorité définissant l'ordre d'exécution : PRIO=<valeur>

Si de nouveaux signaux arrivent pendant l'exécution d'une routine d'interruption, cette routine est interrompue par les routines prioritaires.

Syntaxe

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

Signification

SETINT(<n>) :	Instruction : affectation de l'entrée <n> à une routine d'interruption. La routine d'interruption affectée démarre lorsque l'entrée <n> commute. **Remarque :** L'affectation d'une nouvelle routine à une entrée déjà affectée désactive automatiquement l'ancienne affectation.	
<n> :	Paramètre : Numéro de l'entrée	
	Type :	INT
	Plage de valeurs :	1 ... 8
PRIO= :	Instruction : Définition de la priorité	
<valeur> :	Valeur de priorité	
	Type :	INT
	Plage de valeurs :	1 ... 128 La priorité 1 correspond à la priorité maximale.
<nom> :	Nom du sous-programme (routine d'interruption) qui doit être exécuté.	
BLSYNC :	Si l'instruction SETINT est programmée en même temps que l'instruction BLSYNC, le traitement du bloc de programme en cours continue lorsque le signal d'interruption survient, la routine d'interruption n'étant démarrée qu'ensuite.	
LIFTFAST :	Si l'instruction SETINT est programmée en même temps que LIFTFAST, un "retrait rapide de l'outil du contour" est réalisé si le signal d'interruption survient avant le début de la routine d'interruption (voir "Retrait rapide du contour (SETINT LIFTFAST, ALF) (Page 115)").	

Exemples

Exemple 1 : Affectation des routines d'interruption et définition de priorité

```
Code de programme                    Commentaire
...
N20 SETINT(3) PRIO=1 RETR_Z          ; La routine d'interruption "RETR_Z" doit
                                       démarrer lorsque l'entrée 3 commute.
N30 SETINT(2) PRIO=2 RETR_X          ; La routine d'interruption "RETR_X" doit
                                       démarrer lorsque l'entrée 2 commute.
...
```

Si les entrées commutent simultanément, les routines d'interruption sont exécutées successivement, dans l'ordre des valeurs de priorité : d'abord "RETR_Z", puis "RETR_X".

Exemple 2 : Nouvelle affectation d'une routine d'interruption

```
Code de programme                    Commentaire
...
N20 SETINT(3) PRIO=2 RETR_Z          ; La routine d'interruption "RETR_Z" doit
                                       démarrer lorsque l'entrée 3 commute.
…
N120 SETINT(3) PRIO=1 RETR_X         ; Affectation d'une nouvelle routine
                                       d'interruption à l'entrée 3 : "RETR_X" doit
                                       démarrer au lieu de "RETR_Z" lorsque
                                       l'entrée 3 commute.
```

1.14.4 Désactivation/activation de l'affectation d'une routine d'interruption (DISABLE, ENABLE)

Fonction

Une instruction SETINT se désactive avec DISABLE et se réactive avec ENABLE sans perdre l'affectation entrée → routine d'interruption.

Syntaxe

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

Signification

DISABLE(<n>) :	Instruction : **Désactivation** de l'affectation de la routine d'interruption à l'entrée <n>	
ENABLE(<n>) :	Instruction : **Réactivation** de l'affectation de la routine d'interruption à l'entrée <n>	
<n> :	Paramètre : Numéro de l'entrée	
	Type :	INT
	Plage de valeurs :	1 ... 8

Exemple

```
Code de programme                        Commentaire
...
N20 SETINT(3) PRIO=1 RETR_Z              ; La routine d'interruption "RETR_Z" doit
                                           démarrer lorsque l'entrée 3 commute.
...
N90 DISABLE(3)                           ; Désactivation de l'instruction SETINT de
                                           N20.
...
N130 ENABLE(3)                           ; Réactivation de l'instruction SETINT de N20.
...
```

1.14.5 Suppression d'une affectation de routine d'interruption (CLRINT)

Fonction

Une affectation entrée → routine d'interruption définie avec `SETINT` peut être supprimée avec `CLRINT`.

Syntaxe

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

Signification

`CLRINT(<n>)` :	Instruction : Suppression de l'affectation de la routine d'interruption à l'entrée <n>	
`<n>` :	Paramètre : Numéro de l'entrée	
	Type :	INT
	Plage de valeurs :	1 ... 8

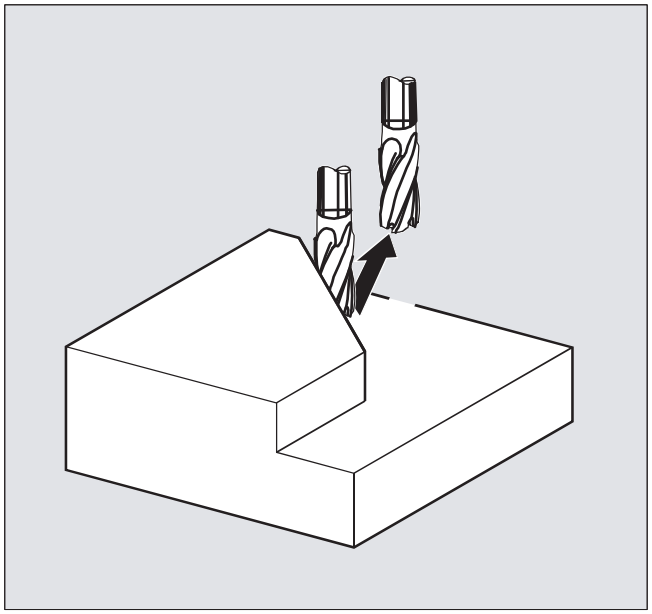
Exemple

Code de programme	**Commentaire**
`...`	
`N20 SETINT(3) PRIO=2 RETR_Z`	`;`
`...`	
`N50 CLRINT(3)`	`; L'affectation de la routine d'interruption "RETR_Z" à l'entrée "3" est supprimée.`
`...`	

1.14.6 Retrait rapide du contour (SETINT LIFTFAST, ALF)

Fonction

Dans le cas d'une instruction `SETINT` avec `LIFTFAST`, un retrait rapide éloigne l'outil du contour de pièce lorsque l'entrée commute.



La suite dépend de l'instruction `SETINT`, c'est-à-dire si celle-ci contient ou non une routine d'interruption en plus de `LIFTFAST` :

Avec une routine d'interruption :	La routine d'interruption est exécutée **après** le retrait rapide.
Sans routine d'interruption :	Après le retrait rapide, l'usinage s'arrête avec une alarme.

Syntaxe

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

Signification

`SETINT(<n>)` :	Instruction : affectation de l'entrée <n> à une routine d'interruption. La routine d'interruption affectée démarre lorsque l'entrée <n> commute.	
<n> :	Paramètre : Numéro de l'entrée	
	Type :	INT
	Plage de valeurs :	1 ... 8
`PRIO=` :	Définition de la priorité	

<valeur> :	Valeur de priorité
	Plage de valeurs : 1 ... 128
	La priorité 1 correspond à la priorité maximale.
<nom> :	Nom du sous-programme (routine d'interruption) qui doit être exécuté.
LIFTFAST :	Instruction : Retrait rapide du contour
ALF=… :	Instruction : Sens de déplacement programmable (figure dans le bloc de déplacement)
	Pour les possibilités de programmation avec ALF, voir chapitre "Sens de retrait rapide du contour (Page 117)".

Conditions marginales

Comportement en cas de frame actif avec fonction miroir

Lors de la détermination du sens de retrait, il est vérifié si un frame avec fonction miroir est actif. Dans ce cas, la gauche et la droite sont inversées pour le sens de retrait par rapport à la direction tangentielle. Les parts directionnelles dans le sens de l'outil ne seront pas prises en compte par la fonction miroir. Ce comportement est activé par le réglage du paramètre machine :

MD21202 $MC_LIFTFAST_WITH_MIRROR = TRUE

Exemple

Un outil brisé doit être remplacé automatiquement par un outil de rechange. L'usinage se poursuit alors avec le nouvel outil.

Programme principal :

Programme principal	**Commentaire**
N10 SETINT(1) PRIO=1 CHANG_O LIFTFAST	; Si l'entrée 1 commute, un retrait rapide éloigne immédiatement l'outil du contour (code n° 7 pour la correction de rayon d'outil G41). Il est suivi de l'exécution de la routine d'interruption "CHANG_O".
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	

Sous-programme :

Sous-programme	Commentaire
PROC CHANG_O SAVE	; Sous-programme avec mise en mémoire de l'état courant
N10 G0 Z100 M5	; Position de changement de l'outil, arrêt de la broche
N20 T11 M6 D1 G41	; Changement d'outil
N30 REPOSL RMB M3	; Réaccostage du contour et retour au programme principal (programmation dans un bloc)

1.14.7 Sens de retrait rapide du contour

Déplacement de retrait

Le plan du déplacement de retrait est déterminé par les codes G suivants :

- LFTXT

  Le plan du déplacement de retrait est déterminé par la tangente de la trajectoire et la direction de l'outil (préréglage).

- LFWP

  Le plan du déplacement de retrait est le plan de travail actif sélectionné avec les fonctions G17, G18 et G19 du code G. Le sens du déplacement de retrait est indépendant de la tangente à la trajectoire. Ceci permet de programmer un retrait rapide paraxial.

- LFPOS

  Retrait de l'axe introduit avec POLFMASK / POLFMLIN à la position d'axe absolue programmée avec POLF.

  ALF n'a aucune influence sur la direction du retrait pour plusieurs axes ainsi que pour plusieurs axes dans un contexte linéaire.

  Bibliographie :
  Manuel de programmation Notions de base ; chapitre : "Retrait rapide en cours de filetage à l'outil"

sens de déplacement programmable (ALF=…)

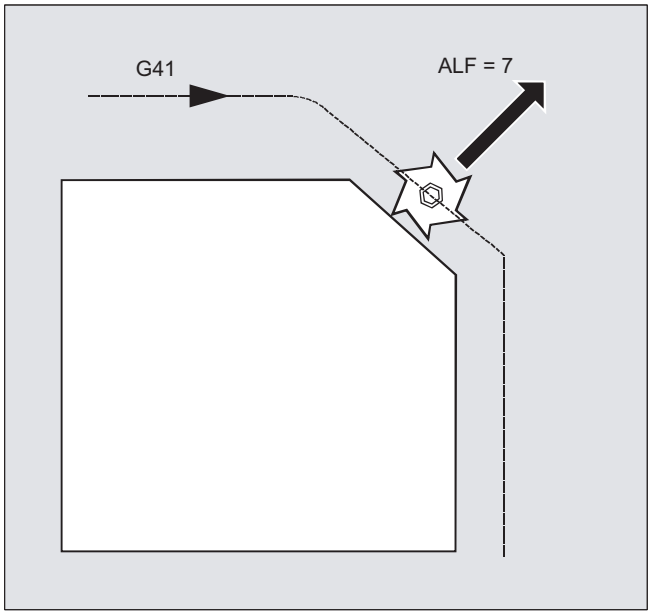
Dans le plan du mouvement de retrait, la direction est programmée avec ALF par pas discrets de 45 degrés.

Les sens de déplacement possibles sont enregistrés dans la commande sous des n° de code spéciaux et peuvent être appelés par le biais de ces numéros.

Exemple :

Code de programme
N10 SETINT(2) PRIO=1 RETR_Z LIFTFAST
ALF=7

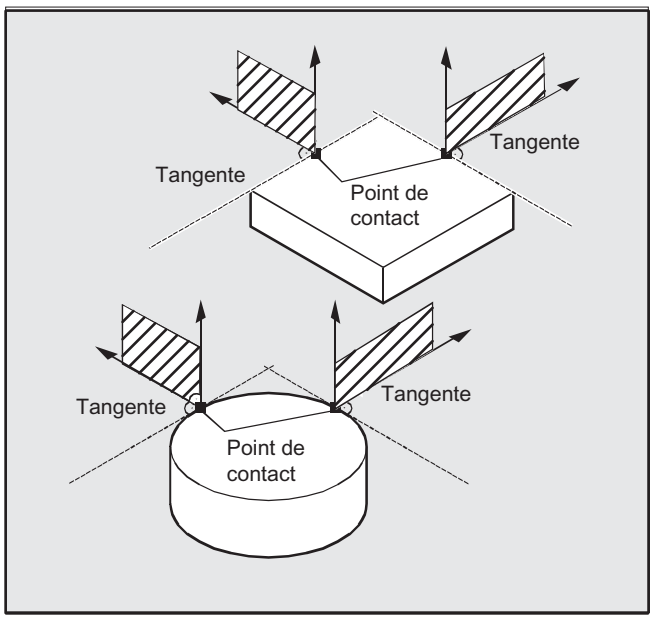
Lorsque G41 est activé (sens d'usinage à gauche du contour), le retrait de l'outil est perpendiculaire au contour.



Plan de référence pour la description du sens de déplacement avec LFTXT

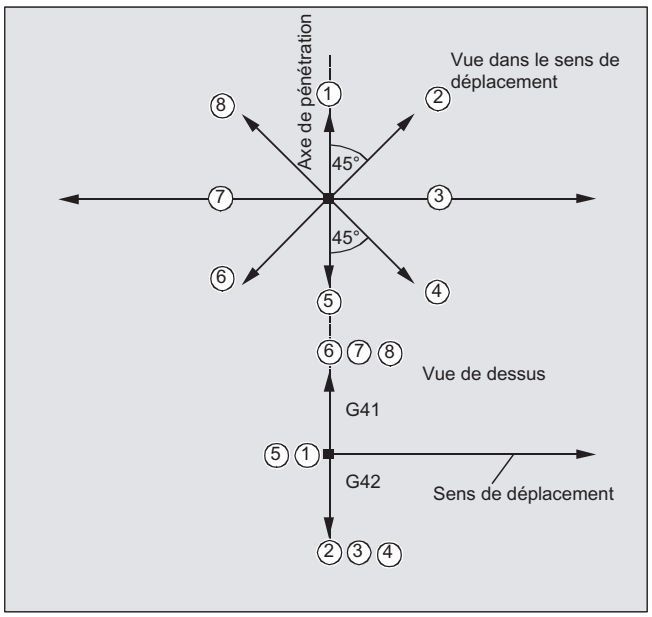
Un plan est défini au point de contact de l'outil avec le contour programmé ; ce plan sert de référence pour préciser le mouvement de retrait avec le numéro de code correspondant.

Le plan de référence est défini à partir de l'axe longitudinal de l'outil (direction de pénétration) et d'un vecteur perpendiculaire à ce dernier et à la tangente au point de contact de l'outil avec le contour.



Numéros de code pour le sens de déplacement avec LFTXT

La figure ci-dessous contient les numéros de code avec les sens de déplacement en partant du plan de référence.



Avec `ALF=1`, le retrait a lieu dans la direction de l'outil.

Avec `ALF=0`, la fonction de retrait rapide est désactivée.

⚠ PRUDENCE

Lorsque la correction de rayon d'outil est activée, il est recommandé de ne pas utiliser :

- les codes 2, 3, 4 pour `G41`
- les codes 6, 7, 8 pour `G42`

car, dans ces cas, l'outil se déplace vers le contour et entre en collision avec la pièce.

Numéros de code pour le sens de déplacement avec LFWP

Avec LFWP, la direction dans le plan de travail est tributaire des configurations suivantes :

- G17 : Plan X/Y (X/Y plane)

  ALF=1: retrait en direction X

  ALF=3 : retrait en direction Y
- G18 : Plan Z / X

  ALF=1: retrait en direction Z

  ALF=3 : retrait en direction X
- G19 : Plan Y/Z (Y/Z plane)

  ALF=1: retrait en direction Y

  ALF=3 : retrait en direction Z

1.14.8 Séquence de déplacement avec des routines d'interruption

Routine d'interruption sans LIFTFAST

Les déplacements d'axe sont freinés sur la trajectoire jusqu'à l'arrêt. Puis la routine d'interruption démarre.

La position d'arrêt est enregistrée comme position d'interruption et est accostée à la fin de la routine d'interruption dans le cas de REPOS avec RMI.

Routine d'interruption avec LIFTFAST

Les déplacements d'axe sont freinés sur la trajectoire. Simultanément, le déplacement LIFTFAST est exécuté comme déplacement superposé. Lorsque le déplacement le long de la trajectoire et le déplacement LIFTFAST se sont immobilisés, la routine d'interruption démarre.

La position où le déplacement LIFTFAST a démarré sur le contour et où l'outil a quitté la trajectoire est enregistrée comme position d'interruption.

Avec LIFTFAST et ALF=0, le comportement de la routine d'interruption est identique à celui de la routine sans LIFTFAST.

Remarque

La distance de retrait rapide des axes géométriques par rapport au contour est réglable dans un paramètre machine.

1.15 Permutation d'axe, permutation de broche (RELEASE, GET, GETD)

Fonction

Un ou plusieurs axes ou broches ne peuvent être interpolés que dans un canal. Si l'on prévoit d'utiliser un axe alternativement dans deux canaux différents (par exemple dans le cas d'un changeur de palettes), il faut d'abord le libérer dans le canal actuel, puis le prendre en charge dans l'autre canal. L'axe est permuté entre les canaux.

Extensions de permutation d'axe

Un axe/broche peut être remplacé avec un arrêt du prétraitement et une synchronisation entre le prétraitement et l'exécution des blocs ou encore sans arrêt du prétraitement. Par ailleurs, une permutation d'axe est également possible via

- une rotation conteneur d'axes AXCTSWE ou AXCTWED via un `GET/GETD` implicite.
- Frame avec rotation lorsque cet axe est combiné avec d'autres axes.
- Actions synchrones, voir Actions synchrones au déplacement, "Permutation d'axe `RELEASE`, `GET`".

Constructeur de la machine-outil

Tenez compte des indications du constructeur de la machine. Des paramètres machines configurables doivent définir de manière univoque un axe pour la permutation d'axe dans tous les canaux et des paramètres machines peuvent aussi modifier le comportement de la permutation d'axe.

Syntaxe

```
RELEASE (nom d'axe, nom d'axe, ...) ou RELEASE (S1)
GET (nom d'axe, nom d'axe, ...) ou GET (S2)
GETD (nom d'axe, nom d'axe, ...) ou GETD (S3)
```

Avec GETD (GET Directly), un axe est extrait directement d'un autre canal. Cela signifie que, pour ce GETD, il n'est pas nécessaire de programmer un RELEASE approprié dans un autre canal. Cela signifie aussi que désormais, une autre communication de canal doit être mise en place (par exemple marques wait).

Signification

`RELEASE (nom d'axe, nom d'axe, …)` :	Déblocage de/des axe(s)
`GET (nom d'axe, nom d'axe, …)` :	Prise en charge de/des axe(s)
`GETD (nom d'axe, nom d'axe, …)` :	Prise en charge directe de/des axe(s)
`nom d'axe` :	Affectation des axes dans le système : AX1, AX2, … ou indication du nom des axes de la machine
`RELEASE(S1)` :	Libération des broches S1, S2, …
`GET(S2)` :	Prise en charge des broches S1, S2, …
`GETD(S3)` :	Prise en charge directe des broches S1, S2, …

Demande GET sans arrêt du prétraitement

Lorsque, après une demande GET **sans** arrêt du prétraitement, l'axe est de nouveau débloqué avec `RELEASE(Axe)` ou `WAITP(Axe)`, le `GET` qui suit entraîne un `GET` **avec** arrêt du prétraitement.

⚠ PRUDENCE

Un axe ou une broche pris(e) en charge avec GET reste assigné à ce canal, même après un RESET de touche ou de programme.

En cas de redémarrage du programme, l'affectation des axes permutés ou des broches permutées doit être programmée si les axes ou les broches doivent être utilisés dans leurs canaux d'origine.

A la mise sous tension de la CN, les axes et les broches sont affectés aux canaux spécifiés dans les paramètres machine.

Exemples

Exemple 1 : permuter des axes entre deux canaux

Sur 6 axes sont utilisés pour le traitement dans le canal 1 : 1., 2., 3. et le 4e axe.
les axes 5 et 6 sont employés dans le canal 2 pour le changement de pièce.

L'axe 2 doit pouvoir être permuté entre les deux canaux et, lors de la mise sous tension de la CN, affecté au canal 1.

Programme "MAIN" dans le canal 1 :

```
Code de programme                Commentaire
INIT (2,"PERMUT2")               ; Sélection du programme PERMUT2 dans le canal 2.
N… START (2)                     ; Démarrage du programme dans le canal 2.
N… GET (AX2)                     ; Prise en charge de l'axe AX2.
...
N… RELEASE (AX2)                 ; Libération de l'axe AX2.
N… WAITM (1,1,2)                 ; Attendre la marque WAIT dans les canaux 1 et 2 pour
                                   la synchronisation dans les deux canaux.
...                              ; Poursuite de l'exécution après la permutation
                                   d'axe.
N… M30
```

Programme "PERMUT2" dans le canal 2 :

```
Programmation                    Commentaire
N… RELEASE (AX2)
N160 WAITM (1,1,2)               ; Attendre la marque WAIT dans les canaux 1 et 2 pour
                                   la synchronisation dans les deux canaux.
N150 GET (AX2)                   ; Prise en charge de l'axe AX2.
...                              ; Poursuite de l'exécution après la permutation
                                   d'axe.
N… M30
```

Exemple 2 : permuter des axes sans synchronisation

Lorsque l'axe ne doit pas être synchronisé, GET ne génère pas d'arrêt du prétraitement.

```
Programmation                Commentaire
N01 G0 X0
N02 RELEASE(AX5)
N03 G64 X10
N04 X20
N05 GET(AX5)                 ; Si aucune synchronisation n'est requise, ce bloc
                               sera non exécutable.
N06 G01 F5000                ; Bloc non exécutable.
N07 X20                      ; Bloc non exécutable, car position X comme dans N04.
N08 X30                      ; Premier bloc exécutable après N05.
...
```

Exemple 3 : activer une permutation d'axes sans arrêt du prétraitement des blocs

Prérequis : La permutation d'axe sans arrêt du prétraitement doit être configurée via un paramètre machine.

```
Programmation                Commentaire
N010 M4 S100
N011 G4 F2
N020 M5
N021 SPOS=0
N022 POS[B]=1
N023 WAITP[B]                ; L'axe B devient un axe banalisé.
N030 X1 F10
N031 X100 F500
N032 X200
N040 M3 S500                 ; L'axe ne déclenche pas d'arrêt du
                               prétraitement/REORG.
N041 G4 F2
N050 M5
N099 M30
```

Lorsque la broche ou l'axe B sont déplacés directement après le bloc N023 comme **axe AP** par exemple sur 180 degrés puis à nouveau sur 1 degré, alors cet axe redevient un axe banalisé et ne déclenche pas d'arrêt du prétraitement dans le bloc `N40`.

Condition préalable

Conditions nécessaires à la permutation d'axe

- L'axe doit être défini par le biais des paramètres machine dans tous les canaux qui veulent utiliser cet axe.
- Le canal auquel l'axe est affecté lors de la mise sous tension doit être défini via le paramètre machine spécifique à un **axe**.

Description

Libération de l'axe : RELEASE

Lors de la libération de l'axe, veillez aux points suivants :

1. L'axe ne doit participer à aucune transformation.
2. En cas de couplage d'axes (commande tangentielle), tous les axes du groupe doivent être libérés.
3. Il est impossible de permuter un axe de positionnement concurrent dans cet état.
4. Dans le cas d'un axe maître Gantry (axe de portique), tous les axes asservis sont aussi permutés.
5. Dans le cas d'un couplage d'axes (déplacement conjugués , couplage pilote , réducteur électronique), seul l'axe pilote du groupe peut être libéré.

Prise en charge de l'axe : GET

La permutation d'axe devient effective avec cette instruction. Le canal dans lequel l'instruction a été programmée prend entièrement en charge l'axe.

Effets de GET :

Permutation d'axe avec synchronisation :

Un axe doit toujours être synchronisé s'il a été affecté entre temps à un canal différent ou à l'AP, et qu'il n'y a pas eu synchronisation avant GET par "WAITP", G74 ou effacement de la distance restant à parcourir.

- Un arrêt du prétraitement a lieu (comme pour STOPRE).
- L'exécution du programme est interrompue jusqu'à ce que la permutation d'axe soit terminée.

"GET" automatique

Si un axe est disponible par principe dans le canal, mais rendu momentanément indisponible comme axe de canal, alors un GET est automatiquement exécuté. Si le ou les axes sont déjà synchronisés, aucun arrêt du prétraitement n'a lieu.

Paramétrer de façon modifiable le comportement de permutation d'axe

Un paramètre machine détermine le moment des axes de la façon suivante :

- La permutation d'axe entre deux canaux s'effectue automatiquement même si l'axe a été banalisé avec WAITP (comportement tel que jusqu'à présent)
- Lors de la demande d'une rotation conteneur d'axes, tous les axes assignables au canal à traiter du conteneur d'axes sont prélevés dans le canal à l'aide de GET ou de GETD implicites. Une permutation d'axe n'est ensuite à nouveau autorisée qu'une fois la rotation conteneur d'axe terminée.
- Lorsqu'un bloc intermédiaire a été intercalé, il est vérifié à l'exécution du bloc si une réorganisation est nécessaire ou non. Une réorganisation est nécessaire uniquement si les états de l'axe figurant dans ce bloc ne correspondent **pas** aux états actuels de l'axe.
- A la place d'un bloc GET avec arrêt du prétraitement et synchronisation entre le prétraitement et l'exécution des blocs, une permutation d'axe peut aussi avoir lieu sans arrêt du prétraitement. Il n'est alors généré qu'un seul bloc intermédiaire avec la demande GET. Pendant l'exécution de ce bloc durant l'exécution du programme, ce dernier vérifie si les états de l'axe dans le bloc correspondent bien aux états d'axe actuels.

Pour de plus amples informations sur la fonction d'une permutation d'axe ou de broche, voir /FB2/ Manuel de fonctions d'extension; GMF, canaux, permutation d'axe (K5).

1.16 Transmettre un axe à un autre canal (AXTOCHAN)

Fonction

Avec la commande CN `AXTOCHAN`, un axe peut être demandé pour transmettre cet axe à un autre canal. L'axe peut aussi bien être amené au canal correspondant par le programme pièce CN que par une action synchrone.

Syntaxe

```
AXTOCHAN(nom d'axe,numéro de canal[,nom d'axe,numéro de
canal[,...]])
```

Signification

`AXTOCHAN` :	Demande d'axe pour un canal donné
`nom d'axe` :	Affectation des axes dans le système : X, Y, … ou indication des noms d'axe machine concernés. Le canal à exécuter ne doit pas être son canal propre et ce ne doit pas être non plus le canal qui possède actuellement le droit d'interpolation pour l'axe
`numéro de canal` :	Numéro du canal devant être assigné à l'axe

Remarque

Axe de positionnement concurrent et axe exclusivement contrôlé par l'AP

Un axe AP ne peut pas changer de canal en tant qu'axe de positionnement concurrent. Un axe exclusivement contrôlé par l'AP ne peut pas être assigné au programme CN.

Bibliographie
Description fonctionnelle Fonctions d'extension ; axes de positionnement (P2)

Exemple

AXTOCHAN dans le programme CN

Les axes X et Y sont connus dans le 1er et dans le 2ème canal. Actuellement, c'est le canal 1 qui détient le droit d'interpolation et le programme suivant démarre dans le canal 1:

```
Code de programme                Commentaire
N110 AXTOCHAN(Y,2)               ; Translater l'axe Y dans le 2ème canal.
N111 M0
N120 AXTOCHAN(Y,1)               ; Aller reprendre l'axe Y (banalisé).
N121 M0
N130 AXTOCHAN(Y,2,X,2)           ; Translater l'axe Y et l'axe X dans le 2ème canal
                                   (axes banalisés).
N131 M0
N140 AXTOCHAN(Y,2)               ; Translater l'axe Y dans le 2ème canal (programme
                                   CN).
N141 M0
```

Informations complémentaires

AXTOCHAN dans le programme CN

Un GET est alors exécuté uniquement en cas de demande de l'axe pour le programme CN dans un canal séparé et la modification d'état réelle est attendue. Lorsque l'axe est demandé pour un autre canal ou s'il doit devenir un axe banalisé dans un canal séparé, alors seulement la demande est générée.

AXTOCHAN à partir d'une action synchrone

Lorsqu'un axe est demandé pour un canal séparé alors AXTOCHAN est représenté sur un GET à partir d'une action synchrone. Dans ce cas, l'axe devient axe banalisé à la première demande pour le canal propre. À la deuxième demande, l'axe est affecté au programme CN d'une manière analogue à la demande GET dans le programme CN. En ce qui concerne la demande GET à partir d'une action synchrone, se reporter au chapitre "Actions synchrones au déplacement".

1.17 Activation des paramètres machine (NEWCONF)

Fonction

L'instruction NEWCONF permet d'activer tous les paramètres machine du niveau de prise d'effet "NEW_CONFIG". La fonction peut également être activée dans l'interface utilisateur IHM en actionnant la touche logicielle "Activer PM".

Lors de l'exécution de la fonction "NEWCONF", un arrêt implicite du prétraitement des blocs a lieu, c'est-à-dire que le déplacement avec interpolation est interrompu.

Syntaxe

```
NEWCONF
```

Signification

NEWCONF : Instruction de prise d'effet de l'ensemble des paramètres machine du niveau de prise d'effet "NEW_CONFIG"

Exécution de NEWCONF pour tous les canaux depuis le programme pièce

Si des paramètres machine axiaux du programme pièce sont modifiés puis activés avec NEWCONF, alors l'instruction NEWCONF active uniquement les paramètres machine entraînant des modifications du canal du programme pièce.

Remarque

Afin d'activer toutes les modifications de façon sûre, l'instruction NEWCONF doit être exécutée dans chaque canal dans lequel doivent être calculés les axes ou fonctions concernés par les modifications des paramètres machine.

Avec NEWCONF, aucun paramètre machine axial n'est activé.

Un RESET axial doit être exécuté pour les axes contrôlés par l'AP.

Exemple

Réseau de trous : réaliser des trous taraudés en différentes positions

```
Code de programme                          Commentaire
N10 $MA_CONTOUR_TOL[AX]=1.0                ; Modifier un paramètre machine.
N20 NEWCONF                                ; Activer des paramètres machine.
...
```

1.18 Ecriture d'un fichier (WRITE)

Fonction

L'instruction `WRITE` permet d'écrire des blocs/données d'un programme pièce à la fin d'un fichier (fichier de journal) spécifié ou à la fin du programme pièce en cours d'exécution. Les blocs/données sont ajoutés à la fin du fichier, c'est-à-dire après `M30`.

Remarque

Si le fichier à décrire avec l'instruction `WRITE` n'existe pas dans la CN, il est créé.

Son lieu de stockage est la mémoire statique de la CN. Pour SINUMERIK 840D sl, il s'agit de la carte CompactFlash Card. Par rapport à SINUMERIK 840D, le temps d'exécution de l'instruction `WRITE` augmente alors d'environ 75 ms.

S'il existe un fichier de même nom sur le disque dur, ce fichier est écrasé après sa fermeture (dans la CN). Aide : Modifier le nom dans la CN en utilisant la touche logicielle "Propriétés" du groupe fonctionnel "Services".

Condition

Le niveau de protection réglé doit être supérieur ou égal au droit WRITE du fichier. Si ce n'est pas le cas, l'accès est refusé avec émission du message d'erreur (valeur en retour de la variable d'erreur = 13).

Syntaxe

```
DEF INT <erreur>
WRITE(<erreur>,"<nom de fichier>","<bloc/données>")
```

Signification

`WRITE` :	Instruction d'insertion d'un bloc ou de données à la fin du fichier spécifié		
`<erreur>` :	Variable pour la valeur d'erreur en retour		
	type.	INT	
	Valeur :	0	Pas d'erreur
		1	chemin d'accès non autorisé
		2	chemin d'accès introuvable
		3	fichier introuvable
		4	type de fichier erroné
		10	fichier plein
		11	fichier en cours d'utilisation
		12	pas de ressources disponibles
		13	pas de droits d'accès
		20	autre erreur

`<nom de fichier>` : Nom du fichier dans lequel le bloc ou les données indiqués doivent être ajoutés

Type : STRING

Lors de la spécification du nom de fichier, les points suivants doivent être respectés :

- Le nom de fichier spécifié ne doit pas contenir de caractères d'espacement ou de caractères de commande (caractères avec un code ASCII ≤ 32), car sinon l'instruction `WRITE` est interrompue avec le code d'erreur 1 "chemin d'accès non autorisé".
- Le nom du fichier peut être spécifié avec l'indication du chemin d'accès et l'extension de fichier :
  - Chemins d'accès

    L'indication du chemin doit être absolue, c'est-à-dire commencer par "/".

    En l'absence d'indication du chemin, le fichier enregistré dans le répertoire actuel (=répertoire du programme sélectionné).
  - Extension de fichier

    Si le nom du fichier ne comporte pas de nom de domaine (_N_), celui-ci est ajouté automatiquement.

    Si, dans le nom du fichier, le quatrième caractère avant la fin est un trait de soulignement "_", les trois caractères suivants sont interprétés comme extension du fichier. Afin de pouvoir utiliser le même nom de fichier pour toutes les instructions de fichier, p. ex. au moyen d'une variable de type STRING, seules les extensions de fichier _SPF et _MPF doivent être utilisées.

    Si aucune extension "_MPF" ou "_SPF" n'est spécifiée, _MPF est ajouté automatiquement.
- La longueur du nom du fichier ne doit pas dépasser 32 octets, la longueur du nom du chemin 128 octets.

Exemple :

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

`<bloc/données>` : Bloc ou données à ajouter dans le fichier spécifié.

Type : STRING

Nota :
LF est ajouté en interne, c'est-à-dire que la chaîne de texte est rallongée d'1 caractère.

Conditions marginales

- **Taille maximale du fichier (→ constructeur de la machine !)**

  La taille maximale possible des fichiers de journal est réglée avec le paramètre machine :

  MD11420 $MN_LEN_PROTOCOL_FILE

  La taille maximale s'applique à tous les fichiers créés avec l'instruction `WRITE`. En cas de dépassement, un message d'erreur est émis et l'enregistrement du bloc ou des données ne s'effectue pas. Si la capacité mémoire est suffisante, un nouveau fichier peut être créé.

Exemples

Exemple 1 : instruction WRITE sans indication absolue du chemin

```
N10 DEF INT ERROR                                          ; Définition de la
                                                             variable d'erreur.
N20 WRITE(ERROR,"TEST1","JOURNAL DU 7.2.97")               ; Ecrire le texte "JOURNAL
                                                             DU 7.2.97" dans le
                                                             fichier _N_TEST1_MPF.
N30 IF ERROR                                               ; Exploitation des
                                                             erreurs.
N40 MSG ("Erreur lors instruction WRITE :" <<ERROR)
N50 M0
N60 ENDIF
...
```

Exemple 2 : instruction WRITE avec indication absolue du chemin

```
...
WRITE(ERROR,"/_N_WKS_DIR/_N_PROT_WPD/_N_PROT_MPF","JOURNAL DU 7.2.97")
...
```

1.19 Suppression d'un fichier (DELETE)

Fonction

L'instruction DELETE permet de supprimer tous les fichiers, qu'ils aient été créés avec l'instruction WRITE ou non. DELETE permet également de supprimer les fichiers ayant été créés avec des niveaux d'accès supérieurs.

Syntaxe

```
DEF INT <erreur>
DELETE(<erreur>,"<nom de fichier>")
```

Signification

DELETE :	Instruction de suppression du fichier spécifié		
<erreur> :	Variable pour la valeur d'erreur en retour		
	type.	INT	
	Valeur :	0	Pas d'erreur
		1	chemin d'accès non autorisé
		2	chemin d'accès introuvable
		3	fichier introuvable
		4	type de fichier erroné
		11	fichier en cours d'utilisation
		12	pas de ressources disponibles
		20	autre erreur

`<nom de fichier>` : Nom du fichier à supprimer

Type : STRING

Lors de la spécification du nom de fichier, les points suivants doivent être respectés :

- Le nom de fichier spécifié ne doit contenir ni caractères d'espacement, ni caractères de commande (caractères avec un code ASCII ≤ 32), car sinon l'instruction `DELETE` est interrompue avec le code d'erreur 1 "chemin d'accès non autorisé".
- Le nom du fichier peut être spécifié avec l'indication du chemin d'accès et l'extension de fichier :
  - Chemins d'accès

    L'indication du chemin doit être absolue, c'est-à-dire commencer par "/".

    En l'absence d'indication du chemin, le fichier est recherché dans le répertoire actuel (= répertoire du programme sélectionné).
  - Extension de fichier

    Si le nom du fichier ne comporte pas de nom de domaine (_N_), celui-ci est ajouté automatiquement.

    Si, dans le nom du fichier, le quatrième caractère avant la fin est un trait de soulignement "_", les trois caractères suivants sont interprétés comme extension du fichier. Afin de pouvoir utiliser le même nom de fichier pour toutes les instructions de fichier, p. ex. au moyen d'une variable de type STRING, seules les extensions de fichier _SPF et _MPF doivent être utilisées.

    Si aucune extension "_MPF" ou "_SPF" n'est spécifiée, _MPF est ajouté automatiquement.
- La longueur du nom du fichier ne doit pas dépasser 32 octets, la longueur du nom du chemin 128 octets.

Exemple :

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

Exemple

Code de programme	Commentaire
N10 DEF INT ERROR	; Définition de la variable d'erreur.
N15 STOPRE	; Arrêt du prétraitement des blocs.
N20 DELETE(ERROR,"/_N_SPF_DIR/_N_TEST1_SPF")	; Supprimer le fichier TEST1 dans le répertoire du sous-programme.
N30 IF ERROR	; Exploitation des erreurs.
N40 MSG ("Erreur lors instruction DELETE : " <<ERROR)	
N50 M0	
N60 ENDIF	

1.20 Lecture des lignes d'un fichier (READ)

Fonction

L'instruction `READ` permet de lire une ou plusieurs lignes dans le fichier spécifié et d'enregistrer les informations lues dans un tableau de type STRING. Chaque ligne lue occupe un élément de ce tableau.

Remarque

Ce fichier doit se trouver dans la mémoire utilisateur statique du NCK (système passif de fichiers).

Condition

Le niveau de protection réglé doit être supérieur ou égal au droit READ du fichier. Si ce n'est pas le cas, l'accès est refusé avec émission du message d'erreur (valeur en retour de la variable d'erreur = 13).

Syntaxe

```
DEF INT <erreur>
DEF STRING[<longueur de chaîne>] <résultat>[<n>,<m>]
READ(<erreur>,"<nom de fichier>",<ligne de début>,<nombre de
lignes>,<résultat>)
```

Signification

`READ` : Instruction de lecture des lignes du fichier spécifié et d'enregistrement de ces lignes dans un tableau de variables.

`<erreur>` : Variable pour la valeur d'erreur en retour (paramètre Call-By-Reference)

type. INT

Valeur :

0	Pas d'erreur
1	chemin d'accès non autorisé
2	chemin d'accès introuvable
3	fichier introuvable
4	type de fichier erroné
13	droits d'accès insuffisants
21	Ligne inexistante (paramètres `<ligne de début>` ou `<nombre de lignes>` supérieurs au nombre de lignes du fichier spécifié).
22	Longueur de tableau de la variable de résultat (`<résultat>`) trop petite.
23	Zone de lignes trop grande (paramètre `<nombre de lignes>` trop grand, si bien que la lecture se poursuit au-delà de la fin du fichier).

`<nom de fichier>` : Nom du fichier à lire (paramètre Call-By-Value)

Type : STRING

Lors de la spécification du nom de fichier, les points suivants doivent être respectés :

- Le nom de fichier spécifié ne doit contenir ni caractères d'espacement, ni caractères de commande (caractères avec un code ASCII ≤ 32), car sinon l'instruction `READ` est interrompue avec le code d'erreur 1 "chemin d'accès non autorisé".
- Le nom du fichier peut être spécifié avec l'indication du chemin d'accès et l'extension de fichier :
  - Chemins d'accès

    L'indication du chemin doit être absolue, c'est-à-dire commencer par "/".

    En l'absence d'indication du chemin, le fichier est recherché dans le répertoire actuel (= répertoire du programme sélectionné).
  - Extension de fichier

    Si le nom du fichier ne comporte pas de nom de domaine ("_N_"), celui-ci est ajouté automatiquement.

    Si, dans le nom du fichier, le quatrième caractère avant la fin est un trait de soulignement "_", les trois caractères suivants sont interprétés comme extension du fichier. Afin de pouvoir utiliser le même nom de fichier pour toutes les instructions de fichier, p. ex. au moyen d'une variable de type STRING, seules les extensions de fichier _SPF et _MPF doivent être utilisées.

    Si aucune extension "_MPF" ou "_SPF" n'est spécifiée, _MPF est ajouté automatiquement.
- La longueur du nom du fichier ne doit pas dépasser 32 octets, la longueur du nom du chemin 128 octets.

Exemple :

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

`<ligne de début>` : Ligne de début de la zone de fichier à lire (paramètre Call-By-Value)

Type : INT

Valeur :	0	Le nombre de lignes, avant la fin du fichier, indiqué par le paramètre `<nombre de lignes>` est lu.
	1 … n	Numéro de la première ligne à lire.

`<nombre de lignes>` : Nombre de lignes à lire (paramètre Call-By-Value)

Type : INT

`<résultat>` :

Variable de résultat (paramètre Call-By-Reference)

Tableau de variables dans lequel le texte lu est enregistré.

Type : STRING (longueur max. : 255)

Si le nombre de lignes indiqué dans le paramètre `<nombre de lignes>` est inférieur à la dimension du tableau `[<n>,<m>]` de la variable de résultat, les autres éléments du tableau ne sont pas modifiés.

Les caractères de commande de fin de ligne "LF" (Line Feed) ou "CR LF" (Carrige Return Line Feed) ne sont **pas** enregistrés dans la variable de résultat.

Si la longueur de la ligne est supérieure à la longueur de chaîne définie, les lignes lues sont tronquées. Il n'y a pas de signalisation d'erreur.

Remarque

La lecture dans un fichier binaire est impossible. L'erreur "Type de fichier erroné" (valeur en retour de la variable d'erreur = 4) est signalée. Les types de fichiers suivants ne sont pas lisibles : _BIN, _EXE, _OBJ, _LIB, _BOT, _TRC, _ACC, _CYC, _NCK.

Exemple

Code de programme	Commentaire
N10 DEF INT ERROR	; Définition de la variable d'erreur.
N20 DEF STRING[255] RESULT[5]	; Définition de la variable de résultat.
N30 READ(ERROR,"/_N_CST_DIR/_N_TESTFILE_MPF",1,5,RESULT)	; Nom du fichier avec identificateur de domaine, extension de fichier et chemin.
N40 IF ERROR <>0	; Exploitation des erreurs.
N50 MSG("ERREUR"<<ERROR<<"POUR INSTRUCTION READ")	
N60 M0	
N70 ENDIF	
...	

1.21 Vérification de l'existence d'un fichier (ISFILE)

Fonction

L'instruction `ISFILE` permet de vérifier si un fichier existe dans la mémoire utilisateur statique du NCK (système passif de fichiers).

Syntaxe

```
<résultat>=ISFILE("<nom de fichier>")
```

Signification

`ISFILE` : Instruction permettant de vérifier que le fichier spécifié existe dans la système passif de fichiers.

`<nom de fichier>` : Nom du fichier dont l'existence doit être vérifiée dans le système passif de fichiers.

Type : STRING

Lors de la spécification du nom de fichier, les points suivants doivent être respectés :

- Le nom de fichier spécifié ne doit contenir ni caractères d'espacement, ni caractères de commande (caractères avec un code ASCII ≤ 32).
- Le nom du fichier peut être spécifié avec l'indication du chemin d'accès et l'extension de fichier :
  - Chemins d'accès

    L'indication du chemin doit être absolue, c'est-à-dire commencer par "/".

    En l'absence d'indication du chemin, le fichier est recherché dans le répertoire actuel (= répertoire du programme sélectionné).
  - Extension de fichier

    Si le nom du fichier ne comporte pas de nom de domaine (_N_), celui-ci est ajouté automatiquement.

    Si, dans le nom du fichier, le quatrième caractère avant la fin est un trait de soulignement "_", les trois caractères suivants sont interprétés comme extension du fichier. Afin de pouvoir utiliser le même nom de fichier pour toutes les instructions de fichier, p. ex. au moyen d'une variable de type STRING, seules les extensions de fichier _SPF et _MPF doivent être utilisées.

    Si aucune extension "_MPF" ou "_SPF" n'est spécifiée, _MPF est ajouté automatiquement.
- La longueur du nom du fichier ne doit pas dépasser 32 octets, la longueur du nom du chemin 128 octets.

Exemple :

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

<résultat> : Variable de résultat pour l'acquisition du résultat de la vérification

type. BOOL

Valeur : TRUE Fichier existant

FALSE Fichier inexistant

Exemple

Code de programme	Commentaire
N10 DEF BOOL RESULT	; Définition de la variable de résultat.
N20 RESULT=ISFILE("TESTFILE")	
N30 IF(RESULT==FALSE)	
N40 MSG("FICHIER INEXISTANT")	
N50 M0	
N60 ENDIF	
...	

ou :

Code de programme	Commentaire
N10 DEF BOOL RESULT	; Définition de la variable de résultat.
N20 RESULT=ISFILE("TESTFILE")	
N30 IF(NOT ISFILE("TESTFILE"))	
N40 MSG("FICHIER INEXISTANT")	
N50 M0	
N60 ENDIF	
...	

1.22 Lecture des informations fichier (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO)

Fonction

Les instructions FILEDATE, FILETIME, FILESIZE, FILESTAT et FILEINFO permettent de lire des informations fichier telles que date / heure du dernier accès en écriture, taille actuelle du fichier, état du fichier ou l'ensemble de ces informations.

Remarque

Ce fichier doit se trouver dans la mémoire utilisateur statique du NCK (système passif de fichiers).

Condition

Le niveau de protection actuellement paramétré doit être égal ou supérieur au droit show du répertoire principal. Si ce n'est pas le cas, l'accès est refusé avec émission du message d'erreur (valeur en retour de la variable d'erreur = 13).

Syntaxe

```
DEF INT <erreur>
DEF STRING[<longueur de chaîne>] <résultat>
FILE....(<erreur>,"<nom de fichier>",<résultat>)
```

Signification

FILEDATE :	L'instruction FILEDATE fournit la **date** du dernier accès en écriture au fichier spécifié.
FILETIME :	L'instruction FILETIME fournit l'**heure** du dernier accès en écriture au fichier spécifié.
FILESIZE :	L'instruction FILESIZE fournit la **taille actuelle** du fichier spécifié.
FILESTAT :	L'instruction FILESTAT fournit l'**état** des droits de lecture, d'écriture et d'exécution pour le fichier spécifié.
FILEINFO :	L'instruction FILEINFO fournit l'**ensemble des informations fichier** mises à disposition par FILEDATE, FILETIME, FILESIZE et FILESTAT pour le fichier spécifié.

`<erreur>` : Variable pour la valeur d'erreur en retour (paramètre Call-By-Reference)

type. INT

Valeur :

0	Pas d'erreur
1	chemin d'accès non autorisé
2	chemin d'accès introuvable
3	fichier introuvable
4	type de fichier erroné
13	droits d'accès insuffisants
22	Longueur de chaîne de la variable de résultat (`<résultat>`) trop petite.

`<nom de fichier>` : Nom du fichier à partir duquel les informations du fichier doivent être lues.

Type : STRING

Lors de la spécification du nom de fichier, les points suivants doivent être respectés :

- Le nom de fichier spécifié ne doit contenir ni caractères d'espacement, ni caractères de commande (caractères avec un code ASCII ≤ 32), car sinon l'instruction `FILE...` est interrompue avec le code d'erreur 1 "chemin d'accès non autorisé".
- Le nom du fichier peut être spécifié avec l'indication du chemin d'accès et l'extension de fichier :
  - Chemins d'accès

    L'indication du chemin doit être absolue, c'est-à-dire commencer par "/".

    En l'absence d'indication du chemin, le fichier est recherché dans le répertoire actuel (= répertoire du programme sélectionné).
  - Extension de fichier

    Si le nom du fichier ne comporte pas de nom de domaine (_N_), celui-ci est ajouté automatiquement.

    Si, dans le nom du fichier, le quatrième caractère avant la fin est un trait de soulignement "_", les trois caractères suivants sont interprétés comme extension du fichier. Afin de pouvoir utiliser le même nom de fichier pour toutes les instructions de fichier, p. ex. au moyen d'une variable de type STRING, seules les extensions de fichier _SPF et _MPF doivent être utilisées.

    Si aucune extension "_MPF" ou "_SPF" n'est spécifiée, _MPF est ajouté automatiquement.
- La longueur du nom du fichier ne doit pas dépasser 32 octets, la longueur du nom du chemin 128 octets.

Exemple :

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

`<résultat>` : Variable de résultat (paramètre Call-By-Reference)

Variable dans laquelle l'information de fichier interrogée est enregistrée.

Type : STRING pour : `FILEDATE`
Format : "jj.mm.aa"
⇒ la longueur de chaîne doit être égale à **8**.

`FILETIME`
Format : " hh:mm.ss "
⇒ la longueur de chaîne doit être égale à **8**.

`FILESTAT`
Format : "rwxsd"
(r: **r**ead, w: **w**rite, x: e**x**ecute, s: **s**how, d: **d**elete)
⇒ la longueur de chaîne doit être égale à **5**.

`FILEINFO`
Format : "rwxsd nnnnnnnn jj.mm.aa hh:mm:ss"
⇒ la longueur de chaîne doit être égale à **32**.

INT pour : `FILESIZE`
La taille du fichier est indiquée en octets.

Exemple

```
N10 DEF INT ERROR                                          ; Définition de la variable
                                                             d'erreur.
N20 STRING[32] RESULT                                      ; Définition de la variable de
                                                             résultat.
N30 FILEINFO(ERROR,"/_N_MPF_DIR/_N_TESTFILE_MPF",RESULT)   ; Nom du fichier avec
                                                             identificateur de domaine,
                                                             extension de fichier et chemin.
N40 IF ERROR <>0                                           ; Exploitation des erreurs
N50 MSG("ERREUR"<<ERROR<<"POUR INSTRUCTION FILEINFO")
N60 M0
N70 ENDIF
...
```

L'exemple pourrait p. ex. fournir le résultat suivant dans la variable de résultat RESULT :

"77777 12345678 26.05.00 13:51:30"

1.23 Calcul du total de contrôle d'un tableau (CHECKSUM)

Fonction

L'instruction CHECKSUM permet de calculer le total de contrôle d'un tableau. La comparaison entre ce total de contrôle et le résultat d'un calcul précédent du total de contrôle permet de déterminer une éventuelle modification de données du tableau.

Application

vérification si le contour d'entrée a été modifié lors du chariotage.

Syntaxe

```
DEF INT <erreur>
DEF STRING[<longueur de chaîne>] <total de contrôle>
DEF ... <tableau>[<n>,<m>,<o>]
<erreur>=CHECKSUM(<total de contrôle>,"<tableau>"[,<colonne de
début>,<colonne de fin>])
```

Signification

CHECKSUM : Instruction de calcul du total de contrôle d'un tableau

<erreur> : Variable pour la valeur d'erreur en retour

type. INT

Valeur :

0	Pas d'erreur
1	symbole introuvable
2	pas de tableau
3	l'index 1 est trop grand
4	l'index 2 est trop grand
5	type de données non valide
10	dépassement de la somme de contrôle

<total de contrôle> : Variable pour le résultat du calcul du total de contrôle (paramètre Call-By-Reference)

Type : STRING

Longueur de chaîne requise : 16

La somme de contrôle est représentée sous forme de chaîne de 16 chiffres hexadécimaux. Aucun caractère de formatage n'est indiqué.

Exemple : "A6FC3404E534047C"

<tableau> :	Nom du tableau dont la somme de contrôle doit être formée (paramètre (Call-By-Value) Type : STRING Longueur de chaîne max. : 32 Les tableaux autorisés sont des tableaux de 1 à 3 dimensions des types de données : BOOL, CHAR, INT, REAL, STRING **Nota :** Les tableaux de paramètres machine ne sont pas admis.
<colonne de début> :	Numéro de la colonne de début du tableau pour le calcul du total de contrôle (paramètre facultatif)
<colonne de fin> :	Numéro de la colonne de fin du tableau pour le calcul du total de contrôle (paramètre facultatif)

Remarque

Les paramètres <colonne de début> et <colonne de fin>sont facultatifs. Lorsque aucun indice de colonne n'est donné, la somme de contrôle est formée sur tout le tableau.

Le résultat d'une somme de contrôle est toujours univoque. En cas de modification d'un élément du tableau, la chaîne de caractères résultat change également.

Exemple

```
Code de programme                                    Commentaire
N10 DEF INT ERROR                                    ; Définition de la variable
                                                       d'erreur.
N20 DEF STRING[16] MY_CHECKSUM                       ; Définition de la variable de
                                                       résultat.
N30 DEF INT MY_VAR[4,4]                              ; Définition de tableau.
N40 MY_VAR=- ...
N50 ERROR=CHECKSUM(MY_CHECKSUM,"MY_VAR",0,2)
...
```

L'exemple pourrait p. ex. fournir le résultat suivant dans la variable de résultat MY_CHECKSUM :

"A6FC3404E534047C"

1.24 Arrondissement (ROUNDUP)

Fonction

La fonction "ROUNDUP" arrondit les valeurs d'entrée de type REAL (nombres rationnels avec point décimal) à l'entier supérieur le plus proche.

Syntaxe

```
ROUNDUP(<valeur>)
```

Signification

`ROUNDUP` :	Instruction d'arrondissement d'une valeur d'entrée
`<valeur>` :	Valeur d'entrée du type REAL

Remarque

Les valeurs d'entrée de type INT (nombres entiers) restent inchangées.

Exemples

Exemple 1 : Arrondis de différentes valeurs d'entrée

Exemple	Arrondi
`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

Exemple 2 : ROUNDUP dans le programme CN

Code de programme

```
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 Technique des sous-programmes

1.25.1 Généralités

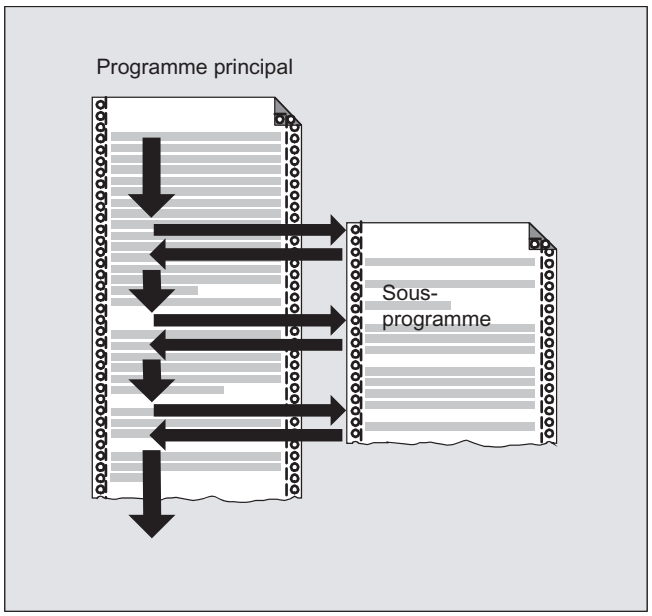
1.25.1.1 Sous-programme

Fonction

La désignation "Sous-programme" date encore de l'époque où les programmes pièce étaient répartis de façon déterminée en programmes principaux et sous-programmes. Les programmes principaux correspondaient aux programmes pièce qui, en vue de leur exécution, étaient sélectionnés, puis démarrés sur la commande. Les sous-programmes correspondaient aux programmes pièce qui étaient appelés par le programme principal.

Cette répartition déterminée n'a plus lieu d'être avec la langage CN SINUMERIK actuel. Chaque sous-programme peut par principe être sélectionné et démarré comme programme principal ou être appelé comme sous-programme par un autre programme pièce.

Ainsi, nous désignerons par la suite par sous-programme, un programme pièce appelé par un autre programme pièce.



Application

Comme dans tous les langages de programmation évolués, les sous-programmes dans le langage CN servent également à transférer des sections de programme, qui sont exécutées plusieurs fois, en tant que programmes complets, autonomes.

Les sous-programmes offrent les avantages suivants :

- Augmentation de la clarté et de la lisibilité des programmes
- Augmentation de la qualité grâce à la réutilisation de sections de programme testées
- Possibilité de créer des bibliothèques d'usinage spécifiques
- Economie de capacité mémoire

1.25.1.2 Nom de sous-programme

Règles à respecter pour les noms

Lors de l'attribution d'un nom de sous-programme, les règles suivantes doivent être respectées :

- Les deux premiers caractères doivent être des lettres (A - Z, a - z).
- Quant aux caractères qui suivent, il peut s'agit d'une combinaison quelconque de lettres, chiffres (0 - 9) et trait de soulignement ("_").
- Il est possible d'utiliser au maximum 31 caractères.

Remarque

Dans le langage CN SINUMERIK, **aucune** distinction n'est faite entre majuscules et minuscules.

Extensions du nom de programme

De façon interne à la commande, un préfixe et un postfixe sont ajoutés comme extension au nom de programme attribué lors la création du programme :

- Préfixe : _N_
- Postfixe :
  - Programmes principaux : _MPF
  - Sous-programmes : _SPF

Utilisation du nom de programme

Pour l'utilisation du nom de programme, p. ex. lors d'un appel de sous-programme, toutes les combinaisons entre préfixe, nom de programme et postfixe sont possibles.

Exemple :

Le sous-programme avec le nom "SUB_PROG" peut être démarré par les appels suivants :

1. `SUB_PROG`
2. `_N_SUB_PROG`
3. `SUB_PROG_SPF`
4. `_N_SUB_PROG_SPF`

Remarque

Noms identiques pour programmes principaux et sous-programmes

Si des programmes principaux (.MPF) et des sous-programmes (.SPF) portent le même nom de programme, il faut indiquer le postfixe respectif lors de l'utilisation du nom de programme dans le programme pièce, afin d'identifier de manière univoque le programme.

1.25.1.3 Imbrication de sous-programmes

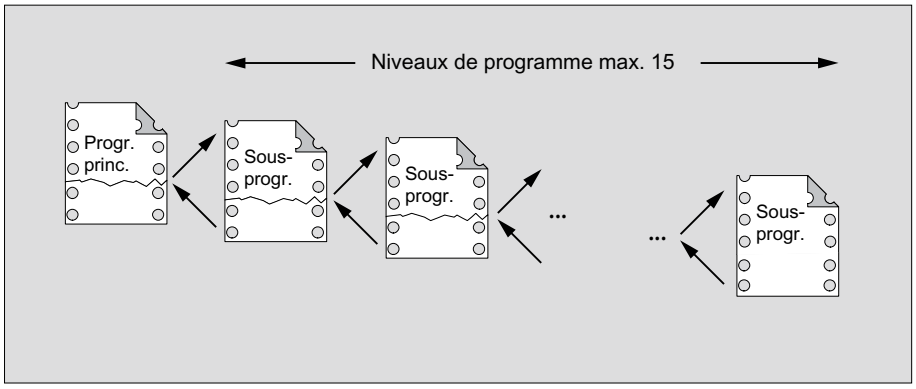
Un programme principal peut appeler des sous-programmes qui, à leur tour, appellent des sous-programmes. L'exécution des programmes est ainsi imbriquée. Chaque programme s'exécute dans un niveau de programme spécifique.

Niveaux d'imbrication

Le langage CN met actuellement à disposition 16 niveaux de programme. Le programme principal s'exécute toujours dans le niveau de programme supérieur, le niveau 0. Un sous-programme s'exécute toujours dans le niveau de programme directement inférieur consécutif à l'appel. Le niveau de programme 1 correspond ainsi au premier niveau de sous-programme.

Cloisonnement des niveaux de programme :

- Niveau de programme 0 : Niveau de programme principal
- Niveau de programme 1 - 15 : Niveau de sous-programme 1 - 15



Routines d'interruption (ASUP)

Si un sous-programme est appelé dans le cadre d'une routine d'interruption, il n'est pas traité dans le niveau de programme actuel (n) dans le canal actif, mais également dans le niveau de programme immédiatement consécutif (n+1). Afin que ceci soit également possible dans le niveau de programme le plus bas, 2 niveaux de programme supplémentaires (16 et 17) sont disponibles en ce qui concerne les routines d'interruption.

Si plus de 2 niveaux de programme sont requis, il faut en prendre compte de manière explicite dans la structuration du programme pièce exécuté dans le canal. Autrement dit, le nombre maximum de niveaux de programme utilisé doit ne doit pas faire en sorte qu'il n'en existe pas suffisamment pour l'exécution de l'interruption.

Si l'exécution de l'interruption requiert p. ex. 4 niveaux de programme, le programme pièce doit être structuré de sorte à ce qu'il occupe au maximum le niveau de programme 13. Si une interruption a ensuite lieu, les 4 niveaux de programme (14 à 17) qu'elle requiert sont disponibles.

Cycles Siemens

Les cycles Siemens requièrent 3 niveaux de programme. L'appel d'un cycle Siemens doit donc être réalisé au plus tard lors de :

- l'exécution du programme pièce : niveau de programme 12
- la routine d'interruption : niveau de programme 14

1.25.1.4 Chemin de recherche

A l'appel d'un sous-programme sans indication de chemin, la commande effectue une recherche dans les répertoires suivants, selon l'ordre spécifié :

Ordre	Répertoire	Description
1.	Répertoire actuel	Répertoire du programme appelant
2.	/_N_SPF_DIR /	Répertoire global du sous-programme
3.	/_N_CUS_DIR /	Cycles utilisateur
4.	/_N_CMA_DIR /	Cycles constructeur
5.	/_N_CST_DIR /	Cycles standard

1.25.1.5 Paramètres formels et paramètres effectifs

On parle de paramètres formels et de paramètres effectifs dans le contexte de la définition et de l'appel de sous-programmes avec transfert de paramètres.

Paramètres formels

Lors de la définition d'un sous-programme, les paramètres à transférer au sous programme, appelés paramètres formels, doivent être définis avec leur type et nom de paramètre.

Les paramètres formels définissent ainsi l'interface du sous-programme.

Exemple :

```
Code de programme                    Commentaire
PROC CONTOUR (REAL X, REAL Y)        ; Paramètres formels : X et Y, tous deux du
                                       type REAL
N20 X1=X Y1=Y                        ; Déplacement de l'axe X1 à la position X et de
                                       l'axe Y1 à la position Y
...
N100 RET
```

Paramètres effectifs

A l'appel d'un sous-programme, des valeurs absolues ou variables, appelés paramètres formels, doivent être transférées au sous-programme.

Les paramètres effectifs fournissent ainsi des valeurs effectives à l'interface du sous-programme.

Exemple :

```
Code de programme                    Commentaire
N10 DEF REAL LARGEUR                 ; Définition de variable
N20 LARGEUR=20.0                     ; Affectation de variables
N30 CONTOUR (5.5, LARGEUR)           ; Appel de sous-programme avec paramètres
                                       effectifs : 5.5 et LARGEUR
...
N100 M30
```

1.25.1.6 Transfert de paramètres

Définition d'un sous-programme avec transfert de paramètres

La définition d'un sous-programme avec transfert de paramètres s'effectue avec le mot-clé `PROC` et une liste complète de tous les paramètres attendus par le sous-programme.

Transfert incomplet de paramètres

A l'appel du sous-programme, il n'est pas toujours nécessaire de transférer de manière explicite l'ensemble des paramètres définis dans l'interface du sous-programme. Si un paramètre est omis, la valeur par défaut "0" est transférée pour ce paramètre.

Pour une identification univoque de l'ordre des paramètres, les virgules doivent cependant toujours être indiquées comme caractères de séparation des paramètres. Le dernier paramètre constitue une exception. Si celui-ci est omis à l'appel, la dernière virgule peut également l'être.

Exemple :

Sous-programme :

```
Code de programme                              Commentaire
PROC SUB_PROG (REAL X, REAL Y, REAL Z)         ;  Paramètres formels : X, Y et Z.
...
N100 RET
```

Programme principal :

```
Code de programme                  Commentaire
PROC MAIN_PROG
...
N30 SUB_PROG(1.0,2.0,3.0)          ; Appel de sous-programme avec transfert complet de
                                     paramètres :
                                     X=1.0, Y=2.0, Z=3.0
…
N100 M30
```

Exemples d'appel de sous-programme dans `N30`, avec indication complète des paramètres :

```
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
```

PRUDENCE
Transfert de paramètres de type Call-by-Reference
Les paramètres transférés via Call-by-Reference ne doivent pas être omis à l'appel du sous-programme.

PRUDENCE
Type de données AXIS
Les paramètres du type de données AXIS ne doivent pas être omis à l'appel du sous-programme.

Vérification des paramètres de transfert :

La variable système $P_SUBPAR [n] avec n = 1, 2, ... permet de vérifier dans le sous-programme, si un paramètre a été transféré de manière explicite ou omis. L'indice n se rapporte à l'ordre des paramètres formels. L'indice n = 1 se rapporte au 1er paramètre formel, l'indice n = 2 au 2ème paramètre formel, etc.

L'extrait de programme suivant illustre à titre d'exemple comment réaliser une vérification pour le 1er paramètre formel :

```
Programmation                                  Commentaire
PROC SUB_PROG (REAL X, REAL Y, REAL Z)         ; Paramètres formels : X, Y et Z.
N20 IF $P_SUBPAR[1]==TRUE                      ; Vérification du 1er paramètre formel
                                                 X.
...                                            ; Ces actions sont réalisées si le
                                                 paramètre formel X a été transféré de
                                                 manière explicite.
N40 ELSE
...                                            ; Ces actions sont réalisées si le
                                                 paramètre formel X n'a pas été
                                                 transféré.
N60 ENDIF
...                                            ; Actions générales
N100 RET
```

1.25.2 Définition d'un sous-programme

1.25.2.1 Sous-programme sans transfert de paramètres

Fonction

Dans le cas de la définition de sous-programmes sans transfert de paramètres, la ligne de définition au début du programme n'est pas nécessaire.

Syntaxe

```
[PROC <nom de programme>]
...
```

Signification

`PROC` :	Instruction de définition au début d'un programme
`<nom du programme>` :	Nom du programme

Exemple

Exemple 1 : Sous-programme avec l'instruction `PROC`

Code de programme	**Commentaire**
`PROC SUB_PROG`	`; Ligne de définition`
`N10 G01 G90 G64 F1000`	
`N20 X10 Y20`	
`...`	
`N100 RET`	`; Retour dans les sous-programmes`

Exemple 2 : Sous-programme sans instruction `PROC`

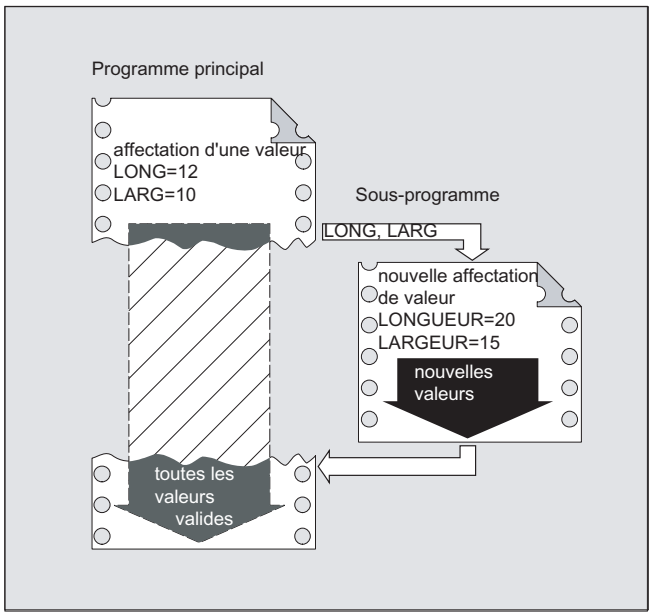
Code de programme	**Commentaire**
`N10 G01 G90 G64 F1000`	
`N20 X10 Y20`	
`...`	
`N100 RET`	`; Retour dans les sous-programmes`

1.25.2.2 Sous-programme avec transfert de paramètres Call-by-Value (PROC)

Fonction

La définition d'un sous-programme avec transfert de paramètres Call-by-Value s'effectue avec le mot-clé `PROC`, suivi du nom de programme et d'une liste complète de tous les paramètres attendus par le sous-programme, y compris le type et le nom. L'instruction de définition doit figurer dans la première ligne du programme.

Le transfert de paramètres Call-by-Value n'a pas d'effets sur le programme appelant. Le programme appelant transfère simplement les valeurs des paramètres effectifs au sous-programme.



Remarque

Il est possible de transférer au maximum 127 paramètres.

Syntaxe

```
PROC <nom de programme> (<type de paramètre> <nom de paramètre>,
...)
```

Signification

`PROC` :	Instruction de définition au début d'un programme
`<nom du programme>` :	Nom du programme
`<type de paramètre>` :	Type de données du paramètre (p. ex. REAL, INT, BOOL)
`<nom de paramètre>` :	Nom du paramètre

IMPORTANT
Le nom de programme indiqué après le mot-clé `PROC` doit correspondre avec le nom de programme attribué dans l'interface utilisateur.

Exemple

Définition d'un sous-programme avec 2 paramètres de type REAL :

Code de programme	**Commentaire**
`PROC SUB_PROG (REAL LONGUEUR, REAL LARGEUR)`	`; Paramètre 1 : Type : REAL, nom : LONGUEUR` `Paramètre 2 : Type : REAL, nom : LARGEUR`
`...`	
`N100 RET`	`; Retour dans les sous-programmes`

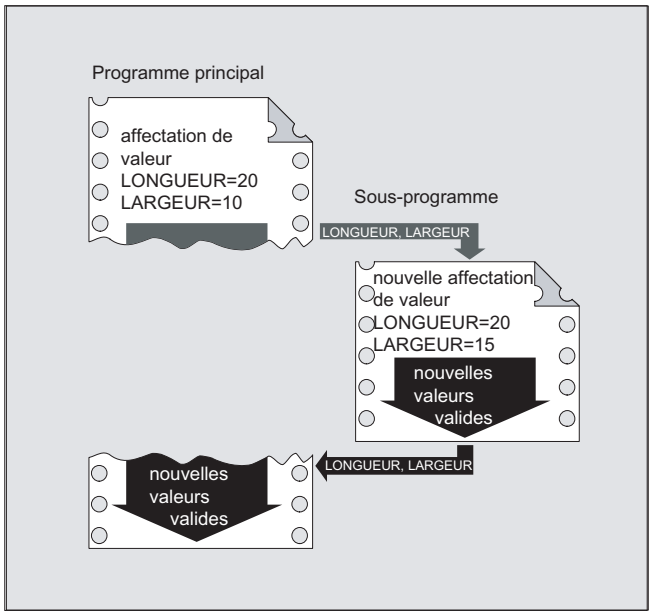
1.25.2.3 Sous-programme avec transfert de paramètres Call-by-Reference (PROC, VAR)

Fonction

La définition d'un sous-programme avec transfert de paramètres Call-by-Reference s'effectue avec le mot-clé `PROC`, suivi du nom de programme et d'une liste complète de tous les paramètres `de type VAR` attendus par le sous-programme, y compris le type et le nom. L'instruction de définition doit figurer dans la première ligne du programme.

Le transfert de paramètres Call-by-Reference permet également de transférer des références à des tableaux.

Le transfert de paramètres Call-by-Reference a des d'effets sur le programme appelant. Le programme appelant transfère au sous-programme une référence au paramètre effectif et lui permet ainsi d'accéder directement à la variable correspondante.



Remarque

Il est possible de transférer au maximum 127 paramètres.

Remarque

Un transfert de paramètres Call-by-Reference est uniquement nécessaire si la variable transmise a été définie dans le programme appelant (LUD). Il n'est pas nécessaire de transférer les variables globales du canal ou à la CN, car il est possible d'y accéder directement à partir du sous-programme.

Syntaxe

```
PROC <nom de programme> (VAR <type de paramètre> <nom de paramètre>,
...)
PROC <nom de programme> (VAR <type de tableau> <nom de tableau>
[<m>,<n>,<o>], ...)
```

Signification

`PROC` :	Instruction de définition au début d'un programme
`VAR` :	Mot-clé pour le transfert de paramètres par référence
`<nom du programme>` :	Nom du programme
`<type de paramètre>` :	Type de données du paramètre (p. ex. REAL, INT, BOOL)
`<nom de paramètre>` :	Nom du paramètre
`<type de tableau>` :	Type de données des éléments de tableau (p. ex. REAL, INT, BOOL)
`<nom de tableau>` :	Nom du tableau
`[<m>,<n>,<o>]` :	Dimension du tableau A l'heure actuelle, l'utilisation de tableaux à 3 dimensions est possible `<m>` : Taille du tableau dans la 1ère dimension `<n>` : Taille du tableau dans la 2ème dimension `<o>` : Taille du tableau dans la 3ème dimension

IMPORTANT

Le nom de programme indiqué après le mot-clé `PROC` doit correspondre avec le nom de programme attribué dans l'interface utilisateur.

Remarque

L'utilisation de tableaux de longueur indéterminée en tant que paramètres formels permet aux sous-programmes de traiter des tableaux de longueur variable. A cet effet, la longueur de la première dimension n'est p. ex. pas indiquée dans la définition d'un tableau bidimensionnel utilisé en tant que paramètre formel. Par contre, la virgule doit y figurer.

Exemple : `PROC <nom de programme> (VAR REAL FELD[ ,5])`

Exemple

Définition d'un sous-programme avec 2 paramètres comme référence au type REAL :

Code de programme	**Commentaire**
`PROC SUB_PROG(VAR REAL LONGUEUR, VAR REAL LARGEUR)`	`; Paramètre 1 : Référence au type : REAL, nom : LONGUEUR` `Paramètre 2 : Référence au type : REAL, nom : LARGEUR`
`...`	
`N100 RET`	

1.25.2.4 Sauvegarde des fonctions G modales (SAVE)

Fonction

Avec l'attribut SAVE, les fonctions G modales actives avant l'appel du sous-programme sont sauvegardées, puis réactivées à la fin du sous-programme.

PRUDENCE
Interruption du contournage Si un sous-programme avec l'attribut SAVE est appelé alors que le contournage est actif, le contournage est interrompu à la fin du sous-programme (saut de retour).

Syntaxe

```
PROC <non de sous-programme> SAVE
```

Signification

SAVE : Sauvegarde des fonctions G modales avant l'appel du sous-programme et restauration à la fin du sous-programme

Exemple

La fonction G modale G91 (cote relative) est activée dans le sous-programme CONTOUR. La fonction G modale G90 (cote absolue) est activée dans le programme principal. En raison de la définition de SAVE dans le sous-programme, G90 agit de nouveau dans la programme principal après la fin du sous-programme.

Définition de sous-programme :

Code de programme	Commentaire
PROC CONTOUR (VALEUR 1 REELLE) SAVE	; Définition de sous-programme avec paramètre SAVE
N10 G91 ...	; Fonction G modale G91 : Cote relative
N100 M17	; Fin de sous-programme

Programme principal :

Code de programme	Commentaire
N10 G0 X... Y... G90	; Fonction G modale G90 : Cote absolue
N20 ...	
...	
N50 CONTOUR (12.4)	; Appel de sous-programme
N60 X... Y...	; Fonction G modale G90 réactivée par SAVE

Contraintes

Frames

Dans le cas de sous-programmes avec l'attribut SAVE, le comportement des frames dépend du type de frame et se règle dans les paramètres machine.

Bibliographie

Description fonctionnelle Fonctions de base ; Axes, Systèmes de coordonnées, Frames (K2)
Chapitre : "Retour de sous-programme avec SAVE"

1.25.2.5 Inhibition de l'exécution bloc par bloc (SBLOF, SBLON)

Fonction

Inhibition du bloc par bloc pour l'ensemble du programme

Les programmes identifiés par SBLOF sont exécutés en entier comme un bloc lorsque l'exécution bloc par bloc est activée, autrement dit l'exécution bloc par bloc est inhibée pour l'ensemble du programme.

SBLOF figure dans la ligne PROC et est actif jusqu'à la fin ou l'abandon du sous-programme. L'instruction de retour permet de déterminer si l'exécution doit ou non être interrompue à la fin du sous-programme :

Retour avec M17 :	Arrêt en fin de sous-programme
Retour avec RET :	Pas d'arrêt en fin de sous-programme

Inhibition du bloc par bloc à l'intérieur du programme

SBLOF doit être le seul élément du bloc. A partir de ce bloc, le mode bloc par bloc est désactivé :

- jusqu'au prochain SBLON

  ou

- jusqu'à la fin du niveau de sous-programme actif.

Syntaxe

Inhibition du bloc par bloc pour l'ensemble du programme :

```
PROC ... SBLOF
```

Inhibition du bloc par bloc à l'intérieur du programme :

```
SBLOF
...
SBLON
```

Signification

PROC :	Première instruction d'un programme
SBLOF :	Instruction de désactivation de l'exécution bloc par bloc SBLOF peut figurer dans un bloc PROC ou dans un bloc spécifique.
SBLON :	Instruction d'activation de l'exécution bloc par bloc SBLON doit figurer dans un bloc spécifique.

Conditions marginales

- **Inhibition du bloc par bloc et affichage de bloc**

  DISPLOF permet d'inhiber l'affichage du bloc courant dans les cycles/sous-programmes. Si DISPLOF est programmé en liaison avec SBLOF, l'appel de cycle/sous-programme est encore affiché en cas d'arrêts bloc par bloc dans le cycle/sous-programme.

- **Inhibition du bloc par bloc dans l'ASUP système ou l'ASUP utilisateur**

  Si l'arrêt bloc par bloc est inhibé dans l'ASUP système ou utilisateur par les réglages du paramètre machine MD10702 $MN_IGNORE_SINGLEBLOCK_MASK (bit0 = 1 et bit1 = 1), la programmation de SBLON permet de réactiver l'arrêt bloc par bloc dans l'ASUP.

  Si l'arrêt bloc par bloc est inhibé dans l'ASUP utilisateur par le réglage du paramètre machine MD20117 $MC_IGNORE_SINGLEBLOCK_ASUP, il n'est **pas** possible de réactiver l'arrêt bloc par bloc dans l'ASUP en programmant SBLON.

- **Particularités de l'inhibition du bloc par bloc pour les différents types d'exécution bloc par bloc**

  Lorsque l'exécution bloc par bloc SBL2 (arrêt après chaque bloc du programme pièce) est activée, il n'y a **pas** d'arrêt dans le bloc SBLON si le bit 12 est mis à "1" dans le paramètre machine MD10702 $MN_IGNORE_SINGLEBLOCK_MASK (inhibition de l'arrêt bloc par bloc).

  Lorsque l'exécution bloc par bloc SBL3 (arrêt après chaque bloc du programme pièce même dans un cycle), l'instruction SBLOF est inhibée.

Exemples

Exemple 1 : Inhibition du bloc par bloc à l'intérieur d'un programme

```
Code de programme            Commentaire
N10 G1 X100 F1000
N20 SBLOF                    ; Désactivation du bloc par bloc
N30 Y20
N40 M100
N50 R10=90
N60 SBLON                    ; Réactivation du bloc par bloc
N70 M110
N80 ...
```

En mode bloc par bloc, la section de programme entre N20 et N60 est exécutée comme un pas.

Exemple 2 : Cycle devant apparaître comme une instruction pour l'utilisateur

Programme principal :

```
Code de programme
N10 G1 X10 G90 F200
N20 X-4 Y6
N30 CYCLE1
N40 G1 X0
N50 M30
```

CYCLE1 :

```
Code de programme                          Commentaire
N100 PROC CYCLE1 DISPLOF SBLOF             ; Inhibition du bloc par bloc
N110 R10=3*SIN(R20)+5
N120 IF (R11 <= 0)
N130 SETAL(61000)
N140 ENDIF
N150 G1 G91 Z=R10 F=R11
N160 M17
```

Le CYCLE1 est exécuté lorsque l'exécution bloc par bloc est active, c'est-à-dire que la touche Start doit être activée une fois pour le traitement du CYCLE1.

Exemple 3 :

Un ASUP lancé par l'AP pour l'activation de décalages d'origine et de corrections d'outil modifiés ne doit pas être visible.

```
Code de programme
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
```

Exemple 4 : Lorsque le bit 12 = 1 dans le paramètre machine MD10702, il n'y a pas d'arrêt.

Situation de départ :

- L'exécution bloc par bloc est activée.
- MD10702 $MN_IGNORE_SINGLEBLOCK_MASK Bit12 = 1

Programme principal :

Code de programme	Commentaire
N10 G0 X0	; Arrêt dans cette ligne du programme pièce
N20 X10	; Arrêt dans cette ligne du programme pièce
N30 CYCLE	; Bloc de déplacement généré par le cycle.
N50 G90 X20	; Arrêt dans cette ligne du programme pièce
M30	

CYCLE :

Code de programme	Commentaire
PROC CYCLE SBLOF	; Inhibition de l'arrêt bloc par bloc
N100 R0 = 1	
N110 SBLON	; Comme le bit 12 = 1 dans le paramètre machine MD10702, il n'y a pas d'arrêt dans cette ligne du programme pièce.
N120 X1	; Dans cette ligne du programme pièce, il y a arrêt.
N140 SBLOF	
N150 R0 = 2	
RET	

Exemple 5 : Inhibition du bloc par bloc dans le cas d'un programme imbriqué

Situation de départ :

L'exécution bloc par bloc est activée.

Imbrication de programmes :

```
Code de programme                          Commentaire
N10 X0 F1000                               ; Dans ce bloc, il y a arrêt.
N20 UP1(0)
      PROC UP1(INT _NR) SBLOF              ; Inhibition de l'arrêt du bloc par bloc.
      N100 X10
      N110 UP2(0)
            PROC UP2(INT _NR)
            N200 X20
            N210 SBLON                     ; Activation de l'arrêt bloc par bloc.
            N220 X22                       ; Dans ce bloc, il y a arrêt.
            N230 UP3(0)
                  PROC UP3(INT _NR)
                  N300 SBLOF               ; Inhibition de l'arrêt du bloc par bloc.
                  N305 X30
                  N310 SBLON               ; Activation de l'arrêt bloc par bloc.
                  N320 X32                 ; Dans ce bloc, il y a arrêt.
                  N330 SBLOF               ; Inhibition de l'arrêt du bloc par bloc.
                  N340 X34
                  N350 M17                 ; SBLOF est actif.
            N240 X24                       ; Dans ce bloc, il y a arrêt. SBLON est
                                             actif.
            N250 M17                       ; Dans ce bloc, il y a arrêt. SBLON est
                                             actif.
      N120 X12
      N130 M17                             ; Dans ce bloc de retour, il y a arrêt.
                                               SBLOF de l'instruction PROC est actif.
N30 X0                                     ; Dans ce bloc, il y a arrêt.
N40 M30                                    ; Dans ce bloc, il y a arrêt.
```

Autres informations

Blocage du bloc par bloc pour des sous-programmes asynchrones

Pour exécuter un sous-programme asynchrone bloc par bloc en une seule étape, il est nécessaire de programmer une instruction PROC avec SBLOF dans ce sous-programme asynchrone. Ceci est également valable pour la fonction "ASUP système modifiable" (MD11610 $MN_ASUP_EDITABLE).

Exemple d'ASUP système modifiable :

```
Code de programme                          Commentaire
N10 PROC ASUP1 SBLOF DISPLOF
N20 IF $AC_ASUP==’H200’
N30 RET                                    ; Pas de REPOS si changement de mode de
                                             fonctionnement.
N40 ELSE
N50 REPOSA                                 ; REPOS dans tous les autres cas.
N60 ENDIF
```

Influence sur le programme dans le bloc par bloc

En exécution bloc par bloc, l'utilisateur peut exécuter le programme pièce bloc après bloc. Il existe les types de réglage suivants :

- SBL1 : Bloc par bloc IPO avec arrêt après chaque bloc de fonction machine.
- SBL2 : Bloc par bloc avec arrêt après chaque bloc.
- SBL3 : Interruption du cycle (l’instruction SBLOF est inhibée par la sélection de SBL3).

Inhibition du bloc par bloc dans le cas d'un programme imbriqué

Si, dans un sous-programme, SBLOF a été programmé à l’intérieur de l’instruction PROC, l’arrêt intervient lors du retour au sous-programme avec M17. L'exécution du bloc suivant dans le programme appelant est ainsi inhibée. Si une inhibition du bloc par bloc a été activée avec SBLOF, sans SBLOF dans un sous-programme dans l'instruction PROC, le programme s'arrête après l'exécution du bloc machine suivant dans le programme appelant. Si vous ne le désirez pas, vous devez reprogrammer SBLON dans le sous-programme juste avant le retour (M17). Dans le cas d'un retour dans un programme de niveau supérieur avec RET l'exécution du programme n'est pas arrêtée.

1.25.2.6 Inhibition de l'affichage du bloc courant (DISPLOF, DISPLON, ACTBLOCNO)

Fonction

Le bloc courant du programme s'affiche par défaut. Dans les cycles ou sous-programmes, l'affichage du bloc courant peut être inhibé par l'instruction DISPLOF. C'est l'appel du cycle ou du sous-programme qui s'affiche alors à la place du bloc courant. L'instruction DISPLON permet d'annuler l'inhibition de l'affichage du bloc courant.

DISPLOF ou DISPLON sont programmées dans la ligne de programme avec l'instruction PROC et s'appliquent à l'ensemble du sous-programme et, implicitement, à tous les sous-programmes appelés par celui-ci et ne contenant pas d'instruction DISPLON ou DISPLOF. Ce comportement s'applique également aux ASUP.

Syntaxe

```
PROC … DISPLOF
PROC … DISPLOF ACTBLOCNO
PROC … DISPLON
```

Signification

DISPLOF : Instruction d'inhibition de l'affichage du bloc courant.

Positionnement : A la fin de la ligne de programme avec l'instruction PROC

Prise d'effet : Jusqu'au retour depuis le sous-programme ou la fin du programme.

Nota :
si, avec l'instruction DISPLOF, vous appelez d'autres sous-programmes depuis le sous-programme, l'affichage du bloc courant y sera également inhibé, à condition que DISPLON n'y soit pas programmée de manière explicite.

DISPLON : Instruction d'annulation de l'inhibition de l'affichage du bloc courant

Positionnement : A la fin de la ligne de programme avec l'instruction PROC

Prise d'effet : Jusqu'au retour depuis le sous-programme ou la fin du programme.

Nota :
si, avec l'instruction DISPLON, vous appelez d'autres sous-programmes depuis le sous-programme, le bloc de programme courant s'y affiche également, à condition que DISPLOF n'y soit pas programmée de manière explicite.

ACTBLOCNO : Utilisé avec l'attribut ACTBLOCNO, DISPLOF a pour effet d'afficher le numéro du bloc courant dans lequel une éventuelle alarme est survenue. Ceci est également valable lorsque DISPLOF est programmé seul dans un niveau de programme inférieur.

Lorsque DISPLOF est programmé sans ACTBLOCNO, c'est par contre le numéro de bloc de l'appel de cycle ou de sous-programme du dernier niveau de programme sans DISPLOF qui s'affiche.

Exemples

Exemple 1 : Inhibition de l'affichage du bloc courant dans le cycle

```
Code de programme                    Commentaire
PROC CYCLE (AXIS TOMOV, REAL         ; Inhibition de l'affichage du bloc courant. L'appel de cycle doit
POSITION) SAVE DISPLOF                 s'afficher à sa place, p. ex. : CYCLE(X,100.0)
DEF REAL DIFF                        ; Contenu du cycle
G01 ...
...
RET                                  ; Retour de sous-programme. Le bloc qui suit l'appel du cycle s'affiche.
```

Exemple 2 : Affichage de bloc en cas d'émission d'une alarme

Sous-programme SUBPROG1 (avec ACTBLOCNO) :

```
Code de programme                              Commentaire
PROC SUBPROG1 DISPLOF ACTBLOCNO
N8000 R10 = R33 + R44
...
N9040 R10 = 66 X100                            ; Déclenchement de l'alarme 12080
...
N10000 M17
```

Sous-programme SUBPROG2 (sans ACTBLOCNO) :

```
Code de programme                    Commentaire
PROC SUBPROG2 DISPLOF
N5000 R10 = R33 + R44
...
N6040 R10 = 66 X100                  ; Déclenchement de l'alarme 12080
...
N7000 M17
```

Programme principal :

```
Code de programme                    Commentaire
N1000 G0 X0 Y0 Z0
N1010 ...
...
N2050 SUBPROG1                       ; Message d'alarme = "12080 canal K1 bloc N9040
                                       erreur de syntaxe dans texte R10="
N2060 ...
N2350 SUBPROG2                       ; Message d'alarme = "12080 canal K1 bloc N2350
                                       erreur de syntaxe dans texte R10="
...
N3000 M30
```

Exemple 3 : annulation de l'inhibition d'affichage du bloc courant

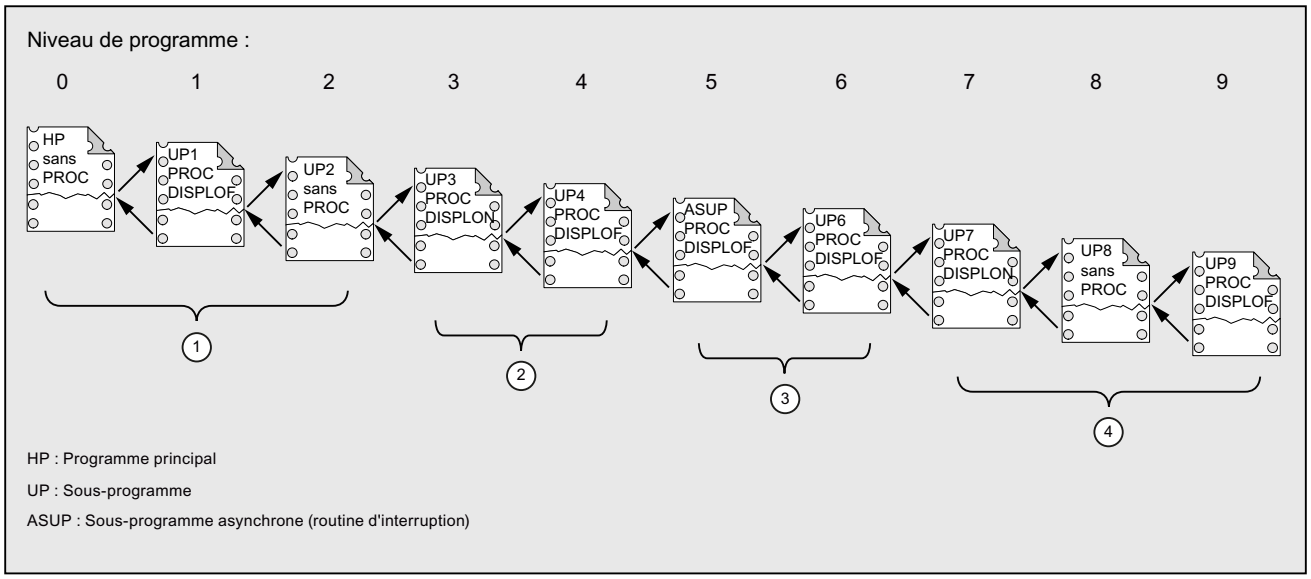
Sous-programme SUB1 avec inhibition :

Code de programme	Commentaire
PROC SUB1 DISPLOF	; Inhiber l'affichage du bloc courant dans le sous-programme SUB1. Afficher à la place le bloc avec l'appel SUB1.
...	
N300 SUB2	; Appeler le sous-programme SUB2.
...	
N500 M17	

Sous-programme SUB2 sans inhibition :

Code de programme	Commentaire
PROC SUB2 DISPLON	; Annuler l'inhibition d'affichage du bloc courant dans le sous-programme SUB2.
...	
N200 M17	; Retour au sous-programme SUB1. L'affichage du bloc courant est de nouveau inhibé dans SUB1.

Exemple 4 : comportement d'affichage pour diverses combinaisons DISPLON/DISPLOF



① L'affichage du bloc courant affiche les lignes du programme pièce du niveau 0.
② L'affichage du bloc courant affiche les lignes du programme pièce du niveau 3.
③ L'affichage du bloc courant affiche les lignes du programme pièce du niveau 3.
④ L'affichage du bloc courant affiche les lignes du programme pièce du niveau 7/8.

1.25.2.7 Identifier les sous-programmes avec prétraitement (PREPRO)

Fonction

Le mot-clé PREPRO permet d'identifier tous les fichiers pendant le démarrage réalisé à la fin de la ligne d'instruction PROC.

Remarque

Ce prétraitement de programme dépend des paramètres machine configurés. Veuillez tenir compte des indications du constructeur de la machine.

Bibliographie :
Description fonctionnelle Fonctions spéciales ; Conditionnement (V2)

Syntaxe

```
PROC … PREPRO
```

Signification

PREPRO : Mot clé pour l'identification de tous les fichiers prétraités pendant le démarrage relatifs aux programmes se trouvant dans les répertoires cycliques

Lecture de sous-programmes avec préparation et appel de sous-programme

Les répertoires de cycle sont traités dans le même ordre, aussi bien à la mise en route de sous-programmes préparés avec des paramètres, qu'à l'appel du sous-programme :

1. _N_CUS_DIR cycles utilisateurs
2. _N_CMA_DIR cycles constructeurs
3. _N_CST_DIR cycles standard

S'il existe des programmes CN de même nom avec un ciblage différent, l'instruction PROC trouvée en premier est activée et l'autre instruction PROC est ignorée sans qu'il y ait de message d'alarme.

1.25.2.8 Retour de sous-programme M17

Fonction

A la fin d'un sous-programme, on trouve l'instruction de retour `M17` (ou l'instruction de fin du programme pièce `M30`). Elle entraîne le retour dans le programme appelant sur le bloc du programme pièce suivant l'appel du sous-programme.

Remarque

`M17` et `M30` sont traités de la même manière dans le langage CN.

Syntaxe

```
PROC <nom de programme>
...
M17/M30
```

Conditions marginales

effet du retour du sous-programme sur le mode contournage

Si `M17` (ou `M30`) figure seul dans le bloc du programme pièce, un contournage actif dans le canal est interrompu.

Pour éviter l'interruption du contournage, `M17` (ou `M30`) doit également être inscrit dans le dernier bloc de déplacement. De plus, le paramètre machine suivant doit être mis sur "0" :

MD20800 $MC_SPF_END_TO_VDI = 0 (pas de sortie M30/M17 dans l'interface CN/AP)

Exemple

1. Sous-programme avec `M17` dans un bloc spécifique

```
Code de programme                Commentaire
N10 G64 F2000 G91 X10 Y10
N20 X10 Z10
N30 M17                          ; Retour avec interruption du contournage.
```

2. Sous-programme avec `M17` dans le dernier bloc

```
Code de programme                Commentaire
N10 G64 F2000 G91 X10 Y10
N20 X10 Z10 M17                  ; Retour sans interruption du contournage.
```

1.25.2.9 Retour de sous-programme RET

Fonction

A la place de l'instruction de M17, il est également possible d'utiliser l'instruction RET dans le sous-programme. RET doit être programmée dans un bloc spécifique du programme pièce. Comme M17, RET entraîne le retour dans le programme appelant sur le bloc du programme pièce suivant l'appel du sous-programme.

Remarque

La programmation de paramètres permet de modifier le comportement de retour de RET (voir "Retour paramétrable dans les sous-programmes (RET ...) (Page 171)").

Application

L'instruction RET doit être utilisée lorsqu'un contournage G64 (G641 ... G645) ne doit pas être interrompu par le saut de retour.

Condition

L'instruction RET peut uniquement être utilisée dans des sous-programmes n'ayant pas été définis avec l'attribut SAVE.

Syntaxe

```
PROC <nom de programme>
...
RET
```

Exemple

Programme principal :

Code de programme	Commentaire
PROC MAIN_PROGRAM	; Début du programme
...	
N50 SUB_PROG	; Appel de sous-programme : SUB_PROG
N60 ...	
...	
N100 M30	; Fin du programme

Sous-programme :

```
Code de programme          Commentaire
PROC SUB_PROG
...
N100 RET                   ; Le retour s'effectue sur le bloc N60 dans le programme
                             principal.
```

1.25.2.10 Retour paramétrable dans les sous-programmes (RET ...)

Fonction

Depuis un sous-programme, le retour dans le programme depuis lequel le sous-programme a été appelé s'effectue généralement avec une fin de sous-programme `RET` ou `M17`, l'exécution reprenant à la ligne de programme qui suit l'appel du sous-programme.

Il existe cependant des cas où l'exécution du programme doit reprendre à un autre endroit. Par exemple :

- Poursuite de l'exécution du programme après l'appel des cycles de chariotage en dialecte ISO (après la description du contour).
- Retour dans le programme principal à partir d'un niveau de sous-programmes quelconque (après un sous-programme asynchrone ASUP aussi) pour traiter une erreur.
- Retour sur plusieurs niveaux de programme pour des applications spéciales dans des cycles de compilation, en dialecte ISO.

Dans ces cas, l'instruction `RET` est programmée avec des "paramètres de retour".

Syntaxe

```
RET("<bloc de destination>")
RET("<bloc de destination>",<bloc suivant le bloc de destination>)
RET("<bloc de destination>",<bloc suivant le bloc de destination>,
<nombre de niveaux de retour>)
RET("<bloc de destination>", ,<nombre de niveaux de retour>)
RET("<bloc de destination>",<bloc suivant le bloc de destination>,
<nombre de niveaux de retour>,
<retour au début du programme>)
RET (, ,<nombre  de niveaux de retour>,<retour au début du
programme>)
```

Signification

`RET` :
Fin du sous-programme (à utiliser à la place de `M17`)

`<bloc de destination>` :
Paramètre de retour 1

Indication du bloc où l'exécution du programme doit reprendre, comme destination de saut.

Si le paramètre de retour 3 n'est pas programmé, la destination de saut se trouve dans le programme depuis lequel le sous-programme courant a été appelé.

Les indications possibles sont :

"<Numéro de bloc>"	Numéro du bloc de destination
"<Repère de saut>"	Repère de saut à définir dans le bloc de destination.
"<Chaîne de caractères>"	Chaîne de caractères devant être connue dans le programme (par exemple nom de programme ou de variable). Les règles suivantes sont valables pour la programmation de la chaîne de caractères dans le bloc de destination : • **Espace à la fin** (contrairement au repère de saut, identifié par un ":" à la fin). • Seuls un numéro de bloc et/ou un repère de saut peuvent **précéder la chaîne de caractères** (**pas d'instructions de programme**).

`<bloc suivant le bloc de destination>` :
Paramètre de retour 2

Se rapporte au paramètre de retour 1.

Type : INT

Valeur :
- 0 : Retour au bloc indiqué dans le paramètre de retour 1.
- > 0 : Retour au bloc qui suit le bloc indiqué dans le paramètre de retour 1.

`<Nombre de niveaux de retour>` :

Paramètre de retour 3

Nombre de niveaux qu'il faut retourner pour parvenir au niveau de programme où l'exécution du programme doit reprendre.

Type : INT

Valeur :
- 1 L'exécution du programme se poursuit au "niveau de programme courant - 1" (comme `RET` sans paramètre).
- 2 L'exécution du programme se poursuit au "niveau de programme courant - 2", un niveau étant sauté.
- 3 L'exécution du programme se poursuit au "niveau de programme courant - 3", deux niveaux étant sautés.
- ...

Plage de valeurs : 1 ... 15

`<Retour au début du programme>` :

Paramètre de retour 4

Type : BOOL

Valeur :
- 1 Si le retour se fait dans le programme principal et si le **dialecte ISO** y est activé, le saut se fera au début du programme.

Remarque

Lorsqu'une chaîne de caractères est indiquée pour la recherche du bloc de destination d'un saut de retour de sous-programme, le système recherche toujours d'abord un repère de saut dans le programme appelant.

Pour qu'une chaîne de caractères définisse une destination de saut univoque, la chaîne de caractères ne doit pas être identique à un nom de repère de saut, le saut de retour de sous-programme étant sinon toujours effectué sur le repère de saut et non pas sur la chaîne de caractères (voir l'exemple 2).

Conditions marginales

Lors d'un retour sur plusieurs niveaux de programme, les instructions SAVE des différents niveaux sont évaluées.

Si un sous-programme modal est actif lors d'un retour sur plusieurs niveaux de programme et que l'instruction de désélection MCALL est programmée pour le sous-programme modal dans l'un des sous-programmes sautés, le sous-programme modal reste actif.

⚠PRUDENCE

Le programmeur doit veiller à ce que l'exécution reprenne avec les réglages modaux corrects lors d'un retour sur plusieurs niveaux de programme. Ceci s'obtient, par exemple, par la programmation d'un bloc principal adéquat.

Exemples

Exemple 1 : Retour dans le programme principal après l'exécution d'un sous-programme asynchrone ASUP

```
Programmation                                  Commentaire
N10010 CALL "UP1"                              ; Niveau de programme 0 (programme
                                                 principal)
  N11000 PROC UP1                              ; Niveau de programme 1
  N11010 CALL "UP2"
    N12000 PROC UP2                            ; Niveau de programme 2
    ...
      N19000 PROC ASUP                         ; Niveau de programme 3 (exécution d'un
                                                 sous-programme asynchrone ASUP)
      ...
      N19100 RET("N10900", ,$P_STACK)          ; Retour dans les sous-programmes
N10900                                         ; Retour dans le programme principal.
N10910 MCALL                                   ; Désactivation du sous-programme modal
N10920 G0 G60 G40 M5                           ; Correction d'autres réglages modaux.
```

Exemple 2 : Chaîne de caractères (<chaîne de caractères>) en tant qu'indication pour la recherche du bloc de destination

Programme principal :

```
Code de programme                      Commentaire
PROC MAIN_PROGRAM
N1000 DEF INT iVar1=1, iVar2=4
N1010 ...
N1200 subProg1                         ; Appel du sous-programme "subProg1"
N1210 M2 S1000 X10 F1000
N1220 ......
N1400 subProg2                         ; Appel du sous-programme "subProg2"
```

Code de programme	Commentaire
N1410 M3 S500 Y20	
N1420 ..	
N1500 lab1: iVar1=R10*44	
N1510 F500 X5	
N1520 ...	
N1550 subprog1: G1 X30	; "subProg1" est défini ici comme repère de saut.
N1560 ...	
N1600 subProg3	Appel du sous-programme "subProg3"
N1610 ...	
N1900 M30	

Sous-programme subProg1 :

Code de programme	Commentaire
PROC subProg1	
N2000 R10=R20+100	
N2010 ...	
N2200 RET("subProg2")	; Retour au bloc N1400 du programme principal

Sous-programme subProg2 :

Code de programme	Commentaire
PROC subProg2	
N2000 R10=R20+100	
N2010 ...	
N2200 RET("iVar1")	; Retour au bloc N1500 du programme principal

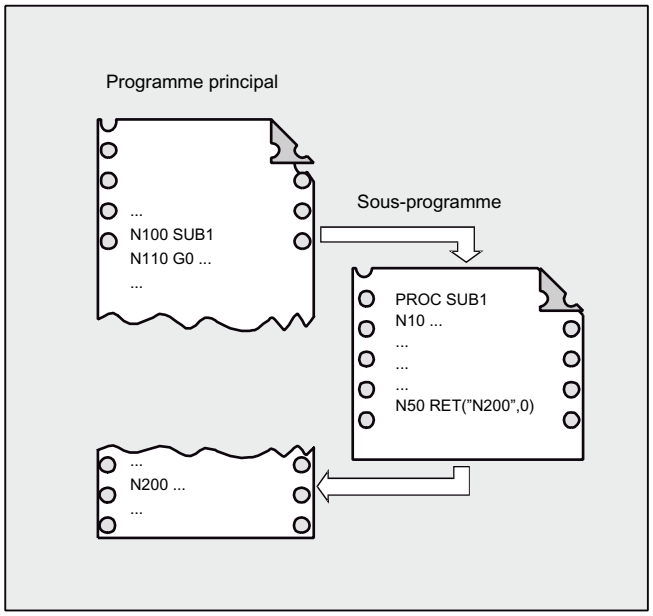
Sous-programme subProg3 :

Code de programme	Commentaire
PROC subProg3	
N2000 R10=R20+100	
N2010 ...	
N2200 RET("subProg1")	; Retour au bloc N1550 du programme principal

Informations complémentaires

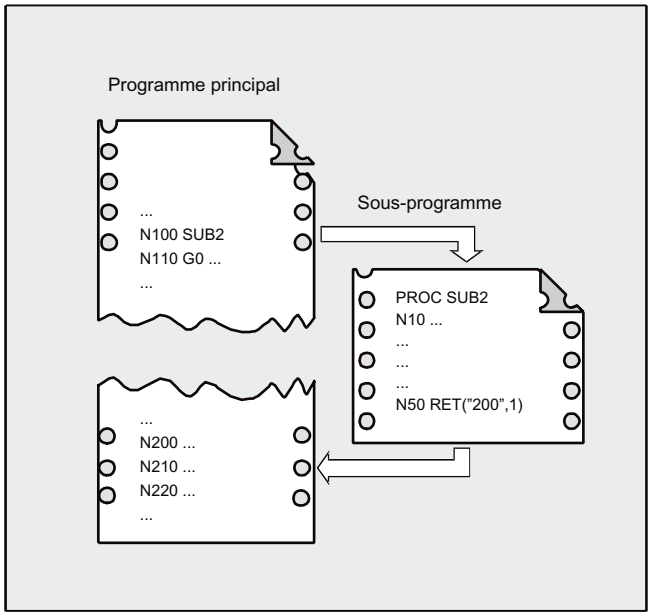
Les graphiques suivants mettent en évidence les différents effets des paramètres de retour 1 à 3.

1. Paramètre de retour 1 = "N200", paramètre de retour 2 = 0



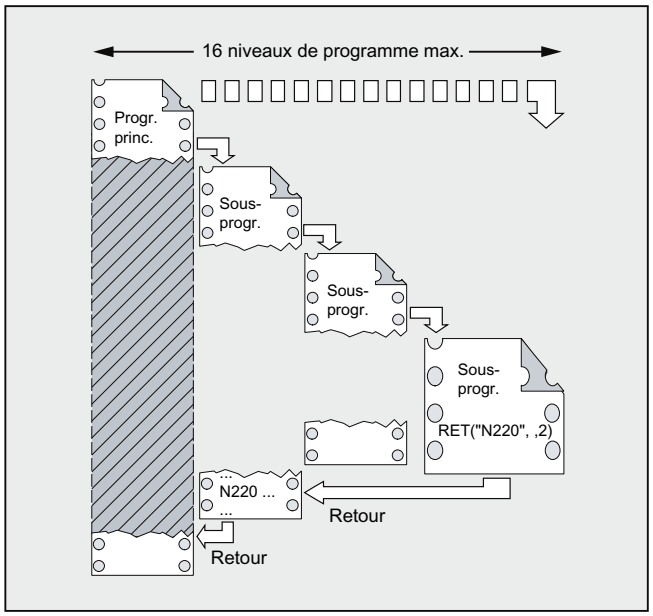
Après l'instruction `RET`, l'exécution du programme reprend au bloc `N200` du programme principal.

2. Paramètre de retour 1 = "N200", paramètre de retour 2 = 1



Après l'instruction `RET`, l'exécution du programme reprend au bloc (`N210`) qui suit le bloc `N200` dans programme principal.

3. Paramètre de retour 1 = "N220", paramètre de retour 3 = 2



Après l'instruction `RET`, l'exécution du programme retourne de deux niveaux de programme et reprend au bloc `N220`.

1.25.3 Appel d'un sous-programme

1.25.3.1 Appel de sous-programme sans transfert de paramètres

Fonction

L'appel d'un sous-programme s'effectue soit avec l'adresse L et le numéro de sous-programme, soit par indication du nom du programme.

Un programme principal peut être appelé aussi en tant que sous-programme. Dans ce cas, la fin du programme `M2` ou `M30` spécifiée dans le programme principal est traitée comme `M17` (fin du programme avec retour au programme appelant).

Remarque

Inversement, un sous-programme peut également être lancé comme programme principal.

Stratégie de recherche de la commande :

*_MPF existe-t-il ?

*_SPF existe-t-il ?

Il en résulte : si le nom du sous-programme appelant est identique au nom du programme principal, c'est le programme principal qui sera rappelé. Pour éviter cet effet qu'on ne désire généralement pas, il convient de nommer sans aucune ambiguïté les programmes principaux et les sous-programmes.

Remarque

Des sous-programmes ne nécessitant pas de transfert de paramètres peuvent également être appelés à partir d'un fichier d'initialisation.

Syntaxe

```
L<numéro>/<nom de programme>
```

Remarque

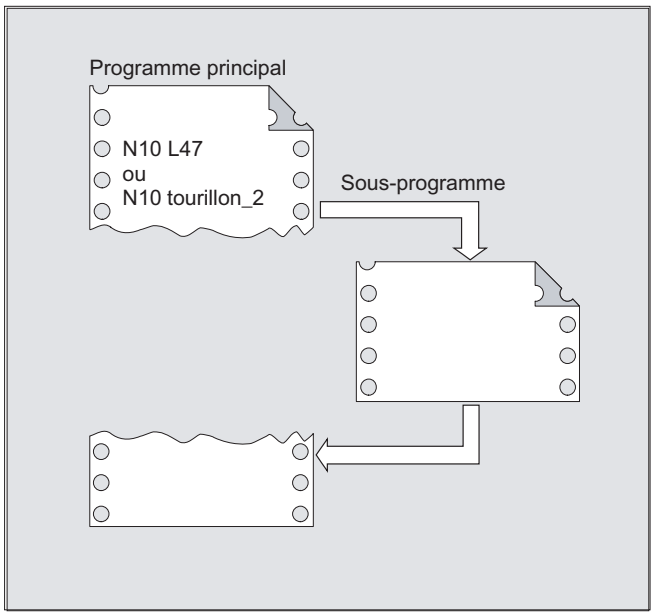
L'appel d'un sous-programme doit toujours être programmé dans le bloc CN spécifique.

Signification

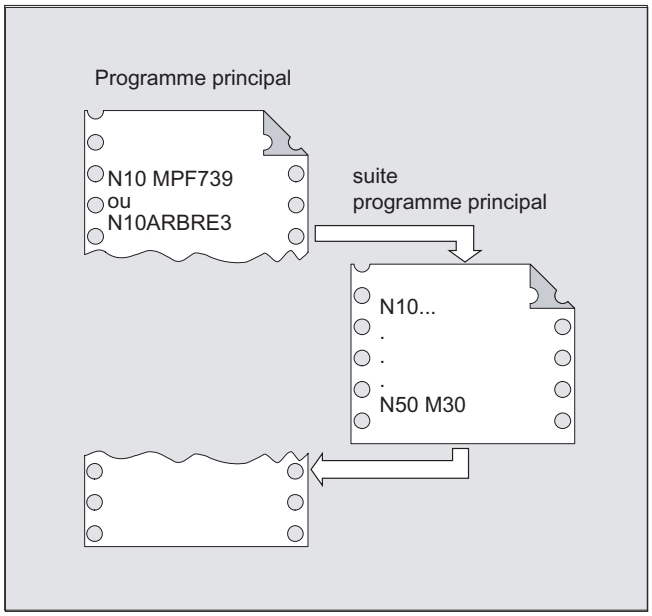
`L` :	Adresse pour l'appel du sous-programme
`<numéro>` :	Numéro du sous-programme Type : INT Valeur : Maximum 7 chiffres en notation décimale **Attention :** Les zéros en début de nom ont une signification (⇒ L123, L0123 et L00123 sont trois sous-programmes différents).
`<nom du programme>` :	Nom du sous-programme (ou programme principal)

Exemples

Exemple 1 : appel d'un sous-programme sans transfert de paramètres



Exemple 2 : appel d'un programme principal comme sous-programme



1.25.3.2 Appel d'un sous-programme avec transfert de paramètres (EXTERN)

Fonction

A l'appel d'un sous-programme avec transfert de paramètres, des variables ou valeurs peuvent être transférées directement (sauf pour les paramètres VAR).

Les sous-programmes avec transfert de paramètres doivent être déclarés dans le programme principal avec l'instruction EXTERN avant leur appel (par exemple en début de programme). Il faut alors indiquer le nom du sous-programme et les types de variables dans l'ordre du transfert.

⚠ PRUDENCE

Les types de variables de même que l'ordre de transfert doivent correspondre aux définitions déclarées sous PROC dans le sous-programme. Les noms des paramètres peuvent être différents dans le programme principal et le sous-programme.

Syntaxe

```
EXTERN <nom de programme>(<type_Par1>,<type_Par2>,<type_Par3>)
...
<Nom de programme>(<valeur_Par1>,<valeur_Par2>,<valeur_Par3>)
```

⚠ PRUDENCE
Un appel de sous-programme doit toujours être programmé dans un bloc CN spécifique.

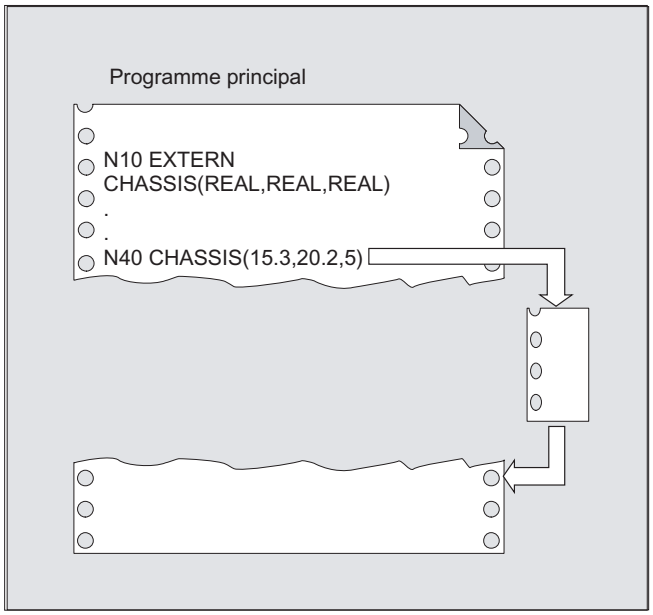
Signification

`<nom du programme>` :	Nom du sous-programme
`EXTERN` :	Mot clé pour la déclaration d'un sous-programme avec transfert de paramètres **Remarque :** L'instruction `EXTERN` ne doit être indiquée que si le sous-programme se trouve dans le répertoire pièce ou dans le répertoire des sous-programmes globaux. Il n'est pas nécessaire de déclarer les cycles avec l'instruction `EXTERN` .
`<type_par1>,<type_par2>,<type_par3>` :	Types de variable des paramètres à transférer, dans l'ordre du transfert
`<valeur_par1>,<valeur_par2>,<valeur_par3>` :	Valeurs de variable des paramètres à transférer

Exemples

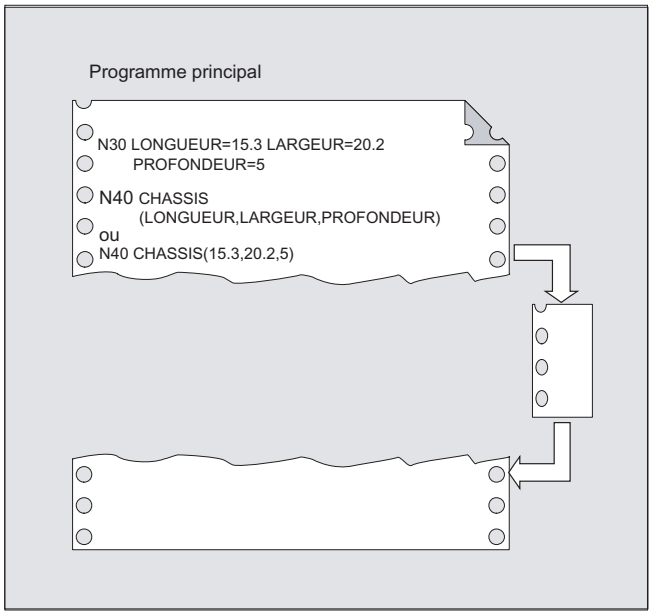
Exemple 1 : appel d'un sous-programme avec déclaration préalable

```
Code de programme                          Commentaire
N10 EXTERN CADRE(REAL,REAL,REAL)           ; Indication du sous-programme.
...
N40 CADRE(15.3,20.2,5)                     ; Appel du sous-programme avec
                                             transfert de paramètres.
```



Exemple 2 : appel d'un sous-programme sans déclaration préalable

Code de programme	**Commentaire**
`N10 DEF REAL LONGUEUR, LARGEUR, PROFONDEUR`	
`N20 …`	
`N30 LONGUEUR=15.3 LARGEUR=20.2 PROFONDEUR=5`	
`N40 CADRE(LONGUEUR,LARGEUR,PROFONDEUR)`	`; ou : N40 CADRE(15.3,20.2,5)`



1.25.3.3 Nombre de répétitions de programme (P)

Fonction

Si un sous-programme doit être exécuté plusieurs fois d'affilée, le nombre souhaité de répétitions peut être programmé sous l'adresse `P`, dans le bloc qui contient l'appel du sous-programme.

> **⚠ PRUDENCE**
>
> **Appel de sous-programme avec répétition et transfert de paramètres**
>
> Les paramètres sont transférés uniquement à l'appel du sous-programme, c'est-à-dire lors de la première exécution. Ils restent inchangés lors des répétitions. Si vous désirez que les paramètres soient modifiés lors de répétitions, vous devez définir des conventions correspondantes dans le sous-programme.

Syntaxe

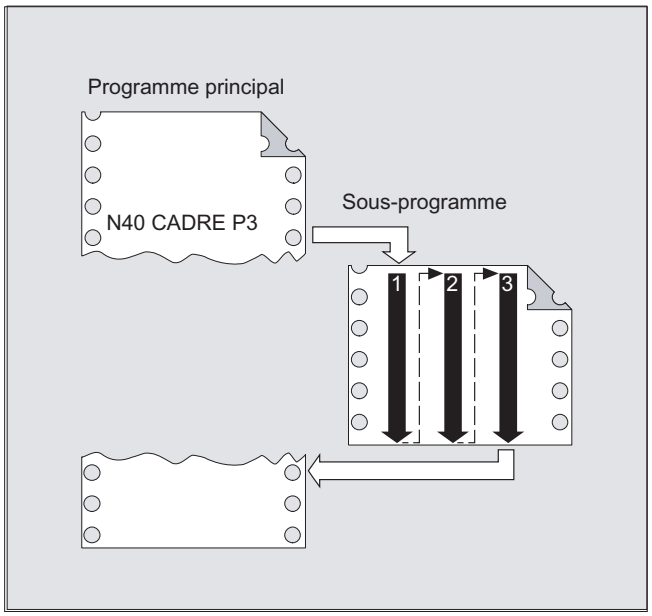
```
<nom du programme> P<valeur>
```

Signification

`<nom du programme>` :	appel de sous-programme	
`P` :	Adresse pour la programmation des répétitions de programme	
`<valeur>` :	Nombre de répétitions de programme	
	Type :	INT
	Plage de valeurs :	1 … 9999 (sans signe)

Exemple

Code de programme	Commentaire
...	
N40 CADRE P3	; Le sous-programme CADRE doit être exécuté trois fois d'affilée.
...	



1.25.3.4 Appel modal d'un sous-programme (MCALL)

Fonction

L'appel modal d'un sous-programme avec MCALL exécute automatiquement le sous-programme après chaque bloc comportant un déplacement de positionnement. Ceci permet d'automatiser l'appel de sous-programmes qui doivent être exécutés à différentes positions de la pièce (par exemple pour la réalisation de réseaux de trous).

La fonction se désactive avec MCALL sans appel de sous-programme, ou par programmation d'un nouvel appel modal pour un autre sous-programme.

⚠ PRUDENCE

Un seul appel MCALL peut être actif à un instant donnée dans l'exécution d'un programme. Les paramètres sont transférés une seule fois lors de l'appel MCALL.

Dans les conditions suivantes, le sous-programme modal appelle un déplacement même sans programmation :

- Programmation des adresses S et F lorsque G0 ou G1 est actif.
- G0/G1 programmé seul dans le bloc ou avec d'autres codes G.

Syntaxe

```
MCALL <nom de programme>
```

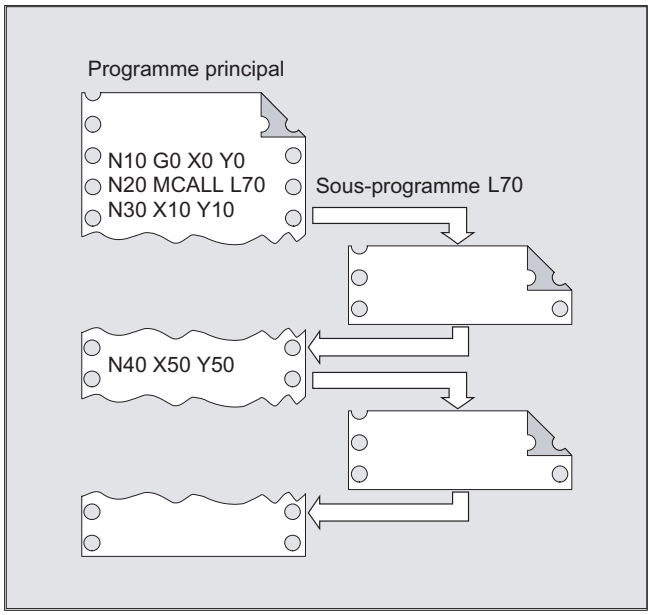
Signification

MCALL :	Instruction pour l'appel modal d'un sous-programme
<nom du programme> :	Nom du sous-programme

Exemples

Exemple 1 :

Code de programme	Commentaire
N10 G0 X0 Y0	
N20 MCALL L70	; Appel modal d'un sous-programme.
N30 X10 Y10	; La position programmée est accostée, puis le sous-programme L70 est exécuté.
N40 X50 Y50	; La position programmée est accostée, puis le sous-programme L70 est exécuté.



Exemple 2 :

Code de programme
N10 G0 X0 Y0
N20 MCALL L70
N30 L80

Dans cet exemple, les blocs CN ci-après figurent avec des axes à interpolation programmés dans le sous-programme L80. L70 est appelé par L80.

1.25.3.5 Appel indirect de sous-programme (CALL)

Fonction

Dans certaines conditions, il peut être nécessaire de faire appel à différents sous-programmes à un endroit donné du programme. Pour ce faire, indiquer le nom du sous-programme dans une variable du type STRING. L'appel du sous-programme s'effectue avec `CALL` et le nom de la variable.

> **⚠ PRUDENCE**
>
> L'appel indirect de sous-programme ne s'adresse qu'à des sous-programmes sans transfert de paramètres. Pour l'appel direct d'un sous-programme, rangez le nom de ce dernier dans une constante de type STRING.

Syntaxe

```
CALL <nom_de_programme>
```

Signification

`CALL` :	Instruction pour l'appel indirect d'un sous-programme
`<nom du programme>` :	Nom du sous-programme (variable ou constante) Type : STRING

Exemple

Appel direct avec constante de type STRING :

```
Code de programme                                          Commentaire
…
CALL "/_N_WKS_DIR/_N_SUBPROG_WPD/_N_PIECE1_SPF"            ; Appeler directement le
                                                             sous-programme PIECE1 avec
                                                             CALL.
…
```

Appel indirect par le biais d'une variable :

```
Code de programme                          Commentaire
…
DEF STRING[100] NOM_PROG                   ; Définir une variable.
NOM_PROG="/_N_SCP_DIR/_N_SUBPROG_WPD       ; Affecter le sous-programme PIECE1 à la
/_N_PIECE1_SPF"                              variable NOM_PROG.
CALL NOM_PROG                              ; Appeler indirectement le sous-programme
                                             PIECE1 par le biais de CALL et de la
                                             variable NOM_PROG.
…
```

1.25.3.6 Appel indirect d'un sous-programme avec indication de la section de programme à exécuter (CALL BLOCK ... TO ...)

Fonction

Avec `CALL` et la combinaison de mots-clés `BLOCK ... TO`, un sous-programme est appelé indirectement et la section de programme repérée par repères de début et de fin est exécutée.

Syntaxe

```
CALL <nom de programme> BLOCK <repère de début> TO <repère de fin>
CALL BLOCK <repère de début> TO <repère de fin>
```

Signification

`CALL` :	Instruction pour l'appel indirect d'un sous-programme
`<nom du programme>` :	Nom du sous-programme (variable ou constante) contenant la section de programme à exécuter (**indication optionnelle**). Type : STRING **Nota :** Si aucun `<nom de programme>` n'est programmé, le système recherche et exécute la section de programme repérée par le `<repère de début>` et le `<repère de fin>` dans le programme actuel.
`BLOCK ... TO ...` :	Combinaison de mots-clés pour l'exécution indirecte d'une section de programme
`<repère de début>` :	Variable indiquant le début de la section de programme à exécuter. Type : STRING
`<repère de fin>` :	Variable indiquant la fin de la section de programme à exécuter. Type : STRING

Exemple

Programme principal :

```
Code de programme                                          Commentaire
...
DEF STRING[20] ETIQUETTE_DEBUT, ETIQUETTE_FIN          ; Définition des variables
                                                           pour le repère de début et
                                                           le repère de fin.
ETIQUETTE_DEBUT="ETIQUETTE_1"
ETIQUETTE_FIN="ETIQUETTE_2"
...
CALL "CONTUR_1" BLOCK STARTLABEL TO ENDLABEL           ; Appel indirect d'un sous-
                                                           programme et identification
                                                           de la section de programme à
                                                           exécuter.
...
```

Sous-programme :

```
Code de programme          Commentaire
PROC CONTOUR_1 ...
ETIQUETTE_1                ; Repère de début : début de l'exécution de la section de
                             programme
N1000 G1 ...
...
ETIQUETTE_2                ; Repère de fin : fin de l'exécution de la section de
                             programme
...
```

1.25.3.7 Appel indirect d'un programme programmé en langage ISO (ISOCALL)

Fonction

ISOCALL permet d'appeler de façon indirecte un programme programmé en langage ISO. L'appel active le mode ISO réglé dans les paramètres machine. Le mode d'usinage initial est rétabli à la fin du programme. En l'absence du mode ISO dans les paramètres machine, l'appel du sous-programme s'effectuera en mode Siemens.

Pour de plus amples informations sur le mode ISO, voir :
Bibliographie :
Description fonctionnelle Dialectes ISO

Syntaxe

```
ISOCALL <nom_programme>
```

Signification

ISOCALL :	Mot clé pour l'appel indirect du sous-programme activant le mode ISO réglé dans les paramètres machine
<nom du programme> :	Nom du programme programmé dans un langage ISO (variable ou constante de type STRING)

Exemple : Appel d'un contour avec programmation de cycles depuis le mode ISO

Code de programme	Commentaire
0122_SPF	; Description d'un contour en mode ISO
N1010 G1 X10 Z20	
N1020 X30 R5	
N1030 Z50 C10	
N1040 X50	
N1050 M99	
N0010 DEF STRING[5] NOM_PROG = "0122"	; (Cycle de) programme pièce Siemens
...	
N2000 R11 = $AA_IW[X]	
N2010 ISOCALL NOM_PROG	
N2020 R10 = R10+1	; Exécution du programme 0122.spf en mode ISO
...	
N2400 M30	

1.25.3.8 Appel de sous-programme avec indication de chemin et paramètres (PCALL)

Fonction

PCALL permet d'appeler des sous-programmes avec indication de chemin absolu et transfert de paramètres.

Syntaxe

```
PCALL <chemin/nom de programme>(<paramètre 1>, …,<paramètre n>)
```

Signification

PCALL :	Mot-clé pour l'appel d'un sous-programme avec indication de chemin absolu.
<chemin/nom du programme> :	Indication du chemin absolu débutant par "/", y compris nom du sous-programme. Si aucune indication de chemin absolu n'est faite, PCALL se comporte comme un appel standard de sous-programme avec descripteur de programme. Le descripteur de programme est indiqué sans texte préliminaire _N_ et sans extension. Si le nom du programme doit être programmé avec texte préliminaire et extension, il convient de le déclarer de manière explicite avec texte préliminaire et extension comme instruction EXTERN.
<paramètre 1>, ... :	Paramètres courants selon l'instruction PROC du sous-programme.

Exemple

Code de programme
PCALL/_N_WKS_DIR/_N_ARBRE_WPD/ARBRE(paramètre1,paramètre2, …)

1.25.3.9 Extension du chemin de recherche pour l'appel de sous-programmes (CALLPATH)

Fonction

L'instruction `CALLPATH` permet d'étendre le chemin de recherche pour l'appel de sous-programmes.

Ainsi il est également possible d'appeler des sous-programmes d'un répertoire pièce qui n'est pas sélectionné, sans indication complète du chemin absolu du sous-programme.

L'extension du chemin de recherche s'effectue avant l'entrée des cycles utilisateur (_N_CUS_DIR).

Les événements suivants annulent de nouveau l'extension du chemin de recherche :

- `CALLPATH` avec espace
- `CALLPATH` sans paramètre
- Fin de programme pièce
- Reset

Syntaxe

```
CALLPATH("<chemin>")
```

Signification

`CALLPATH` :	Mot clé pour l'extension programmable du chemin de recherche. Programmation dans une ligne spécifique du programme pièce.
`<chemin>` :	Constante ou variable de type STRING contenant le chemin absolu d'un répertoire pour l'extension du chemin de recherche. Le chemin commence par "/". L'entrée du chemin doit être complète, avec les préfixes et les suffixes. La longueur maximale du chemin est de 128 octets. Un <nom de chemin> contenant un espace ou un appel de `CALLPATH` sans paramètre annule de nouveau l'instruction du chemin de recherche.

Exemple

```
Code de programme
CALLPATH ("/_N_WKS_DIR/_N_MYWPD_WPD")
```

Le réglage du chemin de recherche est alors le suivant (le point 5 est nouveau) :

1. Répertoire courant/descripteur du sous-programme
2. Répertoire courant/descripteur du sous-programme_SPF
3. Répertoire courant/descripteur du sous-programme_MPF
4. /_N_SPF_DIR/descripteur du sous-programme_SPF
5. **/_N_WKS_DIR/_N_MYWPD/descripteur du sous-programme_SPF**
6. /N_CUS_DIR/_N_MYWPD/descripteur du sous-programme_SPF
7. /_N_CMA_DIR/descripteur du sous-programme_SPF
8. /_N_CST_DIR/descripteur du sous-programme_SPF

Conditions marginales

- CALLPATH vérifie l'existence réelle du chemin programmé. En cas d'erreur, l'exécution du programme pièce est annulée avec l'alarme de correcteur 14009.
- CALLPATH peut également être programmé dans les fichiers INI. L'instruction est alors active pendant la durée d'exécution du fichier INI (fichier WPD-INI ou programme d'initialisation de données actives CN, par exemple frames dans le 1er canal _N_CH1_UFR_INI). Puis le chemin de recherche est de nouveau annulé.

1.25.3.10 Exécution de sous-programme externe (EXTCALL)

Fonction

L'instruction EXTCALL permet de recharger et d'exécuter un sous-programme à partir d'une mémoire programme externe (lecteur local, lecteur réseau, lecteur USB).

Le chemin vers le répertoire de sous-programmes externes peut être préréglé par la donnée de réglage :

SD42700 $SC_EXT_PROG_PATH

Avec le chemin vers le sous-programme ou le descripteur spécifié lors de l'appel EXTCALL on obtient le chemin global du programme a appeler.

Remarque

Les sous-programmes externes ne doivent pas contenir d'instructions de saut telles que GOTOF, GOTOB, CASE, FOR, LOOP, WHILE ou REPEAT.

Les structures IF-ELSE-ENDIF sont possibles.

Les appels de sous-programme et des appels EXTCALL imbriqués sont possibles.

Syntaxe

```
EXTCALL("<chemin/><nom de programme>")
```

Signification

`EXTCALL` :	Instruction d'appel d'un sous-programme externe
`''<chemin/><nom du programme>''` :	Constante/variable de type STRING
	`<chemin/>` : Indication absolue ou relative du chemin (**facultatif**)
	`<nom du programme>` : le nom du programme est spécifié sans le préfixe "_N_".
	L'extension de fichier ("MPF", "SPF") peut être ajoutée au nom du programme avec le caractère "_" ou "." (**facultatif**).
	Exemple : `"ARBRE"` ou `"ARBRE_SPF"` ou `"ARBRE.SPF"`

Remarque

Spécification du chemin : désignations abrégées

Les désignations abrégées suivantes peuvent être utilisées lors de la spécification du chemin :

- **LOCAL_DRIVE :** pour lecteur local
- **CF_CARD :** pour carte CompactFlash
- **USB :** pour connexion USB en face avant

CF_CARD : et **LOCAL_DRIVE :** peuvent être utilisés de façon alternative.

Remarque

Exécution d'un programe externe via lecteur USB

Le transfert de programmes externes à partir d'un lecteur USB externe via une interface USB ne doit être effectué que par le biais de l'interface X203 avec le nom "TCU_1".

IMPORTANT

Exécution d'un programe externe via clé USB (sur connexion USB en face avant)

L'exécution directe depuis une clé USB est déconseillée.

Il n'existe aucune protection contre les problèmes de contact, la déconnexion ou le retrait accidentel de la clé USB pendant le fonctionnement.

Une déconnexion pendant l'usinage d'une pièce provoque un arrêt immédiat de l'usinage, la pièce étant par conséquent endommagée.

Exemple

Exécution depuis le lecteur local

Programme principal :

Code de programme

```
N010 PROC MAIN
N020 ...
N030 EXTCALL ("EBAUCHE")
N040 ...
N050 M30
```

Sous-programme externe :

Code de programme

```
N010 PROC EBAUCHE
N020 G1 F1000
N030 X= ... Y= ... Z= ...
N040 ...
...
...
N999999 M17
```

Le programme principal "MAIN.MPF" se trouve dans la mémoire CN et est sélectionné pour être exécuté.

Le sous-programme à recharger "EBAUCHE.SPF" ou "EBAUCHE.MPF" se trouve sur le lecteur local dans le répertoire "/user/sinumerik/data/prog/WKS.DIR/WST1.WPD".

Le chemin vers le sous-programme est préréglé dans le SD42700 :

SD42700 $SC_EXT_PROG_PATH = "LOCAL_DRIVE:WKS.DIR/WST1.WPD"

Remarque

Sans spécification du chemin dans le SD42700, l'instruction `EXTCALL` pour cet exemple devrait être programmée comme suit :

```
EXTCALL("LOCAL_DRIVE:WKS.DIR/WST1.WPD/EBAUCHE")
```

Informations complémentaires

Appel EXTCALL avec chemin absolu

Si le sous-programme existe sous le chemin spécifié, il est exécuté par l'appel EXTCALL. S'il n'existe pas, l'exécution du programme est abandonnée.

Appel EXTCALL avec chemin relatif / sans chemin

Lors d'un appel EXTCALL avec chemin relatif ou sans chemin, les mémoires de programme présentes sont parcourues selon le modèle suivant :

- Lorsqu'un chemin par défaut est spécifié dans SD42700 $SC_EXT_PROG_PATH, la recherche s'effectue, dans un premier temps, à partir de cet endroit à l'aide de l'indication fournie dans l'appel EXTCALL (nom du programme avec chemin relatif, le cas échéant). Le chemin absolu est déterminé alors par concaténation de chaînes :
  - le chemin préréglé dans le SD42700
  - le caractère "/" comme caractère de séparation
  - et du chemin du sous-programme ou du descripteur du sous-programme, indiqué dans EXTCALL.
- Si le sous-programme appelé n'est pas trouvé sous le chemin préréglé, la recherche s'effectue dans les répertoires du disque de l'utilisateur selon les indications de l'appel EXTCALL.
- La recherche est terminée lorsque le sous-programme est trouvé. Si la recherche ne donne aucun résultat, le programme est abandonné.

Mémoire de chargement réglable (tampon FIFO)

Pour l'exécution d'un programme en mode "Exécution du programme externe" (programme principal ou sous-programme), une mémoire de chargement est nécessaire dans le NCK. La taille de la mémoire de chargement est préréglée sur 30 Ko et peut être modifiée, comme d'autres paramètres machine concernant la mémoire, que par le constructeur de machine-outil en cas de besoin.

Un tampon de rechargement doit être réglé pour chacun des programmes (programmes principaux ou sous-programmes) exécutés simultanément en mode "Exécution d'un programme externe".

RESET, POWER ON

Avec RESET et POWER ON, les appels de sous-programme externes sont interrompus et les mémoires de chargement correspondantes supprimées.

Un sous-programme sélectionné pour "Exécution d'un sous-programme externe" reste sélectionné après un RESET / une fin du programme pièce. La sélection est perdue après POWER ON.

Bibliographie

Pour plus d'informations sur l'"Exécution d'un programme externe", se référer à la :

Description fonctionnelle Fonctions de base ; GMFC, Canal, Mode de programme, Comportement au reset (K1)

1.25.4 Cycles

1.25.4.1 Cycles : paramétrage de cycles utilisateur

Fonction

Des cycles utilisateur peuvent être paramétrés à l'aide des fichiers cov.com et uc.com.

Le fichier cov.com est livré avec des cycles standard et peut être étendu. Le fichier uc.com doit être créé par l'utilisateur.

Ces deux fichiers doivent être chargés dans le système passif de fichiers, dans le répertoire "Cycles utilisateur" ou pourvus de l'indication correspondante de chemin :

```
;$PATH=/_N_CUS_DIR
```

dans le programme.

Fichiers et chemins d'accès

`cov.com_COM`	Liste des cycles
`uc.com`	Description des appels de cycle

Adaptation de cov.com – liste des cycles

Le fichier cov.com fourni avec les cycles standard a la structure suivante :

`%_N_COV_COM`	Nom du fichier
`;$PATH=/_N_CST_DIR`	Indication du chemin
`;Vxxx 11.12.95 Sca Liste des cycles`	Ligne de commentaires
`C1(CYCLE81) Perçage, centrage`	Appel du 1er cycle
`C2(CYCLE82) Perçage, lamage`	Appel du 2ème cycle
`...`	
`C24(CYCLE98) Concaténation de filetages`	Appel du dernier cycle
`M17`	Fin de fichier

Syntaxe

Pour tout nouveau cycle à ajouter, il y a lieu d'insérer une ligne dont la syntaxe est la suivante :

```
C<Numéro> (<Nom du cycle>) Texte de commentaire
```

Numéro : un nombre entier quelconque qui n'a pas encore été utilisé dans le fichier ;

Nom du cycle : le nom du cycle à insérer

Texte de commentaire : un commentaire au choix, relatif au cycle

Exemple :

```
C25 (MON_CYCLE_1) Cycle_utilisateur_1
C26 (CYCLE_SPECIAL)
```

Exemple de fichier uc.com - Description des cycles utilisateur

L'explication qui suit est le développement de l'exemple précédent :
Pour les deux cycles suivants, un nouveau paramétrage doit être créé :

```
Programmation          Commentaire
PROC MON_CYCLE_1 (REAL PAR1, INT PAR2, CHAR PAR3, STRING[10] PAR4)
Le cycle contient les paramètres suivants :
PAR1 :               ; Réel compris dans la plage -1000.001 <= PAR2 <= 123.456,
                       préréglage 100
PAR2 :               ; Nombre entier positif compris dans la plage 0 <= PAR3 <=
                       999999, préréglage 0
PAR3 :               ; 1 caractère ASCII
PAR4 :               ; Chaîne de caractères de longueur 10 pour un nom de sous-
                       programme
...
M17                  ;
```

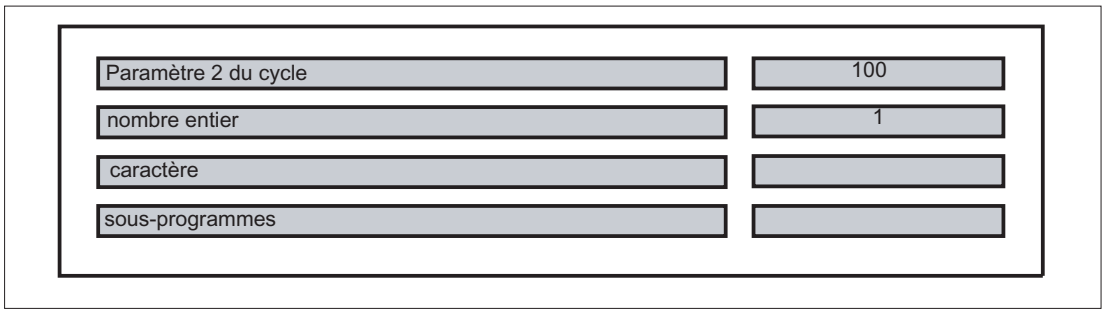
```
Programmation        Commentaire
PROC CYCLE_SPECIAL (REAL VALEUR1, INT VALEUR2)
Le cycle contient les paramètres suivants :
VALEUR1 :            ; réel sans limitation de plage de valeurs ni préréglage
VALEUR2 :            ; Nombre entier sans limitation de plage de valeurs ni
                       préréglage
...
M17                  ;
```

Fichier uc.com correspondant :

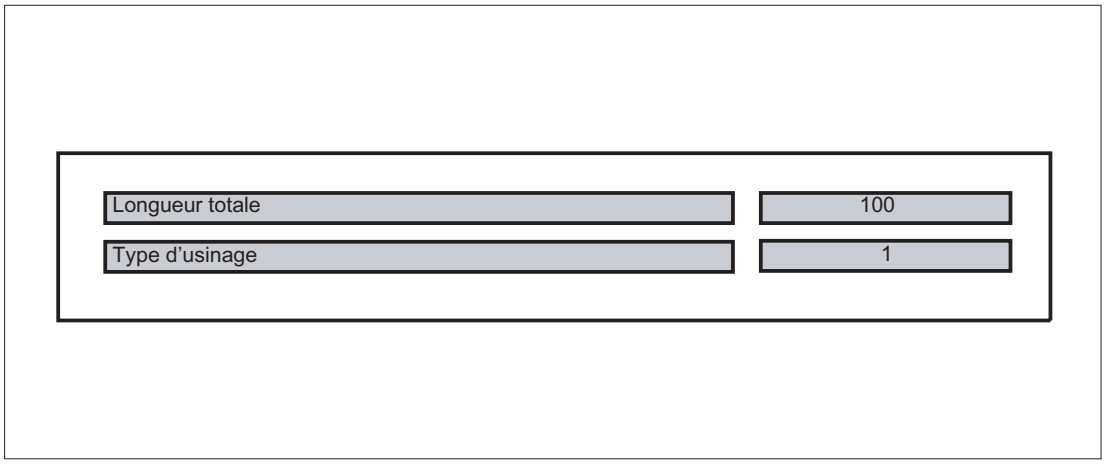
```
Programmation
%_N_UC_COM
;$PATH=/_N_CUS_DIR
//C25(MON_CYCLE_1) Cycle_utilisateur_1
(R/-1000.001 123.456 / 100 /paramètre_2 du cycle)
(I/0 999999 / 1 / nombre entier)
(C//"A" / caractère)
(S///nom du sous-programme)
//C26(CYCLE_SPECIAL)
(R///longueur totale)
(I/*123456/3/type d'usinage)
M17
```

Exemple de deux cycles

Masque d'affichage pour le cycle `MON_CYCLE_1`



Masque d'affichage pour le cycle `CYCLE_SPECIAL`



Description de la syntaxe pour le fichier uc.com - Description des cycles utilisateur

Ligne d'en-tête propre à chaque cycle :
identique au fichier cov.com, précédée de "//"

```
//C <Numéro> (<Nom du cycle>) Texte de commentaire
```

Exemple :

```
//C25 (MON_CYCLE_1) Cycle_utilisateur_1
```

Ligne de description pour chaque paramètre :

```
(<Identificateur pour type de donnée> / <Valeur minimale> <Valeur maximale>
/ <Préréglage> /<Commentaire>)
```

Identificateur pour type de donnée :

R	pour Real
I	pour Integer
C	pour un caractère
S	pour une chaîne de caractères

valeur minimale, maximale (facultatif)

Valeurs limites de la valeur à introduire contrôlées lors de l'introduction ; les valeurs en dehors de cette plage ne peuvent pas être introduites. On peut aussi indiquer des valeurs de sélection auxquelles on pourra accéder avec la touché de basculement ; celles-ci seront scrutées en commençant par "*", d'autres valeurs ne sont alors pas autorisées.

Exemple :

```
(I/*123456/1/type d'usinage)
```

Pour les types Chaîne de caractères et Caractère, il n'y a pas de limite.

Préréglage (facultatif)

C'est la valeur qui est préréglée dans le masque correspondant lors de l'appel du cycle ; elle peut être modifiée par l'opérateur.

Commentaire

C'est un texte comportant 50 caractères au maximum qui est affiché dans le masque d'appel du cycle, devant le champ d'introduction du paramètre.

1.26 Macroprogrammation (DEFINE ... AS)

⚠ PRUDENCE
Les macro-instructions peuvent entraîner une forte modification du langage de programmation de la CN ! Par conséquent, utiliser les macro-instructions avec précaution !

Fonction

Une macro-instruction réunit différentes instructions en une nouvelle instruction globale qui a son propre nom. Des fonctions G, M et H et des sous-programmes L peuvent également être définis sous forme de macro-instructions. En appelant la macro-instruction pendant l'exécution du programme, les instructions programmées dans la macro-instruction sont exécutées l'une après l'autre.

Application

Les suites d'instructions qui reviennent souvent sont programmées en une seule fois sous la forme d'une macro-instruction dans un bloc spécifique (fichier de macro-instructions) ou bien une seule fois en début de programme. La macro-instruction peut alors être appelée et exécutée dans tout programme principal ou sous-programme.

Activation

Pour pouvoir utiliser les macro-instructions d'un fichier de macro-instructions dans le programme CN, il faut charger ce fichier dans la CN.

Syntaxe

Définition d'une macro-instruction :

```
DEFINE <non de macro> AS <instruction 1> <instruction 2> ...
```

Appel dans le programme CN :

```
<Nom de macro>
```

Signification

`DEFINE ... AS` :	Combinaison de mots clés pour la définition d'une macro-instruction
`<nom de macro>` :	Nom de la macro-instruction
	Seuls des descripteurs sont autorisés comme noms de macro-instructions.
	Le nom de la macro-instruction permet de l'appeler depuis le programme CN.
`<instruction>` :	Instruction de programmation devant figurer dans la macro-instruction.

Règles de définition des macro-instructions

- Dans une macro-instruction, vous pouvez définir des descripteurs, des fonctions G, M, H et des noms de programme L quelconques.
- Les macro-instructions peuvent également être définies dans le programme CN.
- Des macro-instructions de fonctions G ne peuvent être définies qu'au niveau de la CN, dans le bloc de macro-instructions.
- Les fonctions H et L sont programmables à deux chiffres.
- Les fonctions M et G sont programmables à trois chiffres.

⚠ PRUDENCE

Les mots-clés et les noms réservés ne doivent pas être définis en surnombre avec des macro-instructions.

Conditions marginales

L'imbrication de macro-instructions n'est pas possible.

Exemples

Exemple 1 : Macro-instruction définie en début de programme

Code de programme	Commentaire
`DEFINE LIGNE AS G1 G94 F300`	`; Définition de la macro-instruction`
`...`	
`...`	
`N70 LINIE X10 Y20`	`; Appel de la macro-instruction`
`...`	

Exemple 2 : Définitions de macro-instruction dans un fichier de macro-instructions

Code de programme	Commentaire
`DEFINE M6 AS L6`	`; Lors du changement d'outil, un sous-programme qui se charge du transfert de données nécessaire est appelé. La fonction M de changement d'outil (par exemple M106) proprement dite est sortie dans le sous-programme.`
`DEFINE G81 AS DRILL(81)`	`; Redéfinition de la fonction G-DIN.`
`DEFINE G33 AS M333 G333`	`; La synchronisation avec l'AP est requise pour le filetage. La fonction G33 initiale a été renommée en G333 par un PM, si bien que la programmation ne change pas pour l'utilisateur.`

Exemple 3 : Fichier externe de macro-instructions

Après le chargement du fichier externe de macro-instructions dans la commande, il est nécessaire de charger ce fichier dans la CN. Les macro-instructions ne peuvent être utilisées dans le programme CN qu'ensuite.

```
Code de programme                Commentaire
%_N_UMAC_DEF
;$PATH=/_N_DEF_DIR              ; Macro-instructions personnalisées
DEFINE PI AS 3.14
DEFINE TC1 AS M3 S1000
DEFINE M13 AS M3 M7             ; Broche sens horaire, arrosage en marche
DEFINE M14 AS M4 M7             ; Broche sens antihoraire, arrosage en marche
DEFINE M15 AS M5 M9             ; Arrêt broche, arrêt arrosage
DEFINE M6 AS L6                 ; Appel du programme de changement d'outil
DEFINE G80 AS MCALL             ; Désactivation du cycle de perçage
M30
```

Gestion des fichiers et programmes 2

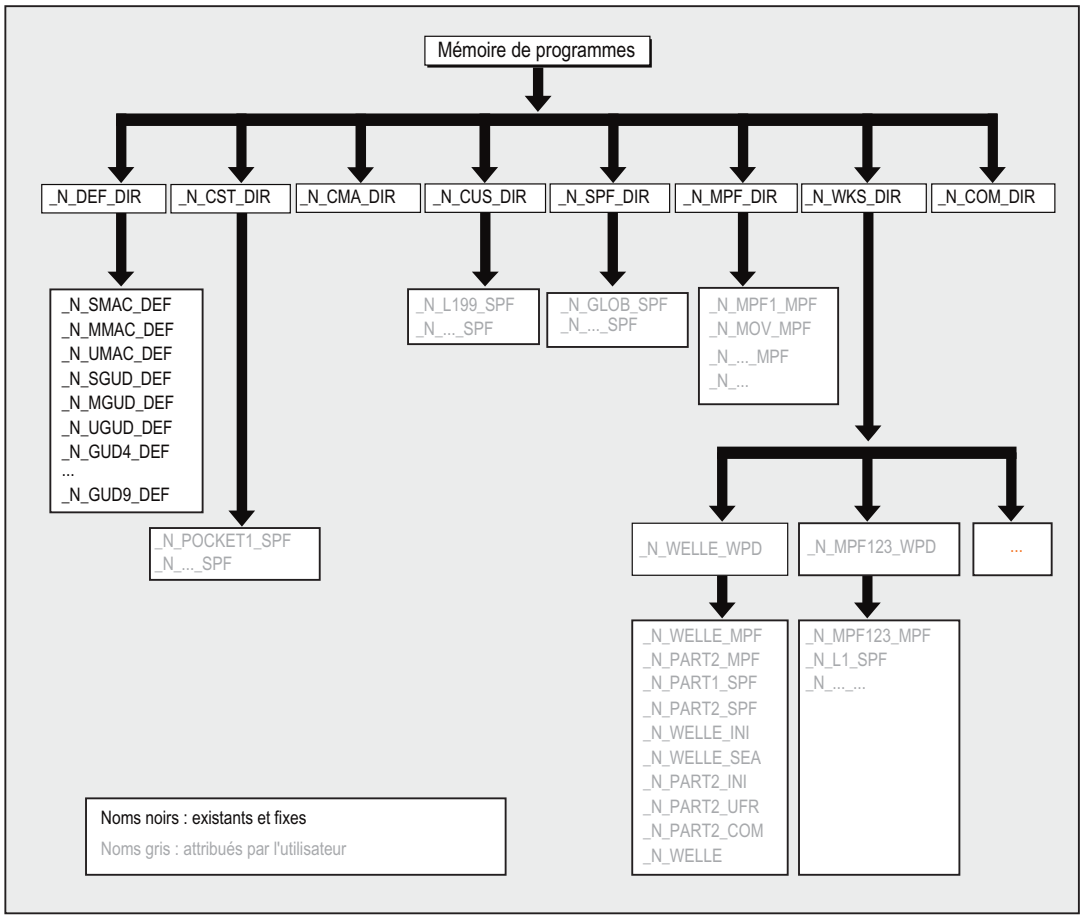
2.1 Mémoire de programmes

Fonction

Les fichiers et programmes (p. ex. programmes principaux et sous-programmes, définitions de macros) sont stockés de manière permanente dans la mémoire de programme (→ système de fichiers passif).

Bibliographie :
Description fonctionnelle Fonctions d'extension ; configuration de la mémoire (S7)

A coté de ceux-ci, il y a un certain nombre de types de fichiers qui peuvent y être stockés de façon provisoire pour être transférés, sur demande (par exemple lors de la réalisation d'une pièce déterminée) dans la mémoire de travail (par exemple pour des raisons d'initialisation).



Répertoires standard

Les répertoires suivants sont créés en version standard :

Répertoire	Contenu
_N_DEF_DIR	Blocs de données et blocs de macros
_N_CST_DIR	Cycles standard
_N_CMA_DIR	Cycles constructeur
_N_CUS_DIR	Cycles utilisateur
_N_WKS_DIR	Pièces
_N_SPF_DIR	Sous-programmes globaux
_N_MPF_DIR	Programmes principaux
_N_COM_DIR	Commentaires

Types de fichiers

Les types de fichiers suivants peuvent être introduits dans la mémoire de programmes :

Type de fichier	Description
nom_MPF	Programme principal
nom_SPF	Sous-programme
Nom_TEA	Paramètres machine
Nom_SEA	Données de réglage
Nom_TOA	Corrections d'outils
Nom_UFR	Décalages d'origine / frames
Nom_INI	Fichier d'initialisation
Nom_GUD	Données utilisateur globales
Nom_RPA	Paramètres R
Nom_COM	Commentaire
Nom_DEF	Définitions pour données utilisateur globales et macro-instructions

Catalogue des pièces (_N_WKS_DIR)

Le catalogue des pièces existe en version standard dans la mémoire de programmes sous la désignation `_N_WKS_DIR`. Il contient les répertoires pièce pour toutes les pièces que vous avez programmées.

Catalogues des pièces (..._WPD)

Pour une gestion plus aisée des données et des programmes, il est possible de regrouper certaines données et certains programmes ou de les ranger dans des répertoires pièce distincts.

Un répertoire pièce contient tous les fichiers nécessaires pour l'usinage d'une pièce. Ceux-ci peuvent être des programmes principaux, des sous-programmes, tous programmes d'initialisation et fichiers de commentaires.

Les programmes d'initialisation sont exécutés une seule fois après la sélection du programme avec le premier lancement du programme pièce (selon le paramètre machine PM11280 $MN_WPD_INI_MODE).

Exemple :

Le répertoire pièce _N_ARBRE_WPD, qui a été créé pour la pièce ARBRE, contient les fichiers suivants :

Fichier	Description
_N_ARBRE_MPF	Programme principal
_N_PART2_MPF	Programme principal
_N_PART1_SPF	Sous-programme
_N_PART2_SPF	Sous-programme
_N_ARBRE_INI	Programme général d'initialisation des données pour la pièce
_N_ARBRE_SEA	Programme d'initialisation des données de réglage
_N_PART2_INI	Programme général d'initialisation des données pour le programme partie 2
_N_PART2_UFR	Programme d'initialisation des données frame pour le programme { partie 2
_N_ARBRE_COM	Fichier de commentaires

Création de répertoires pièce sur un PC externe

La procédure décrite ci-dessous est exécutée à un ordinateur personnel externe. Vous trouverez toutes les informations utiles pour gérer les fichiers et les programmes (du PC à la commande) à partir de la commande, dans votre manuel d'utilisation.

Création de catalogue de pièces avec le chemin ($PATH=…)

Dans la deuxième ligne du fichier, vous spécifiez le chemin de destination sous `$PATH=…`. Le fichier est ensuite rangé sous le chemin indiqué.

Exemple :

Code de programme

```
%_N_ARBRE_MPF
; $PATH=/_N_WKS_DIR/_N_ARBRE_WPD
N10 G0 X… Z…
...
M2
```

Le fichier _N_ARBRE_MPF est créée dans le répertoire /_N_WKS_DIR/_N_ARBRE_WPD.

Création de catalogue de pièces sans chemin

Sans indication de chemin, les fichiers avec l'extension _SPF sont créés dans le répertoire /_N_SPF_DIR, les fichiers avec l'extension _INI dans le répertoire de travail et tous les autres fichiers dans le répertoire /_N_MPF_DIR.

Exemple :

Code de programme

```
%_N_ARBRE_SPF
...
M17
```

Le fichier _N_ARBRE_SPF est créé dans le répertoire /_N_SPF_DIR.

Sélection d'une pièce pour exécution

Un répertoire pièce peut être sélectionné dans un canal en vue de son exécution. S'il existe dans ce répertoire un programme principal **de même nom** ou un seul programme principal (_MPF), il sera sélectionné automatiquement pour l'exécution.

Bibliographie :
/BAD/ Manuel d'utilisation HMI Advanced ; Chapitres "Liste de tâches" et "Sélectionner un programme pour être exécuté"

Chemin de recherche pour l'appel de sous-programmes

Si le chemin d'accès n'est pas indiqué de façon explicite dans le programme pièce lors de l'appel d'un sous-programme (ou d'un fichier d'initialisation), la recherche a lieu selon un chemin prédéfini.

Appel du sous-programme avec chemin absolu

Exemple :

Code de programme

```
...
CALL"/_N_CST_DIR/_N_CYCLE1_SPF"
...
```

Appel du sous-programme sans chemin absolu

En règle générale, les programmes sont appelés sans indication de chemin.

Exemple :

Code de programme

```
...
CYCLE1
...
```

Selon le programme appelé, les répertoires sont parcourus dans l'ordre suivant :

N°	Répertoire	Description
1	répertoire courant / *nom*	Répertoire pièce ou répertoire standard _N_MPF_DIR
2	répertoire courant / *nom_SPF*	
3	répertoire courant / *nom_MPF*	
4	/_N_SPF_DIR / *nom_SPF*	Sous-programmes globaux
5	/_N_CUS_DIR / *nom_SPF*	Cycles utilisateur
6	/_N_CMA_DIR / *nom_SPF*	Cycles constructeur
7	/_N_CST_DIR / *nom_SPF*	Cycles standard

Programmation des chemins de recherche pour l'appel de sous-programmes (CALLPATH)

L'extension du chemin de recherche pour l'appel de sous-programmes se fait avec l'instruction `CALLPATH`.

Exemple :

Code de programme

```
CALLPATH ("/_N_WKS_DIR/_N_MYWPD_WPD")
...
```

Le chemin de recherche est consigné avant la position 5 (cycle utilisateur) conformément à la programmation indiquée.

Pour de plus amples informations sur le chemin de recherche programmable pour l'appel de sous-programmes avec `CALLPATH`, voir chapitre "Extension du chemin de recherche pour l'appel de sous-programmes avec CALLPATH".

2.2 Mémoire de travail (CHANDATA, COMPLETE, INITIAL)

Fonction

La mémoire de travail contient les données système et les données utilisateur courantes qui permettent à la commande de fonctionner (système actif de fichiers). Par exemple :

- Paramètres machine actifs
- Données de correction d'outil
- Décalages d'origine
- ...

programmes d'initialisation

Les programmes d'initialisation servent à l'initialisation des données de la mémoire de travail. Les types de fichier suivants peuvent être utilisés :

Type de fichier	Description
Nom_TEA	Paramètres machine
Nom_SEA	Données de réglage
Nom_TOA	Corrections d'outils
Nom_UFR	Décalages d'origine / frames
Nom_INI	Fichier d'initialisation
Nom_GUD	Données utilisateur globales
Nom_RPA	Paramètres R

Le manuel d'utilisation de l'interface utilisateur contient des informations sur tous les types de fichier.

Zones de données

Vous pouvez intégrer les données dans différentes zones au sein desquelles elles doivent être valides. Ainsi, une commande peut disposer de plusieurs canaux ou, d'une manière générale, également de plusieurs axes.

La répartition est la suivante :

identificateur	Zones de données
NCK	Données spécifiques à NCK
CH<n>	Données spécifiques à un canal (<n> indique le numéro de canal)
AX<n>	Données spécifiques à un axe (<n> indique le numéro de l'axe machine)
TO	Données d'outil
COMPLETE	Toutes les données

Création d'un programme d'initialisation sur PC externe

A l'aide des identificateurs des zones de données et des types de données, vous pouvez déterminer les zones à considérer comme un ensemble lors de la sauvegarde des données :

_N_AX5_TEA_INI	Paramètres machine pour l'axe 5
_N_CH2_UFR_INI	Frames du canal 2
_N_COMPLETE_TEA_INI	Tous les paramètres machine

Après la mise en service de la commande, un enregistrement garantissant un fonctionnement correct de cette dernière est présent dans la mémoire de travail.

Procédure dans le cas des commandes multicanaux (CHANDATA)

`CHANDATA(<numéro de canal>)` pour plusieurs canaux est uniquement autorisé dans le fichier _N_INITIAL_INI. C'est le fichier de mise en service avec lequel toutes les données de la commande sont initialisées.

```
Code de programme                          Commentaire
%_N_INITIAL_INI
CHANDATA(1)
                                           ; Affectation des axes machine au canal 1 :
$MC_AXCONF_MACHAX_USED(0) = 1
$MC_AXCONF_MACHAX_USED[1]=2
$MC_AXCONF_MACHAX_USED[2]=3
CHANDATA(2)
                                           ; Affectation des axes machine au canal 2 :
$MC_AXCONF_MACHAX_USED[0]=4
$MC_AXCONF_MACHAX_USED[1]=5
CHANDATA(1)
                                           ; Paramètres machine axiaux :
                                           ; Fenêtre d'arrêt précis grossier :
$MA_STOP_LIMIT_COARSE[AX1]=0.2             ; Axe 1
$MA_STOP_LIMIT_COARSE[AX2]=0.2             ; Axe 2
                                           ; ; Fenêtre d'arrêt précis fin :
$MA_STOP_LIMIT_FINE[AX1]=0.01              ; Axe 1
$MA_STOP_LIMIT_FINE[AX1]=0.01              ; Axe 2
```

⚠ PRUDENCE

Instruction CHANDATA

Dans le programme pièce, l'instruction `CHANDATA` ne peut être utilisée que pour le canal dans lequel le programme est exécuté ; c'est-à-dire que l'instruction peut être utilisée pour protéger le programme CN contre l'exécution dans un canal non prévu à cet effet.

En cas d'erreur, l'exécution du programme est abandonnée.

Remarque

Les fichiers INI dans des listes de tâches ne contiennent pas d'instruction `CHANDATA`.

Sauvegarde des programmes d'initialisation (COMPLETE, INITIAL)

Les fichiers de la mémoire de travail peuvent être sauvegardés sur un PC externe, puis être rechargés depuis celui-ci.

- La sauvegarde des fichiers s'effectue avec `COMPLETE`.
- `INITIAL` génère un fichier INI (_N_INITIAL_INI) pour toutes les zones.

Chargement de programmes d'initialisation

IMPORTANT
Après le chargement du fichier "INITIAL_INI", tous les paramètres qui ne sont pas définis dans le fichier sont initialisés avec des paramètres par défaut, excepté les paramètres machine. Les **données de réglage, données d'outil, décalages d'origine, valeurs GUD, ...** sont donc réglés à des valeurs par défaut (normalement "ZERO").

Le fichier COMPLETE_TEA_INI convient par exemple au chargement de paramètres machine individuels. Dans ce fichier, la commande n'attend que des paramètres machine. Les autres zones de données restent donc inchangées dans ce cas.

Chargement de programmes d'initialisation

Les programmes INI peuvent également être sélectionnés et appelés comme programme pièce s'ils utilisent uniquement des données d'un canal. Il est ainsi également possible d'initialiser des données commandées par programme.

2.3 Instruction structurelle dans l'éditeur Step (SEFORM)

Fonction

L'instruction structurelle SEFORM est traitée dans l'éditeur Step (aide à la programmation avec l'éditeur) pour générer la vue séquentielle dans le logiciel HMI Advanced. La vue séquentielle améliore la lisibilité du sous-programme CN.

Syntaxe

```
SEFORM(<nom de section>,<niveau>,<icône>)
```

Signification

SEFORM()	Appel de l'instruction structurelle avec les paramètres <nom de section>, <niveau> et <icône>		
<Nom de section>	Descripteur de l'opération d'usinage		
	Type :	STRING	
<Niveau>	Indice du niveau principal ou du niveau inférieur		
	Type :	INT	
	Valeur :	0	Niveau principal
		1, ..., <n>	Niveau inférieur 1, ... , niveau inférieur <n>
<Icône>	Nom de l'icône qui doit être affichée pour cette section.		
	Type :	STRING	

Remarque

Les instructions SEFORM sont générées dans l'éditeur Step.

La chaîne de caractères transférée avec le paramètre <nom de section> est rangée dans la variable OPI en synchronisation avec l'exécution des blocs, de manière analogue à l'instruction MSG. L'information persiste jusqu'à son écrasement par l'instruction SEFROM suivante. Le contenu est effacé par un reset et par une fin de programme pièce.

Les paramètres <niveau> et <icône> sont vérifiés lors de l'exécution du programme pièce, mais ils ne sont pas utilisés.

Bibliographie

Pour d'autres informations sur l'aide à la programmation avec l'éditeur, voir :
Manuel d'utilisation HMI Advanced

Zones de protection 3

3.1 Définition des zones de protection (CPROTDEF, NPROTDEF)

Fonction

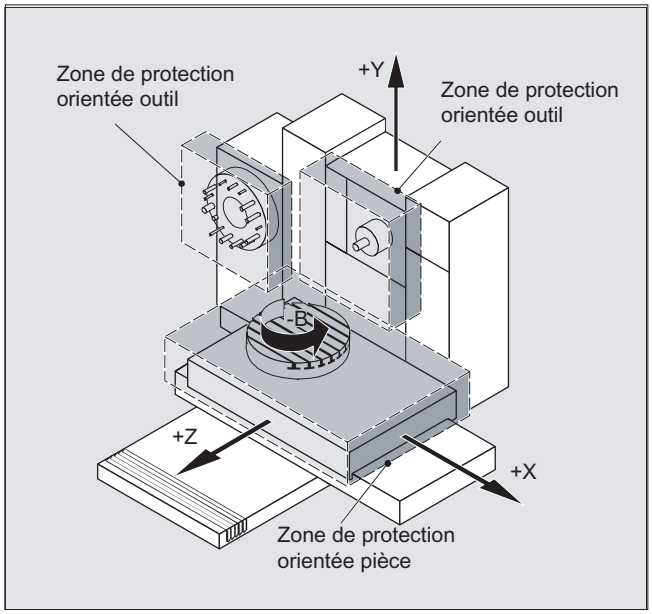
Grâce aux zones de protection, il est possible de protéger différents éléments de la machine, l'outillage et la pièce contre les déplacements erronés.

Zones de protection orientées outil :

Pour les éléments qui font partie de l'outil (outil, porte-outil, etc.).

Zones de protection orientées pièce :

Pour les éléments qui font partie de la pièce (parties de la pièce, table porte-pièce, pinces, mandrins, contre-poupées, etc.)



Syntaxe

```
DEF INT NOT_USED
G17/G18/G19
CPROTDEF/NPROTDEF(<n>,<t>,<applim>,<applus>,<appminus>)
G0/G1/... X/Y/Z...
...
EXECUTE (NOT_USED)
```

Signification

DEF INT NOT_USED :	Variable locale, définition du type de données INTEGER (entier) (cf. chapitre "Actions synchrones au déplacement (Page 551)")
G17/G18/G19 :	Le plan souhaité est sélectionné avant CPROTDEF ou NPROTDEF au moyen de G17/G18/G19 et ne doit pas être modifié avant EXECUTE. La programmation d'une cote (3ème dimension) entre CPROTDEF ou NPROTDEF et EXECUTE n'est pas autorisée.
CPROTDEF :	Définition des zones de protection spécifiques au canal (pour NCU 572/573 uniquement)
NPROTDEF :	Définition des zones de protection spécifiques à la machine
G0/G1/... X/Y/Z... ... :	On peut décrire le contour d'une zone de protection avec 11 déplacements au maximum dans le plan sélectionné. Le premier déplacement est constitué par l'accostage du contour. La zone de protection est le domaine se trouvant à gauche du contour. **Remarque :** Les déplacements figurant entre CPROTDEF ou NPROTDEF et EXECUTE ne sont pas exécutés mais ils définissent la zone de protection.
EXECUTE :	Fin de la définition
<n> :	Numéro de la zone de protection définie
<t> :	Type de la zone de protection TRUE : zone de protection orientée **outil** FALSE : zone de protection orientée **pièce**
<applim> :	Type de limitation dans la 3ème dimension 0: aucune limitation 1: limitation en sens plus 2: limitation en sens moins 3: limitation dans le sens positif et négatif
<applus> :	Valeur de la limitation dans le sens positif de la 3ème dimension
<appminus> :	Valeur de la limitation dans le sens négatif de la 3ème dimension
NOT_USED :	En présence de zones de protection, la variable d'erreur est sans effet avec EXECUTE

Contraintes

Aucune des fonctions suivantes ne doit être active lors de la définition des zones de protection :

- correction du rayon de la fraise ou du tranchant,
- transformation,
- frame.

Aucun accostage du point de référence (G74), aucun accostage de point fixe (G75), aucun arrêt de recherche de blocs, ni aucune fin de programme ne doit être programmé.

Informations complémentaires

Définition des zones de protection

Font partie de la définition des zones de protection :

- `CPROTDEF` pour les zones de protection spécifiques au canal
- `NPROTDEF` pour les zones de protection spécifiques à la machine
- la description du contour de la zone de protection
- la clôture de la définition avec `EXECUTE`

Lors de l'activation de la zone de protection dans le programme pièce CN, on peut déplacer, de façon relative, le point de référence de la zone de protection.

Point de référence de la description du contour

Les zones de protection orientées pièce sont définies dans le système de coordonnées de base.

Les zones de protection orientées outil sont définies par rapport au point de référence F de l'organe porte-outil.

Eléments de contour admissibles

Sont autorisés pour la description du contour de la zone de protection :

- `G0` et `G1` pour les parties rectilignes du contour
- `G2` pour les arcs de cercle dans le sens horaire (seulement pour les zones de protection orientées pièce)
- `G3` pour les arcs de cercle dans le sens antihoraire

Remarque

Si la zone de protection doit être définie par un cercle complet, il faut subdiviser ce dernier en deux cercles partiels. Les séquences `G2`, `G3` et `G3`, `G2` ne sont pas autorisées. Il faut, dans ce cas, insérer un bloc court avec `G1`.

Le dernier point et le premier point de la description du contour doivent coïncider.

Zones de protection extérieures

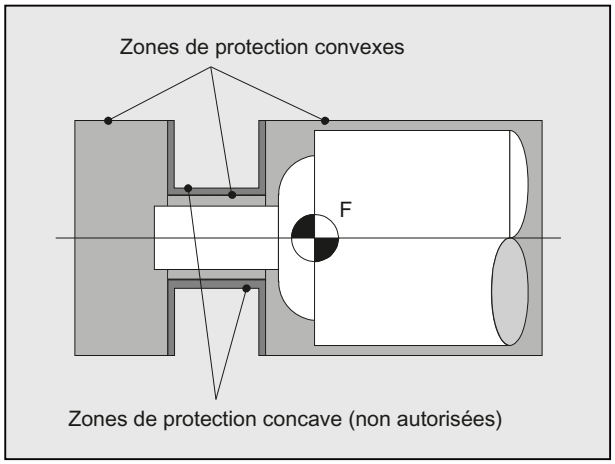
Les zones de protection extérieures (seulement les zones de protection orientées pièce) doivent être décrites dans le sens horaire.

Zones de protection à symétrie de révolution

Dans le cas de zones de protection à symétrie de révolution (par ex. pour un mandrin), l'ensemble du contour doit être décrit (et pas seulement jusqu'au centre de rotation !).

Zones de protection orientées outil

Les zones de protection orientées outil doivent toujours être convexes. S'il faut une zone de protection concave, il faut la fractionner en plusieurs zones de protection convexes.



3.2 Activation/désactivation des zones de protection (CPROT, NPROT)

Fonction

Préactiver ou activer les zones de protection définies précédemment pour assurer une surveillance anticollision ou désactiver les zones de protection actives.

Le nombre maximal de zones de protection qui peuvent être activées simultanément dans un même canal est fixé par un paramètre machine.

Si aucune zone de protection orientée outil n'est activée, la trajectoire de l'outil fera l'objet d'une surveillance rapportée aux zones de protection orientées pièce.

Remarque

Si aucune zone de protection orientée pièce n'est active, il n'y aura aucune surveillance de la trajectoire de l'outil.

Syntaxe

```
CPROT(<n>,<state>,<xMov>,<yMov>,<zMov>)
NPROT(<n>,<state>,<xMov>,<yMov>,<zMov>)
```

Signification

`CPROT` :	Appel de la zone de protection spécifique au canal (pour NCU 572/573 uniquement)
`NPROT` :	Appel de la zone de protection spécifique à la machine
`<n>` :	Numéro de la zone de protection
`<state>` :	Indication de l'état 0: Désactivation de la zone de protection 1: Préactivation de la zone de protection 2: Activation de la zone de protection 3: Préactivation de la zone de protection avec arrêt conditionnel
`<xMov>,<yMov>,<zMov>` :	Décaler dans les axes géométriques la zone de protection qui a été définie

Contraintes

Surveillance de zone de protection avec correction de rayon d'outil active

Lorsque la correction de rayon d'outil est active, une surveillance de zone de protection opérationnelle n'est possible que si le plan de la correction de rayon d'outil est identique au plan des définitions de zone de protection.

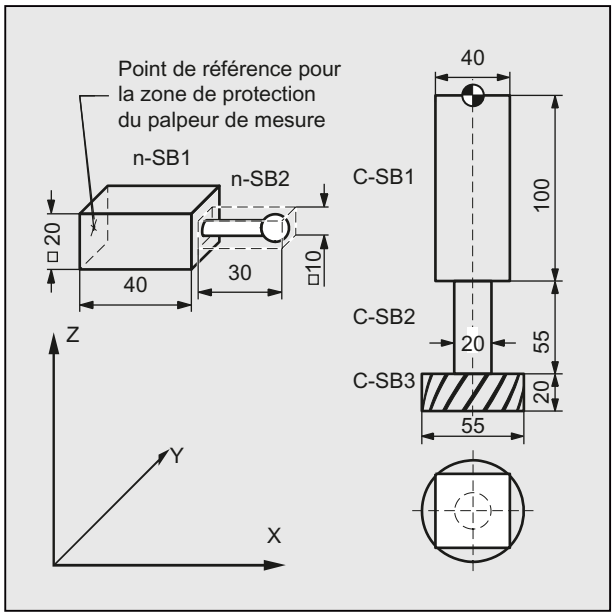
Exemple

Sur une fraiseuse, il convient d'éviter une collision entre la fraise et le palpeur de mesure. La position du palpeur doit être indiquée par un décalage lors de l'activation. A cet effet, on définit les zones de protection suivantes :

- Une zone de protection spécifique à la machine et orientée pièce pour le support du palpeur (n-SB1) et pour le palpeur lui-même (n-SB2).
- Une zone de protection spécifique au canal et orientée outil pour l'organe porte-fraise (c-SB1), la queue de la fraise (c-SB2) et la fraise elle-même (c-SB3).

Toutes les zones de protection sont orientées en Z.

Le point de référence du palpeur de mesure doit se trouver en X = -120, Y = 60 et Z = 80 lors de l'activation.



Code de programme	Commentaire
DEF INT ZONEPROT	; Définition d'une variable auxiliaire
Définition des zones de protection G17	; Réglage de l'orientation
NPROTDEF(1,FALSE,3,10,-10)G01 X0 Y-10	; Zone de protection n-SB1
X40	
Y10	
X0	
Y-10	
EXECUTE(ZONEPROT)	
NPROTDEF(2,FALSE,3,5,-5)	; Zone de protection n-SB2
G01 X40 Y-5	
X70	
Y5	
X40	
Y-5	

Code de programme	**Commentaire**
EXECUTE(ZONEPROT)	
CPROTDEF(1,TRUE,3,0,-100)	; Zone de protection c-SB1
G01 X-20 Y-20	
X20	
Y20	
X-20	
Y-20	
EXECUTE(ZONEPROT)	
CPROTDEF(2,TRUE,3,-100,-150)	; Zone de protection c-SB2
G01 X0 Y-10	
G03 X0 Y10 J10	
X0 Y-10 J-10	
EXECUTE(ZONEPROT)	
CPROTDEF(3,TRUE,3,-150,-170)	; Zone de protection c-SB3
G01 X0 Y-27,5	
G03 X0 Y27,5 J27,5	
X0 Y27,5 J-27,5	
EXECUTE(ZONEPROT)	
Activation des zones de protection :	
NPROT(1,2,-120,60,80)	; Activer zone de protection n-SB1 avec décalage
NPROT(2,2,-120,60,80)	; Activer zone de protection n-SB2 avec décalage
CPROT(1,2,0,0,0)	; Activer zone de protection c-SB1 avec décalage
CPROT(2,2,0,0,0)	; Activer zone de protection c-SB2 avec décalage
CPROT(3,2,0,0,0)	; Activer zone de protection c-SB3 avec décalage

Informations complémentaires

Etat d'activation (<state>)

- **<state>=2**

  Une zone de protection est activée en général dans un programme pièce avec <state> = 2.

  L'état est toujours spécifique au canal, même pour des zones de protection spécifiques à la machine.

- **<state>=1**

  Si la possibilité d'activation d'une zone de protection est prévue dans le programme AP, la préactivation nécessaire à cet effet se fait avec <state> = 1.

- **<state>=3**

  Lors de la préactivation avec arrêt conditionnel, aucun arrêt n'est en principe effectué avant une zone de protection dépassée, préactivée. L'arrêt se produit uniquement, lorsque la zone de protection a été activée. Ceci permet un usinage sans interruption lorsque les zones de protection ne sont activées que dans certains cas. Il faut toutefois tenir compte du fait que la rampe de freinage provoque, le cas échéant, un déplacement dans une zone de protection, si la zone de protection n'a été activée qu'immédiatement avant le déplacement.

  La préactivation avec arrêt conditionnel s'effectue avec <state> = 3.

- **<state>=0**

  La désactivation des zones de protection s'effectue avec <state> = 0. Aucun décalage n'est nécessaire à cet effet.

Décalage de zones de protection en cas de préactivation/activation

Le décalage peut être réalisé dans 1, 2 ou 3 dimensions. L'indication du décalage se rapporte :

- à l'origine machine dans le cas de zones de protection orientées pièce.
- au point de référence F de l'organe porte-outil dans le cas de zones de protection orientées outil.

État après le lancement

Les zones de protection peuvent être activées dès le lancement, après l'accostage du point de référence. Dans ce but, il convient de régler la variable système $SN_PA_ACTIV_IMMED[<n>] ou $SC_PA_ACTIV_IMMED[<n>] sur TRUE. Dans ce cas, les zones de protection sont activées avec <state> = 2 et ne sont pas décalées.

Activation multiple des zones de protection

Une zone de protection peut être activée dans plusieurs canaux à la fois (par ex. : un fourreau de contre-poupée quand deux chariots se font face). La surveillance des zones de protection se fait uniquement si la prise de référence a été effectuée dans tous les axes géométriques.

Tenez compte des remarques suivantes :

- Une zone de protection ne peut pas être activée simultanément avec différents décalages dans un même canal.
- Les zones de protection spécifiques à la machine doivent présenter la même orientation dans les deux canaux.

3.3 Contrôle des violations de zone de protection, des limitations de la zone de travail et des fins de course logiciels (CALCPOSI)

Fonction

La fonction CALCPOSI permet de vérifier si, à partir d'un point de départ donné, les axes géométriques peuvent parcourir un trajet prédéfini sans violer les limites d'axe (limites logicielles), les limitations de la zone de travail, ni les zones de protection.

Pour le cas où le trajet spécifié ne peut pas être parcouru, la valeur maximale autorisée est retournée.

La fonction CALCPOSI est un sous-programme prédéfini. Elle doit être isolée dans un bloc.

Syntaxe

```
Etat=CALCPOSI(_STARTPOS, _MOVDIST, _DLIMIT, _MAXDIST, _BASE_SYS, _TESTLIM)
```

Signification

`Etat`

0: Fonction en ordre,
le déplacement spécifié peut être entièrement exécuté.

–: dans _DLIMIT, au moins une composante est négative.

–: une erreur s'est produite dans un calcul de transformation.

Si le déplacement spécifié ne peut pas être entièrement exécuté, la valeur de retour est une valeur décimale positive et codée :

Position des unités (type de limite violée) :

1: déplacement limité par des fins de course logiciels.

2: déplacement limité par une limitation de la zone de travail.

3: déplacement limité par des zones de protection.

En cas de violation simultanée de plusieurs limites (fins de course logiciels et zones de protection par exemple), la position des unités signale la limite qui réduit le plus le déplacement spécifié.

Position des dizaines

10:

limite violée par la valeur initiale.

20:

limite violée par la droite spécifiée. La valeur de retour correspond également à cette valeur lorsque le point final en soi ne dépasse aucune limite, mais que le déplacement du point de départ au point final provoquerait la violation d'une limite (par exemple la traversée d'une zone de protection ou des fins de course logiciels courbés dans le SCP dans le cas de transformations non linéaires telles que Transmit).

Position des centaines

100:

violation de la valeur limite positive (uniquement lorsque la position des unités est égale à 1 ou à 2, autrement dit dans le cas de fins de course logiciels ou d'une limitation de la zone de travail).

100:

Violation d'une zone de protection du NCK (uniquement si la position des unités est égale à 3).

200:

violation de la valeur limite négative (uniquement lorsque la position des unités est égale à 1 ou à 2, autrement dit dans le cas de fins de course logiciels ou d'une limitation de la zone de travail).

200:

Violation d'une zone de protection spécifique au canal (uniquement si la position des unités est égale à 3).

Position des milliers

1000:

Facteur multipliant le numéro de l'axe qui viole la limite (uniquement lorsque la position des unités est égale à 1 ou à 2, autrement dit dans le cas de fins de course logiciels ou d'une limitation de la zone de travail).

Le comptage des axes commence à 1 et se rapporte aux axes machines dans le cas d'une violation des fins de course logiciels (position des unités = 1), et aux axes géométriques dans le cas d'une violation de la limitation de la zone de travail (position des unités = 2).

1000:

Facteur multipliant le numéro de la zone de protection violée (uniquement si la position des unités est égale à 3).

En cas de violation de plusieurs zones de protection, la position des centaines et la position des milliers signalent la zone de protection qui réduit le plus le déplacement spécifié.

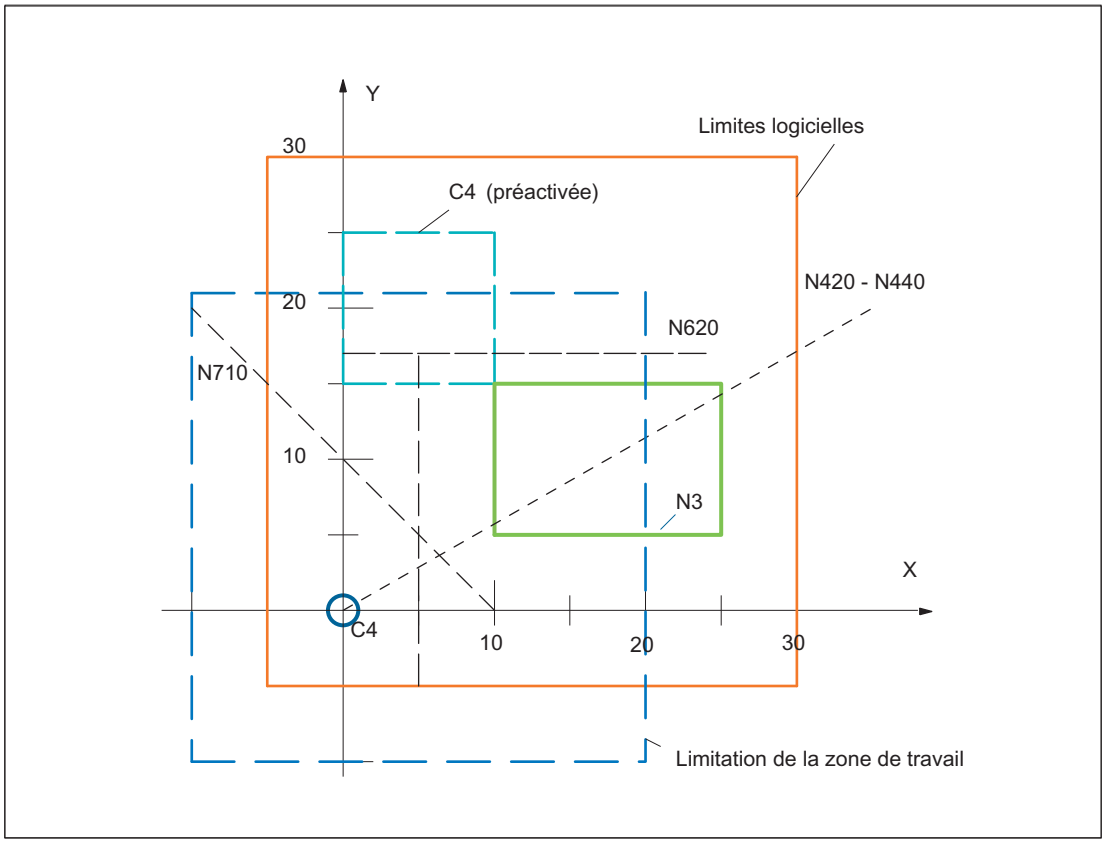
`_STARTPOS` Valeur initiale pour abscisse [0], ordonnée [1] et cote [2] dans le (SCP)

`_MOVEDIST` Consigne de déplacement relative pour abscisse [0], ordonnée [1] et cote [2]

_DLIMIT	[0] - [2]: distances minimales affectées aux axes géométriques. [3]: distance minimale affectée à un axe machine linéaire pour une transformation non linéaire lorsque l'affectation d'un axe géométrique univoque est impossible. [4]: distance minimale affectée à un axe machine rotatif pour une transformation non linéaire lorsque l'affectation d'un axe géométrique univoque est impossible. Uniquement pour les transformations spéciales avec une surveillance des fins de course logiciels.
_MAXDIST	Tableau [0] - [2] pour valeur de retour. Déplacement relatif suivant les trois axes géométriques, sans dépassement de la consigne de distance minimale d'une limite d'axe dans l'un des axes machine concernés. Si le déplacement n'est pas limité, le contenu de ce paramètre de retour est identique à celui de _MOVDIST.
_BASE_SYS	FALSE ou absence d'indication du paramètre : Lors de l'exploitation des indications de position et de longueur, c'est le code G du groupe 13 (G70, G71, G700, G710, pouces/métrique) qui est exploité. Lorsque G70 est actif et que le système de base est métrique (ou lorsque G70 est actif et que le système de base est en pouces), les variables système SCP $AA_IW[X] et $AA_MW[X]) sont fournies dans le système de base et doivent être converties le cas échéant pour pouvoir être utilisées par la fonction CALCPOSI. TRUE : Le système de base de la commande est toujours utilisé pour l'exploitation des indications de position et de longueur, indépendamment de la valeur du code G actif du groupe 13.
_TESTLIM	Limitations à surveiller (codage binaire) : 1: surveillance des fins de course logiciels 2: surveillance des limitations de la zone de travail 3: surveillance des zones de protection activées 4: surveillance des zones de protection préactivées Combinaisons par addition des valeurs. Valeur par défaut : 15 = tout surveiller.

Exemple

Dans l'exemple (voir figure), des fins de course logiciels et des limitations de la zone de travail sont inscrits sur l'axe X. En plus, trois zones de protection sont définies : les deux zones de protection C2 et C4 spécifiques au canal ainsi que la zone de protection N3 du NCK. C2 est une zone de protection active circulaire de 2 mm de rayon, orientée outil. C4 est une zone de protection carrée de 10 mm de côté, préactivée et orientée pièce, et N3 est une zone de protection active rectangulaire de 10 et 15 mm de côté. Dans le programme CN suivant, les zones de protection et les limitations de la zone de travail sont d'abord définies comme sur la figure, puis la fonction CALCPOSI est appelée avec différents paramétrages. Les résultats des différents appels de la fonction CALCPOSI sont résumés dans le tableau qui figure à la fin de l'exemple.



Code de programme	Commentaire
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 _ETAT	

Code de programme	**Commentaire**
N70 cprotdef(2, true, 0)	; Zone de protection orientée outil
N80 g17 g1 x-y0	
N90 g3 i2 x2	
N100 i-x-	
N110 execute(_SB)	
N120 cprotdef(4, false, 0)	; Zone de protection orientée pièce
N130 g17 g1 x0 y15	
N140 x10	
N150 y25	
N160 x0	
N170 y15	
N180 execute(_SB)	
N190 nprotdef(3, false, 0)	; Zone de protection orientée machine
N200 g17 g1 x10 y5	
N210 x25	
N220 y15	
N230 x10	
N240 y5	
N250 execute(_SB)	
N260 cprot(2,2,0, 0, 0)	; Activation ou préactivation des zones de protection
N270 cprot(4,1,0, 0, 0)	
N280 nprot(3,2,0, 0, 0)	
N290 g25 XX=-YY=-	; Définition des limitations de la zone de travail
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.	
; Différents appels de fonctions	; Autre point de départ
N420 _ETAT = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST)	
N430 _ETAT = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,,3)	
N440 _ETAT = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,,1)	

Code de programme	Commentaire
N450 _STARTPOS[0] = 5.	; Autre destination
N460 _STARTPOS[1] = 17.	
N470 _STARTPOS[2] = 0.	
N480 _MOVDIST[0] = 0.	
N490 _MOVDIST[1] =-.	
N500 _MOVDIST[2] = 0.	
; Différents appels de fonctions	
N510 _ETAT = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,,14)	
N520 _ETAT = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,, 6)	
N530 _DLIMIT[1] = 2.	
N540 _ETAT = 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 _ETAT = 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 _ETAT = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST)	
N720 M30	

Résultats des surveillances de l'exemple :

N° de bloc N...	_ETAT	_MAXDIST [0] (= X)	_MAXDIST [1] (= Y)	Observations
420	3123	8.040	4.594	Violation de la zone de protection N3
430	1122	20.000	11.429	Aucune violation d'une zone de protection/limitation de la zone de travail.
440	1121	30.000	17.143	Seule la surveillance des fins de course logiciels est encore active.
510	4213	0.000	0.000	Violation de la zone de protection C4 au point de départ
520	0000	0.000	–.000	La zone de protection préactivée C4 n'est pas surveillée. Le déplacement spécifié peut être entièrement exécuté.
540	2222	0.000	–.000	Comme _DLIMIT[1]=2, le déplacement est réduit par la limitation de la zone de travail.
620	4223	–.000	0.000	A cause de C2 et de _DLIMIT[3], la distance de C4 est de 4 mm au total. La distance C2 –N3 de 0.1 mm ne limite par le déplacement.
710	1221	0.000	21.213	Frame actif avec translation et rotation. Le déplacement admis dans _MOVDIST est valable dans le système de coordonnées décalé et tourné (SCP).

Cas particuliers et autres détails

Toutes les instructions de déplacement préliminaires sont toujours au rayon, même dans le cas d'un axe transversal, avec le code G “DIAMON” activé. Lorsque le chemin d'un des axes concernés ne peut être entièrement parcouru, les chemins des autres axes sont également réduits dans la valeur de retour _MAXDIST, de telle manière que le produit final résultant se trouve sur la trajectoire prescrite.

Une absence de définition de limite logicielle, de limitation de zone de travail et de zone de protection est admise pour un ou plusieurs des axes concernés. L'ensemble des limites n'est surveillé que lorsque les axes concernés on été référencés. Les axes rotatifs éventuellement concernés ne sont surveillés que s'il ne s'agit pas d'axes à valeur modulo.

La surveillance des limites logicielles et des limitations de la zone de travail dépend, comme dans le mode de déplacement normal, des réglages actifs (signaux d'interface pour la sélection des limites logicielles 1 ou limites logicielles 2, GWALIMON/WALIMOF, données de réglage pour l'activation individuelle des limites de la zone de travail et pour déterminer si le rayon de l'outil actif doit être ou non pris en compte lors de la surveillance des limitations de la zone de travail).

Avec certaines transformations cinématiques (par ex. TRANSMIT), la position des axes machine ne peut pas être déterminée de façon univoque dans le système de coordonnées pièce (SCP) (ambiguïté). Dans le mode de déplacement normal, l'univocité provient, en règle générale, de l'historique et de la condition selon laquelle un déplacement continu dans le SCP doit correspondre à un déplacement continu des axes machine. Lors de la surveillance des limites logicielles à l'aide de la fonction CALCPOSI, la position machine actuelle est utilisée dans ces cas pour lever l'ambiguïté. Le cas échéant, il faudra programmer un **STOPRE** avant CALCPOSI pour que la machine dispose de positions d'axe valables.

Il n'est pas assuré que, pour les zones de protection lors d'un mouvement sur la course prescrite, la distance spécifiée sous _DLIMIT[3] soit respectée partout. En cas de modification du point final retourné dans _MOVDIST, il n'est donc pas possible de violer la zone de protection de cette distance. La droite peut, dans sa variation, mais avec une épaisseur quelconque, passer devant une zone de protection.

Remarque

Pour plus de détails sur les limitations de la zone de travail, voir le
/PG/ Manuel de programmation Notions de base,

sur les limites logicielles, voir
/FB1/ Manuel de fonctions de base ; surveillance d'axes, zones de protection (A3).

Instructions de déplacement spéciales 4

4.1 Accostage de positions codées (CAC, CIC, CDC, CACP, CACN)

Fonction

Les instructions suivantes permettent, au moyen de numéros de position, de déplacer des axes linéaires et rotatifs jusqu'à des positions d'axe fixes définies dans des tableaux de paramètres machine. Ce type de programmation est appelé "accostage de positions codées".

Syntaxe

```
CAC(<n>)
CIC(<n>)
CACP(<n>)
CACN(<n>)
```

Signification

`CAC(<n>)`	Accostage de la position codée du numéro de position n
`CIC(<n>)`	Accostage de la position codée n positions en avant (+n) ou en arrière (–n), à partir du numéro de position courant
`CDC(<n>)`	Accostage de la position codée sur le chemin le plus court depuis le numéro de position n (uniquement pour les axes rotatifs)
`CACP(<n>)`	Accostage de la position codée dans le sens positif depuis le numéro de position n (uniquement pour les axes rotatifs)
`CACN(<n>)`	Accostage de la position codée dans le sens négatif depuis le numéro de position n (uniquement pour les axes rotatifs)
`<n>`	Numéro de position dans le tableau de paramètres machine Plage de valeurs : 0, 1, … (nombre maximal de positions dans le tableau - 1)

Exemple : Accostage des positions codées d'un axe de positionnement

```
Code de programme                Commentaire
N10 FA[B]=300                    ; Avance de l'axe de positionnement B
N20 POS[B]=CAC(10)               ; Accostage de la position codée du numéro de
                                   position 10
N30 POS[B]=CIC(-4)               ; Accostage de la position codée du "numéro de
                                   position courant" - 4
```

Bibliographie

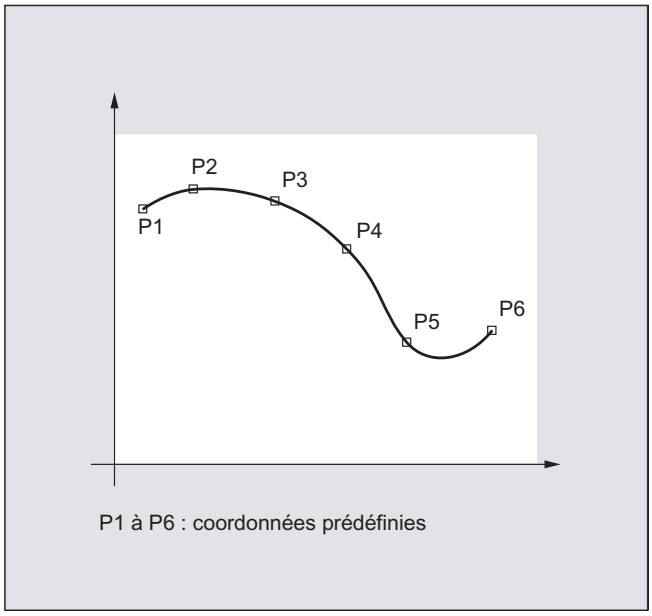
- Description fonctionnelle Fonctions d'extension ; Axes indexés (T1)
- Description fonctionnelle Actions synchrones

4.2 Interpolation de type spline (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL)

Fonction

Pour les contours de pièce de forme courbe quelconque, une description analytique exacte est impossible. C'est pourquoi de tels contours sont décrits approximativement par un nombre limité de points intermédiaires, notamment lors de la numérisation des surfaces. Pour générer une surface de pièce numérisée, il est nécessaire de relier les points intermédiaires et d'en faire une description du contour. C'est ce que permet l'interpolation de type spline.

Un spline définit une courbe composée de polynômes du 2e ou 3e degré. Les propriétés des points intermédiaires d'un spline peuvent être définis **selon le type de spline utilisé**.



SINUMERIK solution line met à disposition des types de spline suivants :

- Spline A
- Spline B
- Spline C

Syntaxe

Généralités :

```
ASPLINE X... Y... Z... A... B... C...
BSPLINE X... Y... Z... A... B... C...
CSPLINE X... Y... Z... A... B... C...
```

Le spline B permet en outre de programmer :

```
PW=<n>
SD=2
PL=<valeur>
```

Les splines A et C permettent en outre de programmer :

`BAUTO` / `BNAT` / `BTAN`

`EAUTO` / `ENAT` / `ETAN`

Signification

Type d'interpolation spline :

`ASPLINE`	Instruction d'activation de l'interpolation de type spline A
`BSPLINE`	Instruction d'activation de l'interpolation de type spline B
`CSPLINE`	Instruction d'activation de l'interpolation de type spline C
	Les instructions `ASPLINE`, `BSPLINE` et `CSPLINE` ont un effet modal et font partie du groupe des instructions de déplacement.

Points intermédiaires et points de contrôle :

`X... Y... Z...` `A... B... C...`	Positions en coordonnées cartésiennes

Poids de point (uniquement spline B) :

`PW`	L'instruction `PW` permet de programmer un "poids de point" pour chaque point intermédiaire.		
`<n>`	"Poids de point"		
	Plage de valeurs :	0 ≤ n ≤ 3	
	Pas :	0.0001	
	Effet :	n > 1	La courbe tend plus fortement.
		n < 1	La courbe tend moins fortement.

Degré spline (uniquement spline B) :

`SD`	Un polygone du troisième degré est appliqué de façon standard. Mais la programmation de `SD=2` permet également d'utiliser un polygone du deuxième degré.

Distance entre les pôles (uniquement spline B) :

`PL` Les distances entre les pôles sont calculées par le système, Mais la commande peut également traiter les distances prescrites en tant que "Paramètre - Intervalle - Longueur" avec l'instruction `PL`.

`<Valeur>` Paramètre-Intervalle-Longueur

Plage de valeurs comme pour le positionnement non indexé :

Comportement de transition au début de la courbe spline (splines A et C uniquement) :

`BAUTO` Aucune prescription pour le comportement de transition. Le début résulte de la position du premier point.

`BNAT` Courbure nulle

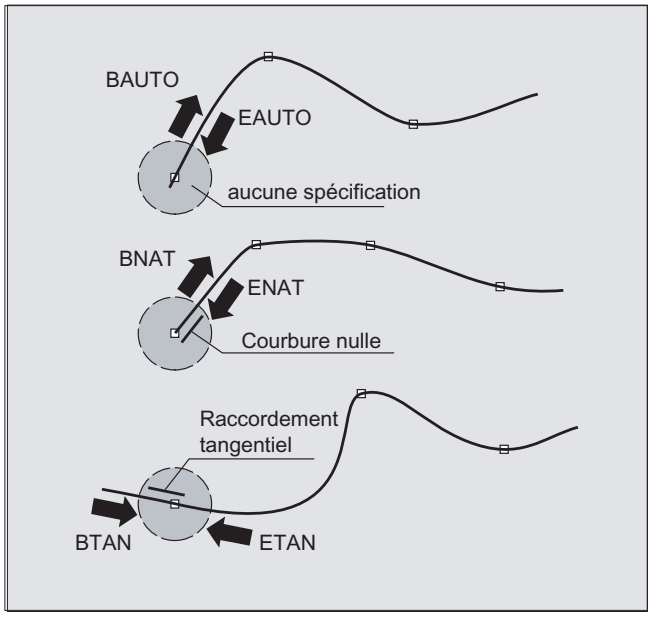
`BTAN` Raccordement tangentiel avec bloc précédent (fonction initialisée)

Comportement de transition à la fin de la courbe spline (splines A et C uniquement) :

`EAUTO` Aucune prescription pour le comportement de transition. La fin résulte de la position du dernier point.

`ENAT` Courbure nulle

`ETAN` Raccordement tangentiel avec bloc précédent (fonction initialisée)



Remarque

Le comportement de transition programmable n'influence pas le spline B. Au point de départ et au point final, le spline B est toujours tangent au polygone de contrôle.

Conditions marginales

- La correction de rayon d'outil est utilisable.
- La surveillance anticollision s'effectue par projection dans le plan.

Exemples

Exemple 1 : Spline B

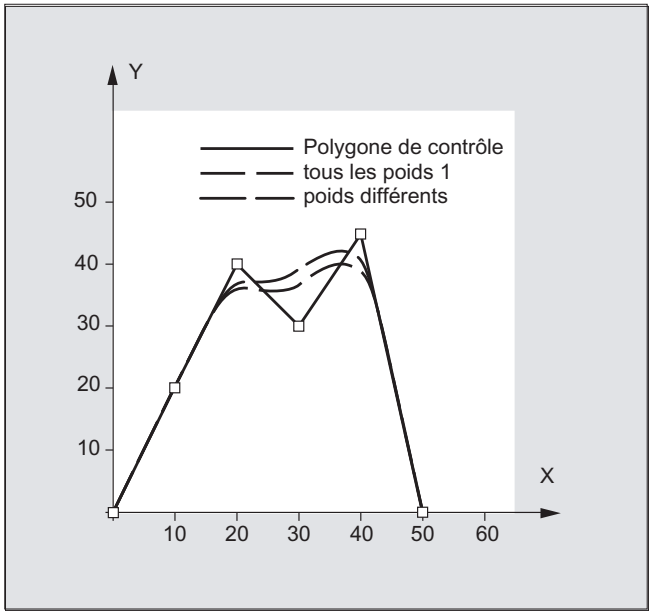
Code de programme 1 (tous les poids 1)

```
N10 G1 X0 Y0 F300 G64
N20 BSPLINE
N30 X10 Y20
N40 X20 Y40
N50 X30 Y30
N60 X40 Y45
N70 X50 Y0
```

Code de programme 2 (différents poids)

```
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
```

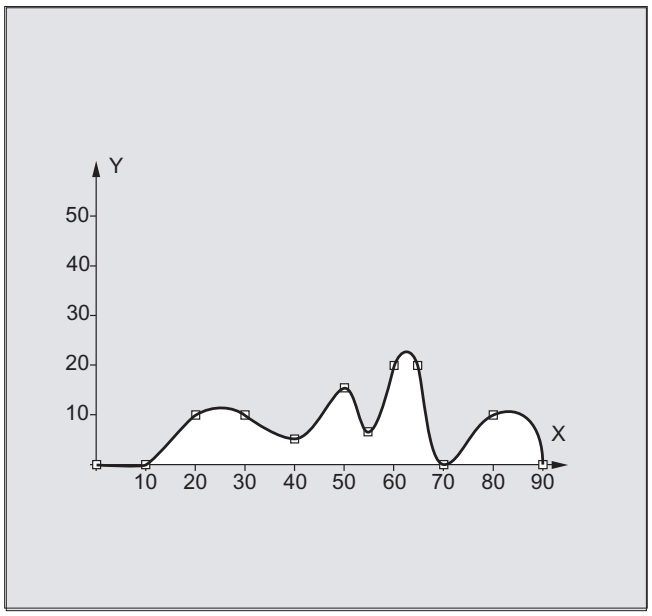
Code de programme 3 (polygone de contrôle)	**Commentaire**
N10 G1 X0 Y0 F300 G64	
N20	; ne s'applique pas
N30 X10 Y20	
N40 X20 Y40	
N50 X30 Y30	
N60 X40 Y45	
N70 X50 Y0	



Exemple 2 : Spline C, courbure nulle au début et à la fin

Code de programme

```
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
```



Exemple 3 : Interpolation spline (spline A) et transformation de coordonnées (ROT)

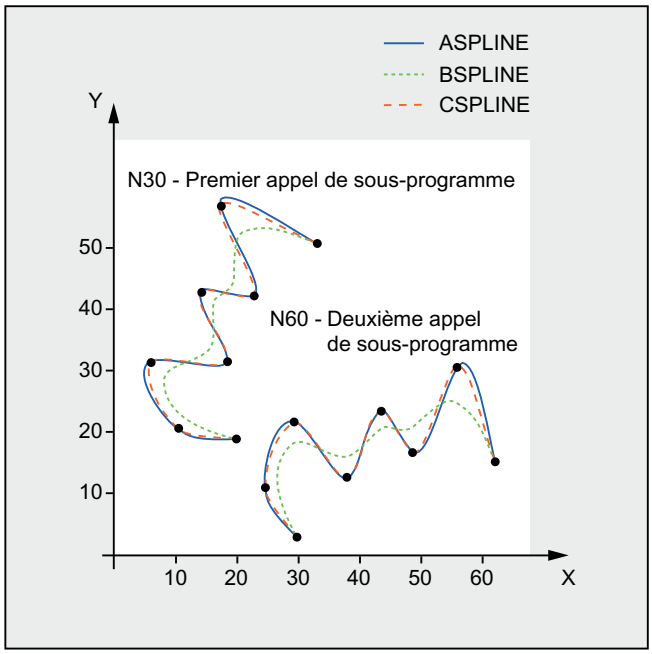
Programme principal :

Code de programme	Commentaire
N10 G00 X20 Y18 F300 G64	; Accostage du point de départ.
N20 ASPLINE	; Activation de l'interpolation de type spline A.
N30 CONTOUR	; Premier appel du sous-programme.
N40 ROT Z-45	; Transformation de coordonnées : rotation du SCP de -45° autour de l'axe Z.
N50 G00 X20 Y18	; Accostage du point de départ du contour.
N60 CONTOUR	; Deuxième appel du sous-programme.
N70 M30	; Fin du programme

Sous-programme "Contour" (contenant les coordonnées des points intermédiaires) :

Code de programme
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

La figure suivante contient la courbe spline résultante de l'exemple de programme (ASPLINE), ainsi que les courbes splines qui auraient résulté d'une interpolation de type spline B ou C (BSPLINE, CSPLINE) :



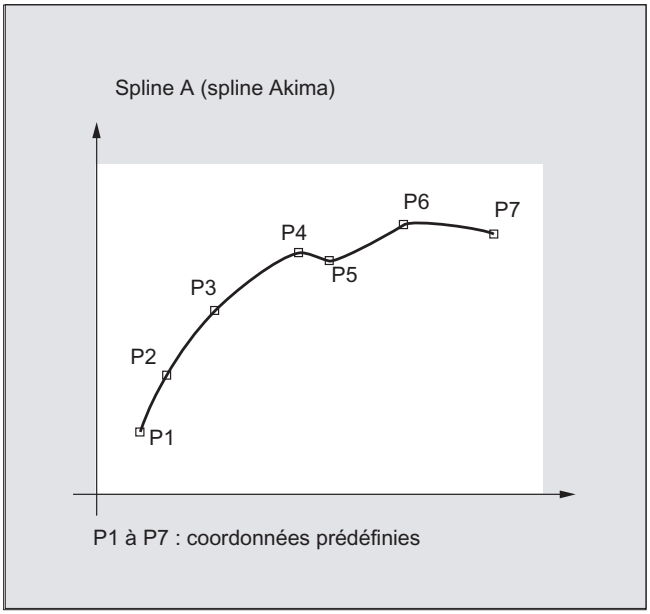
Autres informations

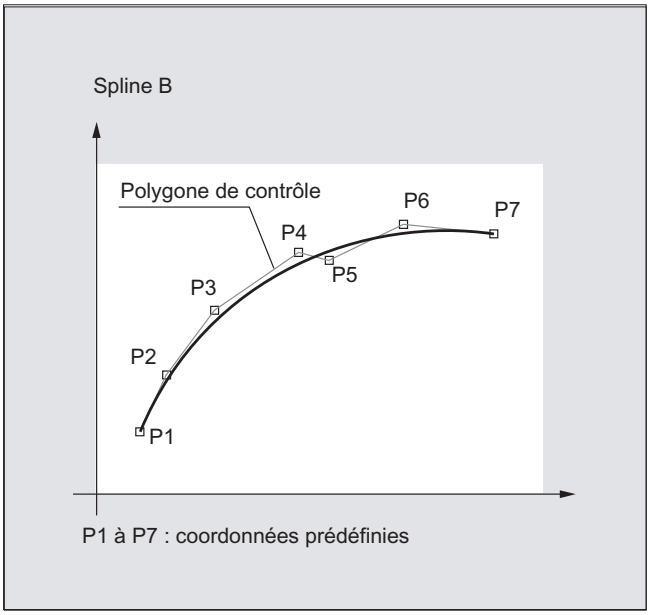
Avantages de l'interpolation de type spline

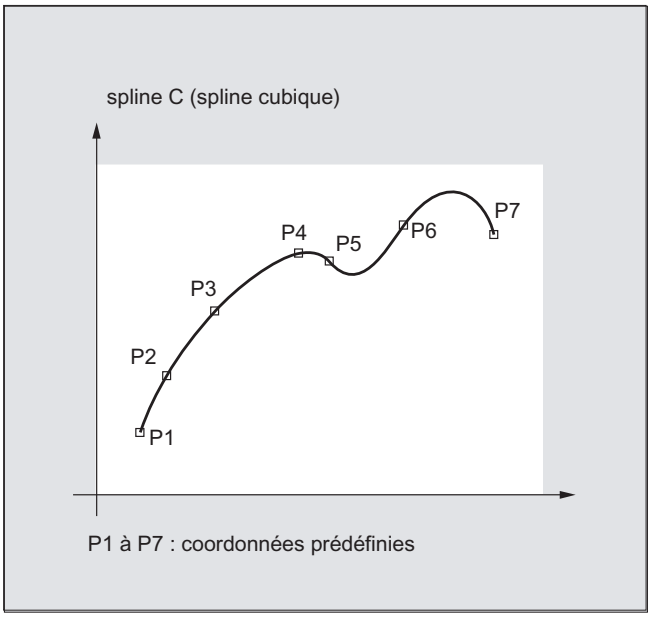
Contrairement à l'utilisation de blocs linéaires `G01`, l'interpolation de type spline présente les avantages suivants :

- Réduction de nombre de blocs requis dans le programme pièce pour la description du contour
- Courbes douces ménageant la mécanique lors des transitions entre les blocs du programme pièce

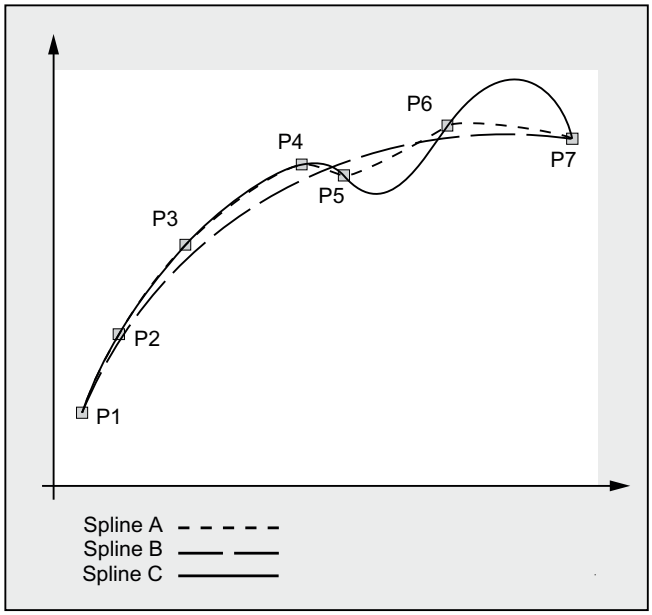
Propriétés et applications des différents types de spline

Type spline	Propriétés et applications
Spline A	 **Propriétés :** • Courbe passant exactement par les points intermédiaires indiqués. • La courbe est caractérisée par la continuité de la tangente, mais pas du rayon de courbure. • Elle ne crée presque pas de vibrations indésirables. • La modification d'un point intermédiaire a un effet local, ce qui signifie qu'elle n'influence pas plus de 6 points intermédiaires voisins. **Application :** Le spline A convient avant tout à l'interpolation de courbes avec de grandes variations de pente (par exemple les courbes en forme d'escalier).

Type spline	Propriétés et applications
Spline B	 **Propriétés :** • Courbe ne passant pas par les points intermédiaires indiqués, mais à proximité de ceux-ci. La courbe tend vers les points intermédiaires. L'utilisation d'un facteur de pondération des points intermédiaires permet en outre d'influencer le tracé de la courbe. • La courbe est caractérisée par la continuité de la tangente et du rayon de courbure. • Elle ne crée aucune vibration indésirable. • La modification d'un point intermédiaire a un effet local, ce qui signifie qu'elle n'influence pas plus de 6 points intermédiaires voisins. **Application :** Le spline B est destiné en premier lieu à servir d'interface avec les systèmes de CAO.

Type spline	Propriétés et applications
Spline C	 **Propriétés :** • Courbe passant exactement par les points intermédiaires indiqués. • La courbe est caractérisée par la continuité de la tangente et du rayon de courbure. • Elle crée fréquemment des vibrations indésirables, en particulier aux endroits avec de grandes variations de pente. • La modification d'un point intermédiaire a un effet global, ce qui signifie qu'elle influence l'ensemble du tracé de la courbe. **Application :** Le spline C convient particulièrement bien lorsque les points se situent sur une courbe analytique connue (cercle, parabole, hyperbole).

Comparaison des trois types de splines avec des points intermédiaires identiques



Nombre minimum de blocs spline

Les instructions ASPLINE, BSPLINE et CSPLINE relient les points finaux de blocs par des courbes de type spline. Pour ce faire, la commande doit effectuer le décodage simultané d'une série de blocs (points finaux) pendant le prétraitement. En version standard, la taille du tampon pour le décodage vaut 10 blocs. Toutes les informations de bloc ne sont pas forcément des fins de bloc spline. La commande nécessite néanmoins un nombre précis de fins de bloc spline provenant de 10 blocs :

Type spline	Nombre minimum de blocs spline
spline A :	Sur 10 blocs, **4** au moins doivent être des blocs spline. Les blocs de commentaire et les calculs de paramètres ne sont pas pris en compte.
spline B :	Sur 10 blocs, **6** au moins doivent être des blocs spline. Les blocs de commentaire et les calculs de paramètres ne sont pas pris en compte.
spline C :	Le nombre minimum de blocs spline nécessaire se calcule de la manière suivante : valeur de PM20160 $MC_CUBIC_SPLINE_BLOCKS + 1 Le nombre de points à l'aide desquels la section de spline est calculée est renseigné dans PM20160. Le réglage par défaut est 8. Dans le cas standard, il faut donc au moins que **9** blocs sur 10 soient des blocs spline.

Remarque

Si la valeur est inférieure à la valeur tolérée, une alarme est émise, de même que si un axe participant à l'interpolation de type spline est programmé comme axe de positionnement.

Groupement de blocs spline courts

Lors de l'interpolation de type spline des blocs spline courts peuvent être créés, qui contribue inutilement à la réduction de la vitesse tangentielle. La fonction "groupement de blocs spline courts" permet de grouper ces blocs de manière à ce que la longueur de bloc en résultant soit assez longue et à ne pas réduire la vitesse tangentielle.

La fonction est activée via le paramètre machine spécifique au canal :

PM20488 $MC_SPLINE_MODE (réglage pour interpolation de type spline)

Bibliographie :
Description fonctionnelle Fonctions de base ; contournage, arrêt précis, look-ahead (B1), chapitre : Groupement de blocs spline courts

4.3 Groupe spline (SPLINEPATH)

Fonction

Les axes à interpoler dans le groupe spline sont sélectionnés avec l'instruction `SPLINEPATH`. L'interpolation de type spline admet jusqu'à 8 axes à interpolation.

Remarque

Si l'instruction `SPLINEPATH` n'est pas programmée de façon explicite, ce sont les trois premiers axes du canal qui seront déplacés en tant que groupe spline.

Syntaxe

La définition du groupe spline se fait dans un bloc séparé :

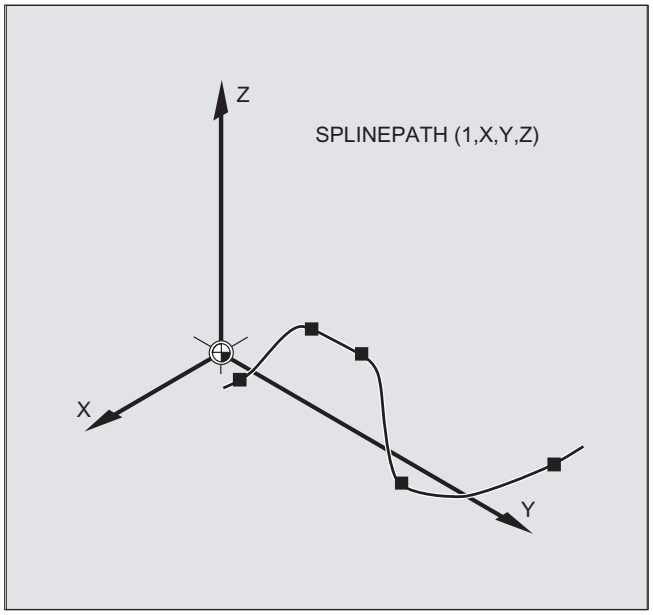
```
SPLINEPATH(n,X,Y,Z,…)
```

Signification

`SPLINEPATH`	Instruction de définition du groupe spline
`n`	=1 (valeur fixe)
`X,Y,Z,…`	Descripteurs des axes à interpoler dans le groupe spline

Exemple : Groupe spline avec trois axes à interpolation

Code de programme	Commentaire
N10 G1 X10 Y20 Z30 A40 B50 F350	
N11 SPLINEPATH(1,X,Y,Z)	; Groupe spline
N13 CSPLINE BAUTO EAUTO X20 Y30 Z40 A50 B60	; Spline C
N14 X30 Y40 Z50 A60 B70	; Points intermédiaires
…	
N100 G1 X… Y…	; Désactivation de l'interpolation de type spline



4.4 Compactage de bloc CN (COMPON, COMPCURV, COMPCAD, COMPOF)

Fonction

Les systèmes de CAO/FAO fournissent généralement des blocs linéaires qui respectent la précision paramétrée. Ceci conduit, dans le cas des contours complexes, à une quantité considérable de données et, éventuellement, à des segments de trajectoire courts. Ceux-ci limitent la vitesse d'usinage.

Grâce à l'utilisation de blocs de polynôme, une fonction compacteur entraîne un rapprochement du contour défini par blocs linéaires. Il en résulte les avantages suivants :

- Réduction de nombre de blocs requis dans le programme pièce pour la description du contour de la pièce
- Transitions progressives entre les blocs
- Augmentation des vitesses tangentielles maximales

Les fonctions compacteur suivantes sont disponibles :

- COMPON

  Seule la **vitesse** est continue aux transitions entre blocs, l'accélération des axes concernés pouvant présenter des échelons.

- COMPCURV

  L'**accélération** est continue aux transitions entre les blocs. La vitesse aussi bien que l'accélération de tous les axes ne varient donc pas brusquement aux transitions entre blocs.

- COMPCAD

  Compactage caractérisé par une optimisation de l'état de surface et de la vitesse, et qui exige beaucoup de temps de calcul et de capacité mémoire. COMPCAD ne devrait être utilisé que lorsque les mesures, permettant une amélioration de l'état de surface, ne peuvent pas être assurées en amont par le programme CAO/FAO.

La fonction compacteur se termine avec `COMPOF`.

Syntaxe

```
COMPON
COMPCURV
COMPCAD
COMPOF
```

Signification

`COMPON` :	Instruction d'activation de la fonction compacteur COMPON.
	Prise d'effet : modale
`COMPCURV` :	Instruction d'activation de la fonction compacteur COMPCURV.
	Prise d'effet : modale
`COMPCAD` :	Instruction d'activation de la fonction compacteur COMPCAD.
	Prise d'effet : modale
`COMPOF` :	Instruction de désactivation de la fonction compacteur active à l'instant.

Remarque

Pour une amélioration supplémentaire de l'état de surface, il est possible d'utiliser la fonction d'arrondissement `G642` et la limitation des à-coups `SOFT`. Ces instructions doivent figurer en début de programme.

Contraintes

- La compression de bloc CN est en règle générale effectué pour des blocs linéaires (`G1`).
- Seuls les blocs qui satisfont à une syntaxe simple sont compactés :

  ```
  N... G1X... Y... Z... F... ; commentaire
  ```

  Tous les autres blocs sont exécutés tels quels (sans compactage).
- Les blocs de déplacement à adresses étendues tels que `C=100` ou `A=AC(100)` sont également compactés.
- Il n'est pas nécessaire de programmer directement les valeurs de position qui peuvent également être indiquées indirectement par des affectations de paramètres telles que `X=R1*(R2+R3)`.
- Lorsque l'option "Transformation d'orientation" est disponible, il est également possible de compacter des blocs CN dans lesquels l'orientation de l'outil (et la rotation de l'outil le cas échéant) est programmée au moyen de vecteurs de direction (voir "Compactage de l'orientation (Page 353)"
- Le compactage est interrompu par toute autre instruction CN telle qu'une sortie de fonction auxiliaire.

Exemples

Exemple 1 : COMPON

```
Code de programme                    Commentaire
N10 COMPON                           ; Fonction compacteur COMPON activée.
N11 G1 X0.37 Y2.9 F600               ; G1 avant point final et avance.
N12 X16.87 Y-.698
N13 X16.865 Y-.72
N14 X16.91 Y-.799
…
N1037 COMPOF                         ; Fonction compacteur désactivée.
…
```

Exemple 2 : COMPCAD

```
Code de programme                    Commentaire
G00 X30 Y6 Z40
G1 F10000 G642                       ; Fonction d'arrondissement G642 activée.
SOFT                                 ; Limitation des à-coups SOFT activée.
COMPCAD                              ; Fonction compacteur COMPCAD activée.
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                               ; Fonction compacteur désactivée.
G00 Z50
M30
```

Bibliographie

Description fonctionnelle Fonctions de base ; contournage, arrêt précis, look-ahead (B1), chapitre : "Compactage de bloc CN"

4.5 Interpolation polynomiale (POLY, POLYPATH, PO, PL)

Fonction

L'interpolation polynomiale (POLY) n'est en fait pas une interpolation de type spline. Elle est destinée avant tout à servir d'interface pour la programmation de courbes spline générées de manière externe. Les sections spline peuvent être programmées directement.

Ce mode d'interpolation évite à la CN de calculer les coefficients de polynôme. Elle peut être utilisée de manière optimale lorsque les coefficients sont issus directement d'un système de CAO ou d'un postprocesseur.

Syntaxe

Polynôme du 3e degré :

```
POLY PO[X]=(xe,a2,a3) PO[Y]=(ye,b2,b3) PO[Z]=(ze,c2,c3) PL=n
```

Polynôme du 5ème degré et nouvelle syntaxe polynomiale :

```
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")
```

Remarque

La somme des coefficients de polynôme et des axes programmés dans un bloc CN ne doit pas dépasser le nombre maximal d'axes autorisés par bloc.

Signification

POLY :	Activation de l'interpolation polynomiale par un bloc avec POLY.
POLYPATH :	Interpolation polynomiale sélectionnable pour les deux groupes d'axes AXIS ou VECT
PO[descripteur d'axe/variable] :	Points finaux et coefficients de polynôme
X, Y, Z :	Descripteur d'axe
xe, ye, ze :	Indication de la position finale de l'axe correspondant ; plage de valeurs comme pour le positionnement non indexé
a2, a3, a4, a5 :	Les coefficients a_2, a_3, a_4, et a_5 sont écrits avec leur valeur ; plage de valeurs comme pour le positionnement non indexé. Le dernier coefficient peut être omis si sa valeur est égale à zéro.
PL :	Longueur de l'intervalle de paramètre sur lequel le polynôme est défini (domaine de définition de la fonction f(p)). L'intervalle commence toujours à 0 et la valeur de p peut aller de 0 à PL. Plage de valeurs théorique de PL : 0,0001 … 99 999,9999 **Remarque :** La valeur de PL est valable pour le bloc dans lequel elle figure. Si PL n'est pas programmé, PL=1.

Activation/désactivation de l'interpolation polynomiale

L'interpolation polynomiale est activée dans le programme pièce au moyen de l'instruction G `POLY`.

L'instruction G `POLY` ainsi que `G0`, `G1`, `G2`, `G3`, `ASPLINE`, `BSPLINE` et `CSPLINE` font partie du 1er groupe G.

Etant uniquement programmés avec un nom et un point final (par ex. `X10`) les axes sont déplacés linéairement. Après avoir programmé tous les axes d'un bloc CN de la sorte, la commande se comporte comme pour G1.

L'interpolation polynomiale est de nouveau désactivée de manière implicite en programmant une autre instruction du 1er groupe G (par ex. `G0`, `G1`).

Coefficient du polynôme

La valeur `(PO[]=)` ou `...=PO(...)` indique tous les coefficients de polynôme d'un axe. Selon le degré du polynôme, plusieurs valeurs séparées par des virgules sont spécifiées. Dans un bloc, différents degrés de polynôme de différents axes sont possibles.

Sous-programme POLYPATH

`POLYPATH(...)` permet de débloquer l'interpolation polynomiale de manière sélective pour certains groupes d'axes :

Uniquement axes à interpolation et axes supplémentaires :	`POLYPATH("AXES")`
Uniquement axes d'orientation : (lors du déplacement avec transformation d'orientation)	`POLYPATH("VECT")`

Les axes respectifs non débloqués sont déplacés linéairement.

Par défaut, l'interpolation polynomiale est débloquée pour les deux groupes d'axes.

Une programmation sans paramètre `POLYPATH( )` permet de désactiver l'interpolation polynomiale pour tous les axes.

Exemple

```
Code de programme                                                    Commentaire
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                  ; Interpolation polynomiale activée
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                                      ; Indications mixtes pour les axes
N27 PO[X]=(10,2.5) PO[Y]=(2.3)                                       ; PL n'est pas programmé ; donc PL=1.
N30 G1 X… Y… Z.                                                      ; Interpolation polynomiale désactivée
…
```

Exemple : Nouvelle syntaxe polynomiale

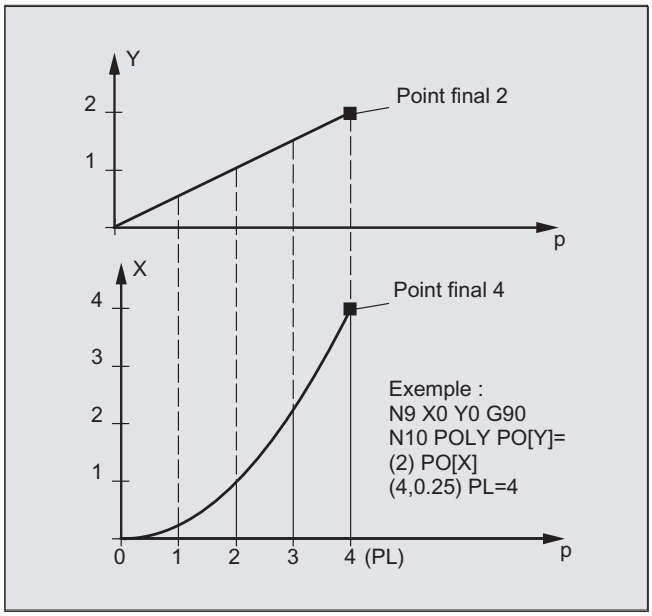
Syntaxe polynomiale encore valable	Nouvelle syntaxe polynomiale
PO[descripteur d'axe]=(.. , ..)	Descripteur d'axe=PO(.. , ..)
PO[PHI]=(.. , ..)	PHI=PO(.. , ..)
PO[PSI]=(.. , ..)	PSI=PO(.. , ..)
PO[THT]=(.. , ..)	THT=PO(.. , ..)
PO[]=(.. , ..)	PO(.. , ..)
PO[variable]=IC(.. , ..)	Variable=PO IC(.. , ..)

Exemple : Cercle dans le plan X/Y

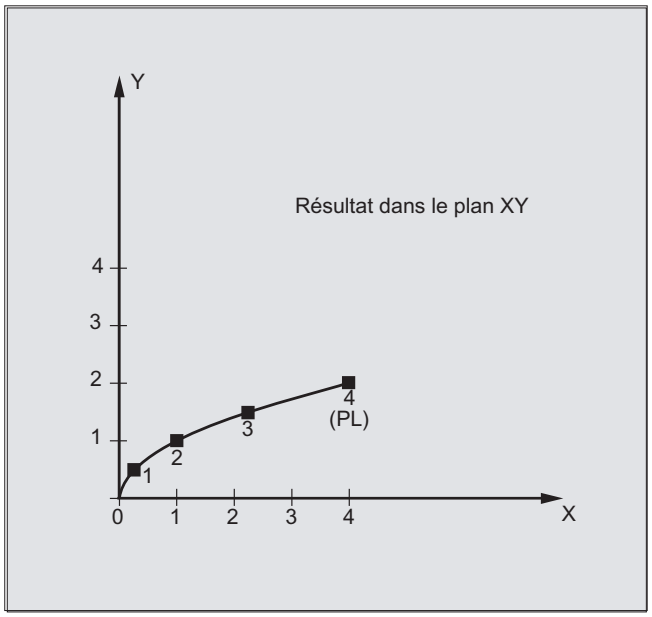
Programmation

```
Code de programme
N9 X0 Y0 G90 F100
N10 POLY PO[Y]=(2) PO[X]=(4,0.25) PL=4
```

Evolution des courbes X(p) et Y(p)



Evolution de la courbe dans le plan XY



Description

La forme générale de la fonction polynomiale est la suivante :

$f(p)= a_0 + a_1p + a_2p^2 +. . . + a_np^n$

où : a_n : coefficients constants
p : Paramètres

Le nombre maximum de polynômes du 5ème degré pouvant être programmés dans la commande est le suivant :

$f(p)= a_0 + a_1p + a_2p^2 + a_3p^3 + a_4p^4 + a_5p^5$

En affectant des valeurs concrètes aux coefficients, il est possible de générer différentes courbes : droites, paraboles et fonctions puissance.

Une courbe droite est créée par $a_2 = a_3 = a_4 = a_5 = 0$:

$f(p)= a_0 + a_1p$

En outre :

a_0 : position de l'axe à la fin du bloc précédent

p = PL

$a_1 = (x_E - a_0 - a_2*p^2 - a_3*p^3) / p$

Des polynômes peuvent être programmés **sans** avoir que l'interpolation polynomiale n'ait été activée au moyen de l'instruction G `POLY`. Dans ce cas, les polynômes programmés ne seront pas interpolés, mais les points finaux programmés pour les axes seront accostés linéairement (`G1`). Les polynômes programmés ne sont déplacés en tant que tels qu'après avoir explicitement activé l'interpolation polynomiale dans le programme pièce (`POLY`).

Particularité : polynôme dénominateur

Pour les axes géométriques, `PO[]=(…)` permet aussi de programmer un polynôme dénominateur commun sans indiquer le nom d'un axe, c'est-à-dire que le déplacement des axes géométriques sera interpolé en tant que quotient de deux polynômes.

Ceci permet, par exemple, de représenter avec exactitude des coniques (cercle, ellipse, parabole, hyperbole).

Exemple :

Code de programme	Commentaire
`POLY G90 X10 Y0 F100`	`; Déplacement linéaire des axes géométriques à la position X10 Y0.`
`PO[X]=(0,-10) PO[Y]=(10) PO[]=(2,1)`	`; Déplacement des axes géométriques en quart de cercle sur position X0 Y10.`

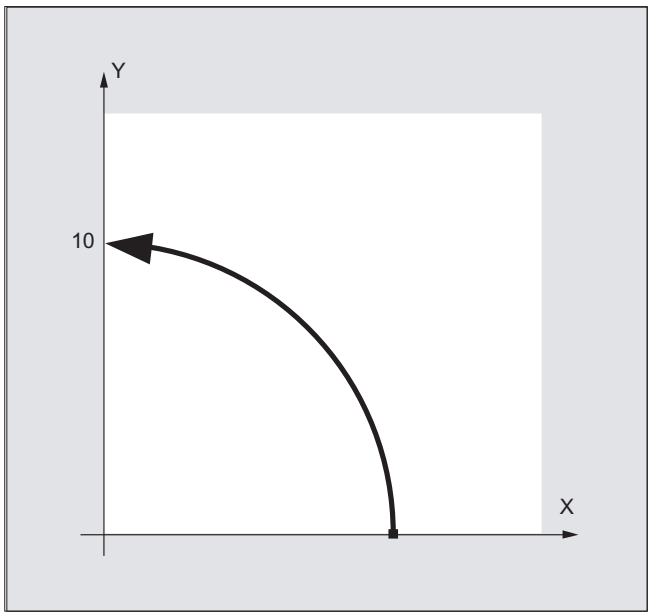
Le coefficient constant (a_0) du polynôme dénominateur est toujours considéré comme étant égal à 1. Le point final programmé est indépendant de `G90` / `G91`.

X(p) et Y(p) sont calculés comme suit à partir des valeurs programmées :

$X(p) = (10 - 10 * p^2) / (1 + p^2)$
$Y(p) = 20 * p / (1 + p^2)$
avec $0 \leq p \leq 1$

Compte tenu de la programmation des points de départ, des points finaux, du coefficient a_2 et de PL=1, les résultats intermédiaires suivants sont obtenus :

Numérateur (X) =	$10 + 0 * p - 10 * p^2$
Numérateur (Y) =	$0 + 20 * p + 0 * p^2$
Dénominateur =	$1 + p^2$



Lorsque l'interpolation polynomiale est activée, la programmation d'un polynôme dénominateur possédant des racines à l'intérieur de l'intervalle `[0,PL]` est rejetée avec une alarme. Le polynôme dénominateur n'a aucune influence sur les déplacements des axes supplémentaires.

Remarque

Dans le cas de l'interpolation polynomiale, on peut activer une correction de rayon d'outil avec `G41`, `G42` et l'utiliser comme pour l'interpolation linéaire ou circulaire.

4.6 Référence réglable de trajectoire (SPATH, UPATH)

Fonction

Pendant l'interpolation polynomiale, l'utilisateur peut souhaiter deux relations différentes entre les axes FGROUP déterminant la vitesse et les autres axes à interpolation : le déplacement de ces derniers doit être synchronisé soit avec la trajectoire S ou avec le paramètre de courbe U des axes FGROUP.

Ces deux modes d'interpolation sont nécessaires dans diverses applications et peuvent être sélectionnés à l'aide des deux instructions de langage modales SPATH et UPATH contenues dans le 45ème groupe de codes G.

Syntaxe

```
SPATH
UPATH
```

Signification

SPATH :	La référence trajectoire pour axes FGROUP est la longueur d'arc
UPATH :	La référence trajectoire pour axes FGROUP est paramètre de courbe

Remarque

UPATH et SPATH déterminent également la relation entre le polynôme décrivant le mot F (FPOLY, FCUB, FLIN) et le déplacement le long de la trajectoire.

Contraintes

La référence de trajectoire paramétrée est sans aucune signification :

- pour l'interpolation linéaire et circulaire
- dans des blocs de filetage
- lorsque tous les axes à interpolation sont compris dans FGROUP.

Exemples

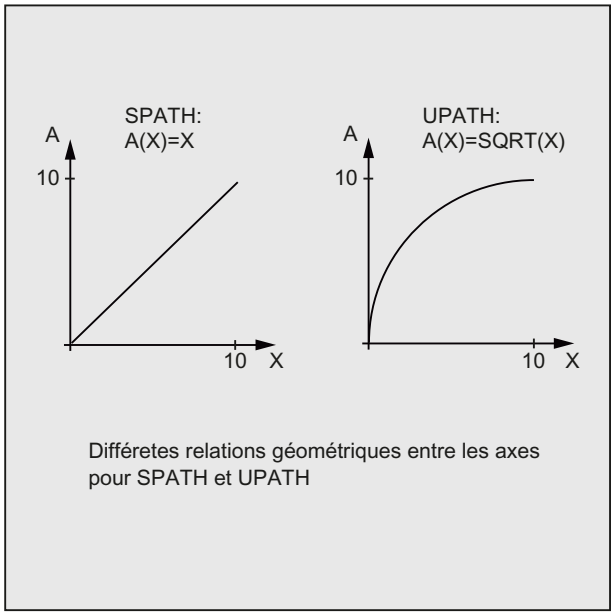
Exemple 1 :

Dans l'exemple suivant, un carré avec une longueur d'arête de 20 mm est arrondi avec G643. Dans ce contexte, les écarts maximaux du contour exact sont définis pour chaque axe avec le paramètre machine MD33100 $MA_COMPRESS_POS_TOL[<n>] spécifique à l'axe.

Code de programme	Commentaire
N10 G1 X… Y… Z… F500	
N20 G643	; Arrondissement interne à un bloc avec G643
N30 XO Y0	
N40 X20 Y0	; Longueur d'arêt (mm) pour les axes
N50 X20 Y20	
N60 X0 Y20	
N70 X0 Y0	
N100 M30	

Exemple 2 :

L'exemple suivant illustre la différence entre les deux modes de pilotage du déplacement. Dans les deux cas, on considère que le préréglage FGROUP(X,Y,Z) est actif.



Différetes relations géométriques entre les axes pour SPATH et UPATH

Code de programme

```
N10 G1 X0 A0 F1000 SPATH
N20 POLY PO[X]=(10,10) A10
```

ou

Code de programme

```
N10 G1 X0 F1000 UPATH
N20 POLY PO[X]=(10,10) A10
```

Dans le bloc N20, le trajet S des axes FGROUP dépend du carré du paramètre de courbe U. Il en résulte donc des positions différentes de l'axe synchrone A le long de la trajectoire de X, selon que SPATH ou UPATH est actif.

Informations complémentaires

Pendant l'interpolation polynomiale - ceci sous-entendant toujours l'interpolation polynomiale au sens strict du terme (POLY), toutes les interpolations de type spline (ASPLINE, BSPLINE, CSPLINE) et une interpolation linéaire avec fonction compresseur (COMPON, COMPCURV) - les positions de tous les axes à interpolation i sont spécifiées par des polynômes pi(U). Dans un bloc CN, le paramètre de courbe U varie de 0 à 1, c'est-à-dire qu'il est normé.

L'instruction FGROUP permet de sélectionner, parmi les axes à interpolation, ceux auxquels doit se rapporter l'avance tangentielle programmée. Une interpolation polynomiale à vitesse constante sur le trajet S de ces axes ne signifie cependant pas en général une variation constante du paramètre de courbe U.

Comportement de la commande en cas de Reset et paramètres machine/d'option

Après un Reset, le code G déterminé par MD20150 $MC_GCODE_RESET_VALUES[44] est actif (45ème groupe de codes G). Pour rester compatible avec des installations existantes, la valeur SPATH est préréglée.

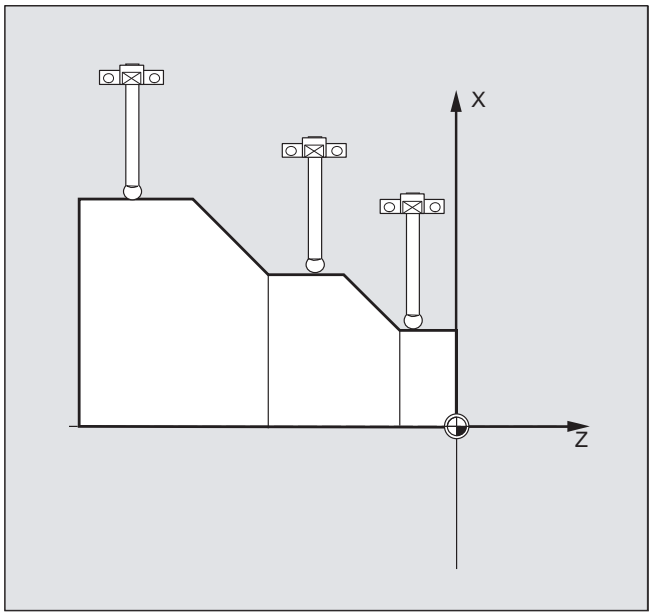
La valeur de position initiale pour le type d'arrondissement est définie par MD20150 $MC_GCODE_RESET_VALUES[9] (10ème groupe de codes G).

Le paramètre machine spécifique à l'axe MD33100 $MA_COMPRESS_POS_TOL[<n>] a une signification étendue : il contient les tolérances pour la fonction compresseur et l'arrondissement avec G642.

4.7 Mesure avec palpeur à déclenchement (MEAS, MEAW)

Fonction

La fonction "Mesure avec palpeur à déclenchement" permet d'accoster des positions réelles sur la pièce, de mesurer les positions de chaque axe programmé dans le bloc de mesure lors du front de déclenchement du palpeur et de les écrire dans la cellule mémoire correspondante.



Programmation de blocs de mesure

Les deux instructions suivantes sont disponibles pour la programmation de la fonction :

- `MEAS`

  L'instruction `MEAS` permet d'effacer la distance restant à parcourir entre la position réelle et la position de consigne.

- `MEAW`

  L'instruction `MEAW` s'utilise pour les tâches de mesure où la position programmée doit impérativement être accostée.

`MEAS` et `MEAW` sont à effet non modal et peuvent être programmées avec des instructions de déplacement. Les avances et modes d'interpolation (G0, G1, ...), de même que le nombre d'axe, doivent être adaptés au problème de mesure concerné.

Lecture des résultats de mesure

Les résultats de mesure pour les axes saisis avec le palpeur de mesure sont disponibles dans les variables suivantes :

- `$AA_MM[<axe>]`
  Résultats de mesure dans le système de coordonnées machine
- `$AA_MW[<axe>]`
  Résultats de mesure dans le système de coordonnées pièce

A la lecture de ces variables, aucun arrêt du prétraitement n'est créé.

Remarque

Avec `STOPRE`, un arrêt du prétraitement doit être programmé dans le programme CN à un emplacement adapté. faute de quoi il y aura lecture de valeurs erronées.

Syntaxe

```
MEAS=<TE> G... X... Y... Z...
MEAW=<TE> G... X... Y... Z...
```

Signification

`MEAS` Instruction : Mesure **avec** effacement de la distance restant à parcourir
Prise d'effet : bloc par bloc (non modale)

`MEAW` Instruction : Mesure **sans** effacement de la distance restant à parcourir
Prise d'effet : bloc par bloc (non modale)

`<TE>` Evénement déclencheur de la mesure
Type : INT
Plage de valeurs : -2, -1, 1, 2

Nota :
il existe au maximum 2 palpeurs de mesure (selon le niveau d'extension).

Signification :

(+)1	Front montant du palpeur de mesure 1 (à l'entrée de mesure 1)
-1	Front descendant du palpeur de mesure 1 (à l'entrée de mesure 1)
(+)2	Front montant du palpeur de mesure 2 (à l'entrée de mesure 2)
-2	Front descendant du palpeur de mesure 2 (à l'entrée de mesure 2)

Nota :
il existe au maximum 2 palpeurs de mesure (selon le niveau d'extension).

`G...` Mode d'interpolation, p. ex. `G0`, `G1`, `G2` ou `G3`

`X... Y... Z...` Point final en coordonnées cartésiennes

Exemple

```
Code de programme                          Commentaire
N10 MEAS=1 G1 F1000 X100 Y730 Z40          ; Bloc de mesure avec palpeur de mesure
                                             de la première entrée de mesure et
                                             interpolation linéaire. Un arrêt du
                                             prétraitement est automatiquement créé.
...
```

Informations complémentaires

État du contrat de mesure

S'il est nécessaire de savoir, dans le programme, si le déclenchement du palpeur a eu lieu ou non, il est possible d'interroger la variable d'état `$AC_MEA[n]` (n = numéro du palpeur) :

Valeur	Signification
0	Tâche de mesure non exécutée
1	Tâche de mesure exécutée correctement (le déclenchement du palpeur a eu lieu)

Remarque

Si le palpeur est dévié dans le programme, la variable est mise à 1. Au début d'un bloc de mesure, la variable est mise automatiquement à l'état initial du palpeur.

Enregistrement des valeurs de mesure

Les positions sont mesurées pour tous les axes à interpolation et axes de positionnement d'un bloc, qui sont déplacés (le nombre maximal d'axes dépend de la configuration de la commande). Avec `MEAS`, le déclenchement du palpeur entraîne un freinage défini du déplacement.

Remarque

Si un axe géométrique est programmé dans un bloc de mesure, les valeurs de mesure sont enregistrées pour tous les axes géométriques courants.

Si un axe concerné par une transformation est programmé dans un bloc de mesure, les valeurs de mesure sont enregistrées pour tous les axes concernés par cette transformation.

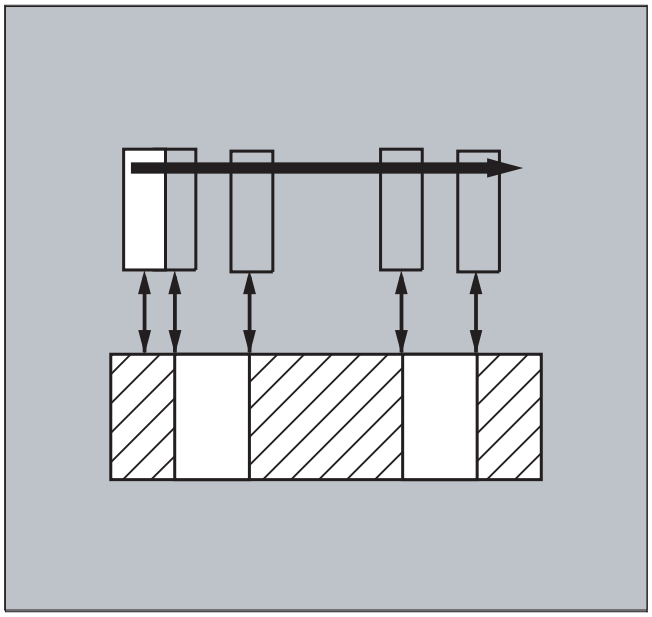
4.8 Fonction de mesure étendue (MEASA, MEAWA, MEAC) (option)

Fonction

Plusieurs types de palpeurs et plusieurs systèmes de mesure peuvent être utilisés pour la mesure axiale.

Les instructions MEASA et MEAWA permettent de saisir à chaque mesure jusqu'à quatre valeurs de mesure pour l'axe programmé et de les ranger dans des variables système en fonction de l'événement déclencheur.

L'instruction MEAC permet d'effectuer des mesures en continu. Les résultats des mesures sont rangés dans des variables FIFO. Pour MEAC, il est également possible de saisir jusqu'à quatre valeurs par mesure.



Lecture des résultats de mesure

Les résultats de mesure sont disponibles dans variables suivantes :

- $AA_MM1...4[<axe>]

  Résultats de mesure dans le système de coordonnées machine
- $AA_MW1...4[<axe>]

  Résultats de mesure dans le système de coordonnées pièce

Syntaxe

```
MEASA[<axe>]=(<mode>,<TE1>,...,<TE4>)
MEAWA[<axe>]=(<mode>,<TE1>,...,<TE4>)
MEAC[<axe>]=(<mode>,<mémoire de mesure>,<TE1>,...,<TE4>)
```

Remarque

MEASA et MEAWA ont un effet non modal et peuvent être programmées dans un bloc. Par contre, si vous programmez MEASA/MEAWA avec MEAS/MEAW dans un même bloc, un message d'erreur sera émis.

Signification

MEASA
Instruction : Mesure axiale avec effacement de la distance restant à parcourir
Prise d'effet : bloc par bloc (non modale)

MEAWA
Instruction : Mesure axiale sans effacement de la distance restant à parcourir
Prise d'effet : bloc par bloc (non modale)

MEAC
Instruction : Mesure axiale continue sans effacement de la distance restant à parcourir
Prise d'effet : bloc par bloc (non modale)

<Axe>
Nom de l'axe de canal utilisé pour la mesure

<Mode>
Nombre à deux chiffres pour l'indication du mode de fonctionnement (mode de mesure et système de mesure)

Mode de mesure (unité) :

0 annulation du contrat de mesure.

1 Jusqu'à 4 événements déclencheurs différents, activables simultanément.

2 Jusqu'à 4 événements déclencheurs activables successivement.

3 Jusqu'à 4 événements déclencheurs activables successivement, cependant sans surveillance de l'événement déclencheur 1 au démarrage (inhibition des alarmes 21700/21703).

Nota :
ce mode n'est pas possible avec MEAC.

Système de mesure (dizaine) :

0 (ou aucune indication) système de mesure actif
1 système de mesure 1
2 système de mesure 2
3 les deux systèmes de mesure

<TE>
Evénement déclencheur de la mesure

Type : INT

Plage de valeurs : -2, -1, 1, 2

Signification :

(+)1 Front montant du palpeur 1
-1 Front descendant du palpeur 1
(+)2 Front montant du palpeur 2
-2 Front descendant du palpeur 2

<mémoire de mesure>
Numéro de la FIFO (mémoire à file d'attente)

Exemples

Exemple 1 : mesure axiale avec effacement de la distance restant à parcourir en mode 1 (évaluation dans l'ordre chronologique)

a) avec 1 système de mesure

```
Code de programme                          Commentaire
...
N100 MEASA[X] = (1,1,-1) G01 X100 F100  ; Mesure en mode 1 avec système de mesure
                                          actif. Attente du signal de mesure avec
                                          front montant/descendant du palpeur 1 sur
                                          le trajet vers X=100.
N110 STOPRE                             ; l'arrêt du prétraitement des blocs,
N120 IF $AC_MEA[1]==FALSE GOTOF ENDE    ; Vérification du succès de la mesure.
N130 R10=$AA_MM1[X]                     ; Mémorisation de la valeur de mesure
                                          correspondant au premier événement
                                          déclencheur programmé (front montant).
N140 R11=$AA_MM2[X]                     ; Mémorisation de la valeur de mesure
                                          correspondant au deuxième événement
                                          déclencheur programmé (front descendant).
N150 FIN :
```

b) avec 2 systèmes de mesure

```
Code de programme                          Commentaire
...
N200 MEASA[X]=(31,1,-1) G01 X100 F100   ; Mesure en mode 1 avec les deux systèmes de
                                          mesure. Attente du signal de mesure avec
                                          front montant/descendant du palpeur 1 sur
                                          le trajet vers X=100.
N210 STOPRE                             ; l'arrêt du prétraitement des blocs,
N220 IF $AC_MEA[1]==FALSE GOTOF ENDE    ; Vérification du succès de la mesure.
N230 R10=$AA_MM1[X]                     ; Mémorisation de la valeur de mesure du
                                          système de mesure 1 en cas de front
                                          montant.
N240 R11=$AA_MM2[X]                     ; Mémorisation de la valeur de mesure du
                                          système de mesure 2 en cas de front
                                          montant.
N250 R12=$AA_MM3[X]                     ; Mémorisation de la valeur de mesure du
                                          système de mesure 1 en cas de front
                                          descendant.
N260 R13=$AA_MM4[X]                     ; Mémorisation de la valeur de mesure du
                                          système de mesure 2 en cas de front
                                          descendant.
N270 FIN:
```

Exemple 2 : mesure axiale avec effacement de la distance restant à parcourir en mode 2 (évaluation dans l'ordre de programmation)

Code de programme	Commentaire
...	
N100 MEASA[X] = (2,1,-1,2,-2) G01 X100 F100	; Mesure en mode 2 avec système de mesure actif. Attente du signal de mesure dans l'ordre suivant : front montant du palpeur 1, front descendant du palpeur 1, front montant du palpeur 2, front descendant du palpeur 2 sur le trajet vers X=100.
N110 STOPRE	; l'arrêt du prétraitement des blocs,
N120 IF $AC_MEA[1]==FALSE GOTOF PALPEUR2	; Vérification du succès de la mesure avec palpeur 1.
N130 R10=$AA_MM1[X]	; Mémorisation de la valeur de mesure correspondant au premier événement déclencheur programmé (front montant du palpeur 1).
N140 R11=$AA_MM2[X]	; Mémorisation de la valeur de mesure correspondant au deuxième événement déclencheur programmé (front montant du palpeur 1).
N150 PALPEUR2:	
N160 IF $AC_MEA[2]==FALSE GOTOF ENDE	; Vérification du succès de la mesure avec palpeur 2.
N170 R12=$AA_MM3[X]	; Mémorisation de la valeur de mesure correspondant au troisième événement déclencheur programmé (front montant du palpeur 2).
N180 R13=$AA_MM4[X]	; Mémorisation de la valeur de mesure correspondant au quatrième événement déclencheur programmé (front montant du palpeur 2).
N190 FIN:	

Exemple 3 : mesure axiale continue en mode 1 (évaluation dans l'ordre chronologique)

a) Mesure de 100 valeurs ou moins

```
Code de programme                                   Commentaire
...
N110 DEF REAL VAL_MESURE[100]
N120 DEF INT boucle=0
N130 MEAC [X] = (1,1,-1) G01 X1000 F100             ; Mesure en mode 1 avec système de mesure actif,
                                                      mémorisation des valeurs de mesure dans $AC_FIFO1,
                                                      attente du signal de mesure avec front descendant du
                                                      palpeur 1 sur le trajet vers X=1000.
N135 STOPRE
N140 MEAC[X]=(0)                                    ; Abandon de la mesure dès l'accostage de la position
                                                      d'axe.
N150 R1=$AC_FIFO1[4]                                ; Mémorisation du nombre de valeurs de mesure
                                                      accumulées dans le paramètre R1.
N160 FOR boucle=0 TO R1-1
N170 VAL_MESURE[boucle] = $AC_FIFO1[0]              ; Lecture et mémorisation des valeurs de mesure
                                                      rangées dans $AC_FIFO1.
N180 ENDFOR
```

b) Mesure avec effacement de la distance restant à parcourir après 10 valeurs de mesure

```
Code de programme                                                   Commentaire
...
N10 WHEN $AC_FIFO1[4]>=10 DO MEAC[x]=(0) DELDTG(x)                  ; Effacer la distance
                                                                      restant à parcourir.
N20 MEAC[x]=(1,1,1,-1) G01 X100 F500
N30 MEAC [X]=(0)
N40 R1=$AC_FIFO1[4]                                                 ; Nombre de valeurs de
                                                                      mesure.
...
```

Informations complémentaires

Contrat de mesure

La programmation d'une tâche de mesure peut avoir lieu dans le programme pièce ou à partir d'une action synchrone (voir chapitre "Action synchrone au déplacement"). On ne peut activer qu'un seul contrat de mesure par axe à un moment donné.

Remarque

L'avance est à adapter au problème de mesure posé.

Dans le cas de `MEASA` et `MEAWA` , des résultats corrects sont garantis uniquement avec des avances pour lesquelles il ne survient pas plus d'un même événement déclencheur et pas plus de quatre événements déclencheurs différents par période d'échantillonnage de l'asservissement de position.

Dans le cas de la mesure continue avec `MEAC` , le rapport entre la période d'interpolation et la période d'échantillonnage de l'asservissement de position ne doit pas être supérieur à 8:1.

Evénement déclencheur

Un événement déclencheur se compose du numéro du palpeur de mesure et du critère de déclenchement du signal de mesure (front montant ou descendant).

Pour chaque mesure, le programme peut traiter respectivement jusqu'à 4 événements déclencheurs des palpeurs, autrement dit jusqu'à deux palpeurs avec deux fronts de mesure chacun. L'ordre du traitement et le nombre maximal de facteurs de déclenchement dépendent du mode sélectionné.

Remarque

Règle valable pour le mode de mesure 1 : un même événement de déclenchement ne doit être programmé qu'une seule fois dans chaque tâche de mesure !

Mode de fonctionnement

Le premier chiffre (dizaines) du mode de fonctionnement permet de sélectionner le système de mesure souhaité. S'il n'existe qu'un seul système de mesure, alors que le second a été programmé, c'est automatiquement le système de mesure existant qui sera pris en compte.

Le second chiffre (unités) permet de sélectionner le mode de mesure souhaité. La procédé de mesure s'adapte ainsi aux possibilités de la commande respective :

- **Mode 1**

  Les événements déclencheurs sont traités dans l'ordre chronologique de leur apparition. Dans ce mode, quand on met en œuvre des cartes 6 axes, on ne peut programmer qu'un seul événement déclencheur ou alors, si on en programme plusieurs, il y a commutation automatique dans le mode 2 (sans avertissement préalable).

- **Mode 2**

  Les événements déclencheurs sont traités dans l'ordre de leur programmation.

- **Mode 3**

  Les événements déclencheurs sont traités dans l'ordre de leur programmation, cependant aucune surveillance de l'événement déclencheur 1 au démarrage n'est réalisée.

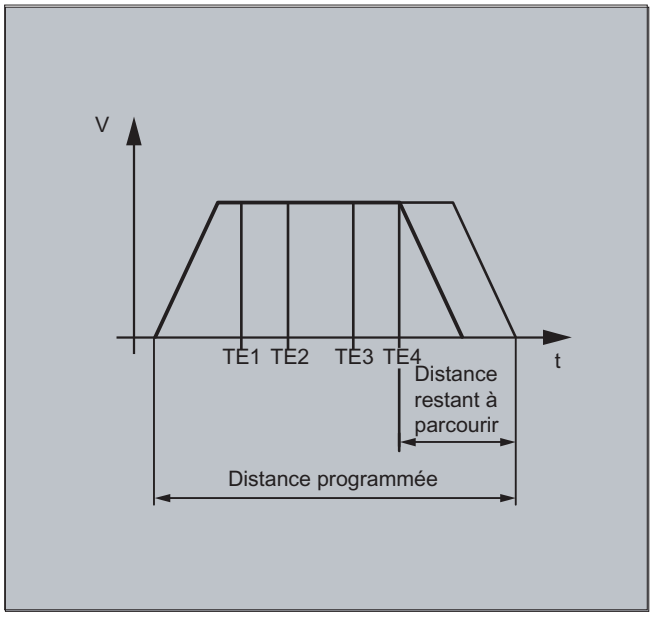
Remarque

Si vous travaillez avec 2 systèmes de mesure, seulement deux événements déclencheurs sont programmables.

Mesure avec et sans effacement de la distance restant à parcourir

Quand vous programmez `MEASA`, l'effacement de la distance restant à parcourir est effectué seulement après la saisie de toutes les valeurs de mesure requises.

Pour des tâches de mesure spéciales où la position programmée doit être accostée impérativement, utilisez la fonction `MEAWA`.



Remarque

MEASA n'est pas programmable dans des actions synchrones. Si on le souhaite, on peut programmer MEAWA plus effacement de la distance restant à parcourir en tant qu'action synchrone.

Quand le contrat de mesure est lancé avec MEAWA à partir d'une action synchrone, les valeurs mesurées sont disponibles uniquement dans le système de coordonnées machine.

Résultats de mesure pour MEASA, MEAWA

Les résultats de mesure sont disponibles sous les variables suivantes :

- dans le système de coordonnées machine :

$AA_MM1[<axe>]	Valeur de mesure du système de mesure programmé pour l'événement déclencheur 1
...	...
$AA_MM4[<axe>]	Valeur de mesure du système de mesure programmé pour l'événement déclencheur 4

- dans le système de coordonnées pièce :

$AA_WM1[<axe>]	Valeur de mesure du système de mesure programmé pour l'événement déclencheur 1
...	...
$AA_WM4[<axe>]	Valeur de mesure du système de mesure programmé pour l'événement déclencheur 4

Remarque

A la lecture de ces variables, aucun arrêt du prétraitement n'est créé. Avec STOPRE, un arrêt du prétraitement doit être programmé à un emplacement adapté. faute de quoi il y aura lecture de valeurs erronées.

Axes géométriques / transformations

Si la mesure axiale doit être lancée pour un axe géométrique, il convient de programmer de façon explicite le même contrat de mesure pour tous les autres axes géométriques. Il en va de même pour les axes qui participent à une transformation.

Exemple :

```
N10 MEASA[Z]=(1,1) MEASA[Y]=(1,1) MEASA[X]=(1,1) G0 Z100
```

ou

```
N10 MEASA[Z]=(1,1) POS[Z]=100
```

Contrat de mesure avec 2 systèmes de mesure

Lorsqu'un contrat de mesure est exécuté avec deux systèmes de mesure, chacun des deux événements déclencheurs possibles est saisi par les deux systèmes de mesure de l'axe en question. Les variables réservées sont ainsi affectées :

`$AA_MM1[<axe>]`	ou	`$AA_MW1[<axe>]`	Valeur de mesure du système de mesure 1 pour l'événement déclencheur 1
`$AA_MM2[<axe>]`	ou	`$AA_MW2[<axe>]`	Valeur de mesure du système de mesure 2 pour l'événement déclencheur 1
`$AA_MM3[<axe>]`	ou	`$AA_MW3[<axe>]`	Valeur de mesure du système de mesure 1 pour l'événement déclencheur 2
`$AA_MM4[<axe>]`	ou	`$AA_MW4[<axe>]`	Valeur de mesure du système de mesure 2 pour l'événement déclencheur 2

Etat du palpeur de mesure

L'état du palpeur est disponible dans la variable système suivante :

$A_PROBE[<n>]

<n>=palpeur

Valeur	Signification
1	Palpeur actionné
0	Palpeur non actionné

Etat du contrat de mesure MEASA, MEAWA

S'il est nécessaire de connaître, dans le programme, l'état de la tâche de mesure, il peut être interrogé avec $AC_MEA[n], <n> = numéro du palpeur de mesure. Aussitôt que tous les événements déclencheurs du palpeur "n" programmés dans un bloc ont eu lieu, cette variable fournit la valeur 1. Sinon, elle fournit la valeur 0.

Remarque

Si la mesure est lancée à partir d'actions synchrones, $AC_MEA n'est plus actualisé. Dans ce cas, il convient d'interroger les nouveaux signaux d'état de l'AP, DB31, … DBX62.3, ou la variable équivalente $AA_MEAACT[<axe>].

Signification :

$AA_MEAACT==1 : mesure activée

$AA_MEAACT==0 : mesure non activée

Mesure continue (MEAC)

Avec `MEAC` , les valeurs de mesure sont fournies dans le système de coordonnées machine et rangées dans la mémoire FIFO[n] indiquée (mémoire à file d'attente). Quand deux palpeurs ont été configurés pour la mesure, les valeurs de mesure du second palpeur seront rangées séparément dans la mémoire FIFO[n+1] configurée à cet effet (par le biais des PM).

La mémoire FIFO est une mémoire à file d'attente dans laquelle on enregistre les valeurs de mesure dans des variables $AC_FIFO suivant le principe "premier entré, premier sorti" (voir chapitre "Actions synchrones au déplacement").

Remarque

Le contenu de la mémoire FIFO ne peut être lu qu'une seule fois. Si vous souhaitez pouvoir réutiliser les valeurs mesurées, mémorisez-les temporairement dans les données utilisateur.

Dès que le nombre de mesurées destinées à la mémoire FIFO dépasse le nombre maximum prévu dans les paramètres machine, il est mis fin automatiquement à la mesure.

Pour réaliser une mesure sans fin, programmez la lecture cyclique des valeurs de mesure. La lecture doit être effectuée au moins à la même fréquence que l'entrée de nouvelles valeurs de mesure.

Erreurs de programmation détectées

Les erreurs de programmation suivantes sont détectées et signalées par un message d'erreur :

- MEASA/MEAWA programmées avec MEAS/MEAW dans un même bloc

  Exemple :

  ```
  N01 MEAS=1 MEASA[X]=(1,1) G01 F100 POS[X]=100
  ```

- MEASA/MEAWA avec un nombre de paramètres <2 ou >5

  Exemple :

  ```
  N01 MEAWA[X]=(1) G01 F100 POS[X]=100
  ```

- MEASA/MEAWA avec événement déclencheur différent de 1/ -1/ 2/ -2

  Exemple :

  ```
  N01 MEASA[B]=(1,1,3) B100
  ```

- MEASA/MEAWA avec mode incorrect

  Exemple :

  ```
  N01 MEAWA[B]=(4,1) B100
  ```

- MEASA/MEAWA avec événement déclencheur programmé deux fois

  Exemple :

  ```
  N01 MEASA[B]=(1,1,-1,2,-1) B100
  ```

- MEASA/MEAWA et axe GEO manquant

  Exemple :

  ```
  N01 MEASA[X]=(1,1) MEASA[Y]=(1,1) G01 X50 Y50 Z50 F100 ;axe GEOX/Y/Z
  ```

- Contrat de mesure hétérogène pour axes géométriques

  Exemple :

  ```
  N01 MEASA[X]=(1,1) MEASA[Y]=(1,1) MEASA[Z]=(1,1,2) G01 X50 Y50 Z50 F100
  ```

4.9 Fonctions spéciales pour l'utilisateur OEM (OEMIPO1, OEMIPO2, G810 à G829)

Fonction

Adresses OEM

L'utilisateur OEM définit la signification des adresses OEM. La fonctionnalité est entrée par le biais de cycles de compilation. Cinq adresses OEM sont réservées. Les descripteurs d'adresse peuvent être spécifiés. Les adresses OEM sont admises dans chaque bloc.

Paramètres

Groupes de fonctions G réservés

Groupe 1 avec `OEMIPO1`, `OEMIPO2`

L'utilisateur OEM peut définir deux noms supplémentaires des fonctions G `OEMIPO1`, `OEMIPO2` . La fonctionnalité est entrée par le biais de cycles de compilation et elle est réservée à l'utilisateur OEM.

- Groupe 31 avec **G810 à G819**
- Groupe 32 avec **G820 à G829**

Deux groupes de fonctions G comptant chacun dix fonctions G OEM sont réservés à l'utilisateur OEM. **Ceci permet de sortir, pour une utilisation externe, les fonctions introduites par l'utilisateur OEM.**

Fonctions et sous-programmes

En outre, l'utilisateur OEM peut créer des fonctions et des sous-programmes prédéfinis avec transfert de paramètres.

4.10 Réduction de l'avance avec décélération aux angles (FENDNORM, G62, G621)

Fonction

Lors de la décélération automatique aux angles, l'avance est réduite juste avant l'angle concerné selon une courbe en forme de cloche. En outre, l'étendue du comportement d'outil pertinent pour l'usinage peut être paramétrée au moyen des données de réglage. Ce sont :

- Début et fin de la réduction de l'avance
- Correction de vitesse avec laquelle l'avance est réduite
- Identification de l'angle pertinent

En tant qu'angles pertinents, on prend en considération les angles dont l'angle interne est inférieur à l'angle paramétré par l'intermédiaire de la donnée de réglage .

La valeur par défaut `FENDNORM` désactive la fonction de correction automatique aux angles.

Bibliographie :
/FBA/ Description fonctionnelle Dialectes ISO

Syntaxe

```
FENDNORM
G62 G41
G621
```

Signification

`FENDNORM`	Décélération automatique aux angles désactivée
`G62`	Décélération aux angles intérieurs avec correction du rayon d'outil activée
`G621`	Décélération à tous les angles avec correction du rayon d'outil activée

G62 intervient seulement au niveau des angles internes avec

- correction active du rayon d'outil `G41`, `G42` et
- contournage actif `G64`, `G641`

L'angle correspondant est accosté avec l'avance réduite qui résulte de :

F * (correction de vitesse pour la réduction de l'avance) * correction de vitesse de l'avance

La réduction maximale de l'avance peut être atteinte lorsque l'outil doit assurer, par rapport à la trajectoire du centre de l'outil, le changement de direction vers l'angle concerné.

G621 intervient comme G62 au niveau de chaque angle formé par les axes définis avec `FGROUP`.

4.11 Critère programmable de fin de déplacement (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA)

Fonction

De façon analogue aux critères de changement de bloc en interpolation de trajectoire (G601, G602 et G603), vous pouvez programmer un critère de fin de déplacement en interpolation d'axes individuels dans un programme pièce ou une action synchrone pour axes de commande/AP.

Selon le réglage du critère de fin de déplacement, les blocs de programme pièce ou de cycle technologique comportant des déplacements d'axes individuels se terminent plus ou moins rapidement. Cela vaut également pour l'AP via FC15/16/18.

Syntaxe

```
FINEA[<axe>]
COARSEA[<axe>]
IPOENDA[<axe>]
IPOBRKA(<axe>[,<instant>])
ADISPOSA(<axe>[,<mode>,<taille de fenêtre>])
```

Signification

`FINEA` :	Fin du déplacement lorsque "Arrêt précis fin" est atteint
`COARSEA` :	Fin du déplacement lorsque "Arrêt précis grossier" est atteint
`IPOENDA` :	Fin de déplacement lorsque "Arrêt d'interpolateur" est atteint
`IPOBRKA` :	Changement de bloc possible dans la rampe de freinage
`ADISPOSA` :	Taille de la fenêtre de tolérance du critère de fin de déplacement
`<axe>` :	Nom d'axe de canal (X, Y, ...)
`<instant>` :	Instant du changement de bloc en % de la rampe de freinage
`<mode>` :	Mode Type : INT Plage de valeurs : 0 Fenêtre de tolérance pas active 1 Fenêtre de tolérance rapportée à la consigne de position 2 Fenêtre de tolérance rapportée à la position réelle
`<taille de fenêtre>` :	Taille de la fenêtre de tolérance Cette valeur est rangée dans la donnée de réglage SD43610 $SA_ADISPOSA_VALUE en synchronisation avec l'exécution des blocs. Type : REAL

Exemples

Exemple 1 : fin de déplacement lorsque l'arrêt d'interpolateur est atteint

```
Code de programme                                                    Commentaire
...
N110 G01 POS[X]=100 FA[X]=1000 ACC[X]=90 IPOENDA[X]                  Déplacement sur la
                                                                     position X100 avec
                                                                     une vitesse
                                                                     tangentielle de 1000
                                                                     tr/min avec une
                                                                     valeur
                                                                     d'accélération de
                                                                     90% et une fin de
                                                                     déplacement lorsque
                                                                     l'arrêt
                                                                     d'interpolateur est
                                                                     atteint.
...
N120 EVERY $A_IN[1] DO POS[X]=50 FA[X]=2000 ACC[X]=140 IPOENDA[X]    ; Déplacement sur la
                                                                     position X50,
                                                                     lorsque l'entrée 1
                                                                     est active, avec une
                                                                     vitesse tangentielle
                                                                     de 2000 tr/min avec
                                                                     une valeur
                                                                     d'accélération de
                                                                     140% et une fin de
                                                                     déplacement lorsque
                                                                     l'arrêt
                                                                     d'interpolateur est
                                                                     atteint.
...
```

Exemple 2 : critère de changement de bloc dans la rampe de freinage dans le programme pièce

```
Code de programme          Commentaire
                           ; Prise d'effet des paramètres par défaut
N40 POS[X]=100             ; Le changement de bloc s'effectue dès que l'axe X a atteint la position
                             100 avec arrêt précis fin.
N20 IPOBRKA(X,100)         ; Activation du critère de changement de bloc "Rampe de freinage".
N30 POS[X]=200             ; Le changement de bloc a lieu dès que l'axe X commence à freiner.
N40 POS[X]=250             ; L'axe X ne freine pas à la position 200, mais poursuit jusqu'à la
                             position 250 et le changement de bloc s'amorce dès que l'axe X commence
                             à freiner.
N50 POS[X]=0               ; L'axe X freine et revient à la position 0, le changement de bloc
                             s'effectue à la position 0 avec arrêt précis fin.
N60 X10 F100
N70 M30
...
```

Exemple 3 : critère de changement de bloc dans la rampe de freinage dans les actions synchrones

```
Code de programme    Commentaire
                     ; Dans le cycle technologique :
FINEA                ; Critère de fin de déplacement "Arrêt précis fin".
POS[X]=100           ; Le changement de bloc de cycle technologique s'effectue dès que l'axe X a atteint la
                       position 100 avec arrêt précis fin.
IPOBRKA(X,100)       ; Activation du critère de changement de bloc "Rampe de freinage".
POS[X]=100           ; POS[X]=100 ; le changement de bloc de cycle technologique a lieu dès que l'axe X
                       commence à freiner.
POS[X]=250           ; L'axe X ne freine pas à la position 200, mais poursuit jusqu'à la position 250 et le
                       changement de bloc a lieu dans le cycle technologique dès que l'axe X commence à
                       freiner.
POS[X]=250           ; L'axe X freine et revient à la position 0, le changement de bloc s'effectue à la
                       position 0 avec arrêt précis fin.
M17
```

Informations complémentaires

Lecture du critère de fin de déplacement

Le critère de fin de déplacement réglé peut être interrogé avec la variable système$AA_MOTEND[<axe>] :

Valeur	Signification
1	Fin du déplacement avec "Arrêt précis fin"
2	Fin du déplacement avec "Arrêt précis grossier"
3	Fin du déplacement avec "Arrêt IPO"
4	Critère de changement de bloc dans la rampe de freinage de l'axe en mouvement
5	Changement de bloc dans la rampe de freinage avec fenêtre de tolérance rapportée à la "consigne de position”
6	Changement de bloc dans la rampe de freinage avec fenêtre de tolérance rapportée à la "position réelle”

Remarque

Après RESET, la dernière valeur programmée est conservée.

Critère de changement de bloc dans la rampe de freinage

La valeur en pourcentage est inscrite dans la donnée de réglage de façon synchrone à l'exécution :

SD43600 $SA_IPOBRAKE_BLOCK_EXCHANGE

Quand aucune valeur n'est transférée, c'est la valeur courante de la donnée de réglage qui est active.

La valeur est réglable dans une plage de 0 % à 100 %.

Fenêtre de tolérance additionnelle pour IPOBRKA

Au critère de changement de bloc dans la rampe de freinage, peut venir s'ajouter une "fenêtre de tolérance" additionnelle pour le critère de changement de bloc. Le déblocage s'effectue uniquement si l'axe :

- a atteint, comme jusqu'ici, la valeur en % de la rampe de freinage qui a été prescrite

  et

- si sa valeur réelle courante ou sa consigne de position n'est pas éloignée de la position finale dans le bloc d'une valeur supérieure à la tolérance prescrite ici.

Bibliographie

Pour plus d'informations sur le critère de changement de bloc des axes de positionnement, référez-vous à la :

- Description fonctionnelle Fonctions d'extension ; axes de positionnement (P2)
- Manuel de programmation Notions de base, chapitre "Régulation par avance"

4.12 Jeu de paramètres servo programmable (SCPARA)

Fonction

Avec SCPARA, vous pouvez définir le jeu de paramètres constitué de PM dans un programme pièce et des actions synchrones (jusqu'à présent, uniquement possible par le biais de l'AP).

DB3n DBB9 Bit3

Pour éviter tout conflit entre demande AP et CN, un bit supplémentaire a été défini dans l'interface AP → NCK.

DB3n DBB9 Bit3 ”Définition jeu de paramètres avec SCPARA bloquée".

Dans le cas où la définition du jeu de paramètres avec SCPARA est bloquée, il n'y a pas de message d'erreur si la fonction est malgré tout programmée.

Syntaxe

```
SCPARA[<axe>]=<valeur>
```

Signification

SCPARA	Définition du jeu de paramètres
<Axe>	Nom d'axe de canal (X, Y, ...)
<Valeur>	Jeu de paramètres désiré (1<= valeur <=6)

Remarque

Le jeu de paramètres actuel peut être interrogé avec la variable système $AA_SCPAR[<axe>].

Pour les fonctions G33, G331 ou G332, la commande sélectionne le jeu de paramètres le plus approprié.

Si le **changement** du **jeu de paramètres servo** doit avoir lieur aussi bien dans un programme pièce ou une action synchrone que dans l'AP, le programme AP utilisateur doit être étendu.

Bibliographie :
/FB1/ Manuel de fonctions de base ; Avances (V1),
Chapitre "Influence sur l'avance".

Exemple

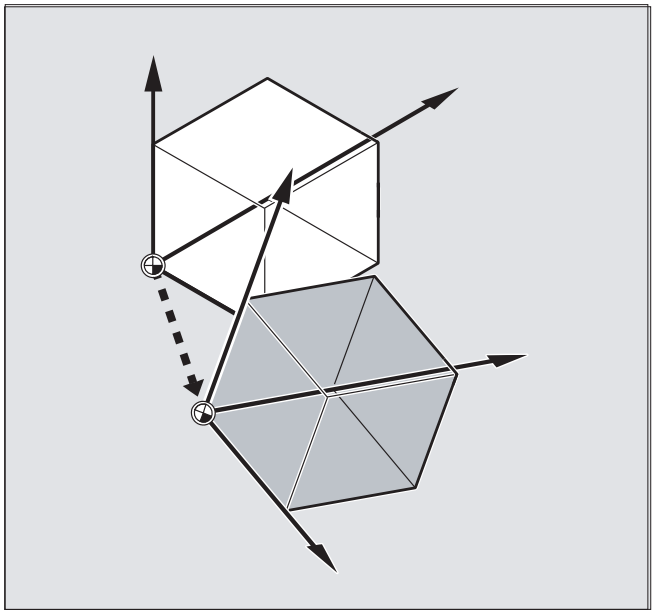
```
Code de programme      Commentaire
...
N110 SCPARA[X]= 3      ; Le 3ème jeu de paramètres est sélectionné pour l'axe X.
...
```

Transformations de coordonnées (FRAMES) 5

5.1 Transformation de coordonnées par variables frames

Fonction

En plus des possibilités précédemment décrites dans le manuel de programmation "Notions de base", il est possible de déterminer des systèmes de coordonnées également avec des variables frames prédéfinies.



Les systèmes de coordonnées suivants sont définis :

SCM : système de coordonnées machine

SCB : système de coordonnées de base

SBR : système de coordonnées de base réglable

SPR : système de coordonnées pièce réglable

SCP : système de coordonnées pièce

Qu'est-ce qu'une variable frame prédéfinie ?

Les variables frames prédéfinies sont des mots-clés dont l'effet est déjà défini dans le langage de la commande et qui peuvent être traitées dans le programme CN.

Variables frames possibles :

- Frame de base (décalage de base)
- frames réglables
- frame programmable

Lecture des affectations de valeurs et des valeurs réelles

Corrélation entre variables frames et frames

Vous pouvez activer une transformation de coordonnées en affectant un frame à une variable frame.

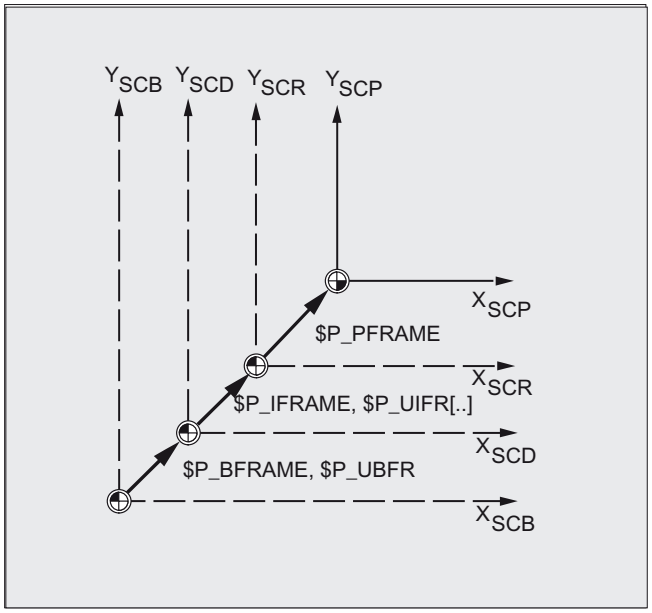
Exemple : `$P_PFRAME=CTRANS(X,10)`

Variable frame :

`$P_PFRAME` signifie : frame programmable courant.

Frame :

`CTRANS(X,10)` signifie : décalage d'origine programmable de l'axe X de 10 mm.



Lecture des valeurs réelles

Les valeurs réelles courantes dans les différents systèmes de coordonnées peuvent être lues dans le programme pièce à l'aide de variables prédéfinies :

$AA_IM[axe] : Lecture valeur réelle dans SCM

$AA_IB[axe] : Lecture valeur réelle dans SCB

$AA_IBN[axe] : Lecture valeur réelle dans SCR

$AA_IEN[axe] : Lecture valeur réelle dans SPR

$AA_IW[axe] : Lecture valeur réelle dans SCP

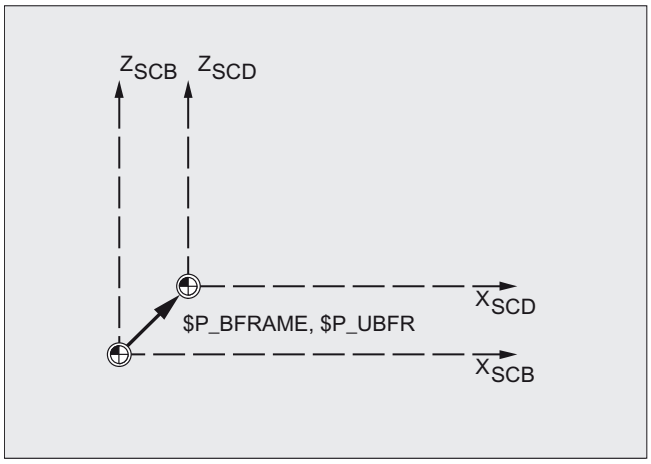
5.1.1 Variables frames prédéfinies ($P_BFRAME, $P_IFRAME, $P_PFRAME, $P_ACTFRAME)

$P_BFRAME

Variable frame de base courant qui établit la corrélation entre le système de coordonnées de base (SCB) et le système de coordonnées de base réglable (SBR).

Si le frame de base défini par $P_UBFR doit être immédiatement actif dans le programme, il faut

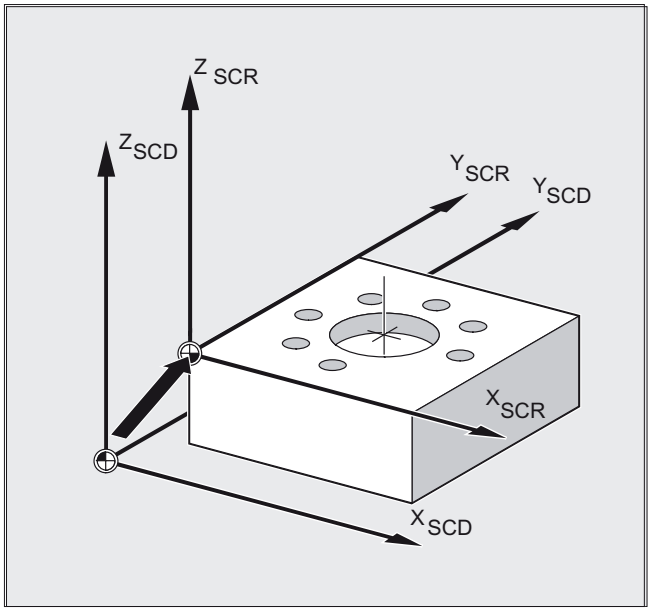
- soit `programmer G500`,G54...G599
- `soit écraser` $P_BFRAME `avec $ $P_UBFR .`



$P_IFRAME

Variable frame réglable courant qui établit la corrélation entre le système de coordonnées de base réglable (SBR) et le système de coordonnées pièce réglable (SPR).

- `$P_IFRAME` correspond à `$P_UIFR[$P_IFRNUM]`
- `$P_IFRAME` contient, après la programmation de G54 par exemple, la translation, la rotation, l'homothétie et la fonction miroir définies par `G54` .

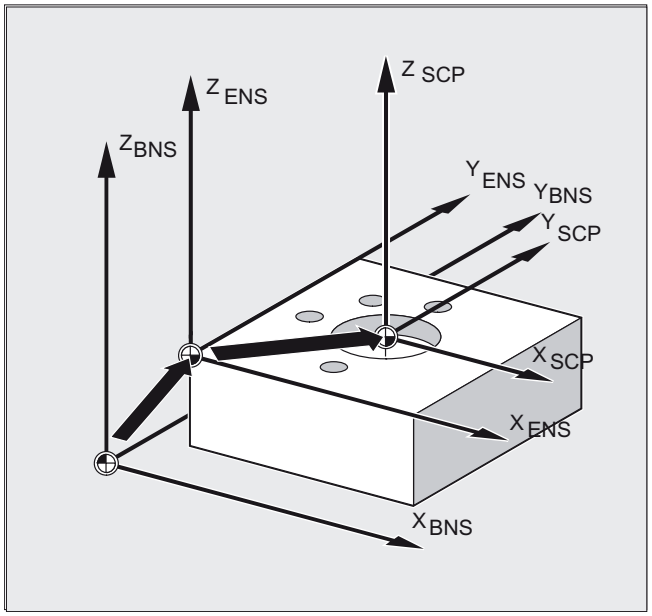


$P_PFRAME

Variable de frame programmable courante qui établit la relation entre le système de coordonnés-pièce réglable (SPR) et le système de coordonnées-pièce (SCP).

`$P_PFRAME` contient le frame résultant, obtenu

- **à partir de la programmation** de `TRANS/ATRANS`, `ROT/AROT`, `SCALE/ASCALE`, `MIRROR/AMIRROR` ou
- **à partir de l'affectation** de `CTRANS`, `CROT`, `CMIRROR`, `CSCALE` au frame programmable.



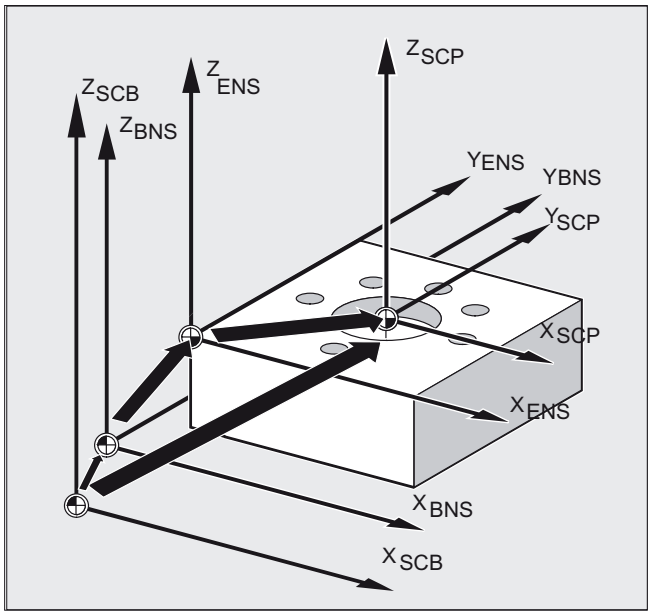
$P_ACTFRAME

Frame résultant courant, obtenu par concaténation

- de la variable frame de base courante `$P_BFRAME`,
- de la variable frame réglable courante `$P_IFRAME` avec frames système et
- de la variable frame programmable courante `$P_PFRAME` avec frames système.

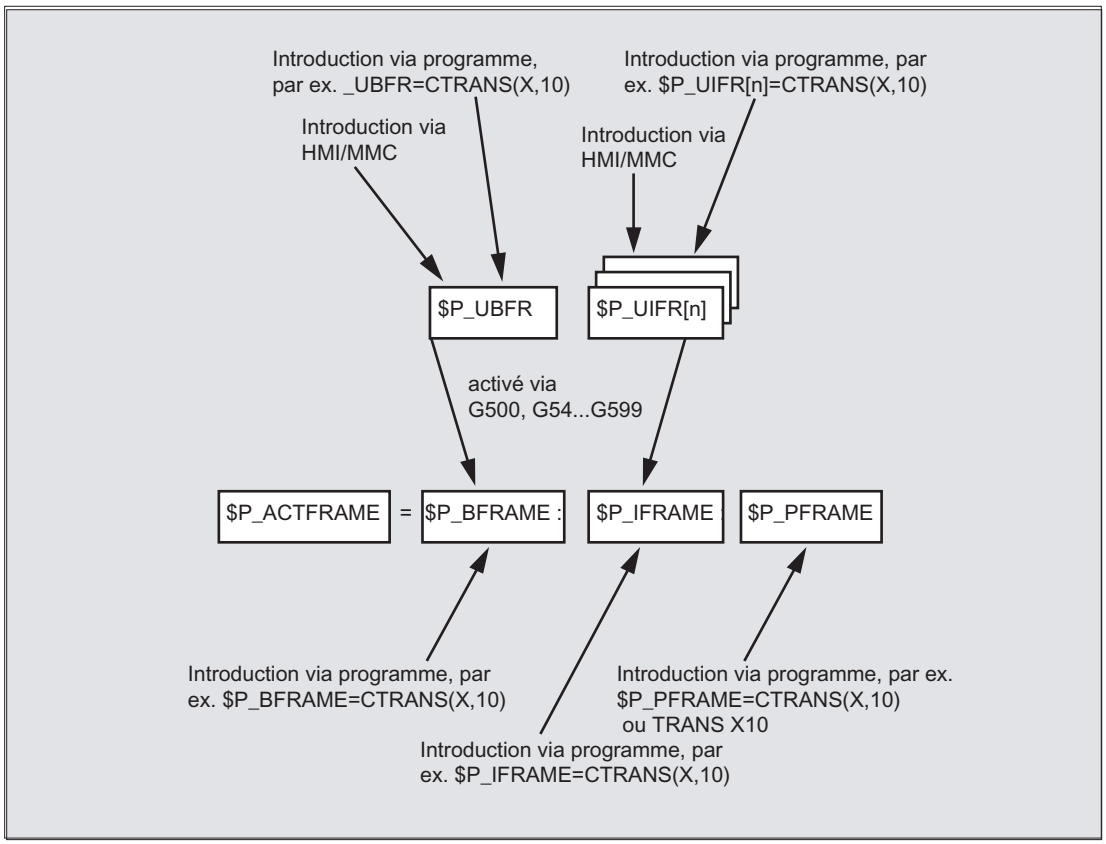
Pour de plus amples informations sur les frames système, voir le chapitre "Frames actifs dans un canal".

`$P_ACTFRAME` décrit l'origine pièce momentanément valide.



Si `$P_BFRAME`, `$P_IFRAME` ou `$P_PFRAME` sont modifiés, `$P_ACTFRAME` est recalculé.

`$P_ACTFRAME` correspond à `$P_BFRAME`:`$P_IFRAME`:`$P_PFRAME`



L'activation du frame de base ou du frame réglable après reset dépend du réglage du PM 20110 RESET_MODE_MASK :

bit0=1, bit14=1 --> `$P_UBFR` (frame de base) activé

bit0=1, bit5=1 --> `$P_UIFR[$P_UIFRNUM]` (frame réglable) activé

Frames de base prédéfinis $P_UBFR

Avec $P_UBFR, vous programmez le frame de base ; celui-ci n'est cependant pas immédiatement actif dans le programme pièce. Le frame de base défini avec $P_UBFR est pris en considération lorsque :

- reset a été effectué, les bits 0 et 14 du PM RESET_MODE_MASK étant à 1,
- les instructions `G500`, `G54...G599` ont été exécutées.

Frames réglables prédéfinis $P_UIFR[n]

Les variables frames prédéfinies $P_UIFR[n] permettent de lire ou d'écrire dans le programme pièce les décalages d'origine réglables `G54` à `G599` .

Ces variables constituent un tableau unidimensionnel de type FRAME nommé `$P_UIFR[n]`.

Mise en correspondance avec les instructions G

La version standard comprend 5 frames réglables `$P_UIFR[0]... $P_UIFR[4]` , c'est-à-dire 5 instructions G équivalentes – `G500` et `G54` à `G57` , sous les adresses desquels des valeurs peuvent être enregistrées.

`$P_IFRAME=$P_UIFR[0]` correspond à `G500`

`$P_IFRAME=$P_UIFR[1]` correspond à `G54`

`$P_IFRAME=$P_UIFR[2]` correspond à `G55`

`$P_IFRAME=$P_UIFR[3]` correspond à `G56`

`$P_IFRAME=$P_UIFR[4]` correspond à `G57`

Le nombre de frames est modifiable par le biais d'un paramètre machine :

`$P_IFRAME=$P_UIFR[5]` correspond à `G505`

`... ... ...`

`$P_IFRAME=$P_UIFR[99]` correspond à `G599`

Remarque

Ceci permet de créer 100 systèmes de coordonnées en tout, qui peuvent être appelés par tous les programmes, par exemple comme origines pour différents montages d'usinage.

⚠ PRUDENCE

La programmation de variables frames et { de frames se fait dans un bloc CN spécifique.
Exception : Programmation d'un frame réglable avec `G54`, `G55` , `...`

5.2 Affectation de valeurs à des variables frames / frames

5.2.1 Affectation de valeurs directes (valeur d'axe, angle, échelle)

Fonction

Vous pouvez affecter directement des valeurs à des frames ou à des variables frames dans le programme CN.

Syntaxe

```
$P_PFRAME=CTRANS (X, valeur d'axe, Y, valeur d'axe, Z, valeur d'axe, …)
$P_PFRAME=CROT (X, angle, Y, angle, Z, angle, …)
$P_UIFR[..]=CROT (X, angle, Y, angle, Z, angle, …)
$P_PFRAME=CSCALE (X, échelle, Y, échelle, Z, échelle, …)
$P_PFRAME=CMIRROR (X, Y, Z)
```

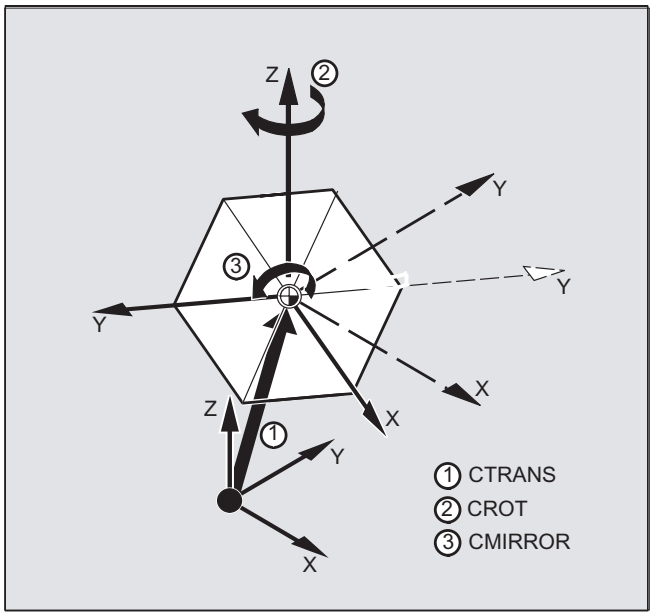
La programmation de $P_BFRAME est analogue à celle de $P_PFRAME.

Signification

`CTRANS`	Décalage (translation) dans les axes indiqués
`CROT`	Rotation autour des axes indiqués
`CSCALE`	Changement d'échelle dans les axes indiqués
`CMIRROR`	Inversion du sens de l'axe indiqué
`X Y Z`	Valeur du décalage en direction de l'axe géométrique indiqué
`Valeur d'axe`	Affectation d'une valeur d'axe au décalage
`Angle`	Affectation d'un angle de rotation autour des axes indiqués
`Echelle`	Changement d'échelle

Exemple

L'affectation des valeurs au frame programmable courant active la translation, la rotation et la fonction miroir.




```
N10 $P_PFRAME=CTRANS(X,10,Y,20,Z,5):CROT(Z,45):CMIRROR(Y)
```

Prérégler des composants de rotation frame avec d'autres valeurs

Avec CROT, prérégler les trois composants UIFR

Code de programme	Commentaire
`$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`	

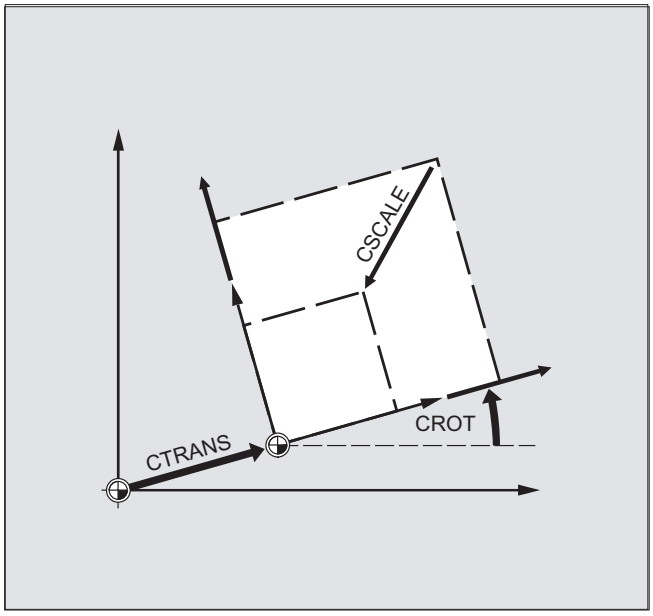
Description

Vous pouvez programmer successivement plusieurs règles opératoires.

Exemple :

```
$P_PFRAME=CTRANS(...):CROT(...):CSCALE...
```

Attention : les instructions doivent être liées entre elles par l'opérateur de concaténation deux-points (…):(…). Les instructions sont ainsi, d'une part, combinées l'une à l'autre et, d'autre part, exécutées de manière additive dans l'ordre programmé.



Remarque

Les valeurs programmées avec les instructions indiquées sont affectées aux frames et mises en mémoire.

Les valeurs ne deviennent actives que lorsqu'elles sont affectées au frame d'une variable frame active `$P_BFRAME` ou `$P_PFRAME` .

5.2.2 Lecture et modification de composantes de frames (TR, FI, RT, SC, MI)

Fonction

Vous avez la possibilité d'accéder à des données **individuelles** d'un frame, par exemple à une valeur de décalage ou à un angle de rotation bien défini. Vous pouvez modifier ces valeurs ou les affecter à une autre variable.

Syntaxe

`R10=$P_UIFR[$P_UIFNUM,X,RT]`	L'angle de rotation RT autour de l'axe X issu du décalage d'origine réglable actuellement valide $P_UIFRNUM doit être affecté à la variable R10.
`R12=$P_UIFR[25,Z,TR]`	La valeur de décalage TR dans Z, issue du bloc de données du frame n° 25 réglé, doit être affectée à la variable R12.
`R15=$P_PFRAME[Y,TR]`	La valeur de décalage TR en Y du frame programmable courant doit être affectée à la variable R15.
`$P_PFRAME[X,TR] = 25`	La valeur de décalage TR en X du frame programmable courant doit être modifiée. X25 s'applique dès à présent.

Signification

`$P_UIFRNUM`	`Cette variable établit automatiquement le lien avec le décalage d'origine réglable actuellement valide.`
`P_UIFR[n,…,…]`	`En indiquant le n° de frame n, vous accédez au frame réglable n° n`
	`Indication de la composante qui doit être lue ou modifiée :`
`TR`	`TR Translation`
`FI`	`FI Translation Fine`
`RT`	`RT Rotation`
`SC`	`SC Scale Changement d'échelle`
`MI`	`MI Fonction miroir`
`X Y Z`	`Est indiqué en outre (voir exemples) l'axe X, Y, Z correspondant.`

Plage de valeurs pour la rotation RT :

Rotation autour du 1er axe géométrique :	-180° à +180°
Rotation autour du 2ème axe géométrique :	-90° à +90°
Rotation autour du 3ème axe géométrique :	-180° à +180°

Description

Appeler le frame

En indiquant la variable système $P_UIFRNUM, vous pouvez accéder directement au décalage d'origine courant, réglé avec $P_UIFR ou G54, G55, …
($P_UIFRNUM contient le n° du frame courant réglé).

Vous appelez tous les autres frames réglables mémorisés $P_UIFR en indiquant le n° { correspondant $P_UIFR[n].

Pour les variables frames prédéfinies et les frames que vous avez définis vous-même, indiquez le nom, par exemple $P_IFRAME.

Appeler les données

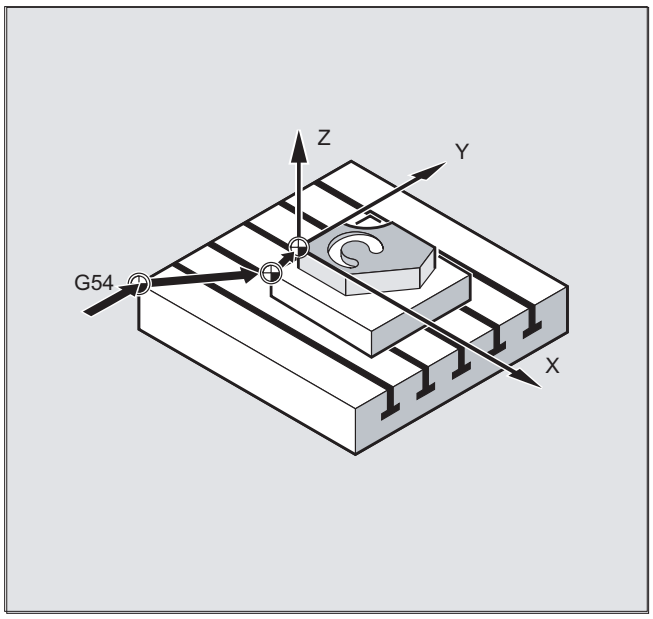
Le nom de l'axe et la composante de frame à laquelle vous voulez accéder ou que vous voulez modifier figurent entre crochets droits, par ex. [X, RT] ou [Z, MI].

5.2.3 Combinaison de frames complets

Fonction

Dans le programme CN, un frame complet peut être affecté à un autre frame ou bien des frames peuvent être concaténés.

On utilise la concaténation de frames pour décrire par exemple plusieurs pièces disposées sur une palette et qui doivent être usinées en un seul cycle.



Pour la description des tâches d'usinage sur palette, les composantes de frame pourraient, par exemple, ne contenir que certaines valeurs partielles dont la concaténation pourraient générer différentes origines pièce.

Syntaxe

Affectation de frames

```
DEF FRAME REGLAGE1                    Les valeurs du frame REGLAGE1, que
REGLAGE1=CTRANS(X,10)                 vous avez défini vous-même, sont
$P_PFRAME=REGLAGE1                    affectées au frame programmable
                                      courant.
DEF FRAME REGLAGE4
REGLAGE4=$P_PFRAME                    Le frame programmable courant est
$P_PFRAME=REGLAGE4                    chargé en mémoire intermédiaire et
                                      renvoyé en mémoire principale, si
                                      nécessaire.
```

Concaténation de frames

Les frames sont concaténés dans l'ordre programmé, les composantes de frame (décalages, rotations, etc.) étant exécutées successivement et de manière additive.

$P_IFRAME=$P_UIFR[15]:$P_UIFR[16]	$P_UIFR[15] contient par exemple des données pour des décalages d'origine. Les données de $P_UIFR[16], par exemple des données pour des rotations, sont traitées ensuite sur cette base.
$P_UIFR[3]=$P_UIFR[4]:$P_UIFR[5]	Le frame réglable 3 est créé par concaténation des frames réglables 4 et 5.

Remarque

N'oubliez pas que les frames doivent être reliés entre eux par l'opérateur de concaténation : (deux-points)

5.2.4 Définition de nouveaux frames (DEF FRAME)

Fonction

Outre les frames réglables prédéfinis décrits jusqu'ici, vous pouvez également créer de nouveaux frames. Il s'agit de variables du type frame auxquelles vous attribuez un nom de votre choix.

Les fonctions CTRANS, CROT, CSCALE et CMIRROR vous permettent d'affecter des valeurs à vos frames dans le programme CN.

Syntaxe

```
DEF FRAME PALETTE1
PALETTE1=CTRANS(…):CROT(…)…
```

Signification

`DEF FRAME`	Création de nouveaux frames.
`PALETTE1`	Nom du nouveau frame
`=CTRANS(...):` `CROT(...)...`	Affectation de valeurs aux fonctions possibles

5.3 Décalage grossier et décalage fin (CFINE, CTRANS)

Fonction

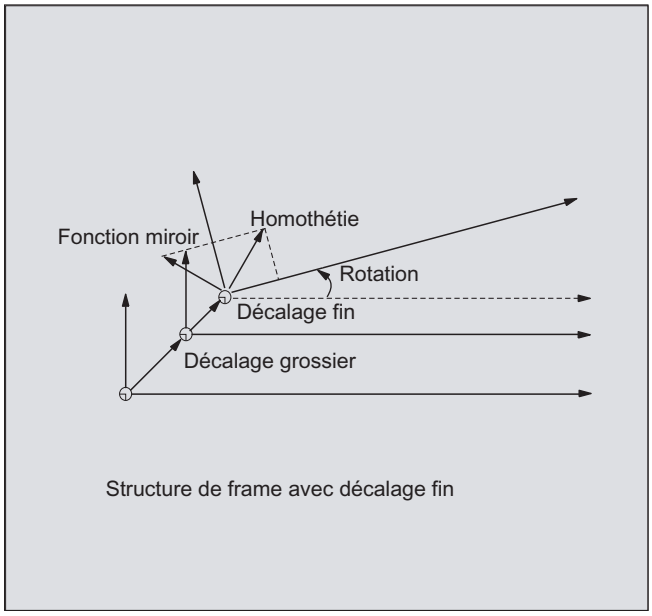
Décalage fin

L'instruction `CFINE(X, ...,Y ...)` permet de programmer un décalage fin du frame de base et de tous les frames réglables.

Un décalage fin n'est possible que si le paramètre machine MD18600 $MN_MM_FRAME_FINE_TRANS=1.

Décalage grossier

CTRANS(...) permet `de définir` le décalage grossier.



Structure de frame avec décalage fin

De la somme du décalage grossier et du décalage fin résulte le décalage total.

Syntaxe

```
$P_UBFR=CTRANS(x, 10) : CFINE(x, 0.1) :
CROT(x, 45)                                  ; Concaténation de décalage,
                                             ; décalage fin et rotation
$P_UIFR[1]=CFINE(x, 0.5 y, 1.0, z, 0.1)      ; L'intégralité du frame est écrasée
                                             avec CFINE
                                             ; y compris le décalage
                                             grossier.
```

L'accès aux composantes décalage fin d'un frame a lieu à l'aide de FI (Translation Fine).

```
DEF REAL FINEX                          ; Définition de la variable FINEX
FINEX=$P_UIFR[$P_UIFNUM, x, FI]         ; Lecture du décalage fin
                                        ; via la variable FINEX
FINEX=$P_UIFR[3, x, FI]$P               ; Lecture du décalage fin
                                        ; de l'axe X dans le 3ème frame
                                        ; via la variable FINEX
```

Signification

`CFINE(x, valeur, y, valeur, z, valeur)`	Décalage fin pour plusieurs axes. Décalage additif (translation).
`CTRANS(x, valeur, y, valeur, z, valeur)`	Décalage grossier pour plusieurs axes. Décalage absolu (translation).
`x y z`	Décalage d'origine des axes (8 max.)
`Valeur`	Composante de translation

Constructeur de la machine-outil

Avec le paramètre machine MD18600 $MN_ MM_FRAME_FINE_TRANS, vous pouvez configurer le décalage fin de deux manières :

0:
le décalage fin ne peut être ni introduit ni programmé. G58 et G59 sont impossibles.

1:
le décalage fin peut être introduit ou programmé pour les frames réglables, les frames de base, les frames programmables, G58 et G59.

Description

Un décalage fin modifié via le tableau de commande HMI n'est actif qu'après activation du frame correspondant, c'est-à-dire après activation par G500, G54...G599. Le décalage fin d'un frame reste actif aussi longtemps que le frame l'est.

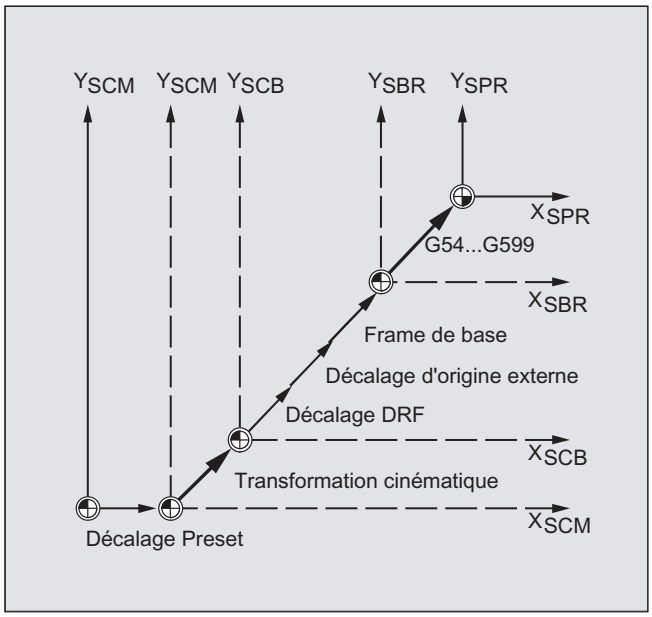
Le frame programmable ne possède pas de composante décalage fin. Si un frame avec décalage fin est affecté au frame programmable, le décalage total de ce dernier résulte de la somme du décalage grossier et du décalage fin. Lors de la lecture du frame programmable, le décalage fin est toujours nul.

5.4 Décalage d'origine externe

Fonction

Il s'agit d'une autre possibilité pour décaler l'origine entre le système de coordonnées de base et le système de coordonnées pièce.

Dans le cas du décalage d'origine externe, vous ne pouvez programmer que des décalages linéaires.



Programmation

La programmation des valeurs de décalage, $AA_ETRANS s'effectue en affectant des valeurs aux variables système spécifiques aux axes.

Affecter une valeur de décalage

```
$AA_ETRANS[axe]=RI
```

RI est la variable de calcul du type REAL qui contient la nouvelle valeur.

Le décalage externe n'est en général pas indiqué dans le programme pièce, mais est spécifié par l'AP.

Remarque

La valeur écrite dans le programme pièce ne prend effet que lorsque le signal correspondant est activé au niveau de l'interface VDI (interface NCU-AP).

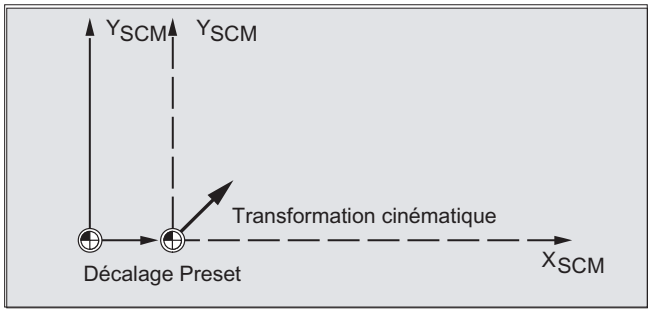
5.5 Décalage Preset (PRESETON)

Fonction

Pour certaines applications spécifiques, il peut être nécessaire d'affecter une nouvelle valeur réelle programmée à la position courante d'un ou de plusieurs axes (à l'arrêt).

⚠PRUDENCE

Avec la fonction PRESETON, le point de référence perd sa validité. Par conséquent, utilisez cette fonction uniquement pour les axes sans point de référence obligatoire. Pour rétablir le système initial, il est nécessaire d'accoster les points de référence avec G74 – voir chapitre "Gestion des fichiers et programmes".



Syntaxe

```
PRESETON(axe, valeur, ...)
```

Signification

`PRESETON`	Préréglage des mémoires de valeurs réelles
`Axe`	Indication de l'axe machine
`Valeur`	nouvelle valeur réelle qui doit s'appliquer à l'axe indiqué

Remarque

Le forçage de la valeur réelle avec des actions synchrones ne devrait s'effectuer qu'avec les mots-clé "WHEN" ou "EVERY".

Exemple

L'affectation des valeurs réelles se fait dans le système de coordonnées machine - les valeurs se rapportent aux axes machine.

```
N10 G0 A760
N20 PRESETON(A1,60)
```

L'axe A accoste la position 760. A la position 760, l'axe machine A1 reçoit la nouvelle valeur réelle 60. { Dès lors, le positionnement s'effectue dans le nouveau système de coordonnées.

5.6 Calcul d'un frame à partir de 3 points mesurés dans l'espace (MEAFRAME)

Fonction

MEAFRAME est une extension du langage 840D pour l'assistance des cycles de mesure.

La fonction MEAFRAME calcule le frame à partir de trois points idéaux et des points mesurés correspondants.

Lorsqu'une pièce est positionnée pour l'usinage, sa position dans le système de coordonnées cartésiennes machine est en général décalée et pivotée par rapport à sa position idéale. Pour un usinage ou une mesure précise, un réglage physique onéreux ou une modification des déplacements dans le programme pièce est nécessaire.

Un frame peut être déterminé par palpage de trois points dans l'espace, dont les positions idéales sont connues. Le palpage s'effectue avec un capteur optique ou à contact qui accoste des trous ou des billes de mesure qui ont été positionnés avec une grande précision sur la plaque support.

Syntaxe

```
MEAFRAME IDEAL_POINT,MEAS_POINT,FIT_QUALITY)
```

Signification

`MEAFRAME`	Calcul d'un frame à partir de 3 points mesurés dans l'espace	
`IDEAL_POINT`	Tableau Real à 2 dim. qui contient les 3 coordonnées des points idéaux	
`MEAS_POINT`	Tableau Real à 2 dim. qui contient les 3 coordonnées des points idéaux	
`FIT_QUALITY`	Variable Real qui fournit les informations suivantes :	
	-1:	Les points idéaux se trouvent approximativement sur une droite : le frame n'a pas pu être calculé. La variable frame en retour comprend un frame neutre.
	-2:	Les points mesurés se trouvent approximativement sur une droite : le frame n'a pas pu être calculé. La variable frame en retour comprend un frame neutre.
	-4:	Le calcul de la matrice de rotation est impossible pour une autre raison.
	Valeur positive :	Somme des distorsions (écarts entre les points) nécessaire pour transférer passer le triangle mesuré dans un triangle congruent au triangle idéal.

Remarque

Qualité de la mesure

Pour que les coordonnées mesurées puissent être mises en relation avec les coordonnées idéales par une rotation/translation combinées, il doit y avoir congruence entre le triangle formé par les points mesurés et le triangle idéal. Ceci est obtenu par un algorithme de compensation qui minimise la somme des carrés des écarts qui permettent de passer du triangle mesuré au triangle idéal.

La distorsion effectivement nécessaire des points mesurés étant le reflet de la qualité de la mesure, elle est fournie par MEAFRAME en tant que variable supplémentaire.

Remarque

Le frame créé par MEAFRAME peut, à l'aide de la fonction ADDFRAME, être transformé en un autre frame de la chaîne de frames.
Voir exemple : Concaténation de frames "Concaténation avec ADDFRAME".

Pour de plus amples informations pour les paramètres de ADDFRAME(FRAME,STRING), voir /FB1/ Manuel de fonctions de base; axes, systèmes de coordonnées, frames (K2), chapitre "Concaténation de frames".

Exemple

Code de programme	Commentaire
	; Programme pièce 1
DEF FRAME CORR_FRAME	

Définition de points de mesure

Programmation	Commentaire
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)	; Pour le test
DEF REAL FIT_QUALITY = 0	
DEF REAL ROT_FRAME_LIMIT = 5	; Permet au maximum une rotation de la position de la pièce de 5 degrés.
DEF REAL FIT_QUALITY_LIMIT = 3	; Permet au maximum un décalage de 3 mm entre le triangle idéal et le triangle mesuré.
DEF REAL SHOW_MCS_POS1[3]	
DEF REAL SHOW_MCS_POS2[3]	
DEF REAL SHOW_MCS_POS3[3]	

Code de programme	**Commentaire**
N100 G01 G90 F5000	
N110 X0 Y0 Z0	
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	; Limitation du 1e angle RPY
SETAL(65020)	
GOTOF NO_FRAME	
ENDIF	
N260 IF CORR_FRAME[Y,RT] > ROT_FRAME_LIMIT	; Limitation du 2e angle RPY
SETAL(65021)	
GOTOF NO_FRAME	
ENDIF	
N270 IF CORR_FRAME[Z,RT] > ROT_FRAME_LIMIT	; Limitation du 3e angle RPY
SETAL(65022)	
GOTOF NO_FRAME	
ENDIF	
N300 $P_IFRAME=CORR_FRAME	; Activation du frame de balayage par un frame réglable ; Vérifier le frame en positionnant les axes géométriques sur les points idéaux
N400 X=IDEAL_POINT[0,0] Y=IDEAL_POINT[0,1] Z=IDEAL_POINT[0,2]	
N410 SHOW_MCS_POS1[0]=$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]	

Code de programme	Commentaire
N620 SHOW_MCS_POS3[1]=$AA_IM[Y]	
N630 SHOW_MCS_POS3[2]=$AA_IM[Z]	
N700 G500	; Désactiver le frame réglable, car préréglage avec le frame nul (sans indication de valeur).
No_FRAME	; Désactiver le frame réglable, car préréglage avec le frame nul (sans indication de valeur)
M0	
M30	

Exemple de concaténation de frames

Concaténation de MEAFRAME à des fins de correction

La fonction `MEAFRAME( )` fournit un frame de correction. Si ce frame de correction est concaténé au frame réglable `$P_UIFR[1]` qui était actif à l'appel de la fonction, par exemple `G54`, on obtient un frame réglable pour d'autres calculs de déplacement ou d'usinage.

Concaténation avec ADDFRAME

Si ce frame de correction est actif à un autre endroit de la chaîne de frames ou si d'autres frames sont actifs avant le frame réglable, la fonction `ADDFRAME( )` peut être utilisée pour la concaténation à l'un des frames de base spécifiques à un canal ou à un frame système.

Ne doivent pas être actifs dans les frames :

- la fonction miroir avec `MIRROR`
- l'homothétie avec `SCALE`

Les paramètres d'entrée pour les valeurs réelles et les valeurs de consigne sont les coordonnées de la pièce. Dans le système de base de la commande, ces coordonnées doivent être indiqués en

- mm ou en pouces `(G71/G70)` et en tant que
- mesure portant `sur le rayon`

(DIAMOF).

5.7 Frames à définition globale pour NCU

Fonction

Les frames à définition globale pour NCU sont disponibles une seule fois par NCU, pour tous les canaux. Ils peuvent être écrasés et lus par tous les canaux. L'activation des frames à définition globale pour NCU a lieu dans le canal correspondant.

Les frames à définition globale permettent d'appliquer des décalages, des changements d'échelle et des fonctions miroir à des **axes de canal ou à des axes machine**.

Liens géométriques et concaténations de frames

Dans le cas de ces frames, il n'existe pas de lien géométrique entre les axes. C'est pourquoi les rotations et la programmation de descripteurs d'axes géométriques sont impossibles.

- Les frames à définition globale ne peuvent contenir aucune composante de rotation. La programmation d'une rotation est refusée avec émission de l'alarme : "18310 Canal %1 Bloc %2 Frame : rotation non autorisée".
- La concaténation de frames à définition globale et de frames spécifiques à un canal est possible. Le frame résultant contient toutes les composantes des frames, y compris les rotations pour tous les axes. L'affectation d'un frame comportant des composantes de rotation à un frame à définition globale est refusée avec émission de l'alarme "Frame : rotation non autorisée".

Frames à définition globale pour NCU

Frames de base à définition globale pour NCU $P_NCBFR[n]

jusqu'à 8 frames de base NCU globaux peuvent être configurés :

Des frames de base spécifiques à un canal peuvent être actifs simultanément.

Les frames à définition globale peuvent être écrasés et lus par tous les canaux d'une NCU. Lors de l'écrasement d'un frame à définition globale, l'utilisateur doit assurer la coordination entre les canaux. Cela est par exemple réalisable avec des marques wait `(WAITMC)`.

Constructeur de la machine-outil

Le nombre de frames de base globaux est configuré par des paramètres machine, voir /FB1/ Machine de fonctions de base ; axes, systèmes de coordonnées, frames (K2).

Frames réglables à définition globale pour NCU $P_UIFR[n]

Tous les frames réglables `G500`, `G54...G599` peuvent être configurés comme étant soit à définition globale pour NCU, soit spécifiques à un canal.

Constructeur de la machine-outil

Tous les frames paramétrables peuvent être reconfigurés en frames globaux à l'aide d'un paramètre machine $MN_MM_NUM_GLOBAL_USER_FRAMES.

En tant que descripteurs d'axes pour les instructions Frame, vous pouvez utiliser des descripteurs d'axes de canaux et des descripteurs d'axes machine. La programmation de descripteurs d'axes géométriques est refusée avec émission d'une alarme.

5.7.1 Frames spécifiques à un canal ($P_CHBFR, $P_UBFR)

Fonction

Les frames réglables ou les frames de base peuvent être écrasés et lus par l'opérateur (ex. HMI Advanced) et par l'AP

- par l'intermédiaire du programme pièce et
- par l'intermédiaire de l'OPi.

Le décalage fin est également possible pour les frames à définition globale. L'inhibition de frames à définition globale a lieu comme pour les frames spécifiques à un canal, avec `G53`, `G153`, `SUPA` et `G500`.

Constructeur de la machine-outil

Le PM 28081 MM_NUM_BASE_FRAMES permet de définir le nombre de frames de base dans un canal. En configuration standard, il existe au moins un frame de base par canal. 8 frames de base sont possibles au maximum par canal. En plus de ces 8 frames de base, 8 frames de base à définition globale pour NCU peuvent également exister dans le canal.

Frames spécifiques à un canal

$P_CHBFR[n]

La variable système `$P_CHBFR[n]` permet de lire et d'écraser les frames de base. Lors de l'écrasement d'un frame de base, le frame de base concaténé résultant n'est pas activé immédiatement, mais uniquement lors de l'exécution d'une instruction `G500`, `G54...G599`. Cette variable sert principalement à enregistrer les opérations d'écriture dans le frame de base effectuées par l'IHM ou l'AP. Ces variables frames sont sauvegardées.

Premier frame de base du canal

Un écrasement de la variable prédéfinie `$P_UBFR` n'active pas simultanément le frame de base d'indice 0 ; l'activation n'a lieu que lors de l'exécution d'une instruction `G500`, `G54...G599`. Cette variable peut également être lue et écrasée dans le programme.

$P_UBFR

`$P_UBFR` est identique à `$P_CHBFR[0]`. En version standard, il y a toujours un frame de base dans le canal, de sorte que la variable système est compatible avec les versions de logiciel précédentes. S'il n'y a pas de frame de base spécifique au canal, l'alarme "Frame : instruction non autorisée" est émise lors de l'écriture ou de la lecture.

5.7.2 Frames actifs dans un canal

Fonction

Les frames actifs dans un canal sont introduits par le programme pièce via les variables système respectives de ces frames. Les frames système en font également partie. Ces variables système permettent de lire et d'écraser le frame système courant dans le programme pièce.

Frames actifs courants dans un canal

Vue d'ensemble

Frames système courants	pour :
$P_PARTFRAME	TCARR et PAROT
$P_SETFRAME	Préréglage des mémoires de valeurs réelles et effleurement
$P_EXTFRAME	Décalage d'origine externe
$P_NCBFRAME[n]	Frames de base courants à définition globale pour NCU
$P_CHBFRAME[n]	Frames de base courants spécifiques à un canal
$P_BFRAME	1er frame de base courant du canal
$P_ACTBFRAME	Frame de base résultant
$P_CHBFRMASK et $P_NCBFRMASK	Frame de base résultant
$P_IFRAME	Frame réglable courant
Frames système courants	pour :
$P_TOOLFRAME	TOROT et TOFRAME
$P_WPFRAME	Points de référence de la pièce
$P_TRAFRAME	Transformations
$P_PFRAME	Frame programmable courant
Frame système courant	pour :
$P_CYCFRAME	Cycles
P_ACTFRAME	Frame résultant courant
Concaténation de frames	Le frame courant contient le frame de base résultant.

$P_NCBFRAME[n] Frames de base courants à définition globale pour NCU

La variable système `$P_NCBFRAME[n]` permet de lire et d'écraser les éléments de tableau des frames de base courants à définition globale. Le frame de base résultant est pris en considération dans le canal lors de l'opération d'écriture.

Le frame modifié n'est actif que dans le canal dans lequel le frame a été programmé. Si le frame doit être modifié pour tous les canaux d'une NCU, vous devez écraser simultanément $P_NCBFR[n] et $P_NCBFRAME[n]. Ensuite, le frame doit encore être activé dans les autres canaux à l'aide de G54, par ex. En cas d'écrasement d'un frame de base, le frame de base résultant est recalculé.

$P_CHBFRAME[n] Frames de base courants spécifiques à un canal

La variable système `$P_CHBFRAME[n]` permet de lire et d'écraser les éléments de tableau des frames de base courants spécifiques à un canal. Le frame de base résultant est pris en considération dans le canal lors de l'opération d'écriture. En cas d'écrasement d'un frame de base, le frame de base résultant est recalculé.

$P_BFRAME 1er frame de base courant du canal

La variable frame prédéfinie `$P_BFRAME` permet de lire et d'écraser, dans le programme pièce, le frame de base courant d'indice 0, qui est valide dans le canal. Ce frame est pris immédiatement en considération.

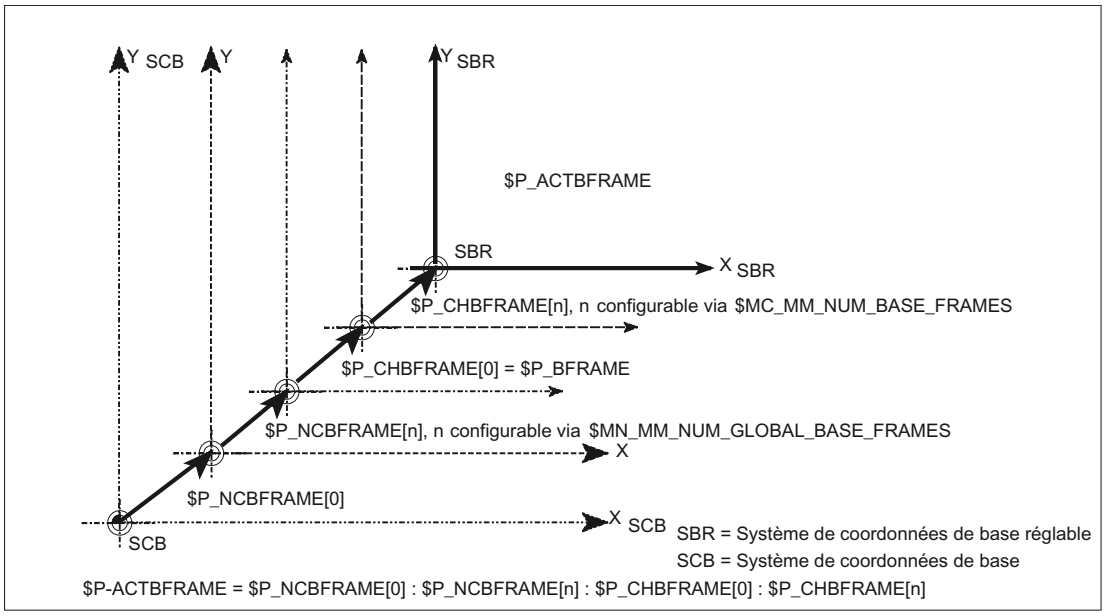
`$P_BFRAME` est identique à `$P_CHBFRAME[0]`. Par défaut, la variable système a toujours une valeur valide. S'il n'y a pas de frame de base spécifique au canal, l'alarme "Frame : instruction non autorisée" est émise lors de l'écriture ou de la lecture.

$P_ACTBFRAME Frame de base résultant

La variable `$P_ACTFRAME` détermine le frame de base résultant concaténé. Elle n'est accessible qu'en lecture.

`$P_ACTFRAME` correspond à

`$P_NCBFRAME[0]` : ... : `$P_NCBFRAME[n]` : `$P_CHBFRAME[0]` : ... : `$P_CHBFRAME[n]`.



$P_CHBFRMASK et $P_NCBFRMASK Frame de base résultant

Avec les variables système `$P_CHBFRMASK` et `$P_NCBFRMASK` , l'utilisateur peut sélectionner les frames de base pour le calcul du frame de base "résultant". Ces variables ne peuvent être programmées que dans le programme et lues via l'OPI. Les valeurs de ces variables sont interprétées en tant que masques binaires et indiquent les éléments de `$P_ACTFRAME` à prendre en considération.

`$P_CHBFRMASK` permet d'indiquer les frames de base spécifiques au canal à prendre en considération et `$P_NCBFRMASK` , les frames de base à définition globale pour NCU à prendre en considération.

La programmation de ces variables entraîne un nouveau calcul du frame de base résultant et du frame résultant. Les valeurs après Reset et les préréglages sont les suivantes :

`$P_CHBFRMASK = $MC_CHBFRAME_RESET_MASK` et

`$P_NCBFRMASK = $MC_CHBFRAME_RESET_MASK`.

par ex.

`$P_NCBFRMASK = 'H81'` ;$P_NCBFRAME[0] : $P_NCBFRAME[7]

`$P_CHBFRMASK = 'H11'` ;$P_CHBFRAME[0] : $P_CHBFRAME[4]

$P_IFRAME Frame réglable courant

La variable frame prédéfinie `$P_IFRAME` permet de lire et d'écraser, dans le programme pièce, le frame réglable courant qui est valide dans le canal. Le frame réglable écrasé est pris immédiatement en considération.

Dans le cas des frames réglables à définition globale pour NCU, le frame modifié n'est actif que dans le canal dans lequel le frame a été programmé. Si le frame doit être modifié pour tous les canaux d'une NCU, il faut écraser simultanément `$P_UIFR[n]` et `$P_IFRAME`. Ensuite, le frame correspondant doit encore être activé dans les autres canaux à l'aide de G54, par ex.

$P_PFRAME Frame programmable courant

`$P_PFRAME` est le frame programmable obtenu à partir de la programmation de `TRANS/ATRANS`, `G58/G59`, `ROT/AROT`, `SCALE/ASCALE`, `MIRROR/AMIRROR` ou de l'affectation de `CTRANS`, `CROT`, `CMIRROR`, `CSCALE` au FRAME programmable.

Variable frame programmable courante qui crée la relation entre le

- système de coordonnées réglable (SCR) et le
- Système de coordonnées pièce (SCP)

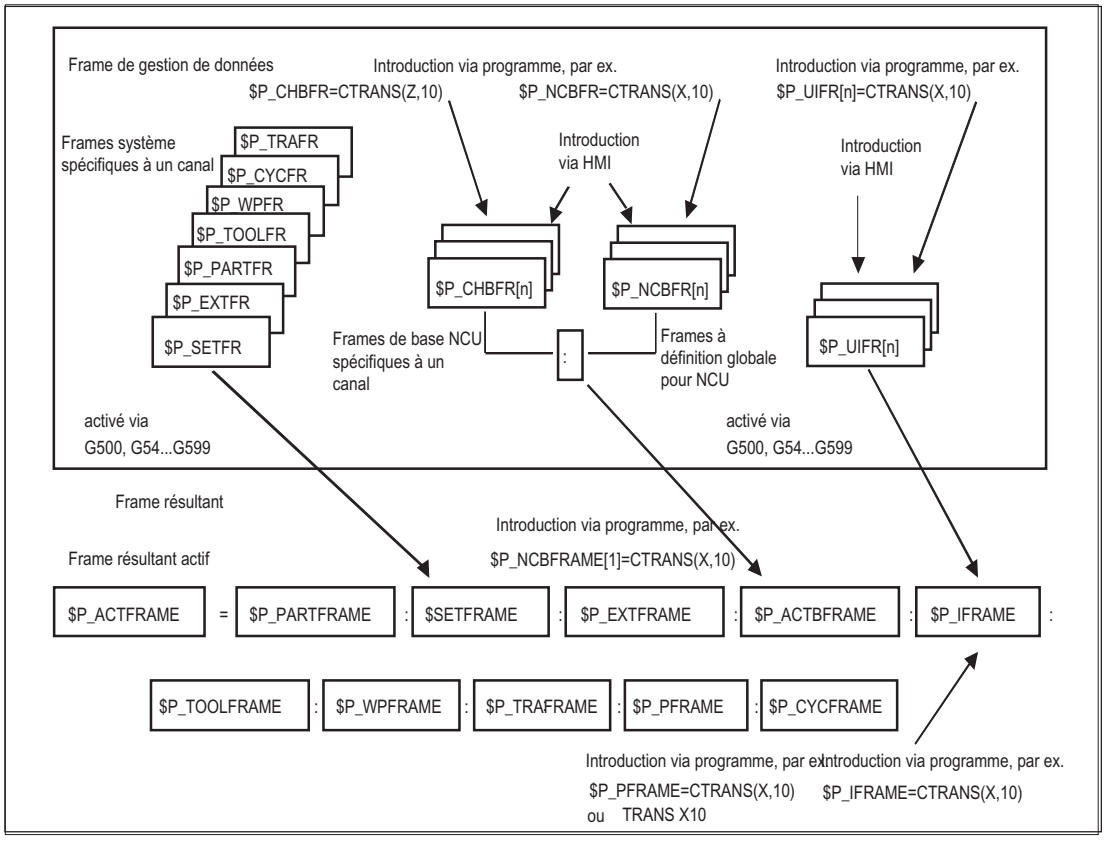
P_ACTFRAME Frame résultant courant

Le frame résultant courant $P_ACTFRAME résulte de la concaténation de tous les frames de base, du frame réglable courant et du frame programmable. Le frame courant est toujours actualisé si une composante de frame subit une modification.

`$P_ACTFRAME` correspond à

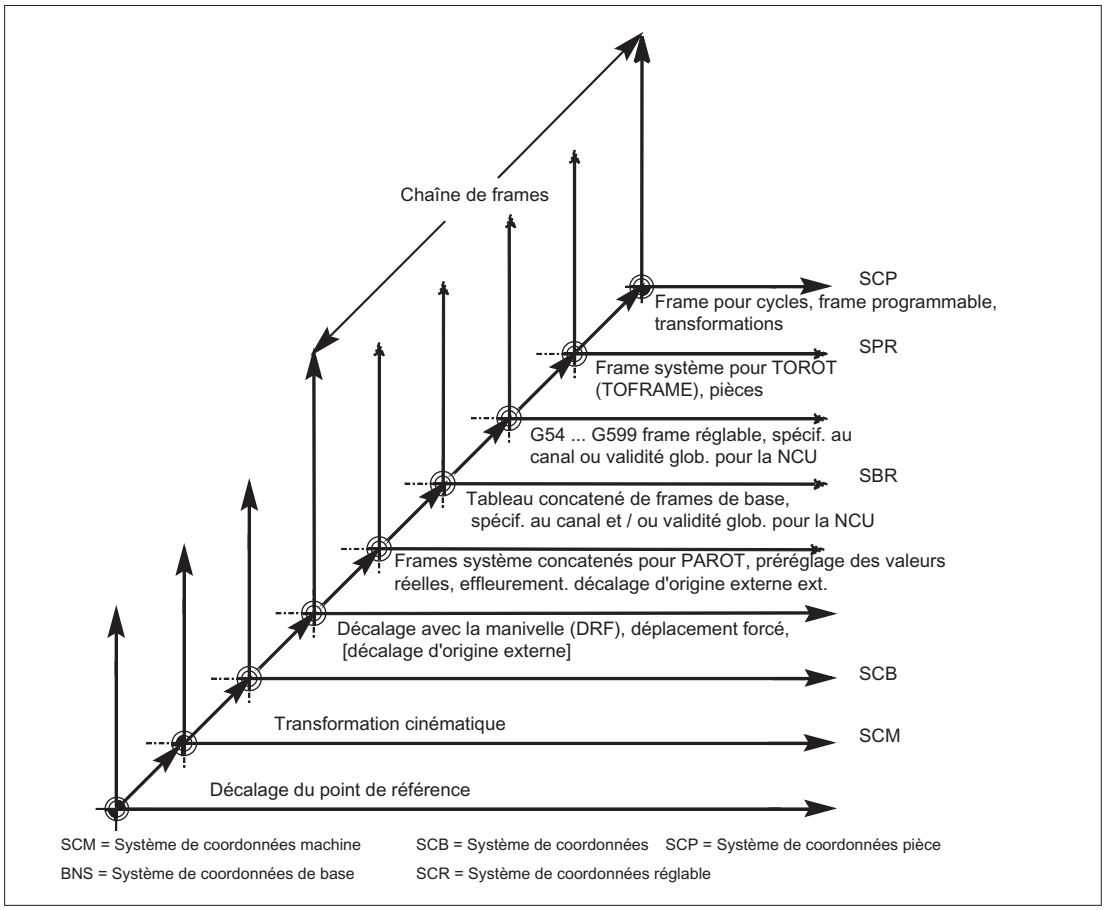
`$P_PARTFRAME : $P_SETFRAME : $P_EXTFRAME : $P_ACTBFRAME : $P_IFRAME :`

`$P_TOOLFRAME : $P_WPFRAME : $P_TRAFRAME : $P_PFRAME : $P_CYCFRAME`



Concaténation de frames

Le frame courant résulte de la concaténation du frame de base résultant, du frame réglable, du frame système et du frame programmable (voir frame résultant courant, ci-dessus).



Transformations 6

6.1 Programmation générale des types de transformation

Fonctionnement général

Pour adapter la commande à différentes cinématiques de machine, il est possible de programmer une sélection de types de transformation avec des paramètres appropriés. Par le biais de ces paramètres, il est possible, pour la transformation choisie, de convenir de l'orientation appropriée de l'outil dans l'espace, mais aussi des mouvements d'orientation des axes rotatifs.

Avec les transformations trois, quatre ou cinq axes, les indications de position programmées se réfèrent toujours à la pointe de l'outil qui est poursuivie perpendiculairement à la surface d'usinage dans l'espace. Les coordonnées cartésiennes sont converties du système de coordonnées de base au système de coordonnées machine et se rapportent aux axes géométriques. Ceux-ci décrivent le point de fonctionnement. Les axes rotatifs virtuels décrivent les orientations de l'outil dans l'espace et se programment avec TRAORI.

Lors de la transformation cinématique, il est possible de programmer des positions dans le système de coordonnées cartésiennes. La commande transforme les déplacements du système de coordonnées cartésiennes programmés avec TRANSMIT, TRACYL et TRAANG en déplacements des axes machine réels.

Programmation

Transformations trois, quatre et cinq axes TRAORI

La transformation d'orientation convenue est activée avec l'instruction TRAORI et les trois paramètres possibles pour numéro transfo, vecteur d'orientation et décalages d'axes rotatifs.

```
TRAORI(numéro transfo, vecteur d'orientation, décalages d'axes rotatifs)
```

Transformations cinématiques

Parmi les transformations cinématique, on compte les transformations convenues
`TRANSMIT(numéro transfo)`

`TRACYL(diamètre d'usinage, numéro transfo)`

`TRAANG(angle de l'axe en oblique, numéro transfo)`

Désactiver la transformation active

`TRAFOOF` permet de désactiver la transformation active à ce moment.

Transformation d'orientation

Transformations trois, quatre et cinq axes TRAORI

Pour usiner de manière optimale des surfaces à relief irrégulier dans la zone de travail de la machine, les machines-outils ont besoin d'axes supplémentaires en plus des trois axes linéaires X, Y et Z. Les axes supplémentaires décrivent l'orientation dans l'espace et sont nommés ci-après "axes d'orientation". On en dispose sous forme d'axes de rotation sur quatre types de machines dotées d'une cinématique différente.

1. Tête pivotante sur deux axes, par exemple tête d'outil à Cardan avec un axe rotatif parallèle à un axe linéaire, le plateau porte-outil étant fixe.
2. Plateau tournant sur deux axes, par exemple tête pivotante fixe avec plateau porte-outil pivotant sur deux axes.
3. Tête pivotante sur un axe et plateau tournant sur un axe, par exemple tête pivotante tournante avec outil orientable et plateau porte-outil pivotant sur un axe.
4. Tête pivotante sur deux axes et plateau tournant sur un axe, par exemple plateau porte-outil pivotant sur un axe et tête pivotante tournante avec outil tournant autour de son propre axe.

Les **transformations 3 et 4 axes** sont des formes particulières des transformations 5 axes et se programment de manière analogue aux transformations 5 axes.

La "**transformation générique 3, 4, 5, 6 axes**" couvre grâce à l'étendue de ses fonctions pour les axes rotatifs disposés à angle droit ainsi que les transformations pour la tête de fraisage à Cardan et peut être activée avec TRAORI comme toute autre transformation d'orientation pour ces quatre types de machine. En cas de transformation générique 5/6 axes, l'orientation de l'outil dispose d'un troisième degré de liberté pour lequel l'outil peut tourner sur son axe propre dans l'espace pour la direction de l'outil.

Bibliographie : /FB3/ Manuel de fonctions spéciales ; Transformation 3 à 5 axes (F2).

Position initiale d'orientation de l'outil indépendante de la cinématique

ORIRESET

Lorsque une transformation d'orientation est active avec TRAORI, alors ORIRESET peut indiquer les trois positions initiales de trois axes d'orientation maxi avec des paramètres A, B, C en option. L'affectation de l'ordre des paramètres programmés aux axes rotatifs s'effectue dans l'ordre des axes d'orientation défini par la transformation. Avec la programmation de ORIRESET(A, B, C), les axes d'orientation sont parcourus de façon linéaire et synchronisée de leur position momentanée à la position initiale indiquée.

Transformations cinématiques

TRANSMIT et TRACYL

Pour les fraisages à effectuer sur des tours, on peut, pour la transformation convenue, programmer soit

1. un usinage frontal avec TRANSMIT dans le dispositif de serrage du tour ou
2. un usinage de rainures quelconques sur des corps cylindriques avec TRACYL.

TRAANG

Si l'axe de pénétration doit aussi pouvoir pénétrer en oblique (pour une utilisation en rectification par exemple), on peut, avec TRAANG, programmer un angle paramétrable pour la transformation convenue.

Déplacement PTP cartésien

Ce que l'on appelle le "déplacement PTP cartésien", dans lequel on peut programmer jusqu'à 8 positions d'articulations STAT= différentes, fait également partie de la transformation cinématique. Les positions sont programmées dans le système de coordonnées cartésiennes, le déplacement de la machine s'effectuant en coordonnées machine.

Bibliographie :
/FB2/ Manuel de fonctions d'extension ; transformation cinématique (M1)

Transformations concaténées

Il est possible d'effectuer deux transformations successives. Pour la deuxième transformation ainsi concaténée, les trajets des axes sont repris à partir de la première transformation.

La première transformation peut être :

- la transformation de l'orientation TRAORI
- Transformation polaire TRANSMIT
- Transformation de cylindre TRACYL
- Transformation d'axe oblique TRAANG

La deuxième transformation doit être de type "axe oblique TRAANG"

6.1.1 Mouvements d'orientation lors des transformations

Déplacements d'outils et déplacements d'orientation

Les déplacements des orientations programmables dépendent en premier lieu du type de machine. pour une transformation trois, quatre ou cinq axes avec TRAORI, les axes de rotation ou les axes linéaires orientables décrivent les déplacements d'orientation de l'outil.

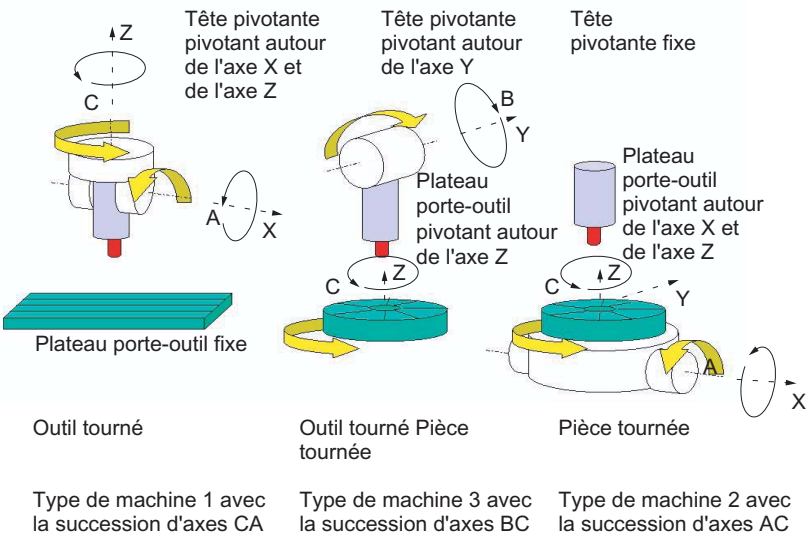
Toute modification de position des axes rotatifs participant à la transformation d'orientation entraîne des déplacements compensateurs des autres axes machine. Ce faisant, la position de la pointe de l'outil reste inchangée.

Les mouvements d'orientation de l'outil peuvent être programmés via les descripteurs d'axe rotatif A…, B…, C… des axes virtuels par l'indication d'angles d'Euler ou d'angles RPY, de vecteurs normaux à la direction ou à la surface, de vecteurs normés pour l'axe de rotation d'un cône ou pour l'orientation intermédiaire sur l'enveloppe d'un cône, selon l'application.

Lors de la transformation cinématique avec TRANSMIT, TRACYL et TRAANG, la commande transforme les déplacements programmés du système de coordonnées cartésiennes en déplacements des axes machine réels.

Cinématique de machine avec transformation trois, quatre et cinq axes TRAORI

Soit l'outil soit le plateau tournant est orientable avec deux axes rotatifs au plus. La combinaison d'une tête pivotante sur un axe et d'un plateau tournant lui aussi sur un axe est également possible.



Type de machine	Programmation de l'orientation
Transformation trois axes Types de machines 1 et 2	Programmation de l'orientation de l'outil uniquement dans le plan perpendiculaire à l'axe rotatif. Il existe **deux** axes de translation (axes linéaires) et **un** axe de rotation (axe rotatif).
Transformation quatre axes Types de machines 1 et 2	Programmation de l'orientation de l'outil uniquement dans le plan perpendiculaire à l'axe rotatif. Il existe **trois** axes de translation (axes linéaires) et **un** axe de rotation (axe rotatif).
Transformation cinq axes Types de machine 3 Tête pivotante sur un axe et plateau tournant sur un axe	Programmation de la transformation d'orientation. Cinématique avec **trois** axes linéaires et **deux** axes rotatifs orthogonaux. Les axes rotatifs sont parallèles à deux des trois axes linéaires. Le premier axe rotatif est déplacé par deux axes linéaires cartésiens. Il fait tourner le troisième axe linéaire avec l'outil. Le second axe rotatif fait pivoter la pièce.

Transformations 5/6 axes génériques

Type de machine	Programmation de la transformation d'orientation
Transformation cinq/six axes générique Types de machine 4 tête orientable à 2 axes avec outil indexable sur lui-même et plateau tournant à un axe	Programmation de la transformation d'orientation. Cinématique avec **trois** axes linéaires et**trois** axes rotatifs orthogonaux. Les axes rotatifs sont parallèles à deux des trois axes linéaires. Le premier axe rotatif est déplacé par deux axes linéaires cartésiens. Il fait tourner le troisième axe linéaire avec l'outil. Le second axe rotatif fait pivoter la pièce. L'orientation de base de l'outil peut être programmée par une rotation supplémentaire sur lui-même avec l'angle de rotation THETA.

A l'appel de la "transformation générique trois, quatre et cinq/six axes", l'orientation de base de l'outil peut de plus être transmise. Les restrictions concernant les directions des axes rotatifs ne s'appliquent plus. Lorsque les axes rotatifs ne sont pas exactement perpendiculaires entre eux ou que les axes rotatifs existants ne sont pas parfaitement parallèles aux axes linéaires, la "transformation générique cinq/six axes" peut fournir de meilleurs résultats pour l'orientation de l'outil.

Transformations cinématiques TRANSMIT, TRACYL et TRAANG

Pour les fraisages à effectuer sur des tours ou sur un axe à pénétration oblique en rectification, ce sont les dispositions d'axes suivantes qui sont valables en standard, en fonction de la transformation:

TRANSMIT	Activation de la transformation polaire
usinage frontal sans démontage de la pièce	un axe rotatif un axe de pénétration dans le plan perpendiculaire à l'axe rotatif un axe longitudinal parallèle à l'axe rotatif

TRACYL	Activation de la transformation de surface latérale de cylindre
Usinage de rainures de forme quelconque sur des corps cylindriques	un axe rotatif un axe de pénétration dans le plan perpendiculaire à l'axe rotatif un axe longitudinal parallèle à l'axe rotatif

TRAANG	Activation de la transformation axe oblique
Usinage avec axe de pénétration oblique	un axe rotatif un axe de pénétration avec angle paramétrable un axe longitudinal parallèle à l'axe rotatif

Déplacement PTP cartésien

Le déplacement de la machine s'effectue en coordonnées machine et se programme avec:

TRAORI	Activation de la transformation
Déplacement point à point PTP	Déplacement à une position dans le système de coordonnées cartésiennes (SCM)
CP	Déplacement avec interpolation des axes cartésiens dans (SCB)
STAT	La position des articulations dépend de la transformation
TU	Selon quel angle faut-il déplacer les axes au plus court?

Déplacement PTP avec une transformation générique cinq/six axes

Le déplacement de la machine s'effectue dans les coordonnées machine et l'orientation de l'outil peut aussi bien être programmé avec des positions d'axes rotatifs qu'avec des vecteurs Euler ou des angles RPY indépendants de la cinématique ou encore des vecteurs de direction.

Une interpolation d'axe rotatif, une interpolation vectorielle avec une interpolation circulaire de grand rayon ou une interpolation du vecteur d'orientation sur une enveloppe de cône sont possibles.

Exemple Transformation trois à cinq axes pour une tête de fraisage à Cardan

La machine-outil comporte au moins 5 axes, dont

- Trois par translation pour des mouvements rectilignes déplaçant le point de fonctionnement à n'importe quelle position dans l'espace de travail.
- Deux axes de pivotement rotatifs disposés selon un angle configurable (généralement 45 degrés) permettant à l'outil d'adopter des orientations dans l'espace qui se limitent à une demi-sphère pour une disposition à 45 degrés.

6.1.2 Aperçu de la transformation d'orientation TRAORI

Types de programmation possibles en relation avec TRAORI

Type de machine	Programmation en cas de transformation TRAORI active
Types de machine 1, 2 ou 3 tête orientable à deux axes ou plateau tournant à deux axes ou une combinaison d'une tête orientable et d'un plateau tournant sur axe chacun.	La succession d'axes des axes d'orientation et la direction d'orientation de l'outil est configurable soit **par rapport à la machine** par le biais des paramètres machine en fonction de la cinématique de machine soit **par rapport à la pièce** avec orientation programmable indépendamment de la cinématique de machine. Les sens de rotation des axes d'orientation dans le système de référence se programment avec: - Système de référence ORIMKS = Système de coordonnées machine - Système de référence ORIWKS = Système de coordonnées pièce La configuration de base est ORIWKS Programmation des axes d'orientation avec: A, B, C positions d'axe machine directes A2, B2, C2 programmation d'angles d'axes virtuels avec - ORIEULER par angle d'Euler (par défaut) - ORIRPY par angle RPY - ORIVIRT1 par axes virtuels d'orientation 1e définition - ORIVIRT2 par axes virtuels d'orientation 2e définition Avec distinction du type d'interpolation : **Interpolation linéaire** - ORIAXES d'axes d'orientation ou d'axes machine **Interpolation circulaire de grand rayon** (interpolation du vecteur d'orientation) - ORIVECT d'axes d'orientation Programmation des axes d'orientation par indication A3, B3, C3 des composantes de vecteur (normale à la direction/surface) Programmation de l'orientation résultante de l'outil A4, B4, C4 du vecteur de la normale à la surface au début du bloc A5, B5, C5 du vecteur de la normale à la surface à la fin du bloc LEAD angle d'avance pour l'orientation de l'outil TILT angle latéral pour l'orientation de l'outil

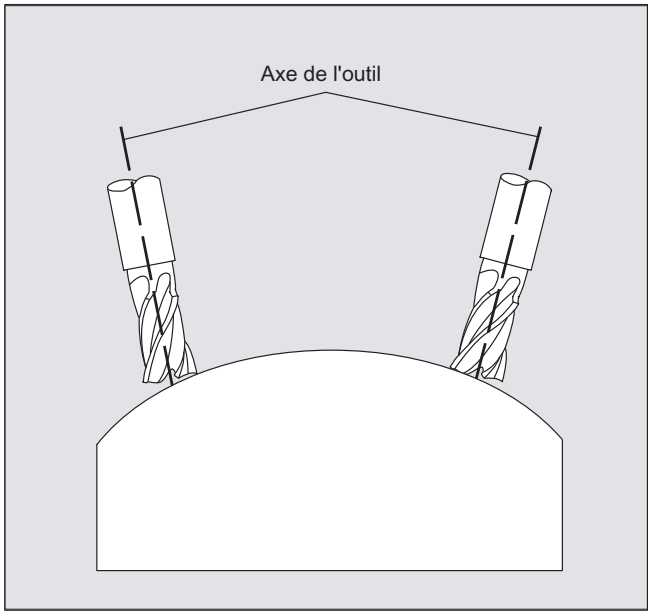
Type de machine	Programmation en cas de transformation TRAORI active
	Interpolation du vecteur d'orientation sur une enveloppe de cône Changements d'orientation sur une enveloppe de cône placée **librement dans l'espace** Enveloppe de cône par interpolation : - ORIPLANE dans le plan (interpolation circulaire de grand rayon) - ORICONCW sur une enveloppe de cône dans le sens horaire - ORICONCCW sur une enveloppe de cône dans le sens antihoraire A6, B6, C6 vecteur de direction (axe de rotation du cône) -OICONIO interpolation sur une enveloppe de cône avec : A7, B7, C7 Vecteurs intermédiaires (orientation de départ et orientation finale) ou - ORICONTO sur une enveloppe de cône transition tangentielle Changements d'orientation par rapport à **une trajectoire** avec - ORICURVE Indication du déplacement de deux points de contact par le biais de PO[XH]=(xe, x2, x3, x4, x5) Polynômes d'orientation jusqu'au 5e degré PO[YH]=(ye, y2, y3, y4, y5) Polynômes d'orientation jusqu'au 5e degré PO[ZH]=(ze, z2, z3, z4, z5) Polynômes d'orientation jusqu'au 5e degré - ORIPATHS lissage du tracé d'orientation avec A8, B8, C8 La phase de réorientation de l'outil correspond: à la direction et à la longueur de parcours de l'outil lors du mouvement de retrait
Types de machine 1 et 3 D'autres types de machine avec rotation supplémentaire de l'outil sur lui-même exigent un troisième axe rotatif Transformation d'orientation, par exemple une transformation générique six axes. Rotations du vecteur d'orientation.	Programmation **des rotations** de l'orientation de l'outil avec LEAD Angle d'avance Angles par rapport au vecteur normal à la surface PO[PHI]=(.. , ..)] Programmation d'un polynôme jusqu'au 5e degré TILT Angle latéral Rotation par rapport à la tangente à la trajectoire (direction Z) PO[PSI] Programmation d'un polynôme jusqu'au 5e degré THETA Angle de rotation (rotation par rapport à la direction en Z de l'outil) THETA= Valeur atteinte en fin de bloc THETA=AC(...) Passage bloc par bloc à l'introduction des cotes en valeurs absolues THETA=IC(...) Passage bloc par bloc à l'introduction des cotes relatives THETA=Θ_e Interpoler l'angle programmé G90/G91 PO[THT]=(..) Programmation d'un polynôme jusqu'au 5e degré Programmation du vecteur de rotation - ORIROTA Rotation absolue - ORIROTR Vecteur de rotation relatif - ORIROTT Vecteur de rotation tangentiel
Orientation relative à la trajectoire pour des modifications d'orientation par rapport à la trajectoire ou rotation du vecteur de rotation tangentiellement à la trajectoire	Changements d'orientation **par rapport à la trajectoire** avec - ORIPATH Orientation de l'outil par rapport à la trajectoire - ORIPATHS additionnellement en cas de coude dans le tracé d'orientation Programmation du vecteur de rotation - ORIROTC Vecteur de rotation tangentiel, Rotation par rapport à la tangente à la trajectoire

6.2 Transformation trois, quatre et cinq axes (TRAORI)

6.2.1 Corrélations générales de tête de fraisage de type cardan

Fonction

Pour réunir des conditions de coupe optimales lors de l'usinage de surfaces courbes, il est nécessaire de pouvoir modifier l'angle d'inclinaison de l'outil.

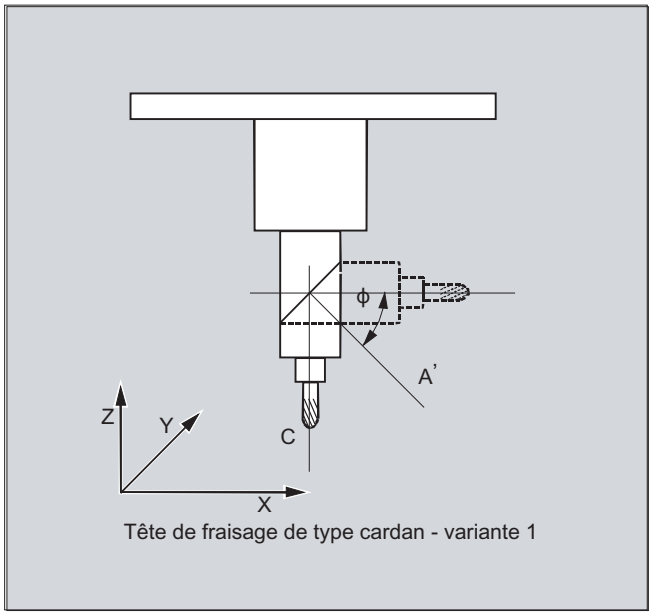


La configuration correspondante de la machine est rangée dans les paramètres machine relatifs aux axes.

Transformation 5 axes

Tête de fraisage de type cardan

Trois axes linéaires (X, Y, Z) et deux axes d'orientation (C, A) déterminent l'angle d'inclinaison et le point d'attaque de l'outil. L'un des deux axes d'orientation - dans l'exemple A' - est incliné, souvent à 45°.

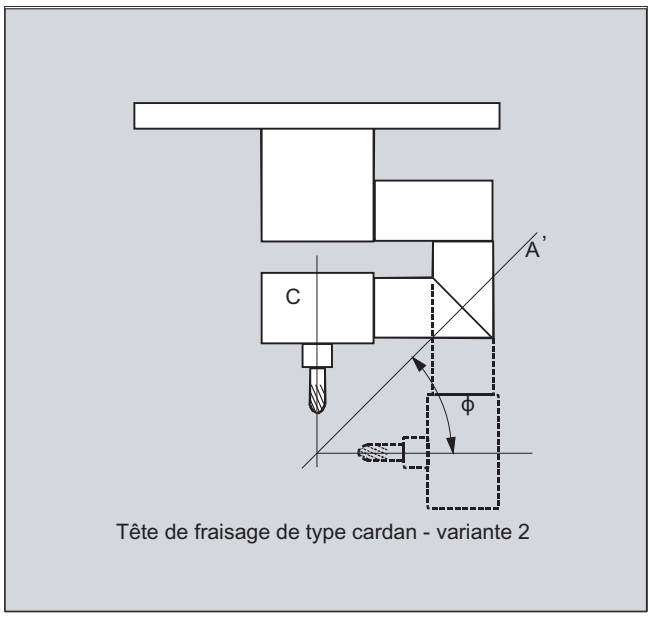


Tête de fraisage de type cardan - variante 1

Dans les exemples montrés ici, vous voyez les dispositions illustrées avec la tête d'outil à Cardan de la cinématique de machine CA!

Constructeur de la machine-outil

La succession d'axes des axes d'orientation et la direction d'orientation de l'outil peuvent se régler en fonction de la cinématique de machine par le biais de paramètres machine.



Tête de fraisage de type cardan - variante 2

Sur cet exemple, A' se trouve sous l'angle φ par rapport à l'axe X

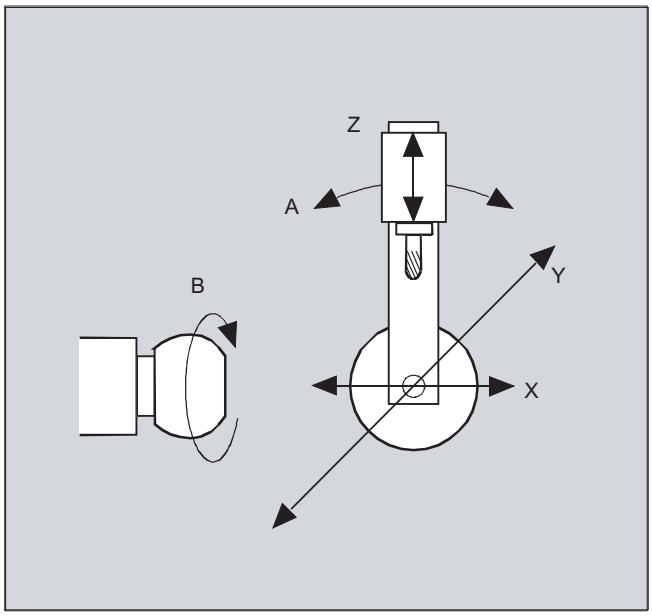
D'une manière générale, les corrélations possibles sont les suivantes:

A' se trouve sous l'angle φ par rapport à	l'axe X
B' se trouve sous l'angle φ par rapport à	l'axe Y
C' se trouve sous l'angle φ par rapport à	l'axe Z

L'angle φ peut être configuré dans la plage 0° à +89° par le biais des paramètres machine.

Avec axe linéaire pivotant

Il s'agit d'une configuration avec une pièce mobile et un outil mobile. La cinématique se compose de trois axes linéaires { (X, Y, Z) et de deux axes rotatifs orthogonaux. Le premier axe rotatif se déplace sur un chariot à deux axes linéaires croisés. L'outil est parallèle au troisième axe linéaire. Le second axe rotatif fait pivoter la pièce. Le troisième axe linéaire (axe de pivotement) se situe dans le plan du chariot croisé.



La succession d'axes des axes rotatifs et la direction d'orientation de l'outil sont peuvent se régler en fonction de la cinématique de machine par le biais de paramètres machine.

Les corrélations possibles sont les suivantes :

Axes :	**Configuration axiale :**
1. Axe rotatif	A A B B C C
2. Axe rotatif	B C A C A B
Axe linéaire pivoté	Z Y Z X Y X

Pour plus d'explications en ce qui concerne les successions d'axes pour la direction d'orientation de l'outil, se reporter à

Bibliographie : /FB3/ Manuel de fonctions spéciales ; Transformation 3 à 5 axes (F2), chapitre tête de fraisage de type cardan, "Paramétrage".

6.2.2 Transformation trois, quatre et cinq axes (TRAORI)

Fonction

L'utilisateur peut configurer deux ou trois axes linéaires et un axe rotatif. On suppose que l'axe rotatif est perpendiculaire au plan d'orientation.

L'orientation de l'outil est possible uniquement dans le plan perpendiculaire à l'axe rotatif. Les transformations sont plus particulièrement destinées aux machines avec outil et pièce mobiles.

La configuration et la programmation des transformations 3 axes et 4 axes sont analogues aux transformations 5 axes.

Bibliographie :
Description fonctionnelle Fonctions spéciales ; Transformations multi-axes (F2)

Syntaxe

```
TRAORI(<n>)
TRAORI(<n>,<X>,<Y>,<Z>,<A>,<B>)
TRAFOOF
```

Signification

`TRAORI` :	Activation de la première transformation d'orientation convenue
`TRAORI(<n>)` :	Activation de la n-ième transformation d'orientation convenue
`<n>` :	numéro de la transformation
	Valeur : 1 ou 2
	Exemple :
	TRAORI(1) active la transformation d'orientation 1
`<X>,<Y>,<Z>` :	Composante du vecteur d'orientation dans laquelle est dirigé l'outil.
`<A>,<B>` :	Décalage programmable pour les axes rotatifs
`TRAFOOF` :	Désactiver la transformation

Orientation de l'outil

En fonction de l'orientation de l'outil sélectionnée, il convient de régler le plan de travail actif (G17, G18, G19) défini dans le programme CN, de façon à ce que la correction de longueur d'outil agisse dans la direction de l'orientation de l'outil.

Remarque

Dès que la transformation est activée, les indications de position (X, Y, Z) se réfèrent toujours à la pointe de l'outil. Toute modification des positions des axes rotatifs impliqués dans la transformation entraîne des déplacements compensatoires des autres axes machine, ce qui permet que la position de la pointe de l'outil reste inchangée.

La transformation d'orientation est toujours dirigée de la pointe de l'outil vers le porte-outil.

Décalage pour les axes d'orientation

L'activation de la transformation d'orientation permet de programmer directement un décalage supplémentaire pour les axes d'orientation.

Les paramètres peuvent être omis si l'ordre correct est respecté lors de la programmation.

Exemple :

```
TRAORI(, , , ,A,B)
```
; lorsqu'un seul offset doit être renseigné

Comme alternative à la programmation directe, l'offset supplémentaire pour les axes d'orientation peut aussi être repris automatiquement à partir du décalage d'origine présentement actif. La prise en charge est définie par le biais des paramètres machine.

Exemples

```
TRAORI(1,0,0,1)          ; L'orientation de base de l'outil pointe dans la direction
                           Z

TRAORI(1,0,1,0)          ; L'orientation de base de l'outil pointe dans la direction
                           Z

TRAORI(1,0,1,1)          ; L'orientation de base de l'outil pointe dans le direction
                           Y/Z (correspond à la position -45°)
```

6.2.3 Variantes de programmation d'orientation et position initiale (ORIRESET)

Programmation d'orientation de l'outil avec TRAORI

En corrélation avec une transformation d'orientation programmable TRAORI, des positions d'axe ou des axes virtuels avec angles ou composants vectoriels peuvent être également programmés par des descripteurs d'axes rotatifs A.., B..., C... en plus des axes linéaires X, Y, Z. Pour les axes d'orientation et les axes machine, différents types d'interpolations sont possibles. Indépendamment des polynômes d'orientation PO[angle] et des polynômes d'axe PO[axe] activés actuellement, on peut programmer plusieurs types de polynômes différents tels que G1, G2, G3, CIP ou POLY.

La modification d'orientation de l'outil peut se programmer également par le biais de vecteurs d'orientation. Ce faisant, l'orientation finale de chaque bloc peut s'effectuer soit par programmation directe du vecteur, soir par programmation des positions des axes rotatifs.

Remarque

Variantes de programmation d'orientation pour transformations trois à cinq axes

Dans le cas de la transformation trois à cinq axes, les variantes

1. A, B, C indication directe des positions des axes machine
2. A2, B2, C2 programmation des angles des axes virtuels par le biais d'angles d'Euler ou d'angles RPY
3. A3 ,B3, C3 indication des composantes vectorielles
4. LEAD, TILT indication des angles d'avance et des angles latéraux par rapport à la trajectoire et à la surface
5. A4, B4, C4 et A5, B5, C5 vecteur normal à la surface en début et en fin de bloc
6. A6, B6, C6 et A7, B7, C7 interpolation du vecteur d'orientation sur une enveloppe de cône
7. A8, B8, C8 réorientation de l'outil, direction et longueur de parcours du mouvement de retrait

s'excluent mutuellement.

Des messages d'alarme empêchent une programmation mixte de valeurs.

Position initiale de l'orientation d'outil ORIRESET

Avec la programmation de ORIRESET(A, B, C), les axes d'orientation sont parcourus de façon linéaire et synchronisée de leur position momentanée à la position initiale indiquée.

Si, pour un axe, aucune position initiale n'est programmée, alors une position définie du paramètre machine correspondant $MC_TRAFO5_ROT_AX_OFFSET_1/2 est employée. Des frames des axes rotatifs éventuellement actifs ne sont pas pris en compte.

Remarque

Uniquement lorsque une transformation d'orientation est active avec TRAORI(...), une position initiale de l'orientation de l'outil peut être programmée indépendamment de la cinématique avec ORIRESET(...) sans alarme 14101.

Exemples

```
1. Exemple de cinématique de machine CA (nom d'axe de canal C, A)
ORIRESET(90, 45)       ;C sur 90 degrés, A sur 45 degrés
ORIRESET(, 30)         ;C sur $MC_TRAFO5_ROT_AX_OFFSET_1/2[0], A sur 30 degrés
ORIRESET( )            ;C sur $MC_TRAFO5_ROT_AX_OFFSET_1/2[0],
                       ;A sur $MC_TRAFO5_ROT_AX_OFFSET_1/2[1],
2. Exemple de cinématique de machine CAC (nom d'axe de canal C, A, B)
ORIRESET(90, 45, 90)   ;C sur 90 degrés, A sur 45 degrés, B sur 90 degrés
ORIRESET( )            ;C sur $MC_TRAFO5_ROT_AX_OFFSET_1/2[0],
                       ;A sur $MC_TRAFO5_ROT_AX_OFFSET_1/2[1],
                       ;B sur $MC_TRAFO5_ROT_AX_OFFSET_1/2[2],
```

Programmation des rotations LEAD, TILT und THETA

Dans le cas de la transformation trois à cinq axes, les rotations de l'orientation de l'outil se programment avec l'angle d'avance LEAD et l'angle latéral TILT.

Pour une transformation avec un troisième axe rotatif, des programmations supplémentaires de C2 (rotations du vecteur d'orientation) sont autorisées aussi bien pour l'orientation avec des composantes de vecteur qu'avec l'indication de l'angle LEAD, TILT.

Avec un troisième axe rotatif supplémentaire, la rotation de l'outil sur lui-même peut être programmée avec l'angle de rotation THETA.

6.2.4 Programmation de l'orientation de l'outil (A..., B..., C..., LEAD, TILT)

Fonction

On dispose des possibilités suivantes pour programmer l'orientation de l'outil:

1. Programmation directe du déplacement des axes rotatifs. Le changement d'orientation s'effectue toujours dans le système de coordonnées de base ou dans le système de coordonnées machine. Les axes d'orientation sont déplacés sous forme d'axes synchrones.
2. Programmation dans les angles Euler ou RPY selon la définition d'angle via `A2, B2, C2`.
3. Programmation du vecteur de direction par le biais de `A3, B3, C3`. Le vecteur de direction est dirigé de la pointe de l'outil vers le porte-outil.
4. Programmation du vecteur normal à la surface en début de bloc avec `A4, B4, C4` et en fin de bloc avec `A5, B5, C5` (fraisage avant).
5. Programmation par le biais de l'angle d'avance `LEAD` et de l'angle latéral `TILT`
6. Programmation de l'axe rotatif du cône comme vecteur normé par le biais de `A6, B6, C6` ou de l'orientation intermédiaire sur l'enveloppe du cône par le biais de `A7, B7, C7`, voir chapitre "Programmation de l'orientation le long d'une enveloppe de cône (ORIPLANE, ORICONxx)".
7. Programmation de réorientation, de direction et longueur de parcours de l'outil pendant le mouvement de retrait via `A8, B8, C8`,
voir chapitre "Lissage du tracé d'orientation (ORIPATHS A8=, B8=, C8=)"

Remarque

Dans tous les cas, la programmation de l'orientation est autorisée uniquement si une transformation d'orientation est activée.

Avantage : ces programmes sont transposables sur n'importe quelle cinématique de machine.

Définition de l'orientation de l'outil via un code G

Remarque

Constructeur de la machine-outil

Le paramètre machine permet de commuter entre angle Euler ou RPY. Avec des paramètres machine adaptés, une commutation aussi bien dépendante qu'indépendante du code G actif du groupe 50 est possible. Les paramétrages suivants sont possibles :

1. Lorsque les deux paramètres machine pour la définition des axes d'orientation et la définition de l'angle d'orientation sont remis à zéro par le code G :
   Les angles programmés avec `A2, B2, C2` sont interprétés **en fonction du paramètre machine** Définition de l'angle de la programmation d'orientation soit comme des angles Euler soit comme des angles RPY.
2. Lorsque le paramètre machine pour la définition des axes d'orientation est mis à un par le code G, la commutation a lieu
   en fonction du code G actif du groupe 50 :
   Les angles programmés avec `A2, B2, C2` sont interprétés selon l'un des codes G actifs `ORIEULER, ORIRPY, ORIVIRT1, ORIVIRT2, ORIAXPOS` et `ORIPY2`. Les valeurs programmées avec les axes d'orientation sont également interprétées comme angles d'orientation selon le code G actif du groupe 50.
3. Lorsque le paramètre machine pour la définition des angles d'orientation au moyen du code G est mis à un et le paramètre machine pour la définition des axes d'orientation au moyen du code G est mis à zéro, la commutation s'effectue
   indépendamment du code G actif dans le groupe 50 :
   les angles programmés avec `A2, B2, C2` sont interprétés en fonction de l'un des codes G `ORIEULER, ORIRPY, ORIVIRT1, ORIVIRT2 ORIAXPOS` et `ORIPY2` actifs. Les valeurs programmées avec les axes d'orientation sont toujours interprétées comme positions d'axe rotatif, indépendamment code G actif du groupe 50.

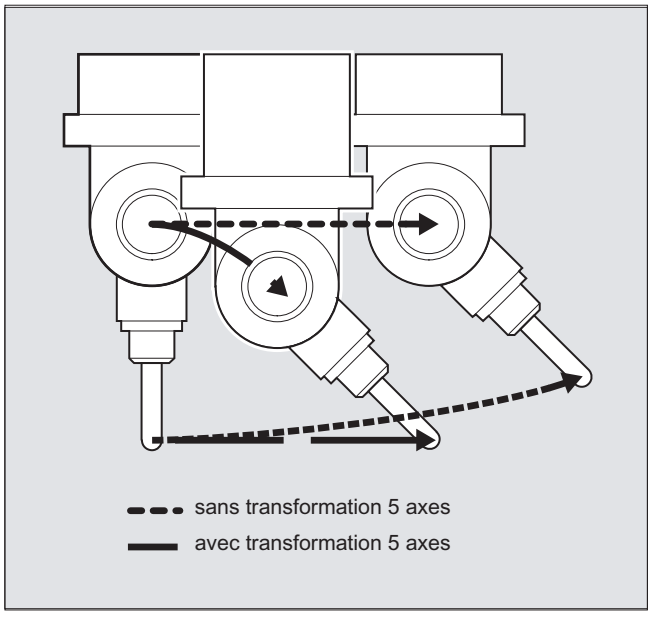
Programmation

`G1 X Y Z A B C`	Programmation du déplacement des axes rotatifs
`G1 X Y Z A2= B2= C2=`	Programmation en angles d'Euler
`G1 X Y Z A3== B3== C3==`	Programmation du vecteur de direction
`G1 X Y Z A4== B4== C4==`	Programmation du vecteur normal à la surface en début de bloc
`G1 X Y Z A5== B5== C5==`	Programmation du vecteur normal à la surface en fin de bloc
`LEAD=`	Angle d'avance pour la programmation de l'orientation de l'outil
`TILT=`	Angle latéral pour la programmation de l'orientation de l'outil

Paramètres

G....	Indication du type de déplacement des axes rotatifs
X Y Z	Indication des axes linéaires
A B C	Indication des positions des axes machine des axes rotatifs
A2 B2 C2	Programmation des angles (Euler ou RPY) des axes virtuels ou axes d'orientation
A3 B3 C3	Indication des composantes du vecteur de direction
A4 B4 C4	Indication, par ex. dans le cas du fraisage de bout en bout, des composantes du vecteur normal à la surface en début de bloc
A5 B5 C5	Indication, par ex. dans le cas du fraisage de bout en bout, des composantes du vecteur normal à la surface en fin de bloc
LEAD	Angle par rapport au vecteur normal à la surface, dans le plan défini par la tangente à la trajectoire et le vecteur normal à la surface
TILT	Angle par rapport au vecteur normal à la surface, dans le plan perpendiculaire à la tangente à la trajectoire

Exemple Comparaison: avec et sans transformation 5 axes



Description

En règle générale, les programmes 5 axes sont générés par des systèmes de CAO/FAO et ne sont pas entrés au pied de la machine. C'est pourquoi les explications suivantes s'adressent principalement aux programmeurs de postprocesseurs.

Le type de programmation d'orientation est déterminée dans le code G du groupe 50 :

`ORIEULER` par angle d'Euler
`ORIRPY` par angle RPY (ordre de rotation ZYX)
`ORIVIRT1` par les axes d'orientation virtuels (définition 1)
`ORIVIRT2` par les axes d'orientation virtuels (définition 2)
`ORIAXPOS` par les axes d'orientation virtuels avec positions d'axe rotatif
`ORIPY2` par angle RPY (ordre de rotation XYZ)

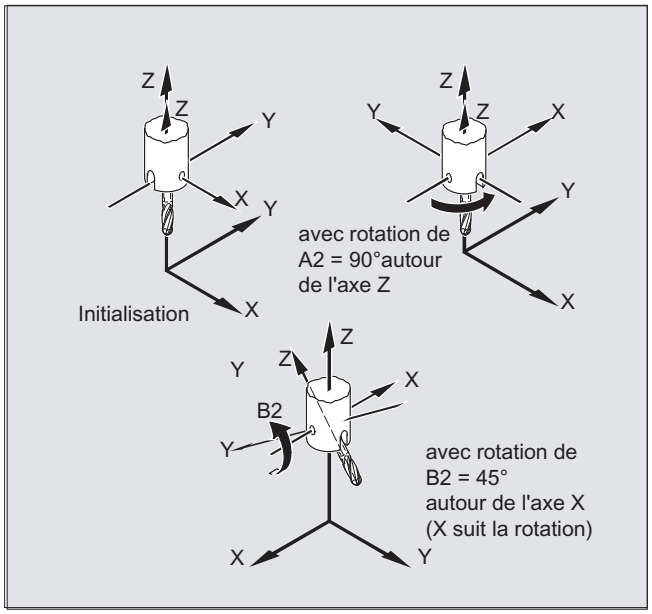
Constructeur de la machine-outil

Le constructeur de la machine-outil peut définir différentes variantes via des paramètres machine. Tenez compte des indications du constructeur de la machine.

Programmation en angles d'Euler ORIEULER

Les valeurs programmées avec `A2, B2, C2` lors de la programmation de l'orientation sont interprétées comme angles d'Euler (en degrés).

Le vecteur d'orientation est obtenu en faisant pivoter { un vecteur en direction Z tout d'abord avec `A2` autour de l'axe Z, puis avec `B2` autour du nouvel axe X et enfin avec `C2` autour du nouvel axe Z.



Dans ce cas, la valeur de `C2` (rotation autour du nouvel axe Z) est sans signification et n'est pas à programmer.

Programmation en angles RPY ORIRPY

Les valeurs programmées avec `A2, B2, C2` lors de la programmation de l'orientation sont interprétées comme angles RPY (en degrés).

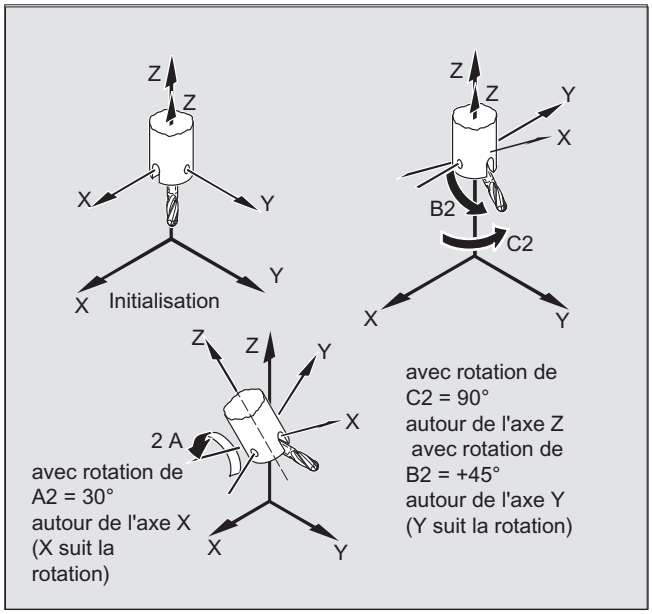
Remarque

Contrairement à la programmation en angles d'Euler { , les trois valeurs agissent ici sur le vecteur d'orientation

Constructeur de la machine-outil

En cas de définition d'angle d'orientation par l'angle RPY, on a, pour les axes d'orientation, $MC_ORI_DEF_WITH_G_CODE = 0

Le vecteur d'orientation est obtenu en faisant pivoter un vecteur en direction Z d'abord avec C2 autour de l'axe Z, puis avec B2 autour du nouvel axe Y et enfin avec A2 autour du nouvel axe X.

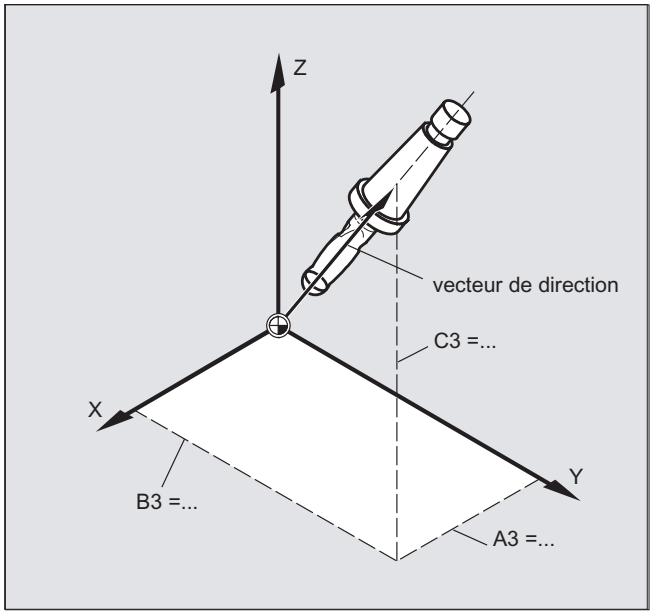


Lorsque le paramètre machine pour la définition des axes d'orientation via le code G $MC_ORI_DEF_WITH_G_CODE = 1, alors :
Le vecteur d'orientation résulte de la rotation d'un vecteur dans la direction Z autour de l'axe Z avec A2, puis autour de l'axe Y avec B2 et enfin autour du nouvel axe X avec C2.

Programmation du vecteur de direction

Les composantes du vecteur de direction sont programmées avec `A3, B3, C3` . Le vecteur est dirigé vers le porte-outil ; la longueur du vecteur est sans importance.

La commande attribue la valeur zéro aux composantes non programmées du vecteur.



Programmation de l'orientation de l'outil avec LEAD= et TILT=

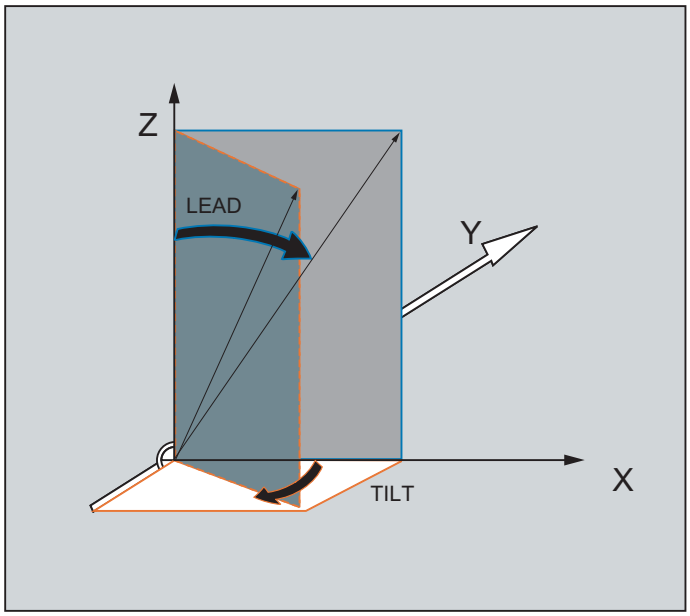
L'orientation résultante de l'outil est déterminée à partir :

- de la tangente à la trajectoire
- du vecteur normal à la surface
  en début de bloc `A4, B4, C4` et en fin de bloc `A5, B6, C5`
- de l'angle d'avance `LEAD`
  dans le plan défini par la tangente à la trajectoire et le vecteur normal à la surface
- de l'angle latéral `TILT` en fin de bloc
  perpendiculaire à la tangente à la trajectoire et par rapport au vecteur normal à la surface

Comportement aux angles rentrants (pour CO 3D)

L'orientation résultante est toujours atteinte en fin de bloc, même si le bloc est tronqué au niveau d'un angle rentrant.

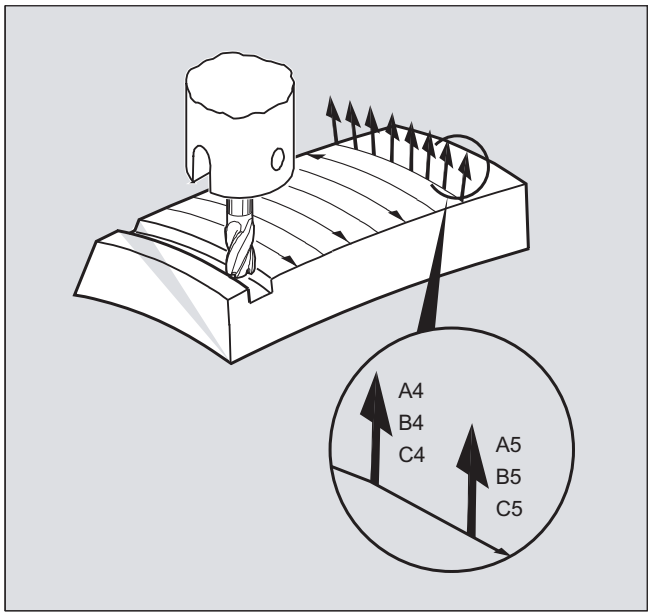
Définition de l'orientation de l'outil avec LEAD= et TILT=



6.2.5 Fraisage en bout (fraisage 3D A4, B4, C4, A5, B5, C5)

Fonction

Le fraisage en bout sert à l'usinage des surfaces à courbure quelconque.



Pour ce type de fraisage 3D, vous avez besoin de la description des différentes trajectoires 3D qui balayent la surface de la pièce, ligne après ligne.

Les calculs sont généralement réalisés à l'aide d'un système de FAO, en tenant compte de la forme et des dimensions de l'outil. Les blocs CN générés sont ensuite lus dans la commande par l'intermédiaire de postprocesseurs.

Programmation de la courbure de la trajectoire

Description des surfaces

La description de la courbure de la trajectoire s'effectue par le biais de vecteurs normaux à la surface ayant les composantes suivantes :

`A4, B4, C4` : vecteur de début en début de bloc

`A5, B5, C5` : vecteur de fin en fin de bloc

Si un bloc ne contient que le vecteur de début, le vecteur normal à la surface restera constant pendant tout le bloc. Si un bloc ne contient que le vecteur de fin, une interpolation sur le grand cercle est effectuée de la valeur de fin du bloc précédent jusqu'à la valeur de fin programmée.

Si le vecteur de début et le vecteur de fin sont programmés, une interpolation sur le grand cercle est également effectuée entre les deux directions. Ceci permet d'obtenir des trajectoires lisses.

Dans la configuration de base, les vecteurs normaux à la surface sont orientés dans la direction Z, indépendamment du plan G17 à G19 activé.

La longueur des vecteurs est sans importance.

La valeur zéro est attribuée par défaut aux composantes de vecteur non programmées.

Dans le cas où ORIWKS - voir chapitre "Référence des axes d'orientation" (ORIWKS, ORIMKS) - est activée, les vecteurs normaux à la surface se réfèrent au frame actif et subissent également une rotation en cas de rotation du frame.

Constructeur de la machine-outil

Le vecteur normal à la surface doit être perpendiculaire à la tangente à la trajectoire avec une zone de tolérance qui est réglable par le biais d'un paramètre machine, sinon une alarme est émise.

6.2.6 Référence des axes d'orientation (ORIWKS, ORIMKS)

Fonction

Lors de la programmation de l'orientation dans le système de coordonnées pièces par le biais

- des angles d'Euler ou RPY ou
- du vecteur d'orientation,

ORIMKS/ORIWKS permettent de régler la trajectoire du mouvement de rotation.

Remarque

Constructeur de la machine-outil

Le mode d'interpolation pour l'orientation est défini par le paramètre machine :

MD21104 $MC_ORI_IPO_WITH_G_CODE

= FALSE : Les fonctions G, ORIWKS et ORIMKS, servent de référence

= TRUE : Les fonctions G du groupe 51 (ORIAXES, ORIVECT, ORIPLANE, ...) servent de référence

Syntaxe

```
ORIMKS=...
ORIWKS=...
```

Signification

ORIMKS	Rotation dans le système de coordonnées machine
ORIWKS	Rotation dans le système de coordonnées pièce

Remarque

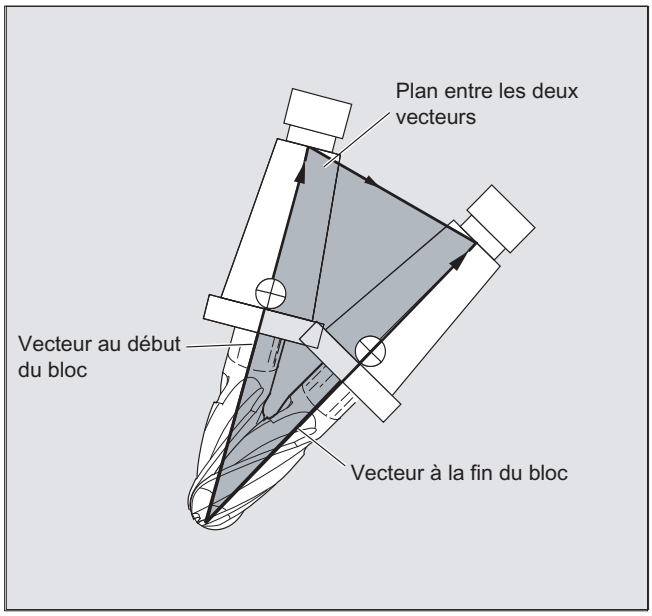
ORIWKS correspond au réglage de base. Si on ne connaît pas d'avance la machine sur laquelle doit fonctionner un programme à 5 axes, il faut par principe choisir ORIWKS. Les déplacements effectivement exécutés par la machine dépendent de sa cinématique.

ORIMKS permet de programmer les mouvements réels de la machine, p. ex. pour éviter des collisions avec des équipements, etc.

Description

Avec ORIMKS, le déplacement effectué par l'outil **dépend** de la cinématique de la machine. Dès qu'il y a un changement d'orientation avec pointe de l'outil fixe, il y a exécution d'une interpolation linéaire entre les positions des axes rotatifs.

Avec ORIWKS, le déplacement de l'outil est **indépendant** de la cinématique de la machine. Dès qu'il y a un changement d'orientation avec pointe de l'outil fixe, l'outil se déplace dans le plan défini par le vecteur de début et le vecteur de fin.



Positions singulières

Remarque

ORIWKS

Les mouvements d'orientation au voisinage de la position singulière d'une machine à 5 axes nécessitent des déplacements importants de la part des axes machine. (C'est le cas par exemple sur une tête pivotante et tournante, avec C comme axe de rotation et A comme axe de pivotement, toutes les positions avec A = 0 étant des positions singulières.)

Constructeur de la machine-outil

Pour ne pas surcharger les axes machine, le pilotage de la vitesse réduit fortement la vitesse tangentielle au voisinage des points singuliers.

Les paramètres machine

$MC_TRAFO5_NON_POLE_LIMIT

$MC_TRAFO5_POLE_LIMIT

permettent de paramétrer la transformation de façon à ce que les déplacements d'orientation au voisinage du pôle passent effectivement par le pôle, ce qui permet un usinage plus rapide.

Les positions singulières sont traitées uniquement avec le PM $MC_TRAFO5_POLE_LIMIT .

Bibliographie :
/FB3/ Manuel de fonctions spéciales ; Transformation 3 à 5 axes (F2),
chapitre "Positions singulières et leur traitement".

6.2.7 Programmation des axes d'orientation (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2)

Fonction

La fonction "Axes d'orientation" décrit l'orientation de l'outil dans l'espace et s'obtient en programmant les décalages pour les axes rotatifs. On peut obtenir un troisième degré de liberté par une rotation supplémentaire de l'outil autour de son propre axe. Cette orientation de l'outil s'effectue au choix dans l'espace par le biais d'un troisième axe rotatif et requiert la transformation six axes. La rotation de l'outil autour de son propre axe se fixe en fonction du type d'interpolation des vecteurs de rotation avec l'angle de rotation THETA, voir chapitre "Rotations de l'orientation de l'outil (ORIROTA/TR/TT, ORIROTC, THETA)".

Programmation

Les axes d'orientation se programment par le biais des descripteurs d'axes A2, B2, C2.

`N... ORIAXES` ou `ORIVECT` `N... G1 X Y Z A B C`	Interpolation linéaire ou circulaire de grand rayon
ou	ou
`N... ORIPLANE`	Interpolation d'orientation du plan
ou	ou
`N... ORIEULER` ou `ORIRPY voire ORIRPY2` `N... G1 X Y Z A2= B2= C2=`	Angle d'orientation d'Euler / RPY
ou	Programmation des angles des axes virtuels
`N... ORIVIRT1` ou `ORIVIRT2` `N... G1 X Y Z A3= B3= C3=`	ou Axes d'orientation virtuels Définition 1 ou 2 Programmation de vecteurs de direction

Pour les modifications d'orientation le long d'une enveloppe de cône située dans l'espace, on peut programmer d'autres décalages d'axes rotatifs des axes d'orientation, voir chapitre "Programmation de l'orientation le long d'une enveloppe de cône (ORIPLANE, ORICONxx).

Paramètres

`ORIAXES`	`Interpolation linéaire des axes machine ou d'orientation`
`ORIVECT`	`Interpolation circulaire de grand rayon (identique avec ORIPLANE)`
`ORIMKS`	`Rotation dans le système de coordonnées machine`
`ORIWKS`	`Rotation dans le système de coordonnées pièce`
	`Pour une description, voir le chapitre "Rotations de l'orientation de l'outil"`
`A= B= C=`	`Programmation de la position des axes machine`

ORIEULER	Programmation d'orientation définie par angles d'Euler
ORIRPY	Programmation d'orientation par l'angle RPY. L'ordre de rotation est XYZ, avec : A2 est l'angle de rotation autour de X B2 est l'angle de rotation autour de Y C2 est l'angle de rotation autour de Z
ORIRPY2	Programmation d'orientation par l'angle RPY. L'ordre de rotation est ZYX, avec : A2 est l'angle de rotation autour de Z B2 est l'angle de rotation autour de Y C2 est l'angle de rotation autour de X
A2= B2= C2=	Programmation des angles des axes virtuels
ORIVIRT1 ORIVIRT2	Programmation de l'orientation avec axes virtuels d'orientation (définition 1), selon PM $MC_ORIAX_TURN_TAB_1 (définition 2), selon PM $MC_ORIAX_TURN_TAB_2
A3= B3= C3=	Programmation du vecteur de direction de l'axe d'orientation

Description

Constructeur de la machine-outil

Le paramètre machine $MC_ORI_DEF_WITH_G_CODE règle le mode de définition des angles A2, B2, C2 programmés :

La définition s'effectue d'après le paramètre machine $MC_ORIENTATION_IS_EULER (par défaut) ou d'après le groupe de codes G 50 (ORIEULER, ORIRPY, ORIVIRT1, ORIVIRT2).

Le paramètre machine $MC_ORI_IPO_WITH_G_CODE permet de déterminer le type d'interpolation : ORIWKS/ORIMKS ou ORIAXES/ORIVECT.

Mode de fonctionnement Jog

Dans ce mode de fonctionnement, les angles d'orientation sont toujours interpolés de façon linéaire. En cas de positionnement incrémental et continu à l'aide des touches de déplacement, un seul axe d'orientation peut être déplacé. Avec les manivelles, le déplacement simultané des axes d'orientation est possible.

En déplacement manuel d'axes d'orientation, le commutateur de correction de l'avance ou le commutateur de correction du rapide en déplacement en rapide spécifique au canal est actif.

Les paramètres machine suivants permettent de définir séparément les vitesses :

$MC_JOG_VELO_RAPID_GEO

$MC_JOG_VELO_GEO

$MC_JOG_VELO_RAPID_ORI

$MC_JOG_VELO_ORI

Remarque

SINUMERIK 840D avec "Paquet de transformation pour manipulateurs"

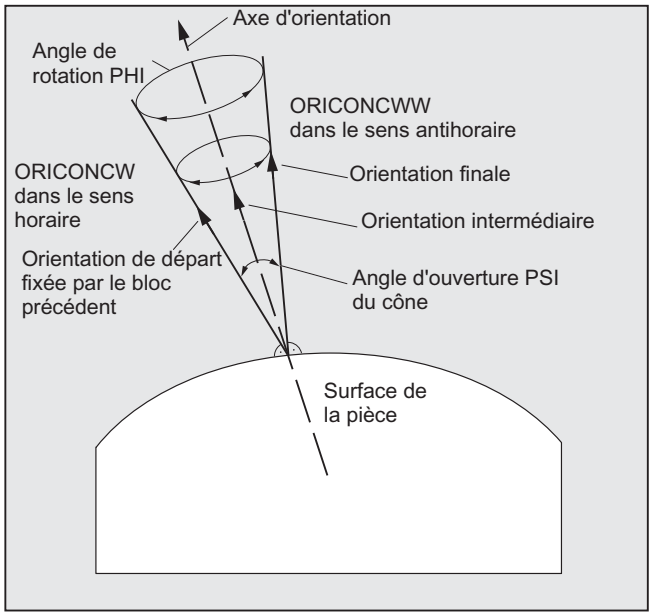
En mode JOG, la fonction "Déplacement manuel cartésien" permet de régler la translation d'axes géométriques dans les systèmes de référence SCM, SCP et TKS, séparément les uns des autres.

Bibliographie :
/FB2/ Description fonctionnelle Fonctions d'extension ; transformation cinématique (M1)

6.2.8 Programmation de l'orientation le long d'une enveloppe de cône (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO)

Fonction

L'orientation étendue permet d'effectuer des changements d'orientation le long d'une enveloppe de cône située dans l'espace. L'interpolation du vecteur d'orientation sur une enveloppe de cône s'effectue avec les instructions modales ORICONxx. Pour l'interpolation dans un plan, on peut programmer l'orientation finale avec ORIPLANE. D'une manière générale, l'orientation de départ est déterminée par les blocs précédents.



Programmation

L'orientation finale se détermine soit par l'indication de la programmation d'angles en angles d'Euler ou RPY avec A2, B2, C2 ou par la programmation des positions des axes rotatifs avec a, B, C. Pour les axes d'orientation le long de l'enveloppe de cône, d'autres indications de programmation sont nécessaires:

- Axe rotatif du cône sous forme de vecteur avec `A6, B6, C6`
- Angle au centre `PSI` avec les descripteurs `NUT`
- Orientation intermédiaire dans l'enveloppe de cône avec `A7, B7, C7`

Remarque

Programmation du vecteur de direction A6, B6, C6 pour l'axe rotatif du cône

La programmation d'une orientation finale n'est pas absolument nécessaire. Si aucune orientation finale n'est indiquée, une enveloppe de cône complète est interpolée avec 360 degrés.

Programmation de l'angle d'ouverture du cône avec NUT=angle

L'indication d'une orientation finale est impérativement nécessaire.

De cette manière, on ne peut pas interpoler une enveloppe de cône complète avec 360 degrés.

Programmation de l'orientation intermédiaire A7, B7, C7 dans l'enveloppe de cône

L'indication d'une orientation finale est impérativement nécessaire. Le changement d'orientation et le sens de rotation sont déterminés de manière univoque par les trois vecteurs Orientation de départ, Orientation finale et Orientation intermédiaire. Ce faisant, les trois vecteurs doivent être différents. Si l'orientation intermédiaire programmée est parallèle à l'orientation de départ ou à l'orientation finale, une interpolation circulaire de grand rayon de l'orientation est effectuée dans le plan défini entre le vecteur de départ et le vecteur final.

Interpolation d'orientation étendue sur une enveloppe de cône

```
N... ORICONCW ou ORICONCCW
N... A6= B6= C6= A3= B3= C3=
ou
N... ORICONTO
N... G1 X Y Z A6= B6= C6=
ou
N... ORICONIO
N... G1 X Y Z A7= B7= C7=
N... PO[PHI]=(a2, a3, A4, A5)
N... PO[PSI]=(b2, b3, b4, b5)
```

Interpolation sur une enveloppe de cône avec

vecteur de direction dans le sens horaire/antihoraire du cône et orientation finale ou

transition tangentielle et indication de l'orientation finale

ou

indication de l'orientation finale et d'une orientation intermédiaire dans l'enveloppe de cône avec

polynômes pourl'angle de rotation et polynômes pour l'angle d'ouverture

Paramètres

ORIPLANE	Interpolation dans le plan (interpolation circulaire de grand rayon)
ORICONCW	Interpolation sur une enveloppe de cône dans le sens horaire
ORICONCCW	Interpolation sur une enveloppe de cône dans le sens anti-horaire
ORICONTO	Interpolation sur une enveloppe de cône avec transition tangentielle
A6= B6= C6=	Programmation de l'axe rotatif du cône (vecteur normé)
NUT=angle	Angle au centre du cône en degrés
NUT=+179	Angle de déplacement inférieur ou égal à 180 degrés
NUT=-181	Angle de déplacement supérieur ou égal à 180 degrés
ORICONIO	Interpolation sur une enveloppe de cône
A7= B7= C7=	Orientation intermédiaire (programmation comme vecteur normé)
PHI	Angle de rotation de l'orientation autour de l'axe d'orientation du cône
PSI	Angle d'ouverture du cône
Polynômes possibles PO[PHI]=(a2, a3, A4, A5) PO[PSI]=(b2, b3, b4, b5)	En dehors des différents angles, on peut aussi programmer des polynômes au maximum du 5e degré

Exemple de différents changements d'orientation

```
…
N10 G1 X0 Y0 F5000
N20 TRAORI(1)            ; Transformation d'orientation activée.
N30 ORIVECT              ; Interpoler l'orientation de l'outil comme vecteur.
…                        ; Orientation de l'outil dans le plan.
N40 ORIPLANE             ; Sélectionner l'interpolation circulaire de grand rayon.
N50 A3=0 B3=0 C3=1
N60 A3=0 B3=1 C3=1       ; Orientation dans le plan Y/Z tournée de 45 degrés,
                           l'orientation (0,1/√2,1/√2) est atteinte en fin de bloc.
…
N70 ORICONCW             ; Programmation de l'orientation sur l'enveloppe de cône :
N80 A6=0 B6=0 C6=1 A3=0  ; Le vecteur d'orientation est interpolé dans le sens
B3=0 C3=1                  horaire sur une enveloppe de cône avec la direction
                           (0,0,1) jusqu'à l'orientation (1/√2,0,1/√2), l'angle de
                           rotation étant de 270 degrés.
N90 A6=0 B6=0 C6=1       ; L'orientation de l'outil effectue un tour complet sur la
                           même enveloppe de cône.
```

Description

Si l'on veut décrire des changements d'orientation sur une enveloppe de cône située à un endroit quelconque dans l'espace, il faut que le vecteur autour duquel la rotation de l'orientation de l'outil doit s'effectuer, soit connu. Il faut en outre que l'orientation de départ et l'orientation finale soient prédéfinies. L'orientation de départ découle du bloc précédent et l'orientation finale doit être soit programmée, soit déterminée par d'autres conditions.

La programmation dans le plan ORIPLANE correspond à ORIVECT

La programmation de l'interpolation circulaire de grand rayon conjugée aux polynômes trigonométriques correspond à l'interpolation linéaire et polynômiale de contours. L'orientation de l'outil est interpolée dans un plan défini par l'orientation de départ et l'orientation finale. Si l'on programme aussi des polynômes, il se peut que le vecteur d'orientation bascule également hors du plan.

Programmation de cercles dans un plan G2/G3, CIP et CT

L'orientation étendue correspond à l'interpolation de cercles dans un plan. En ce qui concerne les possibilités de programmation de cercles avec indication du centre ou du rayon comme G2/G3, cercle via point intermédiaire CIP et cercles tangentiels CT, voir

Bibliographie: Manuel de programmation Bases, "Programmation des instructions de déplacement".

Programmation de l'orientation

Interpolation du vecteur d'orientation sur une enveloppe de cône ORICONxx

Pour l'interpolation d'orientations sur une enveloppe de cône, on peut sélectionner quatre types d'interpolations différents dans le groupe 51 des codes G:

1. Interpolation sur une enveloppe de cône dans le sens horaire `ORICONCW` avec indication de l'orientation finale et de la direction du cône ou de son angle d'ouverture. Le vecteur de direction se programme avec les descripteurs `A6, B6, C6` et l'angle d'ouverture du cône avec le descripteur NUT= plage de valeurs dans l'intervalle 0 à 180 degrés.
2. Interpolation sur une enveloppe de cône dans le sens antihoraire `ORICONCWW` avec indication de l'orientation finale et de la direction du cône ou de son angle d'ouverture. Le vecteur de direction se programme avec les descripteurs `A6, B6, C6` et l'angle d'ouverture du cône avec le descripteur NUT= plage de valeurs dans l'intervalle 0 à 180 degrés.
3. Interpolation sur une enveloppe de cône `ORICONIO` avec indication de l'orientation finale et d'une orientation intermédiaire, programmée avec les descripteurs `A7, B7, C7`.
4. Interpolation sur une enveloppe de cône `ORICONTO` avec transition tangentielle et indication de l'orientation finale. Le vecteur de direction se programme avec les descripteurs `A6, B6, C6`.

6.2.9 Définition de l'orientation de deux points de contact (ORICURVE, PO[XH]=, PO[YH]=, PO[ZH]=)

Fonction

Programmation du changement d'orientation par la deuxième courbe spatiale ORICURVE

Une autre possibilité pour programmer les changements d'orientation est de programmer avec `ORICURVE`, outre la pointe de l'outil le long d'une courbe spatiale, également le mouvement d'un deuxième point de contact de l'outil. On peut ainsi, comme lors de la programmation du vecteur de l'outil lui-même, définir de manière univoque les changements d'orientation de l'outil.

Constructeur de la machine-outil

Veuillez respecter les indications du constructeur de la machine-outil en ce qui concerne les descripteurs d'axes réglables par le biais de paramètres machine pour la programmation de la deuxième trajectoire d'orientation de l'outil.

Programmation

Dans le cas de ce type d'interpolation, on peut, pour les deux courbes spatiales, programmer des points avec `G1` ou des polynômes avec `POLY`. Les cercles et les développantes ne sont pas admis. De plus, une interpolation de type spline peut être activée avec `BSPLINE` et avec une fonction "groupement de blocs spline courts".

Bibliographie :
/FB1/ Description fonctionnelle Fonctions de base ; contournage, arrêt précis, look-ahead (B1), chapitre : Groupement de blocs spline courts

Les autres types de spline `ASPLINE` et `CSPLINE` tout comme l'activation d'un compacteur avec `COMPON`, `COMPCURV` ou `COMPCAD` ne sont pas admissibles.

Le déplacement des deux points de contact de l'outil peut être prédéfini lors de la programmation des polynômes d'orientation pour des coordonnées du 5e degré maximum.

Interpolation d'orientation étendue avec courbe spatiale supplémentaire et polynômes pour coordonnées

```
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)
```

Indication du déplacement du deuxième point de contact de l'outil et polynômes supplémentaires des coordonnées correspondantes

Paramètres

ORICURVE	Interpolation de l'orientation avec indication du déplacement de deux points de contact de l'outil.
XH YH ZH	Descripteur des coordonnées du deuxième point de contact de l'outil du contour supplémentaire en tant que courbe spatiale
Polynômes possibles PO[XH]=(xe, x2, x3, x4, x5) PO[YH]=(ye, y2, y3, y4, y5) PO[ZH]=(ze, z2, z3, z4, z5)	Les courbes spatiales peuvent se programmer avec les points de fin correspondants, mais en plus aussi avec des polynômes.
xe, ye, ze	Points de fin de la courbe spatiale
xi, yi, zi	Coefficients des polynômes du 5e degré maximum

Remarque

Descripteurs XH YH ZH pour la programmation d'une 2e trajectoire d'orientation

On doit choisir les descripteurs de manière à ce qu'il ne se produise pas de conflits avec les autres descripteurs des axes linéaires

, les axes X Y Z

et les axes rotatifs tels que

A2 B2 C2, les angles d'Euler ou les angles RPY

A3 B3 C3, les vecteurs de direction

A4 B4 C4 ou A5 B5 C5, les vecteurs normaux à la surface

A6 B6 C6, les vecteurs de rotation ou les coordonnées de points intermédiaires A7 B7 C7

ou d'autres paramètres d'interpolation.

6.3 Polynômes d'orientation (PO[angles], PO[coordonnée])

Fonction

Indépendamment de l'interpolation polynomiale du groupe 1 des codes G actuellement activée, on peut programmer deux types de polynômes d'orientation jusqu'au 5e degré maximum dans le cas d'une transformation trois à cinq axes.

1. Polynômes pour **angles:** Angle d'avance LEAD, angle latéral TILT par rapport au plan défini par l'orientation de départ et l'orientation finale.
2. Polynômes pour **coordonnées:** XH, YH, ZH de la deuxième courbe spatiale pour l'orientation de l'outil d'un point de référence sur l'outil.

Dans le cas d'une transformation six axes, on peut, en plus de l'orientation de l'outil, programmer la rotation du vecteur de rotation THT avec des polynômes du 5e degré maximum pour les rotations de l'outil lui-même.

Syntaxe

Polynômes d'orientation du type 1 pour **angles**

```
N... PO[PHI]=(a2, a3, A4, A5)
N... PO[PSI]=(b2, b3, b4, b5)
```

Transformation trois à cinq axes

Transformation trois à cinq axes

Polynômes d'orientation du type 2 pour **coordonnées**

```
N… PO[XH]=(xe, x2, x3, x4, x5)
N… PO[YH]=(ye, y2, y3, y4, y5)
N… PO[ZH]=(ze, z2, z3, z4, z5)
```

Descripteur pour les coordonnées de la deuxième trajectoire d'orientation pour l'orientation de l'outil

On peut en outre, dans les deux cas, programmer un polynôme pour la **rotation** dans le cas d'une transformation six axes avec

```
N… PO[THT]=(c2, c3, c4, c5)
```

ou

```
N… PO[THT]=(d2, d3, d4, d5)
```

Interpolation par rapport à la trajectoire de la rotation

Interpolation absolue, relative et tangentielle du vecteur d'orientation par rapport au changement d'orientation

Ceci est possible si la transformation supporte un vecteur de rotation avec un décalage interpolable et programmable par le biais de l'angle de rotation THETA.

Signification

`PO[PHI]`	Angle dans le plan entre l'orientation de départ et l'orientation finale
`PO[PSI]`	Angle décrivant le basculement de l'orientation depuis le plan entre l'orientation de départ et l'orientation finale.
`PO[THT]`	Angle de rotation décrit par la rotation du vecteur de rotation d'un des codes G du groupe 54 programmés avec THETA
`PHI`	Angle d'avance LEAD
`PSI`	Angle latéral TILT
`THETA`	Rotation par rapport à la direction de l'outil en Z
`PO[XH]`	Coordonnée X du point de référence sur l'outil
`PO[YH]`	Coordonnée Y du point de référence sur l'outil
`PO[ZH]`	Coordonnée Z du point de référence sur l'outil

Description

On ne peut pas programmer les polynômes d'orientation

- si les interpolations spline ASPLINE, BSPLINE, CSPLINE sont activées.
  Les polynômes du type 1 pour les angles d'orientation sont possibles pour chaque type d'interpolation sauf Spline c.-à-d. pour l'interpolation linéaire avec vitesse rapide G00 ou avec avance G01
  pour l'interpolation polynômiale avec POLY et
  pour l'interpolation circulaire ou à développante avec G02, G03, CIP, CT, INVCW et INCCCW

  Les polynômes du type 2 pour les coordonnées d'orientation ne sont en revanche possibles que si
  l'interpolation linéaire avec vitesse rapide G00 ou avec avance G01 ou
  l'interpolation polynômiale avec POLY est activée.
- si l'orientation est interpolée au moyen de l'interpolation d'axes ORIAXES. Dans ce cas, on peut programmer directement des polynômes avec PO[A] et PO[B] pour les axes d'orientation A et B.

Polynômes d'orientation du type 1 avec ORIVECT, ORIPLANE et ORICONxx

En cas d'interpolation circulaire de grand rayon et d'interpolation sur enveloppe de cône avec ORIVECT, ORIPLANE et ORICONxx, seuls les polynômes d'orientation du type 1 sont possibles.

Polynômes d'orientation du type 2 avec ORICURVE

Si l'interpolation est activée avec la courbe spatiale supplémentaire ORICURVE, les composantes cartésiennes du vecteur d'orientation sont interpolées et seuls les polynômes d'orientation du type 2 sont possibles.

6.4 Rotations de l'orientation de l'outil (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA)

Fonction

Si l'orientation de l'outil doit pouvoir être modifiée sur les machines avec outil mobile, chaque bloc doit être programmé avec une orientation finale. En fonction de la cinématique de la machine, le sens d'orientation des axes d'orientation ou le sens de rotation du vecteur d'orientation THETA peuvent être programmés. Différents types d'interpolation sont programmables pour ces vecteurs de rotation :

- ORIROTA : Angle de rotation rapporté à un sens de rotation en indication absolue.
- ORIROTR : Angle de rotation relatif par rapport au plan entre orientation de départ et orientation finale.
- ORIROTT : Angle de rotation relatif par rapport à la modification du vecteur d'orientation.
- ORIROTC: Angle de rotation tangentiel à la tangente à la trajectoire.

Syntaxe

A la condition expresse que le type d'interpolation `ORIROTA` soit actif, l'angle de rotation ou le vecteur de rotation peut être programmé selon l'un des quatre modes suivants :

1. Directement dans les positions d'axes rotatifs `A, B, C`
2. En angles d'Euler (en degrés) par le biais de `A2, B2, C2`
3. En angles RPY (en degrés) par le biais de `A2, B2, C2`
4. En vecteur de direction par le biais de `A3, B3, C3` { (angle de rotation à l'aide de `THETA=valeur`).

Si `ORIOTR` ou `ORIOTT` est actif, l'angle de rotation ne peut être programmé que directement à l'aide de `THETA` .

Une rotation peut également être programmée seule dans un bloc, sans qu'un changement d'orientation ne se produise. ORIROTR `et` ORIROTT `n'ont` alors aucune signification. Dans ce cas, l'angle de rotation est toujours interprété par rapport à la direction absolue (`ORIROTA`).

`N... ORIROTA`
`N... ORIROTR`
`N... ORIROTT`
`N... ORIROTC`
	Définition de l'interpolation du vecteur de rotation

`N... A3= B3= C3= THETA=valeur`	Définir la rotation du vecteur d'orientation
`N... PO[THT]=(d2, d3, d4, d5)`	Interpolation de l'angle de rotation à l'aide d'un polynôme du cinquième degré

Signification

`ORIROTA`	Angle de rotation rapporté à un sens de rotation en indication absolue
`ORIROTR`	Angle de rotation relatif par rapport au plan entre orientation de départ et orientation finale
`ORIROTT`	Angle de rotation comme vecteur de rotation tangentiel au changement d'orientation
`ORIROTC`	Angle de rotation comme vecteur de rotation tangentiel à la tangente à la trajectoire
`THETA`	Rotation du vecteur d'orientation
`THETA=valeur`	Angle de rotation en degrés atteint en fin de bloc
`THETA=Θe`	Angle de rotation avec angle final Θ_e du vecteur de rotation
`THETA=AC(…)`	Passage bloc par bloc à l'introduction des cotes en valeurs absolues
`THETA=AC(…)`	Passage bloc par bloc à l'introduction des cotes relatives
`Θe`	L'angle final du vecteur de rotation est actif aussi bien en valeurs absolues avec G90 qu'en valeurs relatives avec G91 (introduction de cotes relatives).
`PO[THT]=(....)`	Polynôme pour l'angle de rotation

Exemple de rotations des orientations

```
Code de programme                          Commentaire
N10 TRAORI                                 ; Activer la transformation d'orientation
N20 G1 X0 Y0 Z0 F5000                      ; Orientation de l'outil
N30 A3=0 B3=0 C3=1 THETA=0                 ; En direction Z avec angle de rotation 0
N40 A3=1 B3=0 C3=0 THETA=90                ; En direction X et rotation de 90 degrés
N50 A3=0 B3=1 C3=0 PO[THT]=(180,90)        ; Aperçu
N60 A3=0 B3=1 C3=0 THETA=IC(-90)           ; En direction Y et rotation sur 180 degrés
N70 ORIROTT                                ; Demeure constant et rotation sur 90 degrés
N80 A3=1 B3=0 C3=0 THETA=30                ; Angle de rotation par rapport au changement
                                             d'orientation
                                           ; Vecteur de rotation formant un angle de 30
                                             degrés par rapport au plan X-Y
```

Lors de l'interpolation du bloc
`N40`, l'angle de rotation est interpolé de manière linéaire de la valeur de départ 0 degré à la valeur finale 90 degrés. Dans le bloc `N50`, l'angle de rotation passe de 90 à 180 degrés conformément à la
parabole $\theta(u) = +90u^2$. En `N60`, une rotation peut également être effectuée sans qu'il se produise de changement d'orientation.

En `N80`, l'orientation de l'outil tourne, passant de la direction Y à la direction X. Ce faisant, le changement d'orientation s'opère dans le plan X-Y et le vecteur de rotation forme un angle de 30 degrés par rapport à ce plan.

Description

ORIROTA

L'angle de rotation `THETA` est interpolé dans l'espace par rapport à une direction déterminée de manière absolue. Le sens de rotation par défaut est fonction des paramètres machine.

ORIROTR

L'angle de rotation `THETA` est interprété de manière relative par rapport au plan défini par l'orientation initiale et l'orientation finale.

ORIROTT

L'angle de rotation `THETA` est interprété de manière relative par rapport au changement d'orientation. Pour`THETA=0`, le vecteur de rotation est interpolé de manière tangentielle par rapport au changement d'orientation et ne se distingue d'`ORIROTR` que si au moins un polynôme pour l'"angle de basculement PSI" a été programmé pour l'orientation. Il en résulte un changement d'orientation qui ne se produit pas dans le plan. La programmation d'un angle de rotation `THETA` supplémentaire permet d'interpoler par exemple le vecteur de rotation de manière à ce qu'il forme toujours une valeur définie par rapport au changement d'orientation.

ORIROTC

Le vecteur de rotation est interpolé par rapport à la tangente à la trajectoire avec un décalage programmable par le biais de l'angle `THETA` . Pour l'angle de décalage, on peut aussi programmer un polynôme `PO[THT]=(c2, c3, c4, c5)` du 5e degré maximum.

6.5 Orientations par rapport à la trajectoire

6.5.1 Types d'orientation par rapport à la trajectoire

Fonction

Cette fonction étendue permet d'atteindre l'orientation relative non seulement en fin de bloc, mais aussi sur tout le tracé de la trajectoire. L'orientation atteinte au cours du bloc précédent est transférée au moyen de l'interpolation circulaire de grand rayon dans l'orientation finale programmée. Par principe, il existe deux possibilités pour programmer l'orientation souhaitée par rapport à la trajectoire:

1. L'orientation de l'outil et la rotation de l'outil sont interpolées avec ORIPATH, ORPATHTS par rapport à la trajectoire.
2. Le vecteur d'orientation est programmé et interpolé de la manière habituelle. La rotation du vecteur d'orientation par rapport à la tangente à la trajectoire s'opère avec ORIROTC.

Syntaxe

Le type d'interpolation de l'orientation et de la rotation de l'outil se programme avec:

`N... ORIPATH`	Orientation par rapport à la trajectoire
`N... ORIPATHS`	Orientation par rapport à la trajectoire avec lissage du tracé d'orientation
`N... ORIROTC`	Interpolation par rapport à la trajectoire du vecteur de rotation

Avec `ORIPATHS`, on peut lisser un coude de l'orientation provoqué par un angle dans le tracé de la trajectoire. La direction et la longueur de parcours du mouvement de retrait se programme par le biais du vecteur avec les composants `A8=X, B8=Y C8=Z .`

Avec ORIPATH/ORIPATHS , on peut programmer différentes références par rapport à la tangente à la trajectoire par le biais des trois angles

- LEAD= indication de l'angle d'avance par rapport à la trajectoire et à la surface
- TILT= indication de l'angle latéral par rapport à la trajectoire et à la surface
- THETA= angle de rotation

pour l'ensemble du tracé de la trajectoire. En plus de l'angle de rotation THETA, on peut aussi programmer des polynômes du 5e degré maximum avec PO[THT]=(...).

Remarque

Constructeur de la machine-outil

Veuillez tenir compte des indications du constructeur de la machine. Par le biais de paramètres machine et de données de réglage configurables, on peut procéder à d'autres réglages en plus du type d'orientation par rapport à la trajectoire. Pour plus d'explications, se reporter à

Bibliographie :
/FB3/ Description fonctionnelle Fonctions spéciales ; Transformation 3 à 5 axes (F2), Chapitre "Orientation"

Signification

L'interpolation des angles LEAD et TILT peut se régler différemment par le biais de paramètres machine:

- La référence de l'orientation de l'outil programmée avec LEAD et TILT est respectée pendant toute la durée du bloc.
- Angle d'avance LEAD: Rotation par rapport à la direction perpendiculaire à la tangente et vecteur normal TILT: Rotation de l'orientation par rapport au vecteur normal.
- Angle d'avance LEAD: Rotation par rapport à la direction perpendiculaire à la tangente et vecteur normal Angle latéral TILT: Rotation de l'orientation par rapport à la direction de la tangente à la trajectoire.
- Angle de rotation THETA: Rotation de l'outil autour de son propre axe avec un troisième axe rotatif supplémentaire en tant qu'axe d'orientation dans le cas de transformation six axes.

Remarque

Orientation relative à la trajectoire non autorisée avec OSC, OSS, OSSE, OSD, OST

L'interpolation d'orientation par rapport à la trajectoire ORIPATH ou ORIPATHS et ORIOTC ne peut pas être programmée avec le lissage du tracé d'orientation avec l'un des codes G du groupe 34. Pour ce faire, il faut qu'OSOF soit activé.

6.5.2 Rotation de l'orientation de l'outil relative à la trajectoire (ORIPATH, ORIPATHS, angle de rotation)

Fonction

Dans le cas d'une transformation six axes, en plus de l'orientation de l'outil d'une manière quelconque dans l'espace, ce dernier peut également tourner autour de son propre axe avec un troisième axe supplémentaire. Dans le cas d'une rotation de l'orientation de l'outil par rapport à la trajectoire avec ORIPATH ou ORIPATHS, la rotation supplémentaire peut se programmer par le biais de l'angle de rotation THETA. En guise d'alternative, les angles LEAD et TILT peuvent être programmés par le biais d'un vecteur perpendiculaire à la direction de l'outil dans le plan.

Constructeur de la machine-outil

Tenez compte des indications du constructeur de la machine. L'interpolation des angles LEAD et TILT peut se régler différemment par le biais de paramètres machine.

Syntaxe

Rotation de l'orientation de l'outil et de l'outil

On active le type d'orientation de l'outil par rapport à la trajectoire avec ORIPATH ou ORIPATHS.

`N... ORIPATH`	Activer le type d'orientation par rapport à la trajectoire
`N... ORIPATHS`	Activer le type d'orientation par rapport à la trajectoire avec lissage du tracé d'orientation

Activation des trois angles possibles avec effet de rotation:

`N... LEAD=`	Angle pour l'orientation programmée par rapport au vecteur normal à la surface
`N... TILT=`	Angle pour l'orientation programmée dans le plan perpendiculaire à la tangente à la trajectoire par rapport au vecteur normal à la surface
`N... THETA=`	;Angle de rotation par rapport au changement d'orientation par rapport à la direction d'outil du troisième axe rotatif

Les valeurs des angles en fin de bloc se programment avec `LEAD=valeur`, `TILT=valeur` ou `THETA=valeur`. En plus des angles constants, on peut programmer des polynômes du 5e degré maximum pour les trois angles.

`N... PO[PHI]=(a2, a3, A4, A5)`	Polynôme de l'angle d'avance LEAD
`N... PO[PSI]=(b2, b3, b4, b5)`	Polynôme de l'angle latéral TILT
`N... PO[THT]=(d2, d3, d4, d5)`	Polynôme de l'angle de rotation THETA

Pendant la programmation, on peut laisser de côté les coefficients de polynôme plus élevés, égaux à zéro. Exemple `PO[PHI]]=a2` donne une parabole pour l'angle d'avance `LEAD`.

Signification

Orientation de l'outil par rapport à la trajectoire

`ORIPATH`	Orientation de l'outil par rapport à la trajectoire
`ORIPATHS`	Orientation de l'outil par rapport à la trajectoire, lissage de coude dans le tracé d'orientation
`LEAD`	Angle par rapport au vecteur normal à la surface, dans le plan défini par la tangente à la trajectoire et le vecteur normal à la surface
`TILT`	Rotation de l'orientation par rapport à la direction Z ou rotation par rapport à la tangente à la trajectoire
`THETA`	Rotation par rapport à la direction de l'outil vers Z
`PO[PHI]`	Polynôme d'orientation pour l'angle d'avance LEAD
`PO[PSI]`	Polynôme d'orientation pour l'angle latéral TILT
`PO[THT]`	Polynôme d'orientation pour l'angle de rotation THETA

Remarque

Angle de rotation THETA

Pour la rotation de l'outil avec un troisième axe rotatif en tant qu'axe d'orientation par rapport à lui-même, une transformation six axes est nécessaire.

6.5.3 Interpolation relative à la trajectoire de la rotation de l'outil (ORIROTC, THETA)

Fonction

Type d'interpolation avec vecteurs de rotation

En plus de la rotation de l'outil (programmée avec ORIROTC) par rapport à la tangente à la trajectoire, on peut aussi interpoler le vecteur de rotation avec un décalage programmable par le biais de l'angle de rotation THETA. À cette occasion, on peut, avec PO[THT], programmer un polynôme du 5e degré maximum pour l'angle de décalage.

Syntaxe

`N... ORIROTC`	Appliquer la rotation de l'outil par rapport à la tangente à la trajectoire
`N... A3= B3= C3= THETA=valeur`	Définir la rotation du vecteur d'orientation
`N... A3= B3= C3= PO[THT]=(c2, c3, c4, c5)`	Interpolation de l'angle de décalage avec polynôme du 5e degré au maximum

Une rotation peut également être programmée seule dans un bloc, sans qu'un changement d'orientation ne se produise.

Signification

Interpolation (par rapport à la trajectoire) de la rotation de l'outil dans le cas d'une transformation six axes

`ORIROTC`	Appliquer le vecteur de rotation tangentiel à la tangente à la trajectoire
`THETA=valeur`	Angle de rotation en degrés atteint en fin de bloc
`THETA=θe`	Angle de rotation avec angle final Θ_e du vecteur de rotation
`THETA=AC(…)`	Passage bloc par bloc à l'introduction des cotes en valeurs absolues
`THETA=IC(…)`	Passage bloc par bloc à l'introduction des cotes relatives
`PO[THT]=(c2, c3, c4, c5)`	Interpolation de l'angle de décalage à l'aide d'un polynôme du 5e degré

Remarque

Interpolation du vecteur de rotation ORIROTC

Si l'on veut appliquer également la rotation de l'outil par rapport à la tangente à la trajectoire dans le sens opposé à la direction d'orientation de l'outil, ce n'est possible que dans le cas d'une transformation six axes.

Dans le cas où ORIROTC est activé

Le vecteur de rotation ORIROTA ne peut pas être programmé. Dans le cas d'une programmation, le message d'alarme 14128 "Programmation absolue de la rotation de l'outil dans le cas où ORIROTC est activé" est émis.

Direction d'orientation de l'outil dans le cas d'une transformation trois à cinq axes

La direction d'orientation de l'outil peut se programmer comme on en a l'habitude avec une transformation trois à cinq axes par le biais d'angles d'Euler ou d'angles RPY ou de vecteurs de direction. Également possibles: changements d'orientation de l'outil dans l'espace par programmation de l'interpolation circulaire de grand rayon ORIVECT, de l'interpolation linéaire des axes d'orientation ORIAXES, de toutes les interpolations sur une enveloppe de cône ORICONxx ainsi que de l'interpolation en plus de la courbe spatiale avec deux points de contact de l'outil ORICURVE.

`G....`	Indication du type de déplacement des axes rotatifs
`X Y Z`	Indication des axes linéaires
`ORIAXES`	Interpolation linéaire des axes machine ou d'orientation
`ORIVECT`	Interpolation circulaire de grand rayon (identique avec ORIPLANE)
`ORIMKS`	Rotation dans le système de coordonnées machine
`ORIWKS`	Rotation dans le système de coordonnées pièce Pour une description, voir le chapitre "Rotations de l'orientation de l'outil"
`A= B= C=`	Programmation de la position des axes machine

`ORIEULER`	Programmation d'orientation par le biais d'angles d'Euler
`ORIRPY`	Programmation de l'orientation par le biais d'angles RPY
`A2= B2= C2=`	Programmation des angles des axes virtuels
`ORIVIRT1` `ORIVIRT2`	Programmation de l'orientation par le biais d'axes d'orientation virtuels (définition 1), selon PM $MC_ORIAX_TURN_TAB_1 (définition 2), selon PM $MC_ORIAX_TURN_TAB_2
`A3= B3= C3=`	Programmation du vecteur de direction de l'axe d'orientation
`ORIPLANE`	Interpolation dans le plan (interpolation circulaire de grand rayon)
`ORICONCW`	Interpolation sur une enveloppe de cône dans le sens horaire
`ORICONCCW`	Interpolation sur une enveloppe de cône dans le sens antihoraire
`ORICONTO`	Interpolation sur une enveloppe de cône avec transition tangentielle
`A6= B6= C6=`	Programmation de l'axe rotatif du cône (vecteur normé)
`NUT=angle`	Angle au centre du cône en degrés
`NUT=+179`	Angle de déplacement inférieur ou égal à 180 degrés
`NUT=-181`	Angle de déplacement supérieur ou égal à 180 degrés
`ORICONIO`	Interpolation sur une enveloppe de cône
`A7= B7= C7=`	Orientation intermédiaire (programmation comme vecteur normé)
`ORICURVE` `XH YH ZH par exemple avec polynôme PO[XH]=(xe, x2, x3, x4, x5)`	Interpolation de l'orientation avec indication du déplacement de deux points de contact de l'outil. Outre le(s) point(s) final(aux) en question, des courbes spatiales supplémentaires sont programmables sous forme de polynômes.

Remarque

Si l'orientation de l'outil est interpolée par le biais des axes d'orientation alors qu'ORIAXES est activé, l'application de l'angle de rotation par rapport à la trajectoire n'est réalisée qu'en fin de bloc.

6.5.4 Lissage du tracé d'orientation (ORIPATHS A8=, B8=, C8=)

Fonction

Dans le cas de changements d'orientation à accélération continue sur le contour, toute interruption des déplacements avec interpolation, pouvant notamment survenir à un angle du contour, est inopportune. Le coude qui en résulte dans le tracé d'orientation peut être lissé en ajoutant un bloc intermédiaire qui lui est propre. Le changement d'orientation s'effectue alors à accélération continue même si ORIPATHS est actif pendant la réorientation. Pendant cette phase, un mouvement de retrait de l'outil peut être exécuté.

Constructeur de la machine-outil

Veuillez respecter les indications du constructeur de la machine-outil en ce qui concerne les paramètres machine et les données de réglage prédéfinis le cas échéant et avec lesquels cette fonction est activée.

Par le biais des paramètres machine, on peut définir comment interpréter le vecteur de retrait.

1. Dans le système de coordonnées de l'outil, on définit la coordonnée Z par la direction de l'outil.
2. Dans le système de coordonnées pièce, on définit la coordonnée Z par le plan actif.

Pour plus d'explications sur la fonction "Orientation par rapport à la trajectoire", se reporter à **Bibliographie:** /FB3/ Manuel de fonctions spéciales ; Transformation 3 à 5 axes (F2).

Syntaxe

Pour des orientations d'outil continues par rapport à l'ensemble de la trajectoire, d'autres indications de programmation sont nécessaires à un angle du contour. La direction et la longueur de parcours de ce mouvement se programme par le biais du vecteur avec les composants A8=X, B8=Y C8=Z:

```
N... ORIPATHS A8=X B8=Y C8=Z
```

Signification

`ORIPATHS`	Orientation de l'outil par rapport à la trajectoire, lissage de coude dans le tracé d'orientation.
`A8= B8= C8=`	Composants de vecteur pour la direction et la longueur de parcours
`X, Y, Z`	Mouvement de retrait dans la direction de l'outil

Remarque

Programmation du vecteur de direction A8, B8, C8

Si la longueur de ce vecteur est égale à zéro, il ne se produit pas de mouvement de retrait.

ORIPATHS

L'orientation de l'outil par rapport à la trajectoire est activée avec ORIPATHS. Dans le cas contraire, l'orientation est transférée par le biais d'une interpolation circulaire de grand rayon de l'orientation de départ vers l'orientation finale.

6.6 Compactage de l'orientation (COMPON, COMPCURV, COMPCAD)

Fonction

Les programmes CN dans lesquels une transformation d'orientation (TRAORI) est active et dans lesquels l'orientation est programmée par le biais de vecteurs de direction peuvent être compactés en respectant des tolérances prédéfinies.

Remarque

Le mouvement d'orientation n'est compacté que si l'interpolation circulaire de grand rayon est activée et dépend donc du code G pour l'interpolation d'orientation. Celui-ci se règle, tout comme la longueur de parcours maximale et une tolérance admise pour chaque axe ou pour l'avance tangentielle pour la fonction compacteur, par le biais de paramètres machine. Veuillez tenir compte des indications du constructeur de la machine.

Programmation

Orientation de l'outil

Si une transformation d'orientation (TRAORI) est active, l'orientation de l'outil peut être programmée de la manière suivante (indépendante de la cinématique) sur les machines à 5 axes :

- Programmation du **vecteur** de direction avec :

  ```
  A3=<...> B3=<...> C3=<...>
  ```

- Programmation des **angles** d'Euler ou des **angles** RPY avec :

  ```
  A2=<...> B2=<...> C2=<...>
  ```

Rotation de l'outil

Sur les machines à **6 axes**, on peut programmer la rotation de l'outil en plus de son orientation.

La programmation de l'angle de rotation s'effectue avec :

```
THETA=<...>
```

Voir "Rotations de l'orientation de l'outil (Page 343)".

Remarque

Les blocs CN dans lesquels la rotation est programmée en plus ne sont compactables que si l'angle de rotation varie **linéairement**. Donc il ne doit pas y avoir de polynôme programmé avec PO[THT]=(...) pour l'angle de rotation.

Forme générale d'un bloc CN compactable

La forme générale d'un bloc CN compactable peut donc être la suivante :

```
N... X=<...> Y=<...> Z=<...> A3=<...> B3=<...> C3=<...> THETA=<...> F=<...>
```

ou

```
N... X=<...> Y=<...> Z=<...> A2=<...> B2=<...> C2=<...> THETA=<...> F=<...>
```

Remarque

Les valeurs de position peuvent être indiquées directement (par exemple X90) ou indirectement par le biais d'affectations de paramètres (par exemple X=R1*(R2+R3)).

Programmation de l'orientation de l'outil par des positions d'axes rotatifs

L'orientation de l'outil peut également être indiquée par des positions d'axes rotatifs, par exemple sous la forme suivante :

```
N... X=<...> Y=<...> Z=<...> A=<...> B=<...> C=<...> THETA=<...> F=<...>
```

Dans ce cas, le compactage est effectué de deux manières différentes selon qu'une interpolation circulaire de grand rayon est exécutée ou non. Si une interpolation circulaire de grand rayon a lieu, le changement d'orientation compacté est représenté de manière habituelle par des polynômes axiaux pour les axes rotatifs.

Précision du contour

Selon le mode de compactage réglé (MD20482 $MC_COMPRESSOR_MODE), les tolérances qui prennent effet pour les axes géométriques et les axes d'orientation lors du compactage sont soit les tolérances spécifiques aux axes (configurées dans le paramètre machine MD33100 $MA_COMPRESS_POS_TOL) ou les tolérances spécifiques aux canaux (réglables dans les données de réglage) suivantes :

SD42475 $SC_COMPRESS_CONTUR_TOL (écart maximal par rapport au contour)

SD42476 $SC_COMPRESS_ORI_TOL (écart maximal par rapport à l'angle d'orientation de l'outil)

SD42477 $SC_COMPRESS_ORI_ROT_TOL (écart maximal par rapport à l'angle de rotation de l'outil) (disponible uniquement pour les machines à 6 axes)

Bibliographie :
Description fonctionnelle Fonctions de base ; Transformation 3 à 5 axes (F2),
chapitre : "Compactage de l'orientation"

Activation/désactivation

Les codes G à effet modal COMPON, COMPCURV et COMPCAD activent les fonctions compacteur.

La fonction compacteur se termine avec COMPOF.

Voir "Compactage de bloc CN (COMPON, COMPCURV, COMPCAD) (Page 247)".

Remarque

Le déplacement d'orientation est uniquement compacté lorsque l'interpolation circulaire de grand diamètre est activée (l'orientation de l'outil se modifie dans le plan défini par l'orientation initiale et l'orientation finale).

Une interpolation circulaire de grand diamètre est exécutée dans les conditions suivantes :

- MD21104 $MC_ORI_IPO_WITH_G_CODE = 0,
  ORIWKS est actif et
  l'orientation est programmée au moyen de vecteurs (avec A3, B3, C3 et A2, B2, C2).
- MD21104 $MC_ORI_IPO_WITH_G_CODE = 1 et
  ORIVECT ou ORIPLANE est actif.

  L'orientation de l'outil peut être programmée sous forme de vecteur de direction ou de positions d'axes rotatifs. Si l'un des codes G ORICONxx et ORICURVE est actif ou si des polynômes sont programmés pour les angles d'orientation (PO[PHI] et PO[PSI]), il n'y a pas d'interpolation circulaire de grand rayon.

Exemple

Dans cet exemple de programmation, un cercle, qui est obtenu par une approximation polygonale, est compacté. L'orientation de l'outil se déplace de façon synchrone sur une surface latérale de cône. Bien que les changements successifs d'orientation programmés soient discontinus, la fonction compacteur génère un tracé lisse de l'orientation.

```
Programmation                    Commentaire
DEF INT NOMBRE=60
DEF REAL RAYON=20
DEF INT COMPTEUR
DEF REAL ANGLE
N10 G1 X0 Y0 F5000 G64

$SC_COMPRESS_CONTUR_TOL=0.05   ;  Ecart maximal par rapport au contour = 0.05 mm
$SC_COMPRESS_ORI_TOL=5         ;  Ecart maximal par rapport à l'orientation
                                  = 5 degrés


TRAORI
COMPCURV
                               ;  Le déplacement a lieu sur un cercle obtenu par une
                                  approximation polygonale. L'orientation se déplace
                                  sur un cône autour de l'axe Z avec un angle au
                                  centre de 45 degrés.
N100 X0 Y0 A3=0 B3=-1 C3=1
N110 FOR COUNTER=0 TO NOMBRE
N120 ANGLE=360*COUNTER/NOMBRE
N130 X=RAYON*cos(ANGLE) Y=RAYON*sin(ANGLE)
A3=sin(ANGLE) B3=-cos(ANGLE) C3=1
N140 ENDFOR
```

6.7 Lissage du tracé d'orientation (ORISON, ORISOF)

Fonction

La fonction "Lissage du tracé d'orientation (ORISON)" permet de lisser des fluctuations d'orientation sur plusieurs blocs. On obtient ainsi un tracé lisse aussi bien de l'orientation que du contour.

Condition

La fonction "Lissage du tracé d'orientation (ORISON)" est uniquement disponible dans les systèmes de transformation à 5/6 axes.

Syntaxe

```
ORISON
...
ORISOF
```

Signification

`ORISON` :	Lissage du tracé d'orientation activé	
	Prise d'effet :	modale
`ORISOF` :	Lissage du tracé d'orientation désactivé	
	Prise d'effet :	modale

Données de réglage

Le lissage du tracé d'orientation nécessite la prise en compte :

- d'une tolérance maximale prédéfinie (écart maximum en degré par rapport à l'angle d'orientation de l'outil)

  et

- d'un trajet maximum prédéfini.

Ces présélections sont définies au moyen de données de réglage :

- SD42678 $SC_ORISON_TOL (tolérance pour le lissage du tracé d'orientation)
- SD42680 O$SC_ORISON_DIST (trajet pour le lissage du tracé d'orientation)

Exemple

```
Code de programme                    Commentaire
...
TRAORI()                             ; Activation de la transformation d'orientation.
ORISON                               ; Activation du lissage d'orientation.
$SC_ORISON_TOL=1.0                   ; Tolérance du lissage d'orientation= 1,0 degré.
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                               ; Désactivation du lissage d'orientation.
...
```

Un pivotement de l'orientation de 90 degrés est réalisé dans le plan XZ, de -45 à +45 degrés. En raison du lissage de son tracé, l'orientation n'atteint plus les valeurs angulaires maximales de -45 ou +45 degrés.

Informations complémentaires

Nombre de blocs

Le lissage du tracé d'orientation s'effectue sur un nombre configuré de blocs figurant dans le paramètre machine MD28590 $MC_MM_ORISON_BLOCKS.

Remarque

Si le lissage du tracé d'orientation est activé avec `ORISON`, alors que la mémoire configurée pour des blocs est insuffisante (MD28590 < 4), un message d'alarme est émis et la fonction ne peut plus être exécutée.

Trajet maximum pour le bloc

Le lissage du tracé d'orientation est uniquement réalisé dans des blocs dont le trajet est inférieur à la longueur de bloc maximale configurée (MD20178 $MC_ORISON_BLOCK_PATH_LIMIT). Les blocs possédant des trajets plus longs interrompent le lissage et leur exécution s'effectue selon leur programmation.

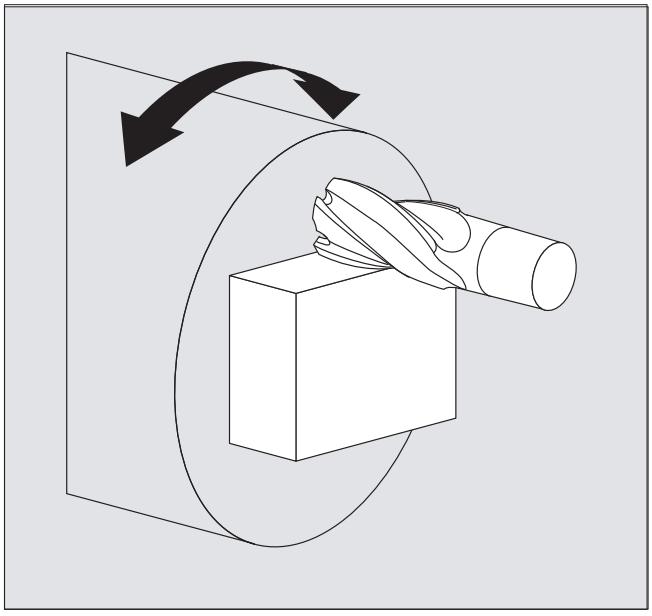
6.8 Transformation cinématique

6.8.1 Opérations de fraisage sur des pièces de tournage (TRANSMIT)

Fonction

La fonction TRANSMIT offre les possibilités suivantes :

- Usinage frontal de pièces de tournage sans démontage de la pièce (trous, contours).
- Pour la programmation de ces opérations, vous pouvez utiliser un système de coordonnées cartésiennes.
- La commande transforme les déplacements programmés dans le système de coordonnées cartésiennes en déplacements des axes machine réels (cas standard) :
  - Axe rotatif
  - Axe de pénétration perpendiculaire à l'axe { de rotation
  - Axe longitudinal parallèle à l'axe rotatif
  - Les axes linéaires sont perpendiculaires les uns par rapport aux autres.
- Un décalage du centre de l'outil par rapport à l'axe de rotation est permis.
- Le pilotage de la vitesse tient compte des limitations définies pour les rotations.



Types de transformation TRANSMIT

Il existe deux types d'usinage TRANSMIT réglables :

- `TRANSMIT` en version standard avec (TRAFO_TYPE_n = 256)
- `TRANSMIT` avec axe linéaire Y supplémentaire (TRAFO_TYPE_n = 257)

Le type de transformation étendu 257 peut être utilisé pour compenser par exemple les corrections de serrage d'un outil avec axe Y réel.

Syntaxe

`TRANSMIT` ou `TRANSMIT(n)`

`TRAFOOF`

Axe rotatif

L'axe rotatif ne peut pas être programmé, étant donné qu'il est affecté comme axe géométrique et qu'il n'est donc pas programmable directement comme axe de canal.

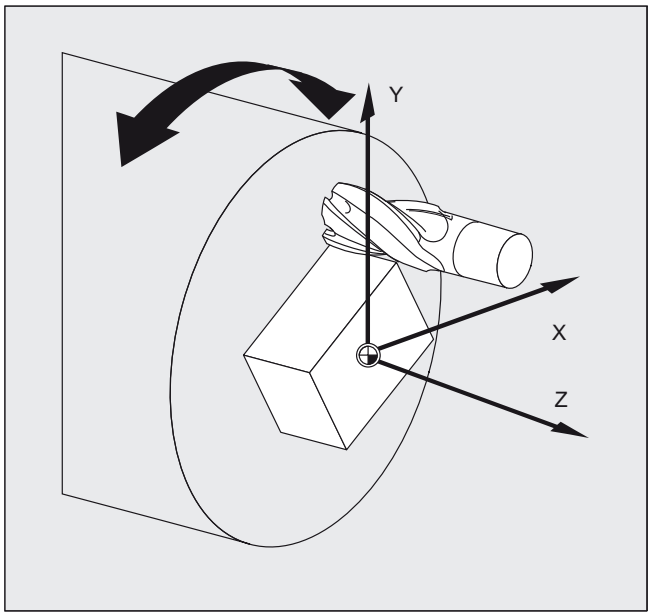
Signification

`TRANSMIT` :	Activation de la première fonction TRANSMIT convenue. Cette fonction est également appelée "transformation polaire".
`TRANSMIT(n)` :	Activation de la nième fonction TRANSMIT convenue, n étant limité à 2 (TRANSMIT(1) correspond à TRANSMIT).
`TRAFOOF` :	Désactivation d'une transformation active
`OFFN` :	Décalage normal au contour : distance séparant l'usinage frontal du contour de référence programmé

Remarque

Une transformation active `TRANSMIT` est également désactivée quand, dans le canal concerné, il y a activation de l'une des autres transformations (par exemple `TRACYL`, `TRAANG`, `TRAORI`).

Exemple



Code de programme	Commentaire
N10 T1 D1 G54 G17 G90 F5000 G94	; Sélection de l'outil
N20 G0 X20 Z10 SPOS=45	; Accostage de la position de départ
N30 TRANSMIT	; Activation de la fonction TRANSMIT
N40 ROT RPL=-45	; Réglage du frame
N50 ATRANS X-2 Y10	
N60 G1 X10 Y-10 G41 OFFN=1OFFN	; Tournage d'ébauche carré ; surépaisseur d'usinage 1 mm
N70 X-10	
N80 Y10	
N90 X10	
N100 Y-10	
N110 G0 Z20 G40 OFFN=0	; Changement d'outil
N120 T2 D1 X15 Y-15	
N130 Z10 G41	
N140 G1 X10 Y-10	; Finition du carré
N150 X-10	
N160 Y10	
N170 X10	
N180 Y-10	
N190 Z20 G40	; Désactiver un frame
N200 TRANS	
N210 TRAFOOF	
N220 G0 X20 Z10 SPOS=45	; Accostage de la position de départ
N230 M30	

Description

Pôle

Il est existe deux possibilités de passer par le pôle :

- Déplacement de l'axe linéaire seul
- Déplacement vers le pôle avec rotation de l'axe rotatif au pôle et sortie du pôle.

La sélection s'effectue à l'aide des PM 24911 et 24951.

TRANSMIT avec axe linéaire Y supplémentaire (type de transformation 257) :

Cette variante de transformation de la transformation polaire utilise la redondance sur une machine avec un autre axe linéaire afin d'exécuter une correction d'outil améliorée. Pour le deuxième axe linéaire, s'applique alors :

- une zone de travail plus réduite et
- l'interdiction d'utiliser le deuxième axe linéaire pour l'exécution du programme pièce.

Le programme pièce et l'affectation des axes correspondants dans le SCB ou le SCM présuppose certains réglages des paramètres machine, voir

Bibliographie

/FB2/ Manuel de fonctions d'extension ; transformations cinématiques (M1)

6.8.2 Transformation de surface latérale de cylindre (TRACYL)

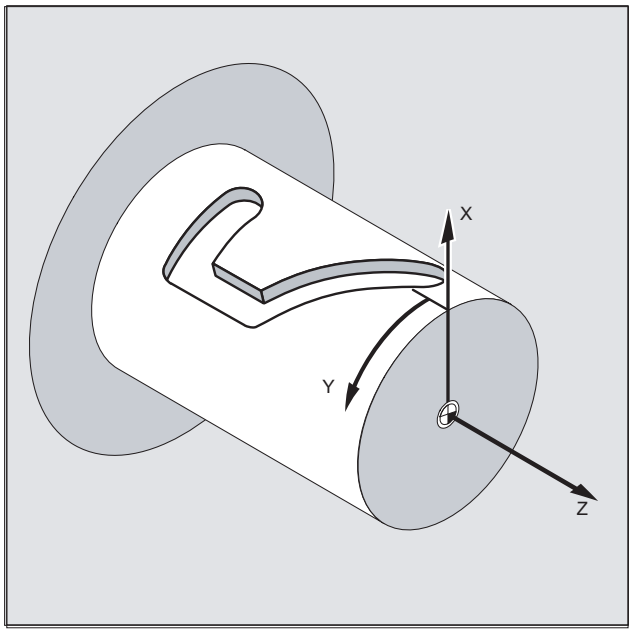
Fonction

La transformation de surface latérale de cylindre TRACYL offre les possibilités suivantes :

Réalisation de :

- rainures longitudinales sur des corps cylindriques,
- rainures transversales sur des corps cylindriques,
- rainures de forme quelconque sur des corps cylindriques.

La forme des rainures est programmée sur la surface développée du cylindre.



Types de transformation TRACYL

Il existe trois types de transformation de surface latérale de cylindre :

- `TRACYL` sans correction des flancs de la rainure : (TRAFO_TYPE_n=512)
- `TRACYL` avec correction des flancs de la rainure : (TRAFO_TYPE_n=513)
- `TRACYL` avec axe linéaire supplémentaire et correction des flancs de la rainure : (TRAFO_TYPE_n=514)
  La correction des flancs de la rainure est paramétrée avec `TRACYL` par le biais du troisième paramètre.

En cas de transformation de surface latérale de cylindre avec correction des flancs de la rainure, l'axe utilisé pour la correction doit se trouver sur zéro (y=0) pour que la rainure puisse être fraisée de façon centrée par rapport à son axe de symétrie, tel qu'il a été programmé.

Utilisation des axes

Les axes suivants ne peuvent pas être utilisés comme axes de positionnement ou d'oscillation :

- l'axe géométrique dans le sens de la circonférence de l'enveloppe du cylindre (axe Y)
- l'axe linéaire supplémentaire en cas de correction des flancs de rainure (axe Z)

Syntaxe

`TRACYL(d)` ou `TRACYL(d, n)` ou

pour le type de transformation 514

```
TRACYL(d, n, correction des flancs de la rainure)
TRAFOOF
```

Axe rotatif

L'axe rotatif ne peut pas être programmé, étant donné qu'il est affecté comme axe géométrique et qu'il n'est donc pas programmable directement comme axe de canal.

Signification

`TRACYL (d)`	Activation de la première fonction TRACYL convenue dans les paramètres machine spécifiques aux canaux. Paramètre d pour le diamètre d'usinage.
`TRACYL (d, n)`	Activation de la nième fonction TRACYL convenue dans les paramètres machine spécifiques aux canaux, n étant limité à 2, TRACYL(d,1) correspond à TRACYL(d).
`D`	Valeur pour le diamètre d'usinage. Le diamètre d'usinage correspond à deux fois la distance entre la pointe de l'outil et le centre de l'axe de rotation. Ce diamètre doit toujours être indiqué et sa valeur doit être supérieure à 1.
`n`	Deuxième paramètre optionnel pour le jeu de données TRACYL 1 (présélectionné) ou 2.
`Correction des flancs de la rainure`	Troisième paramètre optionnel dont la valeur pour TRACYL est présélectionnée à partir du mode des paramètres machine. Plage de valeurs : 0 : Type de transformation 514 traditionnelle sans correction des flancs de rainure 1 : Type de transformation 514 avec correction des flancs de la rainure
`TRAFOOF`	Désactivation de la transformation (SCB et SCM sont à nouveau identiques).
`OFFN`	Décalage normal au contour : Distance séparant le flanc de rainure du contour de référence programmé.

Remarque

Une transformation active `TRACYL` est également désactivée quand, dans le canal concerné, il y a activation de l'une des autres transformations (par exemple `TRANSMIT`, `TRAANG`, `TRAORI`).

Exemple : Définition de l'outil

L'exemple suivant permet de tester le paramétrage de la transformation de cylindre `TRACYL` :

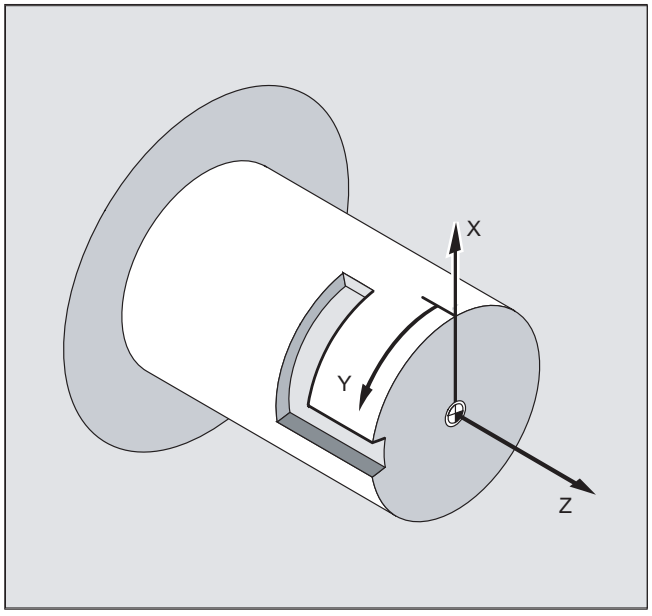
Code de programme	**Commentaire**	
`Paramètres d'outil`	`Signification`	`Remarque`
`Numéro de paramètre (DP)`		
`$TC_DP1[1,1]=120`	`Type d'outil`	`Fraise`
`$TC_DP2[1,1] = 0`	`Position du tranchant`	`Uniquement outils de tournage`

Code de programme	**Commentaire**	
`Géométrie`	`Correction de longueur`	
`$TC_DP3[1,1]=8.`	`Vecteur de correction de longueur`	`Prise en compte selon type`
`$TC_DP4[1,1]=9.`		`et plan`
`$TC_DP5[1,1]=7.`		

Code de programme	Commentaire	
Géométrie	Rayon	
$TC_DP6[1,1]=6.	Rayon	Rayon de l'outil
$TC_DP7[1,1]=0	Largeur de rainure b (scie à rainurer), rayon d'arrondi (outils de fraisage)	
$TC_DP8[1,1]=0	Dépassement k	Uniquement scie à rainurer
$TC_DP9[1,1]=0		
$TC_DP10[1,1]=0		
$TC_DP11[1,1]=0	Angle pour outils de fraisage coniques	

Code de programme	Commentaire	
Usure	Correction de longueur et de rayon	
$TC_DP12[1,1]=0	Les paramètres restants jusqu'à $TC_DP24=0	Cote de base/adaptateur

Exemple : Fraisage d'une rainure à angle droit



Activation de la fonction de transformation de surfaces cylindriques :

Code de programme	Commentaire
N10 T1 D1 G54 G90 F5000 G94	; Sélection de l'outil, compensation de serrage
N20 SPOS=0	; Accostage de la position de départ
N30 G0 X25 Y0 Z105 CC=200	
N40 TRACYL (40)	; Activation de la transformation de surface latérale de cylindre
N50 G19	; Sélection du plan

Usinage d'une rainure en forme de crochet :

Code de programme	Commentaire
N60 G1 X20	; Pénétration de l'outil suivant la profondeur de la rainure
N70 OFFN=12	; Définition d'une distance de 12 mm entre le flanc de rainure et l'axe de rainure
N80 G1 Z100 G42	; Accostage du flanc droit de la rainure
N90 G1 Z50	; Section de rainure parallèle à l'axe du cylindre
N100 G1 Y10	; Section de rainure parallèle à la circonférence du cylindre
N110 OFFN=4 G42	; Accostage du flanc gauche de la rainure ; définition d'une distance de 4 mm entre le flanc et l'axe de rainure
N120 G1 Y70	; Section de rainure parallèle à la circonférence du cylindre
N130 G1 Z100	; Section de rainure parallèle à l'axe du cylindre
N140 G1 Z105 G40	; Retrait du flanc de rainure
N150 G1 X25	; Dégagement
N160 TRAFOOF	
N170 G0 X25 Y0 Z105 CC=200	; Accostage de la position de départ
N180 M30	

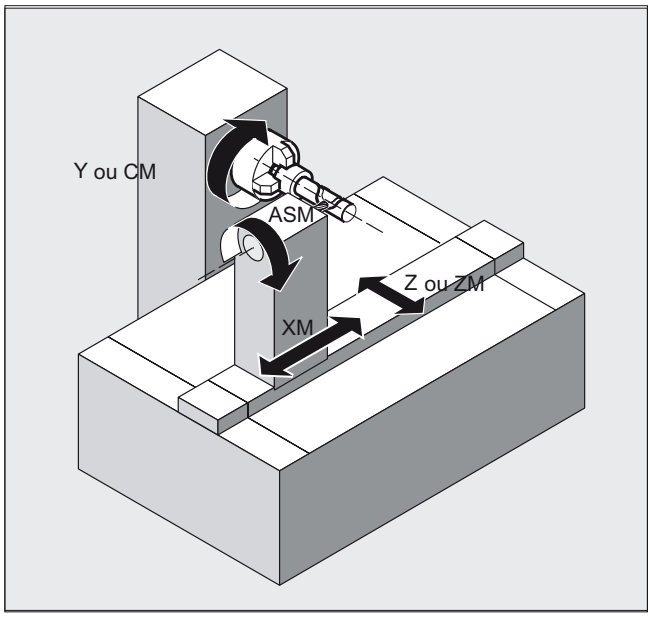
Description

Sans correction du flanc de rainure (type de transformation 512) :

La commande transforme les déplacements programmés dans le système de coordonnées cylindre en déplacements des axes machine réels :

- Axe rotatif
- Axe de pénétration perpendiculaire à l'axe { de rotation
- Axe longitudinal parallèle à l'axe rotatif

Les axes linéaires sont perpendiculaires les uns par rapport aux autres. L'axe de pénétration coupe l'axe rotatif.

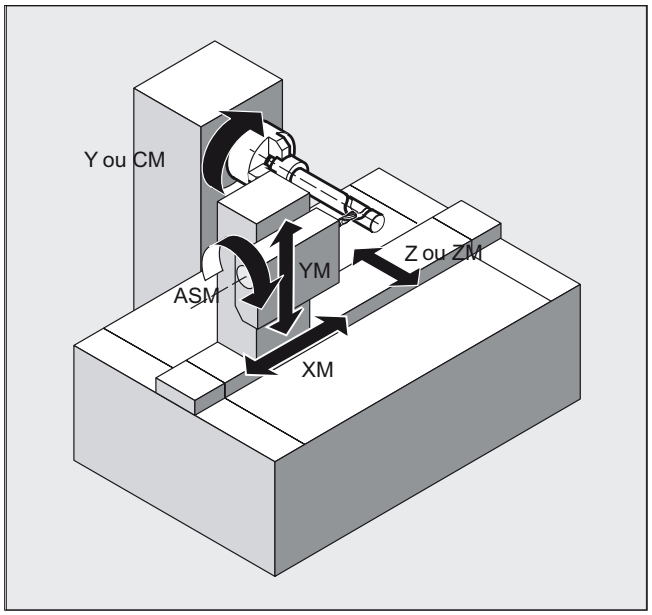


Avec correction du flanc de rainure (type de transformation 513) :

Même cinématique que ci-dessus, mais en plus axe longitudinal parallèle au sens de la circonférence

Les axes linéaires sont perpendiculaires les uns par rapport aux autres.

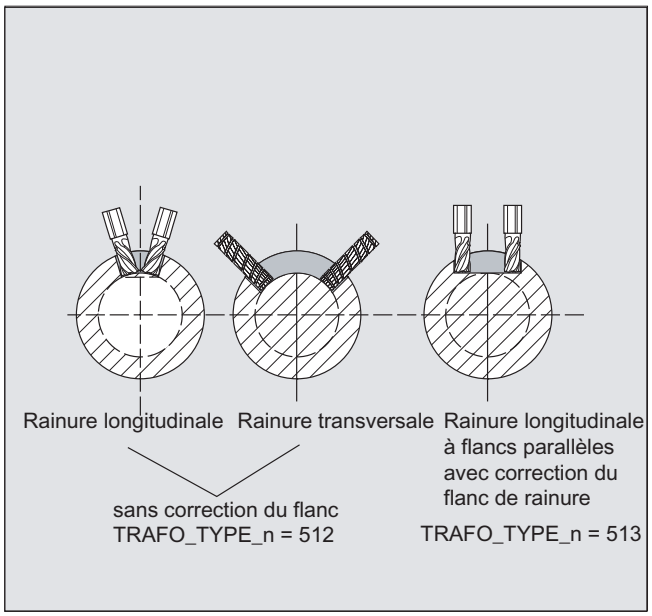
Le pilotage de la vitesse tient compte des limitations définies pour les rotations.



Section de la rainure

Sur la configuration d'axe 1, les rainures le long de l'axe rotatif ne sont parallèles que lorsque la largeur de la rainure correspond exactement au rayon de l'outil.

Les rainures parallèles à la circonférence (rainures transversales) ne sont pas parallèles au début et à la fin.



Avec axe linéaire supplémentaire et avec correction du flanc de rainure (type de transformation 514) :

Cette variante de transformation utilise la redondance sur une machine avec un autre axe linéaire afin d'exécuter une correction d'outil améliorée. Pour le deuxième axe linéaire, s'applique alors :

- une zone de travail plus petite et
- l'interdiction d'utiliser le deuxième axe linéaire pour l'exécution du programme pièce.

Pour le programme pièce et l'affectation des axes correspondants dans le SCB ou le SCM, certains paramètres machine sont remis à zéro. Voir

Bibliographie

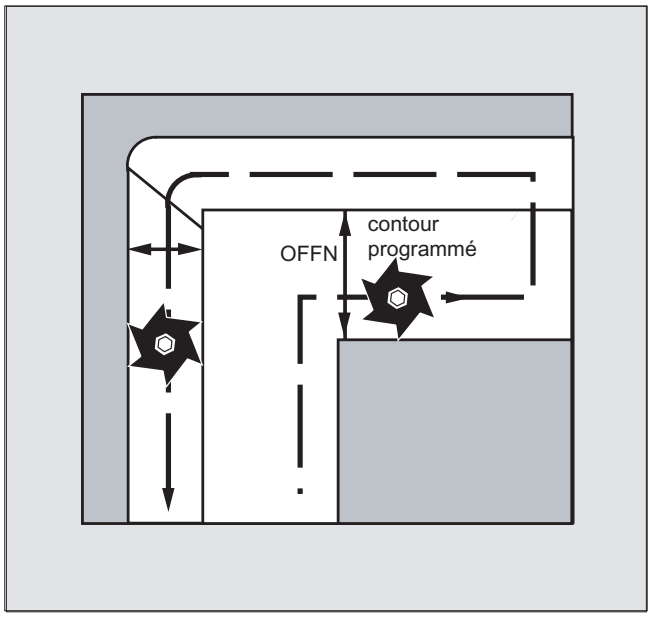
/FB2/ Manuel de fonctions d'extension ; transformations cinématiques (M1)

Décalage normal au contour OFFN (type de transformation 513)

Pour le fraisage d'une rainure avec `TRACYL` , il faut,

- programmer la ligne médiane de la rainure
- et `la demi-largeur de rainure par le biais de` OFFN dans le programme pièce.

`OFFN` n'est activé que lorsque la correction de rayon d'outil a été sélectionnée, afin d'éviter toute détérioration du flanc de rainure). **En outre, OFFN doit être supérieur ou égal au rayon d'outil afin d'empêcher toute détérioration du flanc de rainure opposé.**



Un programme pièce pour le fraisage d'une rainure comprend généralement les étapes suivantes :

1. Sélection de l'outil
2. `Sélection` de TRACYL
3. Sélection du décalage du système de coordonnées approprié (FRAME)
4. Positionnement
5. `Programmation` d'OFFN
6. Sélection de la CRO (correction de rayon d'outil)
7. Bloc d'accostage (CRO et accostage du flanc de la rainure)
8. Contour correspondant à l'axe de symétrie de la rainure
9. Désactivation de la CRO
10. Bloc de retrait (retrait de la CRO et éloignement du flanc de rainure)
11. Positionnement
12. `TRAFOOF`
13. Reprise du décalage de coordonnées initial (FRAME)

Particularités

- Sélection de la CRO :

  La programmation de la CRO ne s'effectue pas par rapport au flanc de rainure, mais par rapport à l'axe de symétrie de rainure programmé. Pour que l'outil se déplace à gauche du flanc de rainure, il convient d'introduire G42 (au lieu de G41). Vous évitez cela en indiquant une largeur de rainure négative avec `OFFN`.

- `OFFN` avec `TRACYL` a un comportement différent de OFFN sans `TRACY` . Comme `OFFN` sans `TRACYL` est également pris en considération si la CRO est activée, il est recommandé d'annuler à nouveau `OFFN` après `TRAFOOF`.
- Il est possible de modifier `OFFN` à l'intérieur du programme pièce. L'axe de symétrie d'une rainure peut donc être désaxé (voir figure).
- Rainures de guidage :

  `TRACYL` ne permet pas de créer la même rainure de guidage que celle fabriquée avec un outil dont le diamètre présente la largeur de rainure. En principe, il n'est pas possible de générer la même géométrie de flanc de rainure avec un outil cylindrique de petite taille et avec un outil plus grand. `TRACYL` minimise l'erreur. Pour éviter des problèmes de précision, il est recommandé d'utiliser un outil dont le rayon n'est que légèrement inférieur à la moitié de la largeur de la rainure.

Remarque

OFFN et CRO

Si TRAFO_TYPE_n = 512, la valeur sous OFFN agit en addition à la CRO.

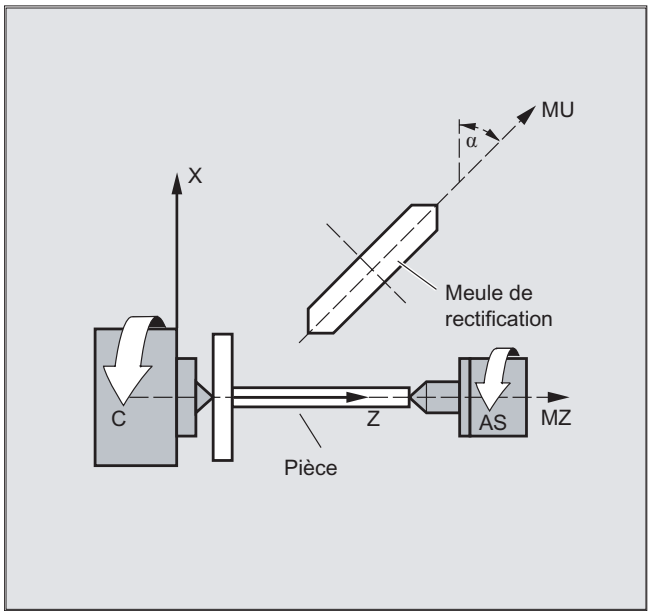
Si TRAFO_TYPE_n = 513, la valeur sous `OFFN` correspond à la demi-largeur de la rainure. Le contour est parcouru avec OFFN-CRO.

6.8.3 Axe oblique (TRAANG)

Fonction

La fonction axe oblique est destinée à l'utilisation en rectification et offre les possibilités suivantes :

- Usinage avec axe de pénétration oblique
- Un système de coordonnées cartésiennes peut être utilisé pour la programmation.
- La commande transforme les déplacements programmés dans le système de coordonnées cartésiennes en déplacements des axes machine réels (cas standard) : axe de pénétration oblique.



Syntaxe

```
TRAANG(α) ou TRAANG(α, n)
TRAFOOF
```

Signification

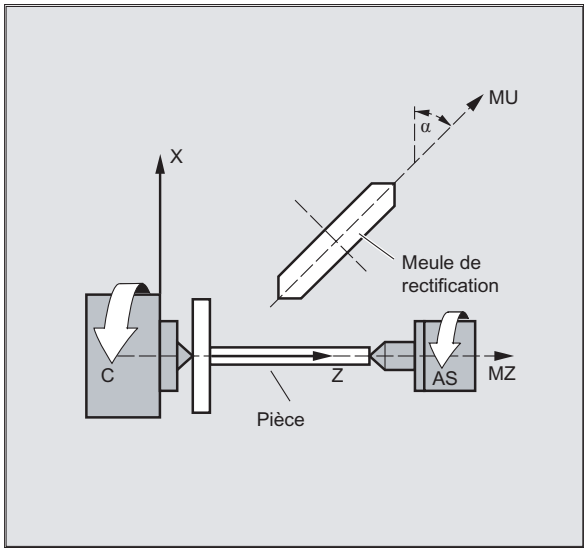
`TRAANG( ) ou TRAANG( ,n)`	Activer la transformation avec le paramétrage de la sélection précédente.
`TRAANG(α)`	Activation de la première transformation convenue d'axe oblique
`TRAANG(α,n)`	Activation de la nième transformation convenue de l'axe oblique, n étant limité à 2. TRAANG(α,1) correspond à TRAANG(α).
`αA`	Angle de l'axe oblique Les valeurs autorisées pour α sont les suivantes : -90 degrés < α < + 90 degrés
`TRAFOOF`	Désactivation de la transformation
`n`	Nombre de transformations convenues

Laisser de côté l'angle α ou zéro

Si l'angle α est laissé de côté (par exemple `TRAANG()`, `TRAANG(, n)`), la transformation est activée avec le paramétrage de la sélection précédente. Lors de la première sélection, la valeur prédéfinie applicable est celle des paramètres machine.

Un angle α = 0 (par exemple `TRAANG(0)`, `TRAANG(0,n)`) constitue un paramétrage définitif et ne correspond plus à la suppression du paramètre dans les versions antérieures.

Exemple



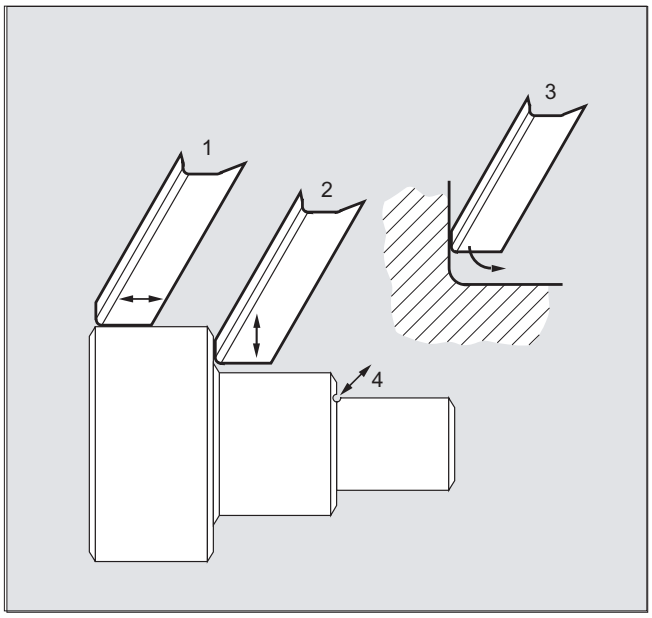
Code de programme	Commentaire
N10 G0 G90 Z0 MU=10 G54 F5000 ->	; Sélection de l'outil, compensation de serrage,
-> G18 G64 T1 D1	Sélection du plan
N20 TRAANG(45)	; Activation de la transformation axe oblique
N30 G0 Z10 X5	; Accostage de la position de départ
N40 WAITP(Z)	; Libération des axes pour l'oscillation
N50 OSP[Z]=10 OSP2[Z]=5 OST1[Z]=-2 ->	; Oscillation jusqu'à la cote finale
-> OST2[Z]=-2 FA[Z]=5000	(Oscillation : voir chapitre "Oscillation")
N60 OS[Z]=1	
N70 POS[X]=4.5 FA[X]=50	
N80 OS[Z]=0	
N90 WAITP(Z)	; Libération des axes d'oscillation en tant qu'axes de positionnement
N100 TRAFOOF	; Désactiver la transformation
N110 G0 Z10 MU=10	; Dégagement
N120 M30	;

-> doit être programmé dans un bloc

Description

Les usinages suivants sont possibles :

1. Cylindrage
2. Dressage
3. Profilage d'un contour donné
4. Rectification en plongée oblique



Constructeur de la machine-outil

Les réglages suivants sont définis par les paramètres machine :

- l'angle entre un axe machine et l'axe oblique,
- la position de l'origine pièce par rapport à l'origine du système de coordonnées convenu par la fonction "axe oblique",
- la réserve de vitesse à disposition sur l'axe parallèle pour le déplacement compensateur,
- la réserve d'accélération à disposition sur l'axe parallèle pour le déplacement compensateur.

Configuration d'axes

Pour pouvoir programmer dans un système de coordonnées cartésiennes, le rapport entre ce système de coordonnées et les axes machine effectifs (MU, MZ) doit être communiqué à la commande :

- Dénomination des axes géométriques
- Affectation des axes géométriques aux axes de canal
  - cas général (axe oblique inactif)
  - axe oblique actif
- Affectation des axes de canal aux numéros d'axe machine
- Identification des broches
- Attribution de noms aux axes machine

A l'exception de "axe oblique actif", la manière de procéder est identique à celle utilisée pour la configuration d'axes normale.

6.8.4 Programmation d'un axe oblique (G05, G07)

Fonction

En mode manuel, la meule peut être déplacé au choix dans le sens cartésien ou dans le sens des axes obliques (l'affichage reste cartésien). Seul l'axe U {réel se déplace, l'affichage de l'axe Z{est actualisé.

Les décalages REPOS doivent retourner en mode manuel dans le sens cartésien.

En "Déplacement PTP", le dépassement de la limitation cartésienne de la zone de travail est surveillé en mode manuel, l'axe correspondant est freiné auparavant. Si le "déplacement PTP" n'est pas actif, l'axe peut être déplacé avec précision jusqu'à la limitation de la zone de travail.

Bibliographie

/FB2/ Manuel de fonctions d'extension ; transformation cinématique (M1)

Syntaxe

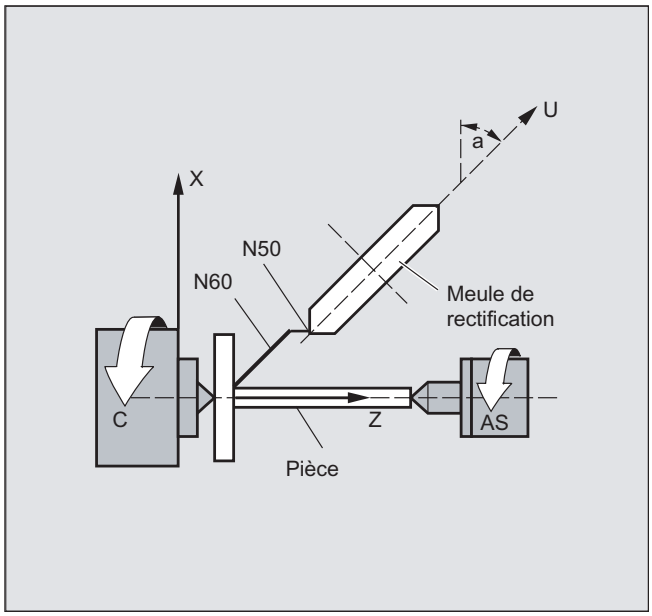
```
G07
G05
```

Les instructions `G07/G05` servent à faciliter la programmation des axes obliques. Des positions peuvent être programmées et affichées dans un système de coordonnées cartésiennes. La correction d'outil et le décalage d'origine seront calculés dans le système cartésien. Après la programmation de l'angle pour l'axe oblique dans le programme CN, la position de départ peut être accostée (`G07`) puis la plongée oblique (`G05`) peut être engagée.

Signification

G07	Accostage de la position de départ
G05	Activation plongée oblique

Exemple



Programmation	Commentaire
N.. G18	; Programmation de l'angle pour l'axe oblique
N50 G07 X70 Z40 F4000	; Accostage de la position de départ
N60 G05 X70 F100	; Plongée oblique
N70 ...	;

6.9 Déplacement PTP cartésien

Fonction

Avec cette fonction, vous pouvez programmer une position dans un système de coordonnées cartésiennes, bien que le déplacement des axes machine s'effectue dans le système de coordonnées machine. La fonction peut être utilisée, par exemple, pour modifier la position d'articulations, si cela entraîne le passage par un point singulier.

Remarque

Cette fonction n'a d'intérêt que si une transformation est activée. En outre, le "déplacement PTP" n'est admissible qu'en liaison avec G0 et G1.

Syntaxe

```
N... TRAORI
N... STAT='B10' TU='B100' PTP
N... CP
```

Déplacement PTP avec une transformation générique cinq/six axes

Lorsque, lors d'une transformation générique 5 / 6 axes avec PTP, un déplacement point-à-point est activé dans le système de coordonnées machine `(ORIMKS)`, alors l'orientation de l'outil peut aussi bien être programmée avec des positions d'axes rotatifs

```
N... G1 X Y Z A B C
```

qu'avec des vecteurs d'Euler ou des angles RPY indépendants de la cinématique

```
N... ORIEULER ou ORIRPY
N... G1 X Y Z A2 B2 C2
```

ou des vecteurs de direction

```
N... G1 X Y Z A3 B3 C3
```

. Une interpolation d'axe rotatif tout comme une interpolation vectorielle avec une interpolation circulaire de grand rayon `ORIVECT` ou une interpolation du vecteur d'orientation sur une enveloppe de cône `ORICONxx` peuvent être parallèlement actives.

Ambiguïtés de l'orientation avec des vecteurs

Lors de la programmation de l'orientation avec des vecteurs, il existe une ambiguïté dans les positions possibles de l'axe rotatif. Les positions d'axe rotatif à accoster peuvent être sélectionnées avec la programmation de `STAT = <...>`. Lorsque

`STAT = 0` est programmé (cela correspond au paramétrage par défaut),
les positions qui ont la distance la plus faible aux positions de départ sont accostées.
Lorsque

`STAT = 1` est programmé,
les positions qui ont la distance la plus importante aux positions de départ sont accostées.

Signification

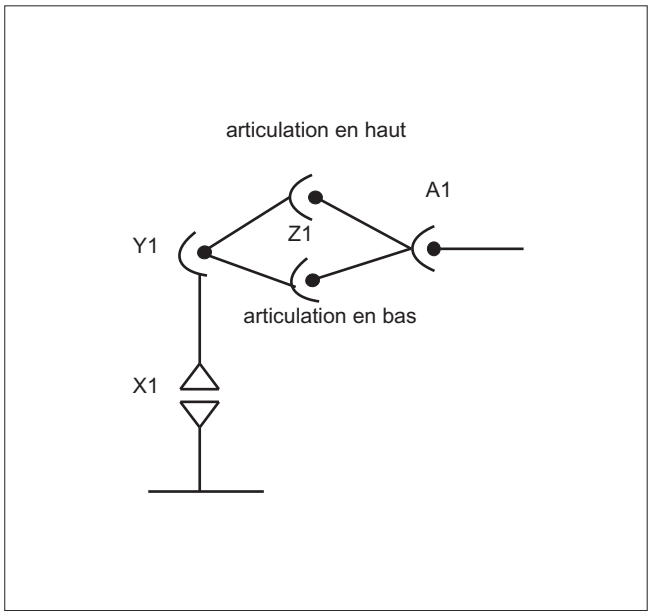
Les commandes `PTP` et `CP` sont des fonctions modales. `CP` est le paramétrage par défaut.

Alors que la programmation de la valeur STAT est à effet modal, la programmation de TU = <...> agit bloc par bloc.

Une autre différence est que la programmation d'une valeur STAT n'a d'effet qu'avec une interpolation vectorielle tandis que la programmation de TU est également évaluée en cas d'interpolation active d'axe rotatif.

`PTP`	**p**oint **t**o **p**oint (déplacement point à point) Le déplacement est effectué en tant que déplacement d'axe synchrone ; l'axe le plus lent parmi les axes participant au déplacement détermine la vitesse.
`CP`	**c**ontinuous **p**ath (déplacement avec interpolation) Le déplacement est effectué sous forme de déplacement cartésien avec interpolation.
`STAT=`	Position des articulations ; valeur en fonction de la transformation.
`TU=`	L'information TURN a un effet non modal. Permet d'accoster de façon univoque des angles d'axe compris entre -360 degrés et +360 degrés.

Exemple



`N10 G0 X0 Y-30 Z60 A-30 F10000`	Position de départ → articulation en haut
`N20 TRAORI(1)`	Activation de la transformation
`N30 X1000 Y0 Z400 A0`	
`N40 X1000 Z500 A0 STAT='B10' TU='B100' PTP`	Changement d'orientation sans transformation → articulation en bas

```
N50 X1200 Z400 CP                    Réactivation de la transformation
N60 X1000 Z500 A20
N70 M30
```

Exemple de déplacement PTP avec une transformation générique cinq axes

Hypothèse : Une cinématique orthogonale CA sert de base.

Code de programme	Commentaire
TRAORI	; Transformation de cinématique CA activée
PTP	; Activer le déplacement PTP
N10 A3 = 0 B3 = 0 C3 = 1	; Positions d'axe rotatifs C = 0 A = 0
N20 A3 = 1 B3 = 0 C3 = 1	; Positions d'axe rotatifs C = 90 A = 45
N30 A3 = 1 B3 = 0 C3 = 0	; Positions d'axe rotatifs C = 90 A = 90
N40 A3 = 1 B3 = 0 C3 = 1 STAT = 1	; Positions d'axe rotatifs C = 270 A = –45

Sélectionner la position d'accostage univoque de l'axe rotatif :

Dans le bloc N40, les axes rotatifs parcourent, du fait de la programmation `STAT = 1`, la trajectoire la plus longue de leur point de départ (C=90, A=90) à leur point final (C=270, A=–45) au lieu de suivre la trajectoire la plus courte jusqu'au point final (C=90, A=45) comme ce serait le cas avec `STAT = 0`.

Description

La commutation du déplacement cartésien au déplacement des axes machine a lieu à l'aide des instructions `PTP` et `CP`.

Déplacement PTP avec une transformation générique cinq/six axes

Lors du déplacement PTP, le TCP ne reste généralement pas stationnaire, contrairement à la transformation 5/6 axes, au cas où seule l'orientation est modifiée. Les positions finales transformées de tous les axes de transformation (axes linéaires et jusqu'à 3 axes rotatifs) sont accostées linéairement sans la transformation soit encore active au sens propre.

Le déplacement PTP est désactivé par la programmation du code G modal `CP`.

Les différentes transformations sont décrites dans l'imprimé :
/FB3/ Manuel de fonctions spéciales ; Pack de transformation Manipulation (TE4).

Programmation de la position des articulations (STAT=)

L'indication de la position en coordonnées cartésiennes et de l'orientation de l'outil ne suffit pas pour déterminer de façon univoque une position de la machine. Selon le type de cinématique, 8 positions distinctes d'articulation sont possibles. Ces positions sont donc spécifiques à une transformation. Afin de pouvoir convertir de façon univoque une position cartésienne en angles d'axe, la position des articulations doit être indiquée avec l'instruction `STAT=` . L'instruction "`STAT`" comporte un bit pour chacune des positions possibles.

Pour plus d'informations sur les bits de position à programmer pour "`STAT`", voir :
/FB2/ Manuel de fonctions d'extension ; transformation cinématique (M1), chapitre "Déplacement PTP cartésien".

Programmation des angles d'axe (TU=)

Pour accoster de façon univoque des angles d'axe < ±360 degrés, l'instruction "TU=" doit être ajoutée à l'information d'angle.

Les axes effectuent le déplacement minimal :

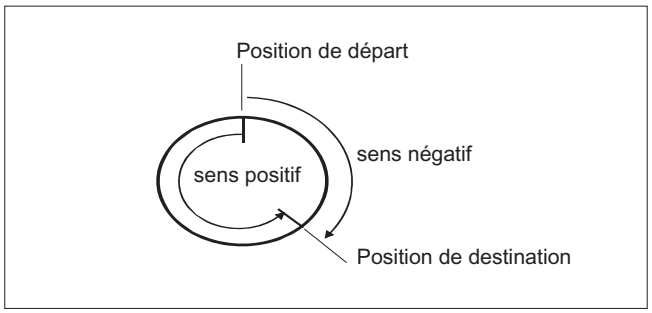
- si TU n'est pas programmée pour une position,
- si l'axe possède une plage de déplacement > ±360 degrés.

Exemple :

La position de destination représentée sur la figure ci-contre peut être accostée en sens négatif ou positif. Le sens est programmé sous l'adresse A1.

A1=225°, TU=bit 0, → sens positif

A1=−135°, TU=bit 1, → sens négatif



Exemple d'évaluation de TU pour transformation générique 5/6 axes et positions de destination

La variable TU contient, pour chaque axe qui entre dans la transformation, un bit indiquant la direction du déplacement. L'affectation des bits TU correspond à la vue de l'axe des canaux rotatifs. L'information TU est uniquement évaluée pour les 3 axes rotatifs maximum qui entrent dans la transformation :

Bit0 : Axe 1, bit TU = 0 : 0 degré <= angle d'axe rotatif < 360 degrés

Bit1: Axe 2, bit TU = 1 : –360 degrés < angle d'axe rotatif < 0 degré

La position de départ d'un axe rotatif est C = 0, avec la programmation de C = 270, l'axe rotatif se déplace sur les positions de destination suivantes :

C = 270: bit TU 0, sens de rotation positif

C = –90: bit TU 1, sens de rotation négatif

Autre comportement

Changement de mode de fonctionnement

La fonction "Déplacement PTP cartésien" n'a d'intérêt que dans les modes de fonctionnement AUTO et MDA. En cas de passage au mode JOG, le réglage courant est conservé.

Si le code G `PTP` est réglé, les axes sont déplacés dans le SCM. Si le code G `CP` est réglé, les axes sont déplacés dans le SCP.

Power On/RESET

Après Power On ou après RESET, le paramétrage dépend du paramètre machine `$MC_GCODE_REST_VALUES[48]`. Le mode de déplacement "`CP`" est préréglé.

REPOS

Si la fonction "Déplacement PTP cartésien" était réglée pendant le bloc interrompu, le repositionnement s'effectue également avec `PTP` .

Corrections de déplacement

Le décalage DRF ou le décalage d'origine externe ne sont possibles qu'avec des restrictions en déplacement PTP cartésien. Aucune correction ne doit être active dans le SCB lors du passage du déplacement PTP au déplacement CP.

Arrondissement entre des déplacements CP et PTP

Vous pouvez programmer un arrondissement de la transition entre des blocs CP et PTP avec `G641` .

La taille de la zone arrondie est le trajet en mm ou inch parcouru sur la trajectoire entre le début et la fin de l'arrondissement. Cette taille doit être indiquée de la manière suivante :

- pour les blocs G0, avec `ADISPOS`
- pour toutes les autres instructions de déplacement, avec `ADIS`

Le calcul du trajet tient compte des adresses F dans le cas des blocs qui ne comportent pas G0. L'avance est celle des axes indiqués dans l'instruction `FGROUP(...)` .

Calcul de l'avance

Dans le cas des blocs CP, les axes cartésiens du système de coordonnées de base sont utilisés pour le calcul.

Dans le cas des blocs PTP, les axes correspondants du système de coordonnées machine sont utilisés pour le calcul.

6.9.1 PTP avec TRANSMIT

Fonction

PTP avec TRANSMIT permet l'accostage à l'aide des blocs G0 et G1 avec optimisation du temps. Au lieu de déplacer les axes du système de coordonnées de base de manière linéaire (CP), ce sont les axes machine qui sont déplacés de manière linéaire (PTP). Ainsi, le comportement du déplacement des axes machine au voisinage du pôle permet d'atteindre le point final de bloc beaucoup plus rapidement.

Le programme pièce continue à être écrit dans le système de coordonnées cartésiennes et tous les décalages de coordonnées, toutes les rotations et programmations de frames restent valables. La simulation sur IHM est également affichée dans le système de coordonnées cartésiennes de la pièce.

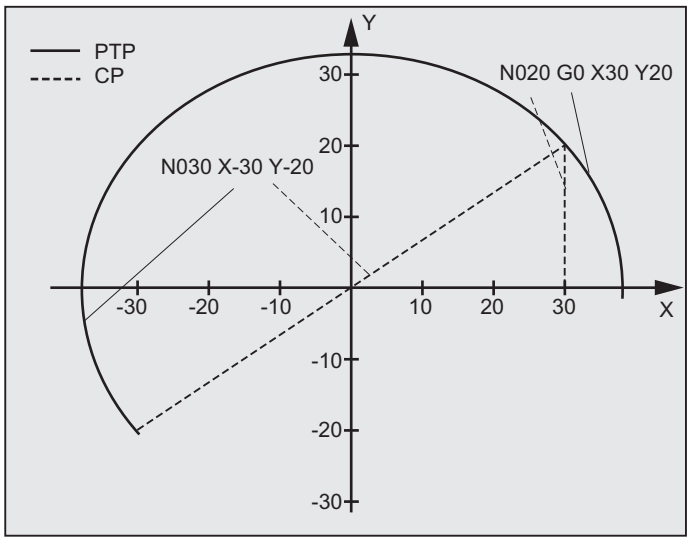
Syntaxe

```
N... TRANSMIT
N... PTPG0
N... G0 ...
...
N... G1 ...
```

Signification

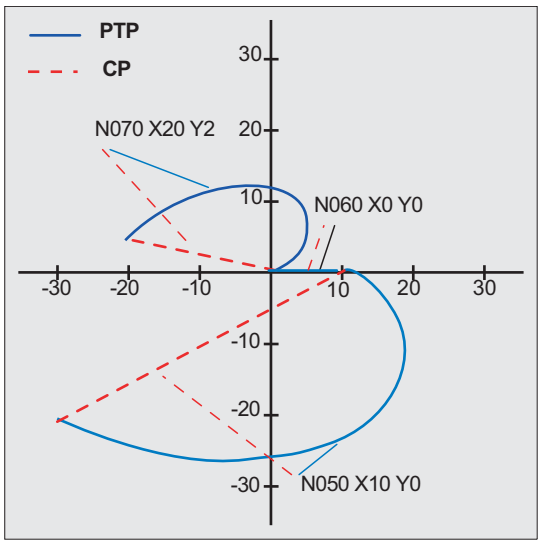
`TRANSMIT`	Activation de la première fonction TRANSMIT convenue (voir chapitre "Opérations de fraisage sur des pièces de tournage : TRANSMIT")
`PTPG0`	**P**oint **t**o **P**oint **G0** (activer automatiquement le déplacement point à point sur chaque bloc G0 puis rebasculer en mode CP) Etant donné que STAT et TU sont à effet modal, la valeur programmée en dernier est toujours valide.
`PTP`	**p**oint **t**o **p**oint (déplacement point à point) Pour TRANSMIT, PTP signifie que dans le système de coordonnées cartésiennes, le déplacement s'effectue sur des spirales d'Archimède soit autour du pôle, soit à partir de celui-ci. Les déplacements d'outil résultants varient sensiblement de ceux en mode CP et sont représentés dans les exemples de programmation concernés.
`STAT=`	Résolution de l'ambiguïté en ce qui concerne le pôle.
`TU=`	TU n'est pas pertinent dans le cas de PTP avec TRANSMIT

Exemple de contournement du pôle avec PTP et TRANSMIT



Code de programme	Commentaire
N001 G0 X30 Z0 F10000 T1 D1 G90	; Position de départ Cote absolue
N002 SPOS=0	
N003 TRANSMIT	; Transformation TRANSMIT
N010 PTPG0	; PTP automatiquement pour chaque bloc G0 puis à nouveau CP
N020 G0 X30 Y20	
N030 X-30 Y-20	
N120 G1 X30 Y20	
N110 X30 Y0	
M30	

Exemple de dégagement du pôle avec PTP et TRANSMIT



Programmation	Commentaire
N001 G0 X90 Z0 F10000 T1 D1 G90	; Position de départ
N002 SPOS=0	
N003 TRANSMIT	; Transformation TRANSMIT
N010 PTPG0	; PTP automatiquement pour chaque bloc G0 puis à nouveau 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	

Description

PTP et PTPG0

PTPG0 est pris en compte pour toutes les transformations qui peuvent exécuter PTP. Dans tous les autres cas, PTPG0 n'est pas pertinent.

Les blocs G0 sont exécutés en mode CP.

La sélection de PTP ou PTPG0 s'effectue dans le programme pièce ou en désélectionnant CP dans le paramètre machine $MC_GCODE_RESET_VALUES[48].

⚠ PRUDENCE

Conditions marginales

En ce qui concerne les déplacements d'outil et les collisions, plusieurs conditions marginales et certaines exclusions de fonctions s'appliquent:

Avec PTP , aucune correction de rayon d'outil (CRO) ne doit être active.

Avec PTPG0, en cas de correction de rayon d'outil (CRO) active, l'accostage se fait par CP.

Avec PTP, l'accostage et retrait en douceur (WAB) n'est pas possible.

Avec PTPG0, en cas d'accostage et retrait en douceur (WAB), l'accostage se fait par CP.

Avec PTP, les cycles de chariotage (CONTPRON, CONTDCON) ne sont pas possibles.

Avec PTPG0, l'accostage dans les cycles de chariotage (CONTPRON, CONTDCON) se fait par CP.

Le chanfreinage (CHF, CHR) et l'arrondissage (RND, RNDM) sont ignorés.

Le compacteur n'est pas compatible avecPTP et il est automatiquement désélectionné dans les blocs PTP.

Une correction d'axe dans l'interpolation ne doit pas être modifiée au cours de l'étape PTP.

Avec G643, le basculement s'effectue automatiquement vers les transitions entre blocs selon précision axiale G642 .

Lorsque PTP est actif, les axes de la transformation ne peuvent pas être simultanément des axes de positionnement.

Bibliographie :

/FB2/ Manuel de fonctions d'extension ; transformation cinématique (M1), chapitre "Déplacement PTP cartésien".

PTP avec TRACON :

PTP peut également être utilisé avec TRACON si la première transformation concaténée prend en charge PTP.

Signification de STAT= et TU= avec TRANSMIT

Si l'axe rotatif pivote de 180 degrés ou si le contour franchit le pôle avec CP, les axes rotatifs peuvent pivoter en fonction du paramètre machine $MC_TRANSMIT_POLE_SIDE_FIX_1/2 [48] de -/+ 180 degrés et être déplacés dans le sens horaire ou antihoraire. Il est possible de définir en outre si l'accostage doit s'effectuer par le pôle ou si la rotation doit s'effectuer autour du pôle.

6.10 Conditions marginales pour l'activation d'une transformation

Fonction

La sélection de transformations est possible par le biais du programme pièce ou en mode MDA. Attention :

- Aucun bloc intermédiaire de déplacement n'est inséré (chanfreins/rayons).
- Une séquence de blocs de type spline doit être terminée, sinon un message s'affiche.
- La correction fine d'outil doit être désactivée (FTOCOF), sinon un message s'affiche.
- La correction de rayon d'outil doit être désactivée (G40), sinon un message s'affiche.
- Une correction de longueur d'outil activée est reprise dans la transformation par la commande.
- Le frame courant activé avant la transformation est désactivé par la commande.
- Une limitation active de la zone de travail est désactivée par la commande pour les axes impliqués dans la transformation (correspond à WALIMOF).
- La surveillance de la zone de protection est désactivée.
- Le contournage et l'arrondissement sont interrompus.
- Tous les axes indiqués dans les paramètres machine doivent être synchronisés au niveau des blocs.
- Les axes permutés sont rétablis, sinon un message apparaît.
- Un message apparaît en présence d'axes dépendants.

Changement d'outil

Un changement d'outil n'est autorisé que si la correction de rayon d'outil est désactivée.

Un changement de la correction de longueur d'outil et une activation/ désactivation de la correction de rayon d'outil ne doivent pas être programmés dans le même bloc.

Changement de frame

Toutes les instructions qui se rapportent uniquement au système de coordonnées de base sont autorisées (FRAME, correction de rayon d'outil). Un changement de frame avec G91 (cotes relatives) ne sera cependant pas traité à part, contrairement à ce qui se passe lorsque la transformation est inactive. L'incrément à parcourir est traité dans le système de coordonnées pièce du nouveau frame, quel que soit le frame qui était actif dans le bloc précédent.

Exclusions

Les axes impliqués dans une transformation ne peuvent pas être utilisés :

- comme axes Preset (alarme),
- pour l'accostage de point fixe (alarme),
- pour la prise de référence (alarme).

6.11 Désactivation d'une transformation (TRAFOOF)

Fonction

L'instruction `TRAFOOF` permet de désactiver l'ensemble des transformations et frames actifs.

Remarque

Les frames nécessaires doivent ensuite être activés par une nouvelle programmation.

Attention :

Les conditions marginales pour l'activation d'une transformation (voir chapitre "Conditions marginales pour l'activation d'une transformation") s'appliquent également à la désactivation d'une transformation.

Syntaxe

```
TRAFOOF
```

Signification

`TRAFOOF`	Instruction de désactivation de l'ensemble des transformations/frames actifs

6.12 Transformations concaténées (TRACON, TRAFOOF)

Fonction

Deux transformations peuvent être activées en série (concaténées) à chaque fois, de sorte que les trajets des axes de la première transformation constituent les données d'entrée pour la deuxième transformation concaténée. Les axes machine effectuent les déplacements résultant de la deuxième transformation.

Deux transformations **peuvent** être concaténées.

Remarque

Un outil est toujours affecté à la première transformation d'une transformation résultante. La deuxième transformation se comporte alors comme si la longueur active de l'outil valait zéro. Seules les longueurs de base réglées par paramètres machine pour l'outil (_BASE_TOOL_) sont actives pour la première transformation.

Constructeur de la machine-outil

Veuillez observer les indications du constructeur de machines concernant des transformations éventuellement prédéfinies par paramètres machine.

Les transformations et les transformations concaténées sont des options. Le catalogue le plus récent indique les transformations qu'il est possible de concaténer en fonction du type de commande.

Applications

- Rectification de contours programmés en tant que lignes d'une surface développée de cylindre (TRACYL) avec une meule oblique, par exemple l'affûtage d'outil.
- Finition d'un contour non circulaire généré avec TRANSMIT à l'aide d'une meule oblique.

Syntaxe

`TRACON(trf,par)`	Activation d'une transformation résultant d'une concaténation.
`TRAFOOF`	

Signification

TRACON	Une transformation résultant d'une concaténation est activée. TRACON() désactive implicitement toute transformation active auparavant.
TRAFOOF	La transformation activée (concaténée) en dernier est désactivée.
trf	Numéro de la transformation résultante : 0 ou 1 pour la première/l'unique transformation concaténée. Si ce paramètre n'est pas programmé, est interprété comme valant 0 ou 1, c'est-à-dire que la première/l'unique transformation est activée. 2 représente la deuxième transformation. (Les valeurs différentes de 0, 1, 2 génèrent une alarme).
par	Un ou plusieurs paramètres séparés par des virgules pour les transformations concaténées qui requièrent des paramètres, par exemple l'angle pour un axe oblique. Si des paramètres ne sont pas programmés, les préréglages ou les valeurs utilisées en derniers lieu sont actifs. Les virgules sont nécessaires pour que les paramètres soient traités dans l'ordre dans lequel ils sont escomptés, si des préréglages doivent être pris en considération pour certains paramètres antéposés. En particulier, si un seul paramètre est indiqué, il faut placer une virgule devant celui-ci, même si la programmation de trf est inutile, par exemple TRACON(, 3.7).

Condition préalable

La **deuxième** transformation doit être de type **"Axe oblique"** (TRAANG). La première transformation peut être :

- Transformations d'orientation (TRAORI), tête de fraisage de type cardan comprise
- TRANSMIT
- TRACYL
- TRAANG

L'instruction d'activation d'une transformation résultant d'une concaténation ne peut être utilisée que si les transformations à concaténer et la transformation résultante sont définies par paramètres machine.

Les conditions marginales indiquées dans les descriptions séparées pour les transformations et les cas particuliers doivent également être respectés lors de l'utilisation à l'intérieur d'une concaténation.

Des informations sur la configuration des paramètres machine des transformations se trouvent dans :

/FB2/ Manuel de fonctions d'extension ; transformations cinématiques (M1) et

/FB3/ Manuel de fonctions spéciales ; Transformations 3 à 5 axes (F2)

Corrections d'outils 7

7.1 Mémoire de correcteurs

Fonction

Structure de la mémoire de correcteurs

Chaque champ de données peut être appelé à l'aide d'un numéro T et d'un numéro D (sauf "Numéros D absolus") et contient, outre les données géométriques de l'outil, d'autres indications telles que le type d'outil.

Structure des numéros D absolus

La "structure horizontale des numéros D" est utilisée lorsque la gestion d'outils a lieu en dehors de NCK. Dans ce cas, les numéros D, c'est-à-dire les correcteurs d'outil, sont créés sans relation avec les outils.

Dans le programme pièce, vous pouvez continuer à programmer T. Ce numéro T est cependant indépendant du numéro D programmé.

Données de tranchant utilisateur

Des données utilisateur de tranchant peuvent être configurées à l'aide de PM. Veuillez tenir compte des indications du constructeur de la machine.

Paramètres d'outil

Remarque

Les différentes valeurs de la mémoire de correction

Les différentes valeurs de la mémoires de correction P1 à P25 peuvent être lues et écrites par le programme grâce à des variables système. Tous les autres paramètres sont réservés.

Les paramètres d'outil $TC_DP6 à $TC_DP8, $TC_DP10 et $TC_DP11 ainsi que $TC_DP15 à $TC_DP17, $TC_DP19 et $TC_DP20 possèdent, en fonction du type d'outil, une autre signification.

[1]Est également valable pour les fraises pour le fraisage avant 3D
[2]En cas de scie à rainurer type d'outil
[3]réservé : N'est pas utilisé par la SINUMERIK 840D.

Paramètre d'outil numéro (DP)	Signification des variables système	Remarque
$TC_DP1	Type d'outil	Voir liste ci-dessous
$TC_DP2	Position du tranchant	Uniquement outils de tournage
Géométrie	**Correction de longueur**	
$TC_DP3	Longueur 1	Prise en compte selon
$TC_DP4	Longueur 2	Type et niveau
$TC_DP5	Longueur 3	
Géométrie	**Rayon**	
$TC_DP6[1] $TC_DP6[2]	Rayon 1 / Longueur 1 Diamètre d	Fraise/outil indexable/meule Scie à rainurer
$TC_DP7[1] $TC_DP7[2]	Longueur 2 / Rayon de congé de fraise conique Largeur de rainure b rayon de congé	Fraises Scie à rainurer
$TC_DP8[1] $TC_DP8[2]	Rayon de courbure 1 pour fraises Projection k	Fraises Scie à rainurer
$TC_DP9[1,3]	Rayon de courbure 2	réservé
$TC_DP10[1]	Angle 1 surface d'attaque de l'outil	Fraises coniques
$TC_DP11[1]	Angle 2 axe longitudinal de l'outil	Fraises coniques
Usure	**Correction de longueur et de rayon**	
$TC_DP12	Longueur 1	
$TC_DP13	Longueur 2	
$TC_DP14	Longueur 3	
$TC_DP15[1] $TC_DP15[2]	Rayon 1 / Longueur 1 Diamètre d	Fraise/outil indexable/meule Scie à rainurer
$TC_DP16[1] $TC_DP16[3]	Longueur 2 / Rayon de congé de fraise conique Largeur de rainure b rayon de congé	Fraises Scie à rainurer
$TC_DP17[1] $TC_DP17[2]	Rayon de courbure 1 pour fraises Projection k	Fraises / fraises avant 3D Scie à rainurer
$TC_DP18[1,3]	Rayon de courbure 2	réservé
$TC_DP19[1]	Angle 1 surface d'attaque de l'outil	Fraises coniques
$TC_DP20[1]	Angle 2 axe longitudinal de l'outil	Fraises coniques
Cote de base/adaptateur	**Corrections de longueur**	
$TC_DP21	Longueur 1	
$TC_DP22	Longueur 2	
$TC_DP23	Longueur 3	
Technologie		
$TC_DP24	Angle de dépouille	Uniquement outils de tournage
$TC_DP25		réservé

Commentaires

Les grandeurs géométriques (par exemple longueur 1 ou rayon) sont constituées de plusieurs composantes. Ces dernières sont additionnées pour donner une grandeur résultante (par exemple la longueur totale 1, le rayon total) qui est alors prise en compte pour la correction.

La valeur zéro est affectée aux corrections inutiles.

Paramètres d'outil $TC-DP1 à $TC-DP23 avec outils de contour

Remarque

Les paramètres d'outils qui ne figurent pas dans le tableau, comme par exemple $TC_DP7, ne sont pas évalués autrement dit leur contenu n'a pas d'importance.

Paramètre d'outil numéro (DP)	Signification	Découper Dn		Remarque
$TC_DP1	Type d'outil			400 à 599
$TC_DP2	Position du tranchant			
Géométrie	**Correction de longueur**			
$TC_DP3	Longueur 1			
$TC_DP4	Longueur 2			
$TC_DP5	Longueur 3			
Géométrie	**Rayon**			
$TC_DP6	Rayon			
Géométrie	**Angle limite**			
$TC_DP10	Angle limite minimal			
$TC_DP11	Angle limite maximal			
Usure	**Correction de longueur et de rayon**			
$TC_DP12	Usure Longueur 1			
$TC_DP13	Usure Longueur 2			
$TC_DP14	Usure Longueur 3			
$TC_DP15	Usure rayon			
Usure	**Angle limite**			
$TC_DP19	Usure de l'angle limite min.			
$TC_DP20	Usure de l'angle limite max.			
Cote de base/adaptateur	**Corrections de longueur**			
$TC_DP21	Longueur 1			
$TC_DP22	Longueur 2			
$TC_DP23	Longueur 3			

Valeur de base et valeur d'usure

Les grandeurs résultantes sont la somme de la valeur de base et de la valeur d'usure (par exemple $TC_DP6 + $TC_DP15 pour le rayon). Pour la longueur d'outil du premier tranchant, la cote de base ($TC_DP21 – $TC_DP23) est par ailleurs également additionnée. De plus, toutes les autres grandeurs pouvant influencer la longueur d'outil même pour un outil conventionnel agissent sur cette longueur d'outil (adaptateur, porte-outil orientable, données de réglage).

Angle limite 1 et 2

Les angles limites 1 ou 2 se rapportent chacun au vecteur du centre du bec de l'outil au point de référence du tranchant et sont comptés en sens anti-horaire.

7.2 Corrections additives

7.2.1 Sélection des corrections additives (DL)

Fonction

Les corrections additives permettent de programmer des compensations qui dépendent du processus d'usinage. Elles se rapportent aux données géométriques d'un tranchant et font donc partie des données de tranchant d'outil.

L'accès aux corrections additives a lieu avec un numéro DL (DL : location dependent ; corrections en fonction du lieu d'utilisation) et leur saisie s'effectue sur l'interface utilisateur.

Application

Les corrections additives permettent de compenser les erreurs de cotes dues au lieu d'utilisation.

Syntaxe

```
DL=<numéro>
```

Signification

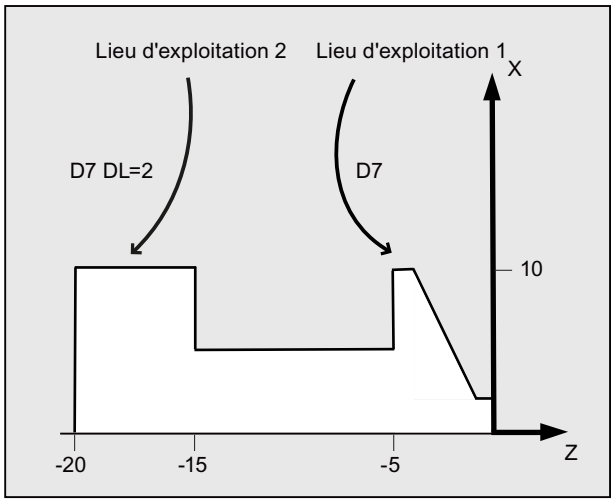
`DL`	Instruction d'activation d'une correction additive
`<Numéro>`	Le bloc de données de correction additive à activer est indiqué dans le paramètre `<numéro>`.

Remarque

La détermination du nombre de corrections additives et leur activation ont lieu par le biais de paramètres machine (→ voir les indications du constructeur de la machine).

Exemple

Le même tranchant est utilisé pour usiner 2 portées de roulement :



Code de programme	**Commentaire**
N110 T7 D7	; La tourelle revolver est positionnée à l'emplacement 7. D7 et DL=1 sont activés et les corrections correspondantes sont exécutées dans le bloc suivant.
N120 G0 X10 Z1	
N130 G1 Z-6	
N140 G0 DL=2 Z-14	; DL=2 est activé en supplément de D7 et la correction correspondante est exécutée dans le bloc suivant.
N150 G1 Z-21	
N160 G0 X200 Z200	; Accostage de la position de changement d'outil.
...	

7.2.2 Définition des valeurs d'usure et de réglage ($TC_SCPxy[t,d], $TC_ECPxy[t,d])

Fonction

La lecture et l'écriture des valeurs d'usure et de réglage s'effectuent à l'aide de variables système. La structure des nouveaux paramètres système est analogue à celle des variables système correspondants pour les outils et les tranchants.

Variables système

Variable système	Signification
$TC_SCPxy[<t>,<d>]	Valeur d'usure associée au paramètre géométrique correspondant par le biais de xy, x étant le numéro de la valeur d'usure et y le numéro du paramètre géométrique.
$TC_ECPxy[<t>,<d>]	Valeur de réglage associée au paramètre géométrique correspondant par le biais de xy, x étant le numéro de la valeur de réglage et y le numéro du paramètre géométrique.
<t> : numéro T de l'outil <d> : numéro D du tranchant de l'outil	

Remarque

Les valeurs d'usure et de réglage sont ajoutées aux valeurs des paramètres géométriques et des autres paramètres de correction (numéro D).

Exemple

La valeur d'usure associée à la longueur 1 est fixée à 1.0 pour le tranchant <d> de l'outil <t>.

Paramètre : $TC_DP**3** (longueur 1, pour outils de tournage)

Valeurs d'usure : $TC_SCP1**3** à $TC_SCP6**3**

Valeurs de réglage : $TC_ECP1**3** à $TC_ECP6**3**

$TC_SCP4**3** [<t>,<d>] = 1.0

7.2.3 Effacer les corrections additives (DELDL)

Fonction

L'instruction DELDL permet d'effacer les corrections additives associées à des tranchants d'outil (libération d'espace mémoire). L'effacement concerne les valeurs d'usure aussi bien que les valeurs de réglage.

Syntaxe

```
DELDL[<t>,<d>]
DELDL[<t>]
DELDL
<Etat>=DELDL[<t>,<d>]
```

Signification

DELDL	Instruction de suppression des corrections additives
<t>	Numéro T de l'outil
<d>	Numéro D du tranchant de l'outil
DELDL[<t>,<d>]	Suppression toutes les corrections additives du tranchant <d> de l'outil <t>.
DELDL[<t>]	Suppression de toutes les corrections additives de tous les tranchants de l'outil <t>.
DELDL	Suppression de toutes les corrections additives de tous les tranchants de tous les outils de l'unité TO (pour le canal dans lequel l'instruction est programmée).
<Etat>	Etat de suppression
	Valeur : Signification :
	0 Effacement exécuté correctement.
	- L'effacement n'a pas eu lieu (lorsque le paramétrage désigne un tranchant précis) ou l'effacement n'a pas été entièrement effectué (lorsque le paramétrage désigne plusieurs tranchants).

Remarque

Les valeurs d'usure et de réglage d'outils actifs ne peuvent pas être effacées (par analogie avec l'effacement de correcteurs D et de données d'outil).

7.3 Correcteur d'outil : Interventions spéciales

Fonction

Les données de réglage SD42900 à SD42960 permettent de gérer l'exploitation du signe pour la longueur d'outil et l'usure.

Ceci concerne également le comportement des composantes d'usure lors de l'application de la fonction miroir à des axes géométriques ou en cas de changement de plan de travail, mais aussi pour la compensation de température dans la direction d'outil.

Valeurs d'usure

Lorsque nous faisons référence à des valeurs d'usure dans ce qui suit, il s'agit de la somme des valeurs d'usure proprement dites ($TC_DP12 à $TC_DP20) et des corrections totales qui comprennent les valeurs d'usure ($SCPX3 à $SCPX11) et les valeurs de réglage ($ECPX3 à $ECPX11).

Pour de plus amples informations sur les corrections totales, voir :
Bibliographie :
Description fonctionnelle Gestion des outils

Données de réglage

Données de réglage	Signification
SD42900 $SC_MIRROR_TOOL_LENGTH	Fonction miroir appliquée aux composantes de longueur d'outil et de cote de base.
SD42910 $SC_MIRROR_TOOL_WEAR	Fonction miroir appliquée aux valeurs d'usure pour les composantes de longueur d'outil.
SD42920 $SC_WEAR_SIGN_CUTPOS	Exploitation du signe des composantes d'usure en fonction de la position du tranchant.
SD42930 $SC_WEAR_SIGN	Inverse le signe des cotes d'usure.
SD42935 $SC_WEAR_TRANSFORM	Transformation des valeurs d'usure.
SD42940 $SC_TOOL_LENGTH_CONST	Affectation des composantes de longueur d'outil aux axes géométriques.
SD42950 $SC_TOOL_LENGTH_TYPE	Affectation des composantes de longueur d'outil indépendamment du type d'outil-
SD42960 $SC_TOOL_TEMP_COMP	Valeur de compensation de la température dans la direction de l'outil. Est également active pour l'orientation de l'outil actuelle.

Bibliographie

Description fonctionnelle Fonctions de base, Correction d'outil (W1)

Autres informations

Prise d'effet des données de réglage modifiées

En cas de modification des données de réglage décrites, les composantes du correcteur d'outil ne sont actualisées que lors de la prochaine sélection du tranchant d'outil. Dans le cas où un outil est déjà actif et où les données de correction modifiées de cet outil doivent prendre effet, il faut appeler à nouveau cet outil.

Ceci est également valable si la longueur résultante de l'outil est modifiée du fait de l'application de la fonction miroir à un axe. L'outil doit faire l'objet d'une nouvelle sélection après la fonction miroir afin que les composantes modifiées de longueur d'outil prennent effet.

Organes porte-outil orientables et nouvelles données de réglage

Les données de réglage SD42900 à SD42940 n'ont pas d'effet sur les composantes d'un organe porte-outil orientable éventuellement actif. En présence d'un organe porte-outil orientable, l'outil est toujours pris en considération avec sa longueur totale résultante (longueur d'outil + usure + cote de base). Lors du calcul de cette longueur totale résultante, la commande tient compte de toutes les modifications provoquées par des données de réglage c'est-à-dire les vecteurs d'un organe porte-outil orientable sont donc indépendants du plan d'usinage.

Remarque

En cas de mise en oeuvre d'organes porte-outil orientables, il sera souvent préférable de définir tous les outils pour un système de base non inversé par fonction miroir, même ceux utilisés uniquement pour un usinage symétrique. Pour l'usinage avec des axes inversés, l'organe porte-outil sera pivoté de telle sorte que la position effective de l'outil soit décrite correctement. Toutes les composantes de longueur d'outil agissent alors automatiquement dans le sens correct, de sorte que l'interprétation de certaines composantes par les données de réglage en fonction de l'état d'inversion d'axes est superflue.

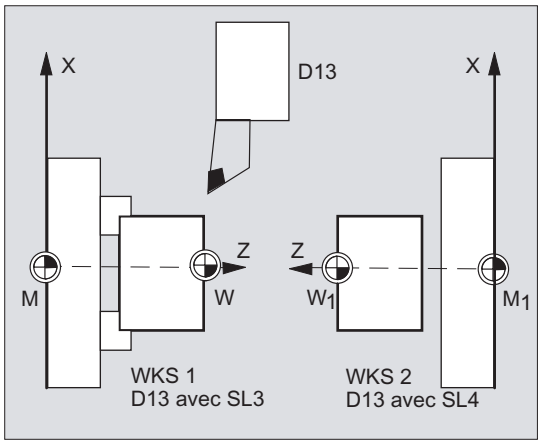
Autres possibilités d'applications

L'utilisation de la fonctionnalité "organe porte-outil orientable" peut aussi s'avérer intéressante si la machine ne permet pas d'orienter physiquement des outils, mais dispose d'outils ayant des orientations diverses et installés à demeure. La cotation des outils peut alors être effectuée uniformément avec une orientation de base, les cotes valables pour l'usinage s'obtenant par rotation d'un organe porte-outil virtuel.

7.3.1 Application de la fonction miroir aux longueurs d'outil

Fonction

Avec des données de réglage SD42900 $SC_MIRROR_TOOL_LENGTH et SD42910 $SC_MIRROR_TOOL_WEAR différentes de zéro, il est possible d'appliquer la fonction miroir aux composantes de longueur d'outil et de cote de base avec les valeurs d'usure par rapport à leurs axes respectifs.



SD42900 $SC_MIRROR_TOOL_LENGTH

Donnée de réglage **différente** de zéro :

Les composantes de longueur d'outil ($TC_DP3, $TC_DP4 et $TC_DP5) et les composantes de cote de base ($TC_DP21, $TC_DP22 et $TC_DP23), dont les axes associés font l'objet de la fonction miroir, sont aussi soumises à la fonction miroir - par inversion de leur signe.

La fonction miroir n'est **pas** appliquée aux valeurs d'usure. Si cela doit être le cas, vous devez régler de façon adéquate la donnée de réglage SD42910 $SC_MIRROR_TOOL_WEAR.

SD42910 $SC_MIRROR_TOOL_WEAR

Donnée de réglage **différente** de zéro :

Les valeurs d'usure pour les composantes de longueur d'outil, dont les axes associés font l'objet de la fonction miroir, sont aussi soumises à la fonction miroir - par inversion de leur signe.

7.3.2 Exploitation du signe de l'usure

Fonction

Avec les données de réglage SD42920 $SC_WEAR_SIGN_CUTPOS et SD42930 $SC_WEAR_SIGN différentes de zéro, il est possible d'inverser l'exploitation du signe des composantes d'usure.

SD42920 $SC_WEAR_SIGN_CUTPOS

Donnée de réglage **différente** de zéro :

Dans le cas des outils à position de tranchant définie (outils de tournage et de rectification – type d'outil 400), l'exploitation du signe des composantes d'usure dans le plan de travail dépend de la position du tranchant. Dans le cas des outils sans position de tranchant définie, cette donnée de réglage n'est pas prise en considération.

Dans le tableau suivant, les cotes dont le signe est inversé par le SD42920 (différent de 0) sont repérées par X :

Position de tranchant	Longueur 1	Longueur 2
1		
2		X
3	X	X
4	X	
5		
6		
7		X
8	X	
9		

Remarque

Les modifications de signe en fonction des SD42920 et SD42910 sont indépendantes. Si le signe d'une cote est modifié par les deux données de réglage par exemple, il restera inchangé.

SD42930 $SC_WEAR_SIGN

Donnée de réglage **différente** de zéro :

Le signe de toutes les cotes d'usure est inversé. Cette donnée agit aussi bien sur la longueur d'outil que sur les autres dimensions, par exemple le rayon de l'outil, le rayon d'arrondi, etc.

S une cote d'usure positive est entrée, l'outil devient "plus court" et "plus fin". A ce sujet, voir chapitre "Correcteur d'outil : Interventions spéciales, Prise d'effet des données de réglage modifiées".

7.3.3 Système de coordonnées de l'usinage actif (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS)

Fonction

En fonction de la cinématique de la machine ou de la présence d'un organe porte-outil orientable, les valeurs d'usure mesurées dans l'un de ces systèmes de coordonnées sont transposées voire transformées dans un système de coordonnées adapté.

Systèmes de coordonnées de l'usinage actif

Des offsets de longueur d'outil peuvent résulter des systèmes de coordonnées suivants et prendre en compte en compte les composantes d'usure de longueur d'outil avec le code G correspondant du groupe 56 dans un outil actif.

- Système de coordonnées machine (SCM)
- Système de coordonnées de base (SCB)
- Système de coordonnées pièce (SCP)
- Système de coordonnées de l'outil (SCO)
- Système de coordonnées de l'outil de transformation cinématique (SCT)

Syntaxe

```
TOWSTD
TOWMCS
TOWWCS
TOWBCS
TOWTCS
TOWKCS
```

Signification

TOWSTD	Valeur de position de base pour corrections sur la longueur d'outil de la valeur d'usure
TOWMCS	Corrections dans la longueur d'outil dans SCM
TOWWCS	Corrections dans la longueur d'outil dans SCP
TOWBCS	Corrections dans la longueur d'outil dans SCB
TOWTCS	Corrections de la longueur d'outil sur le point de référence de l'organe porte-outil (organe porte-outil orientable)
TOWKCS	Corrections de la longueur d'outil de la tête d'outil (transformation cinétique)

Autres informations

Différences

Le tableau suivant représente les principales différences :

Code G	Valeur d'usure	Organe porte-outil orientable actif
TOWSTD	Valeur de position initiale, longueur d'outil	Les valeurs d'usure sont assujetties à la rotation.
TOWMCS	Valeur d'usure dans SCM. TOWMCS est identique à TOWSTD lorsque aucun organe porte-outil orientable n'est actif.	Seul le vecteur de la longueur d'outil résultante tourne, sans prise en compte de l'usure.
TOWWCS	La valeur d'usure est convertie du SCP au SCM.	Le vecteur d'outil est calculé sans prise en compte de l'usure comme pour TOWMCS.
TOWBCS	La valeur d'usure est convertie dans SCB au SCM.	Le vecteur d'outil est calculé sans prise en compte de l'usure comme pour TOWMCS.
TOWTCS	La valeur d'usure est convertie dans le système de coordonnées de l'outil au SCM.	Le vecteur d'outil est calculé sans prise en compte de l'usure comme pour TOWMCS.

TOWWCS , TOWBCS, TOWTCS: Le vecteur d'usure est additionné au vecteur d'outil.

Transformation linéaire

La longueur d'outil est uniquement judicieusement définissable dans SCM lorsque le SCM est issu de SCB par une transformation linéaire.

Transformation non linéaire

Si par exemple une transformation non linéaire est active avec TRANSMIT, alors le SCB est automatiquement utilisé lorsque le SCM est indiqué comme système de coordonnées souhaité.

Pas de transformation cinématique et pas d'organe porte-outil orientable

Lorsque ni une transformation cinématique ni un organe porte-outil orientable n'est actif, alors, mis à part le SCP, les quatre autres systèmes de coordonnées coïncident. Le SCP se différencie alors de tous les autres. Du fait que seules les longueurs d'outil doivent être interprétées, les translations entre les systèmes de coordonnées n'ont plus de signification.

Bibliographie :

Pour d'autres informations sur la correction d'outil, voir :
Description fonctionnelle Fonctions de base, Correction d'outil (W1)

Prise en compte des valeurs d'usure

La donnée de réglage **SD42935 $SC_WEAR_TRANSFORM**détermine quelle composante d'usure parmi les trois composantes :

- Usure
- Corrections totales fines
- Corrections totales grossières

d'une rotation doit être assujettie par une transformation d'adaptateur ou par un organe porte-outil orientable lorsque l'un des codes G suivants est actif :

- TOWSTD Position d'initialisation

  pour corrections sur la longueur d'outil
- TOWMCS Valeurs d'usure

  dans le système de coordonnées machine (SCM)
- TOWWCS Valeurs d'usure

  dans le système de coordonnées pièce (SCP)
- TOWBCS Valeurs d'usure (SCB)

  dans le système de coordonnées de base
- TOWTCS Valeurs d'usure dans le système de coordonnées d'outil sur le mandrin du porte-outil (référence de l'organe porte-outil T)
- TOWKCS Valeurs d'usure dans le système de coordonnées de la tête d'outil avec transformation cinétique

---

Remarque

L'évaluation de chaque composante d'usure (affectation aux axes géométriques, exploitation du signe) est influencée par :

- le plan actif,
- la transformation d'adaptateur,
- les données de réglage suivantes :
  - 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 Longueurs d'outil et changement de plan

Fonction

Avec les données de réglage SD42940 $SC_TOOL_LENGTH_CONST différentes de zéro, il est possible d'affecter les composantes de longueur d'outil telles que la longueur, l'usure et la cote de base aux axes géométriques pour les outils de tournage et de rectification lors d'un changement de plan.

SD42940 $SC_TOOL_LENGTH_CONST

Donnée de réglage **différente** de zéro :

L'affectation des composantes de longueur d'outil (longueur, usure et cote de base) aux axes géométriques n'est pas modifiée en cas de changement de plan de travail (G17 à G19).

Le tableau suivant indique l'affectation des composantes de longueur d'outil aux axes géométriques pour des outils de tournage et de rectification (types 400 à 599) :

Sommaire	Longueur 1	Longueur 2	Longueur 3
17	Y	X	Z
*)	X	Z	Y
19	Z	Y	X
-17	X	Y	Z
-18	Z	X	Y
-19	Y	Z	X

*) Toute valeur différente de 0, qui n'est pas égale à une des six valeurs mentionnées, est interprétée comme la valeur 18.

Le tableau suivant indique l'affectation des composantes de longueur d'outil aux axes géométriques pour tous les autres outils (types < 400 ou > 599) :

Plan de travail	Longueur 1	Longueur 2	Longueur 3
*)	Z	Y	X
18	Y	X	Z
19	X	Z	Y
-17	Z	X	Y
-18	Y	Z	X
-19	X	Y	Z

*) Toute valeur différente de 0, qui n'est pas égale à une des six valeurs mentionnées, est interprétée comme la valeur 17.

Remarque

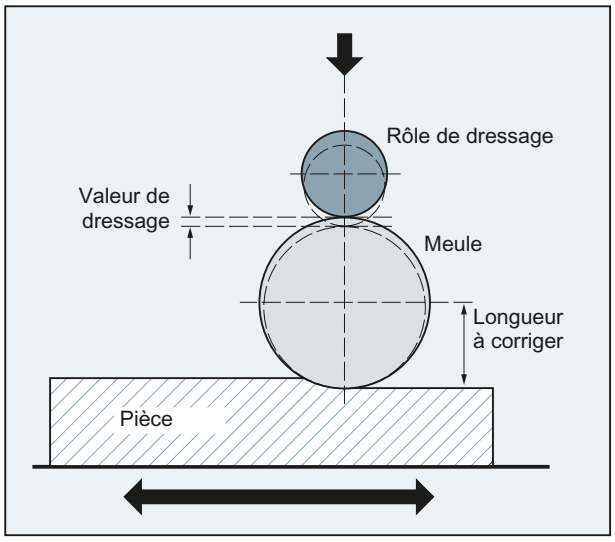
Dans ces tableaux, on suppose que les axes géométriques 1 à 3 sont désignés par X, Y, Z. Pour l'affectation d'une correction à un axe, ce n'est pas le descripteur d'axe mais l'ordre des axes qui est déterminant.

7.4 Correction d'outil en ligne (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF)

Fonction

Lorsque la fonction "Correction d'outil en ligne" est activée, une correction de longueur d'outil de rectification résultant de l'usinage peut être prise en compte immédiatement.

Le dressage CD, dans lequel la meule est dressée parallèlement à l'usinage constitue un exemple d'application :



La correction de longueur d'outil peut être modifiée à partir du canal d'usinage ou d'un canal parallèle (canal de dressage).

En fonction de l'instant souhaité pour l'opération de dressage, on utilise différentes fonction pour l'écriture de la correction d'outil en ligne :

- Ecriture en continu bloc par bloc (`PUTFTOCF`)

  Avec `PUTFTOCF`, le processus de dressage s'effectue en même temps que l'usinage.

  La correction d'outil est modifiée en continu dans le canal d'usinage suivant une fonction polynômiale du premier, second ou troisième degré qui doit être définie préalablement avec `FCTDEF`.

  `PUTFTOCF` agit toujours bloc par bloc, autrement dit dans le bloc de déplacement qui suit.

- Ecriture en continu à effet modal : `ID=1 DO FTOC` (voir "Correction d'outil en ligne (FTOC) (Page 593)")

- Ecriture discrète (`PUTFTOC`)

  Avec `PUTFTOC`, le processus de dressage ne s'effectue pas en même temps que l'usinage à partir d'un canal parallèle. La valeur de correction indiquée par `PUTFTOC` est immédiatement activée dans le canal de destination.

Remarque

La correction d'outil en ligne s'applique uniquement aux outils de rectification.

Syntaxe

Activation/désactivation d'outil en ligne dans le canal de destination :

```
FTOCON
...
FTOCOF
```

Ecriture de la correction d'outil en ligne :

- Continue, bloc par bloc :

```
FCTDEF(<fonction>,<LLimit>,<ULimit>,<a0>,<a1>,<a2>,<a3>)
PUTFTOCF(<fonction>,<valeur de référence>,<paramètre d'outil>,<canal>,<broche>)
...
```

- Discrète :

```
PUTFTOC(<valeur de correction>,<paramètre d'outil>,<canal>,<broche>)
...
```

Signification

`FTOCON` : Activation de la correction d'outil en ligne

`FTOCON` doit être programmée dans un canal dans lequel la correction d'outil doit être activée.

`FTOCOF` : Annulation de la correction d'outil en ligne

Avec `FTOCOF`, la correction n'est pas plus dégagée cependant, dans les données de correction spécifiques au tranchant, le montant complet écrit avec `PUTFTOC/PUTFTOCF` est corrigé.

Nota :
pour la désactivation définitive de la correction d'outil en ligne, `FTOCOF` doit être suivie d'une sélection/désélection de l'outil (`T...`).

`FCTDEF` : `FCTDEF` définit la fonction polynômiale pour `PUTFTOCF`.

Paramètre :

`<fonction>` : Numéro de la fonction polynômiale
Type : INT

`<LLimit>` : valeur limite inférieure
Type : REAL

`<ULimit>` : valeur limite supérieure
Type : REAL

`<a0> ... <a3>` : Coefficients de la fonction polynômiale
Type : REAL

`PUTFTOCF` : Appel de la fonction "Ecriture continue modale de la correction d'outil en ligne"

Paramètre :

`<fonction>` : Numéro de la fonction polynômiale

Type : INT

Nota :
doit correspondre avec l'indication dans `FCTDEF`.

`<valeur de référence>` : Valeur de référence variable à partir de laquelle la correction doit être déduite (p. ex. valeur réelle changeante).

Type : VAR REAL

`<paramètre d'outil>` : Numéro du paramètre d'usure (longueur 1, 2 ou 3) dans lequel additionner la valeur de correction.

Type : INT

`<canal>` : Numéro du canal dans lequel la correction d'outil en ligne doit être active.

Type : INT

Nota :
une indication est uniquement requise lorsque la correction ne doit pas être effective dans le canal actif.

`<broche>` : Numéro de la broche dans laquelle la correction d'outil en ligne doit être active.

Type : INT

Nota :
une indication est uniquement requise lorsqu'il s'agit de corriger une meule non active à la place de l'outil actif mis en oeuvre.

`PUTFTOC` : Appel de la fonction "Ecriture discrète de la correction d'outil en ligne"

Paramètre :

`<valeur de correction>` : Valeur de correction à additionner au paramètre d'usure.

Type : REAL

`<paramètre d'outil>` : voir `PUTFTOCF`

`<canal>` : Numéro du canal dans lequel la correction d'outil en ligne doit être active.

Type : INT

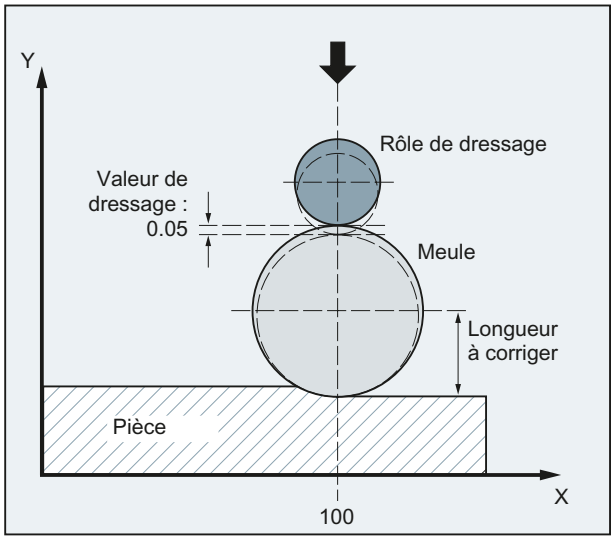
`<broche>` : voir `PUTFTOCF`

Exemple

Rectifieuse plane avec :

- Y : axe de pénétration pour la meule
- V : axe de pénétration pour la molette de dressage
- Canal d'usinage : canal 1 avec les axes X, Z, Y
- Canal de dressage : canal 2 avec l'axe V

Après le début du déplacement de la meule, on souhaite obtenir une profondeur de passe de dressage de la meule de 0,05 à X100. La profondeur de passe de dressage doit entrer en action avec "Ecriture continue de la correction d'outil en ligne".



Programme d'usinage dans le canal 1 :

Code de programme	Commentaire
…	
N110 G1 G18 F10 G90	; Position d'initialisation :
N120 T1 D1	; Sélection de l'outil courant.
N130 S100 M3 X100	; Mise en marche de la broche, accostage de la position initiale.
N140 INIT(2,"DRESSAGE","S")	; Sélection du programme de dressage dans le canal 2.
N150 START (2)	; Lancement du programme de dressage dans le canal 2.
N160 X200	; Accostage de la position de destination.
N170 FTOCON	; Activation de la correction en ligne.
N… G1 X100	; Suite de l'usinage.
N… M30	

Programme de dressage dans le canal 2 :

```
Code de programme                                  Commentaire
…
N40 FCTDEF(1,-1000,1000,-$AA_IW[V],1)              ; Définition de la fonction : droite
                                                     avec pente=1.
N50 PUTFTOCF(1,$AA_IW[V],3,1)                      ; Ecriture continue de la correction
                                                     d'outil en ligne : la longueur 3 de
                                                     la meule courante est corrigée dans
                                                     le canal 1 en fonction du
                                                     déplacement de l'axe V.
N60 V-0.05 G1 F0.01 G91                            ; Mouvement de pénétration pour
                                                     dressage, PUTFTOCF agit uniquement
                                                     dans ce bloc.
…
N… M30
```

Informations complémentaires

Généralités sur la correction d'outil en ligne

Dans le cas de l'écriture en continu (à la période d'appel de l'interpolateur), après l'activation de la fonction de traitement, chaque modification est additionnée dans la mémoire des usures (pour éviter des sauts de valeurs de consigne).

Dans tous les cas : La correction d'outil en ligne peut agir dans chaque canal, pour chaque broche, pour la longueur 1, 2 **ou** 3 des paramètres d'usure.

L'affectation des longueurs aux axes géométriques s'effectue à l'aide du plan de travail courant.

L'affectation des broches à l'outil s'effectue à l'aide des données d'outil pour GWPSON ou TMON, à condition qu'il ne s'agisse pas de la meule courante.

La correction agit toujours sur le paramètre d'usure pour le côté courant de la meule ou le côté gauche quand les outils ne sont pas actifs.

Remarque

Si la correction est identique pour plusieurs côtés de la meule, on applique une règle de concaténation pour que les valeurs soient reprises automatiquement pour le côté suivant de la meule.

Si des corrections en ligne sont prescrites pour un canal d'usinage, les valeurs d'usure pour l'outil courant dans ce canal ne doivent pas être modifiées à partir du programme d'usinage ou au tableau de commande.

La correction d'outil en ligne est également prise en compte pour la vitesse périphérique constante de meule (VPM), ainsi que pour la surveillance d'outil (TMON).

7.5 Activation des corrections d'outil 3D (CUT3DC..., CUT3DF...)

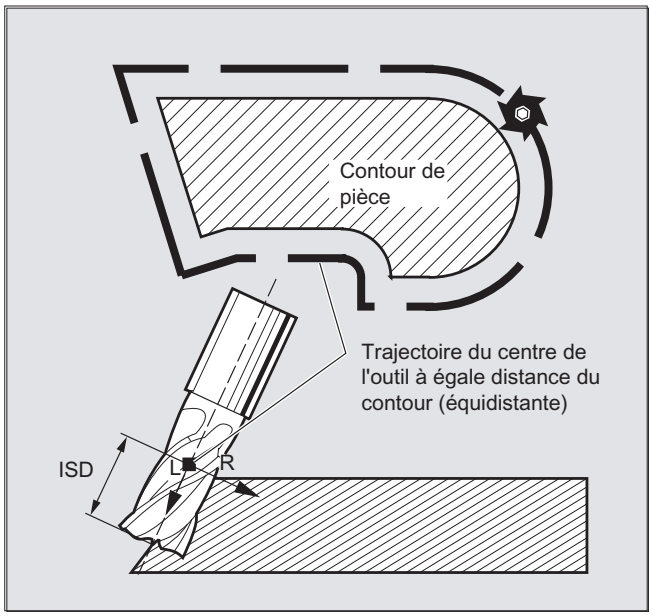
7.5.1 Activation des corrections d'outil 3D (CUT3DC, CUT3DF, CUT3DFS, CUT3DFF, ISD)

Fonction

L'orientation variable de l'outil est prise en compte dans la correction du rayon des outils cylindriques.

Les instructions pour la sélection de la correction de rayon d'outil 3D sont les mêmes que pour la correction de rayon d'outil 2D. La correction à gauche/droite dans le sens de déplacement est indiquée avec G41/G42. L'accostage se fait toujours suivant NORM. La correction de rayon d'outil 3D agit uniquement si la transformation 5 axes est activée.

La correction de rayon d'outil 3D est également appelée correction 5D, vu que, dans ce cas, on dispose de cinq degrés de liberté pour la position de l'outil dans l'espace.



Différence entre corrections de rayon d'outil 2D 1/2 et 3D

Dans le cas de la correction de rayon d'outil 3D, l'orientation de l'outil est variable. Dans le cas de la correction de rayon d'outil 2D 1/2, on considère que l'outil a une orientation constante.

Syntaxe

```
CUT3DC
CUT3DFS
CUT3DFF
CUT3DF
ISD=<valeur>
```

Signification

`CUT3DC`	Activation de la correction de rayon 3D pour le fraisage périphérique
`CUT3DFS`	Correction d'outil 3D pour le fraisage en bout avec orientation d'outil constante. L'orientation de l'outil est définie par `G17` - `G19` et n'est pas influencée par les frames.
`CUT3DFF`	Correction d'outil 3D pour le fraisage en bout avec orientation d'outil constante. L'orientation de l'outil est définie par `G17` - `G19` et, le cas échéant, par la composante rotationnelle d'un frame.
`CUT3DF`	Correction d'outil 3D pour le fraisage en bout avec modification de l'orientation d'outil (uniquement si la transformation 5 axes est activée).
`G40 X... Y... Z...`	Pour la désactivation : bloc linéaire `G0`/`G1` avec axes géométriques
`ISD`	Profondeur de pénétration

Remarque

Les instructions sont à effet modal et figurent dans le même groupe que `CUT2D` et `CUT2DF`. La désactivation n'a lieu que lors du prochain déplacement dans le plan courant. Ceci est toujours valable pour `G40` et est indépendant de l'instruction `CUT`.

Des blocs intermédiaires sont permis lorsque la correction du rayon d'outil 3D est active. Les caractéristiques de la correction de rayon d'outil 2D 1/2 sont applicables.

Conditions marginales

- **G450/G451 et DISC**

  Un bloc à interpolation circulaire est toujours inséré aux angles saillants. `G450/G451` sont sans signification.
  L'instruction `DISC` n'est pas prise en compte.

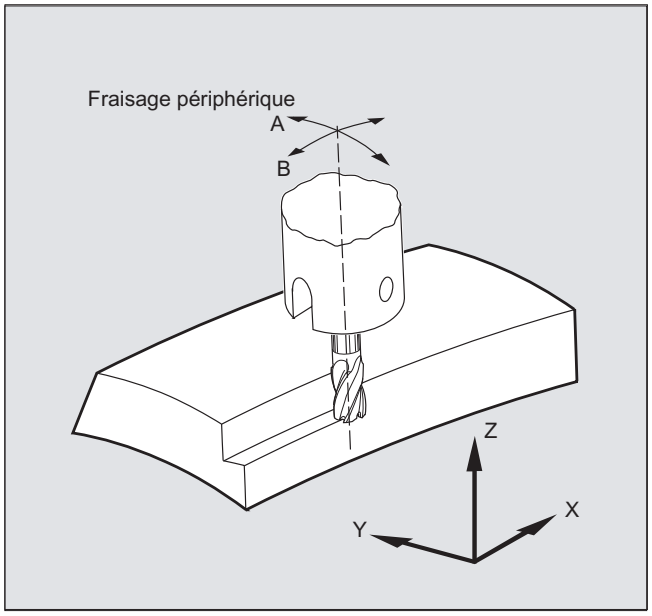
Exemple

Code de programme	Commentaire
N10 A0 B0 X0 Y0 Z0 F5000	
N20 T1 D1	; Appel de l'outil, appel des valeurs de correction d'outil.
N30 TRAORI(1)	; Activation de la transformation
N40 CUT3DC	; Activation de la correction du rayon d'outil 3D
N50 G42 X10 Y10	; Activation de la correction du rayon d'outil
N60 X60	
N70 …	

7.5.2 Correction d'outil 3D : fraisage périphérique, fraisage en bout

Fraisage périphérique

La variante de fraisage périphérique utilisée ici est réalisée par spécification d'un contour (ligne directrice) et de l'orientation correspondante. Dans ce type d'usinage, la forme de l'outil n'a aucune importance pour la trajectoire. Seul le rayon est déterminant au point d'attaque de l'outil.

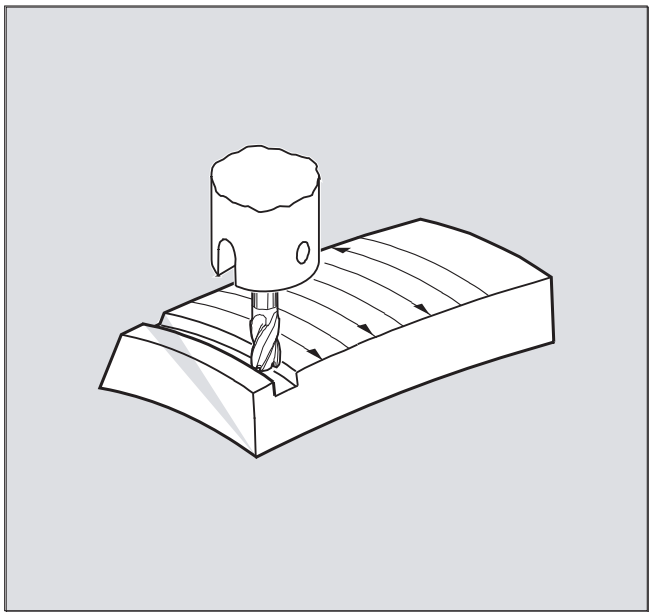


Remarque

La fonction de correction de rayon d'outil 3D est limitée aux outils cylindriques.

Fraisage en bout

Pour ce type de fraisage 3D, vous avez besoin de la description des différentes trajectoires 3D qui balayent la surface de la pièce, ligne après ligne. Les calculs sont généralement réalisés à l'aide d'un système de FAO, en tenant compte de la forme et des dimensions de l'outil. Dans le programme pièce, le postprocesseur écrit non seulement les blocs CN, mais également les orientations d'outil (lorsque la transformation 5 axes est active) et le code G de la correction d'outil 3D souhaitée. Ainsi l'opérateur de la machine a la possibilité d'utiliser des outils légèrement plus petits par rapport à l'outil utilisé pour le calcul des trajectoires CN.



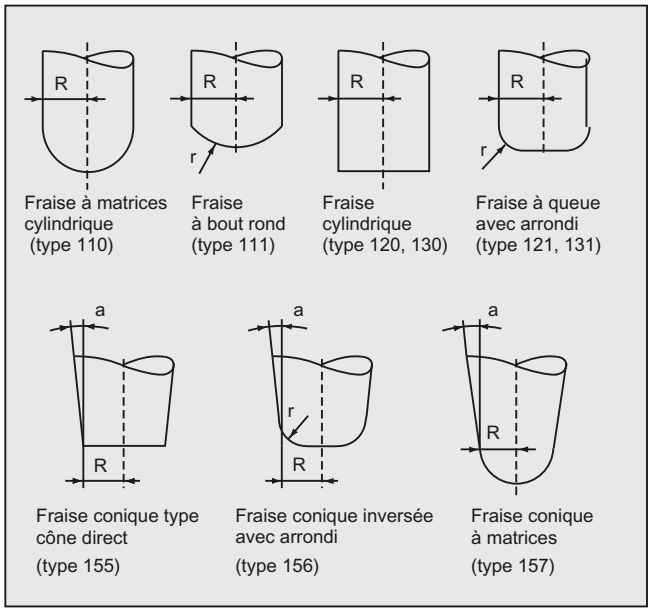
Exemple :

Calcul des blocs CN avec une fraise de 10 mm. L'usinage serait également possible avec un diamètre de fraise de 9,9 mm, le profil des aspérités pouvant cependant varier dans ce cas.

7.5.3 Correction d'outil 3D : formes et données d'outil pour le fraisage en bout

Forme des fraises, données d'outil

Les différentes formes d'outil utilisables pour le fraisage en bout et les valeurs limites des données d'outil sont résumées ci-après. La forme de la tige d'outil n'est pas prise en compte. L'effet des types d'outils 120 et 156 est identique.



Si un numéro de type d'outil qui ne figure pas sur la figure est indiqué dans le programme CN, la commande utilise automatiquement le type d'outil 110 (fraise à bout hémisphérique). Une alarme est émise si les valeurs limites des données d'outil ne sont pas respectées.

Type de fraise	N° de type	R	r	a
Fraise à matrice cylindrique	110	> 0	-	-
Fraise à bout rond	111	> 0	>R	-
Fraise cylindrique deux tailles à queue, fraise pour tête à renvoi d'angle	120, 130	> 0	-	-
Fraise cylindrique deux tailles à queue, fraise pour tête à renvoi d'angle avec arrondi	121, 131	>r	> 0	-
Fraise conique type cône direct	155	> 0	-	> 0
Fraise conique type cône direct avec arrondi	156	> 0	> 0	> 0
Fraise conique à matrices	157	> 0	-	> 0

R = rayon de tige (rayon d'outil)

r = rayon d'arrondi

a = angle entre l'axe longitudinal de l'outil { et l'extrémité supérieure de la surface du tore

- = n'est pas analysé

Données d'outil	Paramètres d'outil	
Cotes de l'outil	Géométrie	Usure
R	$TC_DP6	$TC_DP15
r	$TC_DP7	$TC_DP16
a	$TC_DP11	$TC_DP20

Correction de longueur d'outil

La correction de longueur se réfère à la pointe de l'outil (point d'intersection axe longitudinal/surface frontale).

Correction d'outil 3D, changement d'outil

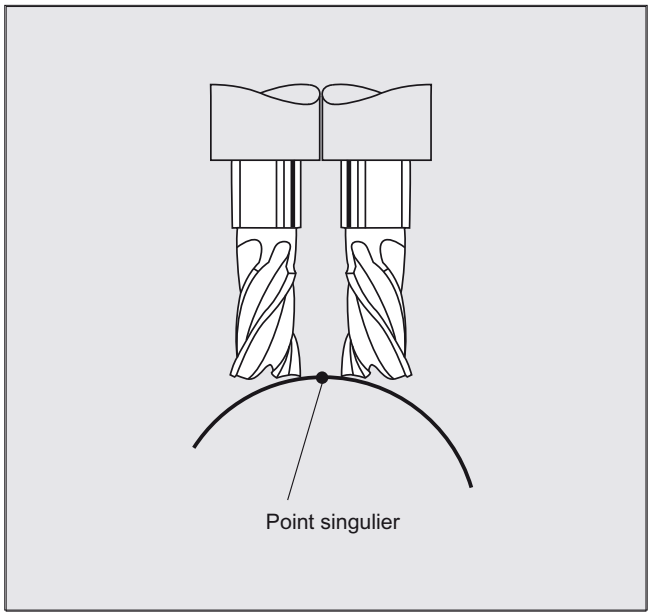
Un nouvel outil de dimensions modifiées (R, r, a) ou de forme différente ne peut être indiqué qu'avec la programmation de `G41` ou `G42` (passage de `G40` à `G41` ou à `G42`, reprogrammation de `G41` ou `G42`). Aucune autre donnée d'outil (par exemple la longueur) n'est concernée par cette règle, ce qui signifie que ces outils peuvent être mis en place même sans reprogrammation de `G41` ou `G42`.

7.5.4 Correction d'outil 3D : Correction sur la trajectoire, courbure de la trajectoire, profondeur de pénétration (CUT3DC, ISD)

Fonction

Correction sur la trajectoire

Dans le cas du fraisage en bout, il faut considérer le cas où le point de contact change brusquement au niveau de la surface de l'outil, comme dans le cas de l'usinage d'une surface convexe avec un outil vertical (exemple ci-contre). L'exemple représenté sur la figure ci-contre peut être considéré comme un cas limite.



Ce cas limite est surveillé par la commande qui détecte immédiatement toute modification brutale du point de contact en s'appuyant sur les angles formés par l'outil et les vecteurs normaux à la surface. A ces emplacements, la commande insère des blocs à interpolation linéaire, de sorte que le déplacement peut être effectué.

Pour le calcul de ces blocs à interpolation linéaire, des plages angulaires admissibles pour l'angle latéral figurent dans des paramètres machine. Une alarme est émise si les plages angulaires admissibles figurant dans des paramètres machines sont dépassées.

Courbure de la trajectoire

La courbure de la trajectoire ne fait pas l'objet d'une surveillance. C'est pourquoi il est également recommandé de n'utiliser que des outils avec lesquels un usinage sans violation du contour est garanti.

Profondeur de pénétration (ISD)

La profondeur de pénétration ISD est prise en compte uniquement si la correction de rayon d'outil 3D est active.

Avec l'instruction `ISD` (Insertion Depth), vous programmez la profondeur de pénétration de l'outil lors du fraisage périphérique. Cela permet de modifier la position du point d'usinage sur la surface latérale de l'outil.

Syntaxe

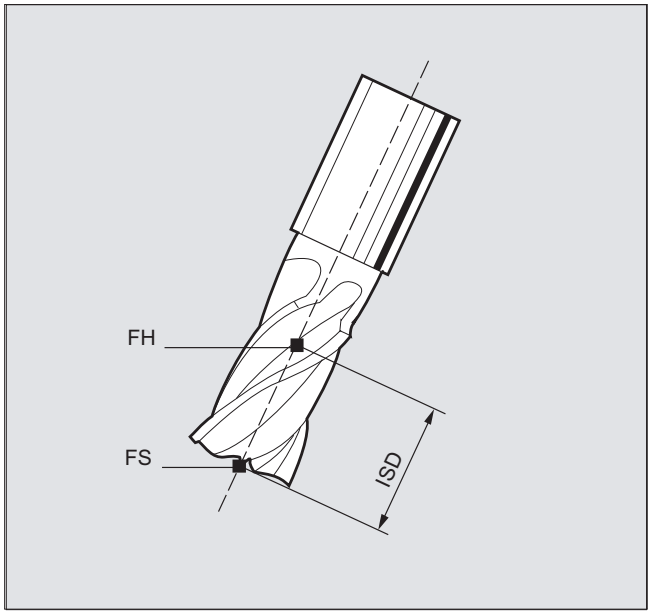
Correction d'outil 3D pour fraisage périphérique
```
CUT3DC
ISD=<valeur>
```

Signification

`CUT3DC`	Activer la correction d'outil 3D pour fraisage périphérique, par exemple le fraisage de poche avec faces latérales obliques.
`ISD`	L'instruction `ISD` indique la distance (`<valeur>`) entre la pointe de la fraise (FS) et le point auxiliaire de la fraise (FH).

Point auxiliaire de la fraise

Le point auxiliaire de la fraise (FH) est obtenu par projection du point d'usinage programmé sur l'axe de l'outil.



Autres informations

Fraisage de poche avec faces latérales obliques pour fraisage périphérique avec CUT3DC

Dans le cas de cette correction de rayon d'outil 3D, un écart du rayon de fraise est compensé par un déplacement perpendiculaire à la surface à usiner. Le plan dans lequel se trouve la surface d'attaque de la fraise reste inchangé lorsque la profondeur de pénétration `ISD` reste inchangée. Une fraise dotée par exemple d'un rayon plus petit par rapport à un outil standard n'atteindrait pas le fond de la poche, qui représente également la surface de délimitation. Pour permettre un déplacement perpendiculaire automatique de l'outil, la commande doit connaître cette surface de délimitation (voir chapitre "Fraisage périphérique 3D avec surfaces de délimitation").

Pour d'autres informations sur la surveillance anticollision, voir :
Bibliographie :
Manuel de programmation Notions de base, chapitre "Corrections d'outil".

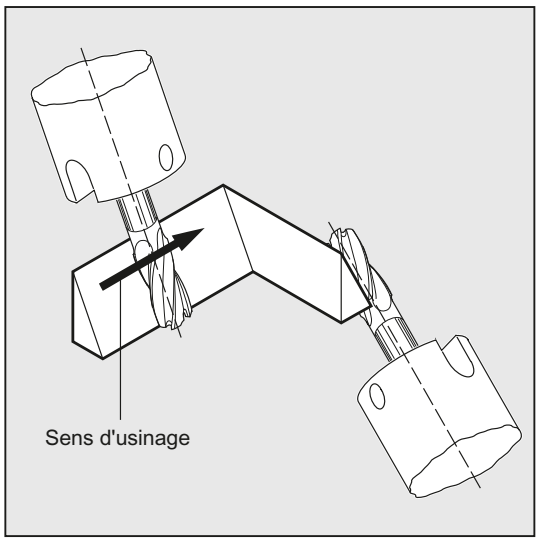
7.5.5 Correction d'outil 3D : angles rentrants/angles saillants et méthode du point d'intersection (G450/G451)

Fonction

Angles rentrants/angles saillants

Les angles saillants et les angles rentrants sont traités séparément. L'appellation angle rentrant ou angle saillant dépend de l'orientation de l'outil.

En cas de changement d'orientation à un angle, il peut arriver que le type d'angle change pendant l'usinage. Dans ce cas, l'usinage est arrêté et un message d'erreur est émis.



Syntaxe

```
G450
G451
```

Signification

`G450`	Arc de raccordement (l'outil contourne les angles de la pièce sur une trajectoire circulaire)
`G451`	Point d'intersection des équidistantes (l'outil s'arrête aux angles de la pièce)

Autres informations

Méthode du point d'intersection pour la correction 3D

Dans le cas du fraisage périphérique 3D, les codes G `G450/G451` sont maintenant traités aux angles saillants, c.-à-d. que le point d'intersection des trajectoires décalées peut être accosté. Jusqu'à la version 4 du logiciel, un cercle était toujours inséré aux angles saillants. La méthode du point d'intersection disponible est particulièrement intéressante pour des programmes 3D générés par des systèmes de CAO. En effet, ces programmes sont souvent constitués de blocs linéaires courts (approximation de courbes lisses), les transitions entre blocs voisins étant pratiquement tangentielles.

En cas d'activation de la correction de rayon d'outil, des cercles étaient jusqu'à présent insérés pour contourner les angles saillants. Comme ces blocs deviennent très courts si les transitions sont pratiquement tangentielles, des baisses de vitesse indésirables se produisent.

Dans ces cas, les deux parties de trajectoires adjacentes sont prolongées de façon analogue à la correction de rayon 2D ½, et le point d'intersection des parties de trajectoires prolongées est accosté.

Le point d'intersection est déterminé en prolongeant les trajectoires décalées des deux blocs concernés dans le plan perpendiculaire à la direction de l'outil au niveau de l'angle. S'il n'existe pas de point d'intersection, l'angle est traité comme précédemment, c.-à-d. qu'un cercle est inséré.

Pour d'autres informations sur la méthode du point d'intersection, voir :
Bibliographie :
Description fonctionnelle Fonctions de base, Correction de rayon d'outil 3D (W5)

7.5.6 Correction d'outil 3D : fraisage périphérique 3D avec surfaces de délimitation

Adaptation du fraisage périphérique 3D aux particularités des programmes CAO

Les programmes CN générés par les systèmes CAO approchent généralement la trajectoire du centre d'un outil standard par une multitude de blocs linéaires courts. Pour que les blocs ainsi générés pour de nombreux contours de pièce reproduisent le plus exactement possible le contour original, il est nécessaire de procéder à certaines adaptations dans le programme pièce.

D'importantes informations qui seraient nécessaires pour une correction optimale, mais qui ne sont plus disponibles dans le programme pièce, doivent être remplacées par des mesures adéquates. Les méthodes typiques utilisées pour la compensation des transitions critiques, soit directement dans le programme pièce ou lors de la détermination du contour réel (par exemple par déplacement de l'outil), sont présentées ci-après.

Applications

En plus des applications typiques dans lesquelles un outil réel décrit la trajectoire du centre au lieu de l'outil standard, les outils cylindriques avec une correction d'outil 3D sont également présentés. Dans ce cas, la trajectoire programmée se rapporte au contour de la surface d'usinage. La surface de délimitation correspondante dépend de l'outil. Comme pour la correction de rayon d'outil conventionnelle, on tient compte du rayon total pour le calcul du décalage vertical par rapport à la surface de délimitation.

7.5.7 Correction d'outil 3D : prise en compte d'une surface de délimitation (CUT3DCC, CUT3DCCD)

Fonction

fraisage périphérique 3D avec outils réels

Dans le cas du fraisage périphérique 3D avec modification continue ou constante de l'orientation de l'outil, la trajectoire du centre de l'outil est souvent programmée pour un outil standard défini. Compte tenu que dans bien des cas réels, on ne dispose pas des outils standard adéquats, on peut utiliser un outil ne différant pas trop sensiblement d'un outil standard.

Avec `CUT3DCCD`, une surface de délimitation, qui décrirait l'outil standard programmé, est prise en compte pour un outil différentiel réel. Le programme CN décrit la trajectoire du centre d'un outil standard.

Avec `CUT3DCC`, une surface de délimitation, qui aurait atteint l'outil standard programmé, est prise en compte dans le cas de l'utilisation d'outils cylindriques. Le programme CN décrit le contour de la surface d'usinage.

Syntaxe

```
CUT3DCCD
CUT3DCC
```

Signification

`CUT3DCCD`	Activation de la correction d'outil 3D pour fraisage périphérique avec outil différentiel sur la trajectoire du centre de l'outil : déplacement perpendiculaire jusqu'à la surface de délimitation.
`CUT3DCC`	Activation de la correction d'outil 3D pour fraisage périphérique avec surfaces de délimitation et correction de rayon 3D : contour de la surface d'usinage

Remarque

Correction de rayon d'outil avec G41, G42

Pour la correction de rayon d'outil avec `G41`, `G42` dans le cas où `CUT3DCCD` ou `CUT3DCC` sont actifs, l'option "Transformation d'orientation" doit être activée.

Outils standard avec arrondi

L'arrondi de l'outil standard est défini par le paramètre d'outil `$TC_DP7`. L'écart de l'arrondi de l'outil réel par rapport à l'outil standard est défini par le paramètre d'outil `$TC_DP16`.

Exemple

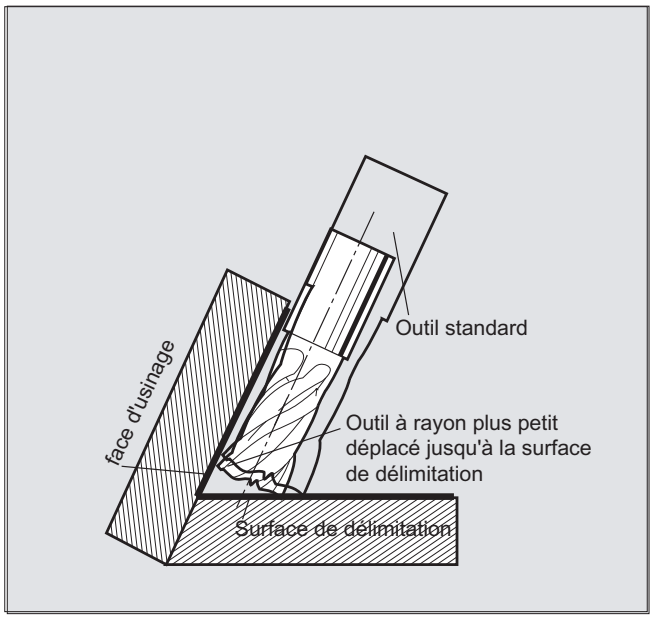
Dimensions d'outil pour une fraise convexe demi-cercle avec un rayon plus petit par rapport à l'outil standard.

Type d'outil	R = rayon de tige	r = rayon de congé
Outil standard avec arrondi	R = $TC_DP6	r = $TC_DP7
Outil réel avec arrondi : types d'outil 121 et 131 fraises convexes demi-cercle (fraise cylindrique)	R' = $TC_DP6 + $TC_DP15 + OFFN	r' = $TC_DP7 + $TC_DP16
Dans cet exemple, $TC_DP15 + OFFN ainsi que $TC_DP16 sont négatifs. le type d'outil ($TC_DP1) est exploité.		
Sont uniquement autorisés les types de fraise avec un corps cylindrique (fraise cylindrique) ainsi que des fraises convexes demi-cercle (types 121 et 131) et, dans un cas limite, la fraise à bout hémisphérique (type 110).	Pour ces types de fraise autorisés, le rayon de congé r est égal au rayon de tige R. Tous les autres types d'outil autorisés sont interprétés comme étant des fraises cylindriques et la cote éventuellement indiquée pour l'arrondi n'est pas prise en compte.	
Sont autorisés tous les types d'outil portant les numéros 1 à 399 à l'exception des numéros 111 et 155 à 157.		

Autres informations

Trajectoire du centre de l'outil avec déplacement jusqu'à la surface de délimitation CUT3DCCD

En cas d'utilisation d'un outil avec un rayon plus petit que celui de l'outil standard correspondant, une fraise pénétrant dans le sens longitudinal avance jusqu'à ce qu'elle touche de nouveau le fond de la poche. Ainsi l'angle formé par la surface d'usinage et la surface de délimitation est évidé autant que l'outil le permet. Il s'agit d'un mode d'usinage mixte par fraisage périphérique et par fraisage en bout. De manière analogue à l'outil avec un rayon réduit, le déplacement s'effectue dans la direction inverse dans le cas d'un outil avec un rayon plus grand.



Par rapport à toutes les autres corrections d'outil du groupe de codes G 22, un paramètre d'outil `$TC_DP6` spécifié pour `CUT3DCCD` est sans signification pour le rayon d'outil et n'influence pas la correction résultante.

Le décalage de la correction résulte de la somme des valeurs suivantes :

- Valeur d'usure du rayon d'outil (paramètre d'outil `$TC_DP15`)
- Décalage d'outil `OFFN` programmé pour le calcul du décalage perpendiculaire à la surface de délimitation.

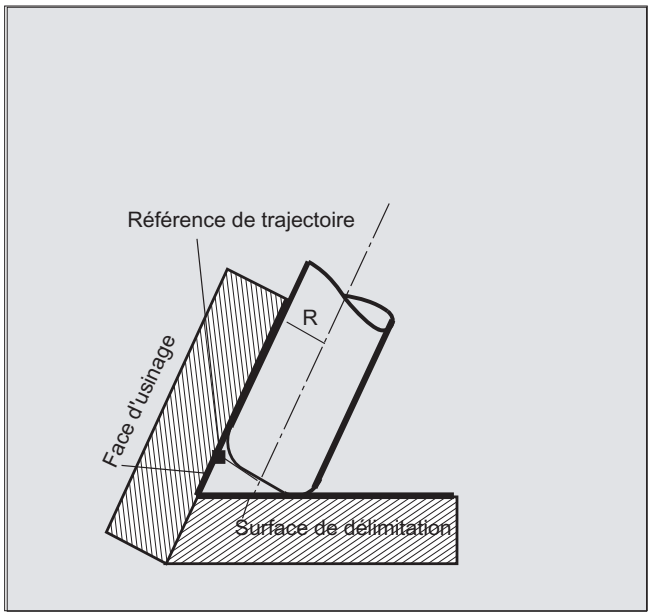
Le programme pièce généré ne permet pas de savoir si la surface usinée se situe à gauche ou à droite de la trajectoire. On part donc d'un rayon positif et d'une valeur d'usure négative de l'outil original. Une valeur d'usure négative décrit toujours un outil de diamètre réduit.

Utilisation d'outils cylindriques

En cas d'utilisation d'outils cylindriques, un déplacement est uniquement nécessaire si la surface d'usinage et la surface de délimitation forment un angle aigu (moins de 90 degrés). L'utilisation de fraises convexes demi-cercle exige un déplacement de l'outil dans le sens longitudinal aussi bien en présence d'un angle aigu que d'un angle obtus.

Correction de rayon 3D avec CUT3DCC, contour de la surface d'usinage

Si CUT3DCC est actif avec une fraise convexe demi-cercle, la trajectoire programmée se rapporte à une fraise cylindrique fictive de diamètre identique. La référence de trajectoire qui en résulte est représentée sur la figure suivante pour l'utilisation d'une fraise convexe demi-cercle.



L'angle formé par la surface d'usinage et la surface de délimitation peut également passer d'un angle aigu à un angle obtus, et vice versa, à l'intérieur d'un bloc.

L'outil utilisé réellement peut être aussi bien plus grand que plus petit que l'outil standard. Cependant le rayon d'arrondi résultant ne doit pas devenir négatif et le signe du rayon d'outil résultant doit rester inchangé.

Avec CUT3DCC, le programme pièce CN se rapporte au contour de la surface d'usinage. Comme pour la correction de rayon d'outil conventionnelle, on tient compte du rayon total qui correspond à la somme des valeurs suivantes :

- Rayon d'outil (paramètre d'outil $TC_DP6)
- Valeur d'usure (paramètre d'outil $TC_DP15)
- Décalage d'outil OFFN programmé pour le calcul du décalage perpendiculaire à la surface de délimitation.

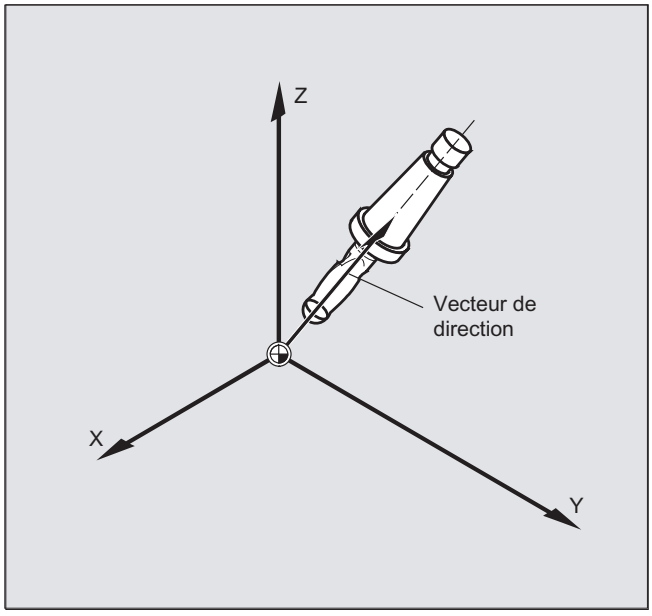
La position de la surface de délimitation est déterminée par la différence des deux valeurs suivantes :

- Dimensions de l'outil standard
- Rayon d'outil (paramètre d'outil $TC_DP6)

7.6 Orientation de l'outil (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST)

Fonction

On entend par orientation de l'outil l'orientation géométrique de l'outil dans l'espace. Sur une machine-outil à cinq axes, l'orientation de l'outil peut être spécifiée par des instructions du programme.



Des déplacements d'arrondissement de l'orientation activés par OSD et OST sont formés différemment selon le type d'interpolation pour l'orientation d'outil.

En cas d'interpolation vectorielle active, le tracé lissé de l'orientation est également interpolé par l'interpolation vectorielle. Par contre, en cas d'interpolation d'axe rotatif active, l'orientation est directement lissée par des déplacements d'axe rotatif.

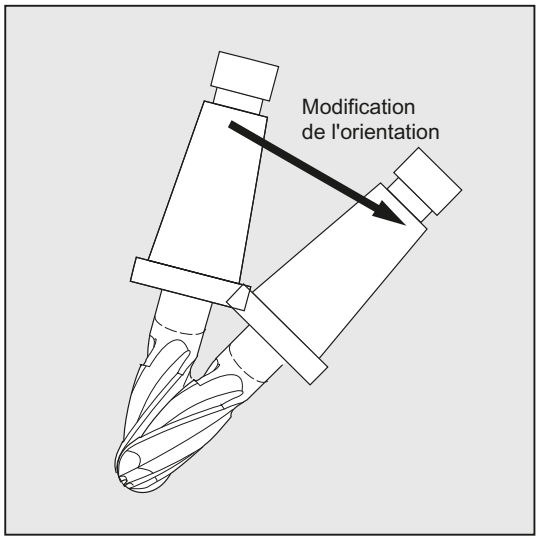
Programmation

Programmation du changement d'orientation :

On peut programmer un changement d'orientation de l'outil par :

- programmation directe des axes rotatifs A, B, C (interpolation d'axe rotatif)
- Angles d'Euler ou RPY
- Vecteur de direction (interpolation vectorielle par indication de A3 ou de B3 ou de C3)
- LEAD/TILT (fraisage en bout)

Le système de coordonnées de référence est soit le { système de coordonnées machine (ORIMKS), soit le système de coordonnées pièce courant (ORIWKS).



Programmation de l'orientation de l'outil :

Instruction	Signification
ORIC :	orientation et déplacement le long de la trajectoire en parallèle
ORID :	orientation et déplacement le long de la trajectoire successifs
OSOF :	aucun lissage de l'orientation
OSC :	orientation constante
OSS :	lissage de l'orientation seulement en début du bloc
OSSE :	lissage de l'orientation en début et en fin de bloc
ORIS :	Vitesse, en degrés par mm, du changement d'orientation lors de l'activation du lissage de l'orientation (s'applique à OSS et OSSE)
OSD :	Arrondissement de l'orientation par prescription de la longueur d'arrondissement avec la donnée de réglage : SD42674 $SC_ORI_SMOOTH_DIST
OST :	Arrondissement de l'orientation par prescription de la tolérance d'angle en degrés en cas d'interpolation vectorielle avec la donnée de réglage : SD42676 $SC_ORI_SMOOTH_TOL En cas d'interpolation d'axe rotatif, la tolérance prescrite est prise comme étant l'écart maximal des axes d'orientation.

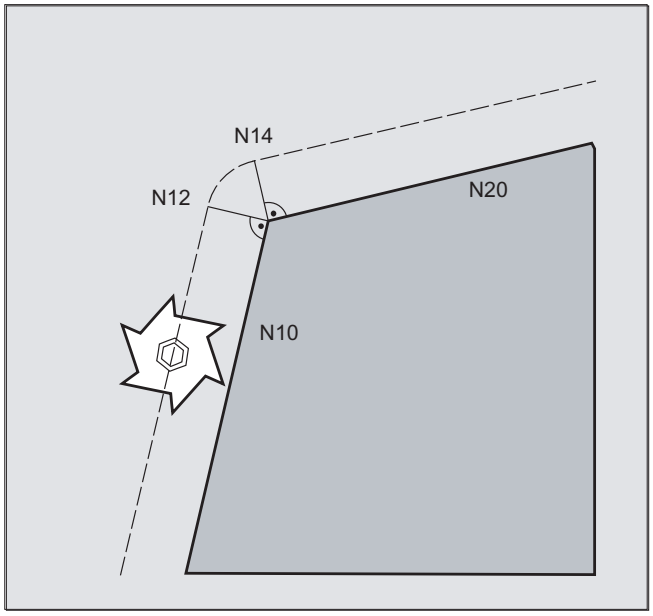
Remarque

Toutes les instructions réalisant un arrondissement de l'orientation de l'outil (OSOF, OSC, OSS, OSSE, OSD et OST) sont regroupées dans le groupe de fonctions G34. Elles sont à effet modal, autrement dit, une seule de ces instructions peut être exécutée à la fois.

Exemples

Exemple 1 : ORIC

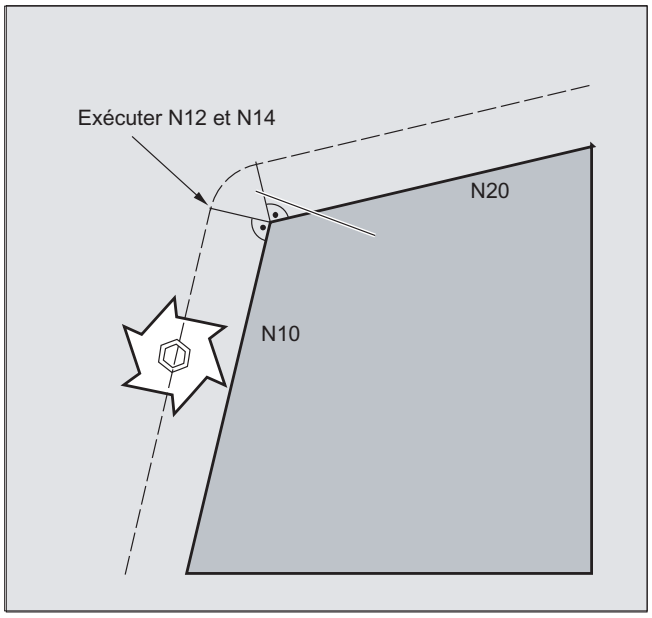
Si, entre les blocs de déplacement `N10` et `N20`, deux ou plusieurs blocs sont programmés avec des changements d'orientation (p. ex. `A2=... B2=... C2=...`) et si `ORIC` est actif, le bloc à interpolation circulaire inséré est réparti sur ces deux blocs intermédiaires conformément à la valeur des changements d'angle.



Code de programme	**Commentaire**
`ORIC`	
`N8 A2=… B2=… C2=…`	
`N10 X… Y… Z…`	
`N12 C2=… B2=…`	`; Le bloc à interpolation circulaire qui est inséré à`
`N14 C2=… B2=…`	`l'angle saillant est réparti sur N12 et N14 en fonction du changement d'orientation. Le mouvement circulaire et le changement d'orientation sont exécutés en parallèle.`
`N20 X =…Y=… Z=… G1 F200`	

Exemple 2 : ORID

Si `ORID` est activée, tous les blocs situés entre les deux blocs de déplacement sont exécutés à la fin du premier bloc de déplacement. Le bloc à interpolation circulaire avec orientation constante est exécuté directement avant le deuxième bloc de déplacement.



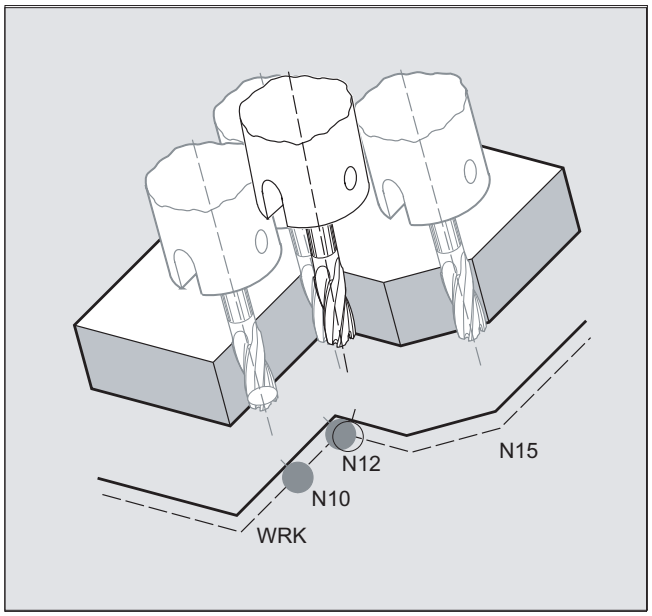
Code de programme	Commentaire
ORID	
N8 A2=… B2=… C2=…	
N10 X… Y… Z…	
N12 A2=… B2=… C2=…	; Le bloc N12 et N14 est exécuté à la fin de N10. Puis le bloc à interpolation circulaire se retire avec l'orientation courante.
N14 M20	; Fonctions auxiliaires, etc.
N20 X… Y… Z…	

Remarque

L'instruction de programme qui est active dans le premier bloc de déplacement d'un angle saillant est déterminante pour le type de changement d'orientation à cet angle saillant.

Sans changement d'orientation : Si l'orientation n'est pas modifiée à la fin du bloc, la section de l'outil est un cercle en contact avec les deux contours.

Exemple 3 : Changement d'orientation à un angle rentrant



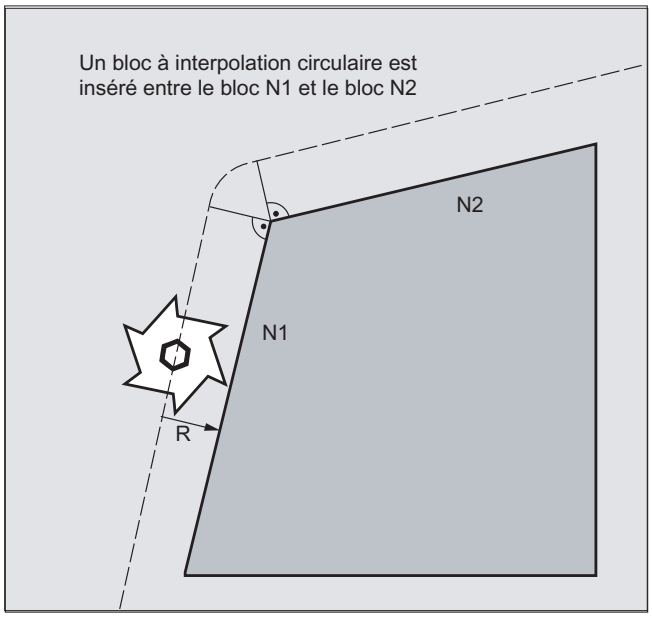
Code de programme
ORIC
N10 X …Y… Z… G1 F500
N12 X …Y… Z… A2=… B2=… C2=…
N15 X …Y… Z… A2=… B2=… C2=…

Informations complémentaires

Comportement aux angles saillants

Un bloc à interpolation circulaire avec un rayon égal à celui de la fraise est toujours inséré aux angles saillants.

Les instructions de programme `ORIC` et `ORID` permettent de définir si les changements d'orientation qui ont été programmés entre le bloc `N1` et le bloc `N2` seront effectués avant le début du bloc à interpolation circulaire inséré ou en même temps que celui-ci.



Si un changement d'orientation est nécessaire aux angles saillants, celui-ci peut s'effectuer parallèlement à l'interpolation ou bien séparément, en même temps que le déplacement le long de la trajectoire.

Avec ORID, les blocs insérés sont exécutés d'abord sans déplacement le long de la trajectoire. Le bloc de raccordement par interpolation circulaire est inséré directement avant le second des deux blocs de déplacement qui forment l'angle.

Si plusieurs blocs d'orientation sont insérés à un angle saillant et si ORIC est actif, le mouvement circulaire est réparti sur les différents blocs insérés, conformément aux valeurs des changements d'orientation de ces blocs.

Arrondissement de l'orientation avec OSD ou OST

Lors de l'arrondissement avec G642, l'écart maximal pour les axes de contour et les axes d'orientation ne peut pas varier beaucoup. La tolérance, la plus petite des deux, détermine la forme du déplacement d'arrondissement et de la tolérance d'angle et permet de lisser assez fortement le tracé de l'orientation sans écarts de contour importants.

L'activation de OSD ou OST permet, avec une longueur d'arrondissement ou une tolérance d'angle prescrites, de lisser "généreusement" de très faibles écarts du tracé de l'orientation sans écarts de contour importants.

Remarque

A la différence de l'arrondissement du contour (et du tracé de l'orientation) avec G642, aucun bloc séparé n'est formé lors de l'arrondissement de l'orientation avec OSD ou OST, mais le déplacement d'arrondissement est directement introduit dans les blocs d'origine programmés.

Avec OSD ou OST, aucune transition entre blocs ne peut être arrondie pour lesquels un changement du type d'interpolation pour l'orientation de l'outil (vecteur → axe rotatif, axe rotatif → vecteur) a lieu. Ces transitions entre blocs peuvent éventuellement être arrondies avec les fonctions d'arrondissement conventionnelles G641, G642 ou G643.

7.7 Numéros D libres, numéro de tranchant

7.7.1 Numéros D libres, numéro de tranchant (adresse CE)

Numéro D

Les numéros D peuvent être utilisés en tant que numéros de correcteur. En outre, CE permet l'adressage du numéro de tranchant. L'écriture du numéro de tranchant est possible à l'aide de la variable système $TC_DPCE.

Préréglage : n° correction == n° tranchant

Le nombre maximal de numéros D (numéros de tranchant) et le nombre maximal de tranchants par outil sont définis dans les paramètres machine (→ constructeur de la machine-outil). Les instructions suivantes sont recommandées uniquement lorsque le nombre maximal de numéros de tranchant (paramètre machine MD18105) est plus grand que le nombre de tranchants par outil (MD18106). Veuillez observer les indications du constructeur de machines.

Remarque

Outre les numéros D relatifs, vous pouvez également attribuer des numéros D "absolus" (1-32000) sans référence à un numéro T (avec la fonction "structure horizontale des numéros D").

Bibliographie

Description fonctionnelle Fonctions de base, Correction d'outil (W1)

7.7.2 Libre affectation des numéros D : contrôle des numéros D (CHKDNO)

Fonction

L'instruction CKKDNO permet un contrôle d'univocité de l'attribution des numéros D. Les numéros D de tous les outils définis dans une unité TO ne peuvent être attribués qu'une fois. La vérification ne porte pas sur les outils de rechange.

Syntaxe

```
state=CHKDNO(Tno1,Tno2,Dno)
```

Signification

`state`	= TRUE :	Les numéros D ont été attribués de façon univoque dans le domaine contrôlé.
	= FALSE :	Une collision entre des numéros D a été constatée ou le paramétrage est incorrect. Tno1, Tno2 et Dno sont les paramètres ayant conduit à la collision. Ces données peuvent être exploitées dans le programme pièce.
`CHKDNO (Tno1,Tno2)`	Tous les numéros D des outils indiqués sont comparés.	
`CHKDNO(Tno1)`	Tous les numéros D de l'outil Tno1 sont comparés à tous les numéros D des autres outils.	
`CHKDNO`	Tous les numéros D de tous les outils sont comparés à tous les numéros D de tous les autres outils.	

7.7.3 Libre affectation des numéros D : modification des numéros D (GETDNO, SETDNO)

Fonction

L'attribution des numéros D doit être univoque. Deux tranchants distincts d'un outil ne peuvent pas posséder le même numéro D.

GETDNO

Cette instruction fournit le numéro D d'un certain tranchant (ce) d'un outil avec le numéro T t. S'il n'existe pas de numéro D correspondant aux paramètres indiqués, la valeur d=0 est attribuée. Si le numéro D n'est pas valable, d contient une valeur supérieure à 32000.

SETDNO

Avec cette instruction, vous attribuez la valeur d au numéro D du tranchant ce de l'outil t. Le paramètre state fournit le résultat de l'instruction (TRUE ou FALSE). S'il n'existe pas d'enregistrement correspondant aux paramètres indiqués, le résultat de l'instruction est FALSE. Les erreurs de syntaxe génèrent une alarme. La valeur 0 ne peut pas être attribuée explicitement à un numéro D.

Syntaxe

```
d = GETDNO (t,ce)
state = SETDNO (t,ce,d)
```

Signification

d	Numéro D du tranchant de l'outil
t	Numéro T de l'outil
ce	Numéro du tranchant (numéro CE) de l'outil
state	Indique si l'instruction a pu être exécutée sans erreur (TRUE ou FALSE).

Exemple de changement de nom d'un numéro D

Programmation	Commentaire
$TC_DP2[1,2] = 120	;
$TC_DP3[1,2] = 5.5	;
$TC_DPCE[1,2] = 3	; Numéro de tranchant CE
...	;
N10 def int N°DAncien, N°DNouveau = 17	;
N20 N°DAncien = GETDNO(1,3)	;
N30 SETDNO(1,3,N°DNouveau)	;

Le nouveau correcteur D17 est affecté au tranchant CE=3. L'accès aux données de ce tranchant a dorénavant lieu avec le numéro D17 ; ceci concerne les paramètres système aussi bien que la programmation avec l'adresse CN.

7.7.4 Libre affectation des numéros D : détermination du numéro T correspondant au numéro D prescrit (GETACTTD)

Fonction

Avec l'instruction GETACTTD, vous déterminez le numéro T correspondant à un numéro D absolu. Il n'y a pas de contrôle d'univocité. En présence de plusieurs numéros D identiques dans une unité TO, l'instruction fournit le numéro T du premier outil trouvé. En cas d'utilisation de la "structure horizontale des numéros D", l'instruction ne présente pas d'intérêt, car son résultat est toujours "1" (pas de numéro T dans la gestion des données).

Syntaxe

```
Etat=GETACTTD(Tnr,Dnr)
```

Signification

n°D	Numéro D pour lequel le numéro T doit être recherché.	
Tnr	Numéro T trouvé	
état	Valeur :	Signification :
	0	Le numéro T a été trouvé. Le paramètre n° T contient sa valeur.
	-1	Il n'existe pas de numéro T pour le numéro D indiqué ; n° T=0.
	-2	Le numéro D n'est pas univoque. Le paramètre n° T contient la valeur du premier outil trouvé comportant le numéro D dont la valeur est n° D.
	-5	La fonction n'a pas pu être exécutée pour une autre raison.

7.7.5 Libre affectation des numéros D : numéros D déclarés non valides (DZERO)

Fonction

L'instruction DZERO sert à faciliter le réoutillage. Les correcteurs déclarés non valides ne font plus l'objet d'un contrôle par l'instruction CHKDNO. Pour avoir de nouveau accès aux correcteurs, il faut procéder à une nouvelle affectation des numéros D avec SETDNO.

Syntaxe

```
DZERO
```

Signification

DZERO	Déclare tous les numéros D de l'unité TO non valides.

7.8 Cinématique d'un organe porte-outil

Conditions

Un organe porte-outil ne peut orienter un outil dans toutes les directions de l'espace que

- si deux axes rotatifs v_1 et v_2 sont présents.
- si les axes rotatifs sont perpendiculaires les uns par rapport aux autres.
- si l'axe longitudinal de l'outil est perpendiculaire au deuxième axe rotatif v_2.

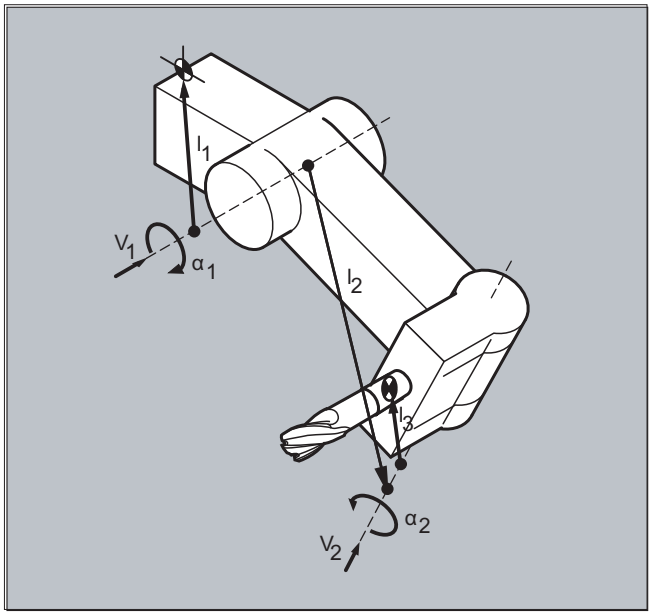
En outre, la condition ci-dessous doit être satisfaite dans le cas des machines sur lesquelles il doit être possible de régler toutes les orientations :

- - l'orientation de l'outil doit être perpendiculaire au premier axe rotatif v_1.

Fonction

La cinématique d'un organe porte-outil ayant deux axes rotatifs au maximum v_1 ou v_2 est décrite à l'aide des 17 variables système allant de $TC_CARR1[m] à $TC_CARR17[m]. Cette description comprend :

- le vecteur distance entre le premier axe rotatif et le point de référence de l'organe porte-outil I_1, le vecteur distance entre le premier et le second axe rotatif I_2, le vecteur distance entre le second axe rotatif et le point de référence de l'outil I_3.
- le vecteur de direction des deux axes rotatifs v_1, v_2.
- les angles rotatifs α_1, $\alpha_2$autour des deux axes. Les angles rotatifs sont positifs s'ils provoquent une rotation en sens horaire pour un observateur regardant dans la direction des axes rotatifs.



Pour les machines à **cinématique décomposée** (l'outil et la pièce sont orientables), les variables système supplémentaires

- $TC_CARR18[m] bis $TC_CARR23[m] ont été définies.

Paramètres

Fonction des variables système pour organes porte-outil orientables			
Désignation	**Composante x**	**Composante y**	**Composante z**
Vecteur offset l_1	$TC_CARR1[m]	$TC_CARR2[m]	$TC_CARR3[m]
Vecteur offset l_2	$TC_CARR4[m]	$TC_CARR5[m]	$TC_CARR6[m]
Axe rotatif v_1	$TC_CARR7[m]	$TC_CARR8[m]	$TC_CARR9[m]
Axe rotatif v_2	$TC_CARR10[m]	$TC_CARR11[m]	$TC_CARR12[m]
Angle rotatif α_1 Angle rotatif α_2	$TC_CARR13[m] $TC_CARR14[m]		
Vecteur offset l_3	$TC_CARR15[m]	$TC_CARR16[m]	$TC_CARR17[m]

Extension des variables système pour organes porte-outil orientables			
Désignation	**Composante x**	**Composante y**	**Composante z**
Vecteur offset l_4	$TC_CARR18[m]	$TC_CARR19[m]	$TC_CARR20[m]
Descripteur d'axe Axe rotatif v_1 Axe rotatif v_2	Descripteur d'axe pour les axes rotatifs $v_1$et v_2 (préréglage avec zéro) $TC_CARR21[m] $TC_CARR22[m]		
Type de cinématique **T**ool **P**art **M**ixed mode	$TC_CARR23[m]		
	Type de cinématique T ->	Type de cinématique P ->	Type de cinématique M
	seul l'outil est orientable (préréglage).	seule la pièce est orientable.	pièce et outil sont orientables.
Décalage des axes rotatifs v_1 et v_2	Angle en degrés des axes rotatifs $v_1$et v_2 en position initiale $TC_CARR24[m] $TC_CARR25[m]		
Décalage angulaire des axes rotatifs v_1 et v_2	Décalage en degrés de la denture Hirth des axes rotatifs $v_1$et v_2 $TC_CARR26[m] $TC_CARR27[m]		
Incrément angulaire v_1 Axe rotatif v_2 Axe rotatif	Incrément en degrés de la denture Hirth des axes rotatifs $v_1$et v_2 $TC_CARR28[m] $TC_CARR29[m]		
Position min. Axe rotatif v_1 Axe de rotatif v_2	Limite logicielle pour position minimale des axes rotatifs $v_1$et v_2 $TC_CARR30[m] $TC_CARR31[m]		
Position max. Axe rotatif v_1 Axe rotatif v_2	Limite logicielle pour position maximale des axes rotatifs $v_1$et v_2 $TC_CARR32[m] $TC_CARR33[m]		
Nom de l'organe porte-outil	À la place d'un nombre, un organe porte-outil peut se voir affecter un nom. $TC_CARR34[m]		
Utilisateur : Nom d'axe 1 Nom d'axe 2 Identificateur	Utilisation prévue par l'utilisateur à l'intérieur des cycles de mesure. $TC_CARR35[m] $TC_CARR36[m] $TC_CARR37[m]		
Position	$TC_CARR38[m]	$TC_CARR39[m]	$TC_CARR40[m]

Extension des variables système pour organes porte-outil orientables			
Décalage fin	Paramètres pouvant être ajoutés **aux valeurs contenues dans les paramètres de base**.		
Vecteur offset l_1	$TC_CARR41[m]	$TC_CARR42[m]	$TC_CARR43[m]
Vecteur offset l_2	$TC_CARR44[m]	$TC_CARR45[m]	$TC_CARR46[m]
Vecteur offset l_3	$TC_CARR55[m]	$TC_CARR56[m]	$TC_CARR57[m]
Vecteur offset l_4	$TC_CARR58[m]	$TC_CARR59[m]	$TC_CARR60[m]
Axe rotatif v_1	$TC_CARR64[m]		
Axe rotatif v_2	$TC_CARR65[m]		

Remarque

Explication des paramètres

"m" est le numéro de l'organe porte-outil à décrire.

Les paramètres `$TC_CARR47` jusqu'à `$TC_CARR54` ainsi que `$TC_CARR61` jusqu'à `$TC_CARR63` ne sont pas définis et génèrent une alarme en cas de tentative d'accès en lecture ou en écriture.

Vous pouvez choisir librement les points initiaux et finaux des vecteurs distances sur les axes. Les angles rotatifs α_1, $\alpha_2$autour des deux axes sont égaux à 0° lorsque l'organe porte-outil est en position initiale. Il existe donc un grand nombre de possibilités pour décrire la cinématique d'un organe porte-outil.

Vous pouvez décrire des organes porte-outil n'ayant qu'un axe rotatif ou aucun axe rotatif en annulant les composantes du vecteur de direction d'un ou des deux axes rotatifs.
Dans le cas d'un organe porte-outil sans axe rotatif, les vecteurs distances ont le même effet que des corrections d'outil supplémentaires, dont les composantes ne sont pas influencées par le basculement entre les plans d'usinage (`G17` à `G19`).

Extension des paramètres

Paramètres des axes rotatifs

Les variables système $TC_CARR24[m] à $TC_CARR33[m], dont les significations sont indiquées ci-dessous, sont introduites :

Décalage des axes rotatifs $v_1$1 et v_2	Changement de position de l'axe rotatif v_1 ou v_2 lorsque l'organe porte-outil orientable est en position initiale.
Décalage **angulaire/incrément** angulaire des axes rotatifs $v_1$et v_2	Décalage ou incrément angulaire de la denture Hirth des axes rotatifs v_1 et v_2. L'angle programmé ou calculé est arrondi à la valeur la plus proche qui s'obtient à l'aide de la formule phi = s + n * d, n étant un nombre entier.
Position minimale et maximale Axes rotatifs v_1, v_2	Position minimale/maximale des axes rotatifs Angles limites (limite logicielle) des axes rotatifs v1 et v2.

Paramètres pour l'utilisateur

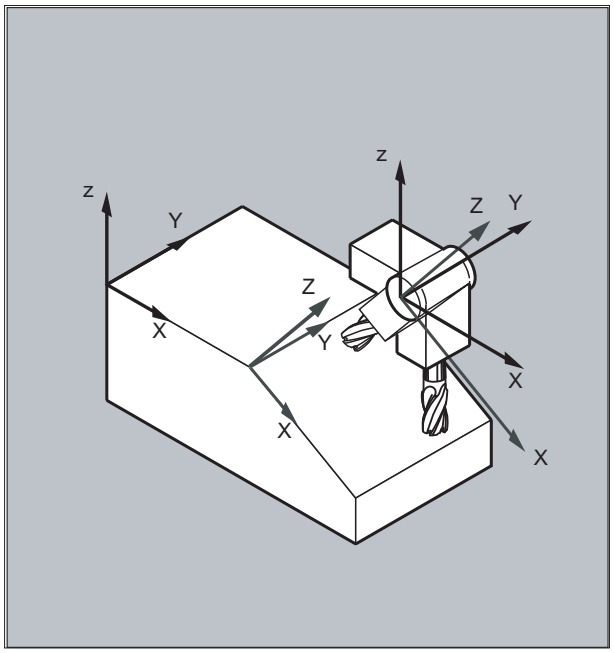
$TC_CARR34 à $TC_CARR40 contiennent des paramètres qui sont mis à la libre disposition de l'utilisateur et qui, jusqu'à la version de logiciel 6.4, ne sont pas exploités en standard à l'intérieur du NCK ou qui sont sans signification.

Paramètres de décalage fin

$TC_CARR41 à $TC_CARR65 contiennent des paramètres de décalage fin pouvant être ajoutés aux valeurs contenues dans les paramètres de base. On obtient une valeur de décalage fin affectée à un paramètre de base lorsque la valeur 40 est ajoutée au numéro de paramètre.

Exemple

L'organe porte-outil utilisé dans l'exemple suivant peut être décrit entièrement par une rotation autour de l'axe Y.



Code de programme	Commentaire
N10 $TC_CARR8[1]=1	; Définition de la composante Y du premier axe rotatif de l'organe porte-outil 1.
N20 $TC_DP1[1,1]=120	; Définition d'une fraise cylindrique.
N30 $TC_DP3[1,1]=20	; Définition d'une fraise cylindrique d'une longueur de 20 mm.
N40 $TC_DP6[1,1]=5	; Définition d'une fraise cylindrique d'un rayon de 5 mm.
N50 ROT Y37	; Définition d'un frame avec rotation de 37° autour de l'axe Y.
N60 X0 Y0 Z0 F10000	; Accostage de la position de départ.
N70 G42 CUT2DF TCOFR TCARR=1 T1 D1 X10	; Réglage correction de rayon, correction de longueur d'outil dans le frame pivoté, sélection organe porte-outil 1, outil 1.

Code de programme	Commentaire
N80 X40	; Effectuer l'usinage sous un angle rotatif de 37°.
N90 Y40	
N100 X0	
N110 Y0	
N120 M30	

Autres informations

Cinématique décomposée

Pour les machines à cinématique décomposée (l'outil et la pièce sont orientables), les variables système supplémentaires `$TC_CARR18[m]` à `$TC_CARR23[m]` permettent de définir :

l'organe porte-pièce orientable à l'aide :

- du vecteur distance entre le second axe rotatif `V2` et le point de référence d'une table porte-pièce pivotante `I4` du troisième axe rotatif.

les axes rotatifs à l'aide :

- des deux descripteurs d'axes de canal associés aux axes rotatifs `V1` et `V2`, et dont les positions peuvent être interrogées pour la détermination de l'orientation.

le type de cinématique à l'aide d'une des valeurs T, P ou M :

- type de cinématique T : seul l'outil est orientable.
- type de cinématique P : seule la pièce est orientable.
- type de cinématique M : l'outil et la pièce sont orientables.

Effacement des données d'organe porte-outil

Avec `$TC_CARR1[0] = 0`, vous pouvez effacer les données de tous les organes porte-outil.

La valeur du paramètre pour le type de cinématique `$TC_CARR23[T] = T` doit être une des trois lettres majuscules ou minuscules (T, P, M) ; ce paramètre ne doit donc pas être effacé.

Modification des données d'organe porte-outil

Vous pouvez modifier chacune des valeurs en lui affectant une nouvelle valeur dans le programme pièce. Tout autre caractère que T, P ou M conduit à une alarme lors de la tentative d'activer l'organe porte-outil orientable.

Lecture des données d'organe porte-outil

Vous pouvez lire chacune des valeurs en l'affectant à une variable dans le programme pièce.

Décalage fin

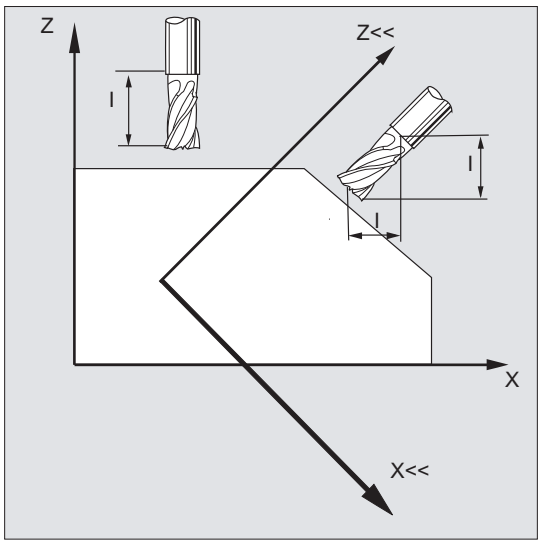
Une valeur de décalage fin non autorisée n'est détectée que lorsqu'un organe porte-outil orientable contenant une telle valeur est activé et que, dans le même temps, la donnée de réglageSD42974 $SC_TOCARR_FINE_CORRECTION = TRUE.

La valeur absolue du décalage fin autorisée est limitée par les paramètres machine à une valeur maximale admissible.

7.9 Correction de longueur d'outil pour organes porte-outils orientables (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ)

Fonction

Les composantes de longueur d'outil varient lorsque l'orientation dans l'espace de l'outil en question change.



Par conséquent, après un changement, par réglage manuel ou remplacement de l'organe porte-outil à orientation spatiale fixe par exemple, les composantes de longueur d'outil sont à redéfinir. Ceci se fait avec les instructions TCOABS et TCOFR.

Avec un organe porte-outil orientable d'un frame actif, il est possible de déterminer, lors de la sélection de l'outil avec TCOFRZ, TCOFRY et TCOFRX, la direction vers laquelle l'outil doit pointer.

Syntaxe

```
TCARR=[<m>]
TCOABS
TCOFR
TCOFRZ
avec TCOFRY
avec TCOFRX
```

Signification

`TCARR=[<m>]` :	Appel de l'organe porte-outil ayant le numéro "m"
`TCOABS` :	Calculer les composantes de longueur d'outil à partir de l'orientation de l'organe porte-outil.
`TCOFR` :	Détermination des composantes de longueur d'outil à partir de l'orientation du frame actif.
`TCOFRZ` :	Organe porte-outil orientable du frame actif dont l'outil est pointé en direction Z
`TCOFRY` :	Organe porte-outil orientable du frame actif dont l'outil est pointé en direction Y
`TCOFRX` :	Organe porte-outil orientable du frame actif dont l'outil est pointé en direction X

Autres informations

Correction de longueur d'outil à partir de l'organe porte-outil (TCOABS)

`TCOABS` calcule la correction de longueur d'outil sur la base des angles d'orientation de l'organe porte-outil, qui sont rangés dans les variables système $TC_CARR13 et $TC_CARR14.

Pour la définition de la cinématique de l'organe porte-outil avec des variables système, voir "Cinématique de l'organe porte-outil (Page 434)".

Pour que la correction de longueur d'outil soit recalculée en cas de changement de frame, il convient de rappeler l'outil.

direction d'outil du frame actif

L'organe porte-outil orientable peut être réglé de telle façon que l'outil soit orienté dans les directions suivantes :

- dans la direction Z avec `TCOFR` ou `TCOFRZ`
- dans la direction Y avec `TCOFRY`
- dans la direction X avec `TCOFRX`

Toute commutation entre `TCOFR` et `TCOABS` provoque un recalcul de la correction de longueur d'outil.

Appel de l'organe porte-outil (TCARR)

Avec `TCARR`, vous appelez les données géométriques de l'organe porte-outil de numéro m (mémoire de correcteurs).

Avec m=0, vous désactivez l'organe porte-outil actif.

Les données géométriques de l'organe porte-outil ne deviennent opérantes qu'après l'appel de l'outil. L'outil sélectionné reste activé même après un changement d'organe porte-outil.

Les données géométriques de l'organe porte-outil peuvent aussi être définies dans le programme pièce, par le biais des variables système correspondantes.

Nouveau calcul de la correction de longueur d'outil (TCOABS) en cas de changement de frame

Pour que la correction de longueur d'outil soit recalculée en cas de changement de frame, il convient de rappeler l'outil.

Remarque

L'orientation de l'outil doit être adaptée manuellement au frame actif.

Lors du calcul de la correction de longueur d'outil, les angles de rotation de l'organe porte-outil font également l'objet d'un calcul. Etant donné que les organes porte-outil à deux axes de rotation ont généralement deux paires d'angles de rotation, avec lesquels on peut adapter l'orientation d'outil au frame actif, les valeurs des angles de rotation rangées dans les variables système doivent correspondre au moins approximativement aux angles de rotation réglés mécaniquement.

Remarque

Orientation de l'outil

La commande n'est pas en mesure de contrôler la possibilité de réglage sur la machine des angles de rotation calculés sur la base de l'orientation du frame.

Si les axes de rotation de l'organe porte-outil sont situés de telle façon qu'il soit impossible à l'outil de rallier l'orientation calculée sur la base de l'orientation du frame, une alarme est émise.

Combiner la correction fine d'outil et les fonctionnalités de la correction de longueur d'outil n'est pas autorisé pour les organes porte-outils orientables. Si vous tentez d'appeler les deux fonctions en même temps, un message d'erreur vous sera donné.

Avec `TOFRAME` il est possible de définir un frame en fonction de la direction dans laquelle est orienté l'organe porte-outil. Pour de plus amples informations, se reporter au chapitre "Frames".

Quand une transformation d'orientation est active (transformation 3, 4, 5 axes), vous pouvez appeler un organe porte-outil ayant une orientation s'écartant de la position zéro, sans déclencher une alarme.

Paramètres de transfert des cycles standard et cycles de mesure

Pour les paramètres de transfert des cycles standard et cycles de mesure, on applique les plages de valeurs bien définies.

Dans le cas des valeurs angulaires, la plage de valeurs est définie ainsi :

- Rotation autour du 1er axe géométrique : -180 degrés jusqu'à +180 degrés
- Rotation autour du 2ème axe géométrique : -90 degrés jusqu'à +90 degrés
- Rotation autour du 3ème axe géométrique : -180 degrés jusqu'à +180 degrés

Voir chapitre Frames, "Rotation programmable (ROT, AROT, RPL)".

Remarque

Lors du transfert de valeurs angulaires à un cycle standard ou de mesure, il convient

d'arrondir à zéro toute valeur inférieure à l'unité de calcul de la CN !

L'unité de calcul de la CN pour les positions angulaires est définie dans le paramètre machine :

PM10210 $MN_INT_INCR_PER_DEG

7.10 Correction de longueur d'outil en ligne (TOFFON, TOFFOF)

Fonction

La variable système $AA_TOFF[<n>] permet de corriger en temps réel et en trois dimensions les longueurs d'outil effectives en fonction des trois directions de l'outil.

Les trois descripteurs d'axes géométriques servent d'indices <n>. De cette façon, le nombre des directions de correction activées est déterminé par les axes géométriques qui sont actifs au même moment.

Toutes les corrections peuvent être actives simultanément.

La fonction de correction des longueurs d'outil en ligne est applicable dans :

- la transformation de l'orientation TRAORI
- l'organe porte-outil orientable TCARR

Remarque

La correction en ligne de longueur d'outil est une **option** qui doit être validée auparavant. Cette fonction n'a de sens qu'en liaison avec la transformation active de l'orientation ou avec un organe porte-outil orientable à l'état activé.

Syntaxe

```
TRAORI
TOFFON(<direction de correction>[,<valeur de
l'offset>])
WHEN TRUE DO $AA_TOFF[<sens de correction>]    ; Dans des actions synchrones.
...
TOFFOF(<sens de correction>)
```

Pour plus d'explications sur la programmation de la correction en ligne de la longueur d'outil dans des actions synchrones au déplacement, voir "Correction en ligne de la longueur d'outil ($AA_TOFF) (Page 596)".

Signification

`TOFFON` : **Activation** de la correction en ligne de la longueur d'outil

- `<sens de correction>` : Direction de l'outil (`X`, `Y`, `Z`) dans laquelle la correction en ligne de la longueur d'outil doit prendre effet.
- `<valeur de l'offset>` : Dès que la fonction est activée, une valeur offset peut être indiquée pour la direction dans laquelle doit s'effectuer la correction et cette valeur fait immédiatement l'objet d'un mouvement de correction.

`TOFFOF` : **Remise à zéro** de la correction en ligne de la longueur d'outil

Les valeurs de correction dans le sens de correction spécifié sont remises à zéro et un arrêt du prétraitement des blocs est déclenché.

Exemples

Exemple 1 : activation de la correction de longueur d'outil

Code de programme	Commentaire
MD21190 $MC_TOFF_MODE =1	; Accostage des valeurs absolues.
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	
N5 DEF REAL XOFFSET	
N10 TRAORI(1)	; Activation de la transformation.
N20 TOFFON(Z)	; Activation de la correction en ligne de la longueur d'outil pour la direction Z de l'outil.
N30 WHEN TRUE DO $AA_TOFF[Z]=10 G4 F5	; Interpolation d'une valeur de correction de longueur d'outil égale à 10 pour la direction Z de l'outil.
...	
N100 XOFFSET=$AA_TOFF_VAL[X]	; Affectation de la correction actuelle dans la direction X.
N120 TOFFON(X,-XOFFSET) G4 F5	; Remise à 0 de la correction de longueur d'outil dans la direction X de l'outil.

Exemple 2 : désactivation de la correction de longueur d'outil

Code de programme	Commentaire
N10 TRAORI(1)	; Activation de la transformation.
N20 TOFFON(X)	; Activation de la correction en ligne de la longueur d'outil pour la direction X de l'outil.
N30 WHEN TRUE DO $AA_TOFF[X]=10 G4 F5	; Interpolation d'une valeur de correction de longueur d'outil égale à 10 pour la direction X de l'outil.
...	
N80 TOFFOF(X)	; L'offset de position de la direction d'outil X est supprimé : ...$AA_TOFF[X]=0 Aucun axe n'est déplacé. L'offset de position est ajouté à la position actuelle dans le SCP en fonction de l'orientation actuelle.

Informations complémentaires

Préparation des blocs

Lors de la préparation des blocs au cours de leur prétraitement, l'offset de longueur d'outil qui agit dans l'exécution de blocs est pris en considération. Afin de pouvoir exploiter au mieux les vitesses d'axes maximales autorisées, il est nécessaire d'arrêter la préparation des blocs avec un arrêt du prétraitement des blocs `STOPRE` pendant la réalisation d'un offset d'outil.

Cet offset d'outil est connu au moment du prétraitement des blocs si les corrections de longueur d'outil n'ont pas été modifiées après le lancement du programme ou - quand il y a eu modification des corrections de longueur d'outil - si le nombre de blocs exécutés est supérieur au nombre de blocs que peut recevoir le tampon IPO entre le prétraitement des blocs et leur exécution.

Variable $AA_TOFF_PREP_DIFF

Le degré de différence entre la correction actuelllement effective dans l'interpolateur et la correction effective au moment de la préparation des blocs peut être interrogée dans la variable $AA_TOFF_PREP_DIFF[<n>].

Réglage des paramètres machine et des données de réglage

Les données système suivantes sont disponibles pour la correction en ligne de la longueur d'outil :

- MD20610 $MC_ADD_MOVE_ACCEL_RESERVE (réserve d'accélération pour déplacement forcé)
- MD21190 $MC_TOFF_MODE

  Le contenu de la variable système$AA_TOFF[<n>] fait l'objet d'un déplacement en absolu ou d'une intégration forcée.
- MD21194 $MC_TOFF_VELO (vitesse de la correction en ligne de la longueur d'outil)
- MD21196 $MC_TOFF_ACCEL (accélération de la correction en ligne de la longueur d'outil)
- Donnée de réglage pour la spécification de valeurs limites :
  SD42970 $SC_TOFF_LIMIT (limite supérieure de la valeur de correction de la longueur d'outil)

Bibliographie :

Description fonctionnelle Fonctions spéciales ; F2 : transformations multi-axes

7.11 Modification des données de tranchant des outils orientables (CUTMOD)

Fonction

La fonction "Modification des données de tranchant des outils orientables" permet de tenir compte de la modification des conditions géométriques, qui résulte de la rotation des outils (essentiellement les outils de tournage, mais aussi les outils de perçage et de fraisage) par rapport à la pièce usinée, pour la correction d'outil.

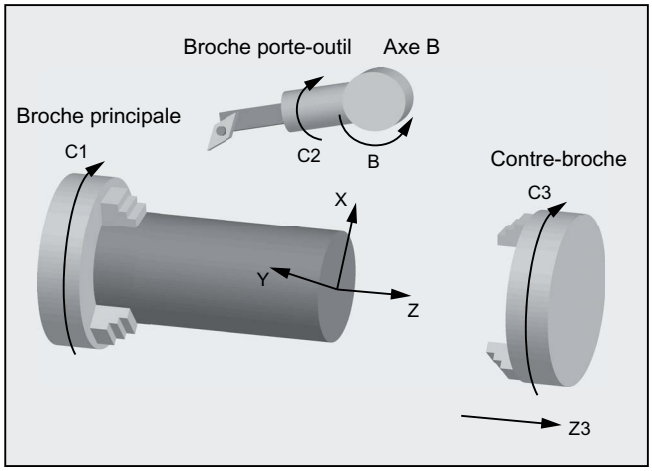


Figure 7-1 Outil orientable sur un tour

La rotation courante de l'outil est toujours déterminée à partir d'un organe porte-outil orientable, actif à l'instant (voir "Correction de longueur d'outil pour organes porte-outils orientables (Page 439)").

L'instruction `CUTMOD` active cette fonction.

Syntaxe

```
CUTMOD=<valeur>
```

Signification

CUTMOD Instruction d'activation de la fonction "Modification des données de tranchant des outils orientables"

<Valeur> Les valeurs suivantes peuvent être affectées à l'instruction CUTMOD :

0 La fonction est désactivée.

Les valeurs fournies par les variables système $P_AD... sont égales aux paramètres d'outil correspondants.

> 0 La fonction est activée si un organe porte-outil orientable portant le numéro indiqué est actif, ce qui signifie que l'activation est liée à un organe porte-outil orientable défini.

Par rapport aux paramètres d'outil correspondants, les valeurs fournies par les variables système $P_AD... sont modifiées le cas échéant en fonction de la rotation active.

La désactivation de l'organe porte-outil orientable désigné désactive la fonction temporairement et l'activation d'un autre organe porte-outil orientable la désactive durablement. Dans le premier cas, la fonction est donc de nouveau active si le même organe porte-outil orientable est réactivé. Dans le deuxième cas, il est nécessaire de l'activer à nouveau même si l'organe porte-outil orientable portant le numéro indiqué est réactivé ultérieurement.

La fonction n'est pas influencée par un reset.

-1 La fonction est toujours activée lorsqu'un organe porte-outil orientable est actif.

Lors d'un changement de l'organe porte-outil ou lorsque celui-ci est désactivé, puis réactivé ultérieurement, il n'est pas nécessaire de réactiver CUTMOD.

-2 La fonction est toujours activée lorsqu'un organe porte-outil orientable dont le numéro est identique à l'organe porte-outil orientable actif à l'instant est actif.

Si aucun organe porte-outil orientable n'est actif, cela équivaut à CUTMOD=0. Si un organe porte-outil orientable est actif, cela équivaut à l'indication directe du numéro de l'organe porte-outil courant.

< -2 Les valeurs inférieures à -2 sont ignorées, ce qui signifie que ce cas est traité comme si CUTMOD n'était pas programmé.

Remarque :
L'utilisation de cette plage de valeurs est déconseillée, car elle est réservée pour d'éventuelles extensions ultérieures.

Remarque

SD42984 $SC_CUTDIRMOD

La fonction qui est activable avec l'instruction CUTMOD remplace la fonction qui est activable au moyen de la donnée de réglage SD42984 $SC_CUTDIRMOD. Cependant la disponibilité de cette fonction reste inchangée. L'utilisation parallèle des deux fonctions étant cependant déconseillée, elle ne peut être activée que lorsque CUTMOD est égal à zéro.

Exemple

L'exemple suivant se rapporte à un outil avec la position de tranchant 3 et un organe porte-outil orientable pouvant tourner l'outil autour de l'axe B.

Les valeurs numériques des commentaires indiquent les positions de fin de bloc en coordonnées machine (SCM), dans l'ordre X, Y, Z.

```
Code de programme                                          Commentaire
N10 $TC_DP1[1,1]=500
N20 $TC_DP2[1,1]=3                                         ; Position du tranchant
N30 $TC_DP3[1,1]=12
N40 $TC_DP4[1,1]=1
N50 $TC_DP6[1,1]=6
N60 $TC_DP10[1,1]=110                                      ; Angle du porte-outil
N70 $TC_DP11[1,1]=3                                        ; Sens de coupe
N80 $TC_DP24[1,1]=25                                       ; Angle de dépouille

N90 $TC_CARR7[2]=0 $TC_CARR8[2]=1 $TC_CARR9[2]=0           ; Axe B
N100 $TC_CARR10[2]=0 $TC_CARR11[2]=0 $TC_CARR12[2]=1       ; Axe 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
```

Explications

Dans le bloc N180, l'outil est d'abord activé avec CUTMOD=0 et organe porte-outil orientable non pivoté. Tous les vecteurs de décalage de l'organe porte-outil étant 0, la position accostée correspond aux longueurs d'outil indiquées dans $TC_DP3[1,1] et $TC_DP4[1,1].

Dans le bloc N200, l'organe porte-outil orientable est activé avec une rotation de 30° autour de l'axe B. La position du tranchant n'étant pas modifiée compte tenu de CUTMOD=0, l'ancien point de référence du tranchant reste déterminant. C'est pourquoi la position qui garde l'ancien point de référence du tranchant à l'origine (ce qui signifie que le vecteur (1, 12) tourne de 30° dans le plan Z/X) est accostée dans le bloc N210.

Contrairement au bloc N200, CUTMOD=2 est actif dans le bloc N260. En raison de la rotation de l'organe porte-outil orientable, la position modifiée du tranchant est 8. Il en résulte également des positions d'axe différentes.

Dans les blocs N220 et N270, la correction de rayon d'outil est activée. La position du tranchant, qui diffère dans les deux sections de programme, n'influence pas les positions finales des blocs dans lesquels la correction de rayon d'outil est active. Les positions correspondantes sont donc identiques. Ce n'est que dans les blocs de désactivation N260 et N300 que les différentes positions du tranchant ont de nouveau un effet.

Autres informations

Prise d'effet des données de tranchant modifiées

La position modifiée du tranchant et le point de référence modifié du tranchant prennent effet immédiatement même s'ils sont programmés pour un outil déjà actif. A cet effet, il n'est pas nécessaire de réactiver l'outil.

Influence du plan de travail actif

Pour la détermination de la position modifiée du tranchant, du sens de coupe et de l'angle du porte-outil ou de l'angle de dépouille, le tranchant doit être considéré dans le plan actif correspondant (G17 - G19).

Si la donnée de réglage SD42940 $SC_TOOL_LENGTH_CONST (modification des composantes de longueur d'outil en cas de changement de plan) contient cependant une valeur valide différente de zéro (plus ou moins 17, 18 ou 19), son contenu détermine le plan dans lequel les grandeurs déterminantes sont considérées.

Variables système

Les variables système suivantes sont disponibles :

Variables système	Signification
$P_CUTMOD_ANG / $AC_CUTMOD_ANG	Fournit l'angle (non arrondi) du plan d'usinage actif, qui a été pris en compte pour la modification des données de tranchant (position du tranchant, sens de coupe, angle de dépouille et angle du porte-outil) pour les fonctions activées avec CUTMOD et $SC_CUTDIRMOD. $P_CUTMOD_ANG se rapporte à l'état courant du prétraitement et $AC_CUTMOD_ANG à l'exécution du bloc courant.
$P_CUTMOD / $AC_CUTMOD	Lecture de la valeur courante valide, qui a été programmée en dernier avec l'instruction CUTMOD (numéro de l'organe porte-outil pour lequel la modification des données de tranchant doit être activée). Si la dernière valeur CUTMOD programmée était = -2 (activation avec l'organe porte-outil orientable actif à l'instant), la valeur de retour de $P_CUTMOD n'est pas -2, mais le numéro de l'organe porte-outil orientable qui était actif au moment de la programmation. $P_CUTMOD se rapporte à l'état courant du prétraitement et $AC_CUTMOD à l'exécution du bloc courant.
$P_CUT_INV / $AC_CUT_INV	Livre la valeur TRUE lorsque l'outil est tourné de sorte que le sens de rotation de la broche doit être inversé. A cet effet, les quatre conditions suivantes doivent être remplies dans le bloc auquel se rapporte l'opération de lecture correspondante : 1. Un outil de tournage ou de rectification est actif (types d'outil 400 à 599 et/ou SD42950 $SC_TOOL_LENGTH_TYPE = 2). 2. La correction du tranchant a été activée avec l'instruction CUTMOD. 3. Un organe porte-outil orientable défini par la valeur numérique de CUTMOD est actif. 4. L'organe porte-outil orientable tourne l'outil autour d'un axe dans le plan de travail (normalement l'axe C) de sorte que la normale résultante du tranchant de l'outil tourne de plus de 90° (normalement 180°) par rapport à la position initiale. Si au moins une des conditions mentionnées n'est pas remplie, le contenu des variables est FALSE. Pour les outils dont la position du tranchant n'est pas définie, la valeur des variables est toujours FALSE. $P_CUT_INV se rapporte à l'état courant du prétraitement et $AC_CUT_INV à l'exécution du bloc courant.

Toutes les variables d'exécution ($AC_CUTMOD_ANG, $AC_CUTMOD et $AC_CUT_INV) peuvent être lues dans des actions synchrones. Un accès en lecture depuis le prétraitement entraîne un arrêt du prétraitement.

Données de tranchant modifiées :

Si une rotation d'outil est active, les données modifiées sont disponibles dans les variables système suivantes :

Variable système	Signification
$P_AD[2]	Position du tranchant
$P_AD[10]	Angle du porte-outil
$P_AD[11]	Sens de coupe
$P_AD[24]	Angle de dépouille

Remarque

Les données sont toujours modifiées par rapport aux paramètres d'outil correspondants ($TC_DP2[..., ...] etc.) lorsque la fonction "Modification des données de tranchant des outils orientables" a été activée avec l'instruction `CUTMOD` et qu'un organe porte-outil orientable entraînant une rotation de l'outil est actif.

Bibliographie

Pour d'autres informations sur la fonction "Modification des données de tranchant des outils orientables", voir :

Description fonctionnelle Fonctions de base, Correction d'outil (W1)

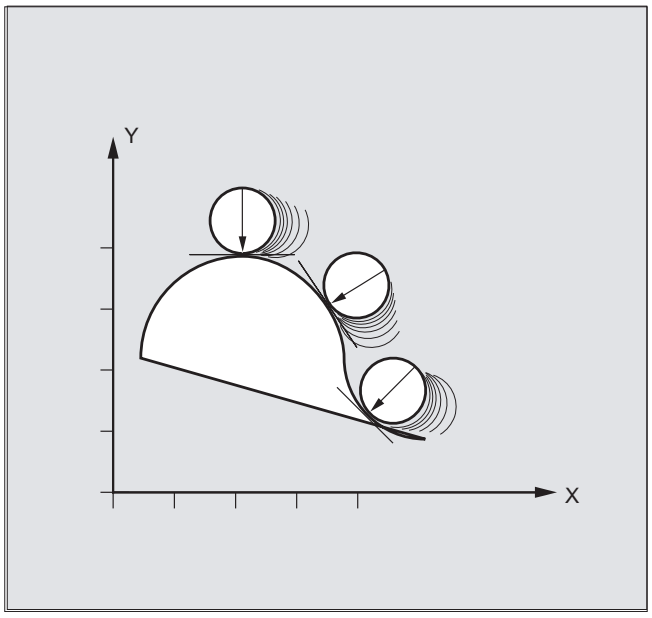
Modes de déplacement

8

8.1 Positionnement tangentiel (TANG, TANGON, TANGOF, TLIFT, TANGDEL)

Fonction

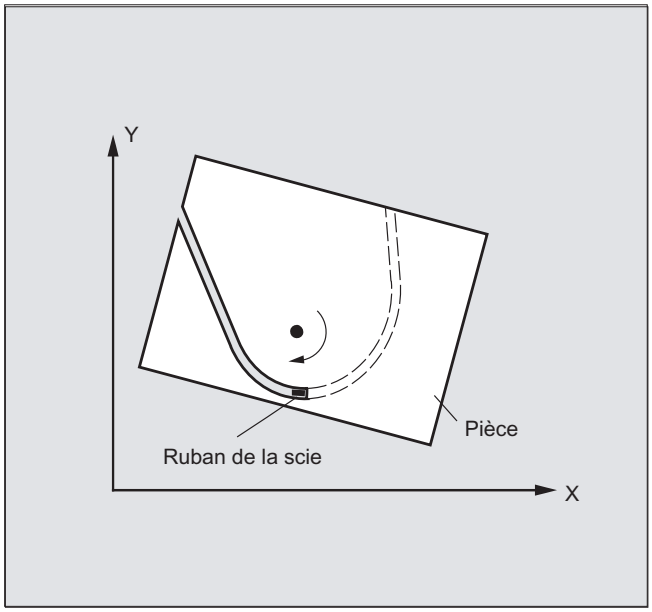
L'axe asservi est, comme son nom l'indique, asservi selon la tangente à la trajectoire déterminée par les axes pilotes. Ainsi, un outil peut être positionné parallèlement au contour. Via l'angle programmé dans l'instruction `TANGON`, l'outil peut être positionné de manière relative par rapport à la tangente.



Application

Le positionnement tangentiel peut par ex. être utilisé pour les applications suivantes :

- positionnement tangentiel d'un outil indexable pour le grignotage
- asservissement du positionnement de la pièce dans le cas d'une scie à ruban (voir figure ci-après)
- positionnement d'un outil de dressage par rapport à une meule
- positionnement d'une molette de coupe pour la transformation du verre ou du papier
- amenée tangentielle d'un fil pour le soudage cinq axes



Syntaxe

Définition de l'asservissement tangentiel :
`TANG (<AxeA>,<AxeP1>,<AxeP2>,<Facteur de couplage>,<SC>,<Opt>)`

Activation du positionnement tangentiel :
`TANGON(<AxeA>,<angle>,<Traj>,<Tolérance angulaire>)`

Désactivation du positionnement tangentiel :
`TANGOF (<AxeA>)`

Activation de la fonction "Insertion d'un bloc intermédiaire aux angles du contour" :
`TLIFT(<AxeA>)`

L'instruction `TLIFT` est spécifiée après l'affection des axes avec `TANG(…)`.

Désactivation de la fonction "Insertion d'un bloc intermédiaire aux angles du contour" :

Répéter l'instruction `TANG(...)` sans la faire suivre d'une instruciton `TLIFT(<AxeA>)`.

Suppression de la définition d'un asservissement tangentiel :
`TANGDEL(<AxeA>)`

Tout asservissement tangentiel défini par l'utilisateur existant doit être supprimé lorsqu'un nouvel asservissement tangentiel doit être défini avec le même axe asservi dans l'appel de préparation `TANG`. La suppression est uniquement possible si le couplage avec `TANGOF (<AxeA>)` a été désactivé.

Signification

`TANG` :	Instruction préparatoire pour la définition d'un asservissement tangentiel
`TANGON` :	Activation du positionnement tangentiel pour l'axe asservi spécifié

TANGOF :	Désactivation du positionnement tangentiel pour l'axe asservi spécifié
TLIFT :	Activation de la fonction "Insertion d'un bloc intermédiaire aux angles du contour"
TANGDEL :	Suppression de la définition d'un asservissement tangentiel
<AxeA> :	Axe asservi : axe rotatif supplémentaire à asservissement tangentiel
<AxeP1>,<AxeP2> :	Axes pilotes : axes en interpolation à partir desquels est définie la tangente pour l'asservissement
<Facteur de couplage> :	Facteur de couplage : rapport entre les variations angulaires de la tangente et de l'axe asservi Réglage par défaut : 1 **Remarque :** Un facteur de couplage de 1 ne doit pas être programmé de manière explicite.
<SC> :	Lettre identifiant le système de coordonnées
	"B" : Système de coordonnées de base (réglage par défaut) **Remarque :** <SC> = "B" ne doit pas être programmé de manière explicite.
	"W" : Système de coordonnées pièces (non disponible)
<Opt> :	Optimisation
	"S" : Standard (réglage par défaut) **Remarque :** <opt> = "S" ne doit pas être programmé de manière explicite.
	"P" : Adaptation automatique de la courbe temporelle de l'axe tangentiel et du contour **Remarque :** <opt> = "P" permet de prendre en compte la dynamique de l'axe asservi lors de la limitation de vitesse des axes pilotes. Ce réglage est conseillé surtout lors de l'utilisation de transformations cinématiques.
<angle> :	Angle de décalage de l'axe asservi
<Traj> :	Trajectoire d'arrondissement de l'axe asservi (nécessaire pour <Opt> = "P")
<Tolérance angulaire> :	Tolérance angulaire de l'axe asservi (facultatif, uniquement traité lorsque <Opt> = "P") **Remarque :** Les paramètres <Traj> et <Tolérance angulaire> limitent de manière ciblée l'erreur entre l'axe asservi et la tangente des axes pilotes.

Exemples

Exemple 1 : définition et activation de l'asservissement tangentiel

Code de programme	Commentaire
N10 TANG(C,X,Y,1,"B","P")	; Définition d'un asservissement tangentiel : l'axe rotatif C doit suivre les axes géométriques X et Y.
N20 TANGON(C,90)	; L'axe C est l'axe asservi. A chaque déplacement des axes à interpolation, il est pivoté pour rester à 90° par rapport à la tangente à la trajectoire.
...	

Remarque

Programmation simplifiée

`TANG(C,X,Y,1,"B","P")` peut être programmé de manière simplifiée comme suit `TANG(C,X,Y,,,"P")`.

Exemple 2 : changement de plan

Code de programme	Commentaire
N10 TANG(A,X,Y,1)	; 1ère définition de l'asservissement tangentiel.
N20 TANGON(A)	; Activation du couplage.
N30 X10 Y20	; Rayon
...	
N80 TANGOF(A)	; Désactivation du 1er couplage.
N90 TANGDEL(A)	; Suppression de la 1ère définition.
...	
TANG(A,X,Z)	; 2ème définition de l'asservissement tangentiel.
TANGON(A)	; Activation du nouveau couplage.
...	
N200 M30	

Exemple 3 : commutation d'axe géométrique et TANGDEL

Aucune alarme n'est générée.

```
Code de programme                    Commentaire
N10 GEOAX(2,Y1)                      ; Y1 est l'axe géométrique 2.
N20 TANG(A,X,Y)                      ; 1ère définition de l'asservissement tangentiel.
N30 TANGON(A,90)                     ; Activation de l'asservissement avec Y1
N40 G2 F8000 X0 Y0 I0 J50
N50 TANGOF(A)                        ; Désactivation de l'asservissement avec Y1.
N60 TANGDEL(A)                       ; Suppression de la 1ère définition.
N70 GEOAX(2,Y2)                      ; Y2 est le nouvel axe géométrique 2.
N80 TANG(A,X,Y)                      ; 2ème définition de l'asservissement tangentiel.
N90 TANGON(A,90)                     ; Activation de l'asservissement avec Y2.
...
```

Exemple 4 : asservissement tangentiel avec optimisation automatique

Y1 est l'axe géométrique 2.

```
Code de programme                              Commentaire
...
N80 G0 C0
N100 F=50000
N110 G1 X1000 Y500
N120 TRAORI
N130 G642                                      ; Arrondissement avec respect de l'écart
                                                 de trajectoire maximal autorisé.
N171 TRANS X50 Y50
N180 TANG(C,X,Y,1,,"P")                        ; Définition de l'asservissement
                                                 tangentiel avec optimisation
                                                 automatique de la vitesse tangentielle.
N190 TANGON(C,0,5.0,2.0)                       ; Activation de l'asservissement
                                                 tangentiel avec optimisation
                                                 automatique : trajectoire
                                                 d'arrondissement 5 mm, tolérance
                                                 angulaire 2 degrés.
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
```

Informations complémentaires

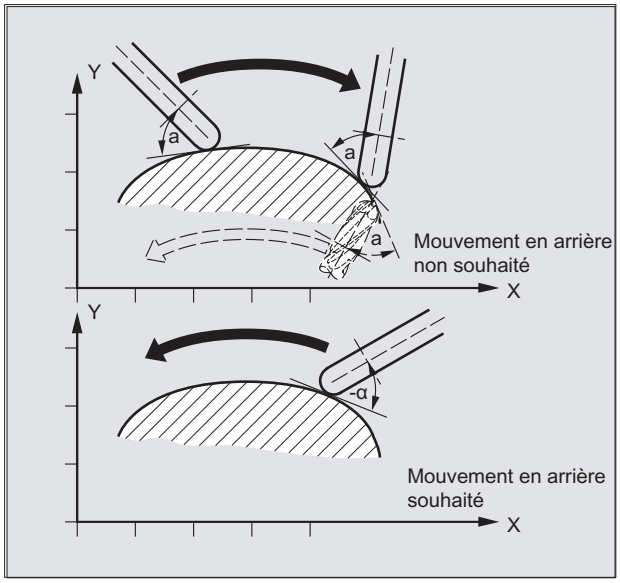
définition de l'axe asservi et des axes pilotes

La définition de l'axe asservi et des axes pilotes s'effectue avec `TANG`.

Un facteur de couplage indique le rapport entre les variations angulaires de la tangente et de l'axe asservi. Sa valeur est en général 1 (préréglage).

angle de décalage par limitation de la zone de travail

Quand les déplacements se font en va-et-vient, la tangente fait un saut de 180° au point d'inversion de la trajectoire, l'axe asservi fait de même. En règle générale, ce comportement n'a pas de sens : le déplacement en arrière doit s'effectuer avec le même angle de décalage négatif que le déplacement en avant :



la zone de travail de l'axe asservi doit à ce but être limitée (`G25`, `G26`). La limitation de la zone de travail doit être active au moment de l'inversion de la trajectoire (`WALIMON`). Dès que l'angle de décalage sort de la zone de travail limitée, la commande tente de l'y faire revenir avec un angle de décalage négatif.

Insertion d'un bloc intermédiaire aux angles du contour (TLIFT)

Quand on s'approche d'un angle de contour, la tangente au contour montre des variations angulaires brusques qui se répercutent sur la consigne de position de l'axe asservi. Celui-ci essaie normalement de compenser ces brusques variations angulaires par une vitesse maximale. Cependant, sur un certain trajet à la suite de l'angle de contour en question, on constate qu'il en résulte un écart par rapport à la position tangentielle désirée. Si cet écart n'est pas tolérable pour des raisons techniques, on peut avec l'instruction `TLIFT` et par le biais d'un bloc intermédiaire créé automatiquement, amener le positionnement tangentiel à s'arrêter avant l'angle de contour et à positionner l'axe asservi dans la nouvelle direction tangentielle.

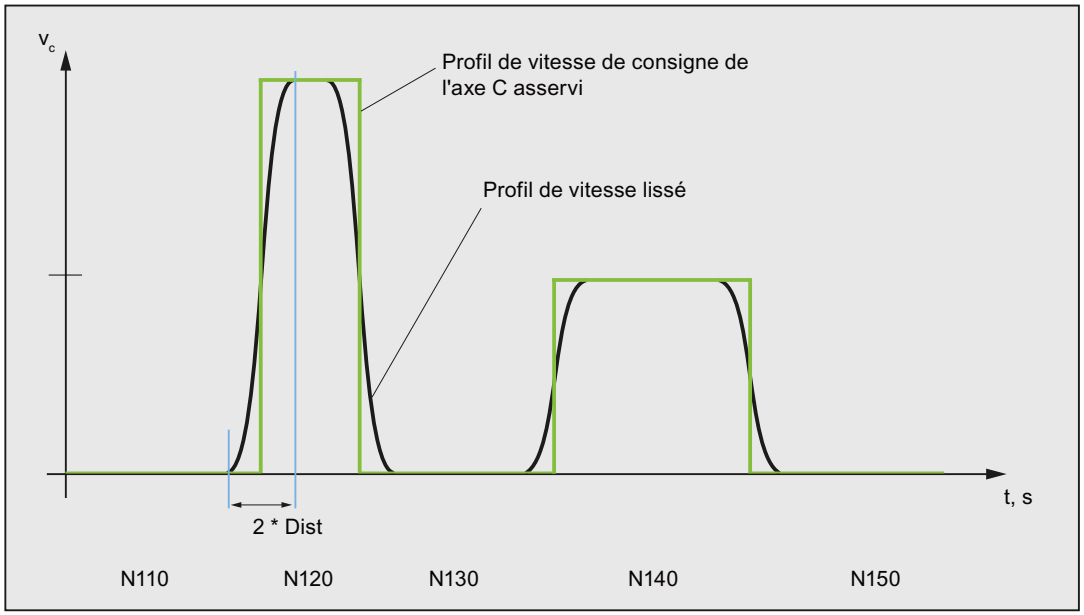
La rotation s'effectue avec l'axe d'interpolation programmé lorsque l'axe asservi a été déplacé une fois comme axe d'interpolation. La fonction `TFGREF[<Axe>] = 0.001` permet d'atteindre dans ce cas une vitesse d'axe maximale de l'axe asservi.

Si l'axe asservi n'a pas été déplacé jusqu'ici comme axe d'interpolation, cet axe est déplacé comme axe de positionnement. La vitesse dépend alors de la vitesse de positionnement qui figure dans les paramètres machine.

Ce positionnement se fait à la vitesse de rotation maximale de l'axe asservi.

Possibilité d'optimisation

Si l'optimisation automatique est activée (`<Opt>="P"`) et que les paramètres Trajectoire d'arrondissement (`<Traj>`) et Tolérance angulaire (`<Tolérance angulaire>`) sont spécifiés pour l'axe asservi, d'éventuelles variations brusques de la vitesse de l'axe asservi dues à des variations du contour de l'axe pilote sont arrondies ou lissées lors de l'asservissement tangentiel. L'axe asservi est alors positionné par anticipation (voir le diagramme) afin de maintenir l'écart à une valeur aussi faible que possible.



Définition de la variation angulaire

La variation angulaire, à partir de laquelle un bloc intermédiaire automatique est introduit, est défini dans le paramètre machine suivant :

MD37400 $MA_EPS_TLIFT_TANG_STEP (angle tangentiel pour la détection d'angles)

Effet sur les transformations

La position de l'axe rotatif asservi peut constituer la valeur initiale pour une transformation.

Positionnement explicite de l'axe asservi

Si l'un de vos axes asservis est positionné de manière explicite, la position indiquée s'ajoute à l'angle de décalage programmé.

Tous les types de trajectoire peuvent être spécifiés (déplacement d'axes à interpolation et d'axes de positionnement).

Etat du couplage

Dans le programme pièce CN, l'état du couplage peut être interrogé avec la variable système $AA_COUP_ACT[<Axe>] :

Valeur	Signification
0	Aucun couplage activé
1,2,3	Asservissement tangentiel actif

8.2 Variation de l'avance (FNORM, FLIN, FCUB, FPO)

Fonction

En vue d'une introduction plus flexible de l'avance, la programmation de l'avance selon DIN 66025 est étendue aux variations linéaires et cubiques.

Les variations cubiques peuvent être programmées directement ou sous la forme de splines interpolés. Par ce moyen, il est possible de programmer des variations continues et lissées, dépendant de la courbure de la pièce à usiner.

Ces variations de la vitesse permettent des variations d'accélération sans à-coup et, de ce fait, la réalisation d'états de surface plus réguliers.

Syntaxe

```
F… FNORM
F… FLIN
F… FCUB
F=FPO(…,…,…)
```

Signification

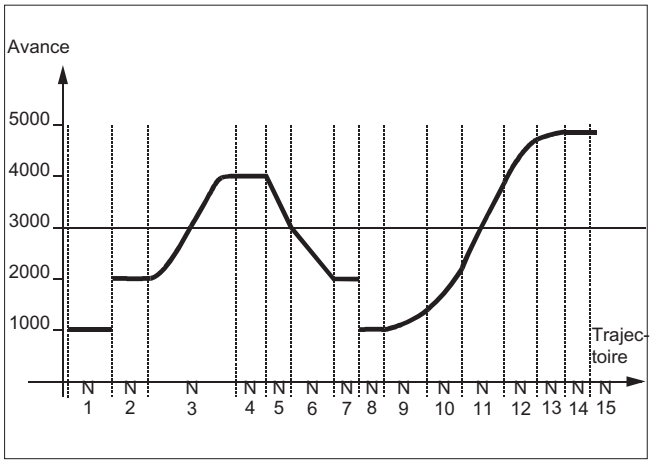
`FNORM`	Préréglage. La valeur de l'avance est définie sur la longueur de la trajectoire du bloc et vaut alors comme valeur modale.
`FLIN`	Variation de l'avance tangentielle **linéaire :** La valeur de l'avance varie linéairement de la valeur en début de bloc à la valeur en fin de bloc et vaut alors comme valeur modale. Ce comportement peut être combiné avec G93 et G94.
`FCUB`	Variation de l'avance tangentielle **cubique :** Les valeurs F programmées dans les blocs - rapportées au point final de bloc - sont raccordées par un spline. Le spline débute et se termine de façon tangentielle aux valeurs d'avance programmées dans le bloc précédent, voire suivant, et agit avec G93 et G94. Si l'adresse F manque dans un bloc, c'est la dernière valeur F programmée qui sera utilisée.
`F=FPO…`	Variation de l'avance tangentielle **définie par un polynôme :** l'adresse F décrit la variation de l'avance selon un polynôme, de la valeur courante jusqu'à la valeur en fin de bloc. La valeur finale vaut alors comme valeur modale.

Optimisation de l'avance sur des sections de trajectoire courbes

Le polynôme d'avance `F=FPO` et le spline d'avance `FCUB` doivent toujours être exécutés à vitesse de coupe constante `CFC` . De ce fait, une variation de consigne d'avance à accélération continue est générée.

Exemple : Différentes courbes d'avance

Vous trouverez, dans cet exemple, la programmation et la représentation graphique de différentes courbes d'avance.

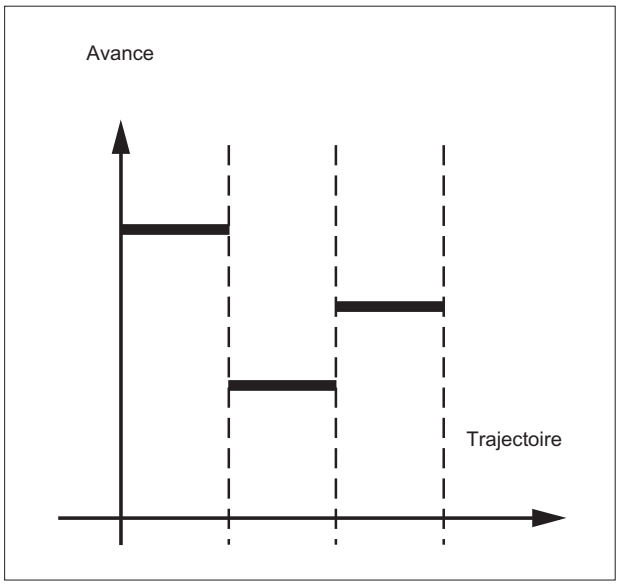


Code de programme	**Commentaire**
N1 F1000 FNORM G1 X8 G91 G64	; Courbe d'avance constante, indication en cotes relatives
N2 F2000 X7	; Variation brusque de la vitesse de consigne
N3 F=FPO(4000, 6000, -4000)	; Courbe d'avance par polynôme avec avance 4000 en fin de bloc
N4 X6	; Avance polynomiale 4000 prise comme valeur modale
N5 F3000 FLIN X5	; Courbe d'avance linéaire
N6 F2000 X8	; Courbe d'avance linéaire
N7 X5	Avance linéaire prise comme valeur modale
N8 F1000 FNORM X5	; Courbe d'avance constante avec variation d'accélération en échelon
N9 F1400 FCUB X8	; Toutes les valeurs de F programmées dans les blocs suivants sont reliées par des splines.
N10 F2200 X6	
N11 F3900 X7	
N12 F4600 X7	
N13 F4900 X5	; Désactiver la courbe spline
N14 FNORM X5	
N15 X20	

FNORM

L'adresse d'avance F désigne l'avance tangentielle en tant que valeur constante selon DIN 66025.

Vous trouverez plus d'informations sur ce sujet dans "Manuel de programmation - Notions de base".

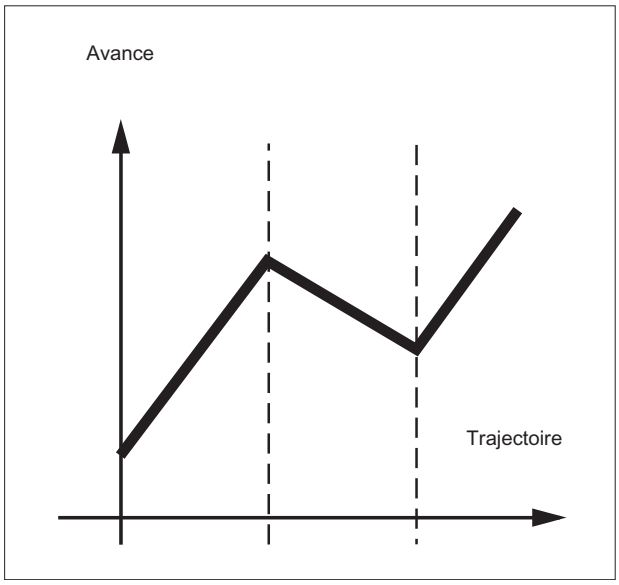


FLIN

L'avance varie de façon linéaire jusqu'à la fin du bloc, depuis la valeur d'avance courante jusqu'à la valeur F programmée.

Exemple :

```
N30 F1400 FLIN X50
```

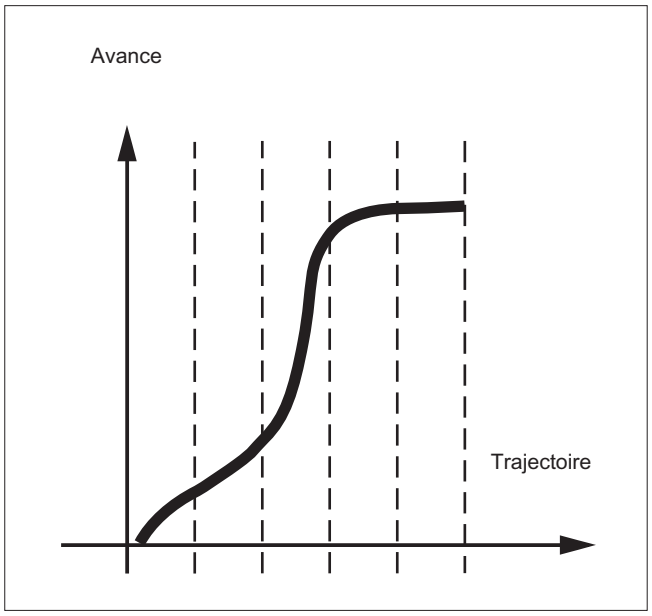


FCUB

L'avance varie de façon cubique jusqu'à la fin du bloc, depuis la valeur courante jusqu'à la valeur F programmée. La commande relie toutes les valeurs programmées dans les blocs avec un FCUB actif par des splines. Les valeurs d'avance servent dans ce cas de points intermédiaires pour le calcul de l'interpolation spline.

Exemple :

```
N50 F1400 FCUB X50
N60 F2000 X47
N70 F3800 X52
```



F=FPO(…,…,…)

La variation de l'avance est directement programmée par le biais d'un polynôme. L'introduction des coefficients du polynôme est la même que pour l'interpolation polynomiale.

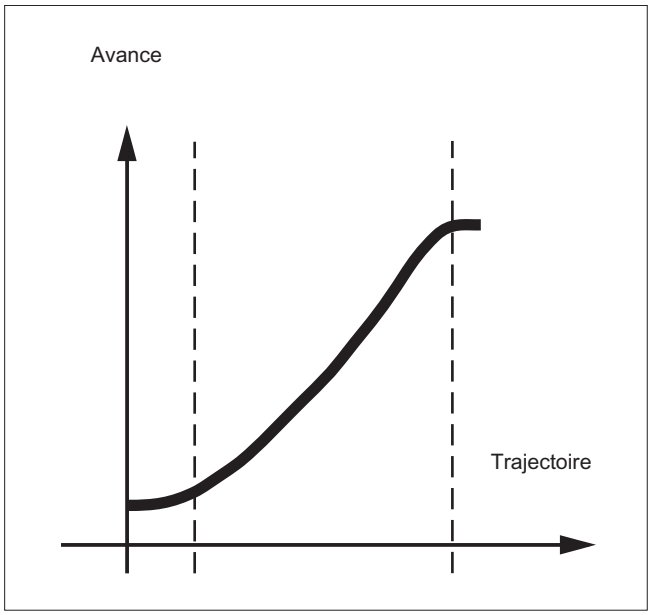
Exemple :

```
F=FPO(endfeed, quadf, cubf)
```

endfeed, quadf et cubf sont des variables précédemment définies.

endfeed :	Avance en fin de bloc
quadf :	Coefficient de polynôme de degré 2
cubf :	Coefficient de polynôme de degré 3

Quand `FCUB` est activée, le spline se raccorde tangentiellement en début et en fin de bloc à l'avance définie par le biais de `FPO` .



Conditions marginales

Les fonctions programmées pour le mode de déplacement restent valables, indépendamment de la variation de l'avance.

La variation programmable de l'avance est toujours valable de façon absolue, indépendamment de `G90` ou `G91`.

La variation de l'avance FLIN et FCUB agit avec

`G93` et `G94`.

`FLIN` et `FCUB` **n'agissent pas avec**

`G95`, `G96/G961` et `G97/G971`.

Compacteur activé COMPON

Lorsque le compacteur `COMPON` est activé, les règles ci-après sont appliquées quand plusieurs blocs sont regroupés en un segment de spline :

FNORM :

Pour le segment de spline, le mot F du dernier bloc qui en fait partie est valide.

FLIN :

Pour le segment de spline, le mot F du dernier bloc qui en fait partie est valide.
La valeur F programmée est valide en fin du segment et est alors accostée de façon linéaire.

FCUB :

Le spline d'avance généré s'écarte au maximum des points finaux programmés d'une valeur égale à la valeur définie dans le paramètre machine `C $MC_COMPRESS_VELO_TOL` .

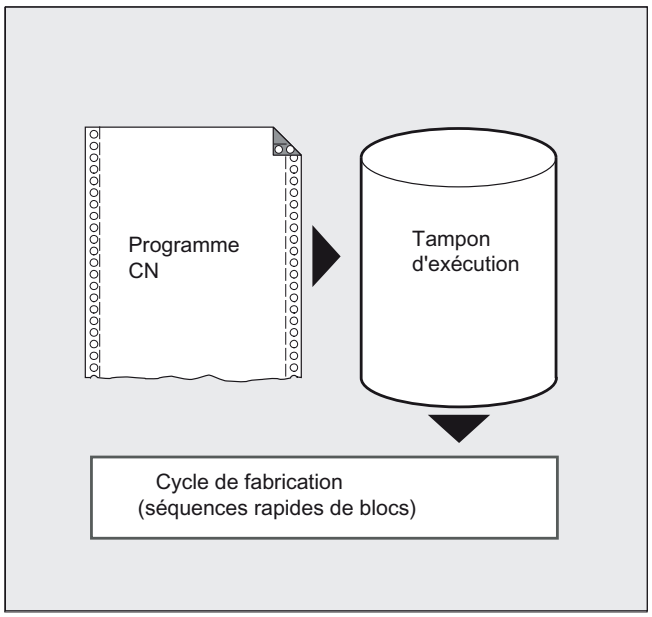
F=FPO(...,...,...)

Ces blocs ne sont pas compactés.

8.3 Exécution du programme avec une mémoire tampon (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE)

Fonction

Selon la configuration, la commande dispose d'une mémoire tampon d'une certaine capacité, capable de stocker les blocs prétraités afin de pouvoir ensuite les transmettre à un rythme rapide lors du déroulement de l'usinage. Ainsi, on peut effectuer des trajets courts à grande vitesse. La mémoire tampon est remplie au fur et à mesure que le permet la commande.



Repérage de la phase d'exécution

La section d'exécution à mémoriser temporairement dans la mémoire tampon est repérée au début par STOPFIFO et à la fin par STARTFIFO. L'exécution des blocs prétraités et mémorisés dans la mémoire tampon débute seulement après l'instruction STARTFIFO ou quand la mémoire tampon est pleine.

Contrôle automatique de la mémoire tampon

L'instruction FIFOCTRL permet d'appeler le contrôle automatique de la mémoire tampon. FIFOCTRL fonctionne d'abord comme STOPFIFO. A chaque programmation, le traitement de la mémoire tampon a lieu seulement lorsqu'elle celle-ci est pleine. Il n'en est pas de même dans le cas de la marche à vide de la mémoire tampon : à partir d'un niveau de remplissage égal à 2/3, la vitesse tangentielle est progressivement réduite avec FIFOCTRL, afin d'éviter une marche à vide complète et une décélération jusqu'à l'arrêt.

l'arrêt du prétraitement des blocs,

Le traitement et la mémorisation temporaire des blocs s'arrêtent lorsque l'instruction STOPRE est programmée dans le bloc. Le bloc suivant n'est exécuté que lorsque tous les blocs prétraités et mémorisés auparavant ont été exécutés en entier. Un arrêt précis (comme G9) a lieu dans le bloc précédent.

Syntaxe

Tableau 8- 1 Repérage de la section d'exécution :

```
STOPFIFO
...
STARTFIFO
```

Tableau 8- 2 Contrôle automatique de la mémoire tampon :

```
...
FIFOCTRL
...
```

Tableau 8- 3 Arrêt du prétraitement des blocs :

```
...
STOPRE
...
```

Remarque

Les instructions STOPFIFO, STARTFIFO, FIFOCTRL et STOPRE doivent être programmées dans un bloc spécifique.

Signification

STOPFIFO : STOPFIFO repère le début de la section d'exécution à mémoriser temporairement dans la mémoire tampon. Avec STOPFIFO, l'exécution est arrêtée et la mémoire tampon remplie jusqu'à ce que :

- STARTFIFO ou STOPRE soient détectés

  ou
- la mémoire tampon soit pleine

  ou
- la fin de programme soit atteinte.

STARTFIFO : STARTFIFO démarre l'exécution rapide de la section de traitement parallèlement au remplissage de la mémoire tampon

FIFOCTRL : Activation du contrôle automatique de la mémoire tampon

STOPRE : Arrêter le prétraitement des blocs

Remarque

Le remplissage de la mémoire tampon n'est ni effectué, ni interrompu lorsque la section d'exécution contient des instructions qui nécessitent un fonctionnement sans tampon (accostage du point de référence, fonctions de mesure, ...).

Remarque

Lors de l'accès à des données d'état de la machine ($SA...), la commande déclenche un arrêt du prétraitement des blocs.

⚠ PRUDENCE

Lorsque la correction d'outil et activée et dans le cas d'interpolations de type spline, il est recommandé de ne pas programmer `STOPRE` afin d'éviter d'interrompre des séquences de blocs contiguës.

Exemple : Arrêter le prétraitement des blocs

```
Code de programme                          Commentaire
...
N30 MEAW=1 G1 F1000 X100 Y100 Z50          ; Bloc de mesure avec palpeur de mesure
                                             de la première entrée de mesure et
                                             interpolation linéaire.
N40 STOPRE                                 ; Arrêt du prétraitement des blocs.
...
```

8.4 Sections de programme interruptibles sous conditions (DELAYFSTON, DELAYFSTOF)

Fonction

Les sections de programme pièce interruptibles sous conditions sont appelées plages d'arrêt temporisé. Il ne doit pas y avoir d'**arrêt** à l'intérieur de certaines sections de programme et l'**avance** ne doit pas être modifiée. En substance, les sections de programme courtes qui servent par exemple à l'usinage d'un filetage, doivent être protégées contre presque tous les événements d'arrêt. Un arrêt éventuel agit seulement après que le traitement de la section de programme a été terminé.

Syntaxe

```
DELAYFSTON
DELAYFSTOF
```

Les instructions se trouvent seules sur une ligne du programme pièce.

Ces deux instructions ne sont autorisées que dans les programmes pièce et non pas dans les actions synchrones.

Signification

`DELAYFSTON`	Définition du début d'une plage dans laquelle des arrêts "en douceur" sont temporisés jusqu'à la fin de la plage d'arrêt temporisé.
`DELAYFSTOF`	Définition de la fin d'une plage d'arrêt temporisé

Remarque

Avec MD11550 $MN_STOP_MODE_MASK Bit 0 = 0 (par défaut), une plage d'arrêt temporisé est définie de manière implicite lorsque `G331/G332` est actif et qu'un déplacement avec interpolation ou `G4` est programmé.

Exemple : Evénements d'arrêt :

Une modification **de l'avance et du verrouillage d'avance** est ignorée dans la plage d'arrêt temporisé. Elle agit seulement après la plage d'arrêt temporisé.

Les événements d'arrêt sont subdivisés en :

événements d'arrêt "en douceur"	Réaction : différée
événements d'arrêt "brusques"	Réaction : immédiate

Sélection de quelques événements d'arrêt provoquant au moins un arrêt à court terme :

Nom de l'événement	Réaction	Paramètre d'interruption
Reset	immédiate	VDI : DB21,… DBX7.7 et DB11, … DBX20.7
PROG_END	Alarme 16954	Prog. CN : M30
INTERRUPT	différée	VDI : FC-9 et ASUP DB10, ... DBB1
SINGLEBLOCKSTOP	différée	Le mode bloc par bloc est activé dans la plage d'arrêt temporisé : la CN s'arrête à la fin du 1er bloc à l'extérieur de la plage d'arrêt temporisé. Le mode bloc par bloc est déjà sélectionné avant la plage d'arrêt temporisé: VDI : "Arrêt CN à chaque fin de bloc" DB21, ... DBX7.2
STOPPROG	différée	VDI : DB21,… DBX7.3 et DB11, … DBX20.5
PROG_STOP	Alarme 16954	Prog. CN : M0 et M1
WAITM	Alarme 16954	Prog. CN : WAITM
WAITE	Alarme 16954	Prog. CN : WAITE
STOP_ALARM	immédiate	Alarme : configuration d'alarme STOPBYALARM
RETREAT_MOVE_THREAD	Alarme 16954	Prog. CN : alarme 16954 pour LFON (arrêt & fastlift impossible dans G33)
WAITMC	Alarme 16954	Prog. CN : WAITMC
NEWCONF_PREP_STOP	Alarme 16954	Prog. CN : NEWCONF
SYSTEM_SHUTDOWN	immédiate	Arrêt système avec 840Di
ESR	différée	Arrêt étendu et retrait
EXT_ZERO_POINT	différée	Décalage d'origine externe
STOPRUN	Alarme 16955	OPI : PI "_N_FINDST" STOPRUN

Signification des réactions

immédiate (événement d'arrêt "brusque")	Arrêt immédiat même dans la plage d'arrêt temporisé
différée (événement d'arrêt "en douceur")	Arrêt (même de courte durée) seulement après la plage d'arrêt temporisé.
Alarme 16954	Le programme est interrompu compte tenu que des instructions de programme non autorisées ont été utilisées dans la plage d'arrêt temporisé.
Alarme 16955	Le programme se poursuit, une action non autorisée s'est produite dans la plage d'arrêt temporisé.
Alarme 16957	La plage de programme (plage d'arrêt temporisé) délimitée par DELAYFSTON et DELAYFSTOF n'a pas pu être activée. Ainsi, chaque arrêt dans cette plage agit immédiatement et n'est pas temporisé.

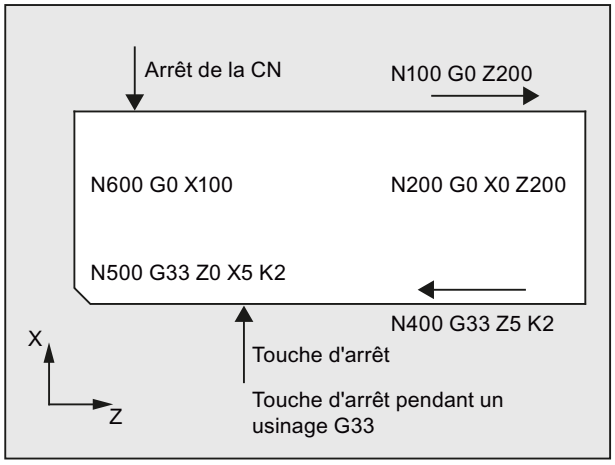
Pour un récapitulatif des autres réactions aux événements d'arrêt, voir :
Bibliographie :
Description fonctionnelle Fonctions de base ; GMF, Canal, Mode de programme, (K1), chapitre "Correction et répercussion sur des événements d'arrêt"

Exemple : Imbrication de plages d'arrêt temporisé dans deux niveaux de programmes

```
Code de programme                Commentaire
N10010 DELAYFSTON()              ; Blocs avec N10xxx comme niveau de programme 1.
N10020 R1 = R1 + 1
N10030 G4 F1                     ; Départ de la plage d'arrêt temporisé.
...
N10040 sous-programme2
...
...                              ; Interprétation du sous-programme 2.
N20010 DELAYFSTON()              ; Sans effet, nouveau départ, 2e niveau.
...
N20020 DELAYFSTOF()              ; Sans effet, fin dans un autre niveau.
N20030 RET
N10050 DELAYFSTOF()              ; Fin de la plage d'arrêt temporisé dans le même
                                   niveau.
...
N10060 R2 = R2 + 2
N10070 G4 F1                     ; Fin de la plage d'arrêt temporisé. Les arrêts
                                   s'activent directement à partir de maintenant.
```

Exemple : Extrait de programme

Dans une boucle, la section de programme suivante est répétée.



On peut voir sur la figure que l'utilisateur appuie sur "Arrêt" dans la plage d'arrêt temporisé et que la CN commence l'opération de freinage à l'extérieur de la plage d'arrêt temporisé, c'est-à-dire dans le bloc `N100`. Ainsi, la CN s'arrête au début de `N100` .

Code de programme	Commentaire
...	
N99 MY_LOOP:	
N100 G0 Z200	
N200 G0 X0 Z200	
N300 DELAYFSTON()	
N400 G33 Z5 K2 M3 S1000	
N500 G33 Z0 X5 K3	
N600 G0 X100	
N700 DELAYFSTOF()	
N800 GOTOB MY_LOOP	

Détails sur la recherche de blocs de type SERUPRO et avances en corrélation avec `G331/G332` Avance lors du taraudage sans porte-taraud compensateur voir :
Bibliographie :
Description fonctionnelle Fonctions de base ; GMF, Canal, Mode de programme (K1)
Description fonctionnelle Fonctions de base ; Avances (V1)

Avantages de la plage d'arrêt temporisé

Une section de programme est traitée sans discontinuité de vitesse.

Si l'utilisateur abandonne le programme par RESET après qu'il a été annulé, le bloc de programme interrompu se trouve après la plage protégée. Ce bloc de programme convient alors comme destination de recherche pour une recherche ultérieure.

Aussi longtemps qu'une plage d'arrêt temporisé est traitée, les axes médians principaux suivants ne sont pas arrêtés :

- les axes de commande et
- les axes de positionnement qui se déplacent avec `POSA` .

L'instruction de programme pièce `G4` est autorisée dans la plage d'arrêt temporisé, contrairement à d'autres instructions de programme pièce qui entraînent un arrêt momentané (par ex., `WAITM`).

Comme un déplacement avec interpolation, `G4` active la plage d'arrêt temporisé ou maintient sa prise d'effet.

Exemple : opérations d'avance

Si la correction chute à 6 % avant la plage d'arrêt temporisé, la correction est active dans cette dernière.

Si la correction chute de 100 % à 6 % dans la plage d'arrêt temporisé, cette dernière est exécutée jusqu'à la fin avec 100 % et continue ensuite avec 6 %.

Le verrouillage d'avance n'agit pas dans la plage d'arrêt temporisé, l'arrêt s'effectue seulement après avoir quitté la plage.

Chevauchement / imbrication :

Si deux plages d'arrêt temporisé se chevauchent, l'une issue des instructions et l'autre issue du paramètre machine PM 11550 : STOP_MODE_MASK, la plage d'arrêt temporisé la plus grande possible est créée.

Les points suivants régissent l'interaction des instructions DELAYFSTON et DELAYFSTOF avec les imbrications et la fin de sous-programme :

1. À la fin du sous-programme dans lequel DELAYFSTON a été appelée, l'instruction DELAYFSTOF est activée de manière implicite.
2. La plage d'arrêt temporisé DELAYFSTON reste sans effet.
3. Si le sous-programme1 appelle le sous-programme2 dans une plage d'arrêt temporisé, le sous-programme2 est une plage d'arrêt temporisé à part entière. Notamment, l'instruction DELAYFSTOF est sans effet dans le sous-programme2.

Remarque

REPOSA est une fin de sous-programme et DELAYFSTON est désactivée dans tous les cas.

S'il se produit un événement d'arrêt "brusque" dans la plage d'arrêt temporisé", cette dernière est complètement désactivée ! Autrement dit, si un autre arrêt quelconque se produit dans cette section de programme, l'exécution est immédiatement arrêtée. Seule une nouvelle programmation (nouvelle instruction DELAYFSTON) permet de commencer une nouvelle plage d'arrêt temporisé.

Si la touche Arrêt est pressée avant la plage d'arrêt temporisé et que la CN doit accoster la plage d'arrêt temporisé pour le freinage, la CN s'arrête dans la plage d'arrêt temporisé et cette dernière demeure désactivée!

En cas d'entrée dans une plage d'arrêt temporisé avec correction égale à 0 %, la plage d'arrêt temporisé **n'est pas** acceptée!

Cela vaut pour tous les événements d'arrêt "en douceur".

AvecSTOPALL , il est possible de freiner dans la plage d'arrêt temporisé. Toutefois, avec STOPALL , tous les autres événements d'arrêt qui étaient temporisés jusque là sont immédiatement activés.

Variables système

La présence d'une plage d'arrêt temporisé peut être déterminée à l'aide de l'instruction $P_DELAYFST dans le programme pièce. Si le bit 0 de la variable système est mis à 1, l'exécution du programme pièce se trouve à ce moment-là dans une plage d'arrêt temporisé.

La présence d'une plage d'arrêt temporisé peut être déterminée à l'aide de l'instruction $AC_DELAYFST dans les actions synchrones. Si le bit 0 de la variable système est mis à 1, l'exécution du programme pièce se trouve à ce moment-là dans une plage d'arrêt temporisé.

Compatibilité

Le préréglage du paramètre machine PM 11550 : STOP_MODE_MASK bit 0 = 0 génère une plage d'arrêt temporisé implicite au cours du groupe de codes G331/G332 et lorsqu'un déplacement avec interpolation ou G4 est programmé.

Bit 0 = 1 permet l'arrêt au cours du groupe de codes G331/G332 et si un déplacement avec interpolation ou G4 est programmé (comportement jusqu'à la version 6 du logiciel). Les instructions DELAYFSTON/DELAYFSTOF doivent être utilisées pour la définition d'une plage d'arrêt temporisé.

8.5 Interdire des positions de programme pour SERUPRO (IPTRLOCK, IPTRUNLOCK)

Fonction

Pour certaines situations mécaniques complexes, il est nécessaire d'interdire l'opération de recherche de bloc SERUPRO sur la machine.

Avec un pointeur d'interruption programmable, il est possible via la "recherche de points d'interruption", d'intervenir avant la position interdite à la recherche.

Il est également possible de définir des plages interdites à la recherche dans les plages de programme pièce dans lesquelles le noyau CN ne peut pas redémarrer. Avec l'abandon de programme, le noyau CN note le bloc traité en dernier sur lequel peut être effectuée la recherche via l'interface utilisateur IHM.

Syntaxe

```
IPTRLOCK
IPTRUNLOCK
```

Les instructions se trouvent seules sur une ligne du programme pièce et permettent l'utilisation d'un pointeur d'interruption programmable

Signification

`IPTRLOCK`	Début de la section de programme interdite à la recherche
`IPTRUNLOCK`	Fin de la section de programme interdite à la recherche

Ces deux instructions ne sont autorisées que dans les programmes pièce et **non pas** dans les actions synchrones.

Exemple

Imbrication de sections de programme interdites à la recherche dans deux niveaux de programmes avec l'instruction implicite IPTRUNLOCK. L'instruction implicite IPTRUNLOCK du sous-programme 1 termine la plage interdite à la recherche.

```
Code de programme                Commentaire
N10010 IPTRLOCK()
N10020 R1 = R1 + 1
N10030 G4 F1                     ; Bloc d'arrêt qui commence la section de programme
                                   interdite à la recherche.
...
N10040 sous-programme2
...                              ; Interprétation du sous-programme 2.
N20010 IPTRLOCK ()               ; Sans effet, nouveau départ.
...
N20020 IPTRUNLOCK ()             ; Sans effet, fin dans un autre niveau.
N20030 RET
...
N10060 R2 = R2 + 2
N10070 RET                       ; Fin de la section de programme interdite à la
                                   recherche.
N100 G4 F2                       ; Le programme principal reprend.
```

Le pointeur d'interruption fournit alors à nouveau une interruption sur 100.

Saisie et recherche de plages interdites à la recherche

Les sections de programme interdites à la recherche sont identifiées à l'aide des instructions IPTRLOCK et IPTRUNLOCK .

L'instruction IPTRLOCK gèle le pointeur d'interruption sur un mode bloc par bloc exécutable (SBL1). Ce bloc est désigné dans ce qui suit par "bloc d'arrêt". Si une interruption de programme se produit après IPTRLOCK , la recherche est possible sur l'interface utilisateur HMI après ce bloc d'arrêt.

Reprise sur le bloc courant

Avec IPTRUNLOCK , le pointeur d'interruption est placé au point d'interruption sur le bloc courant pour la section de programme suivante.

Après qu'une destination de recherche a été trouvée, il est possible de répéter une nouvelle destination de recherche avec le même bloc d'arrêt.

Un pointeur d'interruption édité par l'utilisateur doit toujours être supprimé via HMI.

Règles relatives aux imbrications :

Les points suivants régissent l'interaction des instructions `IPTRLOCK` et `IPTRUNLOCK` avec les imbrications et la fin de sous-programme :

1. À la fin du sous-programme dans lequel `IPTRLOCK` a été appelée, l'instruction `IPTRUNLOCK` est activée de manière implicite.
2. Dans une plage interdite à la recherche,`IPTRLOCK` reste sans effet.
3. Si le sous-programme1 appelle le sous-programme2 dans une plage interdite à la recherche, le sous-programme2 reste complètement interdit à la recherche. Notamment, l'instruction `IPTRUNLOCK` est sans effet dans le sous-programme2.

Pour de plus amples informations voir
/FB/ Manuel de fonctions de base ; GMF, Canal, Mode de programme (K1).

Variable système

La présence d'une plage interdite à la recherche peut être déterminée à l'aide de l'instruction `$P_IPTRLOCK` dans le programme pièce.

Pointeur d'interruption automatique

La fonction de pointeur d'interruption automatique définit automatiquement comme interdit à la recherche un type de couplage précédemment défini. A l'aide du paramètre machine, le pointeur d'interruption automatique est activé pour

- un réducteur électronique avec `EGON`
- un couplage de deux axes par valeur pilote avec `LEADON`

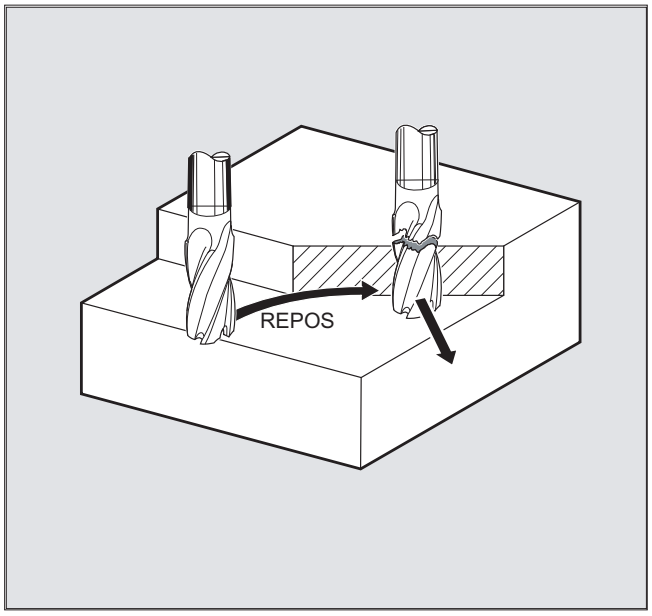
Si le pointeur d'interruption programmé et le pointeur d'interruption automatique se chevauchent, la plage interdite à la recherche la plus grande possible est créée.

8.6 Réaccostage du contour (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN)

Fonction

Si vous stoppez le programme en cours d'usinage et si vous dégagez l'outil pour éviter par exemple un bris d'outil ou parce que vous voulez faire une mesure, vous pouvez programmer le réaccostage du contour à un point de votre choix.

L'instruction REPOS a le même effet qu'un retour de sous-programme (par exemple avec M17). Les blocs suivants dans la routine d'interruption ne seront plus exécutés.



Au chapitre "Routine d'interruption", paragraphe "Programmation flexible CN"du présent manuel de programmation se trouvent de plus amples informations sur l'interruption de l'exécution d'un programme.

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=…
```

Signification

Trajet d'accostage

`REPOSA`	Accostage sur une droite avec tous les axes
`REPOSL`	accostage sur une droite
`REPOSQ DISR=…`	Accostage sur un quart de cercle avec rayon DISR
`REPOSQA DISR=…`	Accostage dans tous les axes sur un quart de cercle avec rayon DISR
`REPOSH DISR=…`	Accostage sur un demi-cercle avec diamètre DISR
`REPOSHA DISR=…`	Accostage dans tous les axes sur un demi-cercle avec rayon DISR

Point de réaccostage

`RMI`	Accostage du point d'interruption
`RMI DISPR=…`	Point d'entrée à une distance DISPR en mm/inch avant le point d'interruption
`RMB`	Accostage du point de début de bloc
`RME`	Accostage du point de fin de bloc
`RME DISPR=…`	Accostage du point de fin de bloc à une distance DISPR avant le point final
`RMN`	Accostage au point de contour suivant
`A0 B0 C0`	Axes dans lesquels le déplacement doit avoir lieu

Exemple : Accostage sur une droite REPOSA, REPOSL

L'outil rallie le point de réaccostage sur une ligne droite.

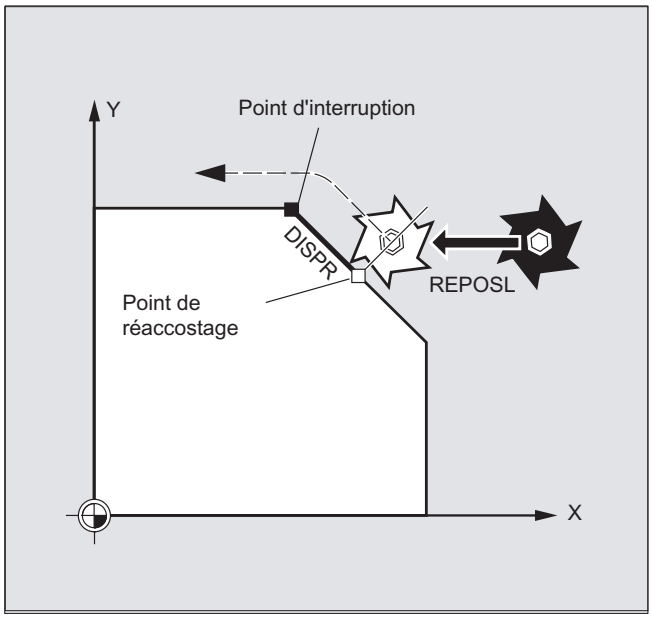
Avec REPOSA, tous les axes sont déplacés automatiquement. Avec REPOSL, vous pouvez préciser les axes à déplacer.

Exemple :

```
REPOSL RMI DISPR=6 F400
```

ou

```
REPOSA RMI DISPR=6 F400
```

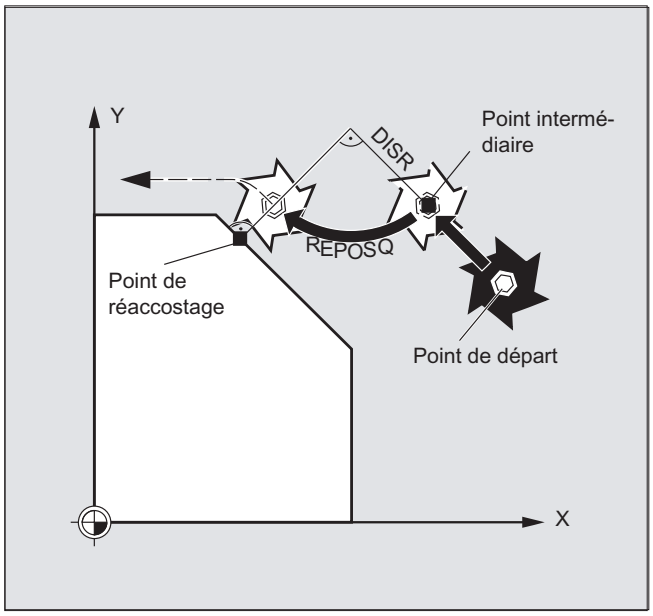


Exemple : Accostage sur un quart de cercle, REPOSQ, REPOSQA

L'outil rallie le point de réaccostage sur un quart de cercle de rayon`DISR=...`. La commande calcule automatiquement le point intermédiaire nécessaire entre le point de départ et le point de réaccostage.

Exemple :

```
REPOSQ RMI DISR=10 F400
```

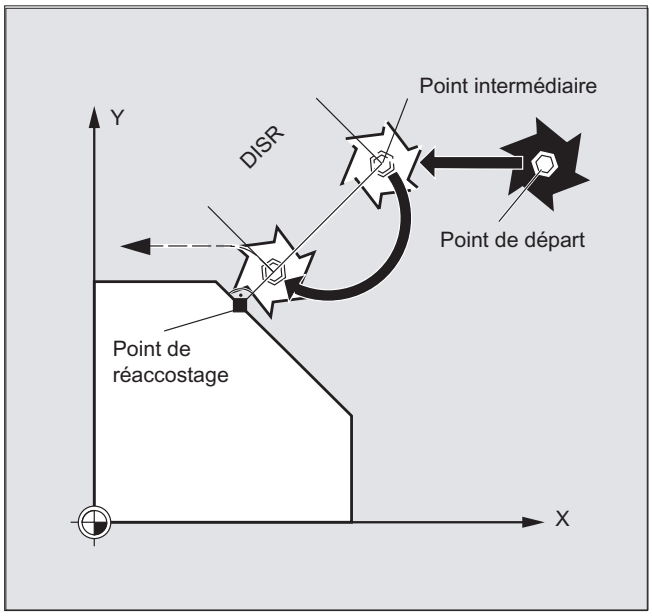


Exemple : Accostage d'outil sur un demi-cercle, REPOSH, REPOSHA

L'outil rallie le point de réaccostage sur un demi-cercle de diamètre`DISR=...`. La commande calcule automatiquement le point intermédiaire nécessaire entre le point de départ et le point de réaccostage.

Exemple :

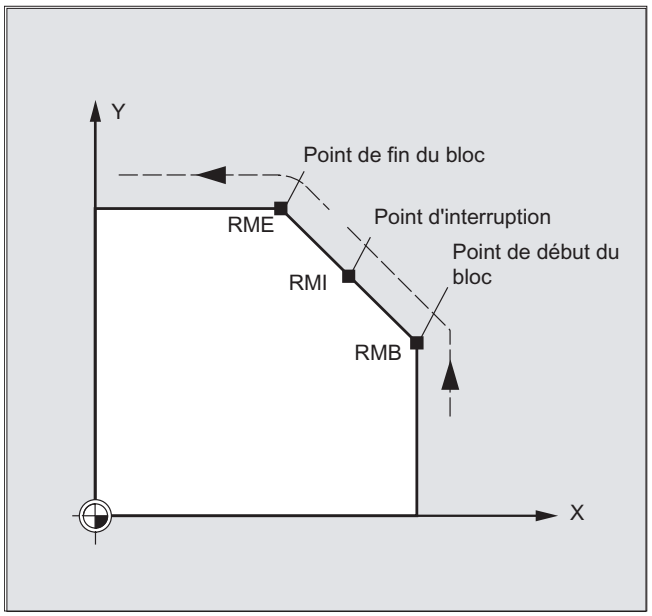
```
REPOSH RMI DISR=20 F400
```



Définir un point de réaccostage (sauf pour l'accostage SERUPRO sous RMN)

En fonction du bloc CN dans lequel l'interruption a eu lieu, vous avez le choix entre trois points de réaccostage :

- RMI, point d'interruption
- RMB, point de début de bloc ou dernier point final
- RME, point de fin de bloc



Avec `RMI DISPR=...` ou `RME DISPR=...`, vous pouvez définir un point de réaccostage se trouvant avant le point d'interruption ou le point de fin de bloc.

Avec `DISPR=...`, vous programmez, en mm/inch, la distance sur le contour **entre** le point de réaccostage et le point d'interruption ou le point de fin de bloc. Ce point ne peut pas figurer au-delà du point de début de bloc – même pour les plus grandes valeurs.

Si `DISPR=...` n'est pas programmé, `DISPR=0` s'applique, c'est-à-dire que le réaccostage a lieu au point d'interruption (pour `RMI`) ou au point de fin de bloc (pour`RME`).

Signe de DISPR

Le signe de DISPR est exploité. Si le signe est positif, le comportement est le même que précédemment.

Si le signe est négatif, le réaccostage a lieu derrière le point d'interruption ou, pour RMB, derrière le point de départ.

La distance point d'interruption-point de réaccostage est la valeur absolue de DISPR. Ce point ne peut pas se trouver au-delà du point de fin de bloc, même pour les plus grandes valeurs absolues.

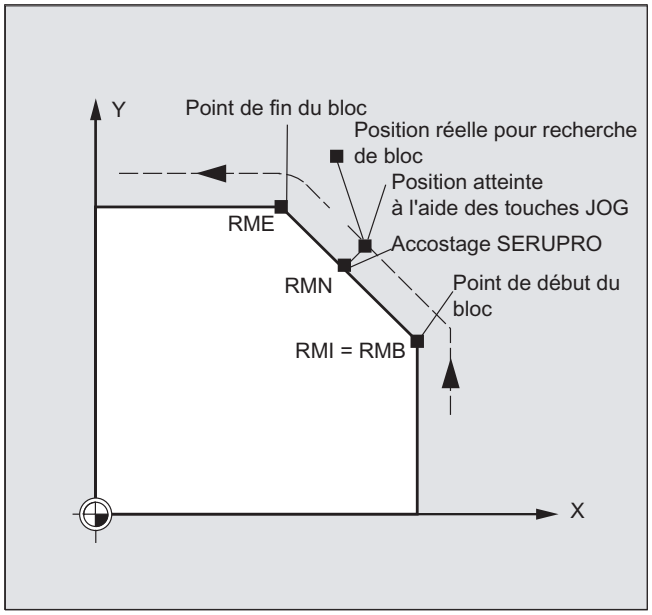
Exemple d'application :

Un capteur surveille les abords d'une griffe de serrage. En cas de violation, un ASUP, qui assure le contournement de la griffe de serrage, est déclenché.

Ensuite, un repositionnement à un point situé derrière la griffe de serrage avec DISPR négatif a lieu et le programme se poursuit.

Accostage SERUPRO sous RMN

Si, lors de l'usinage, une interruption a été forcée sur une position quelconque, le trajet le plus court depuis la position d'interruption est atteint avec l'accostage SERUPRO sous RMN afin de n'exécuter ensuite que la distance restant à parcourir. L'utilisateur lance une opération SERUPRO sur le bloc d'interruption et se positionne, à l'aide des touches JOG, avant la position défectueuse du bloc recherché.



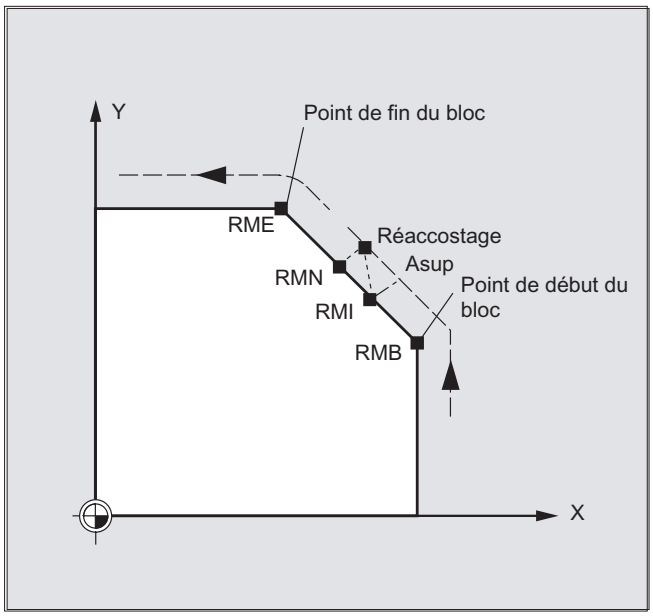
Remarque

SERUPRO

Pour SERUPRO , RMI et RMB sont identiques. RMN n'est pas seulement limité à SERUPRO , il est valide de manière générale.

Accostage au point de contour RMN suivant

Au moment de l'interprétation de `REPOSA` après une interruption, le réaccostage avec `RMN` n'a pas encore complètement commencé, seule la distance restant à parcourir a été traitée. C'est le point de contour suivant du bloc interrompu qui est réaccosté.



Etat du mode REPOS valide

Le mode REPOS valide du bloc interrompu peut être lu via des actions synchrones avec la `$AC_REPOS_PATH_MODE` :

0: Accostage non défini

1 `RMB` : accostage au début

2 `RMI` : accostage au point d'interruption

3 `RME` : accostage au point de fin de bloc

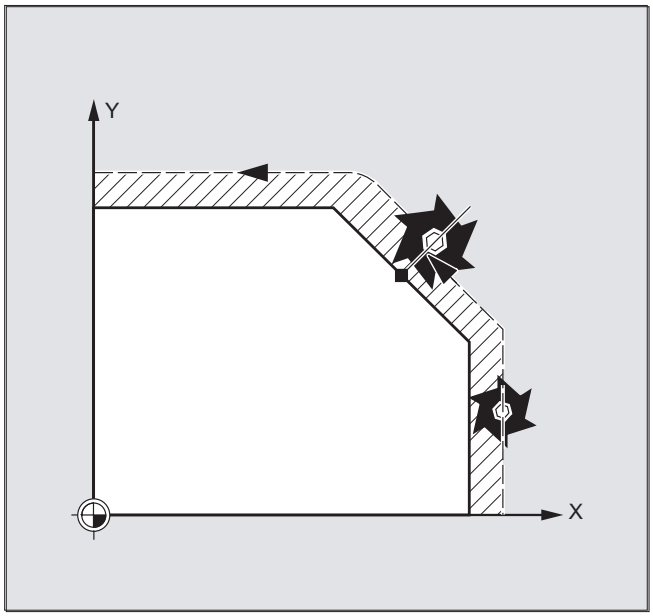
4 `RMN` : accostage au point de contour suivant du bloc interrompu.

Accostage avec un nouvel outil

Si vous avez stoppé l'exécution du programme suite à un bris d'outil :

En programmant le nouveau numéro D, vous poursuivez l'exécution du programme à partir du point de réaccostage et avec des valeurs de correction d'outil modifiées.

Suite aux différentes valeurs de correction d'outil, le point d'interruption risque de ne pas pouvoir être réaccosté. Dans ce cas, c'est le point le plus proche du point d'interruption qui devient point de réaccostage du contour (le cas échéant avec une modification DISPR).



Accostage du contour

Le déplacement à effectuer pour réaccoster le contour est programmable. Indiquez zéro pour les adresses des axes à déplacer.

Avec les instructions `REPOSA`, `REPOSQA` et `REPOSHA` , tous les axes sont repositionnés automatiquement. Il n'est pas nécessaire de préciser les axes.

Quand on programme `REPOSL`, `REPOSQ` et `REPOSH` , tous les axes géométriques se déplacent automatiquement, autrement dit sans qu'il soit nécessaire de le préciser dans l'instruction. Tous les autres axes à repositionner sont à préciser dans l'instruction.

Les points suivants s'appliquent pour les mouvements circulaires REPOSH et REPOSQ :

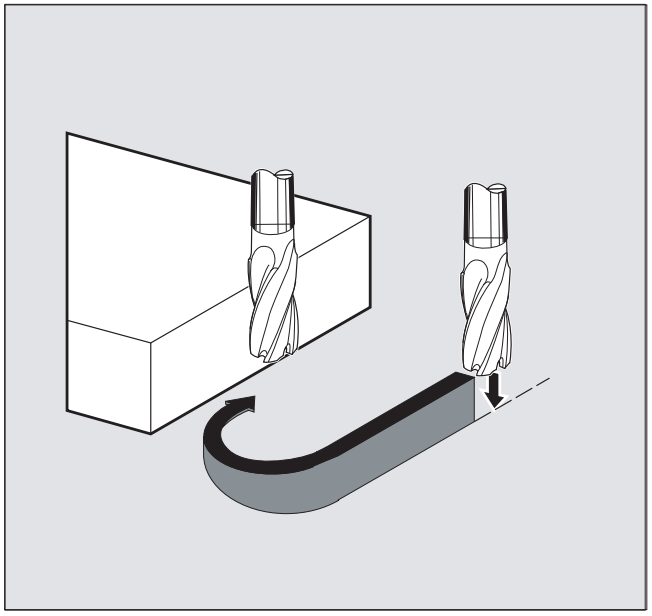
Le mouvement circulaire est décrit dans le plan de travail `G17` à `G19` indiqué.

Si vous indiquez le troisième axe géométrique (pénétration) dans le bloc d'accostage et si la position de l'outil et la position programmée dans la direction de pénétration ne concordent pas, le réaccostage du contour se fera suivant un mouvement hélicoïdal.

Dans les cas suivants, il y a commutation automatique sur

accostage linéaire `REPOSL` :

- Vous n'avez indiqué aucune valeur pour `DISR` .
- Il n'existe aucun sens d'accostage défini (interruption de l'exécution du programme dans un bloc ne contenant aucune information de déplacement).
- Le sens d'accostage est perpendiculaire au plan de travail courant.



8.7 Correction du pilotage des déplacements

8.7.1 Correction de l'à-coup en pourcentage (JERKLIM)

Fonction

L'instruction CN JERKLIM permet de réduire ou d'augmenter l'à-coup maximal possible paramétré par un paramètre machine lors de déplacements avec interpolation dans des sections critiques d'un programme.

Condition

Il est nécessaire que le mode d'accélération SOFT soit activé.

Prise d'effet

La fonction prend effet :

- dans les modes de fonctionnement AUTOMATIQUE,
- uniquement sur des axes à interpolation.

Syntaxe

```
JERKLIM[<axe>]=<valeur>
```

Signification

JERKLIM :	Instruction de correction d'à-coup
<Axe> :	Axe machine dont la valeur limite d'à-coup doit être adaptée.
<Valeur> :	Valeur de correction en pourcentage, en rapport à l'à-coup maximal configuré de l'axe lors d'un déplacement avec interpolation (MD32431 $MA_MAX_AX_JERK). Plage de valeurs : 1 ... 200 La valeur 100 n'entraîne aucune correction de l'à-coup.

Remarque

Le comportement de JERKLIM à la fin du programme pièce ou lors d'un Reset de canal est configuré avec le bit 0 dans le paramètre machine MD32320 $MA_DYN_LIMIT_RESET_MASK :

- Bit 0 = 0 :
  La valeur programmée pour JERKLIM est remise à 100 % avec un Reset de canal/M30.
- Bit 0 = 1 :
  La valeur programmée pour JERKLIM reste inchangée même après un Reset de canal/M30.

Exemple

```
Code de programme            Commentaire
...
N60 JERKLIM[X]=75            ; Dans la direction X, le chariot de déplacement axial
                               ne doit accélérer/décélérer qu'avec 75 % au maximum
                               de l'à-coup autorisé pour l'axe.
...
```

8.7.2 Correction de la vitesse en pourcentage (VELOLIM)

Fonction

L'instruction CN VELOLIM permet de réduire la vitesse maximale possible paramétrée par un paramètre machine pour un axe/une broche en mode axe ou la vitesse de rotation maximale possible d'une broche dépendant du rapport de transmission en mode broche (mode de régulation de vitesse de rotation M3, M4, M5 et mode de positionnement SPOS, SPOSA, M19) dans des sections critiques d'un programme, par ex. pour réduire les sollicitations de la machine ou pour améliorer la qualité d'usinage.

Prise d'effet

La fonction prend effet :

- dans les modes de fonctionnement AUTOMATIQUE,
- sur des axes à interpolation et de positionnement,
- sur des broches en mode broche / mode axe.

Syntaxe

```
VELOLIM[<Axe/Broche>]=<Valeur>
```

Signification

`VELOLIM` :	Instruction de correction de vitesse
`<Axe/Broche>` :	Axe machine ou broche dont la valeur limite de vitesse ou de vitesse de rotation doit être adaptée. **VELOLIM pour broches** Le paramètre machine (MD30455 $MA_MISC_FUNCTION_MASK, Bit 6) permet de configurer dans le programme pièce si `VELOLIM` prend effet indépendamment de son utilisation actuelle en tant que broche ou axe (bit 6 = 1) ou bien s'il doit pouvoir être programmé individuellement pour chaque mode de fonctionnement (bit 6 = 0). Si un effet distinct est configuré, la sélection a lieu lors de la programmation au moyen du descripteur : • Descripteur de broche `S<n>` pour modes broche • Descripteur d'axe, par ex. "`C`", pour mode axe
`<Valeur>` :	Valeur de correction en pourcentage La valeur de correction se rapporte : • Pour des axes / broches en mode axe (si MD30455 Bit 6 = 0) : à la vitesse d'axe maximale configurée (MD32000 $MA_MAX_AX_VELO). • Pour des broches en mode broche ou axe (si MD30455 Bit 6 = 1) : à la vitesse de rotation maximale du rapport de transmission actif (MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT[<n>]) Plage de valeurs : 1 ... 100 La valeur 100 n'entraîne aucune correction de la vitesse ni de la vitesse de rotation.

Remarque

Comportement à la fin du programme pièce et lors d'un Reset de canal

Le comportement de VELOLIM à la fin du programme pièce ou lors d'un Reset de canal est configuré avec le bit 0 dans le paramètre machine MD32320 $MA_DYN_LIMIT_RESET_MASK :

- Bit 0 = 0 :
  La valeur programmée pour VELOLIM est remise à 100 % avec un Reset de canal/M30.
- Bit 0 = 1 :
  La valeur programmée pour VELOLIM reste inchangée même après un Reset de canal/M30.

Remarque

VELOLIM pour broches dans des actions synchrones

Aucune distinction n'est faite entre le mode broche et le mode axe lors de la programmation de VELOLIM dans des actions synchrones. La vitesse de rotation en mode broche et la vitesse en mode axe sont limitées de manière identique, indépendamment du descripteur utilisé lors de la programmation.

Diagnostic

Diagnostic de VELOLIM en mode broche

Une limitation de la vitesse de rotation active par VELOLIM (inférieur à 100 %) peut être détectée en mode broche par la lecture des variables système $AC_SMAXVELO et $AC_SMAXVELO_INFO.

Dans le cas d'une limitation, $AC_SMAXVELO fournit la limite de vitesse de rotation générée par VELOLIM. Dans ce cas, la variable $AC_SMAXVELO_INFO fournit en retour la valeur "16" comme identification pour la cause de la limitation VELOLIM.

Exemples

Exemple 1 : limitation de vitesse d'un axe machine

```
Code de programme            Commentaire
...
N70 VELOLIM[X]=80            ; Dans la direction X, le chariot de déplacement axial
                               ne doit se déplacer qu'au maximum à 80 % de la
                               vitesse autorisée pour l'axe.
...
```

Exemple 2 : limitation de vitesse de rotation d'un broche

```
Code de programme            Commentaire
N05 VELOLIM[S1]=90           ; Limitation de la vitesse de rotation maximale de la
                               broche 1 à 90 % de 1000 tr/min.
...
N50 VELOLIM[C]=45            ; Limitation de la vitesse de rotation à 45 % de
                               1000 tr/min, C étant le descripteur d'axe de S1.
...
```

Données de configuration pour broche 1 (AX5) :

MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT[1,AX5]=1000	;	Vitesse de rotation maximale du rapport de transmission 1 = 1000 tr/min
MD30455 $MA_MISC_FUNCTION_MASK[AX5] = 64	;	Bit 6 = 1 : La programmation de VELOLIM prend en effet en même temps pour le mode broche et le mode axe, indépendamment du descripteur programmé.

8.7.3 Exemple de programme pour JERKLIM et VELOLIM

Le programme suivant représente un exemple d'application pour une limitation proportionnelle de l'à-coup et de la vitesse :

```
Code de programme                    Commentaire
N1000 G0 X0 Y0 F10000 SOFT G64
N1100 G1 X20 RNDM=5 ACC[X]=20
ACC[Y]=30
N1200 G1 Y20 VELOLIM[X]=5            ; Dans la direction X, le chariot doit se
                                       déplacer au maximum à 5% de la vitesse
                                       autorisée pour l'axe.
JERKLIM[Y]=200                       ; Dans la direction X, le chariot peut
                                       accélérer/décélérer au maximum avec 200%
                                       de l'à-coup autorisé pour l'axe.
N1300 G1 X0 JERKLIM[X]=2             ; Dans la direction X, le chariot doit
                                       accélérer/décélérer au maximum avec 2% de
                                       l'à-coup autorisé pour l'axe.
N1400 G1 Y0
M30
```

8.8 Tolérance de contour/orientation programmable (CTOL, OTOL, ATOL)

Fonction

Les instructions CTOL, OTOL et ATOL permettent d'adapter les tolérance d'usinage définies par les paramètres machine et les données de réglage pour les fonctions de compresseur (COMPON, COMPCURV, COMPCAD), les modes d'arrondissement G642, G643, G645, OST et le lissage de l'orientation ORISON dans le programme CN.

Les valeurs programmées s'appliquent jusqu'à ce qu'elles soient reprogrammées ou supprimées par l'affectation d'une valeur négative. Elles sont en outre supprimées à la fin du programme, lors d'un RESET du canal, RESET GMF, RESET NCK (démarrage à chaud) et Power On (démarrage à froid). Après leur suppression, les valeurs définies dans les paramètres machine et données de réglage s'appliquent à nouveau.

Syntaxe

```
CTOL=<valeur>
OTOL=<valeur>
ATOL[<axe>]=<valeur>
```

Signification

CTOL
: Instruction de programmation de la **tolérance de contour**

  CTOL s'applique à :

  - toutes les fonctions de compresseur
  - tous les modes d'arrondissement, à l'exception de G641 et G644

  <valeur> : La valeur de la tolérance de contour est exprimée par une cote.

  Type : REAL

  Unité : pouce/mm (en fonction de la cotation actuellement réglée)

OTOL
: Instruction de programmation de la **tolérance d'orientation**

  OTOL s'applique à :

  - toutes les fonctions de compresseur
  - au lissage de l'orientation ORISON
  - tous les modes d'arrondissement, à l'exception de G641, G644, OSD

  <valeur> : La valeur de la tolérance d'orientation est exprimée par une cote angulaire.

  Type : REAL

  Unité : degré

ATOL Instruction de programmation d'une **tolérance spécifique à l'axe**

ATOL s'applique à :

- toutes les fonctions de compresseur
- au lissage de l'orientation ORISON
- tous les modes d'arrondissement, à l'exception de G641, G644, OSD

<axe> : Nom de l'axe pour lequel il s'agit de programmer une tolérance

<valeur> : Selon le type d'axe (linéaire ou rotatif), la valeur de la tolérance de l'axe est exprimée par une cote ou une cote angulaire.

Type : REAL

Unité : pour axes linéaires : pouce/mm (en fonction de la cotation actuellement réglée)

pour axes rotatifs : degré

Remarque

CTOL et OTOL sont prioritaires par rapport à ATOL.

Conditions marginales

Frames mis à l'échelle

Les frames mis à l'échelle agissent sur la tolérance programmée de la même manière que sur les positions des axes, c'est-à-dire que la tolérance relative reste identique.

Exemple

```
Code de programme          Commentaire
COMPCAD G645 G1 F10000     ; Activer la fonction de compresseur COMPCAD.
X... Y... Z...             ; Ici, les paramètres machine et données de réglage sont
                             actifs.
X... Y... Z...
X... Y... Z...
CTOL=0.02                  ; A partir d'ici, une tolérance de contour de 0,02 mm est
                             active.
X... Y... Z...
X... Y... Z...
X... Y... Z...
ASCALE X0.25 Y0.25 Z0.25   ; A partir d'ici, une tolérance de contour de 0,005 mm
                             est active.
X... Y... Z...
X... Y... Z...
X... Y... Z...
```

Code de programme	Commentaire
CTOL=-1	; A partir d'ici, les paramètres machine et données de réglage sont de nouveau actifs.
X... Y... Z...	
X... Y... Z...	
X... Y... Z...	

Informations complémentaires

Lecture des valeurs de tolérance

Pour des applications plus approfondies ou pour le diagnostic, la lecture des tolérance actuellement en vigueur pour les fonctions de compresseur (COMPON, COMPCURV, COMPCAD), les modes d'arrondissement G642, G643, G645, OST et le lissage de l'orientation ORISON peut être réalisée au moyen de variables système, quel que soit la façon dont ces tolérances ont été obtenues.

- Dans les actions synchrones ou avec arrêt d'avance dans le programme pièce, au moyen des variables système :

$AC_CTOL	Tolérance de contour qui était active lors du traitement du bloc principal actuel. Si aucune tolérance de contour n'est active, `$AC_CTOL` fournit la racine de la somme des carrés des tolérances appliquées aux axes géométriques.
$AC_OTOL	Tolérance d'orientation qui était active lors du traitement du bloc principal actuel. Si aucune tolérance d'orientation n'est active, `$AC_CTOL` fournit la racine de la somme des carrés des tolérances appliquées aux axes d'orientation lorsqu'une transformation d'orientation est active, sinon la valeur "-1".
$AA_ATOL[<axe>]	Tolérance d'axe qui était active lors du traitement du bloc principal actuel. Si une tolérance de contour est active, `$AA_ATOL[<axe géométrique>]` fournit la tolérance de contour divisée par la racine du nombre d'axes géométriques. Si une tolérance et une transformation d'orientation sont actives, `$AA_ATOL[<axe d'orientation>]` fournit la tolérance d'orientation divisée par la racine du nombre d'axes d'orientation.

Remarque

Si aucune valeur de tolérance n'a été programmée, les variables $A ne sont pas suffisamment différenciées pour permettre une distinction entre les tolérances éventuellement différentes de chacune des fonctions, puisqu'elles ne peuvent fournir qu'une seule valeur.

De tels cas sont susceptibles de se produire lorsque les paramètres machine et les données de réglage définissent des tolérances différentes pour les fonctions de compresseur, l'arrondissement et le lissage de l'orientation. Les variables fournissent alors la valeur la plus grande des fonctions en cours.

Si une fonction de compresseur est p. ex. activée avec une tolérance d'orientation égale à 0,1° et un lissage de l'orientation ORISON égal à 1°, la variable $AC_OTOL fournit la valeur "1". Si l'on désactive le lissage de l'orientation, on ne lit plus que la valeur "0.1".

- Sans arrêt d'avance dans le programme pièce, au moyen des variables système :

$P_CTOL	Tolérance de contour programmée
$P_OTOL	Tolérance d'orientation programmée
$PA_ATOL	Tolérance d'axe programmée

Remarque

Si aucune valeur de tolérance n'est programmée, les variables $P fournissent la valeur "-1".

8.9 Tolérance pour déplacements G0 (STOLF)

Facteur de tolérance G0

Contrairement à l'usinage de pièces, les déplacements G0 (vitesse rapide, mouvements d'approche) peuvent être effectués avec une plus grande tolérance. Ceci a pour avantage que les temps de retrait pour des déplacements G0 sont réduits.

Les tolérances pour les déplacements G0 sont réglés par la configuration du facteur de tolérance G0 (MD20560 $MC_G0_TOLERANCE_FACTOR).

Le facteur de tolérance G0 prend uniquement effet si :

- une des fonctions suivantes est active :
  - Fonctions compresseur : COMPON, COMPCURV et COMPCAD
  - Fonctions d'arrondissement : G642 et G645
  - Arrondissement d'orientation : OST
  - Lissage d'orientation : ORISON
  - Lissage pour orientation relative à la trajectoire : ORIPATH
- plusieurs blocs G0 (≥ 2) se suivent.

  Le facteur de tolérance G0 ne prend pas effet pour un bloc G0 individuel étant donné que, **lors de la transition** d'un déplacement non G0 à un déplacement G0 (et vice-versa), c'est toujours la "**plus petite tolérance**" (tolérance d'usinage) qui prend effet !

Fonction

La programmation de `STOLF` dans le programme pièce permet d'écraser temporairement le facteur de tolérance G0 configuré (MD20560). La valeur dans MD20560 n'étant alors pas modifiée. Le facteur de tolérance configuré prend de nouveau effet après un Reset ou à la fin du programme pièce.

Syntaxe

```
STOLF=<Facteur de tolérance>
```

Signification

`STOLF` :	Instruction pour la programmation du facteur de tolérance G0
`<Facteur de tolérance>` :	Facteur de tolérance G0 Le facteur peut aussi bien être supérieur à 1 qu'inférieur à 1. Des tolérances plus grandes peuvent toutefois normalement être réglées pour des déplacements G0. Lorsque `STOLF=1.0` (correspond à la valeur par défaut configurée), les tolérances pour des déplacements G0 sont les mêmes que celles pour des déplacements non G0.

Variables système

Le facteur de tolérance G0 effectif dans le programme pièce ou dans le bloc IPO peut être lu par le biais des variables système.

- Dans les actions synchrones ou avec un du prétraitement des blocs dans le programme pièce, au moyen de la variable système :

  $AC_STOLF Facteur de tolérance G0 actif
  Facteur de tolérance G0 qui était effectif lors du traitement du bloc principal actuel.

- Sans arrêt du prétraitement des blocs dans le programme pièce, au moyen de la variable système :

  $P_STOLF Facteur de tolérance G0 programmé

Si aucune valeur n'est programmée avec `STOLF` dans le programme pièce actif, ces deux variables système fournissent la valeur réglée par MD20560 $MC_G0_TOLERANCE_FACTOR.

Si aucune vitesse rapide (G0) n'est active dans un bloc, ces variables système fournissent toujours la valeur 1.

Exemple

Code de programme	**Commentaire**
`COMPCAD G645 G1 F10000`	`; Fonction compresseur COMPCAD`
`X... Y... Z...`	`; Les paramètres machine et données de réglage prennent effet à ce niveau.`
`X... Y... Z...`	
`X... Y... Z...`	
`G0 X... Y... Z...`	
`G0 X... Y... Z...`	`; Le paramètre machine $MC_G0_TOLERANCE_FACTOR (par ex. =3), donc en fait une tolérance d'arrondissement de $MC_G0_TOLERANCE_FACTOR*$MA_COMPRESS_POS_TOL, prend effet à ce niveau.`
`CTOL=0.02`	
`STOLF=4`	
`G1 X... Y... Z...`	`; Une tolérance de contour de 0,02 mm prend effet à partir de ce niveau.`
`X... Y... Z...`	
`X... Y... Z...`	
`G0 X... Y... Z...`	
`X... Y... Z...`	`; Une tolérance de facteur G0 de 4, donc en fait une tolérance de contour de 0,08 mm, prend effet à partir de ce niveau.`

couplages d'axes

9

9.1 Déplacements conjugués (TRAILON, TRAILOF)

Fonction

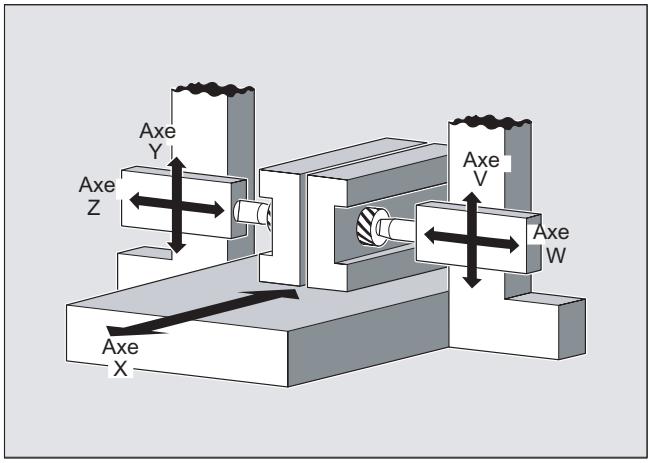
Lors du déplacement d'un axe pilote défini, les axes conjugués qui lui sont affectés (= axes asservis) décrivent les courses qui découlent de ce déplacement, compte tenu d'un facteur de couplage.

L'axe pilote et les axes asservis constituent un groupe d'axes à déplacements conjugués.

Domaines d'application

- Déplacement d'un axe par le biais d'un autre axe simulé. L'axe pilote est un axe simulé et l'axe conjugué un axe réel. Ainsi, l'axe réel peut être déplacé en tenant compte du facteur de couplage.
- Usinage bilatéral avec 2 groupe d'axes à déplacements conjugués :

  1. Axe pilote Y, axe conjugué V
  2. Axe pilote Z, axe conjugué W



Syntaxe

```
TRAILON(<axe asservi>,<axe pilote>,<facteur de couplage>)
TRAILOF(<axe asservi>,<axe pilote>,<axe pilote 2>)
TRAILOF(<axe asservi>)
```

Signification

TRAILON	Instruction pour l'activation et la définition d'un groupe d'axes à déplacements conjugués
	Prise d'effet : modale
<axe asservi>	Paramètre 1 : Désignation de l'axe conjugué (asservi)
	Nota : un axe conjugué peut également être l'axe pilote d'autres axes conjugués. Ceci permet de réaliser des groupes d'axes à déplacements conjugués interdépendants.
<axe pilote>	Paramètre 2 : Désignation de l'axe pilote
<facteur de couplage>	Paramètre 3 : facteur de couplage
	Le facteur de couplage indique le rapport souhaité entre la course de l'axe conjugué et celle de l'axe pilote :
	<facteur de couplage> = course de l'axe conjugué/course de l'axe pilote
	Type : REAL
	Paramétrage par défaut : 1
	En entrant une valeur négative, les déplacements de l'axe pilote et de l'axe conjugué s'effectuent en sens opposé.
	Si le facteur de couplage n'est pas indiqué lors de la programmation, le facteur de couplage 1 est pris en compte automatiquement.
TRAILOF	Instruction de désactivation d'un groupe d'axes à déplacements conjugués
	Prise d'effet : modale
	TRAILOF avec 2 paramètres désactive uniquement le couplage à l'axe pilote indiqué : TRAILOF(<axe asservi>,<axe pilote>)
	Si un axe conjugué possède 2 axes pilotes, TRAILOF peut être appelé avec 3 paramètres pour désactiver les deux couplages : TRAILOF(<axe asservi>,<axe pilote>,<axe pilote 2>)
	La programmation de TRAILOF sans indication d'axe pilote fournit le même résultat : TRAILOF(<axe asservi>)

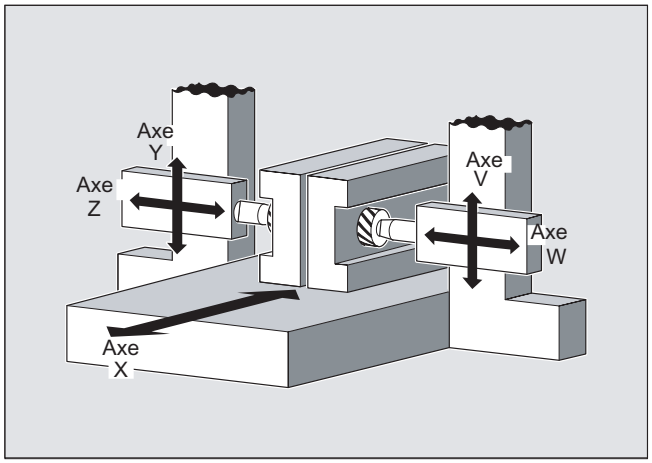
Remarque

Les déplacements conjugués s'effectuent toujours dans le système de coordonnées de base (SCB).

Le nombre de groupes d'axes à déplacements conjugués que vous pouvez activer simultanément dépend uniquement des combinaisons d'axes possibles sur la machine.

Exemple

La pièce doit être usinée simultanément sur deux faces avec la configuration d'axes représentée ci-contre. Pour ce faire, former deux groupes d'axes à déplacements conjugués.



Code de programme	Commentaire
…	
N100 TRAILON(V,Y)	; Activation du 1er groupe d'axes à déplacements conjugués
N110 TRAILON(W,Z,-1)	; Activation du 2ème groupe d'axes à déplacements conjugués. Facteur de couplage négatif : l'axe conjugué se déplace en sens opposé par rapport à l'axe pilote.
N120 G0 Z10	; Déplacement des axes Z et W en sens opposé.
N130 G0 Y20	; Déplacement des axes Y et V dans le même sens.
…	
N200 G1 Y22 V25 F200	; Superposition d'un déplacement dépendant et d'un déplacement indépendant de l'axe conjugué V.
…	
TRAILOF(V,Y)	; Désactivation du 1er groupe d'axes à déplacements conjugués.
TRAILOF(W,Z)	; Désactivation du 2ème groupe d'axes à déplacements conjugués.

Informations complémentaires

Types d'axes

Un groupe d'axes à déplacements conjugués peut combiner à volonté axes linéaires et axes rotatifs. Un axe simulé peut aussi être défini comme axe pilote.

Axes conjugués

A un axe conjugué, vous pouvez affecter deux axes pilotes à la fois. L'affectation s'effectue dans des groupes d'axes à déplacements conjugués différents.

Un axe conjugué peut être programmé avec toutes les instructions de déplacement disponibles (G0, G1, G2, G3, …). En plus des déplacements définis indépendamment, l'axe conjugué décrit les courses déduites de ses axes pilotes avec les facteurs de couplage.

Limitation dynamique

La limitation dynamique dépend du mode d'activation du groupe d'axes à déplacements conjugués :

- Activation dans le programme pièce

  Lorsque l'activation s'effectue dans le programme pièce et que tous les axes pilotes sont des axes du programme dans le canal d'activation, la dynamique de chaque axe conjugué est prise en compte lors du déplacement des axes pilotes, de sorte à ce qu'aucun axe conjugué ne soit surchargé.

  Lorsque l'activation s'effectue dans le programme pièce avec des axes pilotes qui ne sont pas des axes de programme dans le canal d'activation ($AA_TYP ≠ 1), la dynamique de l'axe conjugué n'est pas prise en compte lors du déplacement des axes pilotes. Dans le cas d'axes conjugués, dont la dynamique est inférieure à celle requise pour le couplage, il peut ainsi en résulter une surcharge.

- Activation dans une action synchrone

  Lorsque l'activation s'effectue dans une action synchrone, la dynamique des axes conjugués n'est pas prise en compte lors du déplacement des axes pilotes. Dans le cas d'axes conjugués, dont la dynamique est inférieure à celle requise pour le couplage, il peut ainsi en résulter une surcharge.

⚠ PRUDENCE

Lorsqu'un groupe d'axes à déplacements conjugués est activé dans
- des actions synchrones,
- le programme pièce avec des axes pilotes qui ne sont pas des axes de programme dans le canal de l'axe conjugué,

il relève de la responsabilité expresse de l'utilisateur/du constructeur de la machine de prévoir des mesures adaptées, afin que les déplacements de l'axe pilote n'entraînent pas de surcharge des axes conjugués.

Etat du couplage

L'interrogation de l'état du couplage d'un axe peut être réalisée dans le programme pièce au moyen de la variable système :

$AA_COUP_ACT[<axe>]

Valeur	Signification
0	aucun couplage activé
8	déplacements conjugués actifs

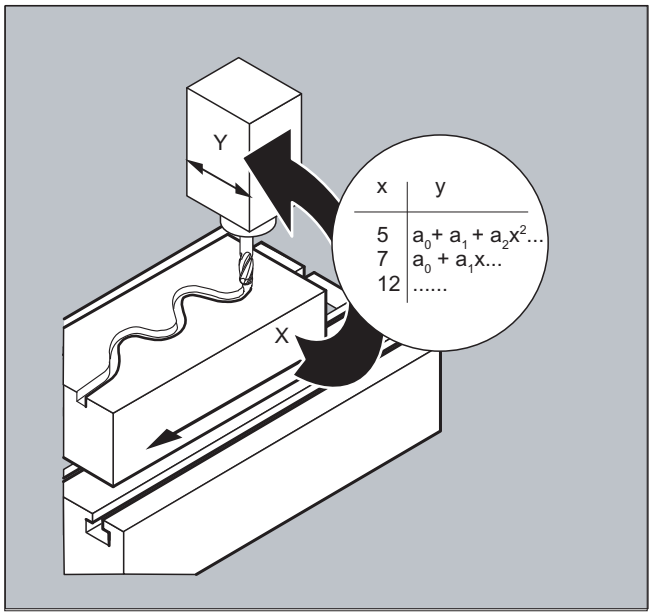
9.2 Tables de courbes (CTAB)

Fonction

Les tables de courbes permettent de programmer des rapports de position et de vitesse entre deux axes (axe pilote et axe asservi). La définition d'une table de courbes s'effectue dans le programme pièce.

Application

Les tables de courbes remplacent des cames mécaniques. la table de courbes est à la base du couplage de deux axes par valeur pilote, puisqu'elle crée le lien fonctionnel entre valeur pilote et valeur asservie : A partir des positions en corrélation réciproque de l'axe asservi et de l'axe pilote, la commande calcule conformément à la programmation, un polynôme équivalent à la came qu'il est appelé à remplacer.

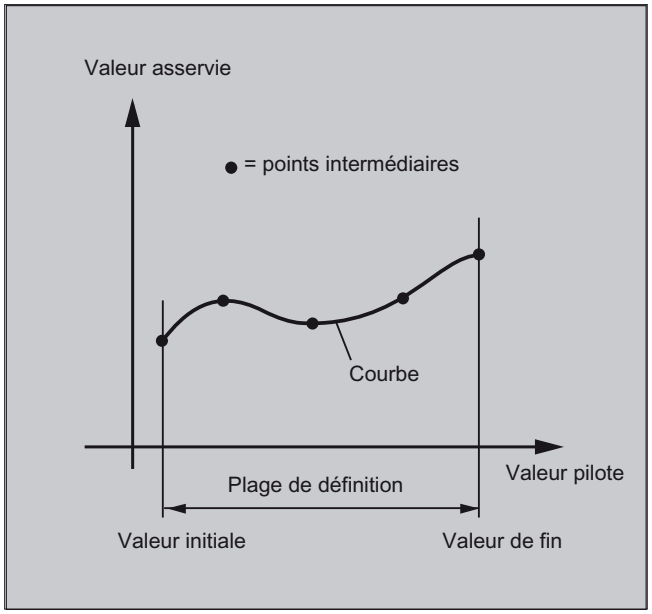


9.2.1 Définition de tables de courbes (CTABDEF, CATBEND)

Fonction

Une table de courbes représente un programme pièce ou une partie d'un programme pièce qui est identifié(e) par l'instruction `CTABDEF` en début et l'instruction `CTABEND` en fin de programme.

Dans cette section du programme pièce, on met en correspondance, par le biais d'instructions de déplacement, des positions isolées de l'axe pilote avec des positions univoques de l'axe asservi ; ces positions servent de points intermédiaires pour le calcul d'une courbe sous forme d'un polynôme du 5ème degré au maximum.



Condition

La définition de tables de courbes requiert la réservation d'un espace mémoire au moyen d'une configuration correspondante des paramètres machine (→ constructeur de la machine !).

Syntaxe

```
CTABDEF(<axe asservi>,<axe pilote>,<n>,<périodicité>[,<lieu
d'enregistrement>])
...
CTABEND
```

Signification

`CTABDEF ( )`	Début de la définition de tables de courbes
`CTABEND`	Fin de la définition de tables de courbes
`<axe asservi>`	Axe dont il s'agit de calculer le déplacement au moyen de la table de courbes
`<axe pilote>`	Axe fournissant les valeurs pilotes pour le calcul du déplacement de l'axe asservi
`<n>`	Numéro (**ID**) de la table de courbes Le numéro d'une table de courbes est univoque et indépendant du lieu d'enregistrement. Des tables portant le même numéro ne peuvent pas se trouver à la fois dans la mémoire statique et dans la mémoire dynamique de la CN.
`<périodicité>`	Périodicité de la table `0` la table n'est pas périodique (elle n'est traitée qu'une seule fois, même dans le cas d'axes rotatifs) `1` la table est périodique par rapport à l'axe pilote `2` la table est périodique par rapport à l'axe pilote et l'axe asservi
`<lieu d'enregistrement>`	Indication du lieu d'enregistrement (facultative) `"SRAM"` La table de courbes est créée dans la mémoire **statique** de la CN. `"DRAM"` La table de courbes est créée dans la mémoire **dynamique** de la CN. **Nota :** si aucune valeur n'est programmée pour ce paramètre, le lieu d'enregistrement par défaut paramétré avec MD20905 $MC_CTAB_DEFAULT_MEMORY_TYPE est utilisé.

Remarque

Ecraser

Une table de courbes est écrasée lorsque son numéro (`<n>`) est utilisé pour la nouvelle définition d'une table de courbes (exception : la table de courbes est active dans un couplage d'axes ou bloquée avec `CTABLOCK`). **L'écrasement n'est pas accompagné d'un avertissement correspondant !**

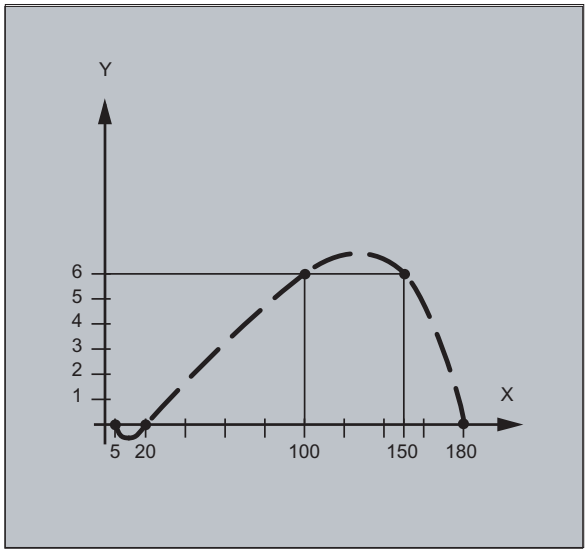
Exemples

Exemple 1 : section de programme comme définition d'une table de courbes

Une partie d'un programme doit servir, sans être modifiée, à la définition d'une table de courbes. L'instruction d'arrêt du prétraitement des blocs STOPRE qui y figure peut y être conservée et sera réactivée aussitôt que cette section de programme ne sera plus utilisée pour la définition d'une table de courbes, c'est-à-dire, dès que CTABDEF et CTABEND auront été supprimés.

```
Code de programme                Commentaire
…
CTABDEF(Y,X,1,1)                 ; Définition d'une table de courbes.
…
IF NOT ($P_CTABDEF)
STOPRE
ENDIF
…
CTABEND
```

Exemple 2 : définition d'une table de courbes non périodique




```
Code de programme          Commentaire
N100 CTABDEF(Y,X,3,0)      ; Début de la définition d'une table de courbes non
                             périodique ayant le numéro 3.
N110 X0 Y0                 ; 1ère instruction de déplacement, définit les valeurs de
                             départ et le 1er point intermédiaire :
                             valeur pilote : 0, valeur asservie : 0
N120 X20 Y0                ; 2ème point intermédiaire :
                             valeur pilote : 0…20, valeur asservie : valeur de
                             départ...0
```

```
Code de programme       Commentaire
N130 X100 Y6            ; 3ème point intermédiaire :
                          valeur pilote : 20…100, valeur asservie : 0…6
N140 X150 Y6            ; 4ème point intermédiaire :
                          valeur pilote : 100…150, valeur asservie : 6…6
N150 X180 Y0            ; 5ème point intermédiaire :
                          valeur pilote : 150…180, valeur asservie : 6…0
N200 CTABEND            ; Fin de la définition. La table de courbes est générée dans
                          sa représentation interne, en tant que polynôme du 5ème
                          degré au maximum. Le calcul de la courbe avec les points
                          intermédiaires indiqués dépend du mode d'interpolation à
                          effet modal sélectionné (interpolation circulaire,
                          linéaire, de type spline). L'état du programme pièce avant
                          le début de la définition est restauré.
```

Exemple 3 : définition d'une table de courbes périodique

Définition d'une table de courbes périodique ayant le numéro 2, une plage de valeurs pilotes de 0 à 360, un déplacement de l'axe asservi de 0 à 45 { avec retour à 0 :

```
Code de programme                          Commentaire
N10 DEF REAL DEPPOS
N20 DEF REAL GRADIENT
N30 CTABDEF(Y,X,2,1)                       ; Début de la définition.
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                               ; Fin de la définition.
; Test de la courbe par couplage de Y à 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)          ; Lecture de la fonction de la table à la
                                             valeur pilote 75.0.
N180 G0 X75 Y=DEPPOS                       ; Positionnement de l'axe pilote et de
                                             l'axe asservi.
Quand le couplage est activé, il n'est pas nécessaire de synchroniser l'axe asservi.
N190 LEADON(Y,X,2)
N200 G1 X110 F1000
N210 LEADOF(Y,X)
N220 M30
```

Informations complémentaires

Valeur de départ et valeur de fin de la table de courbes

Le couple de positions axiales corrélées que l'on indique en premier (il s'agit de la première instruction de déplacement) dans la définition de la table de courbes est considéré comme valeur de départ de la plage de définition de la table de courbes. La valeur finale de la plage de définition de la table de courbes est constituée par la dernière instruction de déplacement.

Langage disponible

Pour définir la table de courbes, vous disposez du langage CN dans son intégralité.

Remarque

Les indications suivantes ne sont pas autorisées dans les définitions de tables de courbes :

- l'arrêt du prétraitement des blocs,
- les variations brusques dans le déplacement de l'axe pilote (en cas de changement de transformation par exemple),
- une instruction de déplacement seule pour l'axe asservi,
- une inversion du déplacement de l'axe pilote, autrement dit la position de l'axe pilote ne doit jamais être ambiguë,
- les instructions `CTABDEF` et `CTABEND` dans des niveaux de programme différents.

Effet d'instructions modales

Toutes les instructions à effet modal qui interviennent dans la définition de la table de courbes deviennent caduques dès que la définition est terminée. En d'autres termes, le programme pièce dans lequel a eu lieu la définition, présente le même état avant et après la définition de la table de courbes.

Affectations à des paramètres R

Les affectations à des paramètres R dans la définition d'une table de courbes sont réinitialisées après `CTABEND`.

Exemple :

```
Code de programme                Commentaire
...
R10=5 R11=20                     ; R10=5
...
CTABDEF
G1 X=10 Y=20 F1000
R10=R11+5                        ; R10=25
X=R10
CTABEND
...                              ; R10=5
```

Activation de ASPLINE, BSPLINE, CSPLINE

Si, dans la définition d'une table de courbes CTABDEF ... CTABEND, une instruction ASPLINE, BSPLINE ou CSPLINE est activée, il est recommandé de programmer au moins un point de départ avant cette activation de courbe de type spline. Cela devrait éviter une activation immédiate après CTABDEF , compte tenu que la spline dépend de la position d'axe courante avant la définition de table de courbes.

Exemple :

Code de programme

```
...
CTABDEF(Y,X,1,0)
X0 Y0
ASPLINE
X=5 Y=10
X10 Y40
...
CTABEND
```

Réutilisation de tables de courbes

Le lien fonctionnel créé entre l'axe pilote et l'axe asservi par le biais de la table de courbes est mémorisé sous le numéro de table qui a été choisi ; il est conservé au-delà de la fin du programme et d'un POWER OFF si la table a été enregistrée dans la mémoire statique de la CN (SRAM).

Une table créée dans la mémoire dynamique (DRAM) est supprimée lors d'un POWER ON et doit éventuellement être recréée.

Une fois créée, une table de courbes est utilisable pour des combinaisons quelconques entre axe pilote et axe asservi, indépendamment des axes qui ont servi à la créer.

Ecrasement de tables de courbes

Dès que vous redéfinissez une table de courbe sous un numéro de table déjà utilisé, vous écrasez la table existante.

Exception : dans un couplage d'axes, une table de courbes est active ou bloquée avec CTABLOCK.

Remarque

L'écrasement d'une table de courbes n'est pas accompagné d'un avertissement particulier !

Définition de table de courbe active ?

La variable système $P_CTABDEF permet à tout moment d'interroger le système à partir du programme pièce, afin de savoir si une définition de table de courbes est active.

Annulation de la définition d'une table de courbes

Après la mise entre parenthèses des instructions de définition de la table de courbes, la section du programme pièce peut à nouveau être utilisée comme un programme pièce réel.

Chargement de tables de courbes via "Exécution d'un programme externe"

Pour l'exécution externe de tables de courbes, la taille du tampon de rechargement (DRAM) doit être sélectionnée via PM18360 $MN_MM_EXT_PROG_BUFFER_SIZE de sorte à ce que celui-ci puisse contenir la définition complète des tables de courbes. Si ce n'est pas le cas, l'exécution du programme pièce est annulée et une alarme est émise.

Sauts de l'axe asservi

En fonction du paramètre machine :
MD20900 $MC_CTAB_ENABLE_NO_LEADMOTION
, des sauts de l'axe asservi sont tolérables en l'absence de déplacement.

9.2.2 Vérification de l'existence d'une table de courbes (CTABEXISTS)

Fonction

L'instruction `CTABEXISTS` permet de vérifier la présence d'un numéro donné de table de courbes dans la mémoire de la CN.

Syntaxe

```
CTABEXISTS(<n>)
```

Signification

`CTABEXISTS`	Vérification de la présence de la table de courbes de numéro `<n>` dans la mémoire statique ou dynamique de la CN 0 La table n'existe pas. 1 La table existe
`<n>`	Numéro (**ID**) de la table de courbes

9.2.3 Suppression d'une table de courbes (CTABDEL)

Fonction

`CTABDEL` permet de supprimer des tables de courbes.

Remarque

Vous ne pouvez pas effacer des tables de courbes quand elles sont activées dans un couplage d'axes.

Syntaxe

```
CTABDEL(<n>)
CTABDEL(<n>,<m>)
CTABDEL(<n>,<m>,<lieu d'enregistrement>)
CTABDEL()
CTABDEL(,,<lieu d'enregistrement>)
```

Signification

`CTABDEL`	Instruction de suppression de tables de courbes
`<n>`	Numéro (**ID**) de la table de courbes à supprimer Lors de la suppression d'une plage de tables de courbes `CTABDEL(<n>,<m>)`, `<n>` indique le numéro de la première table de courbes de la plage.
`<m>`	Lors de la suppression d'une plage de tables de courbes `CTABDEL(<n>,<m>)`, `<m>` indique le numéro de la dernière table de courbes de la plage. `<m>` doit être supérieur à `<n>` !
`<lieu d'enregistrement>`	Indication du lieu d'enregistrement (facultative) Lors d'une suppression **sans** indication de lieu d'enregistrement, les tables de courbes indiquées sont supprimées dans les mémoires statique et dynamique de la CN. Lors d'une suppression **avec** indication de lieu d'enregistrement, seules les tables de courbes indiquées se trouvant dans la mémoire spécifiée sont supprimées. Les autres tables de courbes sont conservées. `"SRAM"` Suppression dans la mémoire **statique** de la CN `"DRAM"` Suppression dans la mémoire **dynamique** de la CN

Si `CTABDEL` est programmé sans indication de la table de courbes à supprimer, **toutes** les tables de courbes ou toutes celles qui se trouvent dans la mémoire spécifiée sont supprimées :

`CTABDEL()`	Supprime toutes les tables de courbes dans les mémoires statique et dynamique de la CN
`CTABDEL(,,"SRAM")`	Supprime toutes les tables de courbes dans la mémoire statique de la CN
`CTABDEL(,,"DRAM")`	Supprime toutes les tables de courbes dans la mémoire dynamique de la CN

Remarque

Si, en cas de suppression multiple `CTABDEL(<n>,<m>)` ou `CTABDEL()`, l'une au moins des tables de courbes à supprimer est active dans un couplage, l'instruction de suppression n'est pas exécutée, autrement dit **aucune** des tables de courbes adressées n'est supprimée.

9.2.4 Verrouillage de tables de courbes contre la suppression et l'écrasement (CTABLOCK, CTABUNLOCK)

Fonction

Les tables de courbes peuvent être protégées contre la suppression et l'écrasement par inadvertance au moyen d'un verrouillage. Ce dernier peut à tout moment de nouveau être annulé.

Syntaxe

Verrouillage :

```
CTABLOCK(<n>)
CTABLOCK(<n>,<m>)
CTABLOCK(<n>,<m>,<lieu d'enregistrement>)
CTABLOCK()
CTABLOCK(,,<lieu d'enregistrement>)
```

Annulation du verrouillage :

```
CTABUNLOCK(<n>)
CTABUNLOCK(<n>,<m>)
CTABUNLOCK(<n>,<m>,<lieu d'enregistrement>)
CTABUNLOCK()
CTABUNLOCK(,,<lieu d'enregistrement>)
```

Signification

CTABLOCK	Instruction de **verrouillage** contre la suppression/l'écrasement
CTABUNLOCK	Instruction d'**annulation** du verrouillage contre la suppression/l'écrasement CTABUNLOCK libère les tables de courbes verrouillées avec CTABLOCK. Les tables de courbe qui agissent dans un couplage actif restent verrouillées et ne peuvent pas être supprimées. Le verrouillage avec CTABLOCK est annulé aussitôt que le couplage actif est désactivé. Cette table peut donc être effacée. Un nouvel appel de CTABUNLOCK n'est pas nécessaire.
<n>	Numéro (**ID**) de la table de courbes à verrouiller/déverrouiller Lors du verrouillage/déverrouillage d'une plage de tables de courbes CTABLOCK(<n>,<m>)/CTABUNLOCK(<n>,<m>), <n> indique le numéro de la première table de courbes de la plage.
<m>	Lors du verrouillage/déverrouillage d'une plage de tables de courbes CTABLOCK(<n>,<m>)/CTABUNLOCK(<n>,<m>), <m> indique le numéro de la dernière table de courbes de la plage. <m> doit être supérieur à <n> !

`<lieu d'enregistrement>`	Indication du lieu d'enregistrement (facultative) Lors du verrouillage/déverrouillage **sans** indication de lieu d'enregistrement, les tables de courbes indiquées sont verrouillées/déverrouillées dans les mémoires statique et dynamique de la CN. Lors du verrouillage/déverrouillage **avec** indication de lieu d'enregistrement, seules les tables de courbes indiquées se trouvant dans la mémoire spécifiée sont verrouillées/déverrouillées. Les tables de courbes restantes ne sont pas verrouillées/déverrouillées. `"SRAM"` Verrouillage/déverrouillage dans la mémoire **statique** de la CN `"DRAM"` Verrouillage/déverrouillage dans la mémoire **dynamique** de la CN

Si `CTABLOCK`/`CTABUNLOCK` est programmé sans indication de la table de courbes à verrouiller/déverrouiller, **toutes** les tables de courbes ou toutes celles qui se trouvent dans la mémoire spécifiée sont verrouillées/déverrouillées.

`CTABLOCK()`	Verrouille toutes les tables de courbes dans les mémoires statique et dynamique de la CN
`CTABLOCK(,,"SRAM")`	Verrouille toutes les tables de courbes dans la mémoire statique de la CN
`CTABLOCK(,,"DRAM")`	Verrouille toutes les tables de courbes dans la mémoire dynamique de la CN
`CTABUNLOCK()`	Déverrouille toutes les tables de courbes dans les mémoires statique et dynamique de la CN
`CTABUNLOCK(,,"SRAM")`	Déverrouille toutes les tables de courbes dans la mémoire statique de la CN
`CTABUNLOCK(,,"DRAM")`	Déverrouille toutes les tables de courbes dans la mémoire dynamique de la CN

9.2.5 Tables de courbes : détermination des propriétés de la table (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD)

Fonction

Ces instructions permettent d'interroger les principales propriétés d'une table de courbes (numéro de la table, état de verrouillage, lieu d'enregistrement, périodicité).

Syntaxe

```
CTABID(<p>)
CTABID(<p>,<lieu d'enregistrement>)
CTABISLOCK(<n>)
CTABMEMTYP(<n>)
TABPERIOD(<n>)
```

Signification

CTABID	Indique le **numéro de la table** inscrit comme <p>ème table de courbes dans la mémoire spécifiée. Exemple : CTABID(1,"SRAM") indique le numéro de la première table de courbes dans la mémoire statique de la CN. La première table de courbes correspond à celle dont le numéro est le plus élevé. **Nota :** si l'ordre des tables de courbes est modifié dans la mémoire entre des appels consécutifs de CTABID, p. ex. pour la suppression de tables de courbes avec CTABDEL, CTABID(<p>,...) peut indiquer une autre table de courbes que précédemment avec le même numéro <p>.
CTABISLOCK	Indique l'**état de verrouillage** de la table de courbes portant le numéro <n> : 0 La table n'est pas verrouillée 1 La table est verrouillée par CTABLOCK 2 La table est verrouillée par le couplage actif 3 La table est verrouillée par CTABLOCK et par le couplage actif -1 La table n'existe pas.
CTABMEMTYP	Indique le **lieu d'enregistrement** de la table de courbes portant le numéro <n> : 0 table dans la mémoire statique de la CN 1 table dans la mémoire dynamique de la CN -1 La table n'existe pas.
CTABPERIOD	Indique la **périodicité** de la table de courbes portant le numéro <n> : 0 la table n'est pas périodique 1 la table est périodique dans l'axe pilote 2 la table est périodique dans l'axe pilote et dans l'axe asservi -1 La table n'existe pas.
<p>	Numéro d'entrée en mémoire
<n>	Numéro (**ID**) de la table de courbes
<lieu d'enregistrement>	Indication du lieu d'enregistrement (facultative) "SRAM" Mémoire **statique** de la CN "DRAM" Mémoire **dynamique** de la CN **Nota :** si aucune valeur n'est programmée pour ce paramètre, le lieu d'enregistrement par défaut paramétré avec MD20905 $MC_CTAB_DEFAULT_MEMORY_TYPE est utilisé.

9.2.6 Lecture des valeurs de la table de courbes (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX)

Fonction

Les valeurs suivantes de la table de courbes peuvent être lues dans le programme pièce :

- Valeurs de l'axe asservi et de l'axe pilote au début et à la fin d'une table des courbes
- Valeurs de l'axe asservi au début et à la fin d'un segment de courbe
- Valeur de l'axe asservi correspondant à une valeur de l'axe pilote
- Valeur de l'axe pilote correspondant à une valeur de l'axe asservi
- Valeur minimale et valeur maximale de l'axe asservi
  - dans la plage de définition entière de la table de courbes

    ou
  - dans un intervalle défini de la table de courbes

Syntaxe

```
CTABTSV(<n>,<gradient>[,<axe asservi>])
CTABTEV(<n>,<gradient>[,<axe asservi>])
CTABTSP(<n>,<gradient>[,<axe pilote>])
CTABTEP(<n>,<gradient>[,<axe pilote>])
CTABSSV(<valeur pilote>,<n>,<gradient>[,<axe asservi>])
CTABSEV(<valeur pilote>,<n>,<gradient>[,<axe asservi>])
CTAB(<valeur pilote>,<n>,<gradient>[,<axe asservi>,<axe pilote>]
CTABINV(<valeur asservie>,<valeur approchée>,<n>,<gradient>[,<axe
asservi>,<axe pilote>]
CTABTMIN(<n>[,<axe asservi>])
CTABTMAX(<n>[,<axe asservi>])
CTABTMIN(<n>,<a>,<b>[,<axe asservi>,<axe pilote>])
CTABTMAX(<n>,<a>,<b>[,<axe asservi>,<axe pilote>])
```

Signification

`CTABTSV` :	Lecture de la valeur de l'axe asservi au **début** de la table de courbes portant le numéro `<n>`
`CTABTEV` :	Lecture de la valeur de l'axe asservi à la **fin** de la table de courbes portant le numéro `<n>`
`CTABTSP` :	Lecture de la valeur de l'axe pilote au **début** de la table de courbes portant le numéro `<n>`
`CTABTEP` :	Lecture de la valeur de l'axe pilote à la **fin** de la table de courbes portant le numéro `<n>`
`CTABSSV` :	Lecture de la valeur de l'axe asservi au **début** du segment de courbe appartenant à la valeur de l'axe pilote spécifié (`<valeur pilote>`)
`CTABSEV` :	Lecture de la valeur de l'axe asservi à la **fin** du segment de courbe appartenant à la valeur de l'axe pilote spécifié (`<valeur pilote>`)

CTAB :	Lecture de la valeur de l'axe asservi correspondant à la valeur spécifiée de l'axe pilote (<valeur pilote>)
CTABINV :	Lecture de la valeur de l'axe pilote correspondant à la valeur spécifiée de l'axe asservi (<valeur asservie>)
CTABTMIN :	Définition de la **valeur minimale** de l'axe asservi : • dans la plage de définition entière de la table de courbes ou • dans un intervalle défini <a> ... <b>
CTABTMAX :	Définition de la **valeur maximale** de l'axe asservi : • dans la plage de définition entière de la table de courbes ou • dans un intervalle défini <a> ... <b>
<n> :	Numéro (**ID**) de la table de courbes
<Gradient> :	Le paramètre <gradient> indique la **pente** de la fonction représentant la table de courbes à la position déterminée
<Axe asservi> :	Axe dont il s'agit de calculer le déplacement au moyen de la table de courbes (facultatif)
<Axe pilote> :	Axe fournissant les valeurs pilotes pour le calcul du déplacement de l'axe asservi (facultatif)
<Valeur asservie> :	Valeur de l'axe asservi pour la lecture de la valeur correspondante de l'axe pilote pour CTABINV
<Valeur pilote> :	Valeur de l'axe pilote : • pour la lecture de la valeur correspondante de l'axe asservi pour CTAB ou • pour la sélection du segment de courbe pour CTABSSV/CTABSEV
<Valeur approchée> :	L'affectation d'une valeur de l'axe pilote à une valeur de l'axe asservi pour CTABINV ne doit pas toujours être univoque. C'est la raison pour laquelle CTABINV requiert, comme paramètre, une valeur approchée de la valeur recherchée de l'axe pilote.
<a> :	Limite **inférieure** de l'intervalle des valeurs pilotes pour CTABTMIN/CTABTMAX
<b> :	Limite **supérieure** de l'intervalle des valeurs pilotes pour CTABTMIN/CTABTMAX **Nota :** l'intervalle des valeurs pilotes <a> ... <b> doit se situer dans la plage de définition de la table de courbes.

Exemples

Exemple 1 :

Déterminer les valeurs de l'axe asservi et de l'axe pilote au début et à la fin de la table de courbes ainsi que les valeurs minimale et maximale de l'axe asservi dans la plage de définition entière de la table de courbes.

```
Code de programme                    Commentaire
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)              ; Début de la définition de la table
N110 X0 Y10                        ; Position de départ du 1er segment
N120 X30 Y40                       ; Position finale du 1er segment de la table = position de départ
                                     du 2ème segment de la table
N130 X60 Y5                        ; Position finale du 2ème segment de table = ...
N140 X70 Y30
N150 X80 Y20
N160 CTABEND                       ; Fin de la définition de la table.
...
N200 STARTPOS=CTABTSV(1,GRADIENT)  ; Valeur de l'axe asservi au début de la table de courbes = 10
N210 ENDPOS=CTABTEV(1,GRADIENT)    ; Valeur de l'axe asservi à la fin de la table de courbes = 20
N220 STARTPARA=CTABTSP(1,GRADIENT) ; Valeur de l'axe pilote au début de la table de courbes = 0
N230 ENDPARA=CTABTEP(1,GRADIENT)   ; Valeur de l'axe pilote à la fin de la table de courbes = 80
N240 MINVAL=CTABTMIN(1)            ; Valeur minimale de l'axe asservi pour Y=5
N250 MAXVAL=CTABTMAX(1)            ; Valeur maximale de l'axe asservi pour Y=40
```

Exemple 2 :

Déterminer les valeurs de l'axe asservi au début et à la fin du segment de courbe correspondant à la valeur de l'axe pilote X=30.

```
Code de programme                    Commentaire
N10 DEF REAL STARTPOS
N20 DEF REAL ENDPOS
N30 DEF REAL GRADIENT
...
N100 CTABDEF(Y,X,1,0)              ; Début de la définition de la table.
N110 X0 Y0                         ; Position de départ du 1er segment
N120 X20 Y10                       ; Position finale du 1er segment de la table = position
                                     de départ du 2ème segment de la table
```

```
Code de programme                                  Commentaire
N130 X40 Y40                                         Position finale du 2ème segment de table = ...
N140 X60 Y10
N150 X80 Y0
N160 CTABEND                                       ; Fin de la définition de la table.
...
N200 STARTPOS=CTABSSV(30.0,1,GRADIENT)             ; Position de départ Y dans le 2ème segment = 10
N210 ENDPOS=CTABSEV(30.0,1,GRADIENT)               ; Position finale Y dans le 2ème segment = 40
```

Informations complémentaires

Utilisation dans des actions synchrones

Il est également possible de lire dans des actions synchrones l'ensemble des instructions de lecture des valeurs d'une table de courbes (voir aussi le chapitre "Actions synchrones au déplacement").

Pour l'utilisation des instructions `CTABINV`, `CTABTMIN` et `CTABTMAX`, il faut s'assurer que :

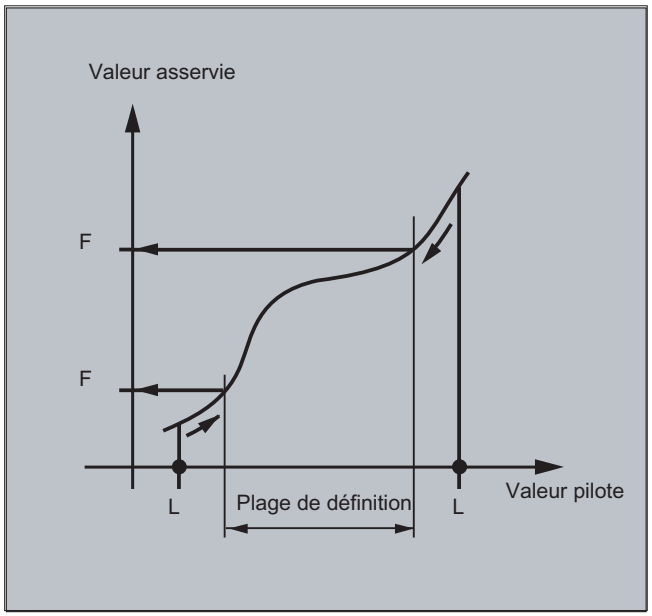
- la puissance de la CN soit suffisante au moment de l'exécution

  ou

- le nombre de segments de la table de courbes soit interrogé avant l'appel afin de pouvoir subdiviser, le cas échéant, la table concernée

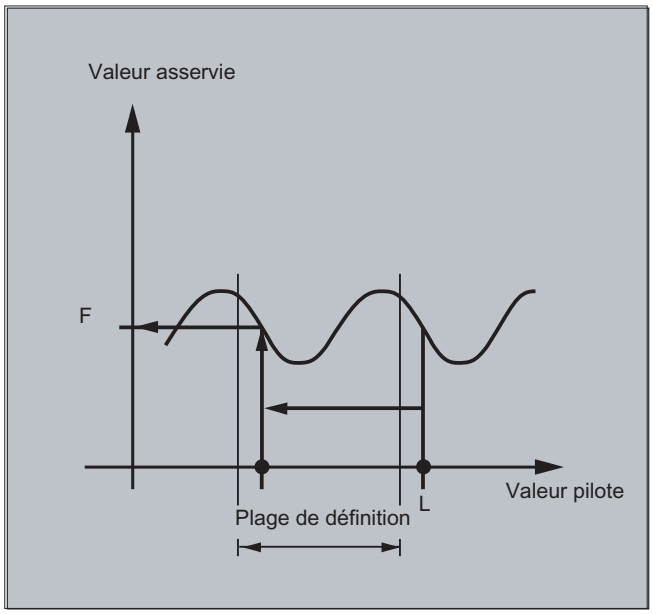
CTAB dans le cas de tables de courbes non périodiques

Lorsque la `<valeur pilote>` se situe en-dehors de la plage de définition, la valeur asservie fournie est la limite supérieure ou inférieure :



CTAB dans le cas de tables de courbes périodiques

Lorsque la <valeur pilote> indiquée se situe en-dehors de la plage de définition, la valeur pilote modulo de la plage de définition est évaluée et la valeur asservie correspondante est fournie :



Valeur approchée pour CTABINV

Pour fournir la valeur pilote recherchée, l'instruction CTABINV requiert une valeur approchée. CTABINV fournit alors la valeur pilote la plus proche de la valeur approchée. Il peut p. ex. s'agir de la valeur pilote de la dernière période d'appel de l'interpolateur.

Pente de la fonction représentant la table de courbes

L'indication de la pente (<gradient>) permet de calculer la vitesse de l'axe pilote ou de l'axe asservi à la position correspondante.

Indication de l'axe pilote ou de l'axe asservi

L'indication facultative de l'axe pilote ou de l'axe asservi est nécessaire lorsque l'axe pilote et l'axe asservi sont configurés avec des unités de longueur différentes.

CTABSSV, CTABSEV

Dans les cas suivants, les instructions CTABSSV et CTABSEV ne conviennent **pas** pour l'interrogation des segments programmés :

- des cercles ou des développantes sont programmés,
- des chanfreins ou des arrondis sont activés avec CHF/RND,
- l'arrondissement avec G643 est activé,
- la compression de blocs de la CN avec COMPON/COMPCURV/COMPCAD est activée.

9.2.7 Tables de courbes : Vérification de l'utilisation des ressources (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL)

Fonction

Ces instructions permettent au programmeur de s'informer sur l'occupation actuelle des ressources requises par les tables de courbes, segments de tables et polynômes.

Syntaxe

```
CTABNO
CTABNOMEM(<lieu d'enregistrement>)
CTABFNO(<lieu d'enregistrement>)
CTABSEGID(<n>,<lieu d'enregistrement>)
CTABSEG(<lieu d'enregistrement>,<type de segment>)
CTABFSEG(<lieu d'enregistrement>,<type de segment>)
CTABMSEG(<type de segment>,<type de segment>)
CTABPOLID(<n>)
CTABPOL(<lieu d'enregistrement>)
CTABFPOL(<lieu d'enregistrement>)
CTABMPOL(<lieu d'enregistrement>)
```

Signification

`CTABNO`	Détermination du nombre total de tables de courbes **définies** (dans les mémoires statique et dynamique de la CN)
`CTABNOMEM`	Détermination du nombre de tables de courbes **définies** dans le `<lieu d'enregistrement>` indiqué
`CTABFNO`	Détermination du nombre de tables de courbes qu'il est **encore possible** d'enregistrer dans le `<lieu d'enregistrement>` indiqué
`CTABSEGID`	Détermination du nombre de segments de courbe avec le `<type de segment>` spécifié qu'il est possible d'utiliser dans la table de courbes de numéro `<n>`
`CTABSEG`	Détermination du nombre de segments de courbe **utilisés** avec le `<type de segment>` spécifié dans le `<lieu d'enregistrement>` indiqué
`CTABFSEG`	Détermination du nombre de segments de courbe avec le `<type de segment>` spécifié qu'il est **encore possible** d'enregistrer dans le `<lieu d'enregistrement>` indiqué
`CTABMSEG`	Détermination du nombre **maximum possible** de segments de courbe avec le `<type de segment>` spécifié qu'il est possible d'enregistrer dans le `<lieu d'enregistrement>` indiqué
`CTABPOLID`	Détermination du nombre de polynômes de courbe utilisés par la table de courbes de numéro `<n>`
`CTABPOL`	Détermination du nombre de polynômes de courbes **utilisés** dans le `<lieu d'enregistrement>` indiqué

CTABFPOL	Détermination du nombre de polynômes de courbes qu'il est **encore possible** d'enregistrer dans le <lieu d'enregistrement> indiqué
CTABMPOL	Détermination du nombre **maximum possible** de polynômes de courbes qu'il est possible d'enregistrer dans le <lieu d'enregistrement> indiqué
<n>	Numéro (**ID**) de la table de courbes
<lieu d'enregistrement>	Indication du lieu d'enregistrement (facultative) "SRAM" Mémoire **statique** de la CN "DRAM" Mémoire **dynamique** de la CN **Nota :** si aucune valeur n'est programmée pour ce paramètre, le lieu d'enregistrement par défaut paramétré avec MD20905 $MC_CTAB_DEFAULT_MEMORY_TYPE est utilisé.
<type de segment>	Indication du type de segment (facultative) "L" Segments linéaires "P" Segments polynômiaux **Nota :** si aucune valeur n'est programmée pour ce paramètre, la somme des segments linéaires et polynômiaux est fournie.

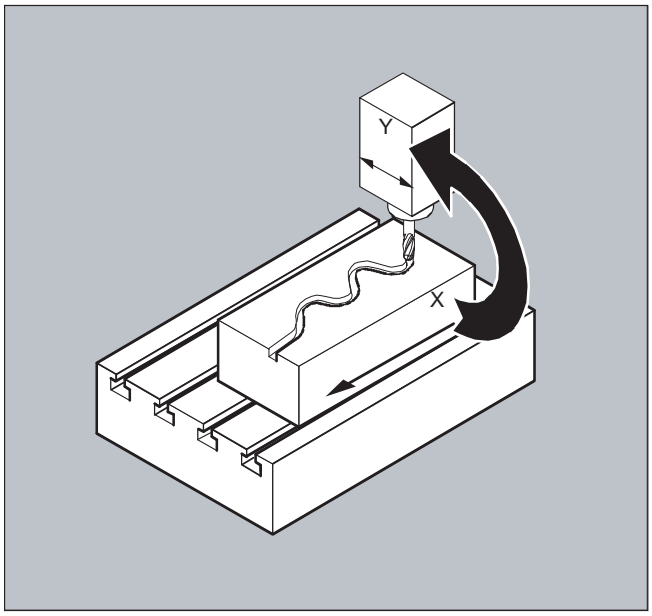
9.3 Couplage de deux axes par valeur pilote (LEADON, LEADOF)

Remarque

Cette fonction n'est pas disponible pour SINUMERIK 828D.

Fonction

Quand il y a couplage de deux axes par valeur pilote, l'axe pilote et l'axe asservi se déplacent de façon synchrone. Une position donnée de l'axe asservi correspond sans équivoque à une position de l'axe pilote (simulée le cas échéant) sur la base d'une table de courbes ou d'un polynôme en découlant.



Axe pilote est le nom donné à l'axe qui fournit les valeurs d'entrée pour la table de courbes.
Axe asservi est le nom donné à l'axe qui prend les positions calculées sur la base de la table de courbes.

Couplage par valeur réelle et couplage par valeur de consigne

Comme valeurs pilotes, autrement dit comme valeurs de départ pour déterminer les positions de l'axe asservi, on peut utiliser :

- les valeurs réelles de l'axe pilote : couplage par valeur réelle
- Valeurs de consigne de la position de l'axe pilote : Couplage par valeur de consigne

Le couplage par valeur pilote se réfère toujours au système de coordonnées de base.

Pour créer des tables de courbes, voir chapitre "Tables de courbes".
Pour le couplage par valeur pilote, voir /FB/, M3, Déplacements conjugués et couplage par valeur pilote.

Syntaxe

```
LEADON(axeA,axeP,n)
LEADOF(axeA,axeP)
```

ou désactivation sans indication de l'axe pilote :

```
LEADOF(AxeA)
```

Le couplage par valeur pilote peut être activé et désactivé aussi bien depuis le programme pièce que pendant les déplacements depuis une action synchrone (voir chapitre "Actions synchrones au déplacement").

Signification

`LEADON`	Activer le couplage par valeur pilote
`LEADOF`	Désactiver le couplage par valeur pilote
`AxeA`	axe conjugué
`AxeP`	Axe pilote
`n`	Numéro de la table de courbe
`$SA_LEAD_TYPE`	Permutation entre le couplage par valeur réelle et le couplage par valeur de consigne

Désactiver le couplage par valeur pilote, LEADOF

Après la désactivation du couplage par valeur pilote, l'axe asservi redevient axe de commande normal.

Couplage de deux axes par valeur pilote et différents états de fonctionnement, RESET

Selon le réglage qui a été fait dans les paramètres machine, les couplages par valeur pilote sont désactivés avec RESET.

Exemple de couplage par valeur pilote à partir d'une action synchrone

Sur une presse, un couplage mécanique conventionnel entre un axe pilote (axe d'emboutissage) et les axes d'un système de transfert est à remplacer par un couplage électronique.

Il démontre comment on peut remplacer sur une presse un système de transfert mécanique par un système de transfert électronique. Les phases de couplage et de découplage sont réalisées sous forme **d'actions synchrones statiques**.

Les axes de transfert et les axes auxiliaires sont pilotés par l'axe pilote VP (axe d'emboutissage) en tant qu'axes asservis, par le biais des tables de courbes.

Axes asservis

X - Axe d'avance ou axe longitudinal
YL - Axe de fermeture ou axe transversal
ZL - Axe de course
U - Avance par rouleaux, axe auxiliaire
V - Tête de dressage, axe auxiliaire
W - Graissage, axe auxiliaire

Actions

Parmi les actions synchrones, on trouve :

- Couplage, `LEADON (axe asservi, axe pilote, numéro de la table de courbe)`
- Découplage, `LEADOF (axe asservi, axe pilote)`
- Préréglage des mémoires de valeurs réelles, `PRESETON (axe, valeur)`
- Définition du mémento, `$AC_MARKER[i]= valeur`
- Type de couplage : valeur pilote réelle/virtuelle
- Accostage de positions d'axes, `POS[axe]=valeur`

Conditions

Des entrées TOR rapides, des variables temps réel `$AC_MARKER` et des comparaisons de positions, combinées avec l'opérateur logique AND, sont exploitées comme conditions.

Remarque

Dans l'exemple qui suit, nous avons utilisé des retours de lignes, des retraits et des caractères **gras** pour une meilleure lisibilité du programme. Pour la commande, tout ce qui figure sous un numéro de ligne représente effectivement une ligne de programme.

Commentaire

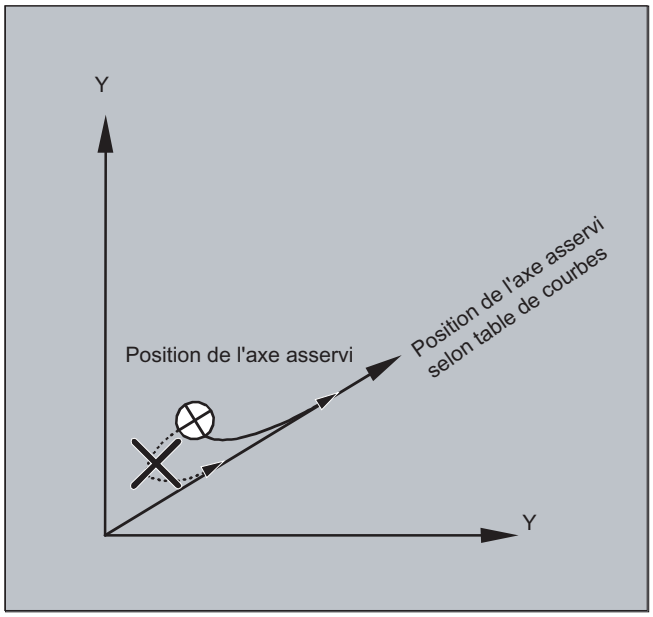
Code de programme	Commentaire
	; Définit toutes les actions synchrones statiques.
	; ****Remise à zéro des mémentos
N2 $AC_MARKER[0]=0 $AC_MARKER[1]=0 $AC_MARKER[2]=0 $AC_MARKER[3]=0 $AC_MARKER[4]=0 $AC_MARKER[5]=0 $AC_MARKER[6]=0 $AC_MARKER[7]=0	
	; **** E1 0=>1 Couplage transfert ACTIF
N10 IDS=1 EVERY ($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) $AC_MARKER[0]=1	
	; **** E1 0=>1 Couplage dispositif d'avance à rouleaux ACTIF
N20 IDS=11 EVERY ($A_IN[1]==1) AND ($A_IN[5]==0) AND ($AC_MARKER[5]==0) DO LEADON(U,LW,4) PRESETON(U,0) $AC_MARKER[5]=1	
	; **** E1 0->1 Couplage tête de dressage ACTIF
N21 IDS=12 EVERY ($A_IN[1]==1) AND ($A_IN[5]==0) AND ($AC_MARKER[6]==0) DO LEADON(V,LW,4) PRESETON(V,0) $AC_MARKER[6]=1	
	; **** E1 0->1 Couplage graissage ACTIF
N22 IDS=13 EVERY ($A_IN[1]==1) AND ($A_IN[5]==0) AND ($AC_MARKER[7]==0) DO LEADON(W,LW,4) PRESETON(W,0) $AC_MARKER[7]=1	
	; **** E2 0=>1 Couplage NON ACTIF
N30 IDS=3 EVERY ($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	

Description

Le couplage par valeur pilote exige la synchronisation de l'axe pilote et de l'axe asservi. Cette synchronisation est obtenue uniquement si l'axe asservi se trouve, au moment où le couplage par valeur pilote est activé, dans la plage de tolérance de la courbe calculée avec la table.

La plage de tolérance pour la position de l'axe asservi est définie par le paramètre machine
`PM 37200  : COUPLE_POS_POL_COARSE A_LEAD_TYPE .`

Si, lors de l'activation du couplage par valeur pilote, l'axe asservi n'est pas encore positionné correctement, le synchronisme s'enclenchera automatiquement dès que la position de consigne calculée pour l'axe asservi approchera de sa position effective. Pendant la synchronisation, l'axe asservi est déplacé dans la direction définie par sa vitesse de consigne (calculée à partir de la vitesse de l'axe pilote et CTAB).

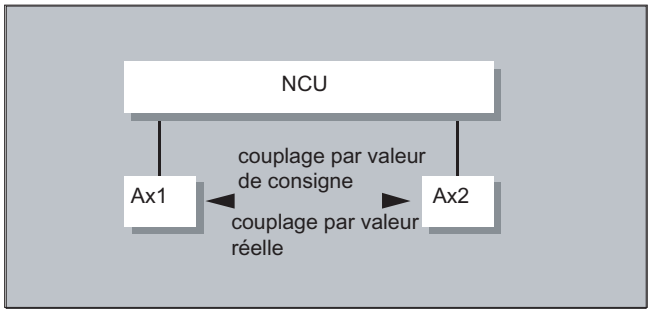


Pas de synchronisation

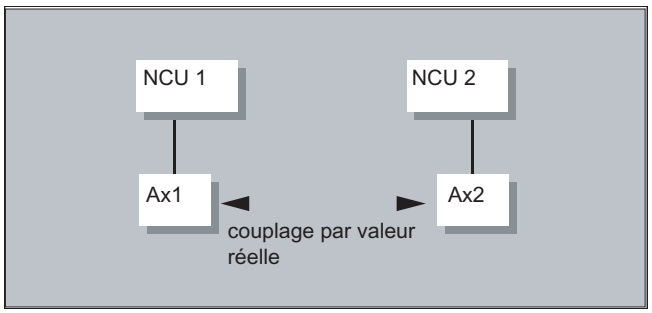
Si la position calculée de l'axe asservi s'éloigne de sa position effective lors de l'activation du couplage par valeur pilote, il n'y a pas de synchronisation.

Couplage par valeur réelle et couplage par valeur de consigne

Comparé au couplage par la valeur réelle, le couplage par la valeur de consigne fournit un meilleur synchronisme entre l'axe pilote et l'axe asservi ; c'est la raison pour laquelle il est préréglé en version standard.



Le couplage par la valeur de consigne est possible uniquement si l'axe pilote et l'axe asservi sont interpolés par la même NCU. Quand l'axe pilote est un axe externe, le couplage de l'axe asservi avec cet axe pilote ne peut se faire que par des valeurs réelles.



Le **basculement** est possible par le biais de la donnée de réglage `$SA_LEAD_TYPE`.

Le basculement entre couplage par valeur réelle et couplage par valeur de consigne doit toujours se faire quand l'axe asservi est en état d'immobilisation. En effet, une resynchronisation après le basculement n'a lieu que dans cet état.

Exemple d'application

Quand la machine est soumise à de fortes vibrations, la lecture des valeurs réelles ne peut pas se faire correctement. Pour mettre en œuvre le couplage par valeur pilote dans un transfert sur presse, il peut être nécessaire de basculer du couplage par valeur réelle au couplage par valeur de consigne, dans les phases d'usinage où les vibrations sont les plus fortes.

Simulation de valeurs pilotes dans un couplage par valeur de consigne

Par le biais d'un paramètre machine, on peut séparer l'interpolateur de l'axe pilote et le servo. On peut alors, dans un couplage par valeur de consigne, créer des valeurs de consigne sans qu'il y ait un déplacement réel de l'axe pilote.

Les valeurs pilotes créées par un couplage par la valeur de consigne peuvent être lues dans les variables suivantes, afin d'être utilisées dans des actions synchrones par exemple :

```
- $AA_LEAD_P                                    Valeur pilote de position
- $AA_LEAD_V                                    Valeur pilote de vitesse
```

Création de valeurs pilotes

Vous pouvez aussi créer des valeurs pilotes avec d'autres procédés que vous avez vous-même programmés. Les valeurs pilotes ainsi générées sont écrites dans les variables

```
- $AA_LEAD_SP                                   Valeur pilote de position
- $AA_LEAD_SV                                   Valeur pilote de vitesse
```

et lues à partir de celles-ci. Pour utiliser ces variables, il convient de définir la donnée de réglage comme suit : `$SA_LEAD_TYPE = 2`.

Etat du couplage

Vous pouvez interroger l'état du couplage dans le programme pièce CN avec la variable système suivante :

`$AA_COUP_ACT[axe]`
0 : aucun couplage activé
16 : couplage par valeur pilote activé

Gestion des états pour les actions synchrones

Les phases de commutation et de couplage sont gérées par des variables temps réel :

`$AC_MARKER[i] = n`
avec :
numéro de mémento i
valeur d'état n

9.4 Réducteur électronique (EG)

Fonction

La fonction "Réducteur Electronique" permet, après un bloc de déplacement linéaire, de faire piloter le déplacement d'un **axe asservi** en fonction de 5 **axes pilotes** au maximum. Les relations entre les axes pilotes et l'axe asservi sont définis, pour chaque axe pilote, par un facteur de couplage.

Le trajet de l'axe asservi est calculé en additionnant les différents trajets des axes pilotes multipliés par les facteur de couplage respectifs. Lors de l'activation d'un groupe d'axes EG, une synchronisation de l'axe asservi à une position définie peut avoir lieu. Un groupe de réducteurs électroniques peut être :

- défini,
- activé,
- désactivé,
- effacé

.

Le déplacement de l'axe asservi peut être déduit des :

- valeurs de consignes des axes pilotes,
- valeurs réelles des axes pilotes.

Dans le cadre de l'extension, des couplages non linéaires entre les axes pilotes et les axes asservis peuvent être réalisés par des **tables de courbes** (voir chapitre Mode de déplacement). Les réducteurs électroniques peuvent être montés en cascade, c'est-à-dire que l'axe asservi d'un réducteur électronique peut être l'axe pilote d'un autre réducteur électronique.

9.4.1 Définition d'un réducteur électronique (EGDEF)

Fonction

Un groupe d'axes EG est déterminé par l'indication de l'axe asservi et d'au moins un (au plus cinq) axes pilotes avec le type de couplage respectif.

Conditions requises

Condition pour la définition d'un groupe d'axes EG :

un couplage d'axe ne doit pas être défini pour l'axe asservi (le cas échéant, un couplage existant doit être effacé avec `EGDEL`).

Syntaxe

```
EGDEF(axe asservi,axe pilote1,type de couplage1,axe pilote2,type de
couplage2,...)
```

Signification

`EGDEF`	Définition d'un réducteur électronique
`axe conjugué`	Axe influencé par les axes pilotes
`Axe pilote1 ,...,axe pilote5`	Axes qui influencent l'axe asservi
`Type de couplage1 ,...,type de couplage5`	Type de couplage Les types de couplage pouvant être différents, ceux-ci doivent être indiqués pour tous les axes pilotes.

Valeur :	Signification :
0	L'axe asservi est influencé par la **valeur réelle** de l'axe pilote correspondant.
1	L'axe asservi est influencé par la **consigne** de l'axe pilote correspondant.

Remarque

Les facteurs de couplage sont préréglés avec la valeur zéro lors de la définition du groupe d'axes EG.

Remarque

`EGDEF` déclenche un arrêt du prétraitement des blocs. La définition de transmission avec `EGDEF` est alors à utiliser de manière inchangée si, sur les systèmes, un ou plusieurs axes pilotes influencent l'axe asservi par des **tables de courbes**.

Exemple

Code de programme	Commentaire
`EGDEF(C,B,1,Z,1,Y,1)`	`; Définition d'un groupe d'axes EG. Les axes pilotes B, Z, Y influencent l'axe asservi C par la valeur de consigne.`

9.4.2 Activation du réducteur électronique (EGON, EGONSYN, EGONSYNE)

Fonction

Il existe 3 variantes pour activer un groupe d'axes EG.

Syntaxe

Variante 1 :

Le groupe d'axes EG est activé de façon sélective sans synchronisation par :

```
EGON(AA,"mode de changement de
bloc",AP1,Z1,N1,AP2,Z2,N2,...,AP5,Z5,N5)
```

Variante 2 :

Le groupe d'axes EG est activé de façon sélective avec synchronisation par :

```
EGONSYN(AA,"mode de changement de
bloc",SynPosAA,[,APi,SynPosAPi,Zi,Ni])
```

Variante 3 :

Le groupe d'axes EG est activé de façon sélective avec synchronisation et le mode d'accostage est défini par :

```
EGONSYNE(AA,"mode de changement de bloc",SynPosAA,mode
d'accostage[,APi,SynPosAPi,Zi,Ni])
```

Signification

Variante 1 :

`AA`	axe asservi	
`Mode de changement de bloc`	Les modes suivants peuvent être utilisés :	
	`"NOC"`	Le changement de bloc s'effectue immédiatement.
	`"FINE"`	Le changement de bloc s'effectue au "synchronisme fin".
	`"COARSE"`	Le changement de bloc s'effectue au "synchronisme grossier".
	`"IPOSTOP"`	Le changement de bloc s'effectue à la valeur de consigne du synchronisme.
`AP1, ... AP5`	Axes pilotes	
`Z1, ... Z5`	Numérateur pour le facteur de couplage i	
`N1, ... N5`	Dénominateur pour le facteur de couplage i	
	Facteur de couplage i = numérateur i / dénominateur i	

Seuls les axes pilotes spécifiés auparavant dans l'instruction `EGDEF` peuvent être programmés. Un axe pilote au moins doit être indiqué.

Variante 2 :

AA	axe asservi	
Mode de changement de bloc	Les modes suivants peuvent être utilisés :	
	"NOC"	Le changement de bloc s'effectue immédiatement.
	"FINE"	Le changement de bloc s'effectue au "synchronisme fin".
	"COARSE"	Le changement de bloc s'effectue au "synchronisme grossier".
	"IPOSTOP"	Le changement de bloc s'effectue à la valeur de consigne du synchronisme.
[,APi,SynPosAPi,Zi,Ni]	(ne pas écrire les crochets droits)	
	Au minimum 1, au maximum 5 suites de :	
AP1, ... AP5	Axes pilotes	
PosSynAPi	Position de synchronisation pour l'axe pilote i	
Z1, ... Z5	Numérateur pour le facteur de couplage i	
N1, ... N5	Dénominateur pour le facteur de couplage i	
	Facteur de couplage i = numérateur i / dénominateur i	

Seuls les axes pilotes spécifiés auparavant dans l'instruction `EGDEF` peuvent être programmés. Les "positions de synchronisme" programmés pour l'axe asservi (`PosSynAA`) et pour les axes pilotes (`PosSynAP`) sont des positions dans lesquelles le couplage d'axes est considéré *synchronisé*. Si le réducteur électronique n'est pas synchronisé à l'activation, l'axe asservi se déplace à la position de synchronisation définie.

Variante 3 :

Les paramètres sont identiques à ceux de la variante 2 avec en plus :

Mode d'accostage	Les modes suivants peuvent être utilisés :	
	"NTGT"	accoster l'entredent suivant avec une optimisation du temps
	"NTGP"	accoster l'entredent suivant avec une optimisation du parcours
	"ACN"	déplacer l'axe rotatif en sens de rotation négatif, en absolu
	"ACP"	déplacer l'axe rotatif en sens de rotation positif, en absolu
	"DCT"	Position de synchronisme programmée avec une optimisation du temps
	"DCP"	Position de synchronisme programmée avec une optimisation du parcours

La variante 3 influe uniquement sur les axes asservis à modulos couplés aux axes pilotes à modulos. L'optimisation du temps tient compte des limites de vitesse de l'axe asservi.

Autres informations

Description des variantes d'activation

Variante 1 :

Les positions des axes pilotes et des axes asservis sont enregistrées, lors de l'activation, en tant que "positions de synchronisme". Les "positions de synchronisme" peuvent être lues à l'aide de la variable `$AA_EG_SYN`.

Variante 2 :

Si le groupe d'axes comprend des axes à valeur modulo, leurs positions sont ramenés à la plage définie par la valeur modulo. Ceci garantit que la position de synchronisation la plus proche est accostée (*synchronisation relative* : par exemple l'entredent suivant). Si le signal d'interface DB (30 +numéro d'axe), DBX 26 bit 4 "Déblocage forçage déplacement axe asservi" n'est pas à 1 pour l'axe asservi, la position de synchronisation n'est pas accostée. Dans ce cas, le programme est arrêté au bloc EGONSYN et l'alarme à effacement automatique 16771 est signalée jusqu'à ce que le signal mentionné ci-dessus soit mis à 1.

Variante 3 :

L'entredent (degrés) est calculé de la manière suivante : 360 * Zi/Ni. Si l'axe asservi est à l'arrêt au moment de l'activation, l'optimisation du parcours génère le même comportement que l'optimisation du temps.

Si l'axe asservi est déjà en déplacement, `NTGP` est synchronisé avec l'entredent suivant indépendamment de sa vitesse actuelle. Si l'axe asservi est déjà en déplacement, `NTGT` est synchronisé avec l'entredent suivant en fonction de sa vitesse actuelle. Le cas échéant, la vitesse est réduite.

Tables de courbes

Si une **table de courbes** est utilisée pour un axe pilote, il faut :

Ni	mettre à 0 le dénominateur du facteur de couplage linéaire. (Le dénominateur 0 serait illicite pour des couplages linéaires). Pour la commande, le dénominateur 0 est l'identificateur qui
Zi	doit être interprété comme numéro de table de courbes à utiliser. La table de courbes avec le numéro indiqué doit être déjà défini lors de l'activation.
APi	L'indication de l'axe pilote correspond à l'indication de l'axe pilote lors du couplage par facteur (couplage linéaire).

Vous trouverez d'autres informations concernant l'utilisation de tables de courbes, la cascade de réducteurs électroniques et leur synchronisation dans :
Bibliographie :
Description fonctionnelle Fonctions spéciales ;couplages d'axes et ESR (M3), chapitre "Déplacements conjugués et couplage par valeur pilote".

Comportement d'un réducteur électronique en cas de mise sous tension, RESET, changement de mode de fonctionnement, recherche de bloc

- Après la mise sous tension, **aucun** couplage n'est actif.
- Les couplages actifs sont conservés après RESET et changement de mode de fonctionnement.
- A recherche de bloc, les instructions d'activation, de désactivation, d'effacement et de définition du réducteur électronique ne sont ni exécutées ni accumulées, mais sautées.

Variables système du réducteur électronique

A l'aide des variables système du réducteur électronique, vous pouvez déterminer, dans le programme pièce, les états courants d'un groupe d'axes EG et, le cas échéant, déclencher des réactions adéquates.

Les variables système du réducteur électronique sont repérées de la manière suivante :

$AA_EG_ ...

ou

$VA_EG_ ...

Bibliographie:
Manuel des variables système

9.4.3 Désactivation du réducteur électronique (EGOFS, EGOFC)

Fonction

Il existe 3 variantes pour désactiver un groupe d'axes EG actif.

Programmation

Variante 1 :

Syntaxe	Signification
`EGOFS(axe asservi)`	Le réducteur électronique est désactivé. L'axe asservi est freiné jusqu'à l'arrêt. Cette instruction déclenche un arrêt du prétraitement des blocs.

Variante 2 :

Syntaxe	Signification
`EGOFS(axe asservi,axepilote1,…,axepilote5)`	Ce paramétrage permet de supprimer **de façon sélective** l'influence de certains axes pilotes sur le déplacement de l'axe asservi.

Un axe pilote au moins doit être indiqué. L'influence des axes pilotes indiqués sur l'axe asservi est supprimée. Cette instruction déclenche un arrêt du prétraitement des blocs. S'il reste des axes pilotes actifs, l'axe asservi continue de se déplacer sous leur influence. Lorsque toutes les influences d'axes pilotes sont désactivées de cette manière, l'axe asservi est freiné jusqu'à l'arrêt.

Variante 3 :

Syntaxe	Signification
`EGOFC(broche asservie1)`	Le réducteur électronique est désactivé. La broche asservie continue de tourner à la vitesse qu'elle a à l'instant de la désactivation. Cette instruction déclenche un arrêt du prétraitement des blocs.

Remarque

Cette variante n'est autorisée que pour des broches.

9.4.4 Suppression de la définition d'un réducteur électronique (EGDEL)

Fonction

La définition d'un groupe d'axes EG ne peut être effacée que si celui-ci a été désactivé auparavant.

Programmation

Syntaxe	Signification
`EGDEL(axe asservi)`	La définition du couplage du groupe d'axes est effacée. Vous pouvez à nouveau définir des groupes d'axes avec `EGDEF` jusqu'à ce que le nombre maximal de groupes d'axes actif simultanément soit atteint. Cette instruction déclenche un arrêt du prétraitement des blocs.

9.4.5 Avance par tour (G95) / réducteur électronique (FPR)

Fonction

L'instruction `FPR` permet également d'indiquer l'axe asservi d'un réducteur électronique en tant qu'axe déterminant l'avance par tours. Dans ce cas, les règles suivantes s'appliquent :

- L'avance dépend de la vitesse de consigne de l'axe asservi du réducteur électronique.
- La vitesse de consigne est calculée à partir des vitesses des broches pilotes et des axes pilotes à valeur modulo (qui ne sont pas des axes à interpolation) et des facteurs de couplage correspondants.
- Les vitesses des axes pilotes linéaires et des axes pilotes sans valeur modulo ainsi que les déplacements forcé de l'axe asservi ne sont pas pris en considération.

9.5 Broche synchrone

Fonction

En mode de "broche synchrone", il y a une broche pilote (BP) et une broche asservie (BA), qui forment la **paire de broches synchrones**. Quand le couplage est activé (mode "broche synchrone"), la broche asservie suit les déplacements de la broche pilote, en fonction du lien fonctionnel qui a été défini auparavant.

Les paires de broches synchrones peuvent être aussi bien configurées de manière fixe pour chaque machine à l'aide de paramètres machine spécifiques à un canal ou définies en fonction de l'application par le programme pièce de la CNC. Deux paires de broches synchrones sont utilisables simultanément par canal CN.

A partir du programme pièce, le couplage peut être

- défini ou modifié,
- activé,
- désactivé,
- effacé.

De plus, en fonction de la version du logiciel,

- les conditions de synchronisme peuvent être contrôlées,
- le mode de changement de bloc peut être modifié,
- le type de couplage, soit couplage par valeur de consigne soit couplage par valeur réelle, peut être sélectionné ou le décalage angulaire entre broche pilote et broche asservi peut être prescrit
- à l'activation du couplage, une programmation précédente de la broche asservie peut être reprise
- un écart de synchronisme mesuré ou encore déjà connu peut être corrigé

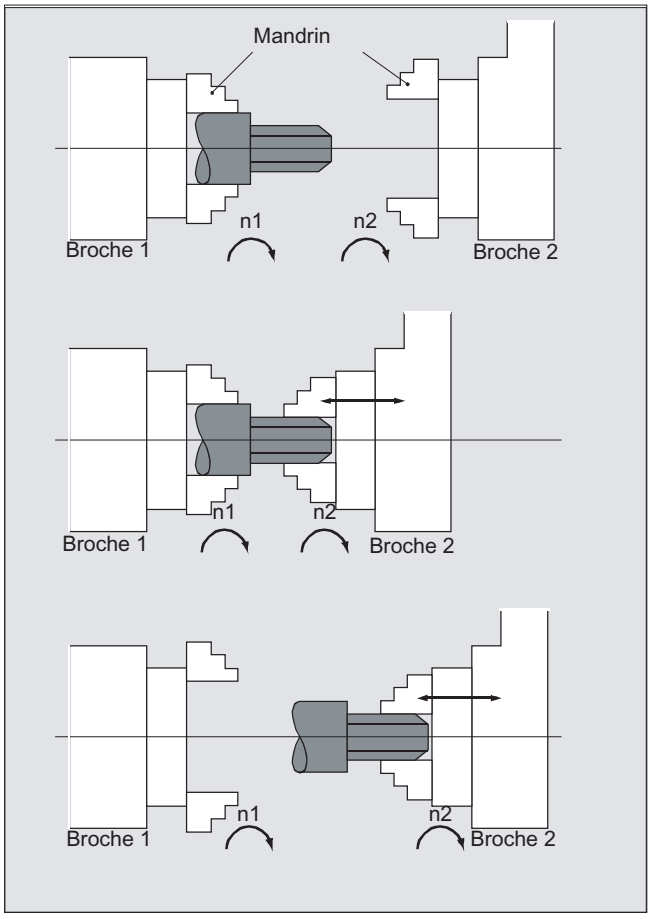
9.5.1 Broche synchrone : Programmation (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC)

Fonction

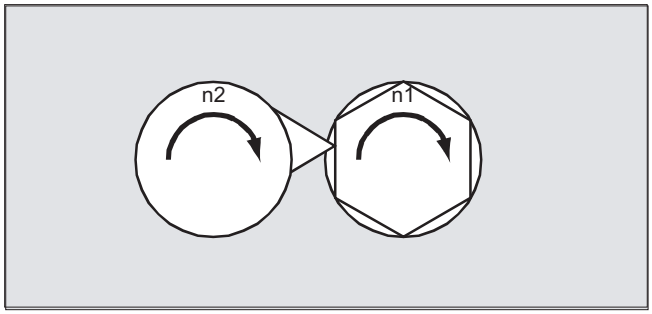
La fonction de broche synchrone permet un déplacement synchrone de deux broches (broche asservie BA et broche pilote BP), par ex. pour un transfert de pièce à la volée.

Cette fonction dispose des modes suivants :

- Synchronisme de vitesse de rotation ($n_{BA} = n_{BP}$)
- Synchronisme de position ($\phi_{BA} = \phi_{BP}$)
- Synchronisme de position avec décalage angulaire ($\phi_{BA} = \phi_{BP}$)



La prédéfinition d'un rapport de transmission différent de 1 entre l'axe pilote et l'axe asservi permet également le traitement de plusieurs faces (rotation de polygône).



Syntaxe

```
COUPDEF(<BA>,<BP>,<RBA>,<RBP>,<Changement de bloc>,<Mode de
couplage>)
COUPON(<BA>,<BP>,<POSBA>)
COUPONC(<BA>,<BP>)
COUPOF(<BA>,<BP>,<POSBA>,<POSBP>)
COUPOFS(<BA>,<BP>)
COUPOFS(<BA>,<BP>,<POSBA>)
COUPRES(<BA>,<BP>)
COUPDEL(<BA>,<BP>)
WAITC(<BA>,<Changement de bloc>,<BP>,<Changement de bloc>)
```

Remarque

Notation abrégée

Les instructions COUPOF, COUPOFS, COUPRES et COUPDEL permettent l'utilisation d'une notation abrégée sans indication de la broche pilote.

Signification

COUPDEF :	Définition/modification spécifique à l'utilisateur du couplage
COUPON :	activer le couplage. La broche asservie se synchronise sur la broche pilote à partir de la vitesse de rotation actuelle
COUPONC :	Prise en charge du couplage à l'activation par la programmation précédente de M3 S... ou M4 S.... Une vitesse de rotation différentielle de la broche asservie est aussitôt prise en charge.
COUPOF :	Désactiver le couplage. • avec changement de bloc immédiat : COUPOF(<S2>,<S1>) • Changement de bloc seulement après dépassement de la ou des positions de coupure <POSBA> ou <POSBP> : COUPOF(<S2>,<S1>,<POSBA>) COUPOF(<S2>,<S1>,<POSBA>,<POSBP>)

COUPOFS :	Désactivation d'un couplage avec arrêt de la broche asservie
	Changement de bloc aussi rapide que possible avec changement de bloc immédiat : COUPOFS(<S2>,<S1>)
	Changement de bloc uniquement après le passage de la position d'arrêt : COUPOFS(<S2>,<S1>,<POSBA>)
COUPRES :	Remettre à zéro les paramètres de couplage sur les PM et SD configurés
COUPDEL :	effacer le couplage défini par l'utilisateur
WAITC :	Attente de la condition de synchronisme (NOC sont supprimés sur l'interpolateur au changement de bloc)
<BA> :	Désignation de la broche asservie

Paramètres optionnels :

<BP> :	Désignation de la broche pilote
	Indication avec numéro de broche : par ex. S2, S1
<RBA>, <RBP> :	Rapport de transmission entre BA et BP.
	<RBA> = numérateur, <RBP> = dénominateur
	Réglage par défaut : <RBA> / <RBP> = 1.0 ; indication facultative du dénominateur
<Changement de bloc> :	Mode de changement de bloc
	Le changement de bloc s'effectue :
	"NOC" sur le champ,
	"FINE" lorsque le "Synchronisme fin" est atteint
	"COARSE" lorsque le "Synchronisme grossier" est atteint
	"IPOSTOP" lorsque IPOSTOP est atteint, c.-à-d. quand la valeur de consigne du synchronisme est atteinte (paramétrage par défaut)
	Le mode de changement de bloc est à effet modal.
<Mode de couplage> :	Type de couplage : couplage entre BA et BP
	"DV" Couplage par la valeur de consigne (préréglage)
	"AV" Couplage par valeur réelle
	"VV" Couplage de vitesse
	Le type de couplage est à effet modal.
<POSBA> :	décalage angulaire entre broche pilote et broche asservie
	Plage de valeurs : 0°… 359,999°
<POSBA>,<POSBP> :	Positions d'arrêt de broche asservie et de broche pilote
	"Le changement de bloc est débloqué après le passage par POS_{BA}, POS_{BP}."
	Plage de valeurs : 0°… 359,999°

Exemples

Exemple 1 : Travail avec broche pilote et broche asservie

```
Programmation                          Commentaire
                                       ; Broche pilote = broche maître = broche 1
                                       ; Broche asservie = broche 2
N05 M3 S3000 M2=4 S2=500               ; La broche pilote tourne à 3000 tr/min,
                                         la broche asservie tourne à 500 tr/min.
N10 COUPDEF(S2,S1,1,1,"NOC","Dv")      ; Définition du couplage (peut également être configuré).
...
N70 SPCON                              ; Placement de la broche pilote en asservissement de
                                         position (contrôle de consigne)
N75 SPCON(2)                           ; Broche asservie intégrée dans la régulation de position.
N80 COUPON(S2,S1,45)                   ; Couplage au vol sur position de décalage = 45 degrés.
...
N200 AA[S2]=100                        ; Vitesse de positionnement = 100 degrés/min
N205 SPOS[2]=IC(-90)                   ; Déplacement forcé de 90 degrés dans le sens négatif.
N210 WAITC(S2,"Fine")                  ; Attente du synchronisme "fin".
N212 G1 X... Y... F...                 ; Usinage
...
N215 SPOS[2]=IC(180)                   ; Déplacement forcé de 180 degrés dans le sens positif.
N220 G4 S50                            ; Durée d'attente = 50 tours de la broche maître
N225 AA[S2]=0                          ; Activation de la vitesse configurée (PM).
N230 SPOS[2]=IC(-7200)                 ; 20 tours. Déplacement dans le sens négatif à la vitesse
                                         configurée.
...
N350 COUPOF(S2,S1)                     ; Découplage au vol, S=S2=3000
N355 SPOSA[2]=0                        ; Arrêt de la broche asservie à 0 degrés.
N360 G0 X0 Y0
N365 WAITS(2)                          ; Attendre la broche 2.
N370 M5                                ; Arrêter la broche asservie.
N375 M30
```

Exemple 2 : Programmation d'une vitesse de rotation différentielle

```
Programmation              Commentaire
                           ;  Broche pilote = broche maître = broche 1
                           ;  Broche asservie = broche 2
N01 M3 S500                ;  Broche pilote tourne à 500 tr/min.
N02 M2=3 S2=300            ;  Broche asservie tourne à 300 tr/min.
...
N10 G4 F1                  ;  Temporisation de la broche maître.
N15 COUPDEF (S2,S1,-1)     ;  Facteur de couplage avec un rapport de transmission de -1:1
N20 COUPON (S2,S1)         ;  Activer le couplage. La vitesse de rotation de la broche asservie est
                              définie à partir de la vitesse de rotation de la broche pilote et du
                              facteur de couplage.
...
N26 M2=3 S2=100            ;  Programmation d'une vitesse de rotation différentielle.
```

Exemple 3 : Exemples de prise en charge d'un déplacement sur la vitesse de rotation différentielle

1. Activer le couplage lors de la programmation précédente de la broche asservie avec `COUPON`

```
Programmation              Commentaire
                           ;  Broche pilote = broche maître = broche 1
                           ;  Broche asservie = broche 2
N05 M3 S100 M2=3 S2=200    ;  Broche pilote tourne à 100 tr/min, broche asservie tourne à 200 tr/min.
N10 G4 F5                  ;  Temporisation = 5 secondes de la broche maître
N15 COUPDEF(S2,S1,1)       ;  Le rapport de transmission BA par rapport à BP est de 1,0 (réglage par
                              défaut).
N20 COUPON(S2,S1)          ;  Couplage au vol sur la broche pilote.
N10 G4 F5                  ;  Broche asservie tourne à 100 tr/min.
```

2. Activer le couplage lors de la programmation précédente de la broche asservie avec `COUPONC`

```
Programmation              Commentaire
                           ;  Broche pilote = broche maître = broche 1
                           ;  Broche asservie = broche 2
N05 M3 S100 M2=3 S2=200    ;  Broche pilote tourne à 100 tr/min, broche asservie tourne à 200 tr/min.
N10 G4 F5                  ;  Temporisation = 5 secondes de la broche maître
N15 COUPDEF(S2,S1,1)       ;  Le rapport de transmission BA par rapport à BP est de 1,0 (réglage par
                              défaut).
N20 COUPONC(S2,S1)         ;  Couplage au vol sur la broche pilote et application de la vitesse de
                              rotation précédente à S2.
N10 G4 F5                  ;  S2 tourne à 100 tr/min + 200 tr/min = 300 tr/min
```

3. Activer le couplage avec la broche asservie à l'arrêt avec COUPON

```
Programmation                    Commentaire
                                 ; Broche pilote = broche maître = broche 1
                                 ; Broche asservie = broche 2
N05 SPOS=10 SPOS[2]=20           ; Broche asservie S2 en mode de positionnement.
N15 COUPDEF(S2,S1,1)             ; Le rapport de transmission BA par rapport à BP est de 1,0 (réglage
                                   par défaut).
N20 COUPON(S2,S1)                ; Couplage au vol sur la broche pilote.
N10 G4 F1                        ; Le couplage est fermé, S2 s'arrête à 20 degrés.
```

4. Activer le couplage avec la broche asservie à l'arrêt avec COUPONC

Remarque

Mode de positionnement ou mode axe

Si la broche asservie se trouve en amont du couplage en mode de positionnement ou en mode axe, la broche asservie se comporte de la même façon pour COUPON(<BA>,<BP>) et pour COUPONC(<BA>,<BP>).

IMPORTANT

Broche pilote et mode d'axe

Si la broche pilote se trouve en mode axe avant que le couplage ne soit défini, la valeur limite de vitesse transmise par le paramètre machine suivant agit également après activation du couplage :

MD32000 $MA_MAX_AX_VELO (vitesse d'axe maximale)

Afin d'éviter ce comportement, l'axe doit être commuté en mode broche (M3 S... ou M4 S...) avant la définition du couplage.

Informations complémentaires

Définir la paire de broches synchrones

Couplage configuré :

Dans le cas du couplage configuré, la broche pilote et la broche asservie sont définies dans les paramètres machine. Les broches configurées ne peuvent pas être modifiées dans le programme pièce. Le paramétrage du couplage peut être réalisé dans le programme pièce avec COUPDEF(condition : qu'aucune protection en écriture ne soit définie).

Couplage défini par l'utilisateur :

COUPDEF permet de définir un nouveau couplage ou de modifier un couplage existant dans le programme pièce. Si un couplage est déjà actif, il doit d'abord être supprimé avec COUPDEL avant la définition d'un nouveau couplage.

Définition du couplage : COUPDEF

Un couplage est de façon complète avec :

COUPDEF(<BA>,<BP>,<RBA>,<RBP>, comportement de changement de bloc, mode de couplage)

Broche asservie (BA) et broche pilote (BP)

Les noms d'axe de la broche asservie (BA) et de la broche pilote (BP) définissent le couplage sans ambiguïté. Ils doivent être programmés dans chaque instruction `COUPDEF`. Les autres paramètres de couplage sont à effet modal et doivent uniquement être programmés en cas de modification.

Exemple :

```
COUPDEF(S2,S1)
```

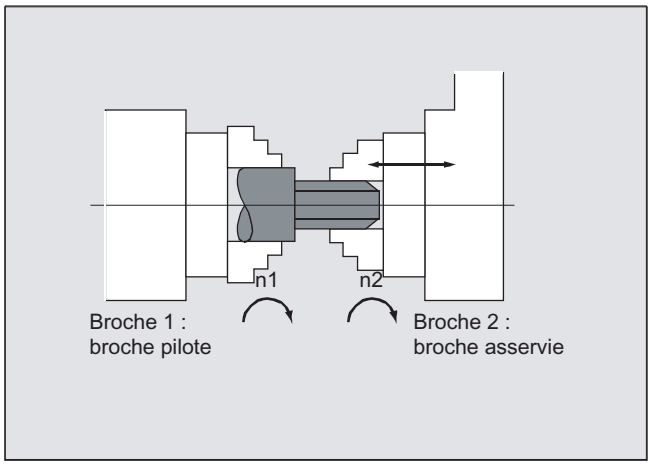
Rapport de transmission RBA / RBP

Le rapport de transmission est indiqué comme rapport de vitesse de rotation entre l'axe asservi (numérateur) et l'axe pilote (dénominateur). Le numérateur doit être programmé. Si aucun dénominateur n'est programmé, le dénominateur = 1.0 est appliqué.

Exemple :

broche asservie S2 et broche pilote S1, rapport de transmission = 1 / 4 = 0.25.

```
COUPDEF(S2,S1,1.0, 4.0)
```



Remarque

Le rapport de transmission peut également être modifié pendant que le couplage est activé et que les broches sont en rotation.

Mode de changement de bloc NOC, FINE, COARSE, IPOSTOP

Pour la programmation du mode de changement de bloc, la notation abrégée suivante est autorisée :

- `"NO"` : immédiat (préréglage)
- `"FI"` : lorsque le "Synchronisme fin" est atteint
- `"CO"` : lorsque le "Synchronisme grossier" est atteint
- `"IP"` : lorsque IPOSTOP est atteint, c.-à-d. quand la valeur de consigne du synchronisme est atteinte

Type de couplage DV, AV

⚠ PRUDENCE

La modification du type de couplage ne peut se faire que lorsque le couplage est désactivé !

Activation du mode synchrone COUPON, POSBA

- Activation du couplage avec une référence angulaire quelconque entre BP et BA :
  - `COUPON(S2,S1)`
  - `COUPON(S2,S1,<POSBA>)`
  - `COUPON(S2)`
- Activation du couplage avec un décalage angulaire <`POSBA>`

  Couplage synchronisé en position pour les pièces profilées.
  `<POSBA>` se réfère à la position 0° de la broche pilote dans le sens de rotation positif
  Plage de valeurs `<POSBA>` : 0°… 359,999°
  - `COUPON(S2,S1,30)`

  Vous pouvez ainsi modifier le décalage angulaire, alors que le couplage est déjà actif.

Positionnement de la broche asservie

Quand le couplage de broches synchrones est activé, on peut toujours encore positionner des broches asservies dans une plage de ±180°, indépendamment du déplacement déclenché par la broche pilote.

Positionnement SPOS

La broche asservie peut être interpolée avec `SPOS =...`

Exemple :

```
SPOS[2]=IC(-90)
```

Pour plus d'informations sur `SPOS`, se référer à la :
Bibliographie :
Manuel de programmation Notions de base

Vitesse de rotation différentielle M3 S... ou M4 S...

Une vitesse de rotation différentielle résulte d'une superposition avec signe de deux sources de vitesse de rotation et est à nouveau programmée pour la broche asservie, par ex. avec `S<n>=...` ou `M<n>=3`, `M<n>=4` en mode de régulation de vitesse de rotation pendant un couplage actif de broche synchrone. La composante de vitesse de rotation est déduite par le facteur de couplage et la broche asservie est additionnée de manière algébrique.

Remarque

Avec le sens de rotation `M3` ou `M4`, la vitesse de rotation `S...` doit également être à nouveau programmée afin d'éviter la signalisation de la programmation manquante par une alarme.

Pour plus d'informations sur la vitesse de rotation différentielle, se référer à la :
Bibliographie :
Description fonctionnelle Fonctions d'extension ; broche synchrone (S3)

Vitesse de rotation différentielle pour COUPONC

Prise en charge d'un déplacement sur la vitesse de rotation différentielle

L'activation d'un couplage de broche synchrone avec `COUPONC` superpose une vitesse de rotation actuellement active à la broche asservie (`M3 S...` ou `M4 S...`).

Remarque

Déblocage de la superposition

La superposition d'une vitesse de rotation de broche (`M3 S...` ou `M4 S...`) par couplage de broche synchrone `COUPONC` est uniquement effective si la superposition est libérée.

Restriction dynamique de la broche pilote

La dynamique de la broche pilote doit être restreinte de sorte à ce que les valeurs limites dynamiques de la broche asservie superposée ne soient pas dépassées.

Vitesse, accélération : FA, ACC, OVRA, VELOLIMA

La programmation de la vitesse et de l'accélération axiales d'une broche asservie s'effectue avec :

- `FA[SPI(S<n>)]` ou `FA[S<n>]` (vitesse axiale)
- `ACC[SPI(S<n>)]` ou `ACC[S<n>]` (accélération axiale)
- `OVRA[SPI(S<n>)]` ou `OVRA[S<n>]` (correction axiale)
- `VELOLIMA[SPI(S<n>)]` ou `VELOLIMA[S<n>]` (augmentation ou réduction de la vitesse axiale)

Avec `<n>` = 1, 2, 3, ... (numéro des broches asservies)

Bibliographie :
Manuel de programmation Notions de base

Remarque

Composante de l'accélération JERKLIMA[S<n>]

La programmation d'un accroissement ou d'une réduction de la vitesse axiale n'a actuellement pas d'effet pour les broches.

Pour plus d'informations sur la configuration de la dynamique axiale, se référer à la :
Bibliographie :
Description fonctionnelle Fonctions d'extension ; axes rotatifs (R2)

Comportement programmable de changement de bloc WAITC

`WAITC` permet de prédéfinir le comportement de changement de bloc en fonction de différentes conditions de synchronisme (grossier, fin, `IPOSTOP`), p. ex. après une modification des paramètres de couplage ou des opérations de positionnement. Si aucune condition de synchronisme n'est spécifiée, la comportement de changement de bloc indiqué dans la définition de `COUPDEF` est appliqué.

Exemple :

Attendre que la condition de synchronisme définie dans `COUPDEF` soit atteinte

```
WAITC( )
```

Attendre que la condition de synchronisme définie dans FINE soit atteinte pour la broche asservie S2 et celle définie dans COARSE pour la broche asservie S4 :

```
WAITC(S2,"FINE",S4,"COARSE")
```

Désactivation du couplage COUPOF

COUPOF permet de prédéfinir le comportement de désactivation du couplage :

- Désactivation du couplage avec changement de bloc immédiat :
  - COUPOF(S2,S1) (avec indication de la broche pilote)
  - COUPOF(S2) (sans indication de la broche pilote)
- Désactivation du couplage après le passage de la position d'arrêt. Le changement de bloc a lieu après le passage de la position d'arrêt.
  - COUPOF(S2,S1,150) (position d'arrêt BA : 150°)
  - COUPOF(S2,S1,150,30) (position d'arrêt BA : 150°, BP : 30°)

Désactivation du couplage avec arrêt de la broche asservie COUPOFS

COUPOFS permet de prédéfinir le comportement de désactivation du couplage avec arrêt de la broche asservie :

- Désactivation du couplage avec arrêt de la broche asservie et changement de bloc immédiat :
  - COUPOFS(S2,S1) (avec indication de la broche pilote)
  - COUPOFS(S2) (sans indication de la broche pilote)
- Désactivation du couplage après le passage de la position d'arrêt avec arrêt de la broche asservie. Le changement de bloc a lieu après le passage de la position d'arrêt.
  - COUPOFS(S2,S1,150) (position d'arrêt BA : 150°)

Effacement de couplages COUPDEL

COUPDEL permet de supprimer le couplage :

- COUPDEL(S2,S1) (avec indication de la broche pilote)
- COUPDEL(S2) (sans indication de la broche pilote)

Remise à zéro des paramètres de couplage COUPRES

COUPRES active les valeurs du couplage définies dans les paramètres machine et des données de réglage :

- COUPRES(S2,S1) (avec indication de la broche pilote)
- COUPRES(S2) (sans indication de la broche pilote)

Variables système

Etat de couplage courant de la broche asservie

La lecture de l'état de couplage courant d'une broche asservie peut s'effectuer avec la variable système suivante :

```
$AA_COUP_ACT[<FS>]
```

Valeur	Signification
0	Aucun couplage activé
4	le couplage de broches synchrones est activé
Nota les autres valeurs de la variable système concernent le mode de l'axe **Bibliographie :** Tables de paramètres variables système	

Décalage angulaire courant

La lecture du décalage angulaire courant d'une broche asservie par rapport à la broche pilote peut s'effectuer avec la variable système suivante :

- `$AA_COUP_OFFS[<FS>]` (valeur de consigne du décalage angulaire)
- `$VA_COUP_OFFS[<FS>]` (valeur effective du décalage angulaire)

Remarque

Si le déblocage des régulateurs est supprimé alors que le couplage et le fonctionnement en poursuite sont actifs, le décalage de position change par rapport à la valeur initialement programmée, lorsque le déblocage des régulateurs est à nouveau validé. Dans ce cas, le décalage de position modifié peut être lu et éventuellement corrigé dans le programme pièce.

9.6 Couplage maître/esclave (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS)

Fonction

Avant la version 6.4 du logiciel, le couplage des axes esclaves avec un axe maître ne peut s'effectuer que lorsque les axes concernés sont à l'arrêt.

La version 6.5 du logiciel autorise le couplage et le découplage de broches en **rotation** et de broches en asservissement de vitesse ainsi que la configuration dynamique.

Syntaxe

`MASLON(Slv1,Slv2,..., )`	
`MASLOF(Slv1,Slv2,..., )`	
`MASLDEF(Slv1,Slv2,..., axe maître)`	Extension pour configuration dynamique
`MASLDEL(Slv1,Slv2,..., )`	Extension pour configuration dynamique
`MASLON(Slv1, Slv2, ..., )`	Extension pour broche esclave

Remarque

L'arrêt implicite du prétraitement des blocs n'est pas exécuté avec `MASLOF/MASLOFS` . Du fait de l'absence de cet arrêt du prétraitement des blocs, les variables système $P ne fournissent pas de valeurs actualisées pour les axes esclaves jusqu'au moment de la reprogrammation.

Signification

Généralités

`MASLON`	Enclencher un couplage temporaire.
`MASLOF`	Séparer un couplage actif. Pour les broches, les extensions doivent être considérées.

Extension d'une configuration dynamique

`MASLDEF`	Créer/modifier un couplage défini par l'utilisateur via les paramètres machine ou à partir du programme pièce.
`MASLOFS`	Supprimer le couplage de façon analogue à `MASLOF` et freiner automatiquement la broche esclave.
`MASLDEL`	Découpler maître/esclave et supprimer la définition du couplage.
`Slv1, Slv2, ...`	Axes esclaves pilotés par un axe maître.
`Axe maître`	Axe pilotant les axes esclaves définis dans un couplage maître/esclave.

Exemples

Exemple 1 : Configuration dynamique d'un couplage maître/esclave

Configuration dynamique d'un couplage maître/esclave à partir du programme pièce :

L'axe résultant de la rotation d'un conteneur d'axes doit devenir l'axe maître.

```
Code de programme    Commentaire
MASLDEF(AUX,S3)      ; S3 maître pour AUX
MASLON(AUX)          ; Activation du couplage pour AUX
M3=3 S3=4000         ; Rotation à droite
MASLDEL(AUX)         ; Suppression de la configuration et séparation du couplage
AXCTSWE(CT1)         ; Rotation conteneur
```

Exemples

Exemple 2 : Couplage par valeur réelle d'un axe esclave

Couplage par valeur réelle entre un axe esclave et la même valeur de l'axe maître à l'aide de PRESETON.

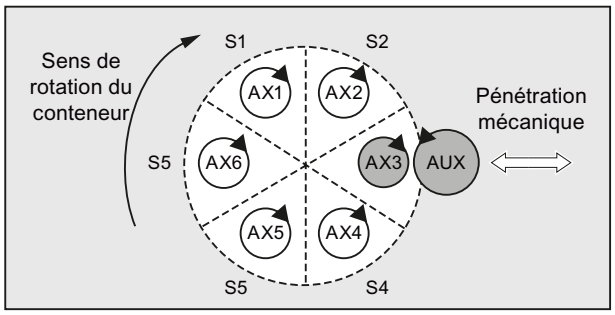
Dans le cas d'un couplage maître/esclave permanent, la valeur réelle de l'axe esclave doit être modifiée à l'aide de PRESETON .

```
Code de programme                                    Commentaire
N37262 $MA_MS_COUPLING_ALWAYS_ACTIVE[AX2]=0          ; Désactiver momentanément
                                                       le couplage permanent.
N37263 NEWCONF
N37264 STOPRE
MASLOF(Y1)                                           ; Désactiver le couplage
                                                       temporaire.
N5 PRESETON(Y1,0,Z1,0,B1,0,C1,0,U1,0)                ; Définir la valeur réelle
                                                       des axes esclaves non
                                                       référencés, dans la mesure
                                                       où ils sont activés avec
                                                       un Power On.
N37262 $MA_MS_COUPLING_ALWAYS_ACTIVE[AX2]=1          ; Activer le couplage
                                                       permanent.
N37263 NEWCONF
```

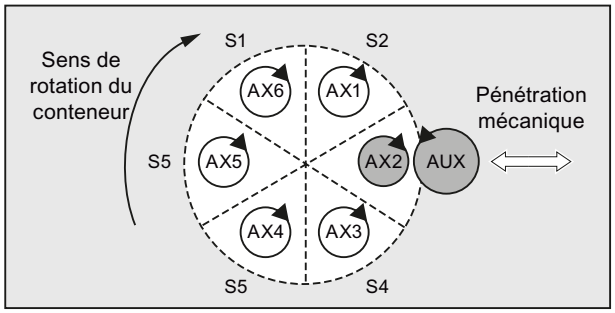
Exemple 3 : Séquence de couplage Position 3/Conteneur CT1

Pour que le couplage avec une autre broche puisse se réaliser après une rotation du conteneur, il convient de supprimer le couplage précédent et sa configuration et de reconfigurer le nouveau couplage.

Situation de départ :



Après rotation de la valeur d'un emplacement :



Bibliographie :

Manuel des fonctions d'extension, plusieurs tableaux de commande et NCU (B3), chapitre : "Conteneur d'axes"

Autres informations

Généralités

MASLOF	Pour les broches en mode de régulation de vitesse, cette instruction est exécutée immédiatement. Les broches esclaves qui tournent à ce moment-là conservent leur vitesse de rotation jusqu'à la reprogrammation de celle-ci.

Extension d'une configuration dynamique

MASLDEF	Définition d'un couplage maître/esclave à partir du programme pièce. Auparavant, la définition s'effectuait exclusivement au moyen de paramètres machine.
MASLDEL	L'instruction supprime l'affectation des axes esclaves à l'axe maître et sépare simultanément le couplage de manière analogue à MASLOF. Dans les paramètres machine, les définitions maître/esclave sont conservées.
MASLOFS	MASLOFS peut être utilisé pour le freinage automatique des broches esclaves lors de la séparation du couplage. Pour les axes et les broches en mode de positionnement, la fermeture et l'ouverture du couplage ne s'opèrent qu'à l'arrêt.

Remarque

Pour l'axe esclave, PRESETON permet de synchroniser la valeur réelle à la même valeur que l'axe maître. Ceci exige une brève désactivation du couplage permanent maître/esclave pour que la valeur réelle de l'axe esclave non référencé puisse être mise à la valeur de l'axe maître avec un Power On. Le couplage permanent est ensuite refermé.

Le réglage du paramètre machine MD37262 $MA_MS_COUPLING_ALWAYS_ACTIVE = 1 active le couplage permanent maître/esclave qui n'a pas d'effet sur les instructions du couplage temporaire.

Mode de couplage des broches

Pour les broches en mode de régulation de vitesse, le mode de couplage est défini explicitement par MASLON, MASLOF, MASLOFS et MASLDEL dans le paramètre machine MD37263 $MA_MS_SPIND_COUPLING_MODE.

Avec le réglage par défaut MD37263 = 0, la fermeture et l'ouverture du couplage de l'axe esclave s'opère uniquement lorsque les axes concernés sont immobilisés. MASLOFS correspond à MASLOF.

Lorsque MD37263 = 1, l'instruction de couplage est exécutée immédiatement, et donc également en mouvement. MASLON ferme immédiatement le couplage et MASLOFS ou MASLOF l'ouvre immédiatement. Avec MASLOFS, les broches esclaves qui tournent à ce moment-là freinent automatiquement ; avec MASLOF, elles conservent leur vitesse de rotation jusqu'à la reprogrammation de celle-ci.

Actions synchrones au déplacement 10

10.1 Notions de base

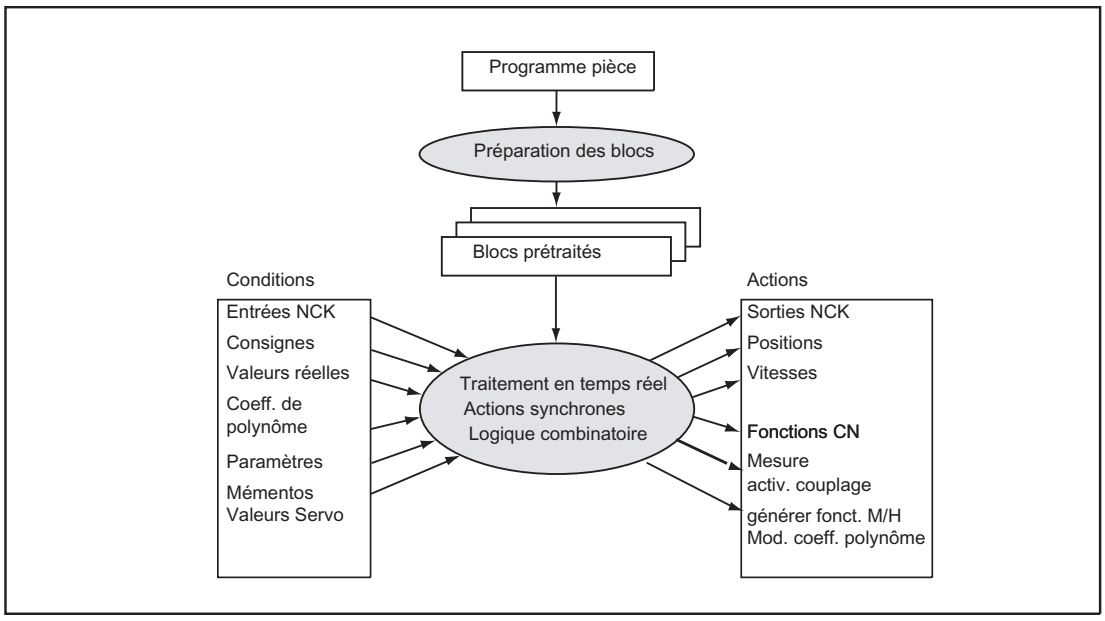
Fonction

Des actions synchrones permettent d'exécuter des actions simultanément aux blocs d'usinage.

Le moment d'exécution des actions peut être défini par des conditions. Les conditions sont surveillées dans la période d'appel de l'interpolateur. Les actions constituent par conséquent une réaction à des événements en temps réel ; leur exécution n'est pas tributaire des limites de blocs.

Par ailleurs, une action synchrone contient des indications sur sa durée de vie et sur la fréquence d'interrogation pour les variables d'exécution programmées et, par conséquent, sur la fréquence d'exécution des actions à lancer. Ainsi, une action peut être lancée une fois seulement ou bien de façon cyclique (toujours dans la période d'appel de l'interpolateur).

Applications possibles

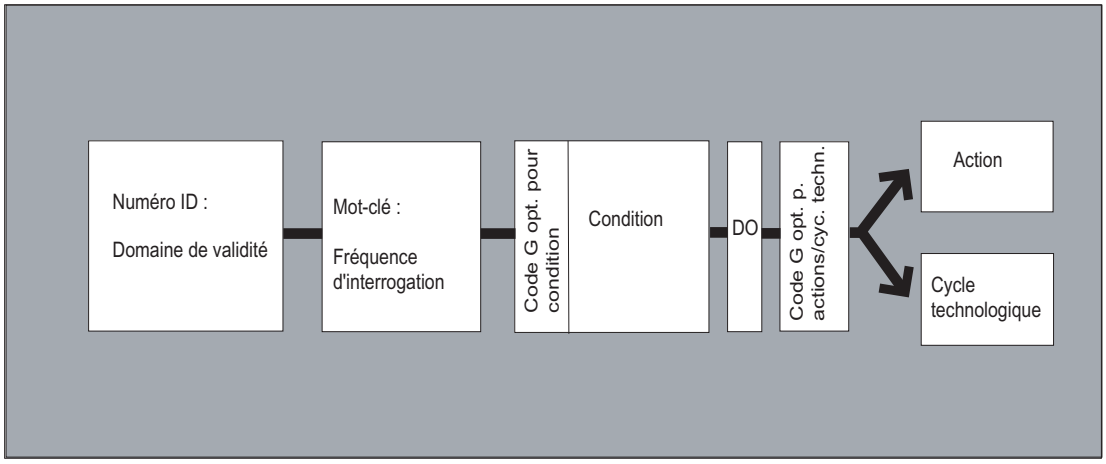


- Optimisation d'applications à temps d'exécution critique (par exemple le changement d'outil)
- Réaction rapide à des événements externes
- Programmation de régulations AC
- Mise en place de fonctions de sécurité
- ....

Programmation

Une action synchrone occupe un bloc à elle seule et agit à partir du prochain bloc exécutable comportant une fonction machine (par exemple un déplacement avec G0, G1, G2, G3).

Les actions synchrones sont composées au maximum de 5 éléments d'instruction avec des tâches différentes :



Syntaxe :

```
DO <action1> <action2> …
<MOT CLE> <condition> DO <action1> <action2> …
ID=<n> <MOT CLE> <condition> DO <action1> <action2> …
IDS=<n> <MOT CLE> <condition> DO <action1> <action2> …
```

Signification :

DO	Instruction de déclenchement de la ou des actions programmées Prise d'effet uniquement si la <condition> est remplie (si celle-ci est programmée). → Voir "Actions"
<Action1> <action2> …	Action(s) à lancer Exemples : • Affectation de variable • Lancement de cycle technologique
<MOT CLE>	Le mot clé (WHEN, WHENEVER, FROM ou EVERY) définit le contrôle cyclique de la <condition> d'une action synchrone. → Voir "Contrôle cyclique de la condition"
<Condition>	Logique combinatoire pour variables d'exécution La condition est contrôlée dans la période d'appel de l'interpolateur.
ID=<n> ou IDS=<n>	Numéro d'identification Le numéro d'identification définit le domaine de validité et la position au sein de la séquence d'exécution. → Voir "Domaine de validité et ordre d'exécution"

Coordination des actions synchrones/cycles technologiques

Les instructions suivantes sont disponibles pour la coordination des actions synchrones/cycles technologiques :

Instruction	Signification
`CANCEL(<n>)`	Effacer des actions synchrones → Voir "Effacer des actions synchrones"
`LOCK(<n>)`	Bloquer des actions synchrones
`UNLOCK(<n>)`	Débloquer des actions synchrones
`RESET`	Remettre le cycle technologique à zéro
	Pour `LOCK`, `UNLOCK` et `RESET` : → voir "Bloquer, débloquer, remettre à zéro"

Exemple

Code de programme	Commentaire
`WHEN $AA_IW[Q1]>5 DO M172 H510`	`; Quand la position réelle de l'axe Q1 dépasse 5 mm, les fonctions auxiliaires M172 et H510 sont transférées à l'interface AP.`

10.1.1 Domaine de validité et ordre d'exécution (ID, IDS)

Fonction

Domaine de validité

Le domaine de validité d'une action synchrone est défini par l'identificateur `ID` ou `IDS` :

Sans ID modal :	Action synchrone à effet non modal en mode automatique
`ID` :	Action synchrone à effet modal agissant en mode automatique jusqu'en fin de programme
`IDS` :	Action synchrone à effet modal statique dans tous les modes de fonctionnement, y compris au-delà de la fin du programme

Applications

- Rectification AC en mode JOG
- Logique combinatoire pour Safety Integrated
- Fonctions de surveillance, réactions aux états machine dans tous les modes de fonctionnement

Ordre d'exécution

Les actions synchrones à effet modal statique sont traitées dans l'ordre de leur numéro `ID` ou `IDS` (`ID=<n>` ou `IDS=<n>`) dans la période d'appel de l'interpolateur.

Les actions synchrones à effet non modal (sans numéro ID) sont traitées dans l'ordre de programmation, après exécution des actions synchrones à effet modal.

Remarque

Des paramètres machines permettent de protéger des actions synchrones à effet modal face à des modifications ou à des suppressions (→ constructeur de la machine-outil).

Programmation

Syntaxe	Signification
Sans ID modal	L'action synchrone est uniquement active en **mode automatique**. Elle est valide uniquement pour le bloc exécutable suivant (bloc avec instruction de déplacement ou autre action machine), autrement dit elle est à effet **non modal**. Exemple : `WHEN $A_IN[3]==TRUE DO $A_OUTA[4]=10`
`ID=<n> ...`	L'action synchrone agit dans les blocs qui suivent et elle a donc un effet **modal**. Elle peut être désactivée par `CANCEL(<n>)` ou par la programmation d'une autre action synchrone sous le même numéro ID. Les actions synchrones actives dans le bloc `M30` retardent la fin du programme. Les actions synchrones avec ID sont uniquement actives en **mode automatique**. Plage de valeurs de <n> : 1 ... 255 Exemple : `ID=2 EVERY $A_IN[1]==1 DO POS[X]=0`
`IDS=<n>`	Les actions synchrones statiques agissent de **façon modale dans tous les modes de fonctionnement**. Elles restent actives au-delà de la fin du programme et peuvent être directement activées après Power On avec un ASUP. Il est ainsi possible d'activer des actions qui doivent être exécutées dans la CN indépendamment du mode de fonctionnement sélectionné. Plage de valeurs de <n> : 1 ... 255 Exemple : `IDS=1 EVERY $A_IN[1]==1 DO POS[X]=100`

10.1.2 Contrôle cyclique de la condition (WHEN, WHENEVER, FROM, EVERY)

Fonction

Un mot-clé définit le contrôle cyclique de la condition d'une action synchrone. Si aucun mot-clé n'est programmé, les actions de l'action synchrone sont exécutées dans chaque période d'appel de l'interpolateur.

Mots-clés

pas de mot-clé	L'exécution de l'action n'est liée à aucune condition. L'action est exécutée de façon cyclique dans la période d'appel de l'interpolateur.
`WHEN`	La condition est interrogée dans chaque période d'appel de l'interpolateur, jusqu'à ce qu'elle soit remplie une première fois ; et l'action s'y rapportent est alors exécutée une fois.
`WHENEVER`	La condition est vérifiée dans chaque période d'appel de l'interpolateur de manière cyclique. L'action est exécutée dans chaque période d'appel de l'interpolateur, aussi longtemps que la condition est remplie.
`FROM`	La condition est contrôlée dans chaque période d'appel de l'interpolateur jusqu'à ce qu'elle soit remplie une première fois. L'action est alors exécutée tant que l'action synchrone est active, c'est-à-dire même si la condition n'est plus remplie.
`EVERY`	La condition est interrogée dans chaque période d'appel de l'interpolateur. L'action est toujours exécutée une seule fois, quand la condition est remplie. Commande par fronts : L'action sera exécutée une nouvelle fois dès que la condition passe de l'état FALSE à l'état TRUE.

Variables d'exécution

Les variables utilisées sont évaluées dans la période d'appel de l'interpolateur. Les variables d'exécution dans les actions synchrones ne déclenchent pas d'arrêt du prétraitement.

Evaluation :

Quand des variables d'exécution apparaissent dans un programme pièce (par exemple une valeur réelle, une position d'une entrée ou sortie TOR, etc.), le prétraitement des blocs est stoppé jusqu'à ce que le bloc précédent soit entièrement traité et que les valeurs des variables d'exécution soient connues.

Exemples

Exemple 1 : Sans mot clé

Code de programme	Commentaire
`DO $A_OUTA[1]=$AA_IN[X]`	`; Sortie de la valeur réelle sur la sortie analogique`

Exemple 2 : WHENEVER

Code de programme	Commentaire
`WHENEVER $AA_IM[X] > 10.5*SIN(45) DO …`	`; Comparaison avec une expression calculée dans le prétraitement des blocs.`
`WHENEVER $AA_IM[X] > $AA_IM[X1] DO …`	`; Comparaison avec d'autres variables d'exécution.`
`WHENEVER ($A_IN[1]==1) OR ($A_IN[3]==0) DO ...`	`; Deux comparaisons combinées entre elles.`

Exemple 3 : EVERY

Code de programme	Commentaire
`ID=1 EVERY $AA_IM[B]>75 DO POS[U]=IC(10) FA[U]=900`	`; Quand la valeur réelle de l'axe B dans le SCM dépasse la valeur 75, l'axe U doit toujours poursuivre le positionnement de 10 avec avance axiale.`

Autres informations

Condition

La condition se présente toujours sous la forme d'une expression logique pouvant être structurée de façon quelconque par des opérateurs booléens. Les expressions booléennes doivent toujours être indiquées entre parenthèses.

La condition est contrôlée dans la période d'appel de l'interpolateur.

Il est possible de programmer un code G avant la condition. Ceci vous permet de définir des réglages particuliers pour le traitement de la condition et l'action/le cycle technologique à exécuter, indépendamment des réglages en vigueur dans le programme pièce. Le découplage des actions synchrones de l'environnement du programme est nécessaire, car les actions synchrones doivent exécuter leurs actions à des instants quelconques dans un état initial défini lorsque des conditions de déclenchement sont remplies.

Applications

Détermination du système d'unités pour le traitement de la condition et l'action à l'aide des codes G `G70`, `G71`, `G700`, `G710`.

Un code G indiqué dans la partie "condition" est valable pour le traitement de la condition et pour l'action, si aucun code G n'est indiqué dans la partie "action".

Un seul **code G** du groupe de codes G peut être programmé dans une partie "condition".

Conditions possibles

- Comparaison de variables d'exécution (E/S analogiques/TOR par exemple)
- Combinaison logique entre des résultats de comparaisons
- Calcul d'expressions temps réel
- Temporisation/Distance du début du bloc
- Distance de la fin du bloc
- Valeurs de mesure, résultats de mesure
- Valeurs Servo
- Vitesses, état des axes

10.1.3 Actions (DO)

Fonction

Dans des actions synchrones, vous pouvez programmer une ou plusieurs actions. Toutes les actions programmées dans un bloc deviennent actives dans la même période d'appel de l'interpolateur.

Syntaxe

```
DO <action1> <action2> ...
```

Signification

`DO`	Déclenche, lorsque la condition est remplie, une action ou un cycle technologique.
`<Action>`	Action déclenchée lorsque la condition est remplie, comme par exemple affecter une valeur à une variable, activer un couplage d'axe, mettre à 1 des sorties du noyau CN, sortir les fonctions M-, S- et H, imposer le code G programmé, ...

Les codes G peuvent être programmés dans la partie "actions/cycles technologiques" d'une action synchrone. Ce code G est valable pour toutes les actions et tous les cycles technologiques du bloc et remplace le code G éventuellement indiqué dans la partie "condition". Si la partie "action" comporte des cycles technologiques, ce code G s'applique, également lorsque les cycles technologiques sont terminés, de façon modale pour toutes les actions suivantes jusqu'au prochain code G.

Un seul code G du groupe de codes G (`G70`, `G71`, `G700`, `G710`) peut être programmé dans la partie "action".

Exemple : Action synchrone avec deux actions

Code de programme	Commentaire
`WHEN $AA_IM[Y] >= 35.7 DO M135 $AC_PARAM=50`	`; Si la condition est remplie, M135 est transférée à l'interface AP et la correction est mise à 50%.`

10.2 Opérateurs pour les conditions et actions

Comparaisons (==, <>, <, >, <=, >=)	Dans des conditions, vous pouvez comparer des variables ou des expressions partielles. Le résultat est toujours du type BOOL. Tous les opérateurs relationnels connus sont admis
Opérateurs booléens (NOT, AND, OR, XOR)	Variables, constantes ou comparaisons peuvent être combinées avec les opérateurs booléens habituels.
Opérateurs sur bits (B_NOT, B_AND, B_OR, B_XOR)	Les opérateurs sur bits suivants sont possibles : B_NOT, B_AND, B_OR, B_XOR.
Opérations arithmétiques (+, -, *, /, DIV, MOD)	Les variables d'exécution peuvent être combinées entre elles ou avec des constantes via des opérations arithmétiques.
Fonctions mathématiques (SIN, COS, TAN, ASIN, ACOS, ABS, TRUNC, ROUND, LN, EXP, ATAN2, POT, SQRT, CTAB, CTABINV).	Les fonctions mathématiques peuvent être appliquées aux variables de type REAL.
Indexation	L'indexation est possible avec des expressions d'exécution.

Exemple

- **Combinaison des opérations arithmétiques**

La multiplication est prioritaire sur l'addition, les parenthèses sont admises. Les opérateurs DIV et MOD sont également admis pour les données de type REAL

Programmation	Commentaire
DO $AC_PARAM[3] = $A_INA[1]-$AA_IM[Z1]	; ; Soustraction de deux ;Variables d'exécution
WHENEVER $AA_IM[x2] < $AA_IM[x1]-1.9 DO $A_OUT[5] = 1	; ;Soustraction d'une constante de variables
DO $AC_PARAM[3] = $INA[1]-4*SIN(45.7 $P_EP[Y])*R4	; Expression constante, calculée dans le prétraitement des blocs

- **Fonctions mathématiques**

Programmation	Commentaire
DO $AC_PARAM[3] = COS($AC_PARAM[1])	; ;

- **Expressions temps réel**

Programmation	Commentaire
ID=1 WHENEVER ($AA_IM[Y]>30) AND ($AA_IM[Y]<40) DO $AA_OVR[S1]=80	; Sélection d'une fenêtre de positionnement
ID=67 DO $A_OUT[1]=$A_IN[2] XOR $AN_MARKER[1]	; Evaluation de 2 signaux booléens
ID=89 DO $A_OUT[4]=$A_IN[1] OR ($AA_IM[Y]>10)	; Sortie du résultat d'une comparaison

- **Variable d'exécution indexée**

Programmation	Commentaire
WHEN…DO $AC_PARAM[$AC_MARKER[1]] = 3	;
N'est pas admis	;
$AC_PARAM[1] = $P_EP[$AC_MARKER]	;

10.3 Variables d'exécution pour actions synchrones

10.3.1 Variables système

Fonction

Des variables système permettent de lire et d'écrire des données de la CN. Les variables système se distinguent entre variables de prétraitement et variables d'exécution. Les variables de prétraitement sont toujours exécutées lors du prétraitement. Les variables d'exécution déterminent systématiquement leurs valeurs par rapport à l'état d'exécution courant.

Désignation

Le nom des variables système commence toujours par le caractère "$" :

Variables de prétraitement :

$M...	Paramètres machine
$S...	Données de réglage, zones de protection
$T...	Paramètres de gestion des outils
$P...	Valeurs programmées, données de prétraitement
$C...	Variables des cycles enveloppes ISO
$O...	Données optionnelles
R ...	Paramètres R

Variables d'exécution :

$$A...	Données d'exécution courantes
$$V...	Données servo
$R...	Paramètres R

Une 2e lettre décrit la possibilité d'accès à la variable :

N...	Valeur globale NCK (valeur de validité généralement)
C...	Valeur spécifique au canal
A...	Valeur spécifique à l'axe

La 2e lettre est généralement utilisée seulement pour les variables d'exécution. Les variables de prétraitement comme $P_ sont en général exécutées sans 2e lettre.

Le préfixe ($ suivi d'une ou de deux lettres) est systématiquement suivi d'un caractère de soulignement et du nom de la variable (le plus souvent la désignation ou l'abréviation en anglais).

Types de données

Les variables d'exécution peuvent posséder les types de données suivants :

INT	Integer pour nombres entiers avec signe
REAL	Real pour nombres réels (rationnels)
BOOL	Boolean TRUE et FALSE
CHAR	Caractère ASCII
STRING	Chaîne de caractères alphanumériques
AXIS	Adresses d'axes et broches

Les variables de prétraitement peuvent en outre posséder le type de données suivant :

FRAME	Transformations de coordonnées

Tableaux de variables

Les variables système peuvent être créées sous forme de tableaux de 1 à 3 dimensions.

Les types de données suivants sont pris en charge : BOOL, CHAR, INT, REAL, STRING, AXIS

Le type de données des indices peut être INT ou AXIS en combinaison quelconque.

Les variables STRING peuvent être créées uniquement en 2 dimensions.

Exemples de définitions de tableau :

```
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]
```

Remarque

L'affichage de variables système à trois dimensions n'est soumis à aucune restriction s'il existe une variable OPI pour la variable système.

10.3.2 Conversion de type implicite

Fonction

Lors de lectures d'affectations et de transmissions de paramètres, des variables de types de données différents peuvent être assignés ou transmis.

La conversion de type implicite déclenche une conversion interne de type de valeurs.

Conversions de type possibles

vers	REAL	INT	BOOL	CHAR	STRING	AXIS	FRAME
de							
REAL	Oui	oui*	oui[1])	–	–	–	–
INT	Oui	Oui	oui[1])	–	–	–	–
BOOL	Oui	Oui	Oui	–	–	–	–

Observations

* En cas de conversion du type REAL en type INT, une partie décimale >= 0,5 est arrondie par excès ; dans le cas contraire, elle est arrondie par défaut (voir fonction ROUND).
En cas de dépassements de valeur, une alarme est déclenchée.

1) La valeur <> 0 équivaut à TRUE, la valeur == 0 équivaut à FALSE

Résultats

```
Conversion de type de REAL ou INTEGER vers BOOL
Résultat BOOL = TRUE          lorsque la valeur de REAL ou INTEGER est différente
                              de zéro
Résultat BOOL = FALSE         lorsque la valeur de REAL ou INTEGER est égale à zéro
Conversion de type de BOOL vers REAL ou INTEGER
Résultat REAL TRUE            lorsque la valeur est BOOL = TRUE (1)
Résultat INTEGER = TRUE       lorsque la valeur est BOOL = TRUE (1)
Conversion de type de BOOL vers REAL ou INTEGER
Résultat REAL FALSE)          lorsque la valeur est BOOL = FALSE (0)
Résultat INTEGER = FALSE      lorsque la valeur est BOOL = FALSE (0)
```

Exemples de conversions de type implicites

```
Conversion de type de INTEGER vers BOOL
$AC_MARKER[1]=561
ID=1 WHEN $A_IN[1] == TRUE DO $A_OUT[0]=$AC_MARKER[1]

Conversion de type de REAL vers BOOL
R401 = 100.542
WHEN $A_IN[0] == TRUE DO $A_OUT[2]=$R401

Conversion de type de BOOL vers INTEGER
ID=1 WHEN $A_IN[2] == TRUE DO $AC_MARKER[4] = $A_OUT[1]]

Conversion de type de BOOL vers REAL
R401 = 100.542
WHEN $A_IN[3] == TRUE DO $R10 = $A_OUT[3]
```

10.3.3 Variables GUD

Variables GUD à action synchrone

Outre des variables système spécifiques, il est également possible d'utiliser des variables utilisateur globales prédéfinies à action synchrone dans les actions synchrones (GUD à action synchrone). Le nombre de variables GUD à action synchrone mises à disposition de l'utilisateur est paramétré selon le type de données et le mode d'accès dans les paramètres machine suivants :

- MD18660 $MM_NUM_SYNACT_GUD_**REAL**[<x>] = <nombre>
- MD18661 $MM_NUM_SYNACT_GUD_**INT**[<x>] = <nombre>
- MD18662 $MM_NUM_SYNACT_GUD_**BOOL**[<x>] = <nombre>
- MD18663 $MM_NUM_SYNACT_GUD_**AXIS**[<x>] = <nombre>
- MD18664 $MM_NUM_SYNACT_GUD_**CHAR**[<x>] = <nombre>
- MD18665 $MM_NUM_SYNACT_GUD_**STRING**[<x>] = <nombre>

L'indice <x> indique le bloc de données (droits d'accès), la valeur <nombre>, le nombre de variables GUD à action synchrone du type de données correspondant (REAL, INT, ...). Une variable de tableau unidimensionnelle est ensuite créée dans le bloc de données respectif pour chaque type de données, avec le schéma de nom suivant : SYG_<type de donnée><droit d'accès>[<indice>] :

Indice <x>		Type de données (MD18660 ... MD18665)					
	Bloc	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]
avec i = 0 à (<nombre> - 1) bloc : _N_DEF_DIR/_N_ ... _DEF, p. ex. pour SGUD ⇒ _N_DEF_DIR/_N_**SGUD**_DEF							

Propriétés

Les variables à action synchrone présentent les propriétés suivantes :

- La lecture et l'écriture des variables GUD à action synchrone sont possibles dans les actions synchrones et les programmes pièce / cycles
- L'accès aux variables GUD à action synchrone peut être réalisé avec BTSS
- L'affichage des variables GUD à action synchrone s'effectue dans l'interface utilisateur de l'IHM, dans la zone de commande "Paramètres"
- Les variables GUD à action synchrone peuvent être utilisées sur l'IHM, dans la vue des variables et dans le journal des variables de l'assistant
- La dimension de tableau pour les variables GUD à action synchrone de type STRING est définie à 32 (31 caractères + \0).
- Même si aucun fichier de définition n'a été créé manuellement pour les données utilisateur globales (GUD), la lecture des variables GUD à action synchrone définies dans des paramètres machine est possible dans le bloc GUD correspondant de l'IHM.

IMPORTANT

La définition de variables utilisateur (GUD, PUD, LUD) portant le même nom que des variables GUD à action synchrone (`DEF ... SYG_xy`) est uniquement possible si aucune variable GUD à action synchrone de même nom n'a été paramétrée (MD18660 - MD18665). Ces variables GUD définies par l'utilisateur ne peuvent **pas** être utilisées dans des actions synchrones.

Droits d'accès

Les droits d'accès définis dans un fichier de définition GUD restent valables et s'appliquent exclusivement aux variable GUD définies dans ce fichier de définition GUD.

Comportement à la suppression

Lorsque le contenu d'un fichier de définition GUD donné est réactivé, l'ancien bloc de données GUD est d'abord supprimé dans le système de fichier actif. Les variables GUD à action synchrone configurées sont également réinitialisées. Cette procédure est également possible sur l'IHM, dans le groupe fonctionnel "Services" > "Définir et activer des données utilisateur (GUD)".

10.3.4 Descripteur d'axe par défaut (NO_AXIS)

Fonction

Les variables ou les paramètres de type AXIS qui n'ont pas été initialisés avec une valeur peuvent être pourvus de descripteurs d'axe par défaut définis. Les variables d'axes non définies sont également initialisées avec cette valeur par défaut.

Des noms d'axe valables non initialisés sont reconnus par une interrogation de la variable "NO_AXIS" dans les actions synchrones. Le descripteur d'axe par défaut configuré par un paramètre machine est affecté à ces descripteurs d'axe non initialisés.

Constructeur de la machine-outil

Au moins un descripteur d'axe valable existant doit être défini et préréglé par des paramètres machine. Mais il est également possible que tous les descripteurs d'axes valables existants soient préréglés. Tenez compte des indications du constructeur de la machine.

Remarque

Les variables nouvellement créées se voient maintenant affecter automatiquement lors de la définition la valeur sauvegardée dans le paramètre machine pour les noms d'axe par défaut.

Pour d'autres informations sur une définition valable par paramètre machine, voir :
Bibliographie :
/FBSY/ Description fonctionnelle Actions synchrones

Syntaxe

```
PROC UP(AXIS PAR1=NO_AXIS, AXIS PAR2=NO_AXIS)
IF PAR1 <>NO_AXIS...
```

Signification

`PROC`	Définition de sous-programme
`SP`	Nom du sous-programme pour son identification
`PARn`	Paramètre n
`NO_AXIS`	Initialisation du paramètre formel avec descripteur d'axe par défaut

Exemple : Définition d'une variable d'axe dans le programme principal

```
Code de programme
DEF AXIS AXVAR
UP( , AXVAR)
```

10.3.5 Marque d'action synchrone ($AC_MARKER[n])

Fonction

La variable de tableau $AC_MARKER[n] peut être lue et écrite dans des actions synchrones. Ces variables peuvent se trouver soit dans la mémoire du système de fichier passif, soit dans celle du système de fichier actif.

Variable d'action synchrone : Type de données INT

$AC_MARKER[n]	Mémento/compteur spécifique au canal de type de données INTEGER
$MC_MM_NUM_AC_MARKER	Paramètre machine pour le paramétrage du nombre de mémentos spécifiques aux canaux pour les actions synchrones au déplacement
n	Indice de tableau des variables 0-n

Exemple d'écriture et de lecture de variables de mémentos

```
Code de programme
WHEN ... DO $AC_MARKER[0] = 2
WHEN ... DO $AC_MARKER[0] = 3
WHENEVER $AC_MARKER[0] == 3 DO $AC_OVR=50
```

10.3.6 Paramètre d'action synchrone ($AC_PARAM[n])

Fonction

Les paramètres d'actions synchrones $AC_PARAM[n] servent aux calculs et comme mémoires tampons dans des actions synchrones. Ces variables peuvent se trouver soit dans la mémoire du système de fichier passif, soit dans celle du système de fichier actif.

Variable de synchronisation : Type de données REAL

Les paramètres ne figurent qu'une fois dans un canal sous le même nom.

`$AC_PARAM[n]`	Variable de calcul pour les actions synchrones au déplacement (REAL)
`$MC_MM_NUM_AC_PARAM`	Paramètre machine pour le paramétrage du nombre de paramètres pour des actions synchrones au déplacement jusqu'à 20000 maximum.
`n`	Indice de tableau du paramètre 0n

Exemple de paramètres d'actions synchrones $AC_PARAM[n]

```
Code de programme
$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 Paramètre de calcul ($R[n])

Fonction

Cette variable de tableau statique sert pour les calculs dans le programme pièce et les actions synchrones.

Syntaxe

Programmation dans le programme pièce :

```
REAL R[n]
REAL Rn
```

Programmation dans des actions synchrones :

```
REAL $R[n]
REAL $Rn
```

Paramètre de calcul

L'utilisation de paramètres de calcul permet de :

- la mise en mémoire des valeurs qui doivent être sauvegardées après la fin d'un programme, après un reset CN ou après un Power On.
- Affichage de valeurs mémorisées dans l'image des paramètres R.

Exemples

Code de programme	Commentaire
WHEN $AA_IM[X]>=40.5 DO $R10=$AA_MM[Y]	; Utilisation de R10 dans l'action synchrone.
G01 X500 Y70 F1000	
STOPRE	; Arrêt du prétraitement
IF R10>20	; Evaluation de la variable de calcul.

Code de programme
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 Lecture et écriture des paramètres machine CN et des données de réglage CN

Fonction

La lecture et l'écriture des paramètres machine et des données de réglage CN sont possibles depuis des actions synchrones. Lors de la lecture et de l'écriture d'éléments de tableau de paramètres machine, un indice peut être omis à la programmation. Si cela a lieu dans un programme pièce, alors, à la lecture, le **premier** élément de tableau est lu et, à l'écriture, tous les éléments du tableau sont décrits avec la valeur.

Dans des actions synchrones, seul le **premier** élément est lu et écrit dans ce cas.

Définition

PM, SD avec

$: Lecture de la valeur au moment de l'interprétation des actions synchrones

$$: Lecture de la valeur dans l'exécution

Lire les valeurs PM et SD lors du prétraitement

Dans l'action synchrone, ces données sont adressées avec le caractère $ et traitées lors du prétraitement.

```
ID=2 WHENEVER $AA_IM[z]<$SA_OSCILL_REVERSE_POS2[Z]-6 DO $AA_OVR[X]=0
; On peut accéder ici à la zone d'inversion 2 non modifiable pour l'oscillation.
```

Lire les valeurs PM et SD lors de l'exécution

Dans l'action synchrone, ces données sont adressées avec les caractères $$ et traitées lors de l'exécution.

```
ID=1 WHENEVER $AA_IM[z]<$$SA_OSCILL_REVERSE_POS2[Z]-6 DO $AA_OVR[X]=0
; On considère ici, que la position d'inversion est susceptible d'être modifiée par
une intervention manuelle pendant l'usinage.
```

Ecrire les PM et SD lors de l'exécution

le droit d'accès en vigueur doit autoriser l'accès en écriture. La prise d'effet pour tous les PM et SD est indiquée dans le **manuel** : /LIS/, Listes (livret 1).

Les `PM` et `SD` à écrire sont à adresser en commençant par `$$` .

Exemple

Code de programme	**Commentaire**
`ID=1 WHEN $AA_IW[X]>10 DO` `$$SN_SW_CAM_PLUS_POS_TAB_1[0]=20`	`;` Modification des positions de commutation de cames logicielles. Remarque: les positions de commutation sont à modifier 2 à 3 périodes d'appel de l'interpolateur avant que la position soit atteinte.
`$$SN_SW_CAM_MINUS_POS_TAB_1[0]=30`	

10.3.9 Variables de temporisation ($AC_Timer[n])

Fonction

La variable système $AC_TIMER[n] permet de lancer des actions après écoulement d'un temps d'attente défini.

Variables de temporisation : Type de données REAL

`$AC_TIMER[n]`	Temporisateur spécifique au canal de type de données REAL
`s`	Unité en secondes
`n`	Indice de la variable de temporisation

Régler la temporisation

L'incrémentation d'une variable de temporisation démarre avec l'affectation d'une valeur :
`$AC_TIMER[n] = value`

n :	Numéro de la variable de temporisation
Value :	Valeur de départ (généralement "0")

Arrêter la temporisation

L'incrémentation d'une variable de temporisation s'arrête avec l'affectation d'une valeur négative :
`$AC_TIMER[n]=-1`

Lire la temporisation

La valeur courante peut être lue que la variable de temporisation soit incrémentée ou arrêtée. Dès que la variable de temporisation a été stoppée par l'affectation de -1, la dernière valeur courante est figée et reste accessible à la lecture.

Exemple

Sortie d'une valeur réelle sur sortie analogique 500 ms après détection d'une entrée numérique :

Code de programme	Commentaire
`WHEN $A_IN[1]==1 DO $AC_TIMER[1]=0`	`; Mise à zéro et`
`WHEN $AC_TIMER[1]>=0.5 DO $A_OUTA[3]=$AA_IM[X] $AC_TIMER[1]=-1`	`démarrage de la temporisation`

10.3.10 Variables FIFO ($AC_FIFO1[n] ... $AC_FIFO10[n])

Fonction

Pour mémoriser des suites de données connexes, vous avez 10 variables FIFO à disposition (mémoire à défilement).
Type de donnée : REAL

Application :

- Mesure cyclique
- Usinage en série

Chaque élément est accessible en lecture et en écriture.

Variable FIFO

Le nombre de variables FIFO disponibles est défini par le paramètre machine MD28260 $MC_NUM_AC_FIFO.

Le nombre de valeurs pouvant être écrites dans une variable FIFO est défini par le paramètre machine MD28264 $MC_LEN_AC_FIFO. Les variables FIFO ont toutes la même longueur.

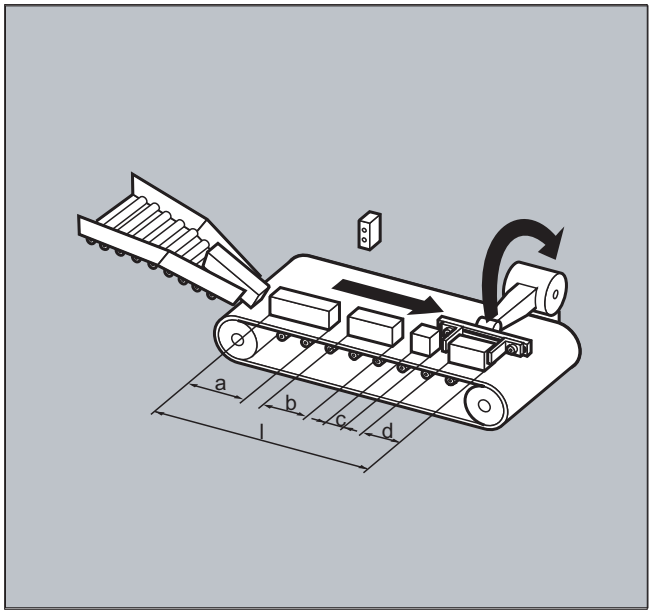
La somme de tous les éléments FIFO n'est calculée que lorsque MD28266 $MC_MODE_AC_FIFO Bit0 = 1.

Les indices **0 à 5** ont une signification particulière :

Indice	Signification	
0	En écriture :	La nouvelle valeur est rangée dans FIFO.
	En lecture :	L'élément le plus ancien est lu et supprimé du FIFO.
1	Accès à l'élément le plus ancien qui est mémorisé	
2	Accès à l'élément le plus récent qui est mémorisé	
3	Somme de tous les éléments FIFO	
4	Nombre d'éléments disponibles dans FIFO Chaque élément FIFO est accessible en lecture et en écriture. La remise à zéro des variables FIFO s'effectue par la remise à zéro du nombre d'éléments, par exemple pour la première variable FIFO : $AC_FIFO1[4]=0	
5	Indice d'écriture relatif au début du FIFO	
6 à n_{max}	Accès au nième élément FIFO	

Exemple : Mémoire circulante

En cours de production, on utilise un tapis roulant pour convoyer des produits de différentes longueurs (a, b, c, d). Par conséquent, sur le tapis roulant ayant une longueur de convoyage, le nombre des produits transportés varie en fonction de leur longueur. Finalement, pour une vitesse constante du tapis roulant, il conviendra d'adapter le prélèvement des produits sur le tapis aux temps d'arrivée variables de ces produits.



Code de programme	Commentaire
`DEF REAL DIST=2.5`	`; Ecartement constant entre les produits déposés.`
`DEF REAL TOTAL=270`	`; Ecartement entre la position de mesure de longueur et la position de prélèvement.`
`EVERY $A_IN[1]==1 DO $AC_FIFO1[4]=0`	`; Réinitialisation du FIFO au début du processus.`
`EVERY $A_IN[2]==1 DO $AC_TIMER[0]=0`	`; Démarrage de la mesure de temps lorsqu'un produit traverse la barrière photoélectrique.`
`EVERY $A_IN[2]==0 DO $AC_FIFO1[0]=$AC_TIMER[0]*$AA_VACTM[B]`	`; Dès que la barrière photoélectrique est libre, calcul de la longueur du produit à partir du temps mesuré et de la vitesse de transport, et mémorisation dans FIFO.`
`EVERY $AC_FIFO1[3]+$AC_FIFO1[4]*ZWI>=TOTAL DO POS[Y]=-30 $R1=$AC_FIFO1[0]`	`; Dès que la somme de toutes les longueurs de produit et de tous les écartements intermédiaires est supérieure/égale à la longueur entre les positions de dépose et de prélèvement, prélèvement du produit du tapis à la position de prélèvement , lecture de la longueur de produit correspondante du FIFO.`

10.3.11 Renseignement sur les types de bloc dans l'interpolateur ($AC_BLOCKTYPE, $AC_BLOCKTYPEINFO, $AC_SPLITBLOCK)

Fonction

En ce qui concerne les actions synchrones, les variables système suivantes sont disponibles pour obtenir des informations sur un bloc courant en cours d'exécution :

- $AC_BLOCKTYPE
- $AC_BLOCKTYPEINFO
- $AC_SPLITBLOCK

Variables de types de bloc et d'infos de types de bloc

$AC_BLOCKTYPE		**$AC_BLOCKTYPEINFO**				
Valeur :		**Valeur :**				
0	**différente de 0**	**T**	**H**	**Z**	**E**	**Signification :**
Bloc d'origine	**Bloc intermédiaire**					**Déclenchement de bloc intermédiaire :**
	1	1	0	0	0	Bloc généré en interne, aucune autre information
	2	2	0	0	1	Chanfreins/arrondis : complétée
	2	2	0	0	2	Chanfreins/arrondis : Cercle
	3	3	0	0	1	ARD : accostage sur une droite
	3	3	0	0	2	ARD : accostage sur un quart de cercle
	3	3	0	0	3	ARD : accostage sur un demi-cercle
						Correction d'outil :
	4	4	0	0	1	bloc d'accostage après STOPRE
	4	4	0	0	2	Blocs de liaison dans le point d'intersection non trouvé
	4	4	0	0	3	Cercle sous forme de point dans les angles rentrants (uniquement dans TRACYL)
	4	4	0	0	4	Arc de contournement (ou section conique) dans les angles saillants
	4	4	0	0	5	Blocs d'accostage dans l'inhibition de la correction
	4	4	0	0	6	Blocs d'accostage dans l'activation répétée de la correction de rayon d'outil
	4	4	0	0	7	Dédoublement du bloc en raison d'une courbure trop forte
	4	4	0	0	8	Blocs de compensation pour le fraisage en bout 3D (vecteur outil || vecteur surface)

$AC_BLOCKTYPE		$AC_BLOCKTYPEINFO				
Valeur :		Valeur :				
0	différente de 0	T	H	Z	E	Signification :
Bloc d'origine	Bloc intermédiaire					Déclenchement de bloc intermédiaire :
						Arrondissement via :
	5	5	0	0	1	G641
	5	5	0	0	2	G642
	5	5	0	0	3	G643
	5	5	0	0	4	G644
						Bloc TLIFT avec :
	6	6	0	0	1	déplacement linéaire de l'axe tangentiel, sans mouvement de retrait
	6	6	0	0	2	déplacement non linéaire de l'axe tangentiel (polynôme) et sans mouvement de retrait
	6	6	0	0	3	mouvement de retrait et déplacement de l'axe tangentiel commencent simultanément
	6	6	0	0	4	mouvement de retrait, l'axe tangentiel ne commence que lorsqu'une certaine position de retrait est atteinte.
						Segmentation du déplacement :
	7	7	0	0	1	la segmentation du déplacement est programmée sans que le poinçonnage ou le grignotage ne soit actif
	7	7	0	0	2	la segmentation du déplacement est programmée avec un poinçonnage ou un grignotage actif
	7	7	0	0	3	la segmentation du déplacement est automatiquement générée en interne
						Cycles de compilation :
	8	ID de l'application				ID de l'application des cycles de compilation ayant créé le bloc

M : position des milliers
C : position des centaines
D : position des dizaines
U : position des unités

Remarque

Dans la position des milliers (T), $AC_BLOCKTYPEINFO contient toujours également la valeur du type de bloc pour le cas où il existe un bloc intermédiaire. Si la valeur dans $AC_BLOCKTYPE est différente de 0, la position des milliers n'est pas reprise.

$AC_SPLITBLOCK	
Valeur :	**Signification :**
0	Bloc programmé non modifié (un bloc généré par le compacteur est également traité en tant que bloc programmé)
1	Il existe un bloc généré en interne ou un bloc d'origine tronqué
3	Le dernier bloc se trouve dans une chaîne de blocs générés en interne ou de blocs d'origine tronqués

Exemple : Comptage de blocs d'arrondissement

```
Code de programme     Commentaire
$AC_MARKER[0]=0
$AC_MARKER[1]=0
$AC_MARKER[2]=0
...
                      ; Définition d'actions synchrones servant à compter
                        des blocs d'arrondissement.
                      ; Tous les blocs d'arrondissement comptent dans $AC_MARKER[0] :
ID=1 WHENEVER ($AC_TIMEC==0) AND ($AC_BLOCKTYPE==5) DO $AC_MARKER[0]=$AC_MARKER[0]+1
...
                      ; Les blocs d'arrondissement générés avec G641 comptent dans $AC_MARKER[1] :
ID=2 WHENEVER ($AC_TIMEC==0) AND ($AC_BLOCKTYPEINFO==5001) DO $AC_MARKER[1]=$AC_MARKER[1]+1
                      ; Les blocs d'arrondissement générés avec G642 comptent dans $AC_MARKER[2] :
ID=3 WHENEVER ($AC_TIMEC==0) AND ($AC_BLOCKTYPEINFO==5002) DO $AC_MARKER[2]=$AC_MARKER[2]+1
...
```

10.4 Actions dans des actions synchrones

10.4.1 Actions possibles dans les actions synchrones

Dans les actions synchrones, les actions se composent d'affectations de valeurs, d'appels de fonctions ou de paramètres, de mots-clés ou de cycles technologiques. Des exécutions complexes sont possibles via des opérateurs.

Les applications possibles sont les suivantes :

- Calculs d'expressions complexes dans la période d'appel de l'interpolateur
- Déplacements d'axes et commandes de la broche
- Modifier et évaluer en ligne les données de réglage des actions synchrones (comme par exemple le transfert des positions et des horodatages de cames logicielles à l'interface AP ou à la périphérie CN)
- sortie de fonctions auxiliaires à l'interface AP
- Mettre en oeuvre des fonctions de sécurité supplémentaires
- Régler le déplacement forcé, la correction d'outil en ligne et la régulation d'écartement
- Exécuter des actions dans tous les modes de fonctionnement
- influer sur les actions synchrones à partir de l'interface AP
- Exécuter des cycles technologiques
- Sortie de signaux numériques et analogiques
- Saisir les performances des actions synchrones dans la période d'appel de l'interpolateur et le temps de calcul du régulateur de position à des fins d'évaluation du taux de charge
- possibilités de diagnostic dans l'interface utilisateur

Action synchrone	Description
DO $V…= DO $A...=	Affectation (valeurs servo) Affectation de variable (variable d'exécution)
DO $AC…[n]= DO $AC_MARKER[n]= DO $AC_PARAM[n]=	Variable d'exécution spéciale Lecture ou écriture de marque d'action synchrone Lecture ou écriture de paramètre d'action synchrone
DO $R[n]=	Lecture ou écriture de variable de calcul
DO $MD...= DO $$SD...=	Lecture de la valeur du paramètre machine au moment de l'interpolation Ecriture de la valeur de la donnée de réglage dans l'exécution des blocs
DO $AC_TIMER[n]=valeur de départ	Temporisateur
DO $AC_FIFO1[n] …FIFO10[n]=	Variables FIFO
DO $AC_BLOCKTYPE= DO $AC_BLOCKTYPEINFO= DO $AC_SPLITBLOCK=	Interpréter le bloc courant (variable d'exécution)
DO M-, S et H par exemple M07	Sortie des fonctions auxiliaires M, S et H
DO RDISABLE	mise en place d'un blocage de la lecture
DO STOPREOF	Annulation de l'arrêt du prétraitement des blocs
DO DELDTG	Effacement rapide de la distance restant à parcourir, sans arrêt du prétraitement des blocs
FTCDEF(polyn., LL, UL , coeff.) DO SYNFCT(polyn., sortie, entrée)	Définition de polynômes Activation de fonctions synchrones : Régulation AC
DO FTOC	Correction d'outil en ligne
DO G70/G71/G700/G710	Définir les système d'unités pour les tâches de positionnement (cotes en pouces ou en mm)
DO POS[axe]= / DO MOV[axe]= DO SPOS[broche]=	Démarrer/positionner/arrêter des axes de commande Démarrer/positionner/arrêter une broche
DO MOV[Axe] = valeur	Lancer/arrêter des déplacements sans fin d'un axe de commande
DO POS[Axe]= FA [Axe]=	Avance d'axe FA
ID=1 ... DO POS[axe]= FA [axe]= ID=2 ... DO POS[axe]= $AA_IM[axe] FA [axe]=	Positionnement à partir d'actions synchrones
DO PRESETON(axe, valeur)	Préréglage des mémoires de valeurs réelles (Preset à partir d'actions synchrones)
ID=1 EVERY $A_IN[1]=1 DO M3 S… ID=2 EVERY $A_IN[2]=1 DO SPOS=	Démarrer/positionner/arrêter une broche
DO TRAILON(AA,AP,facteur de couplage) DO LEADON(AA,AP,NRCTAB,OVW)	Activer les déplacements conjugués Activer le couplage par valeur pilote
DO MEAWA(axe)= DO MEAC(axe)=	Activer la mesure axiale Activer la mesure continue
DO [tableau n, m]=SET(valeur, valeur, ...) DO [tableau n, m]=REP(valeur, valeur, ...)	Initialisation de variables de champ avec listes de valeurs Initialisation de variables de champ avec les mêmes valeurs

Action synchrone	Description
DO SETM(n°_marque)	Définition d'une marque d'attente
DO CLEARM(n°_marque)	Suppression d'une marque d'attente
DO SETAL(N°_alarme)	Régler l'alarme de cycle (fonction de sécurité supplémentaire)
DO FXS[axe]=	Activer le déplacement en butée
DO FXST[axe]=	Modifier le couple de blocage
DO FXSW[axe]=	Modifier la fenêtre de surveillance
DO FOCON[axe]=	Activation du déplacement avec réduction du couple/force (effet modal) FOC
DO FOCOF[axe]=	Désactivation du déplacement avec réduction du couple/force (action synchrone à effet non modal)
ID=2 EVERY $AC_BLOCKTYPE==0 DO $R1=$AC_TANEB	Angle de la tangente à la trajectoire au point final du bloc courant et de la tangente à la trajectoire au point de départ du bloc suivant programmé
DO $AA_OVR=	Correction de l'avance axiale
DO $AC_OVR=	Correction d'avance tangentielle
DO $AA_PLC_OVR	Correction axiale définie par l'AP
DO $AC_PLC_OVR	Correction tangentielle définie par l'AP
DO $AA_TOTAL_OVR	Correction axiale résultante
DO $AC_TOTAL_OVR	Correction tangentielle résultante
$AN_IPO_ACT_LOAD=	Temps de calcul IPO courant
$AN_IPO_MAX_LOAD=	Temps de calcul IPO le plus long
$AN_IPO_MIN_LOAD=	Temps de calcul IPO le plus court
$AN_IPO_LOAD_PERCENT=	Temps de calcul IPO courant par rapport à la période d'appel de l'interpolateur
$AN_SYNC_ACT_LOAD=	Temps de calcul courant pour action synchrone dans tous les canaux
$AN_SYNC_MAX_LOAD=	Temps de calcul le plus long pour action synchrone dans tous les canaux
$AN_SYNC_TO_IPO=	Pourcentage de l'action synchrone totale
DO TECCYCLE	Exécuter un cycle technologique
DO LOCK(n, n, ...)	Bloquer
DO UNLOCK(n, n, ...)	Débloquer
DO RESET(n, n, ...)	RESET d'un cycle technologique
CANCEL(n, n, ...)	Effacer des actions synchrones modales désignées par ID(S) dans le programme pièce

10.4.2 Sortie de fonctions auxiliaires

Fonction

Instant de sortie

La sortie des fonctions auxiliaires s'effectue dans l'action synchrone directement au moment de la sortie de l'action. L'instant de sortie de la fonction auxiliaire, rangé dans un paramètre machine, { n'est pas pris en compte.

Le moment de la sortie est déterminé lorsque la condition est remplie.

Exemple :

Enclencher l'arrosage à une position d'axe bien définie :

```
WHEN $AA_IM[X]>=15 DO M07 POS[X]=20 FA[X]=250
```

Mots-clés autorisés dans les actions synchrones à effet non modal (sans ID modal)

L'utilisation de fonctions auxiliaires dans des actions synchrones à effet non modal (sans ID modal) est uniquement possible avec les mots-clés `WHEN` ou `EVERY`.

Remarque

Les fonctions auxiliaires suivantes ne sont pas autorisées dans une action synchrone :

- `M0`, `M1`, `M2`, `M17`, `M30` : Arrêt/fin du programme (`M2`, `M17`, `M30` possibles dans le cycle technologique)
- `M6` ou fonctions M réglées par le biais de paramètres machine pour le changement d'outil

---

Exemple

Code de programme	Commentaire
`WHEN $AA_IW[Q1]>5 DO M172 H510`	`; Lorsque la position réelle de l'axe Q1 dépasse 5 mm, les fonctions auxiliaires M172 et H510 sont transférées à l'AP.`

10.4.3 Activation du blocage de la lecture (RDISABLE)

Fonction

Lorsque la condition est remplie, RDISABLE arrête la poursuite du traitement des blocs du programme principal. Les actions synchrones au déplacement, qui sont programmées, continuent à être traitées et les blocs suivants continuent à être prétraités.

Dans le contournage, au début d'un bloc avec RDISABLE, il y a toujours déclenchement d'un arrêt précis dans des actions synchrones, que RDISABLE soit actif ou non.

Exemple

Lancer un programme dans la période d'appel de l'interpolateur, en fonction des entrées externes.

```
Code de programme                                  Commentaire
...
WHENEVER $A_INA[2]<7000 DO RDISABLE                ; Si la tension est inférieure à 7V
                                                     à l'entrée 2,
                                                     bloquer la poursuite du programme
                                                     (1000= 1V).
N10 G1 X10                                         ; Lorsque la condition est remplie,
                                                     prise d'effet du blocage de la
                                                     lecture à la fin de N10.
N20 G1 X10 Y20
...
```

10.4.4 Annulation de l'arrêt du prétraitement des blocs (STOPREOF)

Fonction

Quand le prétraitement de blocs a été arrêté par programmation explicite de STOPRE ou par activation implicite par le biais d'une action synchrone active, l'instruction STOPREOF annule cet arrêt du prétraitement des blocs après le prochain bloc d'usinage et ce, dès que la condition est remplie.

Remarque

STOPREOF doit être programmé avec le mot-clé `WHEN` et de façon non modale (sans numéro ID).

Exemple

Branchement rapide dans le programme en fin de bloc.

Code de programme	Commentaire
`WHEN $AC_DTEB<5 DO STOPREOF`	`; Lorsque la distance par rapport au point de fin du bloc est < à 5 mm, annuler l'arrêt du prétraitement des blocs.`
`G01 X100`	`; Après exécution de l'interpolation linéaire, annulation de l'arrêt du prétraitement de blocs.`
`IF $A_INA[7]>500 GOTOF MARQUE1=X100`	`; Quand la tension dépasse 5 V à l'entrée 7, saut à l'étiquette 1.`

10.4.5 Effacement de la distance restant à parcourir (DELDTG)

Fonction

Un effacement de la distance restant à parcourir peut être déclenché pour la trajectoire et les axes indiqués, en fonction d'une condition.

A disposition :

- Effacement rapide de la distance restant à parcourir, avec préparation
- Effacement de la distance restant à parcourir sans préparation

L'effacement de la distance restant à parcourir avec préparation, programmé avec DELDTG, permet de réagir très rapidement à l'événement déclencheur. Il est utilisé par conséquent dans des applications critiques en temps, par exemple quand :

- le temps entre l'effacement de la distance restant à parcourir et le lancement du bloc suivant doit être extrêmement court.
- la condition nécessaire pour l'effacement de la distance restant à parcourir est remplie avec une très forte probabilité.

Remarque

Le descripteur d'axe entre crochet suivant le DELDTG n'est valable que pour **un** axe de positionnement.

Syntaxe

Effacement de la distance restant à parcourir pour la trajectoire

```
DO DELDTG
```

Effacement de la distance axiale restant à parcourir

```
DO DELDTG(axe1) DELDTG(axe2) ...
```

Exemple d'effacement rapide de la distance restant à parcourir pour la trajectoire

Code de programme	Commentaire
WHEN $A_IN[1]==1 DO DELDTG	
N100 G01 X100 Y100 F1000	; Lorsque l'entrée est activée, arrêt du déplacement.
N110 G01 X…	
IF $AA_DELT>50…	

Exemple d'effacement axial rapide de la distance restant à parcourir

```
Code de programme                                   Commentaire
Interruption d'un déplacement de
positionnement :
ID=1 WHEN $A_IN[1]==1 DO MOV[V]=3 FA[V]=700     ;  Démarrage de l'axe
WHEN $A_IN[2]==1 DO DELDTG(V)                   ;  Effacement de la distance restant
                                                   à parcourir, arrêt de l'axe avec
                                                   MOV=0
Effacer la distance restant à parcourir en
fonction de la tension à l'entrée :
WHEN $A_IN[5]==8000 DO DELDTG(V)                ;  Dès que la tension dépasse 8 V à
                                                   l'entrée 5, effacement de la
                                                   distance restant à parcourir de
                                                   l'axe X1. Le déplacement avec
                                                   interpolation se poursuit.
POS[X1]=100 FA[X1]=10 G1 Z100 F1000
```

Autres informations

A la fin du bloc de déplacement, dans lequel on a déclenché un effacement de la distance restant à parcourir avec préparation, un arrêt du prétraitement des blocs est activé de façon implicite.

Ainsi, le déplacement d'interpolation et les déplacements d'axes de positionnement sont interrompus ou stoppés par un effacement rapide de la distance à parcourir à la fin du bloc.

Remarque

Préparation de l'effacement de la distance restant à parcourir :

- ne peut pas être appliqué quand la correction du rayon d'outil est active.
- ne doit être programmé que dans des actions synchrones non modales (sans numéro ID).

10.4.6 Définition d'un polynôme (FCTDEF)

Fonction

Avec FCTDEF, vous pouvez définir des polynômes du 3e degré de la forme $y=a_0+a_1x+a_2x^2+a_3x^3$. Ces polynômes sont utilisés par la correction d'outil en ligne FTOC et par la fonction d'évaluation SYNFCT.

Syntaxe

```
FCTDEF(n° de polynôme,LLIMIT,ULIMIT,a0,a1,a2,a3)
```

Signification

`N° de polynôme`	Numéro du polynôme du 3e degré
`LLIMIT`	limite inférieure de la valeur de la fonction
`ULIMIT`	limite supérieure de la valeur de la fonction
`a0, a1, a2, a3`	coefficients du polynôme

Ces valeurs sont aussi accessibles par des variables système

`$AC_FCTLL[n]`	limite inférieure de la valeur de la fonction
`$AC_FCTUL[n]`	limite supérieure de la valeur de la fonction
`$AC_FCT0[n]`	a_0
`$AC_FCT1[n]`	a_1
`$AC_FCT2[n]`	a_2
`$AC_FCT3[n]`	a_3

Remarque

Ecrire des variables système

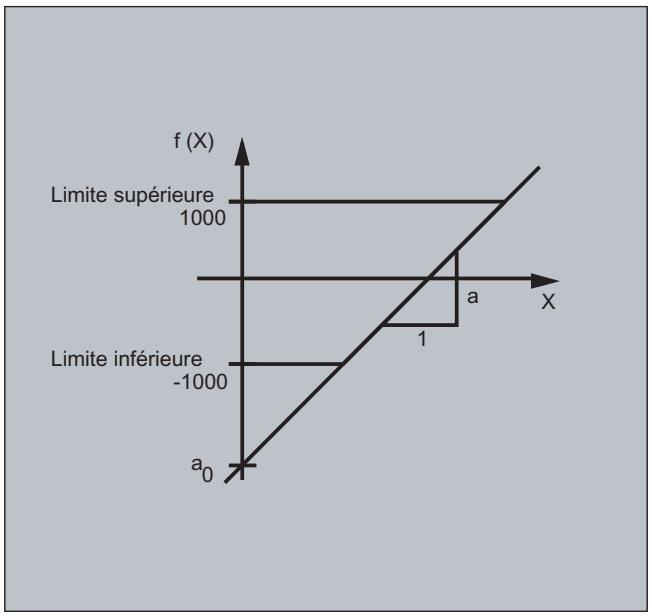
- Les variables système peuvent être écrites à partir du programme pièce ou d'une action synchrone. Pour l'écriture à partir du programme pièce, il convient de programmer STOPRE pour faire la synchronisation avec les blocs.
- Les variables système `$AC_FCTLL[n]`, `$AC_FCTUL[n]`, `$AC_FCT0[n]` à `$AC_FCTn[n]` sont modifiables à partir des actions synchrones

Dans le cas d'une écriture à partir d'actions synchrones, les coefficients du polynôme et les valeurs limites de la fonction sont actifs immédiatement.

Exemple de polynôme pour un segment de droite

Avec une limite supérieure 1000, une limite inférieure -1000, l'ordonnée `a0=$AA_IM[X]` et la pente 1, la définition du polynôme est la suivante :

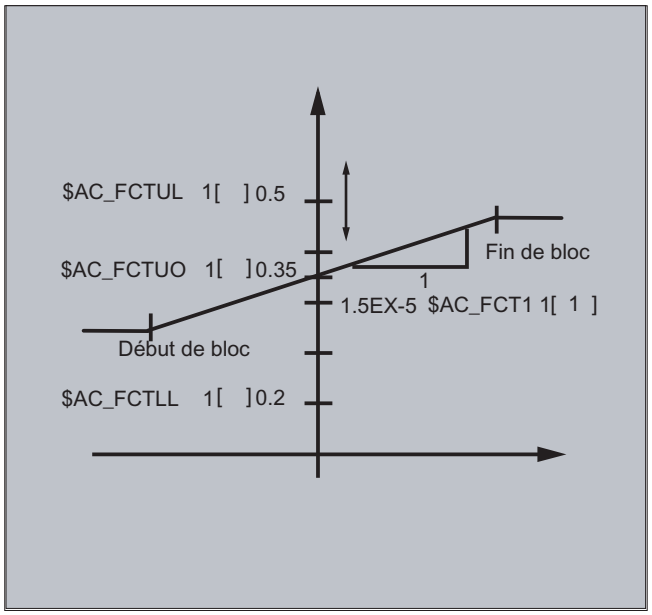
```
FCTDEF(1, -1000,1000,$AA_IM[X],1)
```



Exemple de pilotage de la puissance d'un laser

L'une des applications possibles de la définition d'un polynôme est le pilotage de la puissance d'un laser.

Le pilotage de la puissance d'un laser signifie :
corriger une sortie analogique en fonction par exemple de la vitesse tangentielle.



Code de programme	Commentaire
`$AC_FCTLL[1]=0.2`	`;` Définition des coefficients du polynôme
`$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`	`;` Modification en ligne de la limite supérieure.
`ID=2 DO SYNFCT(1,$A_OUTA[1],$AC_VACTW)`	`;` En fonction de la vitesse tangentielle (rangée dans $AC_VACTW), la puissance du laser est pilotée par le biais de la sortie analogique 1

Remarque

L'utilisation du polynôme défini plus haut se fait avec SYNFCT.

10.4.7 Fonction synchrone (SYNFCT)

Fonction

SYNFCT calcule la valeur de sortie d'un polynôme de 3e degré pondéré avec les variables d'entrée. Le résultat figure dans les variables de sortie et est limité par le haut et par le bas.

On utilise la fonction d'évaluation

- pour la régulation AC (Adaptive Control),
- pour le pilotage de la puissance d'un laser,
- pour le forçage de positions.

Syntaxe

```
SYNFCT(n° polynôme, sortie variable d'exécution, entrée variable d'exécution)
```

Signification

Comme variables de sortie, il est possible de choisir des variables qui sont prises en considération

- avec un effet correcteur additif,
- avec un effet correcteur multiplicatif,
- comme décalage de position,
- directement,

dans l'opération d'usinage.

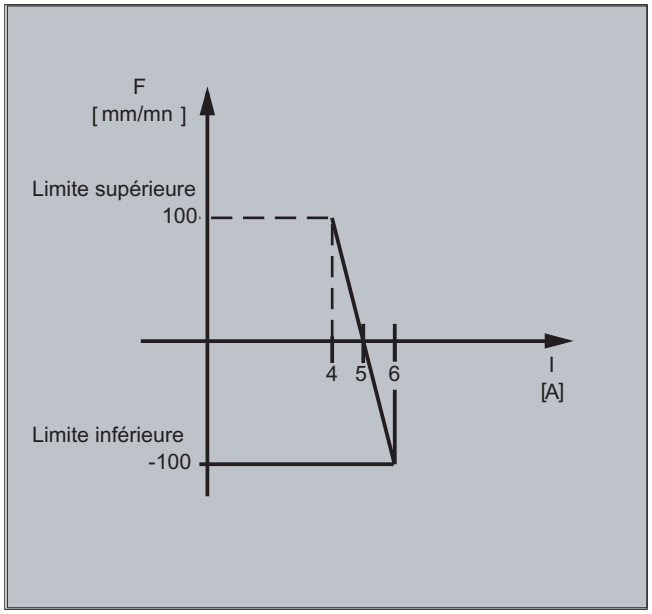
`DO SYNFCT`	Activation de la fonction d'évaluation
`N° de polynôme`	Polynôme défini avec FCTDEF (voir "Définition des polynômes")
`Sortie de variable d'exécution`	Ecrire une variable d'exécution
`Entrée de variable d'exécution`	Lire une variable d'exécution

Exemple de régulation AC (additive)

Correction additive de l'avance programmée

Une avance programmée doit être corrigée de façon additive en fonction du courant de l'axe X (axe de pénétration) :

l'avance peut varier de +/- 100 mm/min, sachant que le courant fluctue de +/-1A autour du point de travail à 5A.



1. Définition d'un polynôme

Détermination des coefficients

$y = f(x) = a_0 + a_1x + a_2x^2 + a_3x^3$

a_1 = -100mm/1 min A

a_0 = -(-100)*5 =500

$a_2 = a_3 = 0$ (pas d'opérateur quadratique, ni cubique)

Limite supérieure = 100

Limite inférieure = -100

Il en résulte :

```
FCTDEF(1,-100,100,500,-100,0,0)
```

2. Activation de la régulation AC

```
ID=1 DO SYNFCT(1,$AC_VC,$AA_LOAD[x])
```

; Avec `$AA_LOAD[x]` lire la charge de l'axe (en % du courant d'entraînement maxi),
; puis calculer la correction de l'avance tangentielle avec le polynôme défini plus haut.

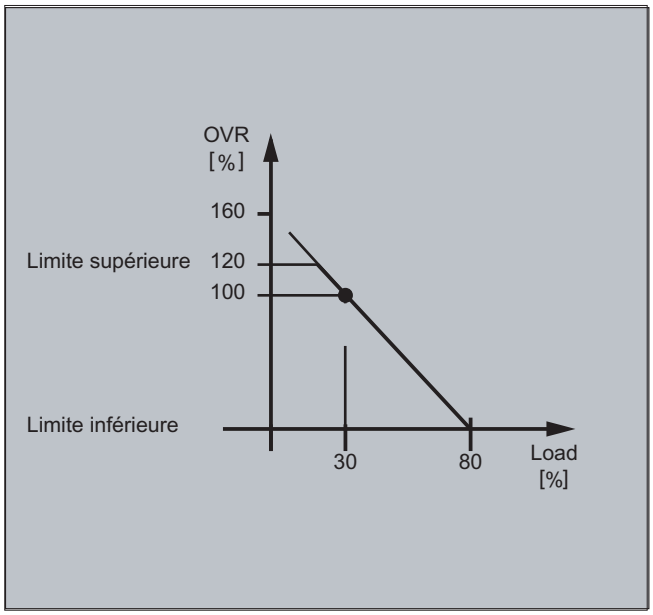
Exemple de régulation AC (multiplicative)

Correction multiplicative de l'avance programmée

L'avance programmée doit être corrigée de façon multiplicative, sachant que l'avance – en fonction de la charge de l'entraînement – ne doit pas dépasser certaines limites :

- Quand la charge d'entraînement est de 80%, le prétraitement des blocs doit être arrêté : Correction = 0.
- Quand la charge d'entraînement est de 30%, le déplacement peut se faire avec l'avance programmée :
  Correction = 100%.

La vitesse d'avance ne doit pas être dépassée de plus de 20 % maximum :
Correction maxi = 120%.



1. Définition d'un polynôme

Détermination des coefficients

$y = f(x) = a_0 + a_1x + a_2x^2 + a_3x^3$

$a_1 = -100\%/(80-30)\% = -2$

$a_0 = 100 + (2*30) = 160$

$a_2 = a_3 = 0$ (pas d'opérateur quadratique, ni cubique)

Limite supérieure = 120

Limite inférieure = 0

Il en résulte :

```
FCTDEF(2,0,120,160,-2,0,0)
```

2. Activation de la régulation AC

```
ID=1 DO SYNFCT(2,$AC_OVR,$AA_LOAD[x])
```

; Avec `$AA_LOAD[x]` lire la charge de l'axe (en % du courant d'entraînement maxi),
; puis calculer la correction de l'avance avec le polynôme défini plus haut.

10.4.8 Régulation d'écartement avec correction limitée ($AA_OFF_MODE)

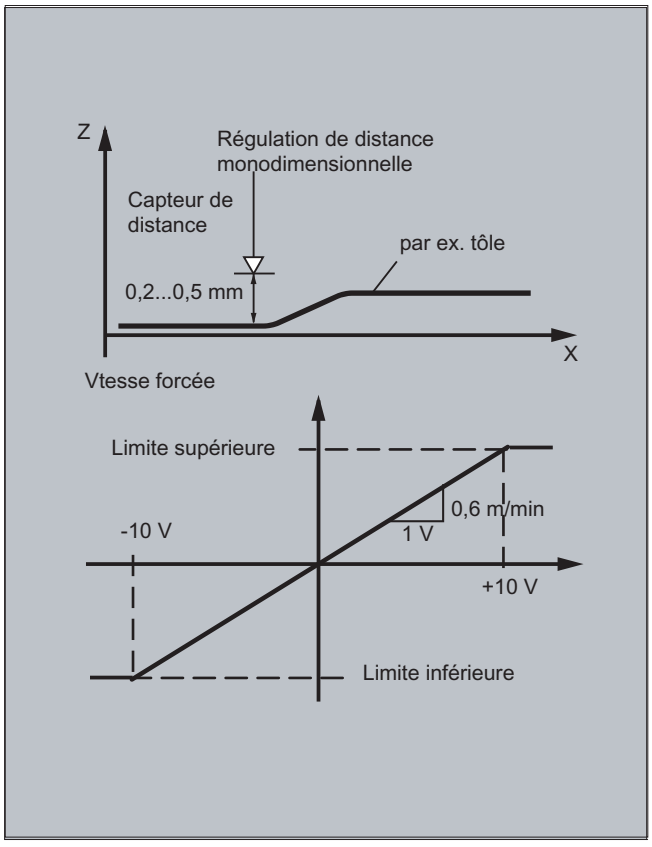
Remarque

Cette fonction n'est pas disponible pour SINUMERIK 828D.

Fonction

Le calcul intégrant des valeurs d'écartement s'effectue avec un contrôle de zone limite :

$AA_OFF_MODE = 1



IMPORTANT
Le gain de la boucle de régulation de niveau supérieur dépend du réglage de la période d'appel de l'interpolateur. Aide : lecture du paramètre machine de la période d'appel de l'interpolateur et prise en compte dans le calcul.

Remarque

Limitation de la vitesse de l'interpolateur forcé par le paramètre machine MD 32020: JOG_VELO pour une période d'appel de l'interpolateur égale à 12 ms.

Formule de vitesse :

$$\frac{0.120\text{mm}}{0.6\text{ms}} / \text{mV} = 0.6 \frac{\text{m}}{\text{min}} / \text{V}$$

Exemple

Sous-programme "AON" : Régulation d'écartement activée

Code de programme	Commentaire
PROC AON	
$AA_OFF_LIMIT[Z]=1	; Définir un seuil.
FCTDEF(1, -10, +10, 0, 0.6, 0.12)	; Définition du polynôme
ID=1 DO SYNFCT(1,$AA_OFF[Z],$A_INA[3])	; Régulation d'écartement activée.
ID=2 WHENEVER $AA_OFF_LIMIT[Z]<>0 DO $AA_OVR[X] = 0	; Blocage de l'axe X en cas de dépassement de la zone limite.
RET	
ENDPROC	

Sous-programme "AOFF" : Régulation d'écartement désactivée

Code de programme	Commentaire
PROC AOFF	
CANCEL(1)	; Suppression de l'action synchrone de régulation d'écartement
CANCEL(2)	; Suppression du contrôle de zone limite
RET	
ENDPROC	

Programme principal "MAIN"

Code de programme	Commentaire
AON	; Régulation d'écartement activée
...	
G1 X100 F1000	
AOFF	; Régulation d'écartement désactivée
M30	

Autres informations

Décalage de position dans le système de coordonnées de base

La variable système $AA_OFF[axe] permet de superposer un déplacement à chaque axe du canal. Ce déplacement forcé agit comme un décalage de position dans le système de coordonnées de base.

Le décalage de position ainsi programmé est immédiatement appliqué sur l'axe en question, que cet axe soit programmé pour être déplacé ou non.

Limiter la sortie de variable d'exécution :

Il est possible de limiter la valeur de correction absolue (sortie de variable d'exécution) à la valeur enregistrée dans la donnée de réglage SD43350 $SA_AA_OFF_LIMIT.

Le type de superposition de l'écartement est défini dans le paramètre machine MD36750 $MA_AA_OFF_MODE.

Valeur	Signification
0	Evaluation proportionnelle
1	Evaluation intégrante

La variable système $AA_OFF_LIMIT[axe] permet de savoir, en fonction de la direction, si la valeur de correction se situe dans la zone limite. Cette variable système peut être interrogée depuis les actions synchrones et, par exemple, arrêter un axe ou déclencher une alarme lorsqu'une valeur limite est atteinte.

0: la valeur de correction n'est pas dans la zone limite
1 la limite de la valeur de correction en sens positif a été atteinte
-1: la limite de la valeur de correction en sens négatif a été atteinte

10.4.9 Correction d'outil en ligne (FTOC)

Fonction

FTOC permet un déplacement forcé pour un axe géométrique, selon un polynôme programmé avec FCTDEF et en fonction d'une valeur de référence qui peut être p. ex. la valeur réelle d'un axe.

Le coefficient a_0 de la définition de fonction FCTDEF(...) est évalué avec FTOC. Les limites supérieure et inférieure dépendent de a_0.

FTOC permet de programmer des corrections d'outil modales en ligne ou des régulations d'écartement comme actions synchrones.

On utilise cette fonction pour la rectification de la pièce et dressage de la meule dans le même canal ou dans des canaux différents (canal d'usinage et de dressage).

Les conditions marginales et les définitions pour le dressage des meules sont valables pour FTOC de manière analogue à la correction d'outil en ligne avec PUTFTOCF (voir "Correction d'outil en ligne (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) (Page 404)").

Syntaxe

```
FCTDEF(<fonction>,<LLimit>,<ULimit>,<a0>,<a1>,<a2>,<a3>)
FTOC(<fonction>,<valeur de référence>,<paramètre d'outil>,<canal>,<broche>)
...
```

Signification

FCTDEF : FCTDEF définit la fonction polynômiale pour FTOC.

Paramètre :

<fonction> : Numéro de la fonction polynômiale
Type : INT
Plage de valeurs : 1 ... 3

<LLimit> : valeur limite inférieure
Type : REAL

<ULimit> : valeur limite supérieure
Type : REAL

<a0> ... <a3> : Coefficients de la fonction polynômiale
Type : REAL

`DO FTOC` : Exécution de la fonction "Ecriture continue modale de la correction d'outil en ligne"

Paramètre :

`<fonction>` : Numéro de la fonction polynômiale

Type : INT

Plage de valeurs : 1 ... 3

Nota :
doit correspondre avec l'indication dans `FCTDEF`.

`<valeur de référence>` : Variable d'exécution pour laquelle une valeur de la fonction doit être calculée avec la fonction polynômiale définie avec `FCTDEF`.

Type : VAR REAL

`<paramètre d'outil>` : Numéro du paramètre d'usure (longueur 1, 2 ou 3) dans lequel additionner la valeur de correction.

Type : INT

`<canal>` : Numéro du canal dans lequel la correction d'outil en ligne doit être active.

Type : INT

Nota :
une indication est uniquement requise lorsque la correction ne doit pas être effective dans le canal actif.

`<broche>` : Numéro de la broche dans laquelle la correction d'outil en ligne doit être active.

Type : INT

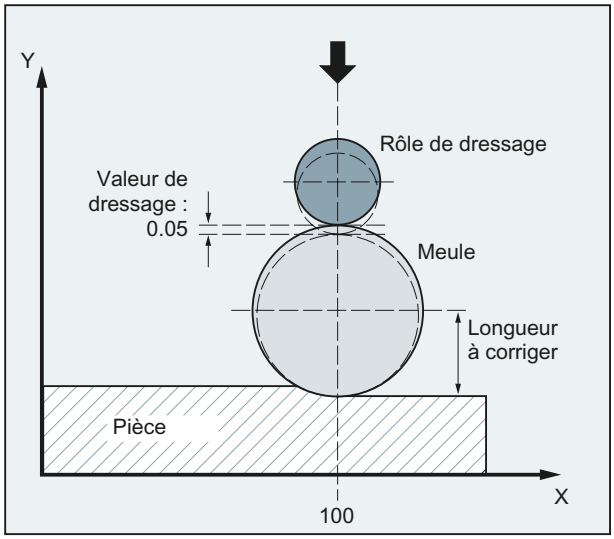
Nota :
une indication est uniquement requise lorsqu'il s'agit de corriger une meule non active à la place de l'outil actif mis en oeuvre.

Remarque

`FTOCON` doit être activé dans le canal de destination.

Exemple

Il s'agit de corriger la longueur de la meule active mise en oeuvre.



Code de programme	Commentaire
`FCTDEF(1,-1000,1000,-$AA_IW[V],1)`	`; Définir la fonction.`
`ID=1 DO FTOC(1,$AA_IW[V],3,1)`	`; Activer la correction d'outil en ligne : la valeur réelle de l'axe V est la valeur d'entrée pour le polynôme 1. Dans le canal 1, le résultat est additionné à la longueur 3 de la meule active comme valeur de correction.`
`WAITM(1,1,2)`	`; Synchronisation avec canal d'usinage.`
`G1 V-0.05 F0.01 G91`	`; Mouvement de pénétration pour le dressage.`
`G1 V-0.05 F0.02`	
`...`	
`CANCEL(1)`	`; Désactiver la correction en ligne.`
`...`	

10.4.10 Correction en ligne de la longueur d'outil ($AA_TOFF)

Fonction

Avec la variable système $AA_TOFF[ ], les longueurs d'outil effectives peuvent être imposées en temps réel dans les trois directions de l'outil.

Les trois descripteurs d'axes géométriques servent d'indices. De cette façon, le nombre des directions de correction activées est déterminé par les axes géométriques qui sont actifs au même moment.

Toutes les corrections peuvent être actives simultanément.

Syntaxe

```
N... TRAORI
N... TOFFON(X,<valeur offset>)
N... WHEN TRUE DO $AA_TOFF[X]
N... TOFFON(Y,<valeur offset>)
N... WHEN TRUE DO $AA_TOFF[Y]
N... TOFFON(Z,<valeur offset>)
N... WHEN TRUE DO $AA_TOFF[Z]
```

Signification

`TOFFON` : **Activation** de la correction en ligne de la longueur d'outil

- `X, Y, Z` : Direction d'outil dans laquelle la correction en ligne de la longueur d'outil doit être efficace.
- `<valeur de l'offset>` : Dès que la fonction est activée, une valeur offset peut être indiquée pour la direction dans laquelle doit s'effectuer la correction et cette valeur fait immédiatement l'objet d'un mouvement de correction.

`TOFFOF` : **Remise à zéro** de la correction en ligne de la longueur d'outil

Les valeurs de correction dans le sens de correction spécifié sont remises à zéro et un arrêt du prétraitement des blocs est déclenché.

`$AA_TOFF[X]=<valeur>` : Superposition dans la direction X

`$AA_TOFF[Y]=<valeur>` : Superposition dans la direction Y

`$AA_TOFF[Z]=<valeur>` : Superposition dans la direction Z

Exemples

Exemple 1 : Activation de la correction de longueur d'outil

Code de programme	Commentaire
N10 TRAORI(1)	; Activation de la transformation.
N20 TOFFON(Z)	; Activation de la correction en ligne de la longueur d'outil pour la direction Z de l'outil.
N30 WHEN TRUE DO $AA_TOFF[Z]=10 G4 F5	; Interpolation d'une valeur de correction de longueur d'outil égale à 10 pour la direction Z de l'outil.
N40 TOFFON(X)	; Activation de la correction en ligne de la longueur d'outil pour la direction X de l'outil.
N50 ID=1 DO $AA_TOFF[X]=$AA_IW[X2] G4 F5	; Dans la direction d'outil X, exécution d'une correction en fonction de la position de l'axe X2.
...	
	; Affectation de la correction actuelle dans la direction X. Remise à 0 de la correction de longueur d'outil dans la direction X de l'outil :
N100 XOFFSET=$AA_TOFF_VAL[X] N120 TOFFON(X,-XOFFSET) G4 F5	

Exemple 2 : Désactivation de la correction de longueur d'outil

Code de programme	Commentaire
N10 TRAORI(1)	; Activation de la transformation.
N20 TOFFON(X)	; Activation de la correction en ligne de la longueur d'outil pour la direction X de l'outil.
N30 WHEN TRUE DO $AA_TOFF[X]=10 G4 F5	; Interpolation d'une valeur de correction de longueur d'outil égale à 10 pour la direction X de l'outil.
...	
N80 TOFFOF(X)	; L'offset de position de la direction d'outil X est supprimé : ...$AA_TOFF[X]=0 Aucun axe n'est déplacé. Le décalage de position est ajouté à la position actuelle dans le SCP conformément à l'orientation donnée.

10.4.11 Déplacements de positionnement

Fonction

Vous pouvez positionner des axes de façon totalement asynchrones au programme pièce, à l'aide d'actions synchrones. La programmation d'axes de positionnement dans des actions synchrones est recommandée pour les processus cycliques ou ceux pilotés par événements. Les axes programmés dans des actions synchrones s'appellent **axes de commande**.

Programmation

Bibliographie :
/PG/ Manuel de programmation Notions de base ; chapitre "Instructions de déplacement"
/FBSY/ Description des fonctions d'actions synchrones ; "Lancement d'axes de commandes"

Paramètres

Le système d'unités pour les tâches de positionnement avec des actions synchrones est défini à l'aide des codes G `G70`/`G71`/`G700`/`G710` .

La programmation de fonctions G dans une action synchrone permet de fixer l'évaluation INCH/METRIC pour cette action synchrone indépendamment du contexte du programme pièce.

10.4.12 Positionnement d'un axe (POS)

Fonction

Contrairement à la programmation à partir du programme pièce, le déplacement axial de positionnement n'a aucun impact sur l'exécution du programme pièce.

Syntaxe

```
POS[axe]=valeur
```

Signification

`DO POS`	Lancer/positionner un axe de commande
`Axe`	Nom de l'axe qui doit être déplacé
`Valeur`	Indication de la valeur à parcourir (selon le mode de déplacement)

Exemples

Exemple 1 :

```
Code de programme                                  Commentaire
ID=1 EVERY $AA_IM[B]>75 DO POS[U]=100              ; Déplacement incrémental de l'axe U de 100 (inch/mm) ou sur
                                                     la position 100 (inch/mm) par rapport à l'origine de la
                                                     commande (en fonction du mode de déplacement).
                                                   ; Déplacer l'axe U selon la distance calculée à partir des
                                                     variables d'exécution :
ID=1 EVERY $AA_IM[B]>75 DO POS[U]=$AA_MW[V]-$AA_IM[W]+13.5
```

Exemple 2 :

Trajet de positionnement de l'axe de positionnement influencé par l'environnement du programme
(pas de fonction G dans la partie action de l'action synchrone) :

```
Code de programme                                  Commentaire
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
```

Dans la partie action de l'action synchrone, l'instruction `G71` entraîne une détermination explicite (métrique) du trajet de positionnement de l'axe de positionnement, indépendamment de l'environnement du programme :

```
Code de programme                                    Commentaire
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 positionne toujours sur
                                                       10mm)
N170 G71 Z10 F10                                     ; Z=10mm X=10mm
N180 M30
```

Si le déplacement de l'axe ne doit pas démarrer au début du bloc, la correction de l'axe peut être maintenue à 0 par une action synchrone jusqu'à l'instant souhaité :

```
Code de programme                                    Commentaire
WHENEVER $A_IN[1]==0 DO $AA_OVR[W]=0 G01 X10 Y25 F750 POS[W]=1500 FA=1000
                                                     ; L'axe de positionnement reste arrêté aussi longtemps
                                                       que l'entrée numérique 1=0.
```

10.4.13 Position dans la zone de référence prescrite (POSRANGE)

Fonction

La fonction POSRANGE( ) permet de déterminer si la position de consigne courante interpolée d'un axe se trouve dans une fenêtre autour d'une position de référence prescrite. Les indications de position peuvent se rapporter à des systèmes de coordonnées définissables.

Lors de l'interrogation de la position réelle de l'axe, la correction modulo est prise en compte.

Remarque

La fonction peut seulement être appelée de l'action synchrone. Lors de l'appel à partir du programme pièce survient l'alarme 14091 %1 bloc %2 fonction non autorisée, indice : %3 appelé avec l'indice 5.

Syntaxe

```
BOOL POSRANGE(Axe, Refpos, Winlimit,[Coord])
```

Signification

`BOOL POSRANGE`	La position courante de l'axe de commande est dans la fenêtre de la position de référence donnée.
`AXIS <Axe>`	Descripteur de l'axe machine, canal ou géométrique
`REAL Refpos`	Position de référence dans le système de coordonnées Coord
`REAL Winlimit`	Montant indiquant la limite de la fenêtre de position
`INT Coord`	Le SCM est actif en option. Il est possible de : 0 pour SCM (système de coordonnées machine) 1 pour SCB (système de coordonnées de base) 2 pour SPR (système de coordonnées réglable) 3 pour SCP (système de coordonnées pièce)

valeur fonction

Position de consigne courante selon l'indication de la position dans le système de coordonnées prescrit

`valeur fonction : TRUE`	lorsque Refpos(Coord) - abs(Winlimit)
`valeur fonction : FALSE`	≤ Actpos(Coord) ≤ Refpos(Coord) + abs(Winlimit) sinon

10.4.14 Démarrer/arrêter un axe (MOV)

Fonction

Avec MOV[axe]=valeur, vous pouvez démarrer un axe de commande sans indication de la position finale. L'axe en question est déplacé dans le sens programmé, jusqu'à ce qu'une nouvelle instruction de déplacement ou de positionnement indique un autre déplacement ou jusqu'à ce qu'il soit immobilisé par une instruction d'arrêt.

Syntaxe

```
MOV[Axe] = valeur
```

Signification

`DO MOV`	Démarrer un déplacement de l'axe de commande
`Axe`	Nom de l'axe qui doit être démarré
`Valeur`	Instruction de démarrage/d'arrêt du déplacement { Le signe détermine le sens du déplacement
	Type de données de la valeur : INTEGER.
`Valeur >0 (normalement +1)`	sens positif
`Valeur <0 (normalement -1)`	sens négatif
`Valeur ==0`	Arrêter le déplacement axial

Remarque

Si vous arrêtez un axe indexé avec `MOV[Axe] = 0` , l'arrêt se fera à la prochaine position indexée.

Exemple

Code de programme	**Commentaire**
`... DO MOV[U]=0`	`;  L'axe U est arrêté.`

10.4.15 Permutation d'axe (RELEASE, GET)

Fonction

Pour un changement d'outil, les axes de commande concernés peuvent être demandés en tant qu'action d'une action synchrone avec GET(Axe). Le type d'axe affecté à ce canal et le droit d'interpolation qui lui est corrélé à ce moment peut être interrogé via la variable système $AA_AXCHANGE_TYP. En fonction de l'état réel et du canal possédant le droit d'interpolation courant de cet axe, différents processus sont possibles.

Lorsque le changement d'outil est terminé, cet axe de commande peut alors être validé comme action d'une action synchrone avec RELEASE(Axe) pour le canal.

Constructeur de la machine-outil

L'axe concerné doit être affecté au canal par le biais des paramètres machine. Tenez compte des indications du constructeur de la machine

Syntaxe

`GET(axe[,axe{,...}])` demande d'axe

`RELAESE(axe[,axe{,...}])` déblocage d'axe

Signification

`DO RELEASE`	Débloquer l'axe en tant qu'axe neutre
`DO GET`	Aller chercher l'axe pour permutation d'axe
`Axe`	Nom de l'axe qui doit être démarré

Exemple de déroulement de programme pour une permutation d'axes de deux canaux

L'axe Z est connu dans le 1er et dans le 2ème canal.

Déroulement du programme dans le 1er canal:

```
Code de programme                          Commentaire
WHEN TRUE DO RELEASE(Z)                    ; L'axe Z devient un axe banalisé.
WHENEVER($AA_TYP[Z]==1) DO RDISABLE        ; Blocage de la lecture tant que l'axe Z est
                                             un axe du programme
N110 G4 F0.1
WHEN TRUE DO GET(Z)                        ; L'axe Z redevient axe du programme de CN
WHENEVER($AA_TYP[Z]==1) DO RDISABLE        ; Blocage de la lecture jusqu'à ce que l'axe
                                             Z soit un axe du programme
N120 G4 F0.1
WHEN TRUE DO RELEASE(Z)                    ; L'axe Z devient un axe banalisé.
WHENEVER($AA_TYP[Z]==1) DO RDISABLE        ; Blocage de la lecture tant que l'axe Z est
                                             un axe du programme
N130 G4 F0.1                               ;
N140 START(2)                              ; Démarrer le 2ème canal
```

Déroulement du programme dans le 2ème canal:

Code de programme	Commentaire
WHEN TRUE DO GET(Z)	; ;Faire passer l'axe Z dans le 2ème canal
WHENEVER($AA_TYP[Z]==0) DO RDISABLE	; ;Blocage de la lecture tant que l'axe Z est dans ; un autre canal
N210 G4 F0.1	
WHEN TRUE DO GET(Z)	; ;L'axe Z devient axe du programme de CN
WHENEVER($AA_TYP[Z]==1) DO RDISABLE	; ;Blocage de la lecture jusqu'à ce que l'axe Z soit un axe du programme
N220 G4 F0.1	
WHEN TRUE DO RELEASE(Z)	; ;L'axe Z dans le 2ème canal est un axe banalisé
WHENEVER($AA_TYP[Z]==1) DO RDISABLE	; ;Blocage de la lecture tant que l'axe Z est un axe du programme
N230 G4 F0.1	
N250 WAITM(10, 1, 2)	; Synchroniser avec le canal 1

Poursuite du déroulement du programme dans le 1er canal:

Code de programme	Commentaire
N150 WAIM(10, 1, 2)	; Synchroniser avec le canal 2
WHEN TRUE DO GET(Z)	; Faire passer l'axe Z dans ce canal
WHENEVER($AA_TYP[Z]==0) DO RDISABLE	; Blocage de la lecture tant que l'axe Z est dans un autre canal
N160 G4 F0.1	
N199 WAITE(2)	
N999 M30	; En attente de la fin du programme dans le canal 2

Exemple de permutation d'axes dans un cycle technologique

L'axe U ($MA_AUTO_GET_TYPE=2) est connu dans le 1er et dans le 2ème canal et actuellement, c'est le canal 1 qui dispose du droit d'interpolation. Le cycle technologique suivant démarre dans le canal 2:

Code de programme	Commentaire
GET(U)	; Faire passer l'axe U dans ce canal
POS[U]=100	; L'axe U doit être déplacé à la position 100.

La ligne du déplacement de l'axe de commande POS[U] n'est exécutée qu'après que l'on a fait passer l'axe U dans le canal 2.

Procédure

L'axe demandé au moment de l'activation de l'action `GET(axe)` peut - en ce qui concerne le type d'axe - être lu pour une permutation d'axes avec la variable système ($AA_AXCHANGE_TYP[<axe>] :

- 0: Axe affecté au programme CN
- 1: Axe affecté à l'AP ou actif en tant qu'axe de commande ou axe pendulaire
- 2: un autre canal dispose du droit d'interpolation
- 3: L'axe est un axe banalisé
- 4: L'axe banalisé est contrôlé par l'AP
- 5: un autre canal dispose du droit d'interpolation, l'axe a été demandé pour le programme CN
- 6: un autre canal dispose du droit d'interpolation, l'axe a été demandé comme axe banalisé
- 7: Axe de l'AP ou actif en tant qu'axe de commande ou axe pendulaire, l'axe a été demandé pour le programme CN
- 8: Axe de l'AP ou actif en tant qu'axe de commande ou axe pendulaire, l'axe a été demandé comme axe banalisé

Conditions marginales

L'axe concerné doit être affecté au canal par le biais des paramètres machine.

Un axe exclusivement contrôlé par l'AP ne peut pas être affecté au programme CN.

Bibliographie :

/FB2/ Manuel de fonctions d'extension ; axes de positionnement (P2)

Demander l'axe depuis un autre canal avec l'action GET

Si, au moment de l'activation de l'action `GET`, un **autre canal dispose du droit d'écriture** (droit d'interpolation) pour l'axe ($AA_AXCHANGE_TYP[<axe>] == 2), l'axe est alors demandé par ce canal par le biais d'une permutation d'axes ($AA_AXCHANGE_TYP[<axe>]==6) et affecté dès que possible au canal qui a fait la demande.

Son état devient alors celui d'un axe banalisé ($AA_AXCHANGE_TYP[<axe>]==3).

Il ne s'opère pas de réorganisation dans le canal qui a fait la demande.

Affectation **en tant qu'axe de programme CN avec réorganisation:**

Si l'axe a déjà été demandé en tant qu'axe banalisé ($AA_AXCHANGE_TYP[<axe>]==6) au moment de l'activation de l'action `GET`, il est demandé pour le programme CN ($AA_AXCHANGE_TYP[<axe>]==5) et affecté dès que possible au programme CN du canal ($AA_AXCHANGE_TYP[<axe>]==0).

Axe déjà affecté au canal demandé

Affectation **en tant qu'axe de programme CN avec réorganisation:**

Si, au moment de l'activation, l'axe demandé est déjà affecté au canal qui fait la demande et qu'il se trouve à l'état d'axe banalisé - non contrôlé par l'AP - ($AA_AXCHANGE_TYP[<axe>]==3), il est alors affecté au programme CN ($AA_AXCHANGE_TYP[<axe>]==0).

L'axe à l'état d'axe banalisé est contrôlé par l'AP

Si l'axe à l'état d'axe banalisé est contrôlé par l'AP ($AA_AXCHANGE_TYP[<axe>]==4), il est demandé en tant qu'axe banalisé ($AA_AXCHANGE_TYP[<axe>]==8) et est bloqué ce faisant, en fonction du bit 0 au paramètre machine MD 10722: AXCHANGE_MASK, pour une permutation d'axes automatique entre canaux (bit 0 == 0). Ceci correspond à ($AA_AXCHANGE_STAT[<axe>] == 1).

L'axe est actif en tant qu'axe de commande banalisé ou axe pendulaire ou est affecté à l'AP.

Si l'axe est actif en tant qu'axe de commande ou axe pendulaire ou qu'il est affecté à l'AP pour le déplacement, axe AP == axe de positionnement piloté par l'AP, ($AA_AXCHANGE_TYP[<axe>]==1), il est demandé en tant qu'axe banalisé ($AA_AXCHANGE_TYP[<axe>]==8) et est bloqué ce faisant, en fonction du bit 0 au paramètre machine MD 10722: AXCHANGE_MASK, pour une permutation d'axes automatique entre canaux (bit 0 == 0). Ceci correspond à ($AA_AXCHANGE_STAT[<axe>] ==1).

Une nouvelle action GET demande alors l'axe pour le programme CN ($AA_AXCHANGE_TYP[<axe>] devient == 7).

L'axe est déjà affecté au programme CN

Si l'axe est déjà affecté au programme CN du canal ($AA_AXCHANGE_TYP[<axe>]==0) ou que cette affectation est demandée, par exemple permutation d'axes déclenchée par le programme CN ($AA_AXCHANGE_TYP[<axe>]==5 ou $AA_AXCHANGE_TYP[<axe>] == 7), il ne se produit pas de changement d'état.

10.4.16 Avance axiale (FA)

Fonction

L'avance axiale pour les axes de commande est à effet modal.

Syntaxe

```
AA[<axe>]=<valeur>
```

Exemple

Code de programme	Commentaire
`ID=1 EVERY $AA_IM[B]>75 DO POS[U]=100 FA[U]=990`	`; Définition d'une valeur d'avance fixe.`
	`; Former la valeur de l'avance à partir de variables d'exécution :`
`ID=1 EVERY $AA_IM[B]>75 DO POS[U]=100 FA[U]=$AA_VACTM[W]+100`	

10.4.17 Fins de course logiciels

Fonction

La limitation de la zone de travail programmée avec G25/G26 est prise en considération pour les axes de commande en fonction de la donnée de réglage $SA_WORKAREA_PLUS_ENABLE.

L'activation et la désactivation de la limitation de la zone de travail par le biais des fonctions G WALIMON/WALIMOF dans le programme pièce n'agissent pas sur les axes de commande.

10.4.18 Coordination d'axe

Fonction

Par principe, un axe est déplacé soit à partir du programme pièce, soit à partir d'une action synchrone comme axe de positionnement.

Mais si un même axe doit être déplacé de façon alternée à partir du programme pièce comme axe à interpolation ou axe de positionnement et à partir d'une action synchrone, alors un transfert coordonné a lieu entre les deux déplacements de l'axe.

Si un axe de commande est ensuite déplacé à partir du programme pièce, ceci exige une réorganisation du prétraitement. Cette réorganisation entraîne de son côté une interruption de l'exécution du programme pièce, comparable à un arrêt du prétraitement des blocs.

Exemple de déplacement de l'axe X alternativement à partir du programme pièce et à partir d'actions synchrones

```
Code de programme                          Commentaire
N10 G01 X100 Y200 F1000                    ;  Axe X programmé dans le programme pièce
…
N20 ID=1 WHEN $A_IN[1]==1 DO               ;  Démarrage du positionnement depuis l'action synchrone lorsque
POS[X]=150 FA[X]=200                          l'entrée numérique est présente
…
CANCEL(1)                                  ;  Désactivation de l'action synchrone
…
N100 G01 X240 Y200 F1000                   ;  X devient axe à interpolation ; avant le déplacement,
                                              apparition d'un temps d'attente dû au transfert d'axe
                                              si l'entrée numérique était 1 et que X a été positionné
                                              depuis l'action synchrone.
```

Exemple de modification de l'instruction de déplacement pour le même axe

```
Code de programme                                          Commentaire
ID=1 EVERY $A_IN[1]>=1 DO POS[V]=100 FA[V]=560             ;  Démarrage du positionnement depuis
                                                              l'action synchrone lorsque l'entrée
                                                              numérique >= 1
ID=2 EVERY $A_IN[2]>=1 DO POS[V]=$AA_IM[V] FA[V]=790       ;  L'axe est asservi, la 2e entrée est
                                                              mise à 1, ce qui signifie que la
                                                              position finale et l'avance
                                                              de l'axe V sont asservies en permanence
                                                              au cours du déplacement
                                                              lorsque deux actions synchrones sont
                                                              actives simultanément.
```

10.4.19 Préréglage des mémoires de valeurs réelles (PRESETON)

Fonction

La position courante de l'axe n'est pas modifiée lors de l'exécution de PRESETON (axe,valeur) ; une nouvelle valeur lui est affectée.

PRESETON peut figurer dans une action synchrone pour :

- des axes rotatifs à modulos qui ont été démarrés dans le programme pièce
- tous les axes de commande qui ont été démarrés à partir de l'action synchrone

Syntaxe

```
DO PRESETON(axe, valeur)
```

Signification

`DO PRESETON`	Prérégler les mémoires de valeurs réelles dans les actions synchrones
`Axe`	Axe, dont l'origine de la commande doit être modifiée
`Valeur`	Valeur par laquelle l'origine de la commande sera modifiée

Restrictions pour les axes

On ne peut pas faire `PRESETON` pour les axes qui participent à la transformation.

Le déplacement d'un même axe à partir du programme pièce ou d'une action synchrone ne peut se faire qu'avec un décalage de temps, par conséquent, lors de la programmation d'un axe à partir du programme pièce, des temps d'attente peuvent apparaître si cet axe était programmé précédemment dans une action synchrone.

Si le même axe est utilisé alternativement, un transfert coordonné a lieu entre les deux mouvements axiaux. L'exécution du programme pièce doit être interrompue dans ce but.

Exemple

Décaler l'origine de la commande d'un axe

```
Code de programme                                   Commentaire
WHEN $AA_IM[a] >= 89.5 DO PRESETON(a4,10.5)     ;  Décaler l'origine de la commande
                                                   de l'axe a de 10,5 unités de
                                                   longueur (pouce ou mm)
                                                   en sens positif
```

10.4.20 Déplacement de broches

Fonction

A partir d'actions synchrones, vous pouvez positionner des broches de façon totalement asynchrone avec le programme pièce. Ce type de programmation est recommandé pour des opérations cycliques ou pilotées par des événements.

Si des instructions concurrentes sont données pour une broche par différentes actions synchrones activées simultanément, ce sera la dernière instruction dans l'ordre chronologique qui sera prise en considération.

Exemple de démarrage/arrêt/positionnement d'une broche

Code de programme	Commentaire
`ID=1 EVERY $A_IN[1]==1 DO M3 S1000`	`; Réglage du sens de rotation et de la vitesse de rotation`
`ID=2 EVERY $A_IN[2]==1 DO SPOS=270`	`; Positionnement de la broche`

Exemple de réglage du sens et de la vitesse de rotation/positionnement de la broche

Code de programme	Commentaire
`ID=1 EVERY $A_IN[1]==1 DO M3 S300`	`; Réglage du sens de rotation et de la vitesse de rotation`
`ID=2 EVERY $A_IN[2]==1 DO M4 S500`	`; Définition d'un nouveau sens de rotation et d'une nouvelle vitesse de rotation`
`ID=3 EVERY $A_IN[3]==1 DO S1000`	`; Définition d'une nouvelle vitesse de rotation`
`ID=4 EVERY ($A_IN[4]==1) AND ($A_IN[1]==0) DO SPOS=0`	`; Positionnement de la broche`

10.4.21 Déplacements conjugués (TRAILON, TRAILOF)

Fonction

Quand on active le couplage à partir de l'action synchrone, l'axe pilote peut être en cours de déplacement. Dans ce cas, l'axe asservi subit une accélération pour atteindre la vitesse de consigne. La position de l'axe pilote au moment de la synchronisation des vitesses constitue la position de départ des déplacements conjugués. Les déplacements conjugués sont décrits dans le chapitre "Mode de déplacement".

Syntaxe

Activer les déplacements conjugués

```
DO TRAILON(axe asservi, axe pilote, facteur de couplage)
```

Désactiver les déplacements conjugués

```
DO TRAILOF(axe asservi, axe pilote, axe pilote 2)
```

Signification

Activer les déplacements conjugués synchrones :

`... DO TRAILON(AA, AP, Fc)`

avec :
AA : axe asservi
AP : axe pilote
Fc : facteur de couplage

Désactiver les déplacements conjugués synchrones :

`... DO TRAILOF(AA, AP, AP2)`

avec :
AA: axe asservi
AP : axe pilote, en option
AP2 : axe pilote 2, optionnel

`... DO TRAILOF(AA)`

Tous les couplages sur l'axe asservi sont désactivés.

Exemple

Code de programme	Commentaire
`$A_IN[1]==0 DO TRAILON(Y,V,1)`	`;` Activation du 1e groupe d'axes à déplacements conjugués, dès que l'entrée TOR est à 1
`$A_IN[2]==0 DO TRAILON(Z,W,-1)`	`;` Activation du 2e groupe d'axes à déplacements conjugués
`G0 Z10`	`;` Déplacement des axes Z et W en sens opposé
`G0 Y20`	`;` Déplacement des axes Y et V dans le même sens
`...`	
`G1 Y22 V25`	`;` Superposition d'un déplacement dépendant et d'un déplacement indépendant de l'axe conjugué "V"
`...`	
`TRAILOF(Y,V)`	`;` Désactivation du 1e groupe d'axes à déplacements conjugués
`TRAILOF(Z,W)`	`;` Désactivation du 2e groupe d'axes à déplacements conjugués

Exemple de conflit évité avec TRAILOF

Pour valider l'accès à l'axe couplé en tant qu'axe de canal, il faut auparavant appeler la fonction `TRAILOF`. Il faut veiller à ce que `TRAILOF` ait été exécuté avant que le canal ne demande l'axe en question. Ce n'est pas le cas dans l'exemple suivant.

```
…
N50 WHEN TRUE DO TRAILOF(Y,X)
N60 Y100
…
```

Ici, l'axe n'est pas libéré à temps, car l'action synchrone avec `TRAILOF` à effet non modal est activée en synchronisme avec `N60` , voir chapitre "Action synchrone au déplacement - Structure, généralités".
Afin d'éviter les situations conflictuelles,

il doit être procédé de la manière suivante

```
…
N50 WHEN TRUE DO TRAILOF(Y,X)
N55 WAITP(Y)
N60 Y100
```

10.4.22 Couplage par valeur pilote (LEADON, LEADOF)

Remarque

Cette fonction n'est pas disponible pour SINUMERIK 828D.

Fonction

Le couplage de deux axes par valeur pilote est programmable sans restriction dans des actions synchrones. La modification d'une table de courbes dans un couplage existant sans une nouvelle synchronisation préalable n'est possible que dans les actions synchrones (facultatif).

Syntaxe

Activer le couplage par valeur pilote

```
DO LEADON(Axe asservi, axe pilote, n°_table de courbes, OVW)
```

Désactiver le couplage par valeur pilote

```
DO TRAILOF(axe asservi, axe pilote, axe pilote 2)
```

Signification

```
Activer le couplage par valeur pilote :
...DO LEADON(AA, AP, N°, OVW)
```

avec :
AA: axe asservi
AP : axe pilote
N° : numéro de la table de courbes enregistrée
OVW : permettre l'écrasement d'un couplage existant par une table de courbes modifiée

```
Désactiver le couplage par valeur pilote :
...DO LEADOF(AA, AP)
```

avec :
AA : axe asservi
AP : axe pilote, en option

```
... DO LEADOF(AA)
```

Forme abrégée sans indication de l'axe pilote

Valider l'accès d'actions synchrones avec la fonction RELEASE

Pour valider l'accès d'actions synchrones à l'axe asservi à coupler, il faut auparavant appeler la fonction RELEASE pour cet axe.

Exemple :

```
RELEASE (XCAN)
ID=1 every SR1==1 to LEADON(AXEC,XCAN,1)
```

OVW=0 (valeur par défaut)

Il ne peut être attribuée aucune table de courbes à un couplage existant sans nouvelle synchronisation. Une modification de la table de courbes requiert la désactivation préalable du couplage existant et une réactivation avec le numéro de table de courbes modifié. Ceci entraîne une nouvelle synchronisation du couplage.

Modification de la table de courbes pour un couplage existant avec OVW=1

Avec `OVW=1` , une nouvelle table de courbes peut être attribuée à un couplage existant. Une nouvelle synchronisation n'est pas requise. L'axe asservi tente au plus vite de suivre la valeur de position indiquée par la nouvelle table de courbes.

Exemple de tronçonnage au vol

Un tube, qui traverse à vitesse continue la zone de travail d'une scie, doit être découpé en tronçons de même longueur.

Axe X : Axe dans lequel se déplace le tube. SCP
Axe X1 : axe machine du tube, SCM
Axe Y : axe dans lequel la scie "se déplace avec" le tube

On suppose que le déplacement de la scie et sa commande sont pilotés par l'AP. Pour vérifier le synchronisme entre le tube et la scie, on peut exploiter les signaux de l'interface AP.

Actions
activer le couplage, `LEADON`
désactiver le couplage, `LEADOF`
prérégler les mémoires de valeurs réelles, `PRESETON`

Code de programme	**Commentaire**
N100 R3=1500	; Longueur d'une partie à tronçonner
N200 R2=100000 R13=R2/300	
N300 R4=100000	
N400 R6=30	; Position de départ de l'axe Y
N500 R1=1	; Condition de départ pour l'axe de la bande
N600 LEADOF(Y,X)	; Suppression d'un éventuel couplage existant
N700 CTABDEF(Y,X,1,0)	; Définition de table
N800 X=30 Y=30	; Paire de valeurs
N900 X=R13 Y=R13	
N1000 X=2*R13 Y=30	
N1100 CTABEND	; Fin de la définition de la table
N1200 PRESETON(X1,0)	; PRESET au début
N1300 Y=R6 G0	; Position de départ de l'axe Y, l'axe est linéaire.
N1400 ID=1 WHENEVER $AA_IW[X]>$R3 DO PRESETON(X1,0)	; PRESET après longueur R3, nouveau départ après tronçonnage
N1500 RELEASE(Y)	
N1800 ID=6 EVERY $AA_IM[X]<10 DO LEADON(Y,X,1)	; Couplage d'Y à X par la table 1 lorsque X < 10
N1900 ID=10 EVERY $AA_IM[X]>$R3-30 DO LEADOF(Y,X)	; Séparation du couplage à > 30 avant la longueur de tronçonnage parcourue
N2000 WAITP(X)	
N2100 ID=7 WHEN $R1==1 DO MOV[X]=1 AA[X]=$R4	; Mise en mouvement continue de l'axe du tube
N2200 M30	

10.4.23 Mesure (MEAWA, MEAC)

Fonction

En comparaison avec l'utilisation dans des blocs de déplacement du programme pièce, la fonction de mesure peut être activée et désactivée à volonté à partir d'actions synchrones.

Pour de plus amples informations sur la fonction de mesure, voir les instructions de déplacement spéciales "Fonction de mesure étendue"

Syntaxe

Mesure axiale sans effacement de la distance restant à parcourir

```
MEAWA[axe]=(mode, événement déclencheur_1, ..._4)
```

Mesure sans effacement de la distance restant à parcourir

```
MEAWA[axe]=(mode, événement déclencheur_1, ..._4)
```

Signification

Code de programme	**Commentaire**	
DO MEAWA	; Activer la mesure axiale	
DO MEAC	; Activer la mesure continue	
Axe	; Nom de l'axe mesuré	
Mode	; Indication de **la décade**décimale 0 : système de mesure actif	Indication de **la**décade unitaire 0 : annulation du contrat de mesure
	Nombre de systèmes de mesure (en fonction du mode) 1 : 1. système de mesure 2 : 2. système de mesure 3 : les deux systèmes de mesure	jusqu'à 4 événements déclencheurs activables 1 : simultanément 2 : successivement 3 : identique à 2 sans la surveillance de l'événement déclencheur1 au départ
Evénement déclencheur_1 à _4	; : front montant, palpeur 1 -1 : front descendant, palpeur 1 facultatif 2 : front montant, palpeur 2 facultatif -2 : front descendant, palpeur 2 facultatif	
Mémoire de mesure	; Numéro de la mémoire à file d'attente FIFO	

10.4.24 Initialisation de variables de tableau (SET, REP)

Fonction

Dans les actions synchrones, des variables de tableau peuvent être initialisées ou être décrites avec certaines valeurs.

Remarque

Seules les variables accessibles en écriture dans des actions synchrones sont possibles. Des paramètres machine ne peuvent être initialisés ainsi. Des variables d'axe ne peuvent pas être indiquées avec la valeur NO_AXIS.

Syntaxe

```
DO TABLEAU[n,m]=SET(<valeur1>,<valeur2>,...)
DO TABLEAU[n,m]=REP(<valeur>)
```

Signification

`TABLEAU[n,m]`	Indices de tableau programmés L'initialisation débute par les indices de tableau programmés. En cas de tableaux à deux dimensions, le 2e indice est incrémenté en premier. En cas d'indices d'axes, celui-ci n'est pas effectué.
`SET(<valeur1>,<valeur2>,...)`	Initialisation avec listes de valeurs Le tableau est décrit par les indices de tableau programmés avec les paramètres de `SET`. Il y a autant d'éléments de tableau attribués que de valeurs programmées. Si le nombre de valeurs programmées est supérieur à celui des éléments de tableau résiduels disponibles, une alarme est déclenchée.
`REP(<valeur>)`	Initialisation avec des valeurs identiques Le tableau est répété par les indices de tableau programmés jusqu''à la fin du tableau avec le paramètre (`<valeur>`) de `REP`.

Exemple

Code de programme	**Commentaire**
`WHEN TRUE DO SYG_IS[0]=REP(0)`	`; Résultat :`

`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`

10.4.25 Activer/effacer des marques d'attente (SETM, CLEARM)

Fonction

Dans des actions synchrones, vous pouvez activer ou effacer des marques d'attente, par exemple pour coordonner des canaux entre eux.

Syntaxe

```
DO SETM(<numéro de marque>)
DO CLEARM(<numéro de marque>)
```

Signification

`SETM`	Instruction de définition de la marque d'attente pour le canal L'instruction `SETM` peut être écrite dans le programme pièce et dans la partie action d'une action synchrone. Elle active la marque (`<numéro de marque>`) pour le canal auquel elle est affectée.
`CLEARM`	Instruction de suppression de la marque d'attente pour le canal L'instruction `CLEARM` peut être écrite dans le programme pièce et dans la partie action d'une action synchrone. Elle supprime la marque (`<numéro de marque>`) pour le canal auquel elle est affectée.
`<Numéro de marque>`	Marque d'attente

10.4.26 Réaction aux erreurs (SETAL)

Fonction

Avec les actions synchrones, il est possible de programmer des réactions aux erreurs avec interrogation des variables d'état et déclenchement d'actions correspondantes.

En réponse à des états d'erreur, les réactions suivantes sont possibles :

- Arrêt de l'axe (correction=0)
- Activation d'une alarme

  Avec SETAL, on peut activer des alarmes cycliques à partir d'actions synchrones.
- mettre sortie à 1
- Toutes les actions possibles dans des actions synchrones

Syntaxe

Activation d'une alarme cyclique :

```
DO SETAL(<numéro d'alarme>)
```

Signification

SETAL	Instruction d'activation d'une alarme cyclique
<Numéro d'alarme>	Numéro de l'alarme Zone d'alarme cyclique pour l'utilisateur : 65000 à 69999

Exemple

Code de programme	Commentaire
`ID=67 WHENEVER ($AA_IM[X1]-$AA_IM[X2])<4.567 DO $AA_OVR[X2]=0`	`; Si la distance de sécurité entre les axes X1 et X2 est trop faible, arrêter axe X2.`
`ID=67 WHENEVER ($AA_IM[X1]-$AA_IM[X2])<4.567 DO SETAL(65000)`	`; Si la distance de sécurité entre les axes X1 et X2 est trop faible, déclencher l'alarme 65000.`

10.4.27 Accostage d'une butée (FXS, FXST, FXSW, FOCON, FOCOF)

Fonction

Les instructions de la fonction "Accostage d'une butée" sont programmées avec les instructions de programme pièce `FXS`, `FXST` et `FXSW` dans les actions synchrones/cycles technologiques.

L'activation peut se faire sans déplacement, la limitation du couple agit immédiatement. Dès que l'axe est déplacé suivant les consignes, la surveillance de la butée est active.

Déplacement avec réduction du couple/force (FOC)

La fonction permet de modifier le couple/force à tout moment par l'intermédiaire d'actions synchrones et peut être activée avec un effet modal ou non modal.

Syntaxe

```
FXS[<axe>]
FXST[<axe>]
FXSW[<axe>]
FOCON[<axe>]
FOCOF[<axe>]
```

Signification

`FXS`	Activation uniquement dans les systèmes à entraînements numériques (VSA, HSA, HLA)
`FXST`	Modification du couple de blocage FXST
`FXSW`	Modification de la fenêtre de surveillance FXSW
`FOCON`	Activation de la limitation de couple/force à effet modal
`FOCOF`	Désactivation de la limitation de couple/force
<Axe>	Descripteur d'axe Sont autorisés : • Descripteur d'axe géométrique • Descripteur d'axe de canal • Descripteur d'axe machine

Remarque

L'activation ne doit avoir lieu qu'une fois.

Exemples

Exemple 1 : Accostage d'une butée (FXS) déclenché par une action synchrone

```
Code de programme                                                    Commentaire
Axe Y                                                                ; Actions synchrones statiques
Activation :
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
                                                                     ; $R1=1 active FXS pour l'axe Y, réduit le
                                                                       couple effectif à 10% et démarre un
                                                                       déplacement en direction de la butée.
N11 IDS=2 WHENEVER ($AA_FXS[Y]==4) DO FXST[Y]=30                      ; Dès que la butée est détectée
                                                                       ($AA_FXS[Y]==4), le couple augmente à
                                                                       30%.
N12 IDS=3 WHENEVER ($AA_FXS[Y]==1) DO FXST[Y]=$R0                    ; Dès que la butée est atteinte, la
                                                                       commande du couple dépend de R0.
N13 IDS=4 WHENEVER (($R3==1) AND $AA_FXS[Y]==1)) DO FXS[Y]=0 FA[Y]=1000 POS[Y]=0
                                                                     ; Désactivation en fonction de R3 et
                                                                       retour.
N20 FXS[Y]=0 G0 G90 X0 Y0                                            ; Exécution normale du programme :
N30 RELEASE(Y)                                                       ; Déblocage de l'axe Y pour le déplacement
                                                                       de l'action synchrone.
N40 G1 F1000 X100                                                    ; Déplacement d'un autre axe :
N50 ...
N60 GET(Y)                                                           ; Réintégrer l'axe Y dans le groupe
                                                                       d'interpolation
```

Exemple 2 : Activation de la limitation de couple/force (FOC)

```
Code de programme                  Commentaire
N10 FOCON[X]                       ; Activation modale de la limitation.
N20 X100 Y200 FXST[X]=15           ; Déplacement de X avec un couple réduit (15%)
N30 FXST[X]=75 X20                 ; Augmentation du couple à 75%, déplacement de X
                                     avec ce couple limité.
N40 FOCOF[X]                       ; Désactivation de la limitation de couple.
```

Autres informations

Activation multiple

Si à la suite d'une erreur de programmation, la fonction est de nouveau appelée après son activation (`FXS[<axe>]=1`), l'alarme suivante est générée :

Alarme 20092 "Accostage d'une butée encore active"

Une programmation qui interroge soit $AA_FXS[ ] ou un mémento spécifique (R1 dans ce cas) dans la condition évite une activation multiple de la fonction "Fragment de programme pièce" :

```
Code de programme
N10 R1=0
N20 IDS=1 WHENEVER ($R1==0 AND
$AA_IW[AX3] > 7) DO R1=1 FXST[AX1]=12
```

Actions synchrones non modales

La programmation d'une action synchrone non modale permet d'activer l'accostage d'une butée pendant un déplacement d'axe.

Exemple :

Code de programme	Commentaire
N10 G0 G90 X0 Y0	
N20 WHEN $AA_IW[X] > 17 DO FXS[X]=1	; Activation de FXS si X atteint une position supérieure à 17mm.
N30 G1 F200 X100 Y110	

Actions synchrones statiques et non modales

Dans les actions synchrones statiques à effet non modal, il est possible d'utiliser les mêmes instructions `FXS`, `FXST` et `FXSW` que dans l'exécution normale du programme pièce. Les valeurs qui seront attribuées peuvent résulter d'un calcul.

10.4.28 Détermination de l'angle de la tangente à la trajectoire dans les actions synchrones

Fonction

La variable système lisible dans les actions synchrones $AC_TANEB (**T**angent **AN**gel at **E**nd of **B**lock) détermine l'angle entre tangente à la trajectoire au point final du bloc courant et la tangente à la trajectoire au point de départ du bloc suivant programmé.

Paramètres

L'angle des tangentes est indiqué en valeur positive entre 0,0 et 180,0 degrés. S'il n'existe aucun bloc suivant, l'angle indiqué est
-180,0 degrés.

La variable système `$AC_TANEB` ne doit pas être lue pour des blocs créés par le système (blocs intermédiaires). La variable système `$AC_BLOCKTYPE`permet d'identifier qu'il s'agit d'un bloc programmé (bloc principal).

Exemple

```
ID=2 EVERY $AC_BLOCKTYPE==0 DO $SR1 = $AC_TANEB
```

10.4.29 Détermination de la correction de vitesse courante

Fonction

La correction de vitesse courante

(Composante CN) peut être lue et écrite avec les variables système :

`$AA_OVR` Correction axiale

`$AC_OVR` Correction d'avance tangentielle

dans des actions synchrones.

La correction de vitesse déterminée par l'AP pour les actions synchrones dans les variables système:

`$AA_PLC_OVR` Correction axiale

`$AC_PLC_OVR` Correction d'avance tangentielle

peut être accédée en lecture.

La correction résultante

pour les actions synchrones dans les variables système :

`$AA_TOTAL_OVR` Correction axiale

`$AC_TOTAL_OVR` Correction d'avance tangentielle

peut être accédée en lecture.

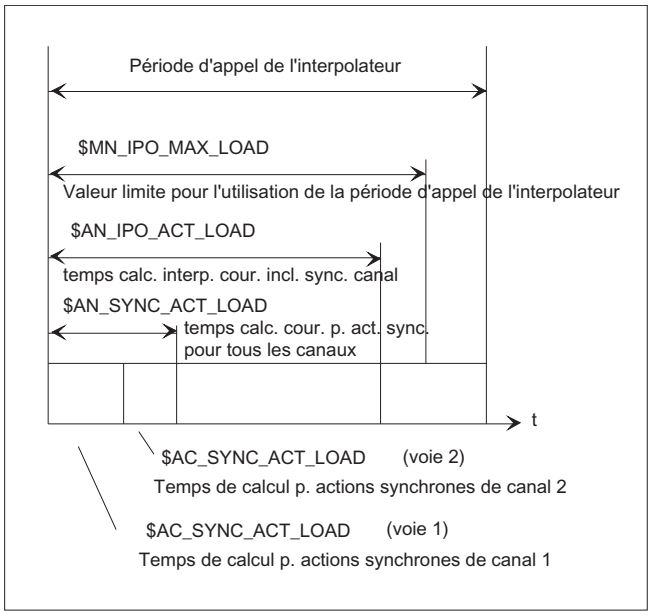
La correction résultante se calcule comme suit :

`$AA_OVR * $AA_PLC_OVR` ou
`$AC_OVR * $AC_PLC_OVR`

10.4.30 Traitement de l'utilisation par rapport à la durée des actions synchrones

Fonction

Dans une période d'appel de l'interpolateur, les actions synchrones doivent être interprétées pendant que les déplacements, et autres, sont calculés par la CN. Les variables système décrites ci-après permettent aux actions synchrones d'accéder à leurs composantes de temps courantes dans la période d'appel de l'interpolateur et de connaître le temps de calcul du régulateur de position.



Signification

Les valeurs des variables sont uniquement valides, lorsque le paramètres machine `$MN_IPO_MAX_LOAD` est supérieur à 0. Autrement, les variables indiquent toujours, aussi bien pour SINUMERIK powerline que pour les systèmes solution line, le temps de calcul net pour lequel les interruptions créées par IHM ne sont plus prises en compte. Le temps de calcul net est défini à partir :

- de la durée des actions synchrones,
- du temps d'asservissement de position et
- du temps de calcul restant de l'interpolateur sans interruptions conditionnées par IHM

Les variables comprennent toujours les valeurs de la période d'appel de l'interpolateur **précédente**.

`$AN_IPO_ACT_LOAD`	Temps de calcul courant de l'interpolateur (incluant les actions synchrones de tous les canaux)
`$AN_IPO_MAX_LOAD`	Temps de calcul le plus long de l'interpolateur (incluant les actions synchrones de tous les canaux)
`$AN_IPO_MIN_LOAD`	Temps de calcul le plus court de l'interpolateur (incluant les actions synchrones de tous les canaux)
`$AN_IPO_LOAD_PERCENT`	Temps de calcul courant de l'interpolateur par rapport à la période d'appel de l'interpolateur (%).
`$AN_SYNC_ACT_LOAD`	Temps de calcul courant des actions synchrones pour tous les canaux
`$AN_SYNC_MAX_LOAD`	Temps de calcul le plus long des actions synchrones pour tous les canaux
`$AN_SYNC_TO_IPO`	Pourcentage de l'ensemble des actions synchrones dans le temps de calcul total de l'interpolateur (pour tous les canaux)
`$AC_SYNC_ACT_LOAD`	Temps de calcul courant pour les actions synchrones dans le canal
`$AC_SYNC_MAX_LOAD`	Temps de calcul le plus long pour les actions synchrones dans le canal
`$AC_SYNC_AVERAGE_LOAD`	Temps de calcul moyen pour les actions synchrones dans le canal
`$AN_SERVO_ACT_LOAD`	Temps de calcul courant du régulateur de position
`$AN_SERVO_MAX_LOAD`	Temps de calcul le plus long du régulateur de position
`$AN_SERVO_MIN_LOAD`	Temps de calcul le plus court du régulateur de position

Variables des informations sur la surcharge :

Le PM `$MN_IPO_MAX_LOAD` permet de définir à partir de quel temps de calcul net de l'interpolateur
(en % de la période d'appel de l'interpolateur)
la variable système `$AN_IPO_LOAD_LIMIT` aura la valeur TRUE. Si la charge courante passe sous cette limite, la variable est à nouveau mise sur FALSE. Si le PM est 0, l'ensemble de la fonction de diagnostic est désactivé.

Par le biais de l'évaluation de `$AN_IPO_LOAD_LIMIT` , l'utilisateur peut définir sa propre stratégie pour éviter un dépassement de niveau.

10.5 Cycles technologiques

Fonction

Dans des actions synchrones, vous pouvez aussi appeler, comme actions, des programmes, qui toutefois doivent contenir uniquement des fonctions admises elles aussi comme actions dans des actions synchrones. De tels programmes sont appelés des cycles technologiques.

Les cycles technologiques sont rangés comme sous-programmes dans la commande.

Plusieurs cycles technologiques ou actions peuvent être exécutés en parallèle dans un canal.

Programmation

Les règles suivantes sont valables pour la programmation de cycles technologiques :

- La fin du programme se programme avec M02/M17/M30/RET.
- Au sein d'un niveau de programme, toutes les actions indiquées dans ICYCOF peuvent être traitées sans cycles d'attente dans une période.
- Il est possible d'interroger jusqu'à 8 cycles technologiques à la suite par action synchrone.
- Les cycles technologiques sont également possibles dans des actions synchrones non modales.
- Il est possible de programmer aussi bien des structures de contrôle IF que des instructions de saut GOTO, GOTOF et GOTOB.
- Règles valables pour les blocs avec DEF et DEFINE :
  - Les instructions DEF et DEFINE sont ignorées dans les cycles technologiques.
  - Elles entraînent un message d'alarme en cas de syntaxe incomplète ou incorrecte.
  - Elles peuvent, sans être créées elles-mêmes, être ignorées sans message d'alarme.
  - Elles sont entièrement prises en compte avec des affectations de valeur comme cycle de programme pièce.

Transfert de paramètres

Un transfert de paramètres sur des cycles technologiques est possible. Sont aussi bien pris en compte des types de données simples transmis comme des paramètres formels "Call by Value" que des paramétrages par défaut qui deviennent valables à l'appel de cycles technologiques. Ce sont :

- Valeurs par défaut programmées lorsque aucun paramètre de transmission n'est programmé.
- Affectation de valeurs initiales aux paramètres par défaut.
- Transmettre des paramètres réels non initialisés avec une valeur par défaut.

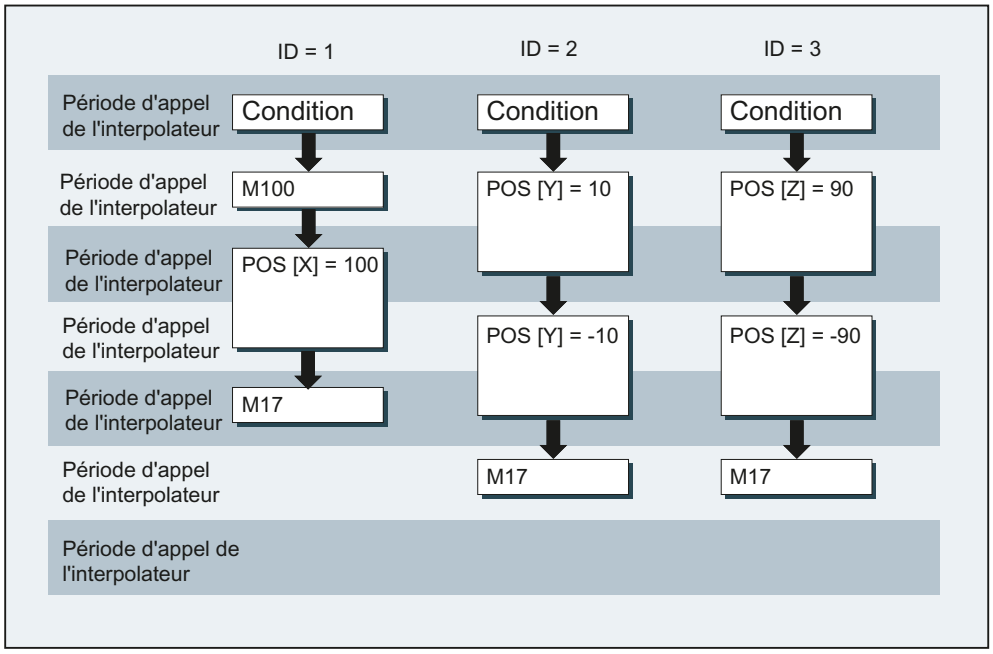
Procédure

Les cycles technologiques sont démarrés, dès que leurs conditions sont remplies. Chaque ligne d'un cycle technologique est exécutée dans une période d'appel de l'interpolateur séparée. Dans le cas d'axes de positionnement, plusieurs périodes d'appel de l'interpolateur sont nécessaires pour leur exécution. D'autres fonctions sont exécutées en une seule période. Dans le cycle technologique, l'exécution des blocs est séquentielle.

Si des actions qui s'excluent réciproquement sont appelées dans une même période d'appel de l'interpolateur, l'action active sera celle qui est appelée par l'action synchrone avec le numéro ID le plus élevé.

Exemples

Exemple 1 : démarrage de programmes d'axe par activation d'entrées numériques



Programme principal :

```
Code de programme                    Commentaire
ID=1 EVERY $A_IN[1]==1 DO AXE_X  ; Démarrage du programme d'axe AXE_X lorsque l'entrée 1 est mise à 1.
ID=2 EVERY $A_IN[2]==1 DO AXE_Y  ; Démarrage du programme d'axe AXE_Y lorsque l'entrée 2 est mise à 1.
ID=3 EVERY $A_IN[3]==1 DO AXE_Z  ; Démarrage du programme d'axe AXE_Z lorsque l'entrée 3 est mise à 1.
M30
```

Programme d'axe AXE_X :

```
Code de programme
M100
POS[X]=100 FA[X]=300
M17
```

Programme d'axe AXE_Y :

```
Code de programme
POS[Y]=10 FA[Y]=200
POS[Y]=-10
M17
```

Programme d'axe AXE_Z :

```
Code de programme
POS[Z]=90 FA[Z]=250
POS[Z]=-90
M17
```

Exemple 2 : Différentes séquences de programme dans le cycle technologique

```
Code de programme
PROC CYCLE
N10 DEF REAL VALEUR=12.3
N15 DEFINE ABC AS G01
```

Les deux blocs sont ignorés sans alarme ni création de variables ou de macros.

```
Code de programme
PROC CYCLE
N10 DEF REAL
N15 DEFINE ABC G01
```

Les deux blocs entraînent en outre une alarme CN parce que la syntaxe n'a pas été écrite correctement.

```
Code de programme
PROC CYCLE
N10 DEF AXIS AXE1=XX2
```

Lorsque l'axe XX2 n'est pas connu, l'alarme 12080 est émise. Sinon, le bloc est ignoré sans alarme et sans création de variable.

```
Code de programme
PROC CYCLE
N10 DEF AXIS AXE1
N15 G01 X100 F1000
N20 DEF REAL VALEUR1
```

Le bloc `N20` conduit toujours à l'alarme 14500, car aucune instruction de définition ne doit succéder à la première instruction du programme.

10.5.1 Variable de contexte ($P_TECCYCLE)

Fonction

La variable $P_TECCYCLE permet de subdiviser les programmes entre programmes d'actions synchrones et programmes d'exécution. Il est ainsi possible de traiter des blocs ou des séquences de programme à syntaxe correcte, également comme cycle de programme pièce.

Interpréter une variable de contexte

La variable système $P_TECCYCLE permet de piloter des interprétations de programme pièce spécifiques au contexte dans des cycles technologiques :

```
IF $P_TECCYCLE==TRUE
...                          ; Séquence de programme pour cycle technologique dans
                               l'action synchrone.
ELSE
...                          ; Séquence de programme pour cycle de programme pièce.
ENDIF
```

Remarque

Un bloc avec une syntaxe de programme erronée ou non autorisée ainsi que des affectations de valeur inconnues entraînent également dans le cycle de programme pièce un message d'alarme.

Exemple

Séquence de programme avec interrogation de $P_TECCYCLE dans le cycle technologique :

```
Code de programme            Commentaire
PROC CYCLE
N10 DEF REAL VALEUR1         ; Est ignoré dans le cycle technologique.
N15 G01 X100 F1000
N20 IF $P_TECCYCLE==TRUE
...                          ; Séquence de programme pour cycle technologique (sans variable
                               VALEUR1).
N30 ELSE
...                          ; Séquence de programme pour cycle de programme pièce (la variable
                               VALEUR1 existe).
N40 ENDIF
```

10.5.2 Paramètres Call-by-Value

Fonction

Les cycles technologiques peuvent être définis avec des paramètres Call-by-Value. Des types de données simples comme INT, REAL, CHAR, STRING, AXIS et BOOL sont possibles comme paramètres.

Remarque

Les paramètres formels qui transmettent des Call-by-Value ne peuvent pas être des tableaux.

Les paramètres effectifs peuvent également être des paramètres par défaut (voir "Initialisation de paramètres par défaut (Page 630)").

Syntaxe

```
ID=1 WHEN $AA_IW[X]>50 DO TEC(IVAL,RVAL,,SVAL,AVAL)
```

Dans le cas de paramètres effectifs non initialisés, une valeur par défaut est transmise :

```
ID=1 WHEN $AA_IW[X]>50 DO TEC(IVAL,RVAL,,SYG_SS[0],AVAL)
```

10.5.3 Initialisation de paramètres par défaut

Fonction

Des paramètres par défaut peuvent également être pourvus d'une valeur initiale dans l'instruction PROC.

Syntaxe

Dans le cycle technologique, attribuer des paramètres par défaut :

```
PROC TEC (INT IVAL=1, REAL RVAL=1.0, CHAR CVAL='A', STRING[10] SVAL="ABC", AXIS
AVAL=X, BOOL BVAL=TRUE)
```

Lorsque un paramètre réel se compose d'un paramètre par défaut, la valeur initiale est transmise par l'instruction PROC. Cela est aussi bien valable dans le programme pièce que dans les actions synchrones.

Exemple

Code de programme	Commentaire
`TEC (IVAL, RVAL, SVAL, AVAL)`	`;` Pour CVAL et BVAL, la valeur initiale est valable.

10.5.4 Pilotage du traitement de cycles technologiques (ICYCOF, ICYCON)

Fonction

Les instructions ICYCOF et ICYCON servent au pilotage du traitement temporel de cycles technologiques.

Avec ICYCOF, tous les blocs d'un cycle technologique sont traités dans une seule période d'appel de l'interpolateur. Toutes les actions nécessitant plusieurs périodes entraînent des processus de traitement parallèles avec ICYCOF.

Application

Avec ICYCON, plusieurs déplacements d'axe de commande peuvent entraîner la temporisation du traitement d'un cycle technologique. Si cela n'est pas souhaité, ICYCOF permet de traiter toutes les actions sans temps d'attente dans une période d'appel de l'interpolateur.

Syntaxe

Pour le traitement cyclique de cycles technologiques, on a :

`ICYCON` Chaque bloc d'un cycle technologique est exécuté dans une période d'appel de l'interpolateur séparée après ICYCON.

`ICYCOF` Tous les blocs suivants d'un cycle technologique sont exécutés dans une période d'appel de l'interpolateur après ICYCOF.

Remarque

Les deux instructions ICYCON et ICYCOF ne sont valables que dans un niveau de programme. Dans le programme pièce, les deux commandes sont simplement ignorées sans réaction.

Exemple de mode de traitement ICYCOF

Code de programme	Commentaire
Période d'appel de l'interpolateur	; 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 Cascadages de cycles technologiques

Fonction

Il est possible d'exécuter jusqu'à 8 cycles technologiques activés en série. Plusieurs cycles technologiques sont donc programmables dans une action synchrone.

Syntaxe

```
ID=1 WHEN $AA_IW[X]>50 DO TEC1($R1) TEC2 TEC3(X)
```

Ordre d'exécution

Les cycles technologiques sont traités à la suite (en cascade) de la gauche vers la droite selon la programmation indiquée précédemment. Lorsque un cycle doit être traité en mode ICYCON, il temporise tous les traitements ultérieurs. Une alarme qui survient interrompt toutes les actions suivantes.

10.5.6 Cycles technologiques en actions synchrones non modales

Fonction

Les cycles technologiques sont également possibles dans des actions synchrones non modales.

Si le temps d'exécution d'un cycle technologique est supérieur au temps de traitement du bloc correspondant, le cycle technologique est interrompu au changement de bloc.

Remarque

Un cycle technologique n'empêche pas le changement de bloc.

10.5.7 Structures de contrôle (IF)

Fonction

Pour des ramifications dans l'ordre de déroulement de cycles technologiques, les structures de contrôle IF peuvent être employées dans des actions synchrones.

Syntaxe

```
IF <condition>
$R1=1
[ELSE] en option
$R1=0
ENDIF
```

10.5.8 Instructions de saut (GOTO, GOTOF, GOTOB)

Fonction

Dans des cycles technologiques, les instructions de saut GOTO, GOTOF, GOTOB sont possibles. les étiquettes indiquées doivent figurer dans le sous-programme afin qu'aucune alarme ne soit générée.

Remarque

Les étiquettes et les numéros de bloc peuvent seulement être des constantes.

Syntaxe

Sauts inconditionnels

`GOTO` Etiquette, numéro de bloc

`GOTOF` Etiquette, numéro de bloc

`GOTOB` Etiquette, numéro de bloc

Instructions de saut et destinations de saut

`GOTO`	Saute tout d'abord en aval puis en amont
`GOTOF`	Saute en aval
`GOTOB`	Saute en amont
`Etiquette :`	Marque de saut
`numéro de bloc`	Destination de saut pour ce bloc
`N100`	Le numéro de bloc est un bloc secondaire
`:100`	Le numéro de bloc est un bloc principal

10.5.9 Bloquer, débloquer, remettre à zéro (LOCK, UNLOCK, RESET)

Fonction

Le déroulement d'un cycle technologique peut être bloqué, libéré ou remis à zéro par une autre action synchrone modale.

Syntaxe

```
LOCK(<n1>,<n2>,...)
UNLOCK(<n1>,<n2>,...)
RESET(<n1>,<n2>,...)
```

Signification

`LOCK`	Instruction de blocage des actions synchrones L'action active est interrompue.
`UNLOCK`	Instruction de déblocage des actions synchrones
`RESET`	Instruction de remise à zéro des cycles technologiques
`<n1>,<n2>,...`	Numéros d'identification des actions synchrones ou des cycles technologiques devant être bloqués, débloqués ou remis à zéro.

Verrouillage d'actions synchrones

Les actions synchrones modales avec les numéros ID **<n> = 1 ... 64** peuvent être verrouillées par l'AP. Dans ce cas, la condition correspondante n'est plus traitée et l'exécution de la fonction dans NCK est bloquée.

Sur un signal de l'interface AP, toutes les actions synchrones peuvent être bloquées en même temps.

Remarque

Une action synchrone programmée est active par défaut et peut être protégée contre un écrasement des données/un blocage par le biais d'un paramètre machine.

L'utilisateur ne doit pas pouvoir modifier les actions synchrones qui ont été définies par le constructeur de la machine.

Exemples

Exemple 1 : Bloquer les actions synchrones (LOCK)

Code de programme

```
N100 ID=1 WHENEVER $A_IN[1]==1 DO M130
...
N200 ID=2 WHENEVER $A_IN[2]==1 DO LOCK(1)
```

Exemple 2 : Débloquer les actions synchrones (UNLOCK)

Code de programme

```
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)
```

Exemple 3 : Interrompre un cycle technologique (RESET)

Code de programme

```
N100 ID=1 WHENEVER $A_IN[1]==1 DO M130
...
N200 ID=2 WHENEVER $A_IN[2]==1 DO RESET(1)
```

10.6 Effacer une action synchrone (CANCEL)

Fonction

L'instruction CANCEL permet d'annuler (supprimer) une action synchrone à effet modal ou statique dans le programme pièce.

Quand une action synchrone est annulée, alors que le déplacement axial de positionnement qu'elle avait enclenché est encore actif, ce déplacement sera mené à terme. Si vous ne le souhaitez pas, vous pouvez freiner le déplacement axial par effacement de la distance restant à parcourir avant l'instruction CANCEL.

Syntaxe

```
CANCEL(<n1>,<n2>,...)
```

Signification

CANCEL :	Instruction de suppression d'actions synchrones programmées
<n1>,<n2>,... :	Numéros d'identification des actions synchrones à supprimer **Nota :** sans indication de numéros d'identification, **toutes** les actions synchrones modales/statiques sont supprimées.

Exemples

Exemple 1 : annuler l'action synchrone

Code de programme	Commentaire
N100 ID=2 WHENEVER $A_IN[1]==1 DO M130	
...	
N200 CANCEL(2)	; Supprime l'action synchrone modale n° 2.

Exemple 2 : suppression de la distance restant à parcourir avant annulation de l'action synchrone

Code de programme	Commentaire
N100 ID=17 EVERY $A_IN[3]==1 DO POS[X]=15 FA[X]=1500	; Lancer le déplacement axial de positionnement.
...	
N190 WHEN ... DO DELDTG(X)	; Mettre fin au déplacement axial de positionnement.
N200 CANCEL(17)	; Supprime l'action synchrone modale n° 17.

10.7 Comportement de la commande dans des états de fonctionnement donnés

POWER ON

Avec POWER ON, aucune action synchrone n'est par principe activée. Les actions synchrones statiques peuvent être activées par le biais d'un sous-programme asynchrone (`ASUP`) lancé par l'AP.

Changement de mode de fonctionnement

Les actions synchrones activées avec le mot-clé `IDS` restent actives après un changement de mode de fonctionnement. Toutes les autres actions synchrones sont désactivées par le changement de mode de fonctionnement (par exemple le positionnement d'un axe) et sont réactivées avec le repositionnement et le retour au mode automatique.

RESET

Un RESET de la CN met fin à toutes les actions synchrones modales et non modales. Les actions synchrones statiques restent activées. De nouvelles actions peuvent être lancées à partir de celles-ci. Si un déplacement de l'axe de commande est actif lors d'un RESET, il est interrompu. Les actions synchrones de type WHEN déjà exécutées ne seront plus traitées après un RESET.

Comportement après un reset		
Action synchrone/ cycle technologique	**à effet modal/non modal**	**statique (IDS)**
	L'action active est annulée, les actions synchrones sont supprimées.	L'action active est annulée, le cycle technologique est réinitialisé.
Axe/ broche de positionnement	Le déplacement est annulé.	Le déplacement est annulé.
Broche en asservissement de vitesse	{$MA_SPIND_ACTIVE_AFTER_RESET==1 : La broche reste activée $MA_SPIND_ACTIVE_AFTER_RESET==0 : La broche s'immobilise.	
Couplage par valeur pilote	$MC_RESET_MODE_MASK, Bit13 == 1 : Le couplage par valeur pilote reste actif $MC_RESET_MODE_MASK, Bit13 == 0 : le couplage par valeur pilote est annulé	
Opérations de mesure	Les opérations de mesure lancées à partir d'actions synchrones sont annulées.	Les opérations de mesure lancées à partir d'actions synchrones statiques sont annulées.

arrêt CN

Les actions synchrones**statiques**restent activées lors de l'arrêt CN. Les déplacements amorcés par les actions synchrones statiques ne sont pas abandonnés. Les actions synchrones **spécifiques au programme** et appartenant au bloc activé restent activées, mais les déplacements qui avaient été amorcés sont abandonnés.

Fin du programme

Fin de programme et action synchrone n'ont aucune influence réciproque. Les actions synchrones en cours sont menées à terme, même après une fin de programme. Les actions synchrones activées dans le bloc M30 le restent au-delà de la fin du programme. Si vous ne le souhaitez pas, vous devez annuler l'action synchrone avant la fin du programme avec `CANCEL`.

Comportement après la fin du programme		
Action synchrone/ cycle technologique	**à effet modal / non modal → sont annulés**	**statiques (IDS) → sont conservés**
Axe/ broche de positionnement	M30 est temporisé jusqu'à ce que l'axe / la broche soit immobilisé(e).	Le déplacement se poursuit.
Broche en asservissement de vitesse	Fin de programme : {$MA_SPIND_ACTIVE_AFTER_RESET==1 : La broche reste activée $MA_SPIND_ACTIVE_AFTER_RESET==0 : La broche s'immobilise. La broche reste active en cas de changement de mode de fonctionnement.	La broche reste active.
Couplage par valeur pilote	$MC_RESET_MODE_MASK, Bit13 == 1 : Le couplage par valeur pilote reste actif $MC_RESET_MODE_MASK, Bit13 == 0 : le couplage par valeur pilote est annulé	Le couplage lancé à partir d'une action synchrone statique est conservé.
Opérations de mesure	Les opérations de mesure lancées à partir d'actions synchrones sont annulées.	Les opérations de mesure lancées à partir d'actions synchrones statiques restent actives.

Recherche de bloc

Les actions synchrones sont accumulées lors de la recherche de bloc et traitées à la reprise du programme ; les actions correspondantes sont relancées le cas échéant. Les actions synchrones statiques restent actives pendant la recherche de bloc. Si la recherche de bloc fait apparaître des coefficients de polynôme programmés avec `FCTDEF` , ces derniers seront appliqués avec effet immédiat.

Interruption du programme par un sous-programme asynchrone ASUP

Début ASUP :

Les actions synchrones au déplacement à effet modal et statiques sont conservées et restent actives dans le sous-programme asynchrone.

Fin ASUP :

Si l'exécution du sous-programme asynchrone ne se poursuit pas avec REPOS, les actions synchrones au déplacement à effet modal et statiques modifiées dans le sous-programme asynchrone restent actives dans le programme principal.

Repositionnement (REPOS)

Après Repositionnement (REPOS), les actions synchrones actives dans le bloc interrompu sont réactivées. Les actions synchrones modales modifiées depuis le sous-programme asynchrone ne sont pas actives après REPOS lors de l'exécution du bloc restant.

Les coefficients de polynôme programmés avec `FCTDEF` ne sont pas influencés par les sous-programmes asynchrones ni par REPOS. Indépendamment de l'endroit où ils ont été programmés, on peut les utiliser à tout moment dans le sous-programme asynchrone et dans le programme principal, même après un REPOS.

Comportement en cas d'alarmes

Les déplacements d'axe et de broche lancés par des actions synchrones sont freinés en cas d'activation d'une alarme avec arrêt de déplacement. L'exécution de toutes les autres actions (telles que mise à 1 d'une sortie) se poursuit.

Si une action synchrone déclenche une alarme par elle-même, une interruption de traitement survient et les actions suivantes de cette action synchrone ne sont plus exécutées. Si l'action synchrone est à effet modal, elle n'est plus traitée dans la période d'interpolation suivante. L'alarme n'est donc émise qu'une seule fois. Le traitement de toutes les autres actions synchrone se poursuit.

Les alarmes, dont la réaction est un arrêt de l'interpréteur, n'entrent en action qu'après la fin du traitement des blocs précodés.

Si un cycle technologique émet une alarme avec arrêt de déplacement, son traitement ne se poursuit pas.

Oscillation 11

11.1 Oscillation asynchrone (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB)

Fonction

Un axe d'oscillation va et vient, avec une avance donnée, entre deux points d'inversion de sens 1 et 2 jusqu'à désactivation de l'oscillation.

Pendant l'oscillation, d'autres axes peuvent être interpolés librement. Une pénétration continue peut être obtenue par un déplacement de contournage ou avec un axe de positionnement. Il n'existe toutefois **aucun lien** entre l'oscillation et le mouvement de pénétration.

Caractéristiques de l'oscillation asynchrone

- L'oscillation asynchrone agit au-delà des limites du bloc et est spécifique à un axe.
- Une activation de l'oscillation, synchronisée avec le bloc, est assurée par le programme pièce.
- Une interpolation commune de plusieurs axes et une correction de la course d'oscillation ne sont pas possibles.

Programmation

Les instructions suivantes permettent d'activer et d'agir sur l'oscillation asynchrone à partir du programme pièce, en fonction de l'exécution du programme CN.

Les valeurs programmées sont inscrites dans les données de réglage correspondantes pendant l'exécution du programme et restent actives jusqu'à la prochaine modification.

Syntaxe

```
OSP1[<axe>]=<valeur> OSP2[<axe>]=<valeur>
OST1[<axe>]=<valeur> OST2[<axe>]=<valeur>
AA[<axe>]=<valeur>
OSCTRL[<axe>]=(<options de réglage>,<option de désactivation>)
OSNSC[<axe>]=<valeur>
OSE[<axe>]=<valeur>
OSB[<axe>]=<valeur>
OS[<axe>]=1
OS[<axe>]=0
```

Signification

`<Axe>` Nom de l'axe d'oscillation

`OS` Activer/désactiver l'oscillation

Valeur : 1 **Activer** l'oscillation
0 **Désactiver** l'oscillation

`OSP1` Définir la position à partir du point d'inversion de sens 1

`OSP2` Définir la position à partir du point d'inversion de sens 2

Nota :
si un déplacement incrémental est actif, la position est calculée de manière incrémentale par rapport à la dernière position d'inversion correspondante programmée dans le programme CN.

`OST1` Définir le temps d'arrêt au point d'inversion de sens 1 en [s]

`OST2` Définir le temps d'arrêt au point d'inversion de sens 2 en [s]

`<valeur>` :
-2 l'interpolation est poursuivie sans attente de l'arrêt précis
-1 attente de l'arrêt précis grossier
0 attente de l'arrêt précis fin
>0 attente de l'arrêt précis fin puis écoulement du temps d'arrêt indiqué

Nota :
l'unité du temps d'arrêt est identique à celle programmée avec`G4`.

`FA` Définir la vitesse d'avance

La vitesse d'avance appliquée ici est celle définie pour l'axe de positionnement. Si aucune vitesse d'avance n'a été définie, c'est la valeur qui figure dans le paramètre machine qui s'applique.

`OSCTRL` Indique les options d'activation et de désactivation

Les valeurs d'option 0 - 3 verrouillent le comportement au point d'inversion à la désactivation. L'une des variantes 0 - 3 peut être sélectionnée. Une combinaison quelconque des paramétrages restant avec la variante sélectionnée est ensuite possible. La combinaison de plusieurs options s'effecue par insertion du signe plus (+).

`<valeur>` :

- 0 A la désactivation de l'oscillation, arrêt au point d'inversion suivant (paramétrage par défaut)

  Nota :
  ceci n'est possible que par réinitialisation des valeurs 1 et 2.
- 1 À la désactivation de l'oscillation, arrêt au point d'inversion de sens 1
- 2 À la désactivation de l'oscillation, arrêt au point d'inversion de sens 2
- 3 À la désactivation de l'oscillation, ne pas accoster de point d'inversion de sens, si aucune passe à lécher n'est programmée
- 4 Après les passes à lécher, accoster la position de fin
- 8 Si l'oscillation est interrompue par effacement de la distance restant à parcourir, le traitement des passes à lécher doit ensuite être terminé et, le cas échéant, la position finale accostée.
- 16 Si l'oscillation est interrompue par effacement de la distance restant à parcourir, la position d'inversion correspondante doit être accostée comme dans le cas de la désactivation.
- 32 L'avance modifiée ne prend effet qu'à partir du prochain point d'inversion de sens
- 64 FA est égal à 0, FA = 0 : le forçage du déplacement est activé

  FA est différent de 0, FA <> 0 : la correction de vitesse est activée
- 128 Dans le cas d'un axe rotatif DC (course minimale)
- 256 Passe à lécher exécutée sous forme de course double (standard). 1=passe à lécher exécutée sous forme de course simple.
- 512 Accoster d'abord la position de départ

OSNSC	Définir le nombre de passes à lécher
OSE	Définir la position finale à accoster (dans le SCP) après désactivation de l'oscillation **Nota :** la programmation de OSE active de façon implicite l'option 4 pour OSCTRL.
OSB	Définir la position de départ à accoster (dans le SCP) avant activation de l'oscillation La position de départ est accostée avant le point d'inversion de sens 1. Si la position de départ coïncide avec la position d'inversion 1, la position d'inversion 2 est ensuite accostée. Lorsque la position de départ est atteinte, aucun temps d'arrêt n'a d'effet, même lorsque la position de départ coïncide avec la position d'inversion 1, par contre un arrêt précis fin est attendu. La condition de précision réglée est respectée. **Nota :** afin que la position de départ puisse être accostée, le bit 9 doit être mis à 1 dans la donnée de réglage SD43770 $SA_OSCILL_CTRL_MASK.

Exemples

Exemple 1 : l'axe d'oscillation doit osciller entre deux points d'inversion

L'axe d'oscillation Z doit osciller entre la position 10 et la position 100. Le point d'inversion de sens 1 doit être accosté avec un arrêt précis fin, le point d'inversion de sens 2 avec un arrêt précis grossier. L'avance de l'axe d'oscillation doit être égale à 250. A la fin de l'usinage, 3 passes à lécher doivent être exécutées et l'axe d'oscillation doit atteindre la position finale 200. L'avance pour l'axe de pénétration doit être égale à 1, la fin de la pénétration en direction X doit être atteinte à la position 15.

```
Code de programme                    Commentaire
WAITP(X,Y,Z)                         ; Position de départ.
G0 X100 Y100 Z100                    ; Commutation en mode d'axe de positionnement.
WAITP(X,Z)
OSP1[Z]=10 OSP2[Z]=100               ; Point d'inversion de sens 1, point d'inversion
                                       de sens 2.
OSE[Z]=200                           ; Position finale.
OST1[Z]=0 OST2[Z]=-1                 ; Temps d'arrêt sur U1 : Arrêt précis fin
                                     ; Temps d'arrêt sur U2 : arrêt précis grossier
FA[Z]=250 FA[X]=1                    ; Avance axe d'oscillation, avance axe de
                                       pénétration.
OSCTRL[Z]=(4,0)                      ; Options de réglage.
OSNSC[Z]=3                           ; 3 passes à lécher.
OS[Z]=1                              ; Démarrer l'oscillation.
WHEN $A_IN[3]==TRUE DO DELDTG(X)     ; Effacement de la distance restant à parcourir.
POS[X]=15                            ; Position de départ axe X
POS[X]=50                              Position finale axe X.
OS[Z]=0                              ; Arrêter l'oscillation.
M30
```

Remarque

La séquence d'instructions `OSP1[Z]=...` à `OSNCS[Z]=...` peut également être programmée dans un bloc.

Exemple 2 : oscillation avec modification en ligne de la position d'inversion

Les données de réglage nécessaires pour l'oscillation asynchrone peuvent être réglées dans le programme pièce.

Si les données de réglage sont écrites directement dans le programme pièce, la modification prend effet dès le prétraitement. Le comportement synchrone peut être obtenu par un arrêt du prétraitement (`STOPRE`).

```
Code de programme                                                    Commentaire
$SA_OSCILL_REVERSE_POS1[Z]=-10
$SA_OSCILL_REVERSE_POS2[Z]=10
G0 X0 Z0
WAITP(Z)
ID=1 WHENEVER $AA_IM[Z] < $$AA_OSCILL_REVERSE_POS1[Z] DO $AA_OVR[X]=0     ; Arrêt de l'axe de
ID=2 WHENEVER $AA_IM[Z] < $$AA_OSCILL_REVERSE_POS2[Z] DO $AA_OVR[X]=0       pénétration lorsque la
                                                                           valeur réelle de l'axe
                                                                           d'oscillation a dépassé
                                                                           le point d'inversion.
OS[Z]=1 FA[X]=1000 POS[X]=40                                         ; Activer l'oscillation.
OS[Z]=0                                                              ; Désactiver l'oscillation.
M30
```

Informations complémentaires

Axe d'oscillation

Pour les axes d'oscillation, il est convenu que :

- chaque axe peut être utilisé comme axe d'oscillation
- Plusieurs axes d'oscillation peuvent être actifs simultanément (au maximum : le nombre des axes de positionnement).
- Quelle que soit la fonction G momentanément active dans le programme, l'interpolation linéaire `G1` est toujours active pour l'axe d'oscillation.

L'axe d'oscillation peut être :

- un axe d'entrée pour la transformation dynamique
- un axe directeur pour les axes Gantry et conjugués
- déplacé :
  - sans limitation des à-coups (BRISK)

    ou
  - avec limitation des à-coups (SOFT)

    ou
  - avec une ligne caractéristique d'accélération coudée (comme les axes de positionnement)

Points d'inversion de sens

Pour déterminer les points d'inversion de l'oscillation, il convient de tenir compte des décalages actuels ci-après :

- Déclaration en absolu

  OSP1[Z]=<valeur>

  Position du point d'inversion = Somme des décalages + valeur programmée
- Déclaration en relatif

  OSP1[Z]=IC(<valeur>)

  Position du point d'inversion = point d'inversion 1 + valeur programmée

Exemple :

Code de programme

```
N10 OSP1[Z]=100 OSP2[Z]=110
...
...
N40 OSP1[Z]= IC(3)
```

WAITP

Pour faire osciller un axe géométrique, il faut le débloquer avecWAITP pour l'oscillation.

Lorsque l'oscillation est terminée, WAITP l'axe d'oscillation est à nouveau défini comme axe de positionnement et peut à nouveau être utilisé normalement.

Oscillation avec actions synchrones au déplacement et temps d'arrêt

Après écoulement des temps d'arrêt réglés, le changement de bloc interne a lieu pendant l'oscillation (visible aux nouvelles distances restant à parcourir des axes). La fonction de désactivation est vérifiée lors du changement de bloc. Elle est déterminée après le réglage de la séquence de déplacements (OSCTRL). *Ce comportement temporel peut être modifié par la correction de l'avance.*

Il est possible qu'une course d'oscillation soit encore effectuée ensuite avant le démarrage des passes à lécher ou l'accostage de la position finale. *On a l'impression que le comportement à la désactivation est modifié. Ceci n'est toutefois pas le cas.*

11.2 Oscillation commandée par des actions synchrones (OSCILL)

Fonction

Avec ce type d'oscillation, le mouvement de pénétration n'est autorisé qu'aux points d'inversion de sens ou dans des zones d'inversion de sens bien définies.

Selon les besoins, l'oscillation peut

- soit être poursuivie
- soit être arrêtée jusqu'à ce que la pénétration soit entièrement effectuée.

Syntaxe

1. **Fixer les paramètres pour l'oscillation**
2. **Définir les actions synchrones au déplacement**
3. **Affecter les axes, fixer la profondeur de pénétration**

Signification

`OSP1[<Axe d'oscillation>]=`	Position du point d'inversion de sens 1
`OSP2[<Axe d'oscillation>]=`	Position du point d'inversion de sens 2
`OST1[<Axe d'oscillation>]=`	Temps d'arrêt au point d'inversion de sens 1 en secondes
`OST2[<Axe d'oscillation>]=`	Temps d'arrêt au point d'inversion de sens 2 en secondes
`FA[<Axe d'oscillation>]=`	Avance de l'axe d'oscillation
`OSCTRL[<Axe d'oscillation>]=`	Options d'activation ou de désactivation
`OSNSC[<Axe d'oscillation>]=`	Nombre de passes à lécher
`OSE[<Axe d'oscillation>]=`	Position de fin
`WAITP(<Axe d'oscillation>)`	Libération de l'axe pour l'oscillation

Affectation des axes, profondeur de pénétration

```
OSCILL[<Axe d'oscillation>]=(<Axe de pénétration 1>,<Axe de
pénétration 2>,<Axe de pénétration 3>)
POSP[<Axe de pénétration>]=(<Position de fin>,<Longueur
partielle>,<Mode>)
```

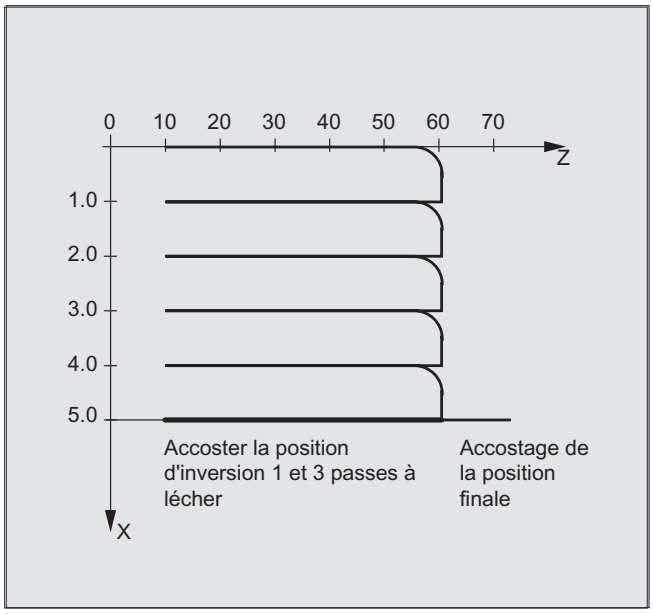
`OSCILL` :	Affecter l'axe ou les axes de pénétration à l'axe d'oscillation
`POSP` :	Définir la pénétration totale et partielle (voir chapitre Gestion des fichiers et programmes)
`Position de fin` :	Position de fin pour l'axe de pénétration, quand toutes les pénétrations partielles ont été effectuées.
`Longueur partielle` :	Valeur de la pénétration partielle au point d'inversion de sens/dans la zone d'inversion de sens
`Mode` :	Division de la pénétration totale en pénétrations partielles = deux dernières pénétrations partielles identiques (préréglage) ; = toutes les pénétrations partielles identiques

Actions synchrones au déplacement

`WHEN… … DO`	Quand..., alors...
`WHENEVER … DO`	A chaque fois que..., alors...

Exemple

Il ne doit pas y avoir de pénétration au point d'inversion de sens 1. Au point d'inversion de sens 2, la pénétration doit déjà commencer à la distance ii2 du point d'inversion de sens accoster position de fin et l'axe d'oscillation ne doit pas attendre la fin de la pénétration partielle. L'axe Z est axe d'oscillation et l'axe X axe de pénétration.



1. Paramètres pour l'oscillation

Code de programme	Commentaire
`DEF INT ii2`	`; définir la variable pour la zone d'inversion de sens 2`
`OSP1[Z]=10 OSP2[Z]=60`	`; définir les points d'inversion de sens 1 et 2`
`OST1[Z]=0 OST2[Z]=0`	`; Point d'inversion 1 : Arrêt précis fin` `Point d'inversion 2 : Arrêt précis fin`
`FA[Z]=150 FA[X]=0.5`	`; avance de l'axe d'oscillation Z, avance de l'axe de pénétration X`
`OSCTRL[Z]=(2+8+16,1)`	`; désactiver l'oscillation au point d'inversion de sens 2 ; après effacement de la distance restant à parcourir, passes à lécher et accostage de la position de fin ; après effacement de la distance restant à parcourir, accostage de la position d'inversion de sens correspondante`
`OSNC[Z]=3`	`; Passes à lécher`
`OSE[Z]=70`	`; position de fin = 70`
`ii2=2`	`; régler la zone d'inversion de sens`
`WAITP(Z)`	`; autorise l'oscillation pour l'axe Z`

2. Action synchrone au déplacement

Code de programme		Commentaire
WHENEVER $AA_IM[Z]<$SA_OSCILL_REVERSE_POS2[Z] DO -> $AA_OVR[X]=0 $AC_MARKER[0]=0	;	A chaque fois que la position courante de l'axe d'oscillation Z est inférieure au début de la zone d'inversion de sens 2 dans le SCM, mettre la correction axiale de l'axe de pénétration X à 0% et le mémento avec l'indice 0 à la valeur 0.
WHENEVER $AA_IM[Z]>=$SA_OSCILL_REVERSE_POS2[Z] DO $AA_OVR[Z]=0	;	A chaque fois que la position courante de l'axe d'oscillation Z est supérieure à la position d'inversion de sens 2 dans le SCM, mettre la correction axiale de l'axe d'oscillation Z à 0%.
WHENEVER $AA_DTEPW[X] == 0 DO $AC_MÉMENTO[0]=1	;	A chaque fois que la distance restant à parcourir pour l'approche partielle est égale à 0, mettre le mémento avec l'indice 0 à la valeur 1.
WHENEVER $AC_MARKER[0]==1 DO $AA_OVR[X]=0 $AA_OVR[Z]=100	;	A chaque fois que le mémento avec l'indice 0 est égal à 1, mettre la correction axiale de l'axe d'approche X à 0%. Ceci permet d'empêcher une approche prématurée (axe d'oscillation Z n'a pas encore quitté la zone d'inversion 2, l'axe d'approche X est toutefois prêt pour une nouvelle approche). Modifier la correction axiale de l'axe d'oscillation Z de 0% (action de la 2ème action synchrone) de nouveau à la valeur 100% pour le déplacement.

-> doit être programmé dans un bloc

3. Démarrer l'oscillation

Code de programme		Commentaire
OSCILL[Z]=(X) POSP[X]=(5,1,1)	;	Démarrage des axes L'axe X est affecté comme axe de pénétration à l'axe d'oscillation Z. L'axe X doit rejoindre la position de fin 5 par incréments de 1.
M30	;	Fin du programme

Description

1. **Fixer les paramètres pour l'oscillation**
   Les paramètres pour l'oscillation sont à fixer dans le programme avant le bloc de déplacement qui contient d'une part la correspondance entre l'axe de pénétration et l'axe d'oscillation et d'autre part l'instruction avec les paramètres de pénétration (voir "Oscillation asynchrone").
2. **Définir les actions synchrones au déplacement {**
   Par le biais des actions synchrones, il est possible de :
   Inhiber le mouvement de pénétration, jusqu'à ce que l'axe d'oscillation se trouve à l'intérieur d'une zone d'inversion
   (ii1, ii2) ou à un point d'inversion (U1, U2).
   Arrêterl'oscillation pendant la pénétration au point d'inversion de sens.
   Redémarrerl'oscillation lorsque la pénétration partielle est**terminée**.
   Définir le lancement de la pénétration partielle suivante .
3. **Affecter les axes d'oscillation et de pénétration** et définir **la pénétration totale et partielle**.

Fixer les paramètres pour l'oscillation

Affectation des axes d'oscillation et de pénétration : OSCILL

```
OSCILL[axe d'oscillation] = (axe de pénétration1, axe de pénétration2, axe de
pénétration3)
```

L'instruction `OSCILL` permet d'effectuer les affectations d'axe et le démarrage de l'oscillation.

Trois axes de pénétration peuvent être affectés au maximum à un axe d'oscillation.

Remarque

Avant le démarrage de l'oscillation, les conditions synchrones pour le comportement des axes doivent être définies.

Définition des pénétrations : POSP

```
POSP[axe de pénétration] = (pos. de fin, pénétr. partielle, mode)
```

Avec l'instruction `POSP` , on fournit les indications suivantes à la commande :

- la pénétration totale (par le biais de la position de fin)
- la valeur de la pénétration partielle au point d'inversion de sens ou dans la zone d'inversion de sens
- le comportement lorsque la position de fin est atteinte (par le biais du mode)

Mode = 0	Pour les deux dernières pénétrations partielles, la distance restant à parcourir jusqu'à la destination est divisée en deux incréments identiques (préréglage).
Mode = 1	Toutes les pénétrations partielles sont identiques. Elles sont calculées à partir de la pénétration totale.

Définir les actions synchrones au déplacement

De façon tout à fait générale, on utilise pour l'oscillation les actions synchrones au déplacement énumérées ci-après.

Des exemples sont donnés qui pourront vous servir de base pour réaliser vos propres programmes d'oscillation.

Remarque

Dans certains cas particuliers, les conditions synchrones peuvent également être programmées différemment.

des mots-clés

`WHEN … DO …`	Quand..., alors...
`WHENEVER … DO`	A chaque fois que..., alors...

Fonctions

Avec les instructions décrites en détail ci-après, vous pouvez réaliser les fonctions

suivantes :

1. Pénétration au point d'inversion de sens.
2. Pénétration dans la zone d'inversion de sens.
3. Pénétration aux deux points d'inversion de sens.
4. Arrêt de l'oscillation au point d'inversion de sens.
5. Redémarrage de l'oscillation.
6. Inhibition puis démarrage de la pénétration partielle suivante.

Toutes les actions synchrones mentionnées ici à titre d'exemple sont basées sur les hypothèses { suivantes :

- point d'inversion de sens 1 < point d'inversion de sens 2
- Z = axe d'oscillation
- X = axe de pénétration

Remarque

Pour plus d'explications, voir le chapitre Actions synchrones au déplacement.

Affecter les axes d'oscillation et de pénétration et définir la pénétration totale et partielle.

Pénétration dans la zone d'inversion de sens

Le mouvement de pénétration doit débuter à l'intérieur d'une zone d'inversion de sens, avant que le point d'inversion de sens ne soit atteint.

Ces actions synchrones inhibent le mouvement de pénétration jusqu'à ce que l'axe d'oscillation se trouve dans une zone d'inversion de sens.

Avec les hypothèses formulées (voir ci-dessus), les instructions sont les suivantes :

Zone d'inversion de sens 1 :

```
WHENEVER
$AA_IM[Z]>$SA_OSCILL_RESE
RVE_POS1[Z]+ii1 DO
$AA_OVR[X] = 0
```

A chaque fois que la position courante de l'axe d'oscillation est supérieure au début de la zone d'inversion de sens 1 dans le SCM, mettre la correction axiale de l'axe de pénétration à 0%.

Zone d'inversion de sens 2 :

```
WHENEVER
$AA_IM[Z]<$SA_OSCILL_RESE
RVE_POS2[Z]+ii2 DO
$AA_OVR[X] = 0
```

A chaque fois que la position courante de l'axe d'oscillation est inférieure au début de la zone d'inversion de sens 2 dans le SC, mettre la correction axiale de l'axe de pénétration à 0%.

Pénétration au point d'inversion de sens

Tant que l'axe d'oscillation n'a pas atteint le point d'inversion, l'axe de pénétration n'effectue aucun déplacement.

Avec les hypothèses formulées (voir ci-dessus), les instructions sont les suivantes :

Zone d'inversion de sens 1 :

```
WHENEVER
$AA_IM[Z]<>$SA_OSCILL_RES
ERVE_POS1[Z] DO
$AA_OVR[X] = 0 →
→ $AA_OVR[Z] = 100
```

A chaque fois que la position courante de l'axe d'oscillation Z est supérieure ou inférieure à la position du point d'inversion de sens 1 dans le SCM, mettre la correction axiale de l'axe de pénétration X à 0% et la correction axiale de l'axe d'oscillation Z à 100%.

Zone d'inversion de sens 2 :

Pour le point d'inversion 2 :

```
WHENEVER
$AA_IM[Z]<>$SA_OSCILL_RES
ERVE_POS2[Z] DO
$AA_OVR[X] = 0 →
→ $AA_OVR[Z] = 100
```

A chaque fois que la position courante de l'axe d'oscillation Zu est supérieure ou inférieure à la position du point d'inversion de sens 2 dans le SCM, mettre la correction axiale de l'axe de pénétration X à 0% et la correction axiale de l'axe d'oscillation Z à 100%.

Arrêt de l'oscillation au point d'inversion de sens

L'axe d'oscillation s'arrête au point d'inversion de sens, le mouvement de pénétration commence au même moment. L'oscillation est poursuivie lorsque la pénétration est entièrement effectuée.

Cette action synchrone peut également être utilisée pour redémarrer le mouvement de pénétration si celui-ci a été arrêté par une action synchrone précédente encore active.

Avec les hypothèses formulées (voir ci-dessus), les instructions sont les suivantes :

Zone d'inversion de sens 1 :

```
WHENEVER
$AA_IM[Z]<>$SA_OSCILL_RES
ERVE_POS1[Z] DO
$AA_OVR[X] = 0 →
→ $AA_OVR[Z] = 100
```

A chaque fois que la position courante de l'axe d'oscillation est égale à la position d'inversion de sens 1 dans le SCM, mettre la correction axiale de l'axe d'oscillation à 0% et la correction axiale de l'axe de pénétration à 100%.

Zone d'inversion de sens 2 :

```
WHENEVER
$AA_IM[Z]<>$SA_OSCILL_RES
ERVE_POS1[Z] DO
$AA_OVR[X] = 0 →
→ $AA_OVR[Z] = 100
```

A chaque fois que la position courante de l'axe d'oscillation Zu est égale à la position d'inversion de sens 2 dans le SCM, mettre la correction axiale de l'axe d'oscillation X à 0% et la correction axiale de l'axe de pénétration à 100%.

Exploitation en ligne du point d'inversion de sens

Si une variable d'exécution marquée par `$$` figure du côté droit de la comparaison, les deux variables seront exploitées en continu à la période d'appel de l'interpolateur et comparées entre elles.

Remarque

Pour plus d'informations à ce sujet, voir le chapitre "Actions synchrones au déplacement".

Redémarrage de l'oscillation

Cette action synchrone sert à reprendre le mouvement d'oscillation, quand la pénétration partielle est terminée.

Avec les hypothèses formulées (voir ci-dessus), les instructions sont les suivantes :

```
WHENEVER $AA_DTEPW[X]==0
DO $AA_OVR[Z]= 100
```

A chaque fois que la distance restant à parcourir pour la pénétration partielle de l'axe de pénétration X est égale à zéro dans le SCM, mettre la correction axiale de l'axe d'oscillation à 100%.

Pénétration partielle suivante

Lorsque la pénétration a été effectuée, il convient d'éviter que la pénétration partielle suivante ne démarre trop tôt.

On utilise pour ce faire un mémento spécifique au canal (`$AC_MARKER[Index]`) qui est mis à 1 à la fin de la pénétration partielle (distance partielle restant à parcourir 0) et qui est effacé à la sortie de la zone d'inversion de sens. Une action synchrone empêche ensuite le mouvement de pénétration suivant.

Selon les hypothèses formulées (voir ci-dessus), les instructions sont les suivantes pour le point d'inversion de sens 1 par exemple :

1. Définir les marques :

```
WHENEVER $AA_DTEPW[X]==0
DO $AC_MARKER[1] = 1
```

A chaque fois que la distance restant à parcourir pour la pénétration partielle de l'axe de pénétration X est égale à zéro dans le SCM, mettre le mémento avec l'indice 1 à la valeur 1.

2. Effacer les marques

```
WHENEVER $AA_IM[Z]<>
$SA_OSCILL_RESERVE_POS1[Z
] DO $AC_MARKER[1]= 0
```

A chaque fois que la position courante de l'axe d'oscillation Z est supérieure ou inférieure à la position du point d'inversion de sens 1 dans le SCM, mettre le mémento 1 à 0.

3. Inhiber la pénétration

```
WHENEVER $AC_MARKER[1]==1
DO $AA_OVR[X] = 0
```

A chaque fois que le mémento 1 est égal, mettre la correction axiale de l'axe de pénétration X à 0%.

Poinçonnage et grignotage 12

12.1 Activation, désactivation

12.1.1 Activation ou désactivation du poinçonnage et du grignotage (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC)

Fonction

Activation/désactivation du poinçonnage ou du grignotage

PON et SON permettent d'activer la fonction de poinçonnage ou de grignotage. SPOF met fin à toutes les fonctions spécifiques au poinçonnage et au grignotage. Les instructions modales PON et SON s'excluent mutuellement, c'est-à-dire que PON désactive SON et vice versa.

Poinçonnage/grignotage à la période d'appel de l'interpolateur

Avec SONS et PONS, vous activez aussi respectivement la fonction poinçonnage et la fonction grignotage.

Contrairement au pilotage du poinçon au niveau de l'interpolation actif avec SON/PON, le pilotage du poinçon s'effectue ici par le biais de signaux au niveau servo. Ceci permet de travailler avec une cadence de frappe plus rapide et donc une meilleure productivité.

Pendant le traitement des signaux à la période d'appel de l'interpolateur, toutes les fonctions susceptibles de modifier la position des axes de grignotage ou de poinçonnage sont verrouillées (p. ex. déplacement par manivelle électronique, modifications de frames via AP, fonctions de mesure).

Poinçonnage avec temporisation

PDELAYON temporise l'application du coup de poinçon. Cette instruction modale comporte une fonctionnalité préparatoire et se place de ce fait, en règle générale, avant PON. Après PDELAYOF, le poinçonnage se poursuit de manière normale.

Remarque

La définition du temps de retard s'effectue dans la donnée de réglage SD42400 $SC_PUNCH_DWELLTIME.

Accélération asservie à la course

PUNCHACC permet de paramétrer une caractéristique d'accélération, définissant différentes accélérations selon la distance entre les trous.

Deuxième interface de poinçonnage

Pour les machines utilisant en alternance une deuxième interface de poinçonnage (deuxième unité de poinçonnage ou support similaire), il est possible de commuter sur une deuxième paire d'entrées et de sorties TOR rapides de la commande (paire d'E/S). La fonctionnalité de poinçonnage/grignotage complète est applicable avec chacune des deux interfaces de poinçonnage. La commutation entre la première et la deuxième interface de poinçonnage s'effectue avec les instructions `SPIF1` et `SPIF2`.

Remarque

Condition : une deuxième paire d'E/S doit être définie dans un paramètre machine pour la fonctionnalité de poinçonnage (→ voir les indications du constructeur de la machine !).

Syntaxe

```
PON G... X... Y... Z...
SON G... X... Y... Z...
SONS G... X... Y... Z...
PONS G... X... Y... Z...
PDELAYON
PDELAYOF
PUNCHACC(<Smini>,<Amini>,<Smaxi>,<Amaxi>)
SPIF1/SPIF2
SPOF
```

Signification

`PON` Activer le poinçonnage

`SON` Activer le grignotage

`PONS` Activer le poinçonnage à la période d'appel de l'interpolateur

`SONS` Activer le grignotage à la période d'appel de l'interpolateur

`SPOF` Désactiver le poinçonnage/grignotage

`PDELAYON` Activer le poinçonnage avec temporisation

`PDELAYOF` Désactiver le poinçonnage avec temporisation

`PUNCHACC` Activer l'accélération en fonction de la course

Paramètre :

- `<Smini>` Distance minimale entre les trous
- `<Amini>` Accélération initiale
  `<Amini>` peut être supérieur à `<Amaxi>`.
- `<Smaxi>` Distance maximale entre les trous
- `<Amaxi>` Accélération finale
  `<Amax>` peut être supérieur à `<Amin>`.

`SPIF1` Activer la **première** interface de poinçonnage

Le pilotage du poinçon s'effectue avec la première paire d'E/S rapides.

`SPIF2` Activer la **deuxième** interface de poinçonnage

Le pilotage du poinçon s'effectue avec la deuxième paire d'E/S rapides.

Nota :
après un RESET ou un démarrage de la commande, la première interface de poinçonnage est toujours active. Si une seule interface de poinçonnage est utilisée, il n'est pas nécessaire de la programmer.

Exemples

Exemple 1 : activer le grignotage

```
Code de programme          Commentaire
...
N70 X50 SPOF               ; Positionnement sans déclenchement de poiçonnage.
N80 X100 SON               ; Activer le grignotage, déclenchement d'un coup avant le
                             déplacement (X=50) et à la fin du déplacement programmé
                             (X=100).
...
```

Exemple 2 : poinçonnage avec temporisation

```
Code de programme          Commentaire
...
N170 PDELAYON X100 SPOF    ; Positionnement sans déclenchement de poinçonnage,
                             activation du déclenchement de poinçonnage temporisé.
N180 X800 PON              ; Activer le poinçonnage. Lorsque la position finale est
                             atteinte, un coup de poinçon est déclenché avec une
                             temporisation.
N190 PDELAYOF X700         ; Désactiver le poinçonnage avec une temporisation,
                             déclenchement normal du poinçonnage à la fin du
                             déplacement programmé.
...
```

Exemple 3 : poinçonnage avec deux interfaces de poinçonnage

```
Code de programme          Commentaire
...
N170 SPIF1 X100 PON        ; A la fin du bloc, le déclenchement d'un coup est
                             réalisé sur la première sortie rapide. Une surveillance
                             du signal "Coup actif" a lieu sur la première entrée.
N180 X800 SPIF2            ; Le deuxième déclenchement d'un coup est réalisé sur la
                             deuxième sortie rapide. Une surveillance du signal
                             "Coup actif" a lieu sur la deuxième entrée.
N190 SPIF1 X700            ; La commande du coup s'effectue avec la première
                             interface pour tous les autres coups.
...
```

Informations complémentaires

Poinçonnage et grignotage à la période d'appel de l'interpolateur (PONS/SONS)

Le poinçonnage et le grignotage à la période d'appel de l'interpolateur ne sont pas possibles simultanément dans plusieurs canaux. L'activation de PONS ou SONS n'est possible que dans un seul canal à la fois.

Accélération en fonction de la course (PUNCHACC)

Exemple :
```
PUNCHACC(2,50,10,100)
```

Distances entre les trous inférieures à 2mm :

Le déplacement a lieu avec une accélération égale à 50% de l'accélération maximale.

Distances entre les trous comprises entre 2mm et 10mm :

L'accélération augmente proportionnellement à la distance, jusqu'à 100%.

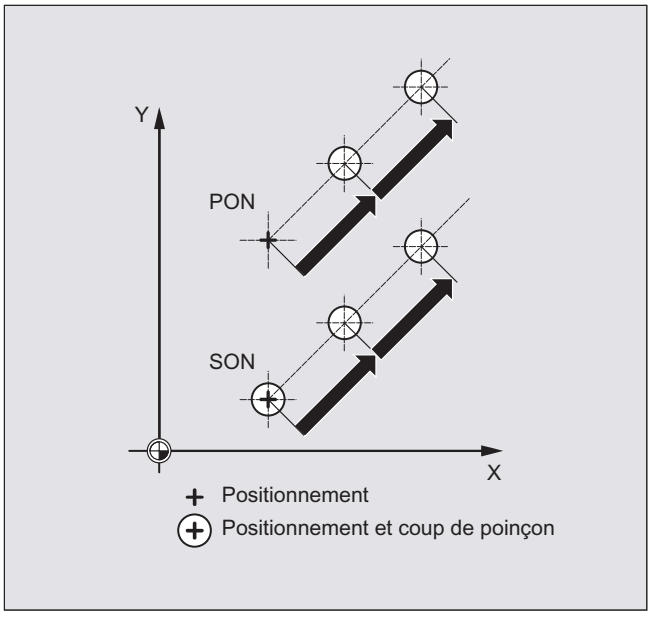
Distances entre les trous supérieures à 10mm :

Déplacement avec une accélération égale à 100%.

Déclenchement du premier coup de poinçon

Selon qu'il s'agit du poinçonnage ou du grignotage, le déclenchement du premier coup de poinçon, après activation de la fonction, intervient à des instants différents :

- PON/PONS :
  - Tous les coups de poinçon - y compris celui du premier bloc après l'activation - sont exécutés en fin de bloc.
- SON/SONS :
  - Le premier coup de poinçon après l'activation du grignotage, est toujours exécuté en début de bloc.
  - Les autres coups de poinçon sont toujours libérés en fin de bloc.



Poinçonnage et grignotage sur place

Un coup de poinçon ne peut être déclenché que si le bloc contient une instruction de déplacement pour les axes de poinçonnage ou de grignotage (axes du plan actif).

Afin de pouvoir néanmoins déclencher un coup de poinçon, tout en restant au même endroit, l'un des axes de poinçonnage/grignotage est programmé avec un déplacement égal à 0.

Utilisation d'outils indexables

Remarque

Pour positionner des outils indexables de façon tangentielle à la trajectoire programmée, utilisez la fonction Positionnement tangentiel.

Utilisation d'instructions M

Grâce à la macroprogrammation, il est toujours possible d'utiliser des fonctions M spéciales au lieu des instructions en language de programmation (compatibilité). Les correspondances avec les systèmes plus anciens sont les suivantes :

```
M20, M23    ≙  SPOF
M22         ≙  SON
M25         ≙  PON
M26         ≙  PDELAYON
```

Exemple de fichier macro :

```
Code de programme                Commentaire
DEFINE M25 AS PON                ; activation du poinçonnage
DEFINE M125 AS PONS              ; Activation du poinçonnage à la période d'appel
                                   de l'interpolateur
DEFINE M22 AS SON                ; activation du grignotage
DEFINE M122 AS SONS              ; Activation du grignotage à la période d'appel de
                                   l'interpolateur
DEFINE M26 AS PDELAYON           ; Activation du poinçonnage avec temporisation
DEFINE M20 AS SPOF               ; Désactivation du poinçonnage, du grignotage
DEFINE M23 AS SPOF               ; Désactivation du poinçonnage, du grignotage
```

Exemple de programmation :

```
Code de programme                Commentaire
...
N100 X100 M20                    ; Positionnement sans déclenchement de poiçonnage.
N110 X120 M22                    ; Activer le grignotage, déclenchement d'un coup
                                   avant et après le déplacement.
N120 X150 Y150 M25               ; Activer le poinçonnage, déclenchement d'un coup
                                   à la fin du déplacement.
...
```

12.2 Préparation automatique du déplacement

Fonction

Segmentation en distances partielles

Quand le poinçonnage ou le grignotage est activé, SPP et SPN provoquent la segmentation en un nombre donné de distances partielles de même longueur (répartition équidistante) de l'ensemble du déplacement programmé pour les axes à interpolation. En interne, à chaque distance partielle correspond un bloc.

Nombre de coups de poinçon

En poinçonnage, le premier coup de poinçon intervient au point final de la première distance partielle ; par contre, en grignotage, il intervient au point de départ de la première distance partielle. Sur la distance globale parcourue, les valeurs sont les suivantes :

Poinçonnage : nombre de coups = nombre de distances partielles

Grignotage : nombre de coups = nombre de distances partielles + 1

Fonctions auxiliaires

Les fonctions auxiliaires sont exécutées dans le premier des blocs générés.

Syntaxe

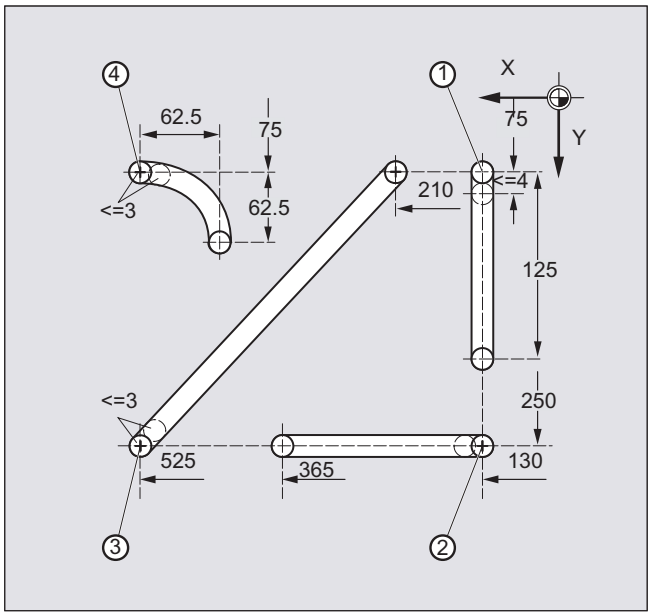
```
SPP=
SPN=
```

Signification

SPP	Taille de la distance partielle (distance maximale entre coups de poinçon consécutifs) ; fonction à effet modal
SPN	Nombre de distances partielles par bloc ; fonction à effet modal

Exemple 1

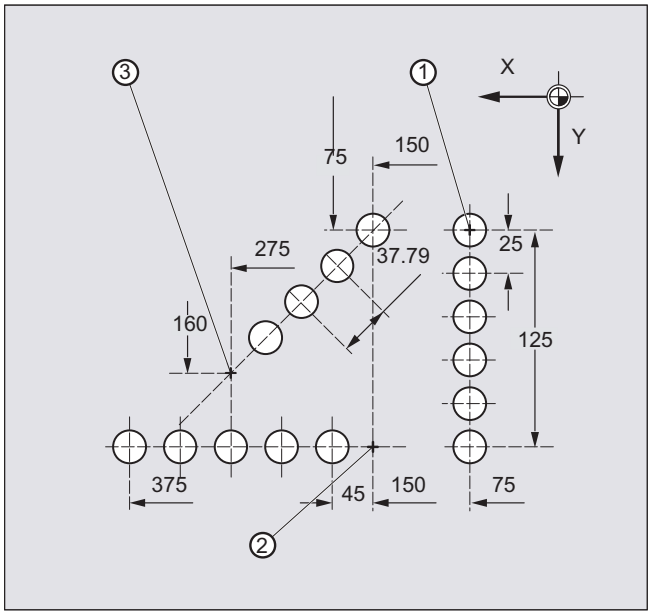
Les parcours de grignotage doivent être décomposés automatiquement en distances partielles identiques.



Code de programme	**Commentaire**
N100 G90 X130 Y75 F60 SPOF	; Positionnement sur le point de départ 1
N110 G91 Y125 SPP=4 SON	; Activer le grignotage ; longueur maximale de distance partielle pour segmentation automatique du déplacement : 4 mm
N120 G90 Y250 SPOF	; Désactiver le grignotage ; positionnement sur le point de départ 2
N130 X365 SON	; Activer le grignotage ; longueur maximale de distance partielle pour segmentation automatique du déplacement : 4 mm
N140 X525 SPOF	; Désactiver le grignotage ; positionnement sur le point de départ 3
N150 X210 Y75 SPP=3 SON	; Activer le grignotage ; longueur maximale de distance partielle pour segmentation automatique du déplacement : 3 mm
N160 X525 SPOF	; Désactiver le grignotage ; positionnement sur le point de départ 4
N170 G02 X-62.5 Y62.5 I J62.5 SPP=3 SON	; Activer le grignotage ; longueur maximale de distance partielle pour segmentation automatique du déplacement : 3 mm
N180 G00 G90 Y300 SPOF	; Désactiver le grignotage

Exemple 2

On souhaite une segmentation automatique de la distance à parcourir pour chaque rangée de trous. Pour la segmentation, la longueur maximale de distance partielle (valeur de SPP) sera programmée.



Code de programme		Commentaire
N100 G90 X75 Y75 F60 PON	;	Positionnement sur le point de départ 1 ; activation du poinçonnage, poinçonner un trou unique
N110 G91 Y125 SPP=25	;	Longueur maximale de distance partielle pour segmentation automatique du déplacement : 25 mm
N120 G90 X150 SPOF	;	Désactiver le poinçonnage ; positionnement sur le point de départ 2
N130 X375 SPP=45 PON	;	Activer le poinçonnage ; longueur maximale de distance partielle pour segmentation automatique du déplacement : 45 mm
N140 X275 Y160 SPOF	;	Désactiver le poinçonnage ; positionnement sur le point de départ 3
N150 X150 Y75 SPP=40 PON	;	Activer le poinçonnage ; au lieu de la longueur de distance partielle programmée de 40 mm, la longueur de distance partielle calculée de 37,79 mm est utilisée.
N160 G00 Y300 SPOF	;	Désactiver le poinçonnage ; positionnement

12.2.1 Segmentation du déplacement dans le cas d'axes à interpolation

Longueur de la distance partielle SPP

Avec `SPP` , vous définissez la distance maximale entre coups de poinçon consécutifs et également la longueur maximale des distances partielles, sur lesquelles le déplacement total doit être réparti. La désactivation de l'instruction se fait avec `SPOF` ou `SPP=0`.

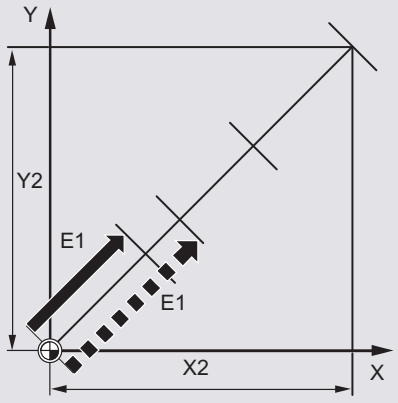
Exemple :

```
N10 SON X0 Y0
N20 SPP=2 X10
```

Le déplacement total de 10 mm est divisé en 5 distances partielles de 2 mm chacune (`SPP=2`).

Remarque

La segmentation du déplacement avec `SPP` donne toujours lieu à des déplacements équidistants : toutes les distances partielles ont la même longueur. Autrement dit, la taille de la distance partielle programmée (valeur de `SPP`) n'est valable que si, et seulement si, le quotient du déplacement total par la valeur de SPP est un entier. Si ce n'est pas le cas, la taille de la distance partielle est réduite, de façon interne, de manière à que le quotient soit un entier.



X2/Y2 Distance à parcourir programmée (bloc de grignotage / poinçonnage)
E1 Distance partielle programmée
E1' Distance partielle automatiquement arrondie

Exemple :

```
N10 G1 G91 SON X10 Y10
N20 SPP=3.5 X15 Y15
```

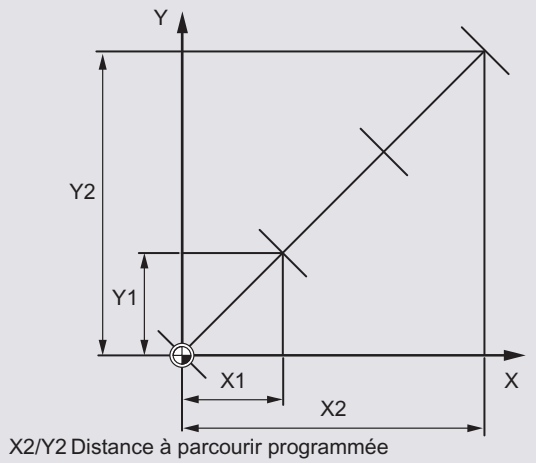
Avec un déplacement total de 15 mm et une distance partielle de 3,5 mm, on obtient un quotient qui n'est pas un nombre entier (4.28). Une réduction de la valeur de SPP a donc lieu jusqu'à l'obtention d'un quotient entier. Il en résulte une taille de distance partielle de 3 mm.

Nombre de distances partielles SPN

Avec SPN , vous précisez le nombre de distances partielles à générer sur l'ensemble du déplacement. La longueur des distances partielles est calculée automatiquement. SPN ayant un effet non modal, il convient d'activer auparavant le poinçonnage ou le grignotage avec PON ou SON.

SPP et SPN dans un même bloc

Si vous programmez, dans un même bloc, aussi bien la longueur d'une distance partielle (SPP) que le nombre de distances partielles (SPN), SPN sera valide seulement pour le bloc en question et SPP le sera pour les blocs suivants. Si SPP a déjà été activé avant SPN , il sera à nouveau actif après le bloc contenant SPN .



X2/Y2 Distance à parcourir programmée
X1 Distance partielle calculée automatiquement en X
Y1 Distance partielle calculée automatiquement en Y

Remarque

Si le poinçonnage/grignotage sont disponibles sur votre commande numérique, vous pouvez utiliser la programmation de la segmentation automatique du déplacement avec SPN et SPP , indépendamment de cette technologie.

12.2.2 Segmentation du déplacement dans le cas d'axes individuels

Si, en plus des axes à interpolation, des axes individuels sont aussi configurés comme axes de poinçonnage/grignotage, ils sont aussi soumis à la segmentation automatique de la distance à parcourir.

Comportement d'un axe individuel en présence de SPP

La longueur programmée de la distance partielle (SPP) s'adresse en premier lieu aux axes à interpolation. C'est la raison pour laquelle la valeur SPP est ignorée dans un bloc qui contient un déplacement d'axe individuel et une valeur SPP, mais pas d'axe à interpolation.

Si, dans un bloc, des axes individuels aussi bien que des axes à interpolation sont programmés, le comportement de l'axe individuel dépend du réglage du paramètre machine concerné.

1. Réglage standard

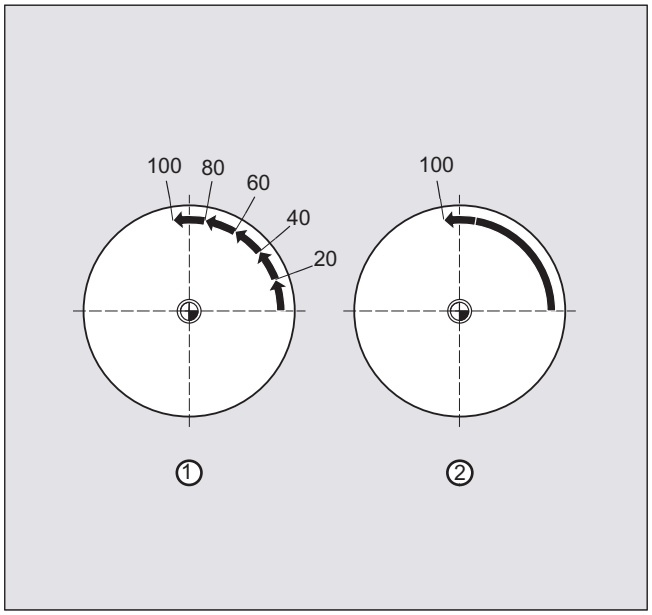
   Le déplacement de l'axe individuel est réparti de façon égale sur les blocs intermédiaires générés par SPP .

Exemple :

```
N10 G1 SON X10 A0
N20 SPP=3 X25 A100
```

En fonction de la distance entre coups de poinçon de 3 mm, le déplacement global de l'axe X (axe à interpolation) de 15 mm donne lieu à 5 blocs.

L'axe A pivote de ce fait de 20° par bloc.



1. Axe individuel sans segmentation en distances partielles

   L'axe individuel parcourt son déplacement global dans le premier des blocs générés.

2. Segmentation variable en distances partielles

   Le comportement de l'axe individuel dépend de l'interpolation des axes à interpolation :

- Interpolation circulaire : Segmentation en distances partielles
- Interpolation linéaire : Pas de segmentation en distances partielles

Comportement avec SPN

Le nombre de distances partielles programmé reste valable, même si un axe à interpolation n'a pas été programmé en même temps.

Condition : l'axe individuel est défini comme axe de poinçonnage-grignotage.

Rectification 13

13.1 Surveillance d'outil spécifique à la rectification dans le programme pièce (TMON, TMOF)

Fonction

Avec l'instruction TMON, vous pouvez activer, dans le programme pièce CN, la surveillance de la géométrie et de la vitesse de rotation pour des outils de rectification (types 400 à 499). La surveillance reste active jusqu'à ce qu'elle soit désactivée dans le programme pièce avec l'instruction TMOF.

Remarque

Respectez les consignes du constructeur de la machine !

Conditions requises

Les paramètres d'outil $TC_TPG1 à $TC_TPG9 spécifiques à la rectification doivent être activés.

Syntaxe

```
TMON (<n° T>)
TMOF (<n° T>)
```

Signification

TMON	Instruction d'**activation** de la surveillance d'outil spécifique à la rectification
TMOF	Instruction de **désactivation** de la surveillance d'outil spécifique à la rectification
<n° T>	Indication du numéro T **Remarque :** L'indication du numéro T est nécessaire uniquement si l'outil portant ce numéro T n'est pas actif.
TMOF(0)	désactiver la surveillance pour tous les outils

Autres informations

Paramètres d'outil spécifiques à la rectification

Liste des paramètres	Signification	type de données
$TC_TPG1	numéro de broche	INT
$TC_TPG2	Règle de concaténation Les paramètres du côté gauche et du côté droit de la meule sont égalisés automatiquement.	INT
$TC_TPG3	Rayon minimal de la meule	REAL
$TC_TPG4	Largeur minimale de la meule	REAL
$TC_TPG5	Largeur courante de la meule	REAL
$TC_TPG6	Vitesse de rotation maximale	REAL
$TC_TPG7	Vitesse périphérique maxi.	REAL
$TC_TPG8	Angle de la meule inclinée	REAL
$TC_TPG9	N° de paramètre pour calcul du rayon	INT

Bibliographie :
Description fonctionnelle Fonctions de base, Correction d'outil (W1)

Activation de la surveillance d'outil par sélection de l'outil

Pour les outils de rectification (types 400 à 499), par le biais d'un paramètre machine, vous pouvez faire en sorte que la surveillance d'outil soit activée de façon implicite au moment de la sélection de l'outil.

Une **seule** surveillance peut être active à la fois pour chaque broche.

Surveillance de la géométrie

Il s'agit de la surveillance du rayon courant et de la largeur courante de la meule.

La surveillance de la valeur de consigne de vitesse de rotation est effectuée de façon cyclique en tenant compte de la correction de vitesse de rotation de broche.

La vitesse de rotation limite est la valeur la plus faible qui découle de la comparaison entre la vitesse de rotation maximale et la vitesse de rotation calculée, à partir de la vitesse périphérique maximale de la meule et du rayon courant de la meule.

Programmation sans numéro T ni D

Un numéro **T** par défaut et un numéro **D** par défaut peuvent être réglés avec un paramètre machine.

Ils ne doivent plus être programmés et prennent effet après Power On/Reset.

Exemple : Usinage avec la même meule

Le paramètre machine vous permet de régler que l'outil actif sera conservé en cas de Reset (voir "Libre affectation des numéros D, numéro de tranchant (Page 429)").

Autres fonctions 14

14.1 Fonctions d'axe (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL)

Fonction

AXNAME est utilisé par exemple pour la création de cycles à caractère général lorsque les noms des axes ne sont pas connus.

AX est utilisé pour la programmation indirecte d'axes géométriques et d'axes synchrones. Le descripteur d'axe est alors enregistré dans une variable de type AXIS ou est fourni par une instruction telle qu'AXNAME ou SPI.

SPI est utilisé lorsque des fonctions d'axe sont programmées pour une broche (par exemple une broche synchrone).

AXTOSPI est utilisé pour convertir un descripteur d'axe en indice de broche (fonction inverse de SPI).

AXSTRING est utilisé pour convertir un descripteur d'axe (type de données AXIS) en chaîne de caractères (fonction inverse de AXNAME).

ISAXIS est utilisé dans des cycles à caractère général, pour vérifier l'existence d'un axe géométrique bien défini et pour éviter qu'un appel consécutif de $P_AXNX soit interrompu du fait d'une erreur.

MODAXVAL est utilisé pour déterminer la position modulo d'axes rotatifs modulo.

Syntaxe

```
AXNAME("chaîne de caractères")
AX[AXNAME("chaîne de caractères")]
SPI(n)
```

AXTOSPI(A) ou AXTOSPI(B) ou AXTOSPI(C)

```
AXSTRING( SPI(n) )
ISAXIS(<numéro d'axe géométrique>)
<Position modulo>=MODAXVAL(<axe>,<position d'axe>)
```

Signification

AXNAME	Convertit la chaîne de caractères d'entrée en un descripteur d'axe ; la chaîne de caractères d'entrée doit comporter un nom d'axe valide.
AX	Descripteur d'axe variable
SPI	Conversion du numéro de broche indiqué en descripteur d'axe ; le paramètre doit être un numéro de broche valide.
n	numéro de broche
AXTOSPI	Convertit un descripteur d'axe en un indice de broche de type Integer. AXTOSPI correspond à la fonction d'inversion pour SPI.
X, Y, Z	Descripteur d'axe de type AXIS comme variable ou comme constante
AXSTRING	La chaîne de caractères avec numéro de broche attribué.
ISAXIS	vérifie si l'axe géométrique indiqué existe
MODAXVAL	Détermination de la position modulo des axes rotatifs modulo, qui correspond au reste modulo par rapport à la zone modulo paramétrée (0 à 360 degrés en réglage par défaut ; les paramètres machine MD30340 MODULO_RANGE_START et MD30330 $MA_MODULO_RANGE permettent de modifier le début et la taille de la zone modulo).

Remarque

Extensions SPI

La fonction d'axe SPI(n) est également utilisable pour la lecture et l'écriture de composantes de frame. Il est ainsi possible, par exemple, d'écrire des frames avec la syntaxe `$P_PFRAME[SPI(1),TR]=2.22`.

La programmation supplémentaire de positions d'axe avec l'adresse `AX[SPI(1)]=<position d'axe>` permet de déplacer un axe, à condition que la broche soit en mode de positionnement ou en mode axe.

Exemples

Exemple 1 : AXNAME, AX, ISAXIS

```
Code de programme                          Commentaire
OVRA[AXNAME("axe transversal")]=10          ; Correction d'axe transversal
AX[AXNAME("axe transversal")]=50.2          ; Position de fin de l'axe transversal
OVRA[S1]=70                                ; Correction de la broche 1
AX[SPI(1)]=180                             ; Position de fin de la broche 1
IF ISAXIS(1) == FALSE GOTOF SUITE          ; Abscisse existante ?
AX[$P_AXN1]=100                            ; Déplacement de l'abscisse
SUITE :
```

Exemple 2 : AXSTRING

En cas de programmation avec AXSTRING[SPI(n)], ce n'est plus l'indice de l'axe auquel la broche est affectée qui est sorti en tant que numéro de broche, mais la chaîne de caractères "Sn".

Code de programme	Commentaire
AXSTRING[SPI(2)]	; Sortie de la chaîne de caractères "S2".

Exemple 3 : MODAXVAL

La position modulo de l'axe rotatif modulo A doit être déterminée.

Le calcul part de la position d'axe 372.55.

La zone modulo paramétrée s'étend de 0 à 360 degrés :

MD30340 MODULO_RANGE_START = 0

MD30330 $MA_MODULO_RANGE = 360

Code de programme	Commentaire
R10=MODAXVAL(A,372.55)	; Position modulo calculée R10 = 12.55.

Exemple 4 : MODAXVAL

Si le descripteur d'axe programmé ne se rapporte pas à un axe rotatif modulo, la valeur à convertir (<position d'axe>) est retournée telle quelle.

Code de programme	Commentaire
R11=MODAXVAL(X,372.55)	; X est l'axe linéaire ; R11 = 372.55.

14.2 Axes géométriques permutables (GEOAX)

Fonction

La fonction "Axes géométriques permutables" permet de modifier, à partir du programme pièce, le groupe d'axes géométriques qui a été configuré dans les paramètres machine. Un axe de canal défini comme axe supplémentaire synchrone peut remplacer un axe géométrique quelconque.

Syntaxe

```
GEOAX(<n>,<axe de canal>,<n>,<axe de canal>,<n>,<axe de canal>)
GEOAX()
```

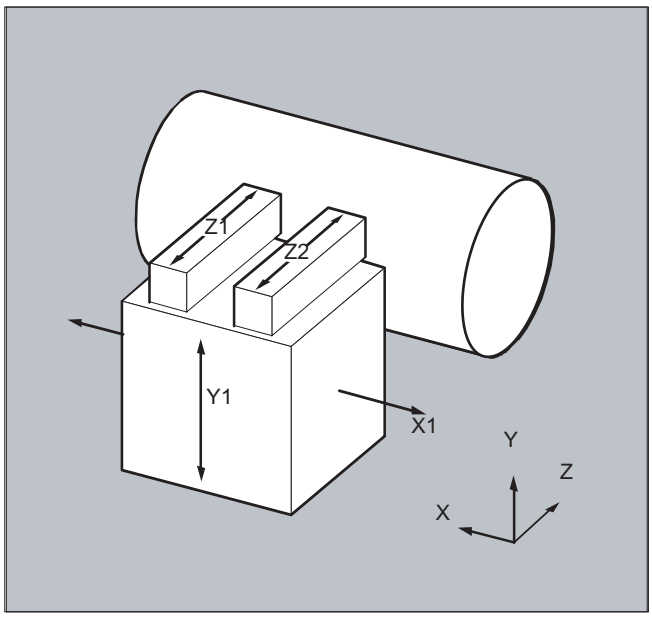
Signification

`GEOAX(...)`	Instruction de permutation des axes géométriques **Remarque :** `GEOAX()` sans indication de paramètre appelle la configuration de base des axes géométriques.
`<n>`	Ce paramètre indique le numéro de l'axe géométrique auquel l'axe de canal dont l'indication suit doit être affecté. Plage de valeurs 1, 2 ou 3 : **Remarque :** **<n>=**0 permet de supprimer du groupe d'axes géométriques l'axe de canal dont l'indication suit, sans contrepartie.
`<Axe de canal>`	Ce paramètre indique le nom de l'axe de canal qui doit être intégré dans le groupe d'axes géométriques.

Exemples

Exemple 1 : Deux axes permutés de façon alternée comme axe géométrique

Un chariot porte-outil peut être déplacé par le biais des axes de canal X1, Y1, Z1, Z2 :



Les axes géométriques sont configurés de sorte que d'abord, après la mise sous tension, Z1 soit actif en tant que 3e axe géométrique avec le nom d'axe géométrique "Z" et qu'il forme le groupe d'axes géométriques avec X1 et Y1.

Dans le programme pièce, les axes Z1 et Z2 doivent ensuite intervenir de façon alternée comme axe géométrique Z :

```
Code de programme     Commentaire
...
N100 GEOAX (3,Z2)     ; Axe de canal Z2 intervenant comme 3e axe géométrique (Z).
N110 G1 ...
N120 GEOAX(3,Z1)      ; Axe de canal Z1 intervenant comme 3e axe géométrique (Z).
...
```

Exemple 2 : Permutation des axes géométriques en présence de 6 axes de canal

Une machine comporte 6 axes de canal désignés par XX, YY, ZZ, U ,V ,W.

Le préréglage de la configuration des axes géométriques effectué dans les paramètres machine est le suivant :

axe de canal XX = 1er axe géométrique (axe X)

axe de canal YY = 2ème axe géométrique (axe Y)

axe de canal ZZ = 3ème axe géométrique (axe Z)

Code de programme	**Commentaire**
N10 GEOAX()	; La configuration de base des axes géométriques est activée.
N20 G0 X0 Y0 Z0 U0 V0 W0	; Tous les axes accostent la position 0 en vitesse rapide.
N30 GEOAX(1,U,2,V,3,W)	; L'axe de canal U devient premier axe géométrique (X), l'axe de canal V second (Y), l'axe de canal W troisième (Z).
N40 GEOAX(1,XX,3,ZZ)	; L'axe de canal XX devient le premier axe géométrique (X), ZZ le troisième (Z). L'axe de canal V reste second axe géométrique (Y).
N50 G17 G2 X20 I10 F1000	; Cercle complet dans le plan X/Y. Déplacement des axes de canal XX et V.
N60 GEOAX(2,W)	; L'axe de canal W devient second axe géométrique (Y).
N80 G17 G2 X20 I10 F1000	; Cercle complet dans le plan X/Y. Déplacement des axes de canal XX et W.
N90 GEOAX()	; Réinitialisation à l'état de base.
N100 GEOAX(1,U,2,V,3,W)	; L'axe de canal U devient premier axe géométrique (X), l'axe de canal V second (Y), l'axe de canal W troisième (Z).
N110 G1 X10 Y10 Z10 XX=25	; Les axes de canal U, V, W se déplacent respectivement sur la position 10. L'axe XX, en tant qu'axe supplémentaire, se déplace sur la position 25.
N120 GEOAX(0,V)	; V est supprimé du groupe d'axes géométriques. U et W restent premier (X) et troisième axe géométrique (Z). Le deuxième axe géométrique (Y) reste non affecté.
N130 GEOAX(1,U,2,V,3,W)	; L'axe de canal U reste premier axe géométrique (X), l'axe de canal V second (Y) et l'axe de canal W troisième (Z).
N140 GEOAX(3,V)	; V devient troisième axe géométrique (Z), écrasant W qui est ainsi extrait du groupe d'axes géométriques. Le deuxième axe géométrique (Y) n'est toujours pas affecté.

Remarque

Configuration d'axes

La correspondance entre les axes géométriques, les axes supplémentaires, les axes de canal et les axes machine, ainsi que la définition des noms des différents types d'axe, est définie par les paramètres machine suivants :

MD20050 $MC_AXCONF_GEOAX_ASIGN_TAB (affectation de l'axe géométrique à

l'axe de canal)

MD20060 $MC_AXCONF_GEOAX_NAME_TAB (nom d'axe géométrique dans le canal)

MD20070 $MC_AXCONF_MACHAX_USED (numéro d'axe machine valable dans le canal)

MD20080 $MC_AXCONF_CHANAX_NAME_TAB (nom d'axe de canal dans le canal)

MD10000 $MN_AXCONF_MACHAX_NAME_TAB (nom d'axe machine)

MD35000 $MA_SPIND_ASSIGN_TO_MACHAX (affectation de la broche à l'axe machine)

Bibliographie :
Description fonctionnelle Fonctions de base ; axes, systèmes de coordonnées, frames (K2)

Restrictions

- La permutation des axes géométriques n'est pas possible dans les cas suivants :
  - Transformation active
  - Interpolation de type spline active
  - Correction de rayon d'outil active
  - Correction d'outil fine active
- Quand l'axe géométrique et l'axe de canal portent le même nom, la permutation de l'axe géométrique est impossible.
- Aucun axe concerné par la permutation ne doit être concerné par une action pouvant durer au-delà des limites de bloc, tel que cela peut être le cas par exemple pour les axes de positionnement de type A et les axes asservis.
- L'instruction `GEOAX` permet uniquement de remplacer des axes géométriques qui existent déjà au moment de la mise sous tension (il est donc impossible d'en définir de nouveaux).
- Pendant la préparation de la table de contour (`CONTPRON`, `CONTDCON`), une permutation d'axe avec `GEOAX` entraîne une alarme.

Conditions marginales

Etat d'un axe après le remplacement

Un axe remplacé par commutation dans le groupe d'axes géométriques est programmable en tant qu'axe supplémentaire au moyen de son nom d'axe de canal après la commutation.

Frames, zones de protection, limitations de la zone de travail

La permutation des axes géométriques supprime tous les frames, les zones de protection et les limitations de la zone de travail.

Coordonnées polaires

Une permutation des axes géométriques avec `GEOAX` met les coordonnées polaires modales à la valeur 0 de manière analogue à un changement de plan avec `G17`-`G19`.

DRF, DO

Un éventuel décalage par manivelle (DRF) ou un décalage d'origine externe (DO) reste actif après la permutation.

Configuration de base des axes géométriques

L'instruction `GEOAX()` appelle la configuration de base du groupe d'axes géométriques.

Après un POWER ON et en cas de commutation en mode de fonctionnement "Prise de référence", on retourne automatiquement à la configuration de base.

Correction de longueur d'outil

Une correction de longueur d'outil active reste active après la permutation. Pour les axes géométriques qui viennent d'être intégrés ou dont la position a été permutée, on considère cependant que le déplacement correspondant n'a pas encore été exécuté. Pour la première instruction de déplacement de ces axes géométriques, le déplacement résultant est donc égal à la somme de la correction de longueur d'outil et du déplacement programmé.

Les axes géométriques qui gardent leur position dans le groupe d'axes après la permutation conservent également leur état relatif à la correction de longueur d'outil.

Configuration d'axe géométrique en cas de transformation active

La configuration d'axe géométrique valable dans une transformation active (définie dans les paramètres machine) ne peut pas être modifiée avec la fonction "Axes géométriques permutables".

S'il est nécessaire de modifier la configuration d'axe géométrique en liaison avec des transformations, ceci est uniquement possible au moyen d'une transformation supplémentaire.

L'activation d'une transformation efface une configuration d'axe géométrique modifiée avec `GEOAX`.

Si les réglages des paramètres machine relatifs à la transformation et à la permutation des axes géométriques sont contradictoires, ce sont les réglages de la transformation qui ont priorité.

Exemple :

Une transformation est active Conformément aux paramètres machine, la transformation doit être conservée après un RESET, mais la configuration de base des axes géométriques doit être rétablie après le RESET. Dans ce cas, la configuration d'axe géométrique qui a été définie avec la transformation est conservée.

14.3 Conteneur d'axes (AXCTSWE, AXCTSWED)

Fonction

Dans le cas des machines à transfert rotatif/multibroches, les axes supportant la pièce se déplacent d'une unité d'usinage à la suivante. Comme les unités d'usinage sont pilotées par différents canaux NCU, les axes supportant la pièce doivent être réaffectés dynamiquement au canal NCU correspondant en cas de changement de poste d'usinage/position. Les conteneurs d'axes ont été prévus à cet effet.

A un instant donné, un seul axe/broche d'ablocage de pièce est actif au niveau de l'unité d'usinage locale. Le conteneur d'axes regroupe tous les axes/broches d'ablocage parmi lesquels seulement un axe/broche est activé pour l'unité d'usinage concernée.

Le remplacement des axes définis dans un conteneur d'axes a lieu par décalage des inscriptions dans le conteneur ("rotation du conteneur d'axes") avec un pas qui peut être défini par une donnée de réglage (nombre de ports).

Depuis le programme pièce, l'appel d'une rotation du conteneur d'axes s'effectue avec l'instruction `AXCTSWE` ou `AXCTSWED`.

Syntaxe

```
AXCTSWE(<conteneur d'axes>)
AXCTSWED(<conteneur d'axes>)
```

Signification

`AXCTSWE`
Instruction de rotation d'un conteneur d'axes

La rotation du conteneur d'axes avec le pas spécifique au conteneur, qui est rangé dans la donnée de réglage SD41700 $SN_AXCT_SWWIDTH[<numéro de conteneur>], a lieu lorsque la commande a reçu les validations de tous les canaux possédant des axes dans le conteneur.

`AXCTSWED`
Instruction de rotation d'un conteneur d'axes sous le seul effet du canal actif (variante de l'instruction pour la mise en service !)

Remarque :
Les axes indiqués dans le conteneur ne sont libérés que si tous les autres canaux, qui possèdent des axes dans le conteneur, sont à l'état Reset.

`<Conteneur d'axes>`
Descripteur du conteneur d'axes pour lequel la rotation doit être exécutée.

Les indications possibles sont :

`CT<numéro de conteneur>`
Le numéro du conteneur d'axes est attaché à la combinaison de lettres CT.

Exemple : `CT3`

`<Nom de conteneur>`
Nom individuel du conteneur d'axes, réglé avec MD12750 $MN_AXCT_NAME_TAB.

Exemple : `A_CONT3`

Autres informations

Conteneur d'axes

Un conteneur d'axes peut contenir :

- des axes locaux et/ou
- Axes Link

Les conteneurs d'axes à axes Link constituent une fonctionnalité commune à toutes les NCUs (globale) qui est coordonnée par la commande. Un conteneur d'axes peut également gérer uniquement des axes locaux.

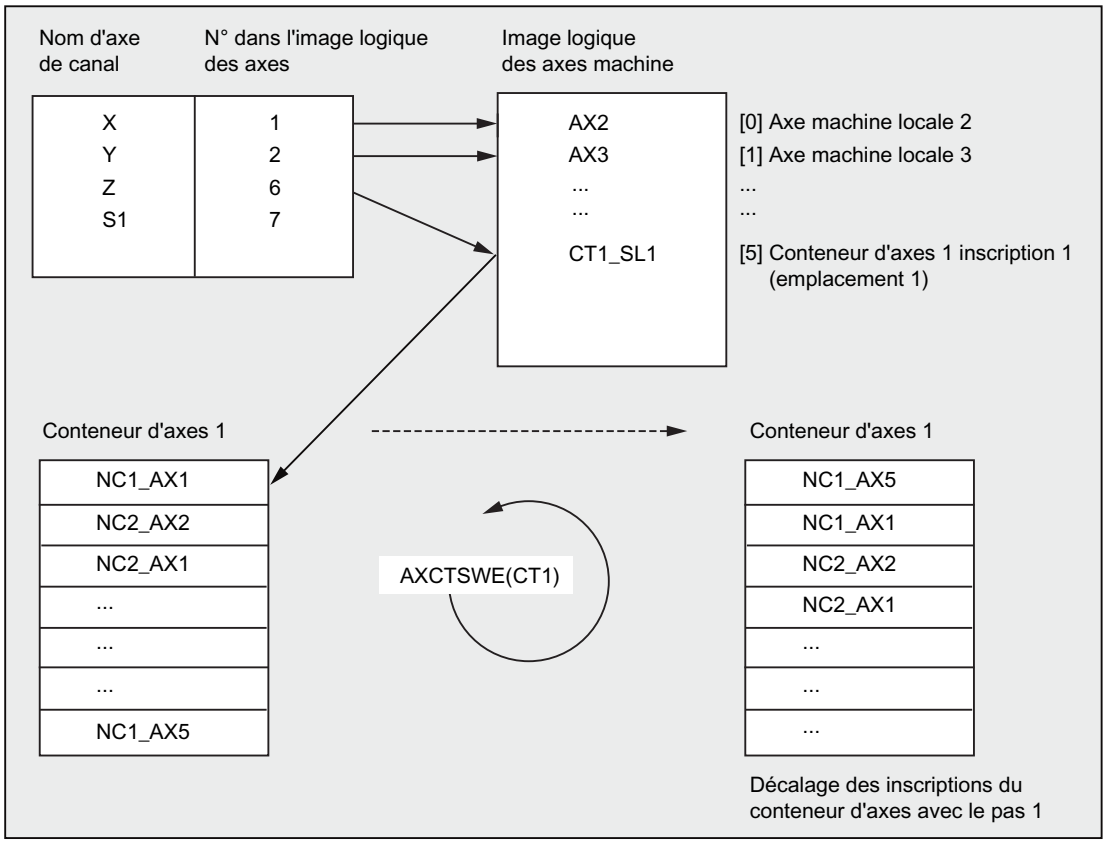
Bibliographie :
Pour des informations détaillées sur la configuration des conteneurs d'axes, voir :
Description fonctionnelle Fonctions d'extension ; Plusieurs panneaux de commande sur plusieurs NCU, Systèmes décentralisés (B3)

Critères de libération

AXCTSWE( )

Chaque canal, dont les axes sont inscrits dans le conteneur indiqué, fournit la validation pour une rotation conteneur (enable), lorsque l'usinage de la position/du poste correspondant est terminé. La rotation du conteneur d'axes avec le pas spécifique au conteneur, qui est rangé dans la donnée de réglage SD41700 $SN_AXCT_SWWIDTH[<numéro de conteneur>], a lieu lorsque la commande a reçu les validations de **tous** les canaux possédant des axes dans le conteneur.

Exemple :



L'axe de canal Z se voit affecté l'axe AX5 de NCU1 à la place de l'axe AX1 de NCU1 après rotation du conteneur d'axes de 1.

AXCTSWED ()

La variante d'instruction `AXCTSWED()` peut être utilisée pour simplifier la mise en service. Le conteneur d'axes tourne du pas qui lui est spécifique et qui est rangé dans la donnée de réglage SD41700 $SN_AXCT_SWWIDTH[<numéro de conteneur>], sous le seul effet du canal actif. Cette instruction ne peut être utilisée que si tous les autres canaux, qui possèdent des axes dans le conteneur, sont à l'état **Reset**.

Prise d'effet

La nouvelle affectation des axes après une rotation de conteneur d'axes concerne toutes les NCU dont les canaux accèdent au conteneur d'axes permuté par intermédiaire de l'image logique des axes machine.

Rotation du conteneur d'axes avec GET/GETD implicite

À la validation d'une rotation du conteneur d'axes, tous les axes du conteneur d'axes affectés au canal sont affectés au canal par le biais de GET ou de GETD. Une restitution des axes ne sera permise à nouveau qu'après la rotation du conteneur d'axes.

Remarque

Ce comportement est réglable avec un paramètre machine. Tenez compte des indications du constructeur de la machine.

Remarque

La rotation du conteneur d'axes avec GET/GETD implicites ne peut **pas** être utilisée pour un axe présentant l'état d'un axe dédié à l'exécution du programme (un axe AP par exemple), car cet axe devrait alors abandonner cet état pour la rotation du conteneur d'axes.

14.4 Attente de la position d'axe définitive (WAITENC)

Fonction

L'instruction de langage WAITENC permet d'attendre dans le programme CN que les positions d'axes synchronisées ou restaurées pour les axes configurés avec MD34800 $MA_WAIT_ENC_VALID = 1 soient disponibles.

Une interruption peut avoir lieu en mode d'attente, par ex. par le démarrage d'un ASUP ou par le changement de mode de fonctionnement en JOG. Le mode d'attente est éventuellement repris avec la poursuite du programme.

Remarque

Le mode d'attente est affiché dans l'interface utilisateur au moyen de l'état d'arrêt "Attendre système de mesure".

Syntaxe

WAITENC peut être programmé dans la section d'un programme CN quelconque.

La programmation doit être réalisée dans un bloc spécifique :

```
...
WAITENC
...
```

Exemple

WAITENC est par ex. utilisé dans le programme d'application commandé par événement .../_N_CMA_DIR/_N_PROG_EVENT_SPF ainsi que le montre l'exemple d'application suivant.

Exemple d'application :retrait d'outil après POWER OFF avec transformation d'orientation

Un usinage avec orientation d'outil a été interrompu par une coupure de courant.
Le programme d'application commandé par événement .../_N_CMA_DIR/_N_PROG_EVENT_SPF est appelé lors du démarrage s'en suivant.

Dans le programme d'application commandé par événement, WAITENC permet d'attendre des positions d'axe synchronisées ou restaurées afin de pouvoir calculer par la suite une grille pour orienter le SCP dans la direction de l'outil.

```
Code de programme                    Commentaire
...
IF $P_PROG_EVENT == 4                ; Démarrage.
    IF $P_TRAFO <> 0                 ; Transformation a été activée.
        WAITENC                      ; Attente des positions valables des axes d'orientation.
        TOROTZ                       ; Rotation de l'axe Z du SCP en direction de l'axe d'outil.
    ENDIF
    M17
ENDIF
...
```

En mode JOG, l'outil peut par la suite, par un déplacement de retrait, être dégagé dans la direction de l'axe d'outil.

14.5 Contrôler le langage CN existant (STRINGIS)

Fonction

La fonction `STRINGIS(...)` permet de contrôler si la chaîne de caractères indiquée est disponible comme élément de langage de programmation CN dans l'étendue actuelle des langages.

Définition

```
INT STRINGIS(STRING <nom>)
```

Syntaxe

```
STRINGIS(<nom>)
```

Signification

`STRINGIS` :	Fonction avec valeur de retour
`<nom>` :	Nom de l'élément à contrôler du langage de programmation CN
Valeur de retour	Le format de la valeur de retour est yxx (décimal).

Eléments du langage de programmation CN

Les éléments suivants du langage de programmation CN peuvent être contrôlés :

- Codes G de tous les groupes de fonctions G existants, par ex. `G0`, `INVCW`, `POLY`, `ROT`, `KONT`, `SOFT`, `CUT2D`, `CDON`, `RMB`, `SPATH`
- Adresses DIN ou CN comme par ex. `ADIS`, `RNDM`, `SPN`, `SR`, `MEAS`
- Fonctions par ex. `TANG(...)` ou `GETMDACT`
- Procédures par ex. `SBLOF`.
- Mots-clés par ex. `ACN`, `DEFINE` ou `SETMS`
- Données système par ex. paramètres machine `$M`... , données de réglage `$S`... ou données optionnelles `$O`...
- Variable système `$A`... , `$V`... , `$P`...
- Paramètre de calcul `R`...
- Noms de cycles activés
- Variables GUD et LUD
- Noms de macros
- Noms d'étiquettes

Valeur de retour

La valeur de retour est uniquement significative dans les 3 premiers chiffres en notation décimale. Le format de la valeur de retour est yxx, y étant alors = information de base et xx = information détaillée.

Valeur de retour	Signification	
000	Le 'nom' de la chaîne de caractères n'est pas connu dans le présent système [1)]	
100	Le 'nom' de la chaîne de caractères est un élément du langage de programmation CN, il n'est toutefois actuellement pas programmable (option/fonction est inactive)	
2xx	Le 'nom' de la chaîne de caractères est un élément programmable du langage de programmation CN (option/fonction est active). L'information détaillée xx contient d'autres informations sur le type de l'élément :	
	xx	**Signification**
	01	Adresse DIN ou CN [2)]
	02	Code G (par ex. `G04`, `INVCW`)
	03	Fonction avec valeur de retour
	04	Fonction sans valeur de retour
	05	Mot-clé (par ex. `DEFINE`)
	06	Paramètre machine (`$M`...) , donnée de réglage (`$S`...) ou donnée optionnelle (`$O`...)
	07	Paramètres système, par ex. variable système (`$`...) ou paramètre de calcul (`R`...)
	08	Cycle (le cycle doit être chargé dans NCK et les programmes de cycle doivent être actifs [3)])
	09	Variable GUD (la variable GUD doit correspondre à celle définie dans les fichiers de définition GUD et les variables GUD doivent être activées)
	10	Nom de macro (le macro doit correspondre à celui défini dans les fichiers de définition Macro et les macros doivent être activés) [4)]
	11	Variable LUD du programme pièce actuel
	12	Code G ISO (mode de langage ISO doit être actif)
400	Le 'nom' de la chaîne de caractères est une adresse CN n'ayant pas été reconnue comme xx == 01 ou xx == 10 et n'étant pas G ni R [2)]	
y00	Pas d'affectation spécifique possible	

1) En fonction de la commande, seul un sous-ensemble des instructions de langage CN Siemens est éventuellement connu, par ex. SINUMERIK 802D sl. Sur ces commandes, la valeur 0 est retournée pour des chaînes de caractères n'étant pas principalement des instructions de langage CN Siemens. Ce comportement peut être modifié au moyen de MD10711 $MN_NC_LANGUAGE_CONFIGURATION. Dans le cas de MD10711 = 1, la valeur 100 est toujours retournée pour des instructions de langage CN Siemens.

2) Les lettre suivantes sont des adresses CN : A, B, C, E, I, J, K, Q, U, V, W, X, Y, Z. Ces adresses CN peuvent également être programmées avec une extension d'adresse. L'extension d'adresse peut être indiquée lors du contrôle avec STRINGIS. Exemple : `201 == STRINGIS("A1")`.
Les lettres : D, F, H, L, M, N, O, P, S, T sont des adresses CN ou des fonctions auxiliaires utilisées de manière définie par l'utilisateur. La valeur 400 est toujours retournée pour ces adresses. Exemple : 400 == STRINGIS("D"). Ces adresses CN ne peuvent pas être indiquées avec extension d'adresse lors du contrôle avec STRINGIS.
Exemple : `000 == STRINGIS("M02")`, mais `400 == STRINGIS("M")`.

3) Les noms de paramètres de cycles ne peuvent pas être contrôlés avec STRINGIS.
4) Des adresses définies comme macro, par ex. G, H, M, L, sont identifiées comme macro

Exemples

Dans les exemples ci-dessous, il est supposé que les éléments de langage CN indiqués comme chaîne de caractères, sauf indication contraire, sont principalement programmables dans la commande.

1. La chaîne de caractères "T" est définie comme fonction auxiliaire :

   ```
   400 == STRINGIS("T")
   000 == STRINGIS ("T3")
   ```

2. La chaîne de caractères "X" est définie comme axe :

   ```
   201 == STRINGIS("X")
   201 == STRINGIS("X1")
   ```

3. La chaîne de caractères "A2" est définie comme adresse CN avec extension :

   ```
   201 == STRINGIS("A")
   201 == STRINGIS("A2")
   ```

4. La chaîne de caractères "INVCW" est définie comme code G nommé :

   ```
   202 == STRINGIS("INVCW")
   ```

5. La chaîne de caractères "$MC_GCODE_RESET_VALUES" est définie comme paramètre machine :

   ```
   206 == STRINGIS("$MC_GCODE_RESET_VALUES")
   ```

6. La chaîne de caractères "GETMDACT" est une fonction de langage CN :

   ```
   203 == STRINGIS("GETMDACT ")
   ```

7. La chaîne de caractères "DEFINE" est un mot-clé :

   ```
   205 == STRINGIS("DEFINE")
   ```

8. La chaîne de caractères "$TC_DP3" est un paramètre système (composantes de longueur d'outil) :

   ```
   207 == STRINGIS("$TC_DP3")
   ```

9. La chaîne de caractères "$TC_TP4" est un paramètre système (taille d'outil) :

   ```
   207 == STRINGIS("$TC_TP4")
   ```

10. La chaîne de caractèrres "$TC_MPP4" est un paramètre système (état de l'emplacement d'outil) :

    – La gestion de magasin d'outils est active : `207 == STRINGIS("$TC_MPP4")` ;

    – La gestion de magasin d'outils n'est pas active : `000 == STRINGIS("$TC_MPP4")`

    Voir également ci-dessous le paragraphe : Gestion de magasin d'outils.

11. La chaîne de caractères "MACHINERY_NAME" est définie comme variable GUD :

    ```
    209 == STRINGIS("MACHINERY_NAME")
    ```

12. La chaîne de caractères "LONGMACRO" est définie comme macro :

    ```
    210 == STRINGIS("LONGMACRO")
    ```

13. La chaîne de caractères "MYVAR" est définie comme variable LUD :

    ```
    211 == STRINGIS("MYVAR")
    ```

14. La chaîne de caractères "XYZ" n'est pas une instruction connue dans NCK, variable GUD, nom de macro ni nom de cycle :

    ```
    000 == STRINGIS("XYZ")
    ```

Gestion de magasin d'outils

Si la fonction de gestion d'outils n'est pas active, STRINGIS fournit pour les paramètres système de la gestion de magasin d'outils, indépendamment du paramètre machine

- MD10711 $MN_NC_LANGUAGE_CONFIGURATION

toujours la valeur 000.

Mode ISO

Si la fonction "Mode ISO" est active :

- MD18800 $MN_MM_EXTERN_LANGUAGE (activation de langages CN externes)
- MD10880 $MN_ MM_EXTERN_CNC_SYSTEM (système de commande devant être adapté)

TRINGIS contrôle la chaîne de caractères indiquée tout d'abord à titre de code G SINUMERIK. Si la chaîne de caractères n'est pas un code G SINUMERIK, elle est par la suite contrôlée à titre de code G ISO.

Des commutations programmées (`G290` (mode SINUMERIK), `G291` (mode ISO)) n'ont aucun effet sur STRINGIS.

Exemple

Les importants paramètres machine pour la fonction STRINGIS(...) ont les valeurs suivantes :

- MD10711 $MN_NC_LANGUAGE_CONFIGURATION = 2 (seules les instructions de langage CN dont les options sont configurées sont considérées comme étant reconnues)
- MD19410 $ON_TRAFO_TYPE_MASK = 'H0' (option : transformations)
- MD10700 $MN_PREPROCESSING_LEVEL='H43' (conditionnement pour cycles actif)

Le programme exemplaire ci-dessous est exécuté sans messsage d'erreur :

```
Code de programme                          Commentaire
N1 R1=STRINGIS("TRACYL")                   ;   R1 == 0, étant donné que TRACYL est constatée
                                           ;   comme ''étant non connue" en raison de
                                           ;   l'absence de l'option de transformation
N2 IF STRINGIS("TRACYL") == 204            ;
N3 TRACYL(1,2,3)                           ;   L'impasse est faite sur N3
N4 ELSE
N5 G00                                     ;   et N5 est exécuté à la place
N6 ENDIF
N7 M30
```

14.6 Lecture de l'appel de la fonction ISVAR et des paramètres machine d'indice de tableau

Fonction

L'instruction ISVAR est une fonction au sens du langage CN avec :

- Valeur de la fonction du type BOOL
- un paramètre de transmission du type STRING

L'instruction ISVAR fournit TRUE, lorsque le paramètre de transfert contient une variable connue de la CN (paramètre machine, donnée de réglage, variable système ou variable générale telle qu'une donnée utilisateur GUD).

Syntaxe

```
ISVAR(<descripteur de variable>)
ISVAR(<descripteur>,[<valeur>,<valeur>])
```

Signification

`<Descripteur de variable>`	Le paramètre de transmission du type chaîne de caractères peut être adimensionnel, monodimensionnel ou bidimensionnel
`<Descripteur>`	Descripteur avec une variable connue de la CN, avec ou sans indice de tableau comme paramètre machine, donnée de réglage, variable système ou variable générale. **Extension :** dans le cas de paramètres machine généraux et spécifiques à un canal, le premier élément de tableau est également lu même si l'indice est manquant
`<Valeur>`	Valeur de la fonction du type BOOL

Vérifications

Conformément au paramètre de transmission, les points suivants sont vérifiés :

- il y a un descripteur
- il s'agit d'un champ monodimensionnel ou bidimensionnel
- un indice de tableau est admis

Si tous ces points sont affirmatifs, l'état TRUE est indiqué en retour. Si un seul des points est négatif ou s'il existe une erreur de syntaxe, c'est l'état FALSE qui est retourné. Les variables axiales sont acceptées en tant qu'indices pour les noms d'axe, mais elles ne sont pas vérifiées plus précisément.

Extension : Lecture du tableau des paramètres machine et des données de réglage sans indice.

L'alarme 12400 "Canal % 1 Bloc % 2 Champ % 3 Elément non disponible" **n'est plus** émise par les paramètres machine **généraux et spécifiques à un canal** si l'indice est manquant.

En outre, **l'indice d'axe au moins** doit être programmé pour les paramètres machine **spécifiques à un axe**. Dans le cas contraire, l'alarme 12400 est émise.

Exemple : Appel de la fonction ISVAR

```
Code de programme                              Commentaire
DEF INT VAR1
DEF BOOL IS_VAR=FALSE                          ; Le paramètre de transmission est une variable
                                                 générale.
N10 IS_VAR=ISVAR("VAR1")                       ; Dans ce cas, IS_VAR est TRUE.
DEF REAL VARARRAY[10,10]
DEF BOOL IS_VAR=FALSE                          ; Différentes variantes de syntaxe
N20 IS_VAR=ISVAR("VARARRAY[,]")                ; IS_VAR est TRUE avec un
                                                 tableau à deux dimensions.
N30 IS_VAR=ISVAR("VARARRAY")                   ; IS_VAR est TRUE ; la variable existe.
N40 IS_VAR=ISVAR("VARARRAY[8,11]")             ; IS_VAR est FALSE ; l'indice de tableau est illicite.
N50 IS_VAR=ISVAR("VARARRAY[8,8")               ; IS_VAR est FALSE ; erreur de syntaxe pour
                                                 "]" manquant.
N60 IS_VAR=ISVAR("VARARRAY[,8]")               ; IS_VAR est TRUE ; l'indice de tableau est autorisé.
N70 IS_VAR=ISVAR("VARARRAY[8,]")               ; IS_VAR est TRUE.
DEF BOOL IS_VAR=FALSE                          ; Le paramètre de transmission est un paramètre
                                                 machine.
N100 IS_VAR=ISVAR("$MC_GCODE_RESET_VALUES[1]"  ; IS_VAR est TRUE.
DEF BOOL IS_VAR=FALSE                          ; Le paramètre de transmission est une variable
                                                 système.
N10 IS_VAR=ISVAR("$P_EP")                      ; Dans ce cas, IS_VAR est TRUE.
N10 IS_VAR=ISVAR("$P_EP[X]")                   ; Dans ce cas, IS_VAR est TRUE.
```

Exemple : Lecture du tableau des paramètres machine avec et sans indice.

Le premier élément est lu pour

```
R1=$MC_EXTERN_GCODE_RESET_VALUES
```

Cela correspond, comme précédemment, à

```
R1=$MC_EXTERN_GCODE_RESET_VALUES[0]
```

ou est lu le premier élément

```
R1=$MA_POSTCTRL_GAIN[X1]
```

Cela correspond, comme précédemment, à

```
R1=$MA_POSTCTRL_GAIN[0, X1]
```

Est également lu le premier élément dans les actions synchrones pour

```
WHEN TRUE DO $R1 = $MC_EXTERN_GCODE_RESET_VALUES
```

Cela correspond, comme précédemment, à

```
WHEN TRUE DO $R1 = $MC_EXTERN_GCODE_RESET_VALUES[0]
```

qui **n'était pas lu** jusqu'à présent, avec génération de l'alarme 12400.

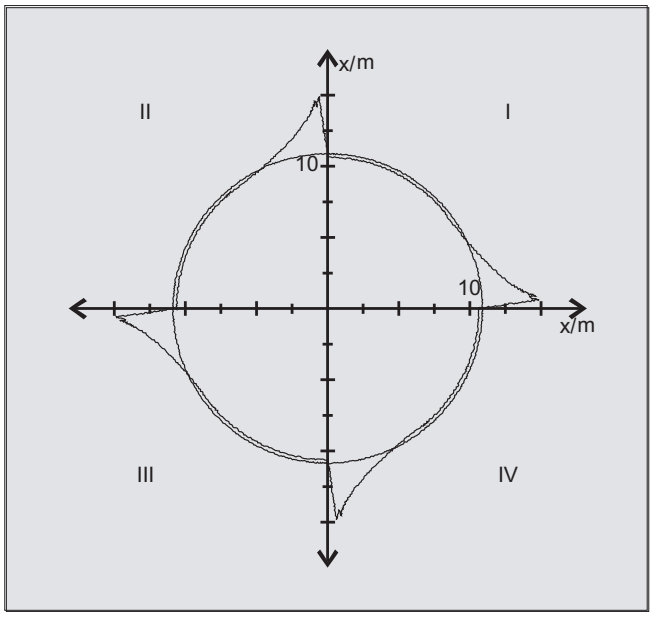
L'alarme 12400 continue à être émise pour

```
R1=$MA_POSTCTRL_GAIN
```

14.7 Apprentissage de compensations (QECLRNON, QECLRNOF)

Fonction .Erreur! Signet non défini..

La compensation des défauts aux transitions entre quadrants (QFK) réduit les violations du contour qui risquent de se produire à l'inversion du sens du déplacement à cause des défauts de linéarité mécaniques (par ex. par friction jeu) ou la torsion. Les paramètres de compensation peuvent être optimisés par la commande au cours d'une phase d'apprentissage à l'aide d'un réseau neuronal, permettant de déterminer les caractéristiques de compensation déterminées automatiquement. L'apprentissage peut être effectué simultanément pour quatre axes au maximum.



Syntaxe

```
QECLRNON
QECLRNOF
```

Activation de la procédure d'apprentissage : QECLRNON

L'activation de la procédure d'apprentissage proprement dite s'effectue dans le programme CN avec l'instruction `QECLRNON` :

```
QECLRNON (X1, Y1, Z1, Q)
```

Les caractéristiques sont modifiées uniquement si cette instruction est active.

Désactivation de l'apprentissage : QECLRNOF

A la fin des déplacements d'apprentissage des axes souhaités, la procédure d'apprentissage se désactive simultanément pour tous les axes avec `QECLRNOF`.

Signification

`QECLRNON (axe.1,…4)`	Activation de la fonction "Apprentissage de la compensation des défauts aux transitions entre quadrants"
`QECLRNO`	Désactivation de la fonction "Apprentissage de la compensation des défauts aux transitions entre quadrants"
`QECLRN.SPF`	Cycle d'apprentissage
`QECDAT.MPF`	Programme CN modèle pour l'affectation des variables système et le paramétrage du cycle d'apprentissage
`QECTEST.MPF`	Programme CN modèle pour le test de circularité

Description

Les déplacements des axes nécessaires à l'apprentissage sont générés à l'aide d'un programme CN. Dans ce dernier, les déplacements d'apprentissage se présentent sous la forme d'un cycle d'apprentissage.

Premier apprentissage

Pour le premier apprentissage, à la mise en service, il existe sur la disquette qui contient le programme de base de l'AP, des programmes CN types pour s'initier aux déplacements d'apprentissage et à l'affectation des variables système QFK :

Réapprentissage

Le réapprentissage permet d'optimiser les courbes caractéristiques déjà apprises. Les données qui se trouvent déjà dans la mémoire utilisateur sont alors utilisées. Pour le réapprentissage, vous adaptez les programmes CN types à vos besoins.

Les paramètres du cycle d'apprentissage (par exemple QECLRN.SPF) doivent, le cas échéant, être modifiés pour le "réapprentissage" :

- régler "mode d'apprentissage" = 1
- réduire le cas échéant le "nombre de cycles d'apprentissage"
- le cas échéant, activer "l'apprentissage par zone" et fixer les limites de ces zones

14.8 Appel interactif des fenêtres depuis le programme pièce (MMC)

Fonction

L'instruction MMC permet de déclencher, à partir du programme pièce, l'affichage de boîtes de dialogue définies par l'utilisateur sur l'IHM.

L'aspect de ces boîtes de dialogue est défini par une configuration purement textuelle (fichier COM dans le répertoire des cycles), le logiciel système IHM restant inchangé.

Une boîte de dialogue définie par l'utilisateur ne peut être appelée simultanément dans différents canaux.

Syntaxe

```
MMC(CYCLES,PICTURE_ON,T_SK.COM,IMAGE,MGUD.DEF,IMAGE_3.AWB,TEST_1,A1"
,"S")
```

Signification

MMC	Appel interactif de boîtes de dialogue à partir du programme pièce pour affichage sur l'IHM.
CYCLES	Groupe fonctionnel dans lequel sont exécutés les dialogues utilisateur qui ont été configurés.
PICTURE_ON ou PICTURE_OFF	Instruction : activation ou désactivation boîte de dialogue
T_SK.COM	Fichier com : Nom du fichier de boîtes de dialogue (cycles utilisateur). L'aspect des boîtes de dialogue y est défini. Les variables utilisateur et/ou les commentaires peuvent être affichés dans la boîte de dialogue.
BOITE	Nom de la boîte de dialogue : la sélection des différentes boîtes de dialogue s'effectue par leurs noms.
MGUD.DEF	Fichier de définition de données utilisateur, dans lequel les variables sont lues/écrites.
BOITE_3.AWB	Fichier graphique
TEST_1	Durée d'affichage ou variable d'acquittement
A1	Variable de texte...",
"S"	Mode d'acquittement : synchrone, acquittement à l'aide de la touche logicielle "OK"

Bibliographie

Pour des informations détaillées sur la programmation de l'instruction MMC (avec des exemples de programmation), voir : Manuel de mise en service.

14.9 Temps d'exécution du programme / Compteur de pièces

14.9.1 Temps d'exécution du programme / Compteur de pièces (vue d'ensemble)

Des informations relatives au temps d'exécution et au nombre de pièces sont disponibles pour aider l'opérateur de la machine-outil.

Ces informations peuvent être traitées comme variables système dans le programme CN et/ou AP. En même temps, ces informations sont disponibles pour l'affichage sur l'interface utilisateur.

14.9.2 Temps d'exécution de programme

Fonction

Pour la surveillance de processus technologiques, la fonction "Temps d'exécution" met à disposition des compteurs chronométriques internes de la CN, qui peuvent être lus dans le programme pièce et dans les actions synchrones au moyen de variables système spécifiques à la CN et au canal.

Le déclencheur de la mesure de temps d'exécution ($AC_PROG_NET_TIME_TRIGGER) est la seule variable système de la fonction, qui est accessible en écriture. Il sert à opérer une mesure sélective de sections de programme, ce qui signifie que l'écriture du déclencheur dans le programme CN permet d'activer et de désactiver la mesure de temps.

Variable système	Signification	Activité
Spécifiques à la CN		
$AN_SETUP_TIME	Temps (en minutes) depuis le dernier démarrage de la commande avec les valeurs par défaut ("démarrage à froid") Remise à "0" automatique à chaque démarrage de la commande avec les valeurs par défaut.	• Activation permanente
$AN_POWERON_TIME	Temps en minutes depuis le dernier démarrage normal de la commande ("démarrage à chaud") Remise à "0" automatique à chaque démarrage normal de la commande.	

Variable système	Signification	Activité
Spécifiques au canal		
$AC_OPERATING_TIME	Temps global en secondes pour l'exécution de programmes CN en mode automatique Remise à "0" automatique de la valeur à chaque démarrage de la commande.	• Activation par le paramètre machine MD27860 • Uniquement en mode AUTO
$AC_CYCLE_TIME	Temps d'exécution du programme CN sélectionné en secondes Remise à "0" automatique de la valeur au démarrage d'un nouveau programme CN.	
$AC_CUTTING_TIME	Temps d'usinage en secondes Le temps d'exécution des axes à interpolation (au moins l'un d'entre eux est actif) est totalisé sans l'activation du rapide dans tous les programmes CN, entre le départ CN et la fin du programme / RESET de la CN. La mesure est suspendue pendant tout arrêt temporisé. Remise à "0" automatique de la valeur à chaque démarrage de la commande avec les valeurs par défaut.	
$AC_ACT_PROG_NET_TIME	Temps d'exécution actuel net en secondes du programme CN courant Remise à "0" automatique au démarrage d'un programme CN.	• Activation permanente • Uniquement en mode AUTO
$AC_OLD_PROG_NET_TIME	Temps d'exécution net du programme terminé correctement à l'instant avec `M30`, en secondes.	
$AC_OLD_PROG_NET_TIME_COUNT	Modifications sur $AC_OLD_PROG_NET_TIME Après un POWER ON, $AC_OLD_PROG_NET_TIME_COUNT est à "0". $AC_OLD_PROG_NET_TIME_COUNT est toujours incrémenté lorsque la commande a récrit $AC_OLD_PROG_NET_TIME.	

Variable système	Signification		Activité
$AC_PROG_NET_TIME_TRIGGER	Déclencheur de la mesure du temps d'exécution :		• Uniquement en mode AUTO
	0	Etat neutre Le déclencheur n'est pas actif.	
	1	Arrêter Arrêt de la mesure et copie de la valeur de $AC_ACT_PROG_NET_TIME dans $AC_OLD_PROG_NET_TIME. $AC_ACT_PROG_NET_TIME est mis à "0", puis continue.	
	2	Départ Lancement de la mesure et mise à "0" de $AC_ACT_PROG_NET_TIME. $AC_OLD_PROG_NET_TIME n'est pas modifié.	
	3	Arrêt Arrêt de la mesure. $AC_OLD_PROG_NET_TIME n'est pas modifié et $AC_ACT_PROG_NET_TIME reste constant jusqu'à la reprise.	
	4	Reprise Reprise de la mesure, c'est-à-dire qu'une mesure arrêtée précédemment reprend. $AC_ACT_PROG_NET_TIME continue. $AC_OLD_PROG_NET_TIME n'est pas modifié.	
Un POWER ON remet toutes les variables système à "0".			
Bibliographie : une description détaillée des variables système énumérées figure dans la : Description fonctionnelle Fonctions de base ; GMF, Canal, Mode de programme, Comportement au reset (K1), chapitre : Temps d'exécution de programme			

Remarque

Constructeur de la machine-outil

Les temporisations activables sont activées au moyen du paramètre machine MD27860 $MC_PROCESSTIMER_MODE.

Le comportement des mesures de temps actives pour certaines fonctions (par ex. GOTOS, correction = 0%, test d'avance actif, test de programme, ASUP, PROG_EVENT, …) est configuré au moyen des paramètres machine MD27850 $MC_PROG_NET_TIMER_MODE et MD27860 $MC_PROCESSTIMER_MODE.

Bibliographie :
Description fonctionnelle Fonctions de base ; GMF, Canal, Mode de programme, Comportement au reset (K1), chapitre : Temps d'exécution de programme

Remarque

Temps restant pour une pièce

Lorsqu'il s'agit de produire des pièces identiques dans la foulée, les valeurs des temporisations :

- temps d'usinage de la dernière pièce produite (voir $AC_OLD_PROG_NET_TIME)

et

- temps d'usinage actuel (voir $AC_ACT_PROG_NET_TIME)

permettent de déterminer le temps restant requis pour une pièce.

Le temps restant s'affiche sur l'interface utilisateur, en plus du temps d'usinage actuel.

IMPORTANT
Utilisation de STOPRE Les variables système $AC_OLD_PROG_NET_TIME et $AC_OLD_PROG_NET_TIME_COUNT n'entraînent pas d'arrêt implicite du prétraitement. En cas d'utilisation dans le programme pièce, ceci n'est pas critique lorsque la valeur des variables système provient de l'exécution précédente du programme. Par contre, si le déclencheur de la mesure du temps d'exécution ($AC_PROG_NET_TIME_TRIGGER) est écrit avec une fréquence élevée, modifiant ainsi fréquemment $AC_OLD_PROG_NET_TIME, il est recommandé d'utiliser un `STOPRE` explicite dans le programme pièce.

Contraintes

- **Recherche de bloc**

  Aucun temps d'exécution de programme n'est déterminé pendant une recherche de bloc.

- **REPOS**

  La durée d'un processus REPOS est ajoutée au temps d'usinage actuel ($AC_ACT_PROG_NET_TIME).

Exemples

Exemple 1 : Mesure du temps de "mySubProgrammA"

Code de programme

```
...
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
```

Après le traitement de la ligne `N80` par le programme, le temps d'exécution net de "mySubProgrammA" figure dans $AC_OLD_PROG_NET_TIME.

La valeur de $AC_OLD_PROG_NET_TIME :

- est conservée au-delà de `M30`,
- est actualisée après chaque exécution de boucle.

Exemple 2 : Mesure du temps de "mySubProgrammA" et de "mySubProgrammC"

Code de programme

```
...
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 Compteurs de pièces

Fonction

La fonction "Compteurs de pièces" met à disposition différents compteurs, qu'il est en particulier possible d'utiliser en interne à la commande pour le comptage de pièces.

Les compteurs sont disponibles sous forme de variables système spécifiques à un canal avec accès en écriture et lecture dans la plage de valeurs de 0 à 999 999 999.

Variable système	Signification
$AC_REQUIRED_PARTS	Nombre de pièces à fabriquer (nombre de pièces requis) Ce compteur permet de définir le nombre de pièces qui, lorsqu'il est atteint, entraîne la remise à "0" du nombre de pièces réel ($AC_ACTUAL_PARTS).
$AC_TOTAL_PARTS	Nombre total de pièces fabriquées (nombre total réel de pièces) Ce compteur indique le nombre total de pièces fabriquées à partir du début du comptage. Remise à "0" automatique de la valeur uniquement en cas de démarrage de la commande avec les valeurs par défaut.
$AC_ACTUAL_PARTS	Nombre de pièces fabriquées (nombre réel de pièces) Ce compteur enregistre le nombre total de pièces fabriquées depuis le début du comptage. Le compteur est automatiquement remis à "0" (à condition que $AC_REQUIRED_PARTS > 0), lorsque le nombre de pièces requis ($AC_REQUIRED_PARTS) est atteint.
$AC_SPECIAL_PARTS	Nombre de pièces comptées par l'utilisateur Ce compteur permet à l'utilisateur de définir son propre mode de comptage des pièces. Il est possible de définir l'émission d'une alarme lorsque le nombre de pièce requis est atteint ($AC_REQUIRED_PARTS). L'utilisateur doit assurer lui-même la remise à zéro du compteur.

Remarque

Chaque compteur de pièces est préréglé à sa valeur par défaut "0" au démarrage de la commande et son contenu peut être lu/écrit indépendamment de son mode d'activation.

Remarque

Des paramètres machine spécifiques au canal permettent de régler le mode d'activation des compteurs, l'instant de leur remise à zéro et l'algorithme de comptage.

Remarque

Comptage de pièces avec fonction M définie par l'utilisateur

Des paramètres machine permettent de définir le déclenchement des impulsions de comptage pour les différents compteurs de pièces au moyen de fonctions M définies par l'utilisateur à la place du déclenchement à la fin du programme `M2`/`M30`.

Bibliographie

Pour plus d'informations sur cette la fonction "Compteurs de pièces", référez-vous à la :

- Description fonctionnelle Fonctions de base ; GMF, Canal, Mode de programme, Comportement au reset (K1), chapitre : Compteurs de pièces

14.10 Alarmes (SETAL)

Fonction

Les alarmes peuvent être définies dans un programme CN. Elles sont affichées dans une fenêtre particulière de l'interface utilisateur. A une alarme correspond obligatoirement une réaction de la commande qui dépend de la catégorie à laquelle appartient l'alarme.

Bibliographie :
Pour de plus amples informations sur les réactions d'alarme, voir : Manuel de programmation.

Syntaxe

```
SETAL(<numéro d'alarme>)
SETAL(<numéro d'alarme>,<chaîne de caractères>)
```

Signification

`SETAL`	Mot clé pour la programmation d'une alarme. `SETAL` doit être programmé dans un bloc CN spécifique.
`<Numéro d'alarme>`	Variable de type INT contenant le numéro d'alarme. La plage admise pour les numéros d'alarmes est comprise entre 60000 et 69999, les numéros 60000 à 64999 étant réservés pour les cycles SIEMENS et les numéros 65000 à 69999 à la disposition de l'utilisateur.
`<Chaîne de caractères>`	Lors de la programmation d'alarmes de cyclique utilisateur, il est possible d'ajouter une chaîne de caractères comportant jusqu'à 4 paramètres. Dans ces paramètres, il est possible de définir des textes utilisateur variables. Les paramètres prédéfinis suivants sont également disponibles :

Paramètres	Signification
%1	numéro de canal
%2	numéro de bloc, étiquette
%3	Indice de texte pour alarmes de cycles
%4	Paramètre d'alarme supplémentaire

Remarque

Les textes d'alarme doivent être configurés dans l'interface utilisateur.

Exemple

Code de programme	Commentaire
...	
N100 SETAL (65000)	; Activer l'alarme n° 65000
...	

Programmes de chariotage personnalisés 15

15.1 Fonctions additionnelles pour le chariotage

Fonctions

Pour le chariotage, on vous propose des cycles d'usinage prêts à l'usage. En plus, vous avez la possibilité de créer vos propres cycles de chariotage grâce aux fonctions élaborées suivantes :

- Création d'une table de contour (CONTPRON)
- Création d'une table de contour codée (CONTDCON)
- Désactivation de la préparation du contour (EXECUTE)
- Détermination de l'intersection entre deux éléments de contour (INTERSEC)

  (uniquement pour des tables créées avec CONTPRON)
- Exécution des éléments de contour d'une table bloc par bloc (EXECTAB)

  (uniquement pour des tables créées avec CONTPRON)
- Calcul de données de cercles (CALCDAT)

Remarque

Ces fonctions sont utilisables de façon universelle et pas seulement pour le chariotage.

Conditions

Avant tout appel des fonctions CONTPRON ou CONTDCON, il faut :

- accoster un point de départ permettant un usinage sans collision,
- désactiver la correction du rayon du tranchant avec G40.

15.2 Création d'une table de contour (CONTPRON)

Fonction

L'instruction `CONTPRON` active la préparation du contour. Les blocs CN appelés par la suite ne sont pas exécutés, mais décomposés en déplacements élémentaires qui sont rangés dans la table de contour. A chaque élément de contour correspond une ligne de la table de contour à deux dimensions. Le nombre de détalonnages qui a été déterminé est indiqué en retour.

Syntaxe

Activer la préparation du contour :

```
CONTPRON(<table de contour>,<type d'usinage>,<détalonnages>,
<sens d'usinage>)
```

Désactiver la préparation du contour et revenir au mode de traitement normal :

```
EXECUTE(<ERREUR>)
```

Voir "Désactivation de la préparation du contour (EXECUTE)"

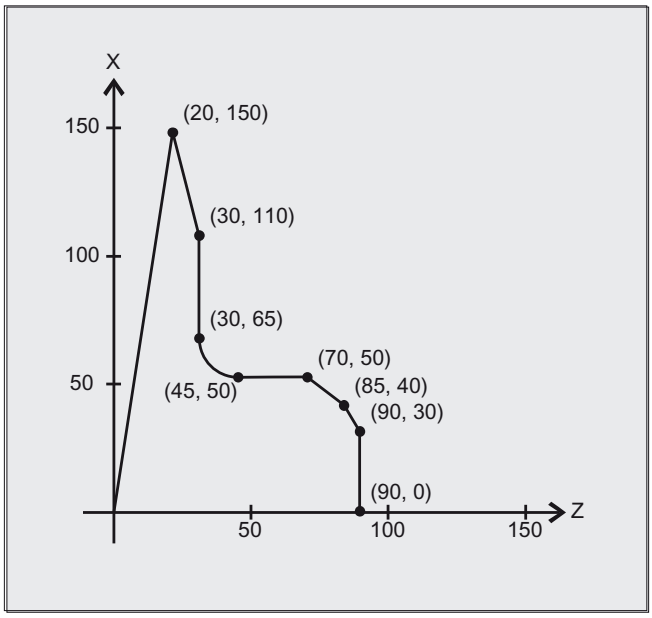
Signification

`CONTPRON`	Instruction d'activation de la préparation du contour pour la création d'une table de contour
`<Table de contour>`	Nom de la table de contour
`<Type d'usinage>`	Paramètre pour le type d'usinage Type : CHAR Valeur : `"G"` longitudinale : usinage intérieur `"L"` longitudinale : usinage extérieur `"N"` transversale : usinage intérieur `"P"` transversale : usinage extérieur
`<Détalonnages>`	Variable de résultat pour le nombre d'éléments de détalonnage générés Type : INT
`<Sens d'usinage>`	Paramètre pour le sens d'usinage Type : INT Valeur : 0 Préparation du contour en avant (valeur par défaut) 1 Préparation du contour dans les deux sens

Exemple 1

Création d'une table de contour avec :

- le nom "TABC"
- 30 éléments de contour au maximum (arcs de cercle, segments de droite)
- une variable pour le nombre d'éléments de détalonnage générés
- une variable pour les messages d'erreur



Programme CN :

Code de programme	Commentaire
N10 DEF REAL KTAB[30,11]	; Table de contour nommée TABC avec 30 éléments de contour au maximum. La valeur 11 (nombre de colonnes de la table) est fixe.
N20 DEF INT NBDETAL	; Variable pour le nombre d'éléments de détalonnage, nommée NBDETAL.
N30 DEF INT ERREUR	; Variable pour la signalisation d'erreur (0=aucune erreur, 1=erreur).
N40 G18	
N50 CONTPRON(TABC,"G",NBDETAL)	; Activer la préparation du contour.
N60 G1 X150 Z20	; N60 à N120 : Description du contour
N70 X110 Z30	
N80 X50 RND=15	
N90 Z70	
N100 X40 Z85	
N110 X30 Z90	
N120 X0	
N130 EXECUTE(ERREUR)	; Fin du remplissage de la table de contour et retour au mode de programme normal.
N140 …	; Suite du traitement de la table.

Table de contour TABC :

Indice Ligne	Colonne									
(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

Signification du contenu des colonnes :

(0) Index pointant l'élément de contour suivant (le numéro de ligne de la même table)

(1) Index pointant l'élément de contour précédent

(2) Codage du mode de contour pour le déplacement

Valeurs possibles pour 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) Point de départ des éléments de contour

(3) = abscisse, (4) = ordonnée dans le plan courant

(5), (6) Point final des éléments de contour

(5) = abscisse, (6) = ordonnée dans le plan courant

(7) Indicateur maxi/mini : indique les maxima et minima locaux du contour

(8) Valeur maximale entre élément de contour et abscisse (dans le cas d'un usinage longitudinal) ou ordonnée (en surfaçage). L'angle dépend du type d'usinage programmé.

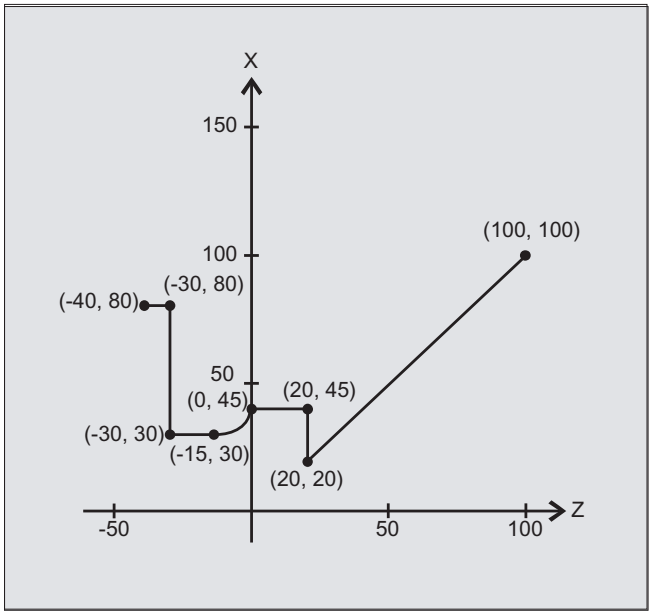
(9), (10) Coordonnées du centre de l'élément de contour, si c'est un arc de cercle..

(9) = abscisse, (10) = ordonnée

Exemple 2

Création d'une table de contour avec

- le nom TABC
- 92 éléments de contour au maximum (arcs de cercle, segments de droite)
- Mode d'usinage : cylindrage, usinage extérieur
- Préparation dans les deux sens



Programme CN :

Code de programme	**Commentaire**
N10 DEF REAL KTAB[92,11]	; Table de contour nommée TABC avec 92 éléments de contour au maximum. La valeur 11 est fixe.
N20 DEF CHAR MU="L"	; Mode de fonctionnement pour CONTPRON : cylindrage, usinage extérieur
N30 DEF INT ED=0	; Nombre d'éléments de détalonnage=0
N40 DEF INT MODE=1	; Préparation dans les deux sens
N50 DEF INT ERR=0	; Signalisation d'erreur
...	
N100 G18 X100 Z100 F1000	
N50 CONTPRON(TABC,MU,ED,MODE)	; Activer la préparation du contour.
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	

Code de programme	**Commentaire**
N170 Z-40	
N180 EXECUTE(ERR)	; Fin du remplissage de la table de contour et retour au mode de programme normal.
...	

Table de contour TABC :

Après la préparation du contour, le contour est disponible dans les deux sens.

Indice	Colonne										
Ligne	(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

Signification du contenu des colonnes et des remarques concernant les lignes 0, 1, 6, 8, 83, 85 et 91

Les significations du contenu des colonnes données dans l'exemple 1 sont valables.

Toujours dans ligne 0 :

1) Précédent : ligne n contient la fin du contour en avant

2) Suivant : ligne n est la fin de la table de contour en avant

Une fois pour les éléments de contour en avant :

3) Précédent : début du contour (en avant)

4) Suivant : fin du contour (en avant)

Toujours dans ligne fin de table de contour (en avant) +1 :

5) Précédent : nombre de détalonnages en avant

6) Suivant : nombre de détalonnages en arrière

Une fois pour les éléments de contour en arrière :

7) Suivant : fin du contour (en arrière)

8) Précédent : début du contour (en arrière)

Toujours dans la dernière ligne de la table :

9) Précédent : ligne n est le début de la table de contour (en arrière)

10) Suivant : ligne n contient le début du contour (en arrière)

Autres informations

Instructions de déplacements autorisées, système de coordonnées

Pour la programmation du contour, les instructions G suivantes sont autorisés :

- Groupe G 1 : G0, G1, G2, G3

Possibles aussi :

- Arrondi et chanfrein
- Programmation de cercle avec CIP et CT

Les fonctions spline, polynôme et filetage provoquent des erreurs.

Entre CONTPRON et EXECUTE, la modification du système de coordonnées par activation d'un frame est illicite. Ceci est également valable pour la permutation entre G70 et G71 ou entre G700 et G710.

Pendant la préparation de la table de contour , une permutation d'axe avec GEOAX entraîne une alarme.

Eléments de détalonnage

La description des différents éléments de détalonnage peut être réalisée sous la forme d'un sous-programme ou de blocs isolés, au choix.

Chariotage indépendamment du sens de contour programmé

La préparation du contour avec CONTPRON a été étendue de sorte qu'après son appel, la table de contour est disponible quel que soit le sens programmé.

15.3 Création d'une table de contour codée (CONTDCON)

Fonction

En cas d'activation de la préparation du contour avec CONTDCON, les blocs CN appelés par la suite sont enregistrés sous forme codée dans une table de contour de 6 colonnes pour simplifier la mémorisation. A chaque élément de contour correspond une ligne de la table de contour. Compte tenu des règles de codage indiquées ci-dessous, vous pouvez créer des programmes en code DIN à partir des lignes de la table, par exemple pour des cycles. Dans la ligne numéro 0 sont enregistrées les données du point de départ.

Syntaxe

Activer la préparation du contour :
`CONTDCON(<table de contour>,<sens d'usinage>)`

Désactiver la préparation du contour et revenir au mode de traitement normal :
`EXECUTE(<ERREUR>)`

Voir "Désactivation de la préparation du contour (EXECUTE)"

Signification

`CONTDCON`	Instruction d'activation de la préparation du contour pour la création d'une table de contour codée	
`<Table de contour>`	Nom de la table de contour	
`<Sens d'usinage>`	Paramètre pour le sens d'usinage	
	Type :	INT
	Valeur :	0 Préparation du contour suivant la séquence de blocs de contour (valeur par défaut)
		1 non admis

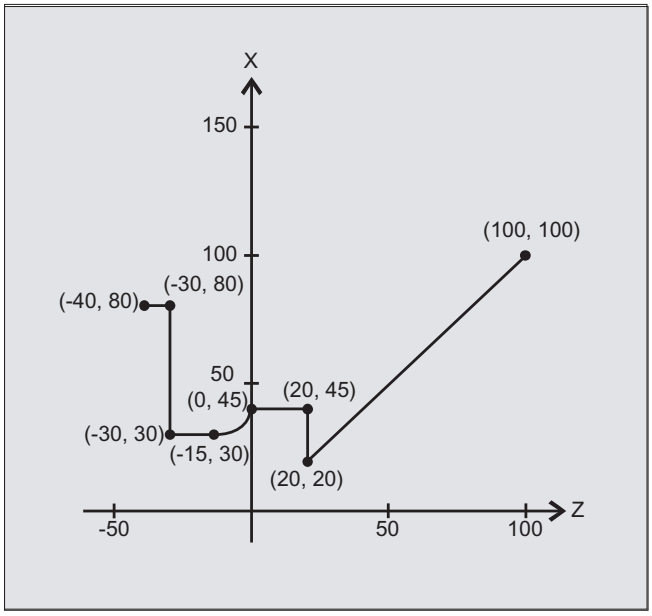
Remarque

Pour CONTDCON, les codes G admis dans la section de programme à décoder sont plus nombreux que pour la fonction CONTPRON. En outre, l'avance et le type d'avance sont enregistrés pour chaque élément de contour.

Exemple

Création d'une table de contour avec :

- le nom "TABC"
- les éléments de contour (arcs de cercle, segments de droite)
- Mode d'usinage : Tournage
- Sens d'usinage : vers l'avant



Programme CN :

Code de programme	Commentaire
N10 DEF REAL KTAB[9,6]	; Table de contour nommée TABC à 9 lignes. 8 blocs de contour { sont donc possibles. La valeur 6 (nombre de colonnes de la table) est imposée.
N20 DEF INT MODE = 0	; Variable pour le sens d'usinage. Valeur par défaut 0 : uniquement dans le sens de contour programmé.
N30 DEF INT ERROR = 0	; Variable pour la signalisation d'erreur.
...	
N100 G18 G64 G90 G94 G710	
N100 G1 X100 Z100 F1000	
N105 CONTDCON (TABC, MODE)	; Appel de la préparation du contour (MODE peut être omis).
N110 G1 Z20 X20 F200	; Description du contour.
N120 G9 X45 F300	

```
Code de programme                                    Commentaire
N130 Z0 F400
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)                                  ; Fin du remplissage de la table
                                                       de contour et retour au mode de
                                                       programme normal.
...
```

Table de contour TABC :

	Indice de colonne					
	0	1	2	3	4	5
Indice ligne	Mode de contour	Point final abscisse	Point final ordonnée	Centre abscisse	Centre ordonnée	Avance
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

Signification du contenu des colonnes :

Ligne 0 : Codes du **point de départ :**

Colonne 0 : 10^0 (position des unités) : G0 = 0
10^1 (position des dizaines) : G70 = 0, G71 = 1, G700 = 2, G710 = 3
Colonne 1 : point de départ abscisse
Colonne 2 : point de départ ordonnée
Colonne 3-4 : 0
Colonne 5 : indice de ligne du dernier élément de contour dans la table

Lignes 1-n : Valeurs des **éléments de contour**

Colonne 0 :	10^0 (position des unités) : G0 = 0, G1 = 1, G2 = 2, G3 = 3
	10^1 (position des dizaines) : G70 = 0, G71 = 1, G700 = 2, G710 = 3
	10^2 (position des centaines) : G90 = 0, G91 = 1
	10^3 (position des milliers) : G93 = 0, G94 = 1, G95 = 2, G96 = 3
	10^4 (position des dizaines de milliers) : G60 = 0, G44 = 1, G641 = 2, G642 = 3
	10^5 (position des centaines de milliers) : G9 = 1
Colonne 1 :	Point final abscisse
Colonne 2 :	Point final ordonnée
Colonne 3 :	centre abscisse en interpolation circulaire
Colonne 4 :	centre ordonnée en interpolation circulaire
Colonne 5 :	Avance

Autres informations

Instructions de déplacements autorisées, système de coordonnées

Pour la programmation du contour, les groupes G et les instructions G suivants sont autorisés :

Groupe G 1 :	G0, G1, G2, G3
groupe G 10 :	G60, G64, G641, G642
groupe G 11 :	G9
groupe G 13 :	G70, G71, G700, G710
groupe G 14 :	G90, G91
groupe G 15 :	G93, G94, G95, G96, G961

Possibles aussi :

- Arrondi et chanfrein
- Programmation de cercle avec CIP et CT

Les fonctions spline, polynôme et filetage provoquent des erreurs.

Entre CONTDCON et EXECUTE, la modification du système de coordonnées par activation d'un frame est illicite. Ceci est également valable pour la permutation entre G70 et G71 ou entre G700 et G710.

Pendant la préparation de la table de contour , une permutation d'axe avec GEOAX entraîne une alarme.

Sens d'usinage

La table de contour générée avec CONTDCON est prévue pour le chariotage dans le sens programmé du contour.

15.4 Détermination de l'intersection entre deux éléments de contour (INTERSEC)

Fonction

INTERSEC détermine le point d'intersection de deux éléments de contour normés de la table de contour créée avec CONTPRON.

Syntaxe

```
<Etat>=INTERSEC(<table de contour_1>[<élément de contour_1>],
<table de contour_2>[<élément de contour_2>],<intersection>,<mode
d'usinage>)
```

Signification

INTERSEC	Mot clé pour déterminer l'intersection de deux éléments de contour issus de tables de contour générées avec CONTPRON		
<Etat>	Variable de l'état de l'intersection		
	Type :	BOOL	
	Valeur :	TRUE	existence d'une intersection
		FALSE	pas d'intersection
<Table de contour_1>	Nom de la première table de contour		
<Elément de contour_1>	Numéro de l'élément de contour de la première table de contour		
<Table de contour_2>	Nom de la deuxième table de contour		
<Elément de contour_2>	Numéro de l'élément de contour de la deuxième table de contour		
<Intersection>	Coordonnées du point d'intersection dans le plan actif (G17/G18 / G19)		
	Type :	REAL	
<Type d'usinage>	Paramètre pour le type d'usinage		
	Type :	INT	
	Valeur :	0	Calcul du point d'intersection dans le plan activé avec le paramètre 2 (valeur par défaut)
		1	Calcul du point d'intersection indépendamment du plan transmis

Remarque

Les variables doivent avoir été définies avant leur utilisation.

La transmission des contours exige le respect des valeurs définies avec CONTPRON :

Paramètres	Signification
2	Codage du mode de contour pour le déplacement
3	Abscisse du point de départ du contour
4	Ordonnée du point de départ du contour
5	Abscisse du point final du contour
6	Ordonnée du point final du contour
9	Coordonnées du centre pour l'abscisse (uniquement pour le contour du cercle)
10	Coordonnées du centre pour l'ordonnée (uniquement pour le contour du cercle)

Exemple

Il s'agit de déterminer l'intersection de l'élément de contour 3 de la table TABNAME1 et de l'élément de contour 7 de la table TABNAME2. Les coordonnées de l'intersection, dans le plan actif, sont rangées dans la variable ISCOORD (1er élément = abscisse, 2ème élément = ordonnée). S'il n'existe pas d'intersection, il y a saut vers PASINTER (aucune intersection trouvée).

```
Code de programme                                                    Commentaire
DEF REAL TABNAME1[12,11]                                             ; Table de contour 1
DEF REAL TABNAME2[10,11]                                             ; Table de contour 2
DEF REAL ISCOORD [2]                                                 ; Variable des coordonnées du
                                                                       point d'intersection.
DEF BOOL ISPOINT                                                     ; Variable de l'état du point
                                                                       d'intersection.
DEF INT MODE                                                         ; Variable pour le type
                                                                       d'usinage.
…
MODE=1                                                               ; Calcul indépendant du plan
                                                                       actif.
N10 ISPOINT=INTERSEC(TABNAME1[3],TABNAME2[7],ISCOORD,MODE)           ; Appel de la fonction
                                                                       Intersection des éléments de
                                                                       contour.
N20 IF ISPOINT==FALSE GOTOF PASINTER                                 ; Saut vers PASINTER.
…
```

15.5 Exécution des éléments de contour d'une table bloc par bloc (EXECTAB)

Fonction

Avec l'instruction EXECTAB, vous pouvez exécuter bloc par bloc les éléments de contour d'une table qui a été créée, par exemple, avec l'instruction CONTPRON.

Syntaxe

```
EXECTAB(<table de contour>[<élément de contour>])
```

Signification

EXECTAB	Instruction d'exécution d'un élément de contour
<Table de contour>	Nom de la table de contour
<Elément de contour>	Numéro de l'élément de contour

Exemple

Les éléments de contour 0 à 2 de la table TABC doivent être exécutés bloc par bloc.

Code de programme	Commentaire
N10 EXECTAB (TABC[0])	; Exécuter l'élément 0 de la table TABC.
N20 EXECTAB(KTAB[1])	; Exécuter l'élément 1 de la table TABC.
N30 EXECTAB(KTAB[2])	; Exécuter l'élément 2 de la table TABC.

15.6 Calcul de données de cercles (CALCDAT)

Fonction

L'instruction `CALCDAT` vous permet de calculer le rayon et les coordonnées du centre d'un cercle à partir de trois ou quatre points connus du cercle. Les points indiqués doivent être distincts. En cas d'indication de 4 points qui ne se trouvent pas exactement sur un cercle, des valeurs moyennes sont calculées pour le centre du cercle et le rayon.

Syntaxe

```
<Etat>=CALCDAT(<points du
cercle>[<nombre>,<type>],<nombre>,<résultat>)
```

Signification

`CALCDAT`
Instruction de calcul du rayon et des coordonnées du centre d'un cercle à partir de 3 ou 4 points

`<Etat>`
Variable de l'état du calcul du cercle

Type : BOOL

Valeur :
- TRUE : Les points indiqués se situent sur un cercle.
- FALSE : Les points indiqués ne se situent **pas** sur un cercle.

`<Points du cercle>[]`
Variable pour l'indication des points du cercle avec les paramètres :
- `<Nombre>` : Nombre de points du cercle (3 ou 4)
- `<Type>` : Type de coordonnées, par exemple 2 pour coordonnées à deux points

`<Nombre>`
Paramètre du nombre de points utilisés pour le calcul (3 ou4)

`<Résultat>[3]`
Variables du résultat :

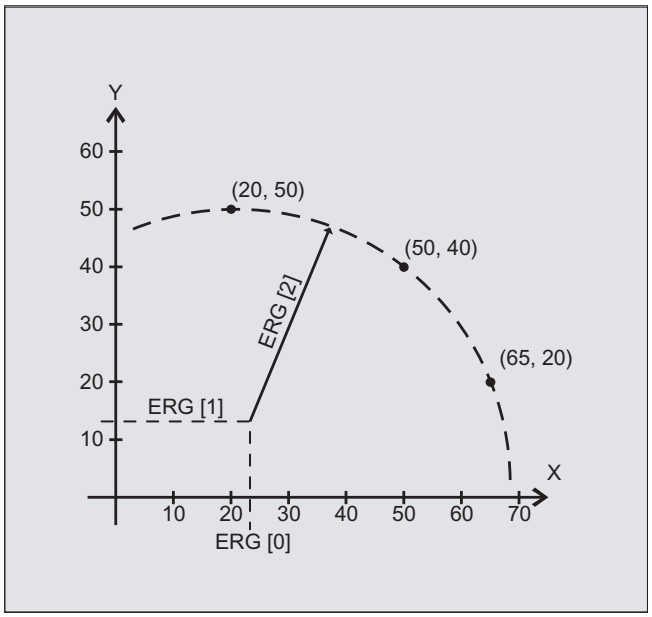
Coordonnées du centre du cercle et rayon
- 0 Coordonnées du centre du cercle : abscisse
- 1 Coordonnées du centre du cercle : ordonnée
- 2 Rayon

Remarque

Les variables doivent avoir été définies avant leur utilisation.

Exemple

On veut savoir si trois points sont situés sur un arc de cercle.



Code de programme	**Commentaire**
`N10 DEF REAL PT[3,2]=(20,50,50,40,65,20)`	`; Variable pour l'indication des points du cercle`
`N20 DEF REAL RES[3]`	`; Variable du résultat`
`N30 DEF BOOL ETAT`	`; Variable d'état`
`N40 ETAT=CALCDAT(POINT,3,RES)`	`; Appel des données calculées.`
`N50 IF ETAT == FALSE GOTOF ERREUR`	`; Saut en cas d'erreur`

15.7 Désactivation de la préparation du contour (EXECUTE)

Fonction

Avec l'instruction EXECUTE, on arrête la préparation du contour et on retourne dans le mode d'exécution normal.

Syntaxe

```
EXECUTE(<ERREUR>)
```

Signification

EXECUTE	Instruction pour terminer la préparation du contour
<ERREUR>	Variable pour la signalisation d'erreur Type : INT La valeur de la variable indique si le contour a pu être préparé correctement : 0 Erreur 1 Pas d'erreur

Exemple

Code de programme

```
...
N30 CONTPRON(…)
N40 G1 X… Z…
...
N100 EXECUTE(…)
...
```

Tableaux

16

16.1 Liste des instructions

Légende :

1) Référence au document contenant la description complète de l'instruction :

PGsl	Manuel de programmation Notions de base
PGAsl	Manuel de programmation Notions complémentaires
BHDsl	Manuel d'utilisation Tournage
BHFsl	Manuel d'utilisation Fraisage
FB1 ( )	Description fonctionnelle Fonctions de base (avec l'indication entre parenthèses de l'abréviation alphanumérique de la description fonctionnelle correspondante)
FB2 ( )	Description fonctionnelle Fonctions d'extension (avec l'indication entre parenthèses de l'abréviation alphanumérique de la description fonctionnelle correspondante)
FB3 ( )	Description fonctionnelle Fonctions spéciales (avec l'indication entre parenthèses de l'abréviation alphanumérique de la description fonctionnelle correspondante)
FBSIsl	Description fonctionnelle Safety Integrated
FBSY	Description fonctionnelle Actions synchrones
FBW	Description fonctionnelle Gestion d'outils

2) Prise d'effet de l'instruction :

m	modale
b	bloc par bloc (non modale)

3) Disponibilité pour SINUMERIK 828D (D = Tournage, F = Fraisage) :

●	Standard
○	Option
-	Non disponible

4) Réglage par défaut en début du programme (dans la version de base de la commande, si elle n'a pas été programmée autrement).

Instruction	Signification	Description, voir 1)	E 2)	828D 3)			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
:	Numéro du bloc principal de la CN, fin des repères de saut, opérateur de concaténation	*PGAsl* Fonctions de calcul (Page 61)		●	●	●	●
*	Opérateur de multiplication	*PGAsl* Fonctions de calcul (Page 61)		●	●	●	●
+	Opérateur d'addition	*PGAsl* Fonctions de calcul (Page 61)		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
-	Opérateur de soustraction	*PGAs*/ Fonctions de calcul (Page 61)		●	●	●	●
<	Opérateur de comparaison, inférieur à	*PGAs*/ Fonctions de calcul (Page 61)		●	●	●	●
<<	Opérateur de concaténation pour chaînes	*PGAs*/ Fonctions de calcul (Page 61)		●	●	●	●
<=	Opérateur de comparaison, inférieur ou égal à	*PGAs*/ Fonctions de calcul (Page 61)		●	●	●	●
=	Opérateur d'affectation	*PGAs*/ Fonctions de calcul (Page 61)		●	●	●	●
>=	Opérateur de comparaison, supérieur ou égal à	*PGAs*/ Fonctions de calcul (Page 61)		●	●	●	●
/	Opérateur de division	*PGAs*/ Fonctions de calcul (Page 61)		●	●	●	●
/0 … … /7	Le bloc sera sauté (1er niveau) Le bloc sera sauté (8e niveau)	*PGs*/		● ○	● ○	● ○	● ○
A	Nom d'axe	*PGAs*/ Programmation de l'orientation de l'outil (A..., B..., C..., LEAD, TILT) (Page 323)	m/b	●	●	●	●
A2	Orientation de l'outil : angle RPY ou angle d'Euler	*PGAs*/ Programmation de l'orientation de l'outil (A..., B..., C..., LEAD, TILT) (Page 323)	b	●	●	●	●
A3	Orientation de l'outil : Composante de vecteur normale à la direction/surface	*PGAs*/ Programmation de l'orientation de l'outil (A..., B..., C..., LEAD, TILT) (Page 323)	b	●	●	●	●
A4	Orientation de l'outil : Vecteur normal à la surface pour le début du bloc	*PGAs*/ Fraisage en bout (fraisage 3D A4, B4, C4, A5, B5, C5) (Page 329)	b	●	●	●	●
A5	Orientation de l'outil : Vecteur normal à la surface pour la fin du bloc	*PGAs*/ Fraisage en bout (fraisage 3D A4, B4, C4, A5, B5, C5) (Page 329)	b	●	●	●	●
ABS	Valeur absolue (montant)	*PGAs*/ Fonctions de calcul (Page 61)		●	●	●	●
AC	Cotation absolue de coordonnées/positions	*PGs*/	b	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
ACC	Influence de l'accélération axiale actuelle	*PGs/*	m	●	●	●	●
ACCLIMA	Influence de l'accélération axiale maximale actuelle	*PGs/*	m	●	●	●	●
ACN	Cotation absolue pour axes rotatifs, accostage de position dans le sens négatif	*PGs/*	b	●	●	●	●
ACOS	Arc cosinus (fonction trigonométrique)	*PGAs/* Fonctions de calcul (Page 61)		●	●	●	●
ACP	Cotation absolue pour axes rotatifs, accostage de position dans le sens positif	*PGs/*	b	●	●	●	●
ACTBLOCNO	Affichage du numéro courant d'un bloc d'alarme, même si "Inhibition de l'affichage du bloc courant" (DISPLOF) est actif !	*PGAs/* Inhibition de l'affichage du bloc courant (DISPLOF, DISPLON, ACTBLOCNO) (Page 164)		●	●	●	●
ADDFRAME	Prise en compte et éventuellement activation d'un frame mesuré	*PGAs/, FB1(K2)* Calcul d'un frame à partir de 3 points mesurés dans l'espace (MEAFRAME) (Page 298)		●	●	●	●
ADIS	Distance de transition entre blocs pour fonctions d'interpolation G1, G2, G3, ...	*PGs/*	m	●	●	●	●
ADISPOS	Distance de transition entre blocs pour vitesse rapide G0	*PGs/*	m	●	●	●	●
ADISPOSA	Dimension de la fenêtre de tolérance pour IPOBRKA	*PGAs/* Critère programmable de fin de déplacement (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) (Page 274)	m	●	●	●	●
ALF	Angle de relèvement rapide	*PGAs/* Retrait rapide du contour (SETINT LIFTFAST, ALF) (Page 115)	m	●	●	●	●
AMIRROR	Fonction miroir programmable	*PGs/*	b	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
AND	ET logique	*PGAs/* Opérations relationnelles et opérations logiques (Page 64)		•	•	•	•
ANG	Angle d'élément de contour	*PGs/*	b	•	•	•	•
AP	Angle polaire	*PGs/*	m/b	•	•	•	•
APR	Droit d'accès en lecture / à l'affichage	*PGAs/* Attribut : droits d'accès (APR, APW, APRP, APWP, APRB, APWB) (Page 38)		•	•	•	•
APRB	Droit d'accès en lecture, BTSS	*PGAs/* Attribut : droits d'accès (APR, APW, APRP, APWP, APRB, APWB) (Page 38)		•	•	•	•
APRP	Droit d'accès en lecture, programme pièce	*PGAs/* Attribut : droits d'accès (APR, APW, APRP, APWP, APRB, APWB) (Page 38)		•	•	•	•
APW	Droit d'accès en écriture	*PGAs/* Attribut : droits d'accès (APR, APW, APRP, APWP, APRB, APWB) (Page 38)		•	•	•	•
APWB	Droit d'accès en écriture, BTSS	*PGAs/* Attribut : droits d'accès (APR, APW, APRP, APWP, APRB, APWB) (Page 38)		•	•	•	•
APWP	Droit d'accès en écriture, programme pièce	*PGAs/* Attribut : droits d'accès (APR, APW, APRP, APWP, APRB, APWB) (Page 38)		•	•	•	•
APX	Définition de la protection d'accès pour l'exécution de l'élément de langage indiqué	*PGAs/* Redéfinition de variables système, variables utilisateur et instruction de langage CN (REDEF) (Page 28)		•	•	•	•
AR	Angle au centre	*PGs/*	m/b	•	•	•	•
AROT	Rotation programmable	*PGs/*	b	•	•	•	•
AROTS	Programmation de rotations de frames avec des angles solides	*PGs/*	b	•	•	•	•

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
AS	Définition d'une macro-instruction	*PGAs*/ Macroprogrammation (DEFINE ... AS) (Page 201)		●	●	●	●
ASCALE	Mise à l'échelle programmable	*PGs*/	b	●	●	●	●
ASIN	Fonction de calcul, arc sinus	*PGAs*/ Fonctions de calcul (Page 61)		●	●	●	●
ASPLINE	Akima Spline	*PGAs*/ Interpolation de type spline (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Page 233)	m	-	○	-	○
ATAN2	arc tangente 2	*PGAs*/ Fonctions de calcul (Page 61)		●	●	●	●
ATOL	Tolérance spécifique à l'axe pour fonctions de compresseur, lissage de l'orientation et modes d'arrondissement	*PGAs*/ Tolérance de contour/orientation programmable (CTOL, OTOL, ATOL) (Page 491)		-	●	-	●
ATRANS	décalage additif programmable	*PGs*/	b	●	●	●	●
AX	Descripteur d'axe variable	*PGAs*/ Fonctions d'axe (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) (Page 669)	m/b	●	●	●	●
AXCTSWE	Rotation de conteneur d'axes	*PGAs*/ Conteneur d'axes (AXCTSWE, AXCTSWED) (Page 677)		-	-	-	-
AXCTSWED	Rotation conteneur d'axes	*PGAs*/ Conteneur d'axes (AXCTSWE, AXCTSWED) (Page 677)		-	-	-	-
AXIS	Descripteur d'axe, adresse d'axe	*PGAs*/ Définition de variables utilisateur (DEF) (Page 22)		●	●	●	●
AXNAME	Conversion de la chaîne d'entrée en descripteur d'axe	*PGAs*/ Fonctions d'axe (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) (Page 669)		●	●	●	●
AXSTRING	Conversion de la chaîne en numéro de broche	*PGAs*/ Fonctions d'axe (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) (Page 669)		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
AXTOCHAN	Demande d'axe pour un canal donné Est possible depuis le programme CN ou à partir d'une action synchrone.	*PGAs*/ Transmettre un axe à un autre canal (AXTOCHAN) (Page 126)		•	•	•	•
AXTOSPI	Conversion du descripteur d'axe en indice de broche	*PGAs*/ Fonctions d'axe (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) (Page 669)		•	•	•	•
B	Nom d'axe	*PGAs*/ Programmation de l'orientation de l'outil (A..., B..., C..., LEAD, TILT) (Page 323)	m/b	•	•	•	•
B2	Orientation de l'outil : angle RPY ou angle d'Euler	*PGAs*/ Programmation de l'orientation de l'outil (A..., B..., C..., LEAD, TILT) (Page 323)	b	•	•	•	•
B3	Orientation de l'outil : composante de vecteur normale à la direction/surface	*PGAs*/ Programmation de l'orientation de l'outil (A..., B..., C..., LEAD, TILT) (Page 323)	b	•	•	•	•
B4	Orientation de l'outil : Vecteur normal à la surface pour le début du bloc	*PGAs*/ Fraisage en bout (fraisage 3D A4, B4, C4, A5, B5, C5) (Page 329)	b	•	•	•	•
B5	Orientation de l'outil : Vecteur normal à la surface pour la fin du bloc	*PGAs*/ Fraisage en bout (fraisage 3D A4, B4, C4, A5, B5, C5) (Page 329)	b	•	•	•	•
B_AND	ET sur bits	*PGAs*/ Opérations relationnelles et opérations logiques (Page 64)		•	•	•	•
B_OR	OU sur bits	*PGAs*/ Opérations relationnelles et opérations logiques (Page 64)		•	•	•	•
B_NOT	Négation sur bits	*PGAs*/ Opérations relationnelles et opérations logiques (Page 64)		•	•	•	•
B_XOR	OU exclusif sur bits	*PGAs*/ Opérations relationnelles et opérations logiques (Page 64)		•	•	•	•

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
BAUTO	Définition de la première section spline par les 3 points suivants	*/PGAs/* Interpolation de type spline (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Page 233)	m	-	○	-	○
BLOC	Définition, en association avec le mot-clé TO, de la section de programme à exécuter dans un sous-programme indirect	*/PGAs/* Appel indirect d'un sous-programme avec indication de la section de programme à exécuter (CALL BLOCK ... TO ...) (Page 188)		●	●	●	●
BLSYNC	L'exécution de la routine d'interruption ne doit commencer qu'au changement de bloc suivant	*/PGAs/* Affectation et démarrage d'une routine d'interruption (SETINT, PRIO, BLSYNC) (Page 111)		●	●	●	●
BNAT [4]	Raccordement naturel avec le premier bloc spline	*/PGAs/* Interpolation de type spline (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Page 233)	m	-	○	-	○
BOOL	Type de données : valeurs booléennes TRUE / FALSE ou 1/0	*/PGAs/* Définition de variables utilisateur (DEF) (Page 22)		●	●	●	●
BOUND	Vérifie si la valeur se trouve la plage de valeurs définie. Egalité retourne la valeur de test.	*/PGAs/* Minimum, maximum et plage de variables (MINVAL, MAXVAL, BOUND) (Page 68)		●	●	●	●
BRISK [4]	Accélération résultante sous forme d'échelon	*/PGs/*	m	●	●	●	●
BRISKA	Activer l'accélération résultante par échelon pour les axes programmés	*/PGs/*		●	●	●	●
BSPLINE	B Spline	*/PGAs/* Interpolation de type spline (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Page 233)	m	-	○	-	○
BTAN	Raccordement tangentiel avec le premier bloc spline	*/PGAs/* Interpolation de type spline (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Page 233)	m	-	○	-	○

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
C	Nom d'axe	*PGAs/* Programmation de l'orientation de l'outil (A..., B..., C..., LEAD, TILT) (Page 323)	m/b	•	•	•	•
C2	Orientation de l'outil : angle RPY ou angle d'Euler	*PGAs/* Programmation de l'orientation de l'outil (A..., B..., C..., LEAD, TILT) (Page 323)	b	•	•	•	•
C3	Orientation de l'outil : composante de vecteur normale à la direction/surface	*PGAs/* Programmation de l'orientation de l'outil (A..., B..., C..., LEAD, TILT) (Page 323)	b	•	•	•	•
C4	Orientation de l'outil : Vecteur normal à la surface pour le début du bloc	*PGAs/* Fraisage en bout (fraisage 3D A4, B4, C4, A5, B5, C5) (Page 329)	b	•	•	•	•
C5	Orientation de l'outil : Vecteur normal à la surface pour la fin du bloc	*PGAs/* Fraisage en bout (fraisage 3D A4, B4, C4, A5, B5, C5) (Page 329)	b	•	•	•	•
CAC	Accostage absolu d'une position	*PGAs/* Accostage de positions codées (CAC, CIC, CDC, CACP, CACN) (Page 231)		•	•	•	•
CACN	Valeur rangée dans table est accostée de façon absolue en sens négatif	*PGAs/* Accostage de positions codées (CAC, CIC, CDC, CACP, CACN) (Page 231)		•	•	•	•
CACP	Valeur rangée dans table est accostée de façon absolue en sens positif	*PGAs/* Accostage de positions codées (CAC, CIC, CDC, CACP, CACN) (Page 231)		•	•	•	•
CALCDAT	Calcul du rayon et du centre d'un cercle passant par 3 ou 4 points	*PGAs/* Calcul de données de cercles (CALCDAT) (Page 715)		•	•	•	•
CALCPOSI	Vérification de la violation de la zone de protection, de la limitation de la zone de travail et des limites logicielles	*PGAs/* Contrôle des violations de zone de protection, des limitations de la zone de travail et des fins de course logiciels (CALCPOSI) (Page 223)		•	•	•	•
CALL	Appel indirect de sous-programme	*PGAs/* Appel indirect de sous-programme (CALL) (Page 187)		•	•	•	•

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
CALLPATH	Chemin de recherche programmable pour l'appel de sous-programmes	*PGAs*/ Extension du chemin de recherche pour l'appel de sous-programmes (CALLPATH) (Page 192)		●	●	●	●
CANCEL	Annuler une action synchrone modale	*PGAs*/ Effacer une action synchrone (CANCEL) (Page 636)		●	●	●	●
CASE	Branchement conditionnel	*PGAs*/ Saut de programme (CASE ... OF ... DEFAULT ...) (Page 86)		●	●	●	●
CDC	Accostage direct d'une position	*PGAs*/ Accostage de positions codées (CAC, CIC, CDC, CACP, CACN) (Page 231)		●	●	●	●
CDOF [4]	Désactivation de la surveillance anticollision	*PGs*/	m	●	●	●	●
CDOF2	Désactivation de la surveillance anticollision, pour fraisage périphérique 3D	*PGs*/	m	●	●	●	●
CDON	Activation de la surveillance anticollision	*PGs*/	m	●	●	●	●
CFC [4]	Avance constante au niveau du contour	*PGs*/	m	●	●	●	●
CFIN	Avance constante uniquement pour courbure concave, pas pour courbure convexe	*PGs*/	m	●	●	●	●
CFINE	Affectation d'un décalage fin à une variable FRAME	*PGAs*/ Décalage grossier et décalage fin (CFINE, CTRANS) (Page 293)		●	●	●	●
CFTCP	Avance constante au point de référence de l'arête tranchante de l'outil, trajectoire centrale	*PGs*/	m	●	●	●	●
CHAN	Spécification du domaine de validité de données	*PGAs*/ Définition de variables utilisateur (DEF) (Page 22)		●	●	●	●
CHANDATA	Réglage du numéro de canal pour des accès à des données de canal	*PGAs*/ Mémoire de travail (CHANDATA, COMPLETE, INITIAL) (Page 210)		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
CHAR	Type de données : Caractère ASCII	*PGAs/* Définition de variables utilisateur (DEF) (Page 22)		●	●	●	●
CHECKSUM	Forme la somme de contrôle du tableau en tant que chaîne de caractères avec une longueur fixe	*PGAs/* Calcul du total de contrôle d'un tableau (CHECKSUM) (Page 142)		●	●	●	●
CHF	Chanfrein ; valeur = longueur du chanfrein	*PGs/*	b	●	●	●	●
CHKDM	Contrôle d'univocité dans un magasin	*FBW*		●	●	●	●
CHKDNO	Contrôle d'univocité des numéros D	*PGAs/* Libre affectation des numéros D : contrôle des numéros D (CHKDNO) (Page 430)		●	●	●	●
CHR	Chanfrein ; valeur = longueur du chanfrein dans la direction du déplacement	*PGs/*		●	●	●	●
CIC	Accostage incrémental d'une position	*PGAs/* Accostage de positions codées (CAC, CIC, CDC, CACP, CACN) (Page 231)		●	●	●	●
CIP	Interpolation circulaire avec point intermédiaire	*PGs/*	m	●	●	●	●
CLEARM	Effacement d'une/de plusieurs marques de coordination entre canaux	*PGAs/* Coordination de programmes (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) (Page 103)		-	-	-	-
CLRINT	Annuler une interruption	*PGAs/* Suppression d'une affectation de routine d'interruption (CLRINT) (Page 114)		●	●	●	●
CMIRROR	fonction miroir par rapport à un axe de coordonnées	*PGAs/* Fonctions de calcul (Page 61)		●	●	●	●
COARSEA	Fin du déplacement lors de l'atteinte de "Arrêt précis grossier"	*PGAs/* Critère programmable de fin de déplacement (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) (Page 274)	m	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
COMPCAD	Activation du compacteur : Gain de surface optimisé pour programmes CAD	*PGAs*/ Compactage de bloc CN (COMPON, COMPCURV, COMPCAD, COMPOF) (Page 247)	m	-	○	-	○
COMPCURV	Activation du compacteur : polynômes à courbure continue	*PGAs*/ Compactage de bloc CN (COMPON, COMPCURV, COMPCAD, COMPOF) (Page 247)	m	-	○	-	○
COMPLETE	Instruction de commande pour la lecture et l'écriture de données	PGAsl Mémoire de travail (CHANDATA, COMPLETE, INITIAL) (Page 210)		●	●	●	●
COMPOF [4]	Désactivation du compacteur	*PGAs*/ Compactage de bloc CN (COMPON, COMPCURV, COMPCAD, COMPOF) (Page 247)	m	-	○	-	○
COMPON	Activation du compacteur	*PGAs*/ Compactage de bloc CN (COMPON, COMPCURV, COMPCAD, COMPOF) (Page 247)		-	○	-	○
CONTDCON	Activation du décodage du contour, sous forme tabulaire	*PGAs*/ Création d'une table de contour codée (CONTDCON) (Page 708)		●	●	●	●
CONTPRON	Activation de la préparation des références	*PGAs*/ Création d'une table de contour (CONTPRON) (Page 702)		●	●	●	●
CORROF	Désélection de tous les déplacements forcés actifs.	*PGs*/		●	●	●	●
COS	Cosinus (fonction trigonométrique)	*PGAs*/ Fonctions de calcul (Page 61)		●	●	●	●
COUPDEF	Définition groupe boîtes électroniques / groupe broches synchrones	*PGAs*/ Broche synchrone : Programmation (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) (Page 535)		○	-	○	-

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
COUPDEL	Suppression groupe boîtes électroniques	*PGAs*/ Broche synchrone : Programmation (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) (Page 535)		○	-	○	-
COUPOF	Activation du groupe boîtes électroniques / groupe broches synchrones	*PGAs*/ Broche synchrone : Programmation (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) (Page 535)		○	-	○	-
COUPOFS	Désactivation du groupe boîtes électroniques / groupe broches synchrones avec arrêt de la broche asservie	*PGAs*/ Broche synchrone : Programmation (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) (Page 535)		○	-	○	-
COUPON	Activation du groupe boîtes électroniques / groupe broches synchrones	*PGAs*/ Broche synchrone : Programmation (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) (Page 535)		○	-	○	-
COUPONC	Activation du groupe boîtes électroniques / groupe broches synchrones avec prise en charge de la programmation précédente	*PGAs*/ Broche synchrone : Programmation (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) (Page 535)		○	-	○	-
COUPRES	Réinitialisation groupe boîtes électroniques	*PGAs*/ Broche synchrone : Programmation (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) (Page 535)		○	-	○	-
CP	Déplacement avec interpolation	*PGAs*/ Déplacement PTP cartésien (Page 376)	m	●	●	●	●
CPRECOF [4]	Désactivation de la précision de contour programmable	*PGs*/	m	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
CPRECON	Précision de contour programmable activée	*PGs/*	m	●	●	●	●
CPROT	Activation/désactivation d'une zone de { protection spécifique à un canal	*PGAs/* Activation/désactivation des zones de protection (CPROT, NPROT) (Page 219)		●	●	●	●
CPROTDEF	Définition d'une zone de protection spécifique au canal	*PGAs/* Définition des zones de protection (CPROTDEF, NPROTDEF) (Page 215)		●	●	●	●
CR	Rayon du cercle	*PGs/*	b	●	●	●	●
CROT	Rotation du système de coordonnées courant	*PGAs/* Fonctions de calcul (Page 61)		●	●	●	●
CROTS	Rotations de frames programmables avec angles solides (rotation dans les axes indiqués)	*PGs/*	b	●	●	●	●
CRPL	Rotation de frame dans un plan quelconque	*FB1(K2)*		●	●	●	●
CSCALE	Facteur d'échelle pour plusieurs axes	*PGAs/* Fonctions de calcul (Page 61)		●	●	●	●
CSPLINE	Spline cubique	*PGAs/* Interpolation de type spline (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Page 233)	m	-	○	-	○
CT	Cercle avec transition tangentielle	*PGs/*	m	●	●	●	●
CTAB	Déterminer position axe asservi correspondant à position axe pilote d'après table de courbe	*PGAs/* Lecture des valeurs de la table de courbes (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Page 513)		-	-	-	-
CTABDEF	Activation définition de table	*PGAs/* Définition de tables de courbes (CTABDEF, CATBEND) (Page 502)		-	-	-	-
CTABDEL	Effacer une table de courbes	*PGAs/* Suppression d'une table de courbes (CTABDEL) (Page 508)		-	-	-	-

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
CTABEND	Désactivation définition de table	*PGAs*/ Définition de tables de courbes (CTABDEF, CATBEND) (Page 502)		-	-	-	-
CTABEXISTS	Vérifie la table de courbe ayant le numéro n	*PGAs*/ Vérification de l'existence d'une table de courbes (CTABEXISTS) (Page 508)		-	-	-	-
CTABFNO	Nombre des tables de courbes encore possibles dans la mémoire	*PGAs*/ Tables de courbes : Vérification de l'utilisation des ressources (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Page 518)		-	-	-	-
CTABFPOL	Nombre des polynômes encore possibles dans la mémoire	*PGAs*/ Tables de courbes : Vérification de l'utilisation des ressources (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Page 518)		-	-	-	-
CTABFSEG	Nombre des segments de courbes encore possibles dans la mémoire	*PGAs*/ Tables de courbes : Vérification de l'utilisation des ressources (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Page 518)		-	-	-	-
CTABID	Indication du numéro de table de la nième table de courbes	*PGAs*/ Tables de courbes : détermination des propriétés de la table (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD) (Page 511)		-	-	-	-
CTABINV	Déterminer position axe pilote correspondant à position axe asservi d'après table de courbe	*PGAs*/ Lecture des valeurs de la table de courbes (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Page 513)		-	-	-	-

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
CTABISLOCK	Retourne l'état de blocage de la table de courbes ayant le numéro n	*PGAs*/ Tables de courbes : détermination des propriétés de la table (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD) (Page 511)		-	-	-	-
CTABLOCK	Verrouillage de la suppression et de l'écrasement	*PGAs*/ Verrouillage de tables de courbes contre la suppression et l'écrasement (CTABLOCK, CTABUNLOCK) (Page 510)		-	-	-	-
CTABMEMTYP	Retourne la mémoire dans laquelle se trouve la table de courbes ayant le numéro n.	*PGAs*/ Tables de courbes : détermination des propriétés de la table (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD) (Page 511)		-	-	-	-
CTABMPOL	Nombre maximum des polynômes possibles dans la mémoire	*PGAs*/ Tables de courbes : Vérification de l'utilisation des ressources (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Page 518)		-	-	-	-
CTABMSEG	Nombre maximum des segments de courbes possibles dans la mémoire	*PGAs*/ Tables de courbes : Vérification de l'utilisation des ressources (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Page 518)		-	-	-	-
CTABNO	Nombre de tables de courbes définies dans SRAM ou DRAM	*FB3(M3)*		-	-	-	-
CTABNOMEM	Nombre de tables de courbes définies dans SRAM ou DRAM	*PGAs*/ Tables de courbes : Vérification de l'utilisation des ressources (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Page 518)		-	-	-	-

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
CTABPERIOD	Restitue la périodicité de la table de courbes de numéro n	*PGAs*/ Tables de courbes : détermination des propriétés de la table (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD) (Page 511)		-	-	-	-
CTABPOL	Nombre des polynômes déjà utilisés dans la mémoire	*PGAs*/ Tables de courbes : Vérification de l'utilisation des ressources (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Page 518)		-	-	-	-
CTABPOLID	Nombre des polynômes de courbes de la table de courbes ayant le numéro n	*PGAs*/ Tables de courbes : Vérification de l'utilisation des ressources (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Page 518)		-	-	-	-
CTABSEG	Nombre des segments de courbes déjà utilisés dans la mémoire	*PGAs*/ Tables de courbes : Vérification de l'utilisation des ressources (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Page 518)		-	-	-	-
CTABSEGID	Nombre des segments de courbes de la table de courbe ayant le numéro n	*PGAs*/ Tables de courbes : Vérification de l'utilisation des ressources (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Page 518)		-	-	-	-
CTABSEV	Indique la valeur de fin de l'axe asservi du segment de la table de courbes	*PGAs*/ Lecture des valeurs de la table de courbes (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Page 513)		-	-	-	-

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
CTABSSV	Indique la valeur de départ de l'axe asservi du segment de la table de courbes	*PGAs*/ Lecture des valeurs de la table de courbes (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Page 513)		-	-	-	-
CTABTEP	Indique la valeur de l'axe pilote à la fin de la table de courbes	*PGAs*/ Lecture des valeurs de la table de courbes (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Page 513)		-	-	-	-
CTABTEV	Indique la valeur de l'axe asservi à la fin de la table de courbes	*PGAs*/ Lecture des valeurs de la table de courbes (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Page 513)		-	-	-	-
CTABTMAX	Indique la valeur maximale de l'axe asservi de la table de courbes	*PGAs*/ Lecture des valeurs de la table de courbes (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Page 513)		-	-	-	-
CTABTMIN	Indique la valeur minimale de l'axe asservi de la table de courbes	*PGAs*/ Lecture des valeurs de la table de courbes (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Page 513)		-	-	-	-
CTABTSP	Indique la valeur de l'axe pilote au début de la table de courbes.	*PGAs*/ Lecture des valeurs de la table de courbes (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Page 513)		-	-	-	-

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
CTABTSV	Indique la valeur de l'axe asservi au début de la table de courbes	*PGAs/* Lecture des valeurs de la table de courbes (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Page 513)		-	-	-	-
CTABUNLOCK	Annulation du verrouillage de la suppression et de l'écrasement	*PGAs/* Verrouillage de tables de courbes contre la suppression et l'écrasement (CTABLOCK, CTABUNLOCK) (Page 510)		-	-	-	-
CTOL	Tolérance de contour pour fonctions de compresseur, lissage de l'orientation et modes d'arrondissement	*PGAs/* Tolérance de contour/orientation programmable (CTOL, OTOL, ATOL) (Page 491)		-	○	-	○
CTRANS	Décalage d'origine pour plusieurs axes	*PGAs/* Décalage grossier et décalage fin (CFINE, CTRANS) (Page 293)		●	●	●	●
CUT2D [4]	Correction d'outil 2D	*PGs/*	m	●	●	●	●
CUT2DF	La correction d'outil 2D agit de façon relative par rapport au frame actuel (plan incliné).	*PGs/*	m	●	●	●	●
CUT3DC	Correction d'outil 3D pour fraisage périphérique	*PGAs/* Activation des corrections d'outil 3D (CUT3DC..., CUT3DF...) (Page 409)	m	-	-	-	-
CUT3DCC	Correction d'outil 3D pour fraisage périphérique avec surfaces de délimitation	*PGAs/* Correction d'outil 3D : prise en compte d'une surface de délimitation (CUT3DCC, CUT3DCCD) (Page 419)	m	-	-	-	-
CUT3DCCD	Correction d'outil 3D pour fraisage périphérique avec surfaces de délimitation avec outil différentiel	*PGAs/* Correction d'outil 3D : prise en compte d'une surface de délimitation (CUT3DCC, CUT3DCCD) (Page 419)	m	-	-	-	-
CUT3DF	Correction d'outil 3D pour fraisage en bout	*PGAs/* Activation des corrections d'outil 3D (CUT3DC..., CUT3DF...) (Page 409)	m	-	-	-	-

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
CUT3DFF	Correction d'outil 3D pour fraisage en bout avec orientation d'outil constante, dépendante du frame actif	*/PGAs/* Activation des corrections d'outil 3D (CUT3DC..., CUT3DF...) (Page 409)	m	-	-	-	-
CUT3DFS	Correction d'outil 3D pour fraisage en bout avec orientation d'outil constante, indépendante du frame actif	*/PGAs/* Activation des corrections d'outil 3D (CUT3DC..., CUT3DF...) (Page 409)	m	-	-	-	-
CUTCONOF [4]	Désactivation de la correction constante de rayon	*/PGs/*	m	●	●	●	●
CUTCONON	Activation de la correction constante de rayon	*/PGs/*	m	●	●	●	●
CUTMOD	Activation de la fonction "Modification des données de correction pour outils indexables"	*/PGAs/* Modification des données de tranchant des outils orientables (CUTMOD) (Page 446)		●	●	●	●
CYCLE...	Cycles de mesure	*/BHDs/BHFs/*					

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
D	Numéro de correction d'outil	*/PGs/*		●	●	●	●
D0	Pour D0, les corrections seront inopérantes pour l'outil en question	*/PGs/*		●	●	●	●
DAC	Programmation au diamètre spécifique à l'axe non modale absolue	*/PGs/*	b	●	●	●	●
DC	Cotation absolue pour axes rotatifs, accoster directement la position	*/PGs/*	b	●	●	●	●
DEF	Définition de variable	*/PGAs/* Définition de variables utilisateur (DEF) (Page 22)		●	●	●	●
DEFINE	Mot-clé pour les définitions de macros	*/PGAs/* Macroprogrammation (DEFINE ... AS) (Page 201)		●	●	●	●
DEFAULT	Branche d'un branchement CASE	*/PGAs/* Saut de programme (CASE ... OF ... DEFAULT ...) (Page 86)		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
DELAYFSTON	Définition du début d'une plage d'arrêt temporisé	*PGAs/* Saut de programme (CASE ... OF ... DEFAULT ...) (Page 86)	m	•	•	•	•
DELAYFSTOF	Définition de la fin d'une plage d'arrêt temporisé	*PGAs/* Sections de programme interruptibles sous conditions (DELAYFSTON, DELAYFSTOF) (Page 468)	m	•	•	•	•
DELDL	Suppression des corrections additives	*PGAs/* Effacer les corrections additives (DELDL) (Page 395)		•	•	•	•
DELDTG	Effacement de la distance restant à parcourir	*PGAs/* Effacement de la distance restant à parcourir (DELDTG) (Page 582)		•	•	•	•
DELETE	Effacement du fichier indiqué. Le nom du fichier peut être indiqué avec chemin d'accès et extension de fichier.	*PGAs/* Suppression d'un fichier (DELETE) (Page 132)		•	•	•	•
DELTOOLENV	Suppression d'enregistrements de description de contournage d'outil	*FB1(W1)*		•	•	•	•
DIACYCOFA	Programmation au diamètre modale spécifique à l'axe : Désactivée dans les cycles	*FB1(P1)*	m	•	•	•	•
DIAM90	Programmation au diamètre pour G90 ; programmation au rayon pour G91	*PGAs/*	m	•	•	•	•
DIAM90A	Programmation au diamètre modale spécifique à l'axe pour G90 et AC, programmation au rayon pour G91 et IC	*PGs/*	m	•	•	•	•
DIAMCHAN	Prise en compte de tous les axes des fonctions d'axe PM dans l'état du canal de la programmation au diamètre.	*PGs/*		•	•	•	•

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
DIAMCHANA	Prise en compte du statut du canal de la programmation au diamètre	*PGs/*		●	●	●	●
DIAMCYCOF	Programmation du diamètre spécifique au canal : Désactivée dans les cycles	*FB1(P1)*	m	●	●	●	●
DIAMOF [4]	Désactivation de la programmation du diamètre Désactivée état initial, voir le constructeur de la machine	*PGs/*	m	●	●	●	●
DIAMOFA	Programmation modale du diamètre spécifique à l'axe : Désactivée état initial, voir le constructeur de la machine	*PGs/*	m	●	●	●	●
DIAMON	Désactivation de la programmation du diamètre activé	*PGs/*	m	●	●	●	●
DIAMONA	Programmation modale du diamètre spécifique à l'axe : Activée validation, voir le constructeur de la machine	*PGs/*	m	●	●	●	●
DIC	Programmation au diamètre spécifique à l'axe non modale relative	*PGs/*	b	●	●	●	●
DILF	Trajectoire de retrait (longueur)	*PGs/*	m	●	●	●	●
DISABLE	Désactivation d'une interruption	*PGAs/* Désactivation/activation de l'affectation d'une routine d'interruption (DISABLE, ENABLE) (Page 113)		●	●	●	●
DISC	Agrandissement arc de raccordement { Correction du rayon d'outil	*PGs/*	m	●	●	●	●
DISCL	Distance entre le point final du mouvement de pénétration rapide et le plan d'usinage	*PGs/*		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
DISPLOF	Inhibition de l'affichage du bloc courant	*PGAs/* Inhibition de l'affichage du bloc courant (DISPLOF, DISPLON, ACTBLOCNO) (Page 164)		●	●	●	●
DISPLON	Annulation de l'inhibition d'affichage du bloc courant	*PGAs/* Inhibition de l'affichage du bloc courant (DISPLOF, DISPLON, ACTBLOCNO) (Page 164)		●	●	●	●
DISPR	Différence sur le contour pour repositionnement	*PGAs/* Réaccostage du contour (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Page 476)	b	●	●	●	●
DISR	Distance de repositionnement	*PGAs/* Réaccostage du contour (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Page 476)	b	●	●	●	●
DITE	Course de freinage en filetage à l'outil	*PGs/*	m	●	●	●	●
DITS	Course d'accélération en filetage à l'outil	*PGs/*	m	●	●	●	●
DIV	Division avec résultat entier	*PGAs/* Fonctions de calcul (Page 61)		●	●	●	●
DL	Sélection de la correction d'outil additive fonction de l'endroit (DL, correction de la somme, mise en place)	*PGAs/* Sélection des corrections additives (DL) (Page 392)	m	-	-	-	-
DO	Mot-clé pour action synchrone, déclenche une action lorsque la condition est remplie	*PGAs/* Actions (DO) (Page 557)		●	●	●	●
DRFOF	Désactivation (effacement) des décalages par manivelle (DRF)	*PGs/*	m	●	●	●	●
DRIVE	Accélération tangentielle fonction de la vitesse	*PGs/*	m	●	●	●	●
DRIVEA	Activation de la courbe d'accélération coudée pour les axes programmés	*PGs/*		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
DYNFINISH	Dynamique pour superfinition	*PGs/*	m	●	●	●	●
DYNNORM	Dynamique normale	*PGs/*	m	●	●	●	●
DYNPOS	Dynamique pour mode positionnement, taraudage	*PGs/*	m	●	●	●	●
DYNROUGH	Dynamique pour ébauche	*PGs/*	m	●	●	●	●
DYNSEMIFIN	Dynamique pour finition	*PGs/*	m	●	●	●	●
DZERO	Déclare tous les numéros D de l'unité TO non valides	*PGAs/* Libre affectation des numéros D : numéros D déclarés non valides (DZERO) (Page 433)		●	●	●	●
EAUTO	Détermination de la dernière section spline avec les 3 derniers points	*PGAs/* Interpolation de type spline (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Page 233)	m	-	○	-	○
EGDEF	Définition d'un réducteur électronique	*PGAs/* Définition d'un réducteur électronique (EGDEF) (Page 526)		-	-	-	-
EGDEL	Suppression de la définition de couplage pour l'axe asservi	*PGAs/* Suppression de la définition d'un réducteur électronique (EGDEL) (Page 532)		-	-	-	-
EGOFC	Désactivation continue de l'entraînement électronique	*PGAs/* Désactivation du réducteur électronique (EGOFS, EGOFC) (Page 531)		-	-	-	-
EGOFS	Désactivation sélective de l'entraînement électronique	*PGAs/* Désactivation du réducteur électronique (EGOFS, EGOFC) (Page 531)		-	-	-	-
EGON	Activation de l'entraînement électronique	*PGAs/* Désactivation du réducteur électronique (EGOFS, EGOFC) (Page 531)		-	-	-	-
EGONSYN	Activation de l'entraînement électronique	*PGAs/* Désactivation du réducteur électronique (EGOFS, EGOFC) (Page 531)		-	-	-	-

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
EGONSYNE	Activation de l'entraînement électronique, avec prédéfinition du mode d'accostage	*PGAs/* Désactivation du réducteur électronique (EGOFS, EGOFC) (Page 531)		-	-	-	-
ELSE	Branchement si condition IF pas remplie	*PGAs/* Boucle de programme avec alternative (IF, ELSE, ENDIF) (Page 96)		●	●	●	●
ENABLE	Activation d'une interruption	*PGAs/* Désactivation/activation de l'affectation d'une routine d'interruption (DISABLE, ENABLE) (Page 113)		●	●	●	●
ENAT [4]	Raccordement naturel avec le bloc de déplacement suivant	*PGAs/* Interpolation de type spline (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Page 233)	m	-	○	-	○
ENDFOR	Dernière ligne d'une boucle de comptage FOR	*PGAs/* Boucle de comptage (FOR ... TO ..., ENDFOR) (Page 99)		●	●	●	●
ENDIF	Dernière ligne d'un branchement IF	*PGAs/* Boucle de programme avec alternative (IF, ELSE, ENDIF) (Page 96)		●	●	●	●
ETIQUETTE_FIN	Repère de fin pour répétitions du programme pièce via REPEAT	*PGAs/, FB1(K1)* Répétition de sections de programme (REPEAT, REPEATB, ENDLABEL, P) (Page 88)		●	●	●	●
ENDLOOP	Dernière ligne d'une boucle sans fin LOOP	*PGAs/* Boucle de programme sans fin (LOOP, ENDLOOP) (Page 98)		●	●	●	●
ENDPROC	Dernière ligne d'un programme commençant par PROC			●	●	●	●
ENDWHILE	Dernière ligne d'une boucle WHILE	*PGAs/* Boucle de programme avec condition en début de boucle (WHILE, ENDWHILE) (Page 100)		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
ETAN	Raccordement tangentiel avec le bloc de déplacement suivant en début de courbe de type spline	*PGAs/* Interpolation de type spline (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Page 233)	m	-	○	-	○
EVERY	Exécution d'une action synchrone lors de la transition de la condition de FALSE à TRUE	*PGAs/* Contrôle cyclique de la condition (WHEN, WHENEVER, FROM, EVERY) (Page 555)		●	●	●	●
EX	Mot-clé pour l'affectation en notation exponentielle	*PGAs/* Variables utilisateur prédéfinies : Paramètre de calcul (R) (Page 18)		●	●	●	●
EXECSTRING	Transfert d'une variable STRING avec la ligne du programme pièce à exécuter	*PGAs/* Programmation indirecte de lignes de programme pièce (EXECSTRING) (Page 60)		●	●	●	●
EXECTAB	Exécution d'un élément d'une table de déplacements	*PGAs/* Programmation indirecte de lignes de programme pièce (EXECSTRING) (Page 60)		●	●	●	●
EXECUTE	Activation de l'exécution du programme	*PGAs/* Désactivation de la préparation du contour (EXECUTE) (Page 717)		●	●	●	●
EXP	fonction exponentielle ex	*PGAs/* Fonctions de calcul (Page 61)		●	●	●	●
EXTCALL	Exécution de sous-programme externe	*PGAs/* Exécution de sous-programme externe (EXTCALL) (Page 193)		●	●	●	●
EXTERN	Déclaration d'un sous-programme avec transfert de paramètres	*PGAs/* Appel de sous-programme sans transfert de paramètres (Page 178)		●	●	●	●
F	Valeur d'avance (l'arrêt temporisé est programmé aussi avec F en liaison avec G4)	*PGs/*		●	●	●	●
FA	Avance axiale	*PGs/*	m	●	●	●	●
FAD	Avance d'approche pour accostage et retrait en douceur	*PGs/*		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
FALSE	Constante logique : faux	*PGAs*/ Définition de variables utilisateur (DEF) (Page 22)		●	●	●	●
FB	Avance non modale	*PGs*/		●	●	●	●
FCTDEF	Définir une fonction polynôme	*PGAs*/ Correction d'outil en ligne (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) (Page 404)		-	-	-	-
FCUB	Avance variable selon une courbe de type spline cubique	*PGAs*/ Variation de l'avance (FNORM, FLIN, FCUB, FPO) (Page 460)	m	●	●	●	●
FD	Avance tangentielle pour correction par manivelle	*PGs*/	b	●	●	●	●
FDA	Avance axiale pour correction par manivelle	*PGs*/	b	●	●	●	●
FENDNORM	Décélération aux angles OFF	*PGAs*/ Réduction de l'avance avec décélération aux angles (FENDNORM, G62, G621) (Page 273)	m	●	●	●	●
FFWOF [4]	Désactivation de la commande anticipatrice	*PGs*/	m	●	●	●	●
FFWON	Activation de la commande anticipatrice	*PGs*/	m	●	●	●	●
FGREF	Rayon de référence pour les axes rotatifs ou facteurs de référence de trajectoire pour les axes d'orientation (interpolation vectorielle)	*PGs*/	m	●	●	●	●
FGROUP	Sélection de l'axe ou des axes avec avance tangentielle	*PGs*/		●	●	●	●
FI	Paramètre d'accès aux données frame : Décalage fin	*PGAs*/ Lecture et modification de composantes de frames (TR, FI, RT, SC, MI) (Page 289)		●	●	●	●
FIFOCTRL	Commande de la mémoire tampon d'exécution	*PGAs*/ Exécution du programme avec une mémoire tampon (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE) (Page 465)	m	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
FILEDATE	Fournit la date du dernier accès en écriture au fichier	*PGAs*/ Lecture des informations fichier (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) (Page 139)		●	●	●	●
FILEINFO	Fournit la somme de FILEDATE, FILESIZE, FILESTAT et FILETIME	*PGAs*/ Lecture des informations fichier (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) (Page 139)		●	●	●	●
FILESIZE	Fournit la taille actuelle du fichier	*PGAs*/ Lecture des informations fichier (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) (Page 139)		●	●	●	●
FILESTAT	Fournit l'état du fichier des droits lecture, écriture, exécution, affichage, suppression (rwxsd)	*PGAs*/ Lecture des informations fichier (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) (Page 139)		●	●	●	●
FILETIME	Fournit l'heure du dernier accès en écriture au fichier	*PGAs*/ Lecture des informations fichier (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) (Page 139)		●	●	●	●
FINEA	Fin du déplacement lors de l'atteinte de "Arrêt précis fin"	*PGAs*/ Critère programmable de fin de déplacement (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) (Page 274)	m	●	●	●	●
FL	Vitesse limite pour axes synchrones	*PGs*/ Lecture et modification de composantes de frames (TR, FI, RT, SC, MI) (Page 289)	m	●	●	●	●
FLIN	avance variable linéairement	*PGAs*/ Variation de l'avance (FNORM, FLIN, FCUB, FPO) (Page 460)	m	●	●	●	●
FMA	Plusieurs avances axiales	*PGs*/	m	-	-	-	-
FNORM [4]	avance normale selon DIN66025	*PGAs*/ Variation de l'avance (FNORM, FLIN, FCUB, FPO) (Page 460)	m	●	●	●	●
FOCOF	Désactivation du déplacement avec réduction du couple/force	*PGAs*/ Accostage d'une butée (FXS, FXST, FXSW, FOCON, FOCOF) (Page 620)	m	○	-	○	-

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
FOCON	Activation du déplacement avec réduction du couple/force	*PGAs/* Accostage d'une butée (FXS, FXST, FXSW, FOCON, FOCOF) (Page 620)	m	○	-	○	-
FOR	Boucle de comptage avec nombre fixe d'itérations	*PGAs/* Boucle de comptage (FOR ... TO ..., ENDFOR) (Page 99)		●	●	●	●
FP	Point fixe : numéro du point fixe à accoster	*PGs/*	b	●	●	●	●
FPO	Variation de l'avance programmée par le biais d'un polynôme	*PGAs/* Variation de l'avance (FNORM, FLIN, FCUB, FPO) (Page 460)		-	-	-	-
FPR	Identification axe rotatif	*PGs/*		●	●	●	●
FPRAOF	Désactivation de l'avance par tour	*PGs/*		●	●	●	●
FPRAON	Activation de l'avance par tour	*PGs/*		●	●	●	●
FRAME	Type de données pour la définition des systèmes de coordonnées	*PGAs/* Définition de nouveaux frames (DEF FRAME) (Page 292)		●	●	●	●
FRC	Avance pour rayon et chanfrein	*PGs/*	b	●	●	●	●
FRCM	Avance modale pour rayon et chanfrein	*PGs/*	m	●	●	●	●
FROM	L'action est exécutée lorsque la condition est remplie une fois et tant que l'action synchrone est active	*PGAs/* Contrôle cyclique de la condition (WHEN, WHENEVER, FROM, EVERY) (Page 555)		●	●	●	●
FTOC	Modifier la correction d'outil fine	*PGs/* Correction d'outil en ligne (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) (Page 404)		●	●	●	●
FTOCOF [4]	Désactivation de la correction d'outil fine active en ligne	*PGAs/* Correction d'outil en ligne (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) (Page 404)	m	●	●	●	●
FTOCON	Activation de la correction d'outil fine active en ligne	*PGAs/* Correction d'outil en ligne (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) (Page 404)	m	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
FXS	Déplacement en butée activé	*PGs*/ Accostage d'une butée (FXS, FXST, FXSW, FOCON, FOCOF) (Page 620)	m	●	●	●	●
FXST	Limite de couple pour l'accostage de butée	*PGs*/ Accostage d'une butée (FXS, FXST, FXSW, FOCON, FOCOF) (Page 620)	m	●	●	●	●
FXSW	Fenêtre de surveillance pour l'accostage de butée	*PGs*/ Accostage d'une butée (FXS, FXST, FXSW, FOCON, FOCOF) (Page 620)		●	●	●	●
FZ	Avance	*PGs*/	m	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
G0	Interpolation linéaire en vitesse rapide (déplacement à vitesse rapide)	*PGs*/	m	●	●	●	●
G1 [4]	Interpolation linéaire avec avance	*PGs*/	m	●	●	●	●
G2	Interpolation circulaire sens horaire	*PGs*/	m	●	●	●	●
G3	Interpolation circulaire sens antihoraire	*PGs*/	m	●	●	●	●
G4	Arrêt temporisé, prédéterminé dans le temps	*PGs*/	b	●	●	●	●
G5	Rectification en plongée oblique	*PGAs*/ Axe oblique (TRAANG) (Page 371)	b	●	●	●	●
G7	Mouvement compensatoire pour la rectification en plongée oblique	*PGAs*/ Axe oblique (TRAANG) (Page 371)	b	●	●	●	●
G9	Arrêt précis - réduction de vitesse	*PGs*/	b	●	●	●	●
G17 [4]	Sélection du plan de travail X/Y	*PGs*/	m	●	●	●	●
G18	Sélection du plan de travail Z/X	*PGs*/	m	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
G19	Sélection du plan de travail Y/Z	*PGs/*	m	•	•	•	•
G25	Limitation inférieure de la zone de travail	*PGs/*	b	•	•	•	•
G26	Limitation supérieure de la zone de travail	*PGs/*	b	•	•	•	•
G33	Filetage à pas constant	*PGs/*	m	•	•	•	•
G34	Filetage avec pas en progression linéaire	*PGs/*	m	•	•	•	•
G35	Filetage avec pas en régression linéaire	*PGs/*	m	•	•	•	•
G40 [4]	Désactivation de la correction du rayon de l'outil	*PGs/*	m	•	•	•	•
G41	Correction du rayon de l'outil à gauche du contour	*PGs/*	m	•	•	•	•
G42	Correction du rayon de l'outil à droite du contour	*PGs/*	m	•	•	•	•
G53	Inhibition du décalage d'origine courant (non modal)	*PGs/*	b	•	•	•	•
G54	1er décalage d'origine réglable	*PGs/*	m	•	•	•	•
G55	2. Décalage d'origine réglable	*PGs/*	m	•	•	•	•
G56	3. Décalage d'origine réglable	*PGs/*	m	•	•	•	•
G57	4. Décalage d'origine réglable	*PGs/*	m	•	•	•	•
G58	Décalage d'origine programmable axial absolu, décalage grossier	*PGs/*	b	•	•	•	•
G59	Décalage d'origine programmable axial additif, décalage fin	*PGs/*	b	•	•	•	•
G60 [4]	Arrêt précis - réduction de vitesse	*PGs/*	m	•	•	•	•
G62	Décélération aux angles intérieurs avec correction du rayon d'outil activée (G41, G42)	*PGAs/* Réduction de l'avance avec décélération aux angles (FENDNORM, G62, G621) (Page 273)	m	•	•	•	•

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
G63	Taraudage avec porte-taraud compensateur	*PGs/*	b	●	●	●	●
G64	Contournage	*PGs/*	m	●	●	●	●
G70	Cotation en pouce pour indications géométriques (longueurs)	*PGs/*	m	●	●	●	●
G71 [4]	Cotation métrique pour indications géométriques (longueurs)	*PGs/*	m	●	●	●	●
G74	Accostage point de référence	*PGs/*	b	●	●	●	●
G75	Accostage d'un point fixe	*PGs/*	b	●	●	●	●
G90 [4]	Introduction des cotes en valeurs absolues	*PGs/*	m/b	●	●	●	●
G91	Indication de cotes relatives	*PGs/*	m/b	●	●	●	●
G93	Avance en sens inverse du temps tr/min	*PGs/*	m	●	●	●	●
G94 [4]	Avance linéaire F en mm/min ou pouce/min et degré/min	*PGs/*	m	●	●	●	●
G95	Avance par tour F en mm/tr, inch/tr	*PGs/*	m	●	●	●	●
G96	Activation de la vitesse de coupe constante (comme G95)	*PGs/*	m	●	●	●	●
G97	Désactivation de la vitesse de coupe constante (comme G95)	*PGs/*	m	●	●	●	●
G110	Programmation du pôle par rapport à la dernière position de consigne programmée	*PGs/*	b	●	●	●	●
G111	Programmation du pôle par rapport à l'origine pièce courante	*PGs/*	b	●	●	●	●
G112	Programmation du pôle par rapport au dernier pôle valable	*PGs/*	b	●	●	●	●
G140 [4]	Sens d'accostage WAB fixé par G41/G42	*PGs/*	m	●	●	●	●
G141	Sens d'accostage WAB à gauche du contour	*PGs/*	m	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
G142	Sens d'accostage WAB à droite du contour	*PGs/*	m	•	•	•	•
G143	Sens d'accostage WAB en fonction tangente	*PGs/*	m	•	•	•	•
G147	Accostage en douceur en ligne droite	*PGs/*	b	•	•	•	•
G148	Retrait en douceur en ligne droite	*PGs/*	b	•	•	•	•
G153	Inhibition des frames courants, y compris le frame de base	*PGs/*	b	•	•	•	•
G247	Accostage en douceur en quart de cercle	*PGs/*	b	•	•	•	•
G248	Retrait en douceur en quart de cercle	*PGs/*	b	•	•	•	•
G290	Basculement vers mode SINUMERIK en MARCHE	*FBW*	m	•	•	•	•
G291	Basculement vers mode ISO2/3 en MARCHE	*FBW*	m	•	•	•	•
G331	Taraudage sans porte-taraud compensateur, pas positif, filetage à droite	*PGs/*	m	•	•	•	•
G332	Taraudage sans porte-taraud compensateur, pas négatif, filetage à gauche	*PGs/*	m	•	•	•	•
G340 [4]	Accostage dans l'espace (simultanément en profondeur et dans le plan (hélice))	*PGs/*	m	•	•	•	•
G341	D'abord approche dans l'axe perpendiculaire (z), puis accostage dans le plan	*PGs/*	m	•	•	•	•
G347	Accostage en douceur en demi-cercle	*PGs/*	b	•	•	•	•
G348	Retrait en douceur en demi-cercle	*PGs/*	b	•	•	•	•
G450 [4]	arc de raccordement	*PGs/*	m	•	•	•	•
G451	Point d'intersection des équidistantes	*PGs/*	m	•	•	•	•

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
G460 [4]	Activation de la détection des violations du contour pour le bloc d'accostage et de retrait	*PGs/*	m	•	•	•	•
G461	Insertion d'un cercle dans le bloc CRO	*PGs/*	m	•	•	•	•
G462	Insertion d'une droite dans le bloc CRO	*PGs/*	m	•	•	•	•
G500 [4]	Désactivation de tous les frames réglables, les frames de base sont actifs	*PGs/*	m	•	•	•	•
G505 ... G599	5 ... 99. Décalage d'origine réglable	*PGs/*	m	•	•	•	•
G601 [4]	Changement de bloc lors arrêt précis fin	*PGs/*	m	•	•	•	•
G602	Changement de bloc lors arrêt précis grossier	*PGs/*	m	•	•	•	•
G603	Changement de bloc en fin de bloc IPO	*PGs/*	m	•	•	•	•
G621	Décélération à tous les angles	*PGAs/* Réduction de l'avance avec décélération aux angles (FENDNORM, G62, G621) (Page 273)	m	•	•	•	•
G641	Contournage avec arrondissement selon le critère de trajet (= distance de transition programmable)	*PGs/*	m	•	•	•	•
G642	Contournage avec arrondissement avec prise en compte des tolérances définies	*PGs/*	m	•	•	•	•
G643	Contournage avec arrondissement avec prise en compte des tolérances définies (internes au bloc)	*PGs/*	m	•	•	•	•
G644	Contournage avec arrondissement utilisant la dynamique maximale possible	*PGs/*	m	•	•	•	•

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
G645	Contournage avec arrondissement d'angles et changements de blocs tangentiels avec prise en compte de tolérances définies	*PGsl*	m	•	•	•	•
G700	Cotation en pouce pour indications géométriques et technologiques (longueurs, avance)	*PGsl*	m	•	•	•	•
G710 [4]	Cotation métrique pour indications géométriques et technologiques (longueurs, avance)	*PGsl*	m	•	•	•	•
G751	Accostage d'un point fixe via un point intermédiaire	PGsl	b	•	•	•	•
G810 [4], ..., G819	Groupe G réservé pour l'utilisateur OEM	*PGAsl* Fonctions spéciales pour l'utilisateur OEM (OEMIPO1, OEMIPO2, G810 à G829) (Page 272)		•	•	•	•
G820 [4], ..., G829	Groupe G réservé pour l'utilisateur OEM	*PGAsl* Fonctions spéciales pour l'utilisateur OEM (OEMIPO1, OEMIPO2, G810 à G829) (Page 272)		•	•	•	•
G931	Indication de l'avance par durée du déplacement		m	•	•	•	•
G942	Gel de l'avance linéaire et de la vitesse de coupe constante ou de la vitesse de rotation de broche		m	•	•	•	•
G952	Gel de l'avance par tour et de la vitesse de coupe constante ou de la vitesse de rotation de broche		m	•	•	•	•
G961	vitesse de coupe constante et avance linéaire	*PGsl*	m	•	•	•	•
G962	Avance linéaire ou avance par tour et vitesse de coupe constante	*PGsl*	m	•	•	•	•
G971	Geler la vitesse de broche et avance linéaire	*PGsl*	m	•	•	•	•

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
G972	Gel de l'avance linéaire ou de l'avance par tour et de la vitesse de rotation de broche constante	*/PGs/*	m	●	●	●	●
G973	Avance rotative sans limitation de vitesse de la broche	*/PGs/*	m	●	●	●	●
GEOAX	Affecter de nouveaux axes de canal aux axes géométriques 1 à 3	*/PGAs/* Axes géométriques permutables (GEOAX) (Page 672)		●	●	●	●
GET	Echange de l'axe libéré entre des canaux	*/PGAs/* Permutation d'axe, permutation de broche (RELEASE, GET, GETD) (Page 121)		●	●	●	●
GETACTT	Détermination de l'outil actif parmi un groupe d'outils de même nom	*/FBW/*		●	●	●	●
GETACTTD	Détermination du numéro T correspondant à un numéro D absolu	*/PGAs/* Libre affectation des numéros D : détermination du numéro T correspondant au numéro D prescrit (GETACTTD) (Page 432)		●	●	●	●
GETD	Echange direct de l'axe entre des canaux	*/PGAs/* Permutation d'axe, permutation de broche (RELEASE, GET, GETD) (Page 121)		●	●	●	●
GETDNO	Fournit le numéro D d'un tranchant (CE) d'une outil (T)	*/PGAs/* Libre affectation des numéros D : modification des numéros D (GETDNO, SETDNO) (Page 431)		●	●	●	●
GETEXET	Lecture du numéro T changé	*/FBW/*		●	●	●	●
GETFREELOC	Recherche d'un emplacement vide dans les magasins pour un outil donné	*/FBW/*		●	●	●	●
GETSELT	Délivrer le numéro T présélectionné	*/FBW/*		●	●	●	●
GETT	Déterminer le numéro T correspondant à un nom d'outil	*/FBW/*		●	●	●	●
GETTCOR	Lecture de longueurs d'outil ou de composantes de longueurs d'outil	*/FB1(W1)/*		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
GETTENV	Lecture des numéros T, D et DL	*FB1(W1)*		•	•	•	•
GOTO	Instruction de saut avec destination en aval puis en amont (vers fin puis vers début de programme)	*PGAsl* Sauts de programme sur repères de saut (IF, GOTOB, GOTOF, GOTO, GOTOC) (Page 83)		•	•	•	•
GOTOB	Saut avec destination en amont (vers début de programme)	*PGAsl* Sauts de programme sur repères de saut (IF, GOTOB, GOTOF, GOTO, GOTOC) (Page 83)		•	•	•	•
GOTOC	Comme GOTO, mais avec suppression de l'alarme 14080 "Destination de saut pas trouvée"	*PGAsl* Sauts de programme sur repères de saut (IF, GOTOB, GOTOF, GOTO, GOTOC) (Page 83)		•	•	•	•
GOTOF	Saut avec destination en aval (vers fin programme)	*PGAsl* Sauts de programme sur repères de saut (IF, GOTOB, GOTOF, GOTO, GOTOC) (Page 83)		•	•	•	•
GOTOS	Retour au début du programme	*PGAsl* Retour au début du programme (GOTOS) (Page 82)		•	•	•	•
GP	Mot clé pour la programmation indirecte d'attributs de positions	*PGAsl* Programmation indirecte d'attributs de position (PB) (Page 57)		•	•	•	•
GWPSOF	Désactiver vitesse périphérique de meule constante (VPM)	*PGsl*	b	•	•	•	•
GWPSON	Activer vitesse périphérique de meule constante (VPM)	*PGsl*	b	•	•	•	•
H...	Sortie de fonction auxiliaire à l'AP	*PGsl/FB1(H2)*		•	•	•	•
HOLES1	Cycle de taraudage, rangée de trous	*BHDsl/BHFsl*		•	•	•	•
HOLES2	Cycle de taraudage, cercle de trous	*BHDsl/BHFsl*		•	•	•	•
I	Paramètres d'interpolation	*PGsl*	b	•	•	•	•
I1	Coordonnée point intermédiaire	*PGsl*	b	•	•	•	•

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
IC	Saisie de cotes relatives	*PGs/*	b	●	●	●	●
ICYCOF	Exécuter tous les blocs d'un cycle technologique dans une période d'appel de l'interpolateur après ICYCOF.	*PGAs/* Pilotage du traitement de cycles technologiques (ICYCOF, ICYCON) (Page 631)		●	●	●	●
ICYCON	Traitement de chaque bloc d'un cycle technologique dans une période d'appel de l'interpolateur après ICYCON	*PGAs/* Pilotage du traitement de cycles technologiques (ICYCOF, ICYCON) (Page 631)		●	●	●	●
ID	Identification pour actions synchrones modales	*PGAs/* Domaine de validité et ordre d'exécution (ID, IDS) (Page 553)	m	●	●	●	●
IDS	Identification pour actions synchrones statiques modales	*PGAs/* Domaine de validité et ordre d'exécution (ID, IDS) (Page 553)		●	●	●	●
IF	Début d'un saut conditionnel dans le programme pièce / cycle technologique	*PGAs/* Boucle de programme avec alternative (IF, ELSE, ENDIF) (Page 96)		●	●	●	●
INDEX	Déterminer l'indice d'un caractère dans une chaîne de caractères	*PGAs/* Recherche de caractères/chaînes de caractères dans une chaîne (INDEX, RINDEX, MINDEX, MATCH) (Page 78)		●	●	●	●
INIPO	Initialisation des variables pour PowerOn	*PGAs/* Définition de variables utilisateur (DEF) (Page 22)		●	●	●	●
INIRE	Initialisation des variables pour Reset	*PGAs/* Définition de variables utilisateur (DEF) (Page 22)		●	●	●	●
INICF	Initialisation des variables pour NewConfig	*PGAs/* Définition de variables utilisateur (DEF) (Page 22)		●	●	●	●
INIT	Sélection d'un programme CN spécifié pour exécution dans un canal déterminé	*PGAs/* Coordination de programmes (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) (Page 103)		-	-	-	-

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
INITIAL	Génération d'un fichier INI pour tous les domaines	*PGAs/* Mémoire de travail (CHANDATA, COMPLETE, INITIAL) (Page 210)		●	●	●	●
INT	Type de données : valeur entière avec signe	*PGAs/* Définition de variables utilisateur (DEF) (Page 22)		●	●	●	●
INTERSEC	Calculer le point d'intersection entre deux éléments de contour	*PGAs/* Détermination de l'intersection entre deux éléments de contour (INTERSEC) (Page 712)		●	●	●	●
INVCCW	Déplacer développante dans le sens anti-horaire	*PGs/*	m	-	-	-	-
INVCW	Déplacer développante dans le sens horaire	*PGs/*	m	-	-	-	-
INVFRAME	Calculer le frame inverse à partir d'un frame	*FB1(K2)*		●	●	●	●
IP	Paramètre d'interpolation variable	*PGAs/* Programmation indirecte (Page 53)		●	●	●	●
IPOBRKA	Critère de déplacement à partir du point de déclenchement de la rampe de freinage	*PGAs/* Critère programmable de fin de déplacement (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) (Page 274)	m	●	●	●	●
IPOENDA	Fin du déplacement lors de "Arrêt interpolateur"	*PGAs/* Critère programmable de fin de déplacement (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) (Page 274)	m	●	●	●	●
IPTRLOCK	Gel du début de la section de programme interdit à la recherche sur le bloc de fonction machine suivant.	*PGAs/* Interdire des positions de programme pour SERUPRO (IPTRLOCK, IPTRUNLOCK) (Page 473)	m	●	●	●	●
IPTRUNLOCK	Définition de la fin de la section de programme interdit à la recherche sur le bloc courant du moment d'interruption.	*PGAs/* Interdire des positions de programme pour SERUPRO (IPTRLOCK, IPTRUNLOCK) (Page 473)	m	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
ISAXIS	Vérifier si l'axe géométrique 1 indiqué comme paramètre existe	*PGAs/* Fonctions d'axe (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) (Page 669)		●	●	●	●
ISD	Profondeur de pénétration	*PGAs/* Activation des corrections d'outil 3D (CUT3DC, CUT3DF, CUT3DFS, CUT3DFF, ISD) (Page 409)	m	●	●	●	●
ISFILE	Vérifier s'il existe un fichier dans la mémoire d'application NCK	*PGAs/* Vérification de l'existence d'un fichier (ISFILE) (Page 137)		●	●	●	●
ISNUMBER	Vérifier si une chaîne de caractères peut être convertie en un nombre	*PGAs/* Conversion de type à partir de STRING (NUMBER, ISNUMBER, AXNAME) (Page 74)		●	●	●	●
ISOCALL	Appel indirect d'un programme programmé en langage ISO	*PGAs/* Appel indirect d'un programme programmé en langage ISO (ISOCALL) (Page 190)		●	●	●	●
ISVAR	Vérifier si le paramètre de transfert contient une variable connue de la CN	*PGAs/* Lecture de l'appel de la fonction ISVAR et des paramètres machine d'indice de tableau (Page 687)		●	●	●	●
J	Paramètres d'interpolation	*PGs/*	b	●	●	●	●
J1	Coordonnée point intermédiaire	*PGs/*	b	●	●	●	●
JERKA	Activer, pour les axes programmés, le comportement à l'accélération réglé à l'aide du PM			●	●	●	●
JERKLIM	Réduction ou accroissement de l'à-coup axial maximal	*PGAs/* Correction de l'à-coup en pourcentage (JERKLIM) (Page 486)	m	●	●	●	●
JERKLIMA	Réduction ou accroissement de l'à-coup axial maximal	*PGs/*	m	●	●	●	●
K	Paramètres d'interpolation	*PGs/*	b	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
K1	Coordonnée point intermédiaire	*PGsI*	b	●	●	●	●
KONT	Contournement du contour lors de la correction d'outil	*PGsI*	m	●	●	●	●
KONTC	Accostage / retrait avec polynôme à courbure continue	*PGsI*	m	●	●	●	●
KONTT	Accostage / retrait avec polynôme tangentiel	*PGsI*	m	●	●	●	●
L	Numéro de sous-programme	*PGAsI* Appel de sous-programme sans transfert de paramètres (Page 178)	b	●	●	●	●
LEAD	Angle d'avance 1. Orientation de l'outil 2. Polynôme d'orientation	*PGAsI* Programmation de l'orientation de l'outil (A..., B..., C..., LEAD, TILT) (Page 323)	m	 ● -	 ● -	 ● -	 ● -
LEADOF	Couplage organe pilote désactivé	*PGAsI* Couplage de deux axes par valeur pilote (LEADON, LEADOF) (Page 520)		-	-	-	-
LEADON	Couplage organe pilote activé	*PGAsI* Couplage de deux axes par valeur pilote (LEADON, LEADOF) (Page 520)		-	-	-	-
LENTOAX	Fournit des informations sur l'affectation des longueurs L1, L2 et L3 de l'outil actif à l'abscisse, à l'ordonnée et à la valeur	*FB1(W1)*		●	●	●	●
LFOF [4]	Retrait rapide pour le filetage à l'outil désactivé	*PGsI*	m	●	●	●	●
LFON	Retrait rapide pour le filetage à l'outil activé	*PGsI*	m	●	●	●	●
LFPOS	Retrait de l'axe introduit avec POLFMASK ou POLFMLIN sur la position d'axe absolue programmée avec POLF	*PGsI*	m	●	●	●	●
LFTXT	Plan du déplacement de retrait rapide déterminé par la tangente de la trajectoire et la direction actuelle de l'outil	*PGsI*	m	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
LFWP	Plan du déplacement de retrait rapide déterminé par le plan de travail actuel (G17/G18/G19)	*PGs/*	m	●	●	●	●
LIFTFAST	Retrait rapide	*PGs/* Retrait rapide du contour (SETINT LIFTFAST, ALF) (Page 115)		●	●	●	●
LIMS	Limitation de la vitesse de rotation pour G96/G961 et G97	*PGs/*	m	●	●	●	●
LLI	Valeur limite inférieure de variables	*PGAs/* Attribut : valeurs limites (LLI, ULI) (Page 34)		●	●	●	●
LN	Logarithme naturel	*PGAs/* Fonctions de calcul (Page 61)		●	●	●	●
LOCK	Bloquer une action synchrone avec ID (bloquer un cycle technologique)	*PGAs/* Bloquer, débloquer, remettre à zéro (LOCK, UNLOCK, RESET) (Page 634)		●	●	●	●
LONGHOLE	Cycle de fraisage, trous oblongs radiaux	*BHDs//BHFs/*		-	-	-	-
LOOP	Début d'une boucle sans fin	*PGAs/* Boucle de programme sans fin (LOOP, ENDLOOP) (Page 98)		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
M0	Arrêt programmé	*PGs/*		●	●	●	●
M1	Arrêt facultatif	*PGs/*		●	●	●	●
M2	Fin du programme principal avec retour au début du programme	*PGs/*		●	●	●	●
M3	Sens de rotation de broche à droite	*PGs/*		●	●	●	●
M4	Sens de rotation de broche à gauche	*PGs/*		●	●	●	●
M5	Arrêt de la broche	*PGs/*		●	●	●	●
M6	Changement d'outil	*PGs/*		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
M17	Fin de sous-programme	*PGs/*		•	•	•	•
M19	Positionnement de la broche sur la position inscrite dans le SD43240	*PGs/*		•	•	•	•
M30	Fin du programme, comme M2	*PGs/*		•	•	•	•
M40	Changement automatique de rapport de boîte de vitesses	*PGs/*		•	•	•	•
M41 ... M45	Rapport de boîte de vitesses 1 ... 5	*PGs/*		•	•	•	•
M70	Basculement dans mode axe	*PGs/*		•	•	•	•
MASLDEF	Définition du couplage d'axes maître/esclave	*PGAs/* Couplage maître/esclave (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) (Page 546)		•	•	•	•
MASLDEL	Découplage maître/esclave et suppression de la définition du couplage	*PGAs/* Couplage maître/esclave (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) (Page 546)		•	•	•	•
MASLOF	Arrêter un couplage temporaire	*PGAs/* Couplage maître/esclave (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) (Page 546)		•	•	•	•
MASLOFS	Désactivation d'un couplage temporaire avec arrêt automatique de l'axe esclave	*PGAs/* Couplage maître/esclave (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) (Page 546)		•	•	•	•
MASLON	Enclencher un couplage temporaire	*PGAs/* Couplage maître/esclave (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) (Page 546)		•	•	•	•
MATCH	Rechercher une chaîne dans une chaîne	*PGAs/* Recherche de caractères/chaînes de caractères dans une chaîne (INDEX, RINDEX, MINDEX, MATCH) (Page 78)		•	•	•	•

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
MAXVAL	Valeur supérieure de deux variables (fonction arithmétique)	*PGAs*/ Minimum, maximum et plage de variables (MINVAL, MAXVAL, BOUND) (Page 68)		●	●	●	●
MCALL	Appel modal d'un sous-programme	*PGAs*/ Appel modal d'un sous-programme (MCALL) (Page 185)		●	●	●	●
MEAC	Mesure sans effacement de la distance restant à parcourir	*PGAs*/ Fonction de mesure étendue (MEASA, MEAWA, MEAC) (option) (Page 262)	b	-	-	-	-
MEAFRAME	Calcul d'un frame à partir de points mesurés	*PGAs*/ Calcul d'un frame à partir de 3 points mesurés dans l'espace (MEAFRAME) (Page 298)		●	●	●	●
MEAS	Mesure avec palpeur à déclenchement	*PGAs*/ Mesure avec palpeur à déclenchement (MEAS, MEAW) (Page 259)	b	●	●	●	●
MEASA	Mesure avec effacement de la distance restant à parcourir	*PGAs*/ Fonction de mesure étendue (MEASA, MEAWA, MEAC) (option) (Page 262)	b	-	-	-	-
MEASURE	Méthode de calcul pour la mesure de pièce et la mesure d'outil	*FB2(M5)* Mesure avec palpeur à déclenchement (MEAS, MEAW) (Page 259)		●	●	●	●
MEAW	Mesure avec palpeur à déclenchement sans suppression de la distance restant à parcourir	*PGAs*/ Mesure avec palpeur à déclenchement (MEAS, MEAW) (Page 259)	b	●	●	●	●
MEAWA	Mesure sans effacement de la distance restant à parcourir	*PGAs*/ Fonction de mesure étendue (MEASA, MEAWA, MEAC) (option) (Page 262)	b	-	-	-	-
MI	Accès aux données frame : Fonction miroir	*PGAs*/ Lecture et modification de composantes de frames (TR, FI, RT, SC, MI) (Page 289)		●	●	●	●
MINDEX	Déterminer l'indice d'un caractère dans une chaîne de caractères	*PGAs*/ Recherche de caractères/chaînes de caractères dans une chaîne (INDEX, RINDEX, MINDEX, MATCH) (Page 78)		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
MINVAL	Valeur inférieure de deux variables (fonction trigonométrique)	*PGAs*/ Minimum, maximum et plage de variables (MINVAL, MAXVAL, BOUND) (Page 68)		●	●	●	●
MIRROR	Fonction miroir programmable	*PGAs*/	b	●	●	●	●
MMC	Appel interactif de boîtes de dialogue à partir du programme pièce pour affichage sur l'IHM	*PGAs*/ Appel interactif des fenêtres depuis le programme pièce (MMC) (Page 691)		●	●	●	●
MOD	Division modulo	*PGAs*/ Fonctions de calcul (Page 61)		●	●	●	●
MODAXVAL	Détermination de la position modulo d'un axe rotatif modulo	*PGAs*/ Fonctions d'axe (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) (Page 669)		●	●	●	●
MOV	Déplacer axe de positionnement	*PGAs*/ Démarrer/arrêter un axe (MOV) (Page 602)		●	●	●	●
MSG	Messages programmables	*PGs*/	m	●	●	●	●
MVTOOL	Instruction de langage pour le déplacement d'un outil	*FBW*		●	●	●	●
N	Numéro de bloc CN complémentaire	*PGs*/		●	●	●	●
NCK	Spécification du domaine de validité de données	*PGAs*/ Définition de variables utilisateur (DEF) (Page 22)		●	●	●	●
NEWCONF	Application des paramètres machine modifiés (correspond à "Activation des paramètres machine")	*PGAs*/ Activation des paramètres machine (NEWCONF) (Page 128)		●	●	●	●
NEWT	Créer un nouvel outil	*PGAs*/ Lecture et modification de composantes de frames (TR, FI, RT, SC, MI) (Page 289)		●	●	●	●
NORM [4]	Positionnement à la normale au point de départ / final en correction d'outil	*PGs*/	m	●	●	●	●
NOT	NON logique (Negation)	*PGAs*/ Opérations relationnelles et opérations logiques (Page 64)		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
NPROT	Activation/désactivation d'une zone de protection spécifique à la machine	*PGAs*/ Activation/désactivation des zones de protection (CPROT, NPROT) (Page 219)		●	●	●	●
NPROTDEF	Définition d'une zone de protection spécifique à la machine	*PGAs*/ Définition des zones de protection (CPROTDEF, NPROTDEF) (Page 215)		●	●	●	●
NUMBER	Convertir une chaîne de caractères en un nombre	*PGAs*/ Conversion de type à partir de STRING (NUMBER, ISNUMBER, AXNAME) (Page 74)		●	●	●	●
OEMIPO1	Interpolation OEM 1	*PGAs*/ Fonctions spéciales pour l'utilisateur OEM (OEMIPO1, OEMIPO2, G810 à G829) (Page 272)	m	●	●	●	●
OEMIPO2	Interpolation OEM 2	*PGAs*/ Fonctions spéciales pour l'utilisateur OEM (OEMIPO1, OEMIPO2, G810 à G829) (Page 272)	m	●	●	●	●
OF	Mot-clé d'un branchement CASE	*PGAs*/ Saut de programme (CASE ... OF ... DEFAULT ...) (Page 86)		●	●	●	●
OFFN	Surépaisseur d'usinage par rapport au contour programmé	*PGs*/	m	●	●	●	●
OMA1	Adresse OEM 1		m	●	●	●	●
OMA2	Adresse OEM 2		m	●	●	●	●
OMA3	Adresse OEM 3		m	●	●	●	●
OMA4	Adresse OEM 4		m	●	●	●	●
OMA5	Adresse OEM 5		m	●	●	●	●
OR	Opérateur logique, combinaison OU	*PGAs*/ Opérations relationnelles et opérations logiques (Page 64)		●	●	●	●
ORIAXES	Interpolation linéaire des axes machine ou des axes d'orientation	*PGAs*/ Programmation des axes d'orientation (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) (Page 333)	m	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
ORIAXPOS	Angles d'orientation définis par axes virtuels d'orientation avec positions d'axe rotatif		m	•	•	•	•
ORIC [4]	les changements d'orientation aux angles saillants se font pendant l'exécution du bloc circulaire inséré	*PGAsl* Orientation de l'outil (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) (Page 423)	m	•	•	•	•
ORICONCCW	Interpolation sur une enveloppe de cercle dans le sens antihoraire	*PGAsl/FB3(F3)* Programmation de l'orientation le long d'une enveloppe de cône (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) (Page 335)	m	•	•	•	•
ORICONCW	Interpolation sur une enveloppe de cercle dans le sens horaire	*PGAsl/FB3(F4)* Programmation de l'orientation le long d'une enveloppe de cône (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) (Page 335)	m	•	•	•	•
ORICONIO	Interpolation sur une enveloppe de cercle avec indication d'une orientation intermédiaire	*PGAsl/FB3(F4)* Programmation de l'orientation le long d'une enveloppe de cône (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) (Page 335)	m	•	•	•	•
ORICONTO	Interpolation sur une surface circulaire avec transition tangentielle (indication de l'orientation finale)	*PGAsl/FB3(F5)* Programmation de l'orientation le long d'une enveloppe de cône (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) (Page 335)	m	•	•	•	•
ORICURVE	Interpolation de l'orientation avec indication du mouvement de deux points de contact de l'outil	*PGAsl/FB3(F6)* Définition de l'orientation de deux points de contact (ORICURVE, PO[XH]=, PO[YH]=, PO[ZH]=) (Page 339)	m	•	•	•	•
ORID	les changements d'orientation se font avant l'exécution du bloc circulaire inséré	*PGAsl* Orientation de l'outil (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) (Page 423)	m	•	•	•	•

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
ORIEULER	Orientation définie par angles d'Euler	*PGAs*/ Programmation des axes d'orientation (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) (Page 333)	m	●	●	●	●
ORIMKS	Orientation d'outil dans le système de coordonnées machine	*PGAs*/ Référence des axes d'orientation (ORIWKS, ORIMKS) (Page 331)	m	●	●	●	●
ORIPATH	Orientation de l'outil par rapport à la trajectoire	*PGAs*/ Rotation de l'orientation de l'outil relative à la trajectoire (ORIPATH, ORIPATHS, angle de rotation) (Page 348)	m	●	●	●	●
ORIPATHS	Orientation de l'outil par rapport à la trajectoire, lissage de coude dans le tracé d'orientation	*PGAs*/ Rotation de l'orientation de l'outil relative à la trajectoire (ORIPATH, ORIPATHS, angle de rotation) (Page 348)	m	●	●	●	●
ORIPLANE	Interpolation dans un plan (correspond à ORIVECT) Interpolation circulaire de grand rayon	*PGAs*/ Programmation de l'orientation le long d'une enveloppe de cône (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) (Page 335)	m	●	●	●	●
ORIRESET	Position initiale de l'orientation de l'outil avec jusqu'à 3 axes d'orientation	*PGAs*/ Variantes de programmation d'orientation et position initiale (ORIRESET) (Page 321)		●	●	●	●
ORIROTA	Angle de rotation rapporté à un sens de rotation en indication absolue	*PGAs*/ Rotations de l'orientation de l'outil (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) (Page 343)	m	●	●	●	●
ORIROTC	Vecteur de rotation tangentiel à la tangente à la trajectoire	*PGAs*/ Rotations de l'orientation de l'outil (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) (Page 343)	m	●	●	●	●
ORIROTR	Angle de rotation relatif par rapport au plan entre orientation de départ et orientation finale	*PGAs*/ Rotations de l'orientation de l'outil (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) (Page 343)	m	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
ORIROTT	Angle de rotation relatif par rapport à la modification du vecteur d'orientation	*PGAs*/ Rotations de l'orientation de l'outil (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) (Page 343)	m	•	•	•	•
ORIRPY	Orientation définie par angles RPY (XYZ)	*PGAs*/ Programmation des axes d'orientation (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) (Page 333)	m	•	•	•	•
ORIRPY2	Orientation définie par angles RPY (ZYX)	*PGAs*/ Programmation des axes d'orientation (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) (Page 333)	m	•	•	•	•
ORIS	Modification d'orientation	*PGAs*/ Orientation de l'outil (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) (Page 423)	m	•	•	•	•
ORISOF [4]	Lissage du tracé d'orientation désactivé	*PGAs*/ Lissage du tracé d'orientation (ORISON, ORISOF) (Page 357)	m	•	•	•	•
ORISON	Lissage du tracé d'orientation activé	*PGAs*/ Lissage du tracé d'orientation (ORISON, ORISOF) (Page 357)	m	•	•	•	•
ORIVECT	Interpolation circulaire de grand rayon (identique avec ORIPLANE)	*PGAs*/ Programmation des axes d'orientation (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) (Page 333)	m	•	•	•	•
ORIVIRT1	Angles d'orientation définis par axes virtuels d'orientation (définition 1)	*PGAs*/ Programmation des axes d'orientation (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) (Page 333)	m	•	•	•	•
ORIVIRT2	Angles d'orientation définis par axes virtuels d'orientation (définition 1)	*PGAs*/ Programmation des axes d'orientation (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) (Page 333)	m	•	•	•	•

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
ORIWKS [4]	Orientation d'outil dans le système de coordonnées pièce	*PGAs*/ Référence des axes d'orientation (ORIWKS, ORIMKS) (Page 331)	m	●	●	●	●
OS	Activation / désactivation de l'oscillation	*PGAs*/ Oscillation asynchrone (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) (Page 641)		-	-	-	-
OSB	Oscillation : Point de départ	*FB2(P5)*	m	-	-	-	-
OSC	Lissage constant de l'orientation de l'outil	*PGAs*/ Orientation de l'outil (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) (Page 423)	m	●	●	●	●
OSCILL	Axes : 1 à 3 axes de pénétration	*PGAs*/ Oscillation commandée par des actions synchrones (OSCILL) (Page 647)	m	-	-	-	-
OSCTRL	Options d'oscillation	*PGAs*/ Oscillation asynchrone (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) (Page 641)	m	-	-	-	-
OSD	Arrondissement de l'orientation d'outils via la spécification de la longueur d'arrondissement avec SD	*PGAs*/ Orientation de l'outil (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) (Page 423)	m	●	●	●	●
OSE	Point d'arrêt de l'oscillation	*PGAs*/ Oscillation asynchrone (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) (Page 641)	m	-	-	-	-
OSNSC	Oscillation : nombre d'extractions par étincelage	*PGAs*/ Oscillation asynchrone (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) (Page 641)	m	-	-	-	-
OSOF [4]	Désactivation du lissage de l'orientation de l'outil	*PGAs*/ Oscillation asynchrone (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) (Page 641)	m	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
OSP1	Oscillation : point d'inversion gauche	*PGAs*/ Oscillation asynchrone (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) (Page 641)	m	-	-	-	-
OSP2	point d'inversion droit	*PGAs*/ Oscillation asynchrone (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) (Page 641)	m	-	-	-	-
OSS	Lissage de l'orientation de l'outil en fin de bloc	*PGAs*/ Orientation de l'outil (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) (Page 423)	m	●	●	●	●
OSSE	Lissage de l'orientation de l'outil en début et en fin de bloc	*PGAs*/ Orientation de l'outil (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) (Page 423)	m	●	●	●	●
OST	Arrondissement de l'orientation d'outils via la spécification de la tolérance angulaire en degré avec SD (écart maximal de l'allure de variation de l'orientation programmée)	*PGAs*/ Orientation de l'outil (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) (Page 423)	m	●	●	●	●
OST1	Oscillation : point d'arrêt au point d'inversion gauche	*PGAs*/ Oscillation asynchrone (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) (Page 641)	m	-	-	-	-
OST2	Oscillation : point d'arrêt au point d'inversion droit	*PGAs*/ Oscillation asynchrone (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) (Page 641)	m	-	-	-	-
OTOL	Tolérance d'orientation pour fonctions de compression, lissage de l'orientation et modes d'arrondissement	*PGAs*/ Tolérance de contour/orientation programmable (CTOL, OTOL, ATOL) (Page 491)		-	●	-	●
OVR	Correction de vitesse de rotation	*PGAs*/	m	●	●	●	●
OVRA	Correction de vitesse de rotation axiale	*PGAs*/	m	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
OVRRAP	Correction de la vitesse rapide	*PGAs*/	m	●	●	●	●
P	nombre d'exécutions du sous-programme	*PGAs*/ Nombre de répétitions de programme (P) (Page 183)		●	●	●	●
PAROT	Orientation du système de coordonnées pièce sur la pièce	*PGs*/	m	●	●	●	●
PAROTOF	Désactiver la rotation de frame par rapport à l'outil	*PGs*/	m	●	●	●	●
PCALL	Appel d'un sous-programme avec indication de chemin absolu et transfert de paramètres.	*PGAs*/ Appel de sous-programme avec indication de chemin et paramètres (PCALL) (Page 191)		●	●	●	●
PDELAYOF	Poinçonnage avec temporisation désactivé	*PGAs*/ Activation ou désactivation du poinçonnage et du grignotage (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) (Page 655)	m	-	-	-	-
PDELAYON [4]	Poinçonnage avec temporisation activé	*PGAs*/ Activation ou désactivation du poinçonnage et du grignotage (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) (Page 655)	m	-	-	-	-
PHU	Unité physique d'une variable	*PGAs*/ Définition de variables utilisateur (DEF) (Page 22)		●	●	●	●
PL	1. spline B : Distance entre les pôles 2. interpolation polynômiale : longueur de l'intervalle des paramètres pour l'interpolation polynômiale	*PGAs*/ 1. Interpolation de type spline (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Page 233) 2. Interpolation polynomiale (POLY, POLYPATH, PO, PL) (Page 250)	b	- -	○ -	- -	○ -
PM	par minute	*PGs*/		●	●	●	●
PO	Coefficient polynômial pour l'interpolation polynômiale	*PGAs*/ Interpolation polynomiale (POLY, POLYPATH, PO, PL) (Page 250)	b	-	-	-	-

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
POCKET3	Cycle de fraisage, axe carré (fraise quelconque)	*/BHDsl//BHFsl*		●	●	●	●
POCKET4	Cycle de fraisage, axe circulaire (fraise quelconque)	*/BHDsl//BHFsl*		●	●	●	●
POLF	Position de retrait LIFTFAST	*/PGsl//PGAsl*	m	●	●	●	●
POLFA	Lancement de la position de retrait des axes individuels avec $AA_ESR_TRIGGER	*/PGsl*	m	●	●	●	●
POLFMASK	Libération des axes pour le retrait sans corrélation entre les axes	*/PGsl*	m	●	●	●	●
POLFMLIN	Libération des axes pour le retrait avec corrélation linéaire entre les axes	*/PGsl*	m	●	●	●	●
POLY	interpolation polynomiale	*/PGAsl* Interpolation polynomiale (POLY, POLYPATH, PO, PL) (Page 250)	m	-	-	-	-
POLYPATH	Interpolation polynomiale sélectionnable pour les deux groupes d'axes AXIS ou VECT	*/PGAsl* Interpolation polynomiale (POLY, POLYPATH, PO, PL) (Page 250)	m	-	-	-	-
PON	Poinçonnage ACTIVE	*/PGAsl* Activation ou désactivation du poinçonnage et du grignotage (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) (Page 655)	m	-	-	-	-
PONS	Poinçonnage activé à la période d'appel de l'interpolateur	*/PGAsl* Activation ou désactivation du poinçonnage et du grignotage (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) (Page 655)	m	-	-	-	-
POS	Positionnement d'un axe	*/PGsl*		●	●	●	●
POSA	Positionnement d'un axe sur plusieurs blocs	*/PGsl*		●	●	●	●
POSM	Positionnement d'un magasin	*/FBW*		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
POSP	Positionnement en trajets partiels (oscillation)	*PGs/*		●	●	●	●
POSRANGE	Déterminer si la position de consigne courante interpolée d'un axe se trouve dans une fenêtre autour d'une position de référence prescrite	*PGAs/* Position dans la zone de référence prescrite (POSRANGE) (Page 601)		●	●	●	●
POT	Puissance 2 (fonction mathématique)	*PGAs/* Fonctions de calcul (Page 61)		●	●	●	●
PR	Par rotation	*PGs/*		●	●	●	●
PREPRO	Identifier les sous-programmes avec prétraitement	*PGAs/* Identifier les sous-programmes avec prétraitement (PREPRO) (Page 168)		●	●	●	●
PRESETON	Forçage de valeurs réelles pour les axes programmés.	*PGAs/* Décalage Preset (PRESETON) (Page 296)		●	●	●	●
PRIO	Mot-clé pour la définition de priorités lors du traitement d'interruptions	*PGAs/* Affectation et démarrage d'une routine d'interruption (SETINT, PRIO, BLSYNC) (Page 111)		●	●	●	●
PROC	Première instruction d'un programme	*PGAs/* Appel de sous-programme avec indication de chemin et paramètres (PCALL) (Page 191)		●	●	●	●
PTP	Déplacement point à point	*PGAs/* Déplacement PTP cartésien (Page 376)	m	●	●	●	●
PTPG0	Déplacement point à point uniquement pour G0, sinon CP	*PGAs/* PTP avec TRANSMIT (Page 381)	m	●	●	●	●
PUNCHACC	Accélération asservie au grignotage	*PGAs/* Activation ou désactivation du poinçonnage et du grignotage (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) (Page 655)		-	-	-	-
PUTFTOC	Correction d'outil fine pour dressage simultané	*PGAs/* Correction d'outil en ligne (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) (Page 404)		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
PUTFTOCF	Correction fine d'outil selon fonction définie avec FCTDEF pour dressage simultané	*PGAsl* Correction d'outil en ligne (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) (Page 404)		●	●	●	●
PW	Spline B (poids ponctuel)	*PGAsl* Interpolation de type spline (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Page 233)	b	-	○	-	○
QECLRNOF	Apprentissage de la compensation des défauts aux transitions entre quadrants désactivé	*PGAsl* Apprentissage de compensations (QECLRNON, QECLRNOF) (Page 689)		●	●	●	●
QECLRNON	Apprentissage de la compensation des défauts aux transitions entre quadrants activé	*PGAsl* Apprentissage de compensations (QECLRNON, QECLRNOF) (Page 689)		●	●	●	●
QU	Sortie rapide de fonction supplémentaire (auxiliaire)	*PGsl*		●	●	●	●
R...	Paramètre de calcul également comme descripteur d'adresse paramétrable et avec extension numérique	*PGAsl* Variables utilisateur prédéfinies : Paramètre de calcul (R) (Page 18)		●	●	●	●
RAC	Programmation au rayon spécifique à l'axe non modale absolue	*PGsl*	b	●	●	●	●
RDISABLE	blocage de l'introduction via l'interface.	*PGAsl* Activation du blocage de la lecture (RDISABLE) (Page 580)		●	●	●	●
READ	Permet de lire une ou plusieurs lignes dans le fichier indiqué, les informations lues étant rangées dans un tableau	*PGAsl* Lecture des lignes d'un fichier (READ) (Page 134)		●	●	●	●
REAL	Type de données : Variable à virgule flottante avec signe (nombres réels)	*PGAsl* Définition de variables utilisateur (DEF) (Page 22)		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
REDEF	Réglage pour paramètres machine, éléments de langage CN et variables système : groupes d'utilisateurs pour lesquels ils sont affichés	*PGAs*/ Redéfinition de variables système, variables utilisateur et instruction de langage CN (REDEF) (Page 28)		●	●	●	●
RELEASE	Libérer axes machine pour l'échange	*PGAs*/ Permutation d'axe, permutation de broche (RELEASE, GET, GETD) (Page 121)		●	●	●	●
REP	Mot-clé pour initialisation de tous les éléments d'un tableau avec la même valeur	*PGAs*/ Définition et initialisation de variables de tableau (DEF, SET, REP) (Page 44)		●	●	●	●
REPEAT	Itération d'une boucle	*PGAs*/ Répétition de sections de programme (REPEAT, REPEATB, ENDLABEL, P) (Page 88)		●	●	●	●
REPEATB	Itération d'une ligne de programme	*PGAs*/ Répétition de sections de programme (REPEAT, REPEATB, ENDLABEL, P) (Page 88)		●	●	●	●
REPOSA	réaccostage linéaire du contour avec tous les axes	*PGAs*/ Réaccostage du contour (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Page 476)	b	●	●	●	●
REPOSH	Réaccostage du contour avec demi-cercle	*PGAs*/ Réaccostage du contour (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Page 476)	b	●	●	●	●
REPOSHA	Réaccostage du contour avec tous les axes ; axes géométriques en demi-cercle	*PGAs*/ Réaccostage du contour (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Page 476)	b	●	●	●	●
REPOSL	Réaccostage linéaire du contour	*PGAs*/ Réaccostage du contour (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Page 476)	b	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
REPOSQ	Réaccostage du contour en quart de cercle	*PGAs/* Réaccostage du contour (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Page 476)	b	•	•	•	•
REPOSQA	Réaccostage linéaire du contour avec tous les axes ; axes géométriques en quart de cercle	*PGAs/* Réaccostage du contour (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Page 476)	b	•	•	•	•
RESET	Remettre le cycle technologique à zéro	*PGAs/* Bloquer, débloquer, remettre à zéro (LOCK, UNLOCK, RESET) (Page 634)		•	•	•	•
RESETMON	Instruction de langage pour activation de la consigne	*FBW*		•	•	•	•
RET	Fin de sous-programme	*PGAs/* Retour paramétrable dans les sous-programmes (RET ...) (Page 171)		•	•	•	•
RIC	Programmation au rayon spécifique à l'axe non modale relative	*PGs/* Permutation d'axe, permutation de broche (RELEASE, GET, GETD) (Page 121)	b	•	•	•	•
RINDEX	Déterminer l'indice d'un caractère dans une chaîne de caractères	*PGAs/* Recherche de caractères/chaînes de caractères dans une chaîne (INDEX, RINDEX, MINDEX, MATCH) (Page 78)		•	•	•	•
RMB	réaccostage du point de début de bloc	*PGAs/* Réaccostage du contour (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Page 476)	m	•	•	•	•
RME	réaccostage du point final du bloc	*PGAs/* Réaccostage du contour (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Page 476)	m	•	•	•	•

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
RMI [4]	réaccostage du point d'interruption	*PGAs/* Réaccostage du contour (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Page 476)	m	●	●	●	●
RMN	réaccostage du point de contour le plus proche	*PGAs/* Réaccostage du contour (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Page 476)	m	●	●	●	●
RND	Arrondir un angle	*PGs/*	b	●	●	●	●
RNDM	Arrondissement modal	*PGs/*	m	●	●	●	●
ROT	Rotation programmable	*PGs/*	b	●	●	●	●
ROTS	Programmation de rotations de frames avec des angles solides	*PGs/*	b	●	●	●	●
ROUND	Arrondissement des décimales	*PGAs/* Fonctions de calcul (Page 61)		●	●	●	●
ROUNDUP	Arrondissement d'une valeur d'entrée	*PGAs/* Arrondissement (ROUNDUP) (Page 144)		●	●	●	●
RP	Rayon polaire	*PGs/*	m/b	●	●	●	●
RPL	Rotation dans le plan	*PGs/*	b	●	●	●	●
RT	Paramètre d'accès aux données frame : Rotation	*PGAs/* Lecture et modification de composantes de frames (TR, FI, RT, SC, MI) (Page 289)		●	●	●	●
RTLIOF	G0 sans interpolation linéaire (interpolation axiale individuelle)	*PGs/*	m	●	●	●	●
RTLION	G0 avec interpolation linéaire	*PGs/*	m	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
S	Vitesse de rotation de broche (pour G4, G96/G961, autre signification)	*PGs/*	m/b	●	●	●	●
SAVE	Attribut pour sauvegarde d'informations en cas d'appel de sous-programme	*PGAs/* Sauvegarde des fonctions G modales (SAVE) (Page 157)		●	●	●	●
SBLOF	Inhibition du bloc par bloc	*PGAs/* Inhibition de l'exécution bloc par bloc (SBLOF, SBLON) (Page 158)		●	●	●	●
SBLON	Annulation de l'inhibition du bloc par bloc	*PGAs/* Inhibition de l'exécution bloc par bloc (SBLOF, SBLON) (Page 158)		●	●	●	●
SC	Paramètre d'accès aux données frame : Mise à l'échelle	*PGAs/* Lecture et modification de composantes de frames (TR, FI, RT, SC, MI) (Page 289)		●	●	●	●
SCALE	Mise à l'échelle programmable	*PGs/*	b	●	●	●	●
SCC	Affectation sélective d'un axe transversal pour G96/G961/G962. Les descripteurs d'axes peuvent être des axes géométriques, des axes de canal ou des axes machine.	*PGs/*		●	●	●	●
SCPARA	Programmation du jeu de paramètres servo	*PGAs/* Jeu de paramètres servo programmable (SCPARA) (Page 278)		●	●	●	●
SD	Degré de spline :	*PGAs/* Interpolation de type spline (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Page 233)	b	-	○	-	○
SEFORM	Instruction structurante dans l'éditeur Step pour générer la vue de l'opération dans le logiciel HMI Advanced.	*PGAs/* Instruction structurelle dans l'éditeur Step (SEFORM) (Page 213)		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
SET	Mot-clé pour initialisation de tous les éléments d'un tableau avec les valeurs mentionnées	*PGAsl* Définition et initialisation de variables de tableau (DEF, SET, REP) (Page 44)		●	●	●	●
SETAL	Activation d'une alarme	*PGAsl* Alarmes (SETAL) (Page 699)		●	●	●	●
SETDNO	Affectation du numéro D du tranchant (CE) d'un outil (T)	*PGAsl* Libre affectation des numéros D : modification des numéros D (GETDNO, SETDNO) (Page 431)		●	●	●	●
SETINT	Définir la routine d'interruption qui doit être activée lorsqu'une entrée NCK bascule	*PGAsl* Affectation et démarrage d'une routine d'interruption (SETINT, PRIO, BLSYNC) (Page 111)		●	●	●	●
SETM	Définition de repères dans le canal spécifique	PGAsl Coordination de programmes (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) (Page 103)		-	-	-	-
SETMS	Retour à la broche maître définie dans le paramètre machine			●	●	●	●
SETMS (n)	La broche n doit devenir broche maître	*PGsl*		●	●	●	●
SETMTH	Définir le numéro du porte-outil maître	*FBW*		●	●	●	●
SETPIECE	Tenir compte du nombre de pièces pour tous les outils affectés à la broche	*FBW*		●	●	●	●
SETTA	Activer un outil du groupe d'usure	*FBW*		●	●	●	●
SETTCOR	Modification de composantes d'outils avec prise en compte de l'ensemble des conditions marginales	*FB1(W1)*		●	●	●	●
SETTIA	Désactiver un outil du groupe d'usure	*FBW*		●	●	●	●
SF	Décalage du point de départ pour filetage	*PGsl*	m	●	●	●	●
SIN	Sinus (fonction trigonométrique)	*PGAsl* Fonctions de calcul (Page 61)		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
SIRELAY	Activer les fonctions de sécurité paramétrées avec SIRELIN, SIRELOUT et SIRELTIME	*FBSIs/*		-	-	-	-
SIRELIN	Initialiser les grandeurs d'entrée du bloc fonctionnel	*FBSIs/*		-	-	-	-
SIRELOUT	Initialiser les grandeurs de sortie du bloc fonctionnel	*FBSIs/*		-	-	-	-
SIRELTIME	Initialiser les temporisations du bloc fonctionnel	*FBSIs/*		-	-	-	-
SLOT1	Cycle de fraisage, rainures radiales	*BHDsl/BHFsl*		●	●	●	●
SLOT2	Cycle de fraisage, rainure radiale	*BHDsl/BHFsl*		●	●	●	●
SOFT	Accélération sur trajectoire avec limitation des à-coups	*PGsl*	m	●	●	●	●
SOFTA	Activer l'accélération avec limitation des à-coups pour les axes programmés	*PGsl*		●	●	●	●
SON	Grignotage ACTIVE	*PGAsl* Activation ou désactivation du poinçonnage et du grignotage (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) (Page 655)	m	-	-	-	-
SONS	Grignotage activé à la période d'appel de l'interpolateur	*PGAsl* Activation ou désactivation du poinçonnage et du grignotage (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) (Page 655)	m	-	-	-	-
SPATH [4]	La référence trajectoire pour axes FGROUP est la longueur d'arc	*PGAsl* Référence réglable de trajectoire (SPATH, UPATH) (Page 256)	m	●	●	●	●
SPCOF	Commutation la broche maître ou broche avec numéro n de l'asservissement de position sur la régulation de la vitesse de rotation	*PGsl*	m	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
SPCON	Commutation la broche maître ou broche avec numéro n de la régulation de la vitesse de rotation sur l'asservissement de position	*PGAs/*	m	●	●	●	●
SPI	Conversion du numéro de broche en descripteur d'axe	*PGAs/* Fonctions d'axe (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) (Page 669)		●	●	●	●
SPIF1 [4]	E/S NCK rapides pour poinçonnage/grignotage, octet 1	*FB2(N4)*	m	-	-	-	-
SPIF2	E/S NCK rapides pour poinçonnage/grignotage, octet 2	*FB2(N4)*	m	-	-	-	-
SPLINEPATH	Détermination du groupe Spline	*PGAs/* Groupe spline (SPLINEPATH) (Page 245)		-	○	-	○
SPN	Nombre de distances partielles par bloc	*PGAs/* Préparation automatique du déplacement (Page 660)	b	-	-	-	-
SPOF [4]	Coup désactivé, poinçonnage, grignotage désactivés	*PGAs/* Activation ou désactivation du poinçonnage et du grignotage (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) (Page 655)	m	-	-	-	-
SPOS	Position de broche	*PGs/*	m	●	●	●	●
SPOSA	Position de broche { au-delà des limites de bloc	*PGs/*	m	●	●	●	●
SPP	Longueur d'une distance partielle	*PGAs/* Préparation automatique du déplacement (Page 660)	m	-	-	-	-
SQRT	Racine carrée (fonction mathématique) (square root)	*PGAs/* Fonctions de calcul (Page 61)		●	●	●	●
SR	Course de retrait de l'oscillation pour une action synchrone	*PGs/*	b	-	-	-	-

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
SRA	Course axiale de retrait de l'oscillation déclenché par entrée externe pour une action synchrone	*PGs/*	m	-	-	-	-
ST	Temps d'arrêt de l'étincelage de l'oscillation pour une action synchrone	*PGs/*	b	-	-	-	-
STA	Temps d'arrêt de l'étincelage de l'oscillation axiale pour une action synchrone	*PGs/*	m	-	-	-	-
START	Lancement des programmes sélectionnés simultanément dans plusieurs canaux, à partir du programme en cours	*PGAs/* Coordination de programmes (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) (Page 103)		-	-	-	-
STARTFIFO [4]	Exécution ; remplissage en parallèle du tampon d'exécution	*PGAs/* Exécution du programme avec une mémoire tampon (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE) (Page 465)	m	●	●	●	●
STAT	Position des articulations	*PGAs/* Déplacement PTP cartésien (Page 376)	b	●	●	●	●
STOLF	Facteur de tolérance G0	*PGAs/* Tolérance pour déplacements G0 (STOLF) (Page 495)	m	-	-	-	-
STOPFIFO	Arrêt de l'exécution remplissage du tampon d'exécution jusqu'à la reconnaissance de STARTFIFO, la saturation du tampon d'exécution ou la fin du programme	*PGAs/* Exécution du programme avec une mémoire tampon (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE) (Page 465)	m	●	●	●	●
STOPRE	Arrêt du prétraitement des blocs jusqu'à ce que tous les blocs préparés aient été exécutés	*PGAs/* Exécution du programme avec une mémoire tampon (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE) (Page 465)		●	●	●	●
STOPREOF	Annulation de l'arrêt du prétraitement des blocs	*PGAs/* Annulation de l'arrêt du prétraitement des blocs (STOPREOF) (Page 581)		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
STRING	Type de données : Chaîne de caractères	*PGAs*/ Définition de variables utilisateur (DEF) (Page 22)		●	●	●	●
STRINGFELD	Sélection d'un caractère individuel dans le tableau des sauts programmé	*PGAs*/ Sélection d'un caractère individuel (STRINGVAR, STRINGFELD) (Page 80)		●	●	●	●
STRINGIS	Contrôle le langage CN existant et si les noms de cycles CN , les variables utilisateurs, les macros et les noms d'étiquettes appartenant spécifiquement à cette commande existent, sont valables, définis ou encore actifs.	*PGAs*/ Contrôler le langage CN existant (STRINGIS) (Page 683)		●	●	●	●
STRINGVAR	Sélection d'un caractère individuel dans la chaîne programmée	*PGAs*/ Sélection d'un caractère individuel (STRINGVAR, STRINGFELD) (Page 80)		-	-	-	-
STRLEN	Déterminer la longueur d'une chaîne de caractères	*PGAs*/ Déterminer la longueur d'une chaîne de caractères (STRLEN) (Page 78)		●	●	●	●
SUBSTR	Déterminer l'indice d'un caractère dans une chaîne de caractères	*PGAs*/ Sélection d'une chaîne de caractères partielle (SUBSTR) (Page 80)		●	●	●	●
SUPA	Inhibition du décalage d'origine actuel, y compris les décalages programmés, les frames système, les décalages par manivelle (DRF), le décalage externe d'origine et le déplacement forcé	*PGs*/	b	●	●	●	●
SVC	Vitesse de coupe d'outil	*PGs*/	m	●	●	●	●
SYNFCT	Evaluation d'un polynôme en fonction d'une condition dans une action synchrone au déplacement	*PGAs*/ Fonction synchrone (SYNFCT) (Page 586)		●	●	●	●
SYNR	Lecture synchrone de la variable, c'est-à-dire au moment de l'exécution	*PGAs*/ Définition de variables utilisateur (DEF) (Page 22)		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
SYNRW	Lecture et écriture synchrones de la variable, c'est-à-dire au moment de l'exécution	*PGAs* Définition de variables utilisateur (DEF) (Page 22)		●	●	●	●
SYNW	Ecriture synchrone de la variable, c'est-à-dire au moment de l'exécution	*PGAs* Définition de variables utilisateur (DEF) (Page 22)		●	●	●	●
T	Appel d'outil (changement uniquement s'il est fixé dans les paramètres machine ; sinon instruction M6 nécessaire)	*PGs*		●	●	●	●
TAN	Tangente (fonction trigonométrique)	*PGAs* Fonctions de calcul (Page 61)		●	●	●	●
TANG	Définition du groupe d'axes à asservissement tangentiel	*PGAs* Positionnement tangentiel (TANG, TANGON, TANGOF, TLIFT, TANGDEL) (Page 453)		-	-	-	-
TANGDEL	Suppression de la définition du groupe d'axes à asservissement tangentiel	*PGAs* Positionnement tangentiel (TANG, TANGON, TANGOF, TLIFT, TANGDEL) (Page 453)		-	-	-	-
TANGOF	Asservissement tangentiel désactivé	*PGAs* Positionnement tangentiel (TANG, TANGON, TANGOF, TLIFT, TANGDEL) (Page 453)		-	-	-	-
TANGON	Asservissement tangentiel activé	*PGAs* Positionnement tangentiel (TANG, TANGON, TANGOF, TLIFT, TANGDEL) (Page 453)		-	-	-	-
TCA	Sélection /changement d'outil indépendant de l'état de l'outil	*FBW*		●	●	●	●
TCARR	Appeler un organe porte-outil (numéro "m")	*PGAs* Correction de longueur d'outil pour organes porte-outils orientables (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) (Page 439)		-	●	-	●
TCI	Mettre en place l'outil de la mémoire temporaire dans le magasion	*FBW*		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
TCOABS[4]	Détermination des composantes de longueur d'outil à partir de l'orientation d'outil courante	*PGAs*/ Correction de longueur d'outil pour organes porte-outils orientables (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) (Page 439)	m	-	●	-	●
TCOFR	Détermination des composantes de longueur d'outil à partir de l'orientation du frame actif.	*PGAs*/ Correction de longueur d'outil pour organes porte-outils orientables (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) (Page 439)	m	-	●	-	●
avec TCOFRX	Lors de la sélection d'outil, détermination d'orientation d'outil d'un frame activé, outil pointé en direction X	*PGAs*/ Correction de longueur d'outil pour organes porte-outils orientables (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) (Page 439)	m	-	●	-	●
avec TCOFRY	Lors de la sélection d'outil, détermination d'orientation d'outil d'un frame activé, outil pointé en direction Y	*PGAs*/ Correction de longueur d'outil pour organes porte-outils orientables (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) (Page 439)	m	-	●	-	●
TCOFRZ	Lors de la sélection d'outil, détermination d'orientation d'outil d'un frame activé, outil pointé en direction Z	*PGAs*/ Correction de longueur d'outil pour organes porte-outils orientables (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) (Page 439)	m	-	●	-	●
THETA	angle de rotation	*PGAs*/ Rotations de l'orientation de l'outil (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) (Page 343)	b	●	●	●	●
TILT	Angle latéral	*PGAs*/ Programmation de l'orientation de l'outil (A..., B..., C..., LEAD, TILT) (Page 323)	m	●	●	●	●
TLIFT	Insertion d'un bloc intermédiaire aux angles du contour dans le cas d'une commande tangentielle	*PGAs*/ Positionnement tangentiel (TANG, TANGON, TANGOF, TLIFT, TANGDEL) (Page 453)		-	-	-	-

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
TMOF	Désactivation de la surveillance d'outil	*PGAs/* Surveillance d'outil spécifique à la rectification dans le programme pièce (TMON, TMOF) (Page 667)		•	•	•	•
TMON	Activation de la surveillance d'outil	*PGAs/* Surveillance d'outil spécifique à la rectification dans le programme pièce (TMON, TMOF) (Page 667)		•	•	•	•
TO	Désigne la valeur finale d'une boucle de comptage FOR	*PGAs/* Boucle de comptage (FOR ... TO ..., ENDFOR) (Page 99)		•	•	•	•
TOFF	Offset de longueur d'outil dans la direction de la composante de longueur d'outil agissant parallèlement à l'axe géométrique indiqué dans l'indice	*PGs/*	m	•	•	•	•
TOFFL	Offset de longueur d'outil dans la direction de la composante de longueur d'outil L1, L2 ou L3	*PGs/*	m	•	•	•	•
TOFFOF	Initialisation de la correction de la longueur d'outil en ligne	*PGAs/* Correction de longueur d'outil en ligne (TOFFON, TOFFOF) (Page 443)		•	•	•	•
TOFFON	Activation de la correction en ligne de la longueur d'outil	*PGAs/* Correction de longueur d'outil en ligne (TOFFON, TOFFOF) (Page 443)		•	•	•	•
TOFFR	Offset de rayon d'outil	*PGs/*	m	•	•	•	•
TOFRAME	Orienter l'axe Z du SCP parallèlement à l'orientation de la pièce par rotation du frame	*PGs/*	m	•	•	•	•
TOFRAMEX	Orienter l'axe X du SCP parallèlement à l'orientation de la pièce par rotation du frame	*PGs/*	m	•	•	•	•
TOFRAMEY	Orienter l'axe Y du SCP parallèlement à l'orientation de la pièce par rotation du frame	*PGs/*	m	•	•	•	•

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
TOFRAMEZ	Identique à TOFRAME	*PGs/*	m	●	●	●	●
TOLOWER	Transformer les caractères d'une chaîne de caractères en minuscules	*PGAs/* Conversion en minuscules/majuscules (TOLOWER, TOUPPER) (Page 77)		●	●	●	●
TOOLENV	Enregistrer tous les états actuels significatifs pour l'exploitation des données d'outil mémorisées dans la mémoire	*FB1(W1)*		●	●	●	●
TOROT	Orienter l'axe Z du SCP parallèlement à l'orientation de la pièce par rotation du frame	*PGs/*	m	●	●	●	●
TOROTOF	Rotations de frames en direction de l'outil désactivées	*PGs/*	m	●	●	●	●
TOROTX	Orienter l'axe X du SCP parallèlement à l'orientation de la pièce par rotation du frame	*PGs/*	m	●	●	●	●
TOROTY	Orienter l'axe Y du SCP parallèlement à l'orientation de la pièce par rotation du frame	*PGs/*	m	●	●	●	●
TOROTZ	Identique à TOROT	*PGs/*	m	●	●	●	●
TOUPPER	Transformer les caractères d'une chaîne de caractères en majuscules	*PGAs/* Conversion en minuscules/majuscules (TOLOWER, TOUPPER) (Page 77)		●	●	●	●
TOWBCS	Valeurs d'usure dans le système de coordonnées de base (SCB)	*PGAs/* Système de coordonnées de l'usinage actif (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) (Page 400)	m	-	●	-	●
TOWKCS	Valeurs d'usure dans le système de coordonnées de la tête d'outil avec transformation cinétique (diffère du SCM avec rotation d'outil)	*PGAs/* Système de coordonnées de l'usinage actif (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) (Page 400)	m	-	●	-	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
TOWMCS	Valeurs d'usure dans le système de coordonnées machine (SCM)	*PGAs*/ Système de coordonnées de l'usinage actif (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) (Page 400)	m	-	●	-	●
TOWSTD	Valeur de position de base pour corrections sur la longueur d'outil	*PGAs*/ Système de coordonnées de l'usinage actif (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) (Page 400)	m	-	●	-	●
TOWTCS	Valeurs d'usure dans le système de coordonnées outil (point de référence du porte-outil T sur le mandrin du porte-outil)	*PGAs*/ Système de coordonnées de l'usinage actif (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) (Page 400)	m	-	●	-	●
TOWWCS	Valeurs d'usure dans le système de coordonnées pièce (SCP)	*PGAs*/ Système de coordonnées de l'usinage actif (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) (Page 400)	m	-	●	-	●
TR	Composante de décalage d'une variable de frame	*PGAs*/ Lecture et modification de composantes de frames (TR, FI, RT, SC, MI) (Page 289)		●	●	●	●
TRAANG	Transformation axe oblique	*PGAs*/ Axe oblique (TRAANG) (Page 371)		-	-	○	-
TRACON	Transformation en cascade	*PGAs*/ Transformations concaténées (TRACON, TRAFOOF) (Page 387)		-	-	○	-
TRACYL	Cylindre : transformation des surfaces latérales	*PGAs*/ Transformation de surface latérale de cylindre (TRACYL) (Page 362)		○	○	○	○
TRAFOOF	Désactiver les transformations actives dans le canal	*PGAs*/ Transformations concaténées (TRACON, TRAFOOF) (Page 387)		●	●	●	●
TRAILOF	Déplacement conjugué synchrone à l'axe désactivé	*PGAs*/ Déplacements conjugués (TRAILON, TRAILOF) (Page 497)		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
TRAILON	Déplacement conjugué synchrone à l'axe activé	*PGAs*/ Déplacements conjugués (TRAILON, TRAILOF) (Page 497)		●	●	●	●
TRANS	Décalage programmable	*PGs*/	b	●	●	●	●
TRANSMIT	Transformation polaire (traitement de surface frontale)	*PGAs*/ Opérations de fraisage sur des pièces de tournage (TRANSMIT) (Page 359)		○	○	○	○
TRAORI	Transformation 4, 5 axes, transformation générique	*PGAs*/ Transformation trois, quatre et cinq axes (TRAORI) (Page 320)		-	●	-	●
TRUE	Constante logique : vrai	*PGAs*/ Définition de variables utilisateur (DEF) (Page 22)		●	●	●	●
TRUNC	troncature des décimales	*PGAs*/ Correction de la précision pour les erreurs relationnelles (TRUNC) (Page 66)		●	●	●	●
TU	Angle d'axe	*PGAs*/ Déplacement PTP cartésien (Page 376)	b	●	●	●	●
TURN	Nombre de tours pour une hélice	*PGs*/	b	●	●	●	●
ULI	Valeur limite supérieure de variables	*PGAs*/ Attribut : valeurs limites (LLI, ULI) (Page 34)		●	●	●	●
UNLOCK	Libérer une action synchrone avec ID (poursuivre un cycle technologique)	*PGAs*/ Bloquer, débloquer, remettre à zéro (LOCK, UNLOCK, RESET) (Page 634)		●	●	●	●
UNTIL	Condition de sortie d'une boucle REPEAT	*PGAs*/ Boucle de programme avec condition en début de boucle (WHILE, ENDWHILE) (Page 100)		●	●	●	●
UPATH	La référence trajectoire pour axes FGROUP est paramètre de courbe	*PGAs*/ Référence réglable de trajectoire (SPATH, UPATH) (Page 256)	m	●	●	●	●
VAR	Mot-clé : type du transfert de paramètres	*PGAs*/ Appel d'un sous-programme avec transfert de paramètres (EXTERN) (Page 180)		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
VELOLIM	Réduction de la vitesse axiale maximale	*PGAs/* Correction de la vitesse en pourcentage (VELOLIM) (Page 487)	m	●	●	●	●
VELOLIMA	Réduction ou dépassement de la vitesse axiale maximale de l'axe asservi	*PGs/*	m	●	●	●	●
WAITC	Attendre que le critère de changement de bloc de couplage soit rempli pour les axes/broches	*PGAs/* Coordination de programmes (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) (Page 103)		-	-	○	-
WAITE	Attente de la fin du programme dans un autre canal.	*PGAs/* Coordination de programmes (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) (Page 103)		-	-	-	-
WAITENC	Attente des positions d'axe synchronisées ou restaurées	*PGAs/* Attente de la position d'axe définitive (WAITENC) (Page 681)		-	-	-	-
WAITM	Attente de la marque dans canal indiqué ; terminer le bloc précédent par un arrêt précis.	*PGAs/* Coordination de programmes (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) (Page 103)		-	-	-	-
WAITMC	Attente de la marque dans canal indiqué ; arrêt précis uniquement si les autres canaux n'ont pas encore atteint la marque.	*PGAs/* Coordination de programmes (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) (Page 103)		-	-	-	-
WAITP	Attente de la fin du déplacement de l'axe de positionnement	*PGs/*		●	●	●	●
WAITS	Attente jusqu'à ce que la position de la broche soit atteinte	*PGs/*		●	●	●	●
WALCS0	Limitation de la zone de travail SCP désélectionnée	*PGs/*	m	●	●	●	●
WALCS1	Groupe de limitations de la zone de travail SCP 1 actif	*PGs/*	m	●	●	●	●
WALCS2	Groupe de limitations de la zone de travail SCP 2 actif	*PGs/*	m	●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
WALCS3	Groupe de limitations de la zone de travail SCP 3 actif	*PGs/*	m	●	●	●	●
WALCS4	Groupe de limitations de la zone de travail SCP 4 actif	*PGs/*	m	●	●	●	●
WALCS5	Groupe de limitations de la zone de travail SCP 5 actif	*PGs/*	m	●	●	●	●
WALCS6	Groupe de limitations de la zone de travail SCP 6 actif	*PGs/*	m	●	●	●	●
WALCS7	Groupe de limitations de la zone de travail SCP 7 actif	*PGs/*	m	●	●	●	●
WALCS8	Groupe de limitations de la zone de travail SCP 8 actif	*PGs/*	m	●	●	●	●
WALCS9	Groupe de limitations de la zone de travail SCP 9 actif	*PGs/*	m	●	●	●	●
WALCS10	Groupe de limitations de la zone de travail SCP 10 actif	*PGs/*	m	●	●	●	●
WALIMOF	Désactivation de la limitation de la zone de travail SCB	*PGs/*	m	●	●	●	●
WALIMON [4]	Activation de la limitation de la zone de travail SCB	*PGs/*	m	●	●	●	●
WHEN	L'action est exécutée de manière cyclique lorsque la condition est remplie.	*PGAs/* Contrôle cyclique de la condition (WHEN, WHENEVER, FROM, EVERY) (Page 555)		●	●	●	●
WHENEVER	L'action est exécutée de manière unique lorsque la condition est remplie une fois.	*PGAs/* Contrôle cyclique de la condition (WHEN, WHENEVER, FROM, EVERY) (Page 555)		●	●	●	●
WHILE	Début d'une boucle WHILE	*PGAs/* Boucle de programme avec condition en début de boucle (WHILE, ENDWHILE) (Page 100)		●	●	●	●
WRITE	Ecrire un texte dans le système de fichiers. Ajoute un bloc à la fin du fichier indiqué.	*PGAs/* Ecriture d'un fichier (WRITE) (Page 129)		●	●	●	●

Instruction	Signification	Description, voir [1]	E [2]	828D [3]			
				PPU260 / 261		PPU280 / 281	
				D	F	D	F
WRTPR	Temporise la tâche de traitement sans interrompre le contournage	*PGAs/*		•	•	•	•
X	Nom d'axe	*PGs/*	m/b	•	•	•	•
XOR	OU exclusif logique	*PGAs/* Opérations relationnelles et opérations logiques (Page 64)		•	•	•	•
Y	Nom d'axe	*PGs/*	m/b	•	•	•	•
Z	Nom d'axe	*PGs/*	m/b	•	•	•	•

Annexes

A

A.1 Liste des abréviations

A	Sortie
AS	Automate
ASCII	American Standard Code for Information Interchange : Code standard américain pour l'échange d'information
ASIC	Application Specific Integrated Circuit : Circuit utilisateur
ASUP	Sous-programme asynchrone
AV	Notions complémentaires
AWL	Liste des instructions
BA	Mode de fonctionnement
GMFC	Groupe à mode de fonctionnement commun
PRÊT	Prêt à fonctionner
BuB, B&B	Conduite et supervision
DCB	Décimal codé en binaire : chiffres décimaux codés en binaire
MCC	Mini-console de commande
BIN	Fichiers binaires (**Bin**ary Files)
BIOS	Basic Input Output System
SCB	Système de coordonnées de base
IU	Interface utilisateur
BOT	Boot Files : fichiers boot pour SIMODRIVE 611 digital
BT	Tableau de commande
OPI	Interface du tableau de commande
CAO	Conception Assistée par Ordinateur
FAO	Fabrication Assistée par Ordinateur
CNC	Computerized Numerical Control: Commande numérique CNC
COM	Communication
CP	Communication Processor : processeur de communication
CPU	Central Processing Unit : unité centrale de traitement
CR	Carriage Return : retour chariot
CRT	Cathode Ray Tube : tube cathodique
CSB	Central Service Board : carte AP
CTS	Clear To Send : signalisation de l'état Prêt à l'émission (PAE) des interfaces série
CUTCOM	Cutter radius compensation: Correction de rayon de l'outil
CNA	Convertisseur numérique-analogique
DB	Bloc de données dans l'AP
DBB	Octet de bloc de données dans l'AP
DBW	Mot de bloc de données dans l'AP

DBX	Bit de bloc de données dans l'AP
DC	Direct Control : déplacement de positionnement de l'axe rotatif par le plus court chemin sur la position absolue pendant un tour.
DCD	Carrier Detect
DDE	Dynamic Data Exchange
ETTD	Equipement terminal de traitement de données
DIN	Deutsche Industrie Norm (Norme industrielle allemande)
DIO	Data Input/Output : affichage de la transmission de données
DIR	Directory : Répertoire
DLL	Dynamic Link Library : bibliothèque de liens dynamiques
DOE	Matériel de transmission de données
DOS	Disk Operating System
DPM	Dual Port Memory
DPR	Dual Port RAM
DRAM	Dynamic Random Access Memory
DRF	Differential resolver function : Fonction de résolveur différentiel (manivelle électronique)
DRY	Dry Run : Avance de marche d'essai
DSB	Decoding Single Block : décodage bloc par bloc
DW	Mot de données
E	Entrée
E/S	Entrée/sortie
E/R	Module alimentation/réinjection (alimentation) de SIMODRIVE 611digital
Code EIA	Code spécial de bande perforée : perforations par caractère toujours en nombre impair
ENC	Encoder : codeur de valeurs réelles
EPROM	Erasable Programmable Read Only Memory (mémoire morte reprogrammable électroniquement)
ERROR	Error from printer
FB	Bloc fonctionnel
FBS	Ecran plat
FC	Function Call : bloc fonctionnel dans l'AP
FDB	Base de données produits
FDD	Floppy Disk Drive (lecteur de disquette)
FEPROM	Flash-EPROM : mémoire Flash (non volatile)
FIFO	First in First Out : mémoire fonctionnant sans adressage et dont les données sont lues dans l'ordre de leur stockage.
IPOF	Interpolateur fin
FM	Module de fonction
FPU	Floating Point Unit : module à virgule flottante
FRA	Bloc de frame
FRAME	Bloc de données (cadre)
CRF	Correction rayon de fraise
FST	Feed Stop : arrêt avance

FUP	Plan des fonctions (méthode de programmation pour AP)
PB	Programme de base
GUD	Global User Data : Données utilisateur globales
HD	Hard Disc : Disque dur
HEX	Abréviation pour nombre hexadécimal
HiFu	Fonction auxiliaire
HMI	Human Machine Interface : interface utilisateur de la commande numérique pour l'usinage, la programmation et la simulation.
HMS	Système de mesure à résolution élevée
EBR	Entraînement de la broche principale
HW	Matériel
MS	Mise en service
IF	Déblocage impulsion du module d'entraînement
IC (GD)	Communication implicite (données globales)
CIVA	Interpolative Compensation : Compensation avec interpolation
IM	Interface module : module de couplage
IMR	Interface-Module Receive : carte de couplage pour réception
IMS	Interface-Module Send : carte de couplage pour émission
INC	Increment : Déplacement en manuel incrémental
INI	Initializing Data : Données d'initialisation
IPO	Interpolateur
ISA	International Standard Architecture
ISO	International Standard Organization
Code ISO	Code spécial de bande perforée : perforations par caractère toujours en nombre pair
JOG	Jogging : mode "réglage"
K1 .. K4	Canal 1 à canal 4
Bus C	Bus de communication
KD	Rotation du système de coordonnées
CONT	Schéma à contacts (méthode de programmation pour AP)
K_v	Gain de boucle
R_t	Rapport de transmission
LCD	Liquid Crystal Display : affichage à cristaux liquides
DEL	Light Emitting Diode : diode électroluminescente
LF	Line Feed
LMS	Système de mesure de position
RP	Régulateur de position
LUD	Local User Data : données locales utilisateur
Mo	Mégaoctet
PM	Paramètres machine
MDA	Manual Data Automatic : introduction manuelle des données
CM	Circuit de mesure
SCM	Système de coordonnées machine
MLFB	Numéro de référence. Code produit lisible par machine

MPF	Fichier programme principal: programme pièce CN (programme principal)
MPI	Multi Point Interface : interface multipoint
MS	Microsoft (éditeur de logiciels)
TCM	Tableau de commande machine
NC	Numerical Control: unité de commande numérique
NCK	Numerical control kernel : noyau de la commande numérique avec préparation des blocs, interpolation, etc.
NCU	Numerical Control Unit : unité matérielle de NCK
NRK	Désignation du programme système de NCK
SI	Signal d'interface
NURBS	Non Uniform Rational B-Spline (courbes NURBS)
DO	Décalage d'origine
OB	Bloc d'organisation dans l'AP
OEM	Original Equipment Manufacturer
OP	Tableau de commande : terminal opérateur
OPI	Operators Panel Interface : coupleur de tableau de commande
OPT	Options : options
OSI	Open Systems Interconnection : normalisation de la communication entre ordinateurs
Bus P	Bus de périphérie
PC	Personal Computer
PCIN	Nom du logiciel pour l'échange de données avec la commande
PCMCIA	Personal Computer Memory Card International Association : Association de normalisation de cartes mémoire enfichables, standard
PCU	PC-Unit PC-Box (unité de traitement)
PG	Console de programmation
AP	Automate programmable : Automate Programmable
POS	Positionnement
RAM	Random Access Memory : Mémoire de programmes accessible en lecture et en écriture
REF	Accostage de point de référence
REPOS	Fonction repositionnement
RISC	Reduced Instruction Set Computer : type de processeur à jeu d'instructions réduit et exécution rapide des instructions
ROV	Rapid Override : correction du rapide
RPA	R-Parameter Active : zone de mémoire de NCK pour numéros de paramètre R
RPY	Roll Pitch Yaw : type de rotation d'un système de coordonnées
RTS	Request To Send : signalisation de commande d'interfaces de données série, demande d'émission
SBL	Single Block : Bloc par bloc
SD	Données de réglage
SDB	Bloc de données système
SEA	Setting Data Active : identificateur (type de fichier) pour données de réglage
SFB	Bloc fonctionnel système

SFC	System Function Call
TL	Touche logicielle
SKP	Skip : Saut de bloc optionnel
SM	moteur pas à pas
SPF	Sub Program File : Sous-programme
SPS	Automate programmable
SRAM	Mémoire statique (bufferisée)
CRP	Correction de rayon de plaquette
CEPV	Compensation d'erreur de pas de vis de transmission
SSI	Serial Synchron Interface : interface série synchrone
SW	Logiciel
SYF	System Files : fichiers système
TEA	Testing Data Active : identificateur de paramètres machine
TO	Tool Offset : Correction d'outil
TOA	Tool Offset Active : identificateur (type de fichier) pour corrections d'outil
TRANSMIT	Transform Milling into Turning : conversion de coordonnées pour fraisage sur tour
UFR	User Frame : Décalage d'origine
SP	Sous-programme
EAV	Entraînement d'avance
V.24	Interface série (spécification des lignes d'échange de données entre ETTD et ETCD)
SCP	Système de coordonnées pièce
WKZ	Outil
WLK	Correction de longueur d'outil
WOP	Programmation orientée atelier
WPD	Work Piece Directory : Répertoire pièce
WRK	Correction du rayon d'outil
CO	Correction d'outil
WZW	Changement d'outil
ZOA	Zero Offset Active : identificateur (type de fichier) pour données de décalage d'origine
µC	micro-contrôleur

A.2 Remarques sur la documentation

Le présent document est constamment en cours développement en termes de qualité et de convivialité. Veuillez nous aider dans cet effort en nous communiquant vos remarques et propositions d'amélioration par courriel ou télécopie à :

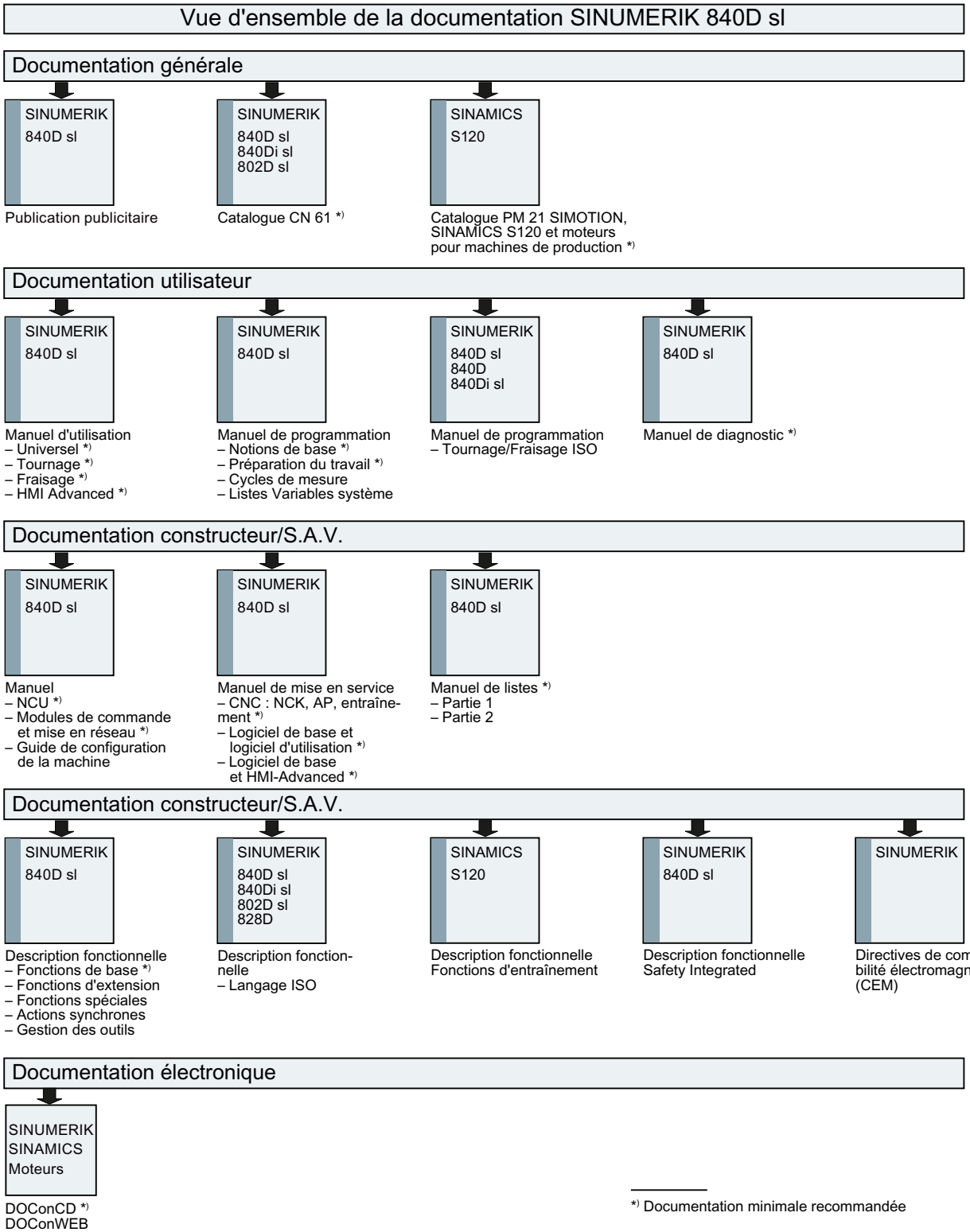
Courriel : mailto:docu.motioncontrol@siemens.com

Télécopie : +49 9131 - 98 2176
Veuillez utiliser le formulaire de télécopie au recto.

Destinataire : SIEMENS AG I DT MC MS1 Postfach 3180 D-91050 Erlangen Télécopie : +49 9131 - 98 2176 (documentation)	Emetteur Nom : Adresse de votre société/service Rue : Code postal : Ville : Téléphone : / Télécopie : /

Propositions et/ou corrections

A.3 Vue d'ensemble de la documentation



Glossaire

Accélération avec limitation des à-coups

Pour obtenir une accélération optimale tout en ménageant les organes mécaniques de la machine, il est possible de basculer, dans le programme d'usinage, entre accélération sous forme d'échelon et accélération continue (sans à-coup).

Accostage d'un point fixe

Les machines-outils peuvent accoster des points fixes qui ont été définis, tels que point de changement d'outil, point de chargement, point de changement de palette, etc. Les coordonnées de ces points sont mémorisées dans la commande. La commande déplace les axes concernés dans la mesure du possible, en → mode rapide.

Actions synchrones

1. Sortie de fonct. auxiliaire

   Pendant l'usinage de la pièce, il est possible de sortir du programme CNC des fonctions technologiques (→ fonctions auxiliaires) pour les transmettre à l'AP. A l'aide de ces fonctions auxiliaires, il est possible, par exemple, de piloter des équipements supplémentaires de la machine-outil tels que fourreaux, préhenseurs, mandrins, etc.

2. Sortie de fonctions auxiliaires rapide

   Dans le cas de fonctions de commutation critiques en temps, il est possible de minimiser les délais d'acquittement pour les → fonctions auxiliaires et d'éviter les points d'arrêt inutiles dans le processus d'usinage.

Adresse

L'adresse permet d'identifier un opérande ou une plage d'opérandes déterminée, par ex. une entrée, une sortie etc.

Adresse d'axe

Voir → Descripteur d'axe

Alarmes

Tous les → messages et alarmes sont affichés à l'écran du panneau de commande, en texte clair, avec leur horodatage et le symbole correspondant au critère d'effacement. Les alarmes sont affichées séparément des messages.

1. Alarmes et messages dans le programme pièce

   L'affichage en texte clair des alarmes et messages peut être déclenché directement dans le programme pièce.

2. Alarmes et messages de l'AP

   Les alarmes et messages de la machine peuvent être affichés en texte clair directement à partir du programme AP. Des kits de blocs fonctionnels supplémentaires ne sont pas nécessaires.

Anticipation

La fonction **Look Ahead** est une fonction d'anticipation qui permet le pilotage optimal de la vitesse d'usinage, un nombre paramétrable de blocs de déplacement à l'avance.

AP

Automate **P**rogrammable : → Automate programmable. Composant de la → CN : Commande d'adaptation pour le traitement de la logique de la machine-outil.

Archivage

Sortie de fichiers et/ou répertoires sur une unité de mémorisation **externe**.

Arrêt orienté de broche

Arrêt de la broche de pièce dans une position angulaire prédéfinie, par exemple pour entreprendre un usinage complémentaire à un emplacement déterminé.

Arrêt précis

Dans le cas d'une instruction d'arrêt précis programmée, la position indiquée dans un bloc peut être accostée de façon précise et éventuellement très lentement. Pour limiter la durée d'ap proche, des → limites d'arrêt précis sont définies pour le mode rapide et l'avance.

Auto

Mode de fonctionnement d'un système à CN Mode de fonctionnement des systèmes de commandes numériques dans lequel un → programme pièce est sélectionné et exécuté de façon continue.

Automate programmable

Les automates programmables (AP) sont des commandes électroniques dont la fonction est mémorisée sous la forme d'un programme dans le mécanisme de commande. La structure et le câblage de l'appareil ne dépendent donc pas de la fonction de l'automate. L'automate programmable possède la structure d'un calculateur. Elle se compose d'une CPU (unité centrale) avec mémoire, cartes d'entrée/sortie et système de bus interne. La périphérie et le langage de programmation sont orientés vers les besoins de la technique de commande.

Avance en inverse du temps

Avec SINUMERIK 840D, à la place de la vitesse d'avance pour le déplacement des axes, vous pouvez programmer le temps nécessaire au trajet d'un bloc (G93).

Avance tangentielle

L'avance tangentielle agit sur les si possible → axes à interpolation. Elle correspond à la somme géométrique des avances des → axes géométriques.

Axe à arrondissage

Les axes à arrondissage provoquent une rotation de la pièce ou de l'outil dans une position angulaire correspondant à une grille d'indexation. L'axe à arrondissage est considéré comme étant "en position" lorsque l'incrément d'indexation est atteint.

Axe à interpolation

Les axes à interpolation sont tous les axes d'usinage du → canal qui sont pilotés par l' → interpolateur de telle manière qu'ils démarrent, accélèrent, s'arrêtent et atteignent le point final simultanément.

Axe C

Axe de la broche porte-pièce autour duquel des mouvements de rotation et de positionnement pilotés ont lieu.

Axe de base

Axe dont la valeur réelle ou la valeur de consigne est prise en compte pour le calcul d'une valeur de compensation.

Axe de compensation

Axe dont la valeur de consigne et la valeur réelle sont modifiées par la valeur de compensation.

Axe de positionnement

Axe exécutant un mouvement auxiliaire sur une machine-outil, (par exemple, magasin d'outils, transport de palettes). Les axes de positionnement sont des axes qui ne sont pas en interpolation avec les → axes d'interpolation.

Axe de synchronisme

L'axe synchrone est → l'axe Gantry dont la position de consigne est toujours dévirée du déplacement de → l'axe directeur, et dont le déplacement est donc synchrone. Du point de vue de l'utilisateur et du programmeur, l'axe synchrone "n'existe pas".

Axe directeur

L'axe directeur est → l'axe Gantry qui existe du point de vue de l'utilisateur et du programmeur, et qui peut donc être influencé comme un axe CN normal.

axe géométrique

Les axes géométriques permettent la description d'un domaine à deux ou trois dimensions dans le système de coordonnées pièce.

Axe linéaire

L'axe linéaire est un axe qui, contrairement à l'axe circulaire, décrit une droite.

Axe rotatif

Les axes rotatifs provoquent une rotation de la pièce ou de l'outil dans une position angulaire prédéfinie.

Axe synchrone

Pour effectuer leur course, les axes synchrones ont besoin du même temps que les axes géométriques pour leur trajectoire.

Axes

Selon leurs fonctionnalités, les axes CNC sont classés en deux catégories :

- Axes : à interpolation
- axes auxiliaires : axes de déplacement et de positionnement sans interpolation, mais avec une avance spécifique. Les axes auxiliaires ne sont généralement pas concernés par l'usinage proprement dit, par exemple les axes de dispositifs de chargement d'outils, de magasins d'outils.

Axes machine

Dans le cas d'une machine-outil, axes ayant une existence matérielle.

Bloc

On désigne par "blocs" tous les fichiers nécessaires à l'élaboration et à l'exécution du programme.

Bloc de données

1. Entité de données de → l'AP, accessible pour les programmes → HIGHSTEP.
2. Entité de données de la → CN : Les blocs de données contiennent des définitions de données utilisateur globales. Ces données peuvent être initialisées directement dans la définition.

Bloc de programme

Les blocs de programme contiennent les programmes principaux et sous-programmes des → programmes pièce.

Bloc de programme pièce

Partie d'un → programme pièce, par un LF (Line Feed). On distingue les → blocs principaux et les → blocs auxiliaires.

Bloc principal

Bloc précédé de ":" contenant toutes les indications pour pouvoir exécuter une tâche dans un → programme pièce.

Bloc secondaire

Bloc introduit par "N", comportant des informations pour une opération, par exemple une indication de position.

Blocs intermédiaires

Les déplacements avec → correction d'outil sélectionnée (G41/G42) peuvent être interrompus par un nombre limité de blocs intermédiaires (blocs sans mouvement axial dans le plan de correction). Dans ce cas, la correction d'outil peut encore être prise en compte. Le nombre autorisé de blocs intermédiaires que la commande peut lire à l'avance peut être réglé par l'intermédiaire d'un paramètre système.

Câble de liaison

Les câbles de liaison sont constitués de 2 conducteurs et de 2 connecteurs ; ils peuvent être prééquipés ou assemblés par l'utilisateur. Ces câbles permettent de relier la → CPU avec une → console de programmation ou d'autres CPU via l'→ interface multipoint (MPI).

Canal

Un canal se distingue par le fait qu'il est en mesure d'exécuter un → programme pièce indépendamment des autres canaux. Le canal pilote exclusivement les axes et broches qui lui sont assignés. Les séquences d'un programme pièce issues de canaux différents peuvent être coordonnées par une → synchronisation.

Canal d'usinage

La structure multicanal permet de réduire les temps morts grâce à des déplacements en parallèle (par exemple le déplacement d'un portique de chargement simultanément à l'usinage). Un canal CNC est une commande CNC à part entière avec décodage, prétraitement des blocs et interpolation.

CN

Numerical Control : Une commande numérique comprend tous les constituants nécessaires à la commande d'une machine-outil : → NCK, → PLC, HMI, → COM.

Remarque

Pour les commandes SINUMERIK 840D, le terme de commande CNC serait plus approprié : Computerized Numerical Control.

CNC

Voir → CN

Code de programmation

Caractères et chaînes de caractères ayant une signification définie dans le langage de programmation pour les → programmes pièce.

COM

Module de la commande numérique pour l'exécution et la coordination de la communication.

Commande anticipatrice, dynamique

Les imprécisions de → contour dues aux écarts de traînage, peuvent être pratiquement éliminées à l'aide d'une commande anticipatrice dynamique dépendante de l'accélération. Il en résulte une précision d'usinage remarquable, même pour des → vitesses tangentielles élevées. La commande anticipatrice peut être sélectionnée et désélectionnée pour chaque axe, par l'intermédiaire du → programme pièce.

Commutateur à clé

Le commutateur à clé situé sur le → tableau de commande machine possède 4 positions, qui sont affectées à des fonctions par le système d'exploitation de la commande. Le commutateur à clé peut être actionné à l'aide de trois clés de différentes couleurs qui peuvent être retirées dans les positions indiquées.

Compensation avec interpolation

A l'aide de la compensation avec interpolation, les **erreurs de pas de vis** et les **erreurs du système de mesure** dues à la production peuvent être compensées (SSFK, MSFK).

Compensation d'erreur de pas de vis de transmission

Compensation, par la commande, des imprécisions mécaniques d'une vis à bille participant à l'avance, au moyen de valeurs mémorisées de mesure des écarts.

Compensation des défauts aux transitions entre quadrants

Les erreurs de contour survenant aux jonctions entre les quadrants, qui résultent de rapports de frottement variables au niveau des glissières, peuvent en grande partie être éliminées au moyen d'une compensation des défauts aux transitions entre quadrants. Le paramétrage de cette compensation passe par un test de circularité.

Compensation du jeu

Compensation du jeu mécanique d'une machine, par exemple dans le cas de vis à bille. Pour chaque axe, il est possible d'indiquer une compensation de jeu distincte.

Contour

Contour de la → pièce

Contour de la pièce finie

Contour de la pièce qui a été usinée. Voir → Pièce brute.

Contour de pièce

Contour programmé de la → pièce à fabriquer/usiner.

Contournage

Le but du contournage est d'éviter une décélération importante des → axes à interpolation aux limites de bloc du programme pièce et de passer au bloc suivant si possible avec la même vitesse tangentielle.

Coordonnées polaires

Système de coordonnées permettant de définir la position d’un point dans un plan par la distance qui sépare ce point de l’origine et par l’angle formé par le rayon vecteur avec un axe défini.

correction

Possibilité d'intervention manuelle ou programmable, qui permet à l'utilisateur de corriger des avances ou des vitesses de rotation programmées, pour les adapter à une pièce ou à un matériau spécifique.

Correction d'outil

Prise en compte des dimensions de l'outil lors du calcul de la trajectoire.

Correction de l'avance par commutateur

La correction de vitesse d'avance réglée par commutateur au → tableau de commande machine ou prescrite par l'→ AP (0-200%) se superpose à la vitesse d'avance programmée.). La vitesse d'avance programmée peut faire l'objet d'une correction supplémentaire dans le programme d'usinage, par le biais d'un facteur programmable (de 1 à 200%).

Correction de rayon de l'outil

Pour pouvoir programmer directement un → contour de pièce souhaité, la commande doit piloter, en tenant compte du rayon de l'outil utilisé, une trajectoire équidistante pour le contour programmé (G41/G42).

Correction de rayon de plaquette

La programmation d'un contour est basée sur le concept d'un outil parfaitement pointu ou tranchant. Ceci n'étant pas réalisable en pratique, le rayon du bec de la plaquette ou du tranchant de l'outil doit être indiqué à la commande numérique. Celle-ci le prend en compte et pilote le centre de la courbure selon une trajectoire décalée de la valeur du rayon et équidistante au contour.

Cote absolue

Indication de la destination du déplacement d'un axe par une cote qui se rapporte à l'origine du système de coordonnées sélectionné. Voir → Cote relative.

Cote relative

Aussi appelée cote incrémentale : Programmation du point final de déplacement d'un axe au moyen d'une distance à parcourir et d'un sens, par référence à un point déjà atteint. Voir → Cote absolue.

Courbure

La courbure d'un contour est l'inverse du rayon r du cercle osculateur (cercle de courbure) : k = 1/r.

CPU

Central Processing Unit, voir → Automate programmable

Cycles

Sous-programmes protégés pour l'exécution d'usinages récurrents sur la → pièce.

Cycles standard

Des cycles standard sont disponibles pour les tâches d'usinage répétitives :

- pour le perçage/fraisage
- pour la technologie tournage

Dans le groupe fonctionnel "Programme", les cycles disponibles sont affichés dans le menu "Assistance aux cycles". Après sélection du cycle d'usinage souhaité, les paramètres sont affichés en clair pour l'affectation de valeurs.

Débit de transfert

Vitesse (bits/s) à laquelle une transmission de données a lieu.

Décalage d'origine

Définition d'un nouveau point de référence pour un système de coordonnées, réglable en fonction d'une origine existante et d'un → frame.

1. réglable

   SINUMERIK 840D : Un nombre configurable de décalages d'origine réglables est disponible pour chaque axe CNC. Les décalages sélectionnables par l'intermédiaire des fonctions G peuvent également être activés.

2. Externe

   En plus de tous les décalages qui définissent la position de l'origine pièce, on peut superposer un décalage d'origine externe par manivelle (décalage DRF) ou par le biais de l'AP.

3. Programmable

   Avec l'instruction `TRANS`, des décalages d'origine sont programmables pour tous les axes d'interpolation et de positionnement.

Décalage d'origine externe

Décalage d'origine imposé par → l'AP.

Définition de variables

Une définition de variable comprend la définition d'un type de données et un nom de variable. Avec le nom de la variable, la valeur des variables peut être activée.

déplacement en manuel incrémental

Indication de la longueur du trajet au moyen d'un nombre d'incréments (incrément). Le nombre d'incréments peut être mémorisé sous la forme de données de réglage ou sélectionné au moyen des touches marquées 10, 100, 1000, 10000.

des mots-clés

Mots ayant une notation fixe et qui, dans le langage de programmation, ont une signification particulière pour des → programmes pièces.

Descripteur

Les mots de la norme DIN 66025 sont complétés par des descripteurs (noms) de variables (variables de calcul, variables système, variables utilisateur), de sous-programmes, de mots-clés et de mots à plusieurs lettres adresse. Ces adjonctions ont la même signification que les mots dans la syntaxe des blocs. Les descripteurs doivent être uniques. Un même descripteur ne peut être utilisé pour désigner plusieurs objets.

Descripteur d'axe

Selon DIN 66217, les axes sont désignés par X, Y, Z dans un → système de coordonnées cartésiennes.

Les descripteurs A, B, C sont affectés aux → axes rotatifs tournant autour de X, Y, Z. Les axes additionnels, parallèles aux axes indiqués, peuvent être caractérisés par d'autres adresses.

Diagnose

1. Groupe fonctionnel de la commande
2. La commande est dotée d'un programme d'autodiagnostic et d'aides aux tests pour la maintenance : Affichage d'états, d'alarmes et de données pour la maintenance

Données de réglage

Données qui communiquent des caractéristiques de la machine-outil à la commande numérique selon une procédure définie par le logiciel système.

DRF

Differential resolver function : Fonction de résolveur différentiel. Cette fonction entraîne un décalage d'origine incrémental en relation avec une manivelle électronique en mode automatique.

Editeur

L'éditeur permet de créer, de modifier, de compléter, de fusionner et d'insérer des programmes/textes/blocs de programme.

Editeur de texte

Voir → Editeur

Effacement général

Lors de l'effacement général, les mémoires suivantes de la → CPU sont effacées :

- la → mémoire de travail,
- la zone de lecture/d'écriture de la → mémoire de chargement,
- la → mémoire système,
- la → mémoire de sauvegarde.

Entraînement

L'entraînement est l'unité de la CN, qui exécute la régulation de vitesse de rotation et de couple en fonction des consignes de la CN.

Entrées/sorties numériques rapides

Via les entrées TOR, il est possible de lancer par exemple des routines de programme CNC rapides (routines d'interruption). Au moyen des sorties TOR de la CNC, il est possible de déclencher des fonctions de commutation rapides pilotées par le programme (SINUMERIK 840D).

Fin de course logiciel

Les fins de course logiciels limitent la plage de déplacement d'un axe et empêchent l'arrêt brutal du chariot sur l'interrupteur de fin de course matériel. Pour chaque axe, 2 paires de valeurs peuvent être spécifiées et activées individuellement via l'→ AP.

Fonction miroir

La fonction miroir inverse les signes des coordonnées d’un contour pour un axe. La fonction miroir peut être appliquée simultanément à plusieurs axes.

Fonctions auxiliaires

Les fonctions auxiliaires permettent de transférer, dans les → programmes pièces, des → paramètres à un → AP, qui, à ce niveau, déclenchent des réactions définies par le constructeur de la machine.

Fonctions de sécurité

La commande dispose de fonctions de surveillance actives en permanence, capables de détecter les défaillances dans la → CNC, la commande d'adaptation (→ AP) et la machine suffisamment tôt, de telle sorte que les dommages peuvent être en grande partie évités sur la pièce, l'outil ou la machine. En cas de défaillance, le déroulement de l'usinage est interrompu et les entraînements sont arrêtés, la cause de la défaillance sauvegardée en mémoire et affichée sous forme d'alarme. Dans le même temps, la présence d'une alarme CNC est signalée à l'AP.

Frame

Un frame est une règle opératoire qui transpose un système de coordonnées cartésiennes en un autre système de coordonnées cartésiennes. Le frame contient les composants → décalage d'origine , → rotation , → mise à l'échelle , → fonction miroir.

Frames programmables

Avec les → frames programmables, il est possible de définir en dynamique, au cours du traitement du programme pièce, de nouveaux points d'origine du système de coordonnées. Une distinction est opérée entre la définition absolue, au moyen d'un nouveau frame et la définition additive, par référence à un point d'origine existant.

Gain

Gain de boucle, grandeur de régulation d'une boucle de régulation.

Géométrie

Description d'une → pièce dans le → système de coordonnées pièce.

Gestion du programme pièce

La gestion des programmes pièce peut être organisée d'après les → pièces. La taille de la mémoire de travail détermine le nombre des programmes et données à gérer. Chaque fichier (programmes et données) peut se voir attribuer un nom ne comportant pas plus de 24 caractères alphanumériques.

Groupe à mode de fonctionnement commun

Des axes et des broches de même appartenance technologique peuvent être regroupés en un groupe à mode de fonctionnement commun (GMFC). Les axes et les broches regroupés dans un GMFC peuvent être commandés par un ou plusieurs → canaux. Les canaux du GMFC se trouvent toujours dans le même → mode de fonctionnement.

HIGHSTEP

Synthèse des possibilités de programmation pour → l'AP du système AS300/AS400.

Homothétie

Elément d'un → frame réalisant des changements d’échelle spécifiques aux axes.

Interface série V.24

Pour la saisie/sortie des données, une interface série V.24 (RS232) se trouve sur la PCU 20 et deux interfaces V.24 sont sur la PCU 50/70. Par l'intermédiaire de ces interfaces, il est possible de charger et de sauvegarder des programmes d'usinage ainsi que des données constructeur et utilisateur.

Interface utilisateur

L'interface utilisateur (IU) permet la communication entre l'utilisateur et la commande CNC. Elle est constituée d'un écran qui comporte des touches logicielles horizontales et verticales.

Interpolateur

Unité logique du → NCK qui détermine, en fonction de la position de destination dans le programme pièce, des valeurs intermédiaires pour les déplacements à effectuer sur les différents axes.

Interpolation circulaire

→ L'outil doit se déplacer sur un cercle, entre des points définis du contour et avec une avance donnée, pour l'usinage de la pièce.

Interpolation de type spline

Avec l'interpolation de type spline, la commande peut créer une forme de courbe lisse à partir d'un petit nombre de points intermédiaires d'un contour de consigne.

Interpolation hélicoïdale

L'interpolation hélicoïdale est particulièrement adaptée à la fabrication simple de filetages intérieurs ou extérieurs avec des fraises de forme et pour le fraisage de rainures de graissage.

Dans ce cas, l'hélice se compose de deux déplacements :

- Déplacement circulaire dans un plan
- Déplacement linéaire perpendiculaire à ce plan

Interpolation linéaire

L'outil est déplacé sur une droite jusqu'au point cible et la pièce est alors usinée.

interpolation polynomiale

L'interpolation polynomiale permet de générer les formes de courbe les plus diverses, notamment **droite, parabole, fonctions exponentielles** (SINUMERIK 840D).

JOG

Mode de fonctionnement de la commande (mode de réglage) : Le mode de fonctionnement JOG permet de régler la machine. Des axes ou broches spécifiques peuvent être déplacés au moyen des touches de sens en mode manuel. Autres fonctionnalités du mode Jog : → Accostage du point de référence, → Repos et → Preset (Forcer la valeur réelle).

Langage évolué CNC

Ce langage évolué offre les éléments de programmation suivants : → Variable définie par l'utilisateur, → Variable système, → Macroprogrammation.

Limitation de la zone de travail

Cette fonctionnalité permet de limiter la plage de déplacement des axes, en plus des fins de course. Pour chaque axe, il est possible de définir une paire de valeurs décrivant la zone de travail protégée.

Limitation programmable de la zone de travail

Restriction de la zone de déplacement de l'outil sur un espace défini par des limitations programmées.

Limite d'arrêt précis

Si tous les axes d'interpolation atteignent leur limite d'arrêt précis, la commande se comporte comme si elle avait atteint précisément son objectif. Cet état est suivi d'un changement de bloc du → programme pièce.

Macroprogrammation

Regroupement d'une certaine quantité d'instructions sous un descripteur. Le descripteur représente, au sein du programme, la quantité d'instructions regroupées.

Masse

La masse correspond à la totalité des parties inactives reliées entre elles sur un moyen d'exploitation et ne pouvant pas adopter une tension dangereuse par contact, même en cas d'anomalie.

MDA

Mode de fonctionnement de la commande : Manual Data Automatic. Dans le mode de fonctionnement MDA, des blocs de programme et des séquences de blocs peuvent être introduits et exécutés immédiatement avec la touche Départ programme (sans exécution d'un programme principal ou sous-programme).

Mémoire de chargement

Dans le cas de la CPU 314, la mémoire de chargement de → l'AP est équivalente à la → mémoire de travail.

Mémoire de corrections

Zone de données de la commande, dans laquelle des données de correction d'outil sont mémorisées.

Mémoire de programmes AP

SINUMERIK 840D : Dans la mémoire de travail AP, le programme AP utilisateur et les données utilisateur sont mémorisés ensemble avec le programme de base de l'AP.

mémoire de travail

La mémoire vive est une mémoire RAM se trouvant dans la → CPU et dans laquelle le processeur accède au programme utilisateur durant l'exécution du programme.

Mémoire système

La mémoire système est une mémoire de la CPU, dans laquelle les données ci-après sont sauvegardées :

- Données nécessaires au système d'exploitation
- Opérandes de temps, compteurs, mémentos.

mémoire utilisateur

Tous les programmes et toutes les données, tels que les programmes pièce, les sous-programmes, les commentaires, les corrections d'outil, les décalages d'origine / frames ainsi que les données utilisateur de programme et de canal, peuvent être rangés dans une mémoire CNC utilisateur commune.

Messages

Tous les messages et → alarmes programmés dans le programme pièce et reconnus par le système sont affichés à l'écran du tableau de commande, en texte clair, avec leur horodatage et le symbole correspondant au critère d'effacement. Les alarmes sont affichées séparément des messages.

Mode de fonctionnement

Type de fonctionnement d'une commande SINUMERIK. Les modes de fonctionnement → Jog, → MDA, → Automatique sont définis.

Module de périphérie

Les modules de périphérie établissent la relation entre unité centrale et processus.

Les modules de périphérie sont :

- → les modules d'entrées/sorties TOR,
- → les modules d'entrées/sorties analogiques,
- → les modules de simulation.

Mot de données

Entité de données de deux octets faisant partie d'un → bloc de données.

NCK

Numerical control kernel : Composants de la commande numérique qui exécutent les → programmes pièces et surtout coordonnent les déplacements pour la machine-outil.

Nom d'axe

Voir → Descripteur d'axe

NRK

Numeric Robotic Kernel (système d'exploitation du → NCK)

NURBS

Le pilotage des déplacements et l'interpolation de trajectoire dans la commande sont exécutés sur la base de NURBS (**N**on **U**niform **R**ational **B**-**S**plines). Ainsi, avec SINUMERIK 840D, pour toutes les interpolations, il existe un déplacement unifié qui est interne à la commande.

OEM

Pour les constructeurs de machine qui veulent créer leur propre interface utilisateur ou introduire des fonctions spécifiques à une technologie dans la commande, des aménagements sont prévus pour des solutions individuelles (applications OEM) pour le système SINUMERIK 840D.

Origine machine

Point fixe de la machine-outil, auquel se rapportent tous les systèmes de mesure (déduits).

Origine pièce

L'origine pièce est l'origine du → système de coordonnées pièce. Elle est définie par des distances par rapport à l'→ origine machine.

Outil

Objet monté sur la machine-outil dont la pointe ou le tranchant génère la forme à obtenir (par ex. outil de tournage, fraise, foret, faisceau LASER ...)

Paramètres R

Paramètres de calcul ; peuvent être renseignés ou interrogés dans le → programme pièce aux fins jugées utiles par le programmeur.

Pièce

Pièce à fabriquer/usiner sur la machine-outil.

Pièce brute

Pièce à l'état brut, avant le commencement de l'usinage.

Pile de sauvegarde

La pile de sauvegarde permet la protection du → programme utilisateur de la → CPU contre les coupures du réseau et la sauvegarde des zones de données, mémentos, horodatages et compteurs de façon rémanente.

Pilotage de la vitesse

Dans le cas de mouvements de déplacement de très faible amplitude par bloc, il est possible, pour pouvoir atteindre une vitesse acceptable, de faire appel à une évaluation anticipée sur plusieurs blocs (→ Look Ahead).

Plage de déplacement

La plage de déplacement maximale autorisée pour les axes linéaires recouvre ± 9 décades. La valeur absolue dépend de la définition et de la résolution sélectionnées du régulateur de position et du système d'unités (inch ou métrique).

Point de référence

Point de la machine-outil, auquel le système de mesure des → axes machine fait référence.

Point machine fixe

Point défini de façon univoque par la machine-outil, par exemple point de référence machine.

Pré-coïncidence

Changement de bloc déjà effectué si le trajet s'est approché de la position finale d'un delta prédéterminé.

Programmation AP

L'AP est programmé à l'aide du logiciel **STEP 7**. Le logiciel de programmation STEP 7 est basé sur le système d'exploitation **WINDOWS** et contient les fonctions de la programmation STEP 5, avec des perfectionnements innovateurs.

Programme de transfert des données PCIN

PCIN est un utilitaire pour l’émission et la réception de données utilisateur CNC via l’interface série, telles que programmes pièce, corrections d’outils, etc. Le programme PCIN est exécutable sous MSDOS sur PC industriel standard.

Programme pièce

Suite d'instructions transmises à la commande numérique pour la fabrication d'une → pièce déterminée. Réalisation d'un usinage défini sur une → pièce brute.

Programme principal

→ Programme pièce, désigné par un numéro ou par un descripteur, dans lequel il est possible d'appeler d'autres programmes principaux, sous-programmes ou → cycles.

Programme utilisateur

Les programmes utilisateur destinés aux systèmes d'automatisation S7-300 sont créés avec le logiciel de programmation STEP 7. De conception modulaire, ils sont constitués de différents blocs.

Les principaux types de blocs sont les suivants :

- Blocs de codes

  Ces blocs contiennent les instructions STEP 7.

- Blocs de données

  Ces blocs contiennent des constantes et des variables pour le programme STEP 7.

Recherche de bloc

La fonction "Recherche de blocs" permet de rechercher une partie quelconque du programme pièce où l'usinage doit être démarré ou repris (pour le test de programmes pièce ou après une interruption de l'usinage).

Redémarrage

Chargement du programme après une mise sous tension.

Réseau

Un réseau est une association de plusieurs S7-300 et d'autres terminaux, par exemple une console de programmation, reliés entre eux au moyen d'un → câble de liaison. Par l'intermédiaire du réseau, il y a un échange de données entre les appareils connectés.

Retrait orienté d'outil

`RETTOOL` : En cas d'interruption d'une opération d'usinage (par exemple, rupture d'outil), l'outil peut être retiré par le biais d'une instruction, avec une orientation spécifiée, sur une distance définie.

Retrait rapide du contour

Lorsqu'une interruption est rencontrée, il est possible d'engager un mouvement au moyen du programme d'usinage CNC, qui rend possible un retrait rapide de l'outil hors du contour de la pièce qui vient d'être usinée. En outre, l'angle de retrait et la distance de retrait peuvent être programmés. Un retrait rapide peut être suivi de l'exécution d'une routine d'interruption supplémentaire (SINUMERIK 840D).

Rotation

Elément d'un → frame définissant la rotation d'un système de coordonnées d'un angle défini.

Routines d'interruption

Les routines d'interruption sont des → sous-programmes spécifiques qui peuvent être lancées par les événements (signaux externes) du processus de traitement. Lorsqu'un bloc de programme pièce est interrompu en cours de traitement, la position d'interruption des axes est automatiquement enregistrée.

RT

Rapport de transmission

Sous-programme

Séquence d'instructions d'un → programme pièce pouvant être appelée plusieurs fois avec des valeurs de paramètres différentes. Les sous-programmes sont appelés à partir d'un programme principal. Tous les sous-programmes peuvent être protégés contre la lecture et l'affichage par des personnes non autorisées. Les → cycles sont une forme de sous-programmes.

Sous-programme asynchrone

Programme pièce qui peut être démarré de façon asynchrone (indépendamment de l'état du programme actuel) par un signal d'interruption (par ex. signal "Entrée CN rapide").

Spline C

Le spline C est le spline le plus connu et le plus utilisé. Les raccordements au niveau des points intermédiaires sont caractérisés par la continuité de la tangente et du rayon de courbure. Des polynômes de 3me degré sont utilisés.

Surveillance du contour

Pour évaluer la précision du contour, on utilise l'écart de traînage à l'intérieur d'une plage de tolérance définie. Un écart de traînage excessivement élevé peut s'expliquer, par exemple, par une surcharge du système d'entraînement. Dans ce cas, une alarme se déclenche et les axes sont immobilisés.

Synchronisation

Instructions figurant dans les → programmes pièce pour la coordination des séquences dans les différents → canaux, à des emplacements d'usinage définis.

Système anglo-saxon

Système de mesure dans lequel les distances sont exprimées en "inch" et en fractions d'inch.

Système de coordonnées

Voir → Système de coordonnées machine, → Système de coordonnées pièce

Système de coordonnées de base

Système de coordonnées cartésiennes qui découle du système de coordonnées machine par une transformation.

Dans le → programme pièce, le programmeur utilise les noms des axes du système de coordonnées de base. Si aucune → transformation n'est active, le système de coordonnées de base est parallèle au → système de coordonnées machine. Les deux systèmes ne diffèrent que par les → descripteurs d'axe.

Système de coordonnées machine

Système de coordonnées référencé aux axes de la machine-outil.

Système de coordonnées pièce

L'origine du système de coordonnées pièce est l'→ origine pièce. Dans les programmes écrits dans un système de coordonnées pièce, les cotes et les sens sont définis par rapport à ce système.

Système de mesure en métrique et en inch

Dans le programme d'usinage, les valeurs de position et valeurs de pas peuvent être programmées en inch. Indépendamment du système de mesure programmable (G70/G71) la commande est réglée sur un système de base.

Système de mesure métrique

Système normalisé d'unités : pour les longueurs, par exemple mm (millimètre), m (mètre).

Table de compensation

Table des points intermédiaires. Elle indique, pour des positions sélectionnées de l'axe de base, les valeurs de compensation de l'axe de compensation.

Tableau de commande machine

Tableau de commande de la machine-outil comportant des organes de commande tels que touches, commutateurs rotatifs, etc. et des éléments d'affichage simples tels que les LED. Il permet d'exercer une influence indirecte sur la machine-outil par l'intermédiaire de l'AP.

Taraudage sans porte-taraud compensateur

Cette fonction permet de percer des filetages sans porte-taraud compensateur. A travers le processus d'interpolation de la broche comme un axe circulaire et axe de perçage, les filetages sont découpés précisément à la profondeur finale, par exemple filetages à trou borgne (condition préalable : utilisation de l'axe de la broche).

Touche logicielle

Touche dont le libellé est représenté par un champ de l'écran, qui s'adapte dynamiquement à la situation de commande courante. Les touches de fonction librement affectables (softkeys) sont assignées par voie logicielle à des fonctions définies.

Transformation

Décalage d'origine additif ou absolu d'un axe.

Unité TOA

Chaque → zone TOA peut contenir plusieurs unités TOA. Le nombre des unités TOA possibles est limité via le nombre maximal de → canaux actifs. Une unité TOA comprend un seul bloc de données d'outils et un seul bloc de données de magazine. Un bloc de données de porte-outil peut également être contenu (facultatif).

Usinage de surfaces obliques

Les opérations d'alésage et de fraisage portant sur des surfaces de pièce qui ne se trouvent pas dans les plans de coordonnées de la machine peuvent être exécutées confortablement avec l'aide de la fonction Usinage de surfaces obliques.

Valeur de compensation

Différence entre la position d'axe mesuré par l'indicateur de mesure et la position d'axe programmée et souhaitée.

Variable définie par l'utilisateur

L'utilisateur peut, à des fins quelconques, déclarer des variables personnalisées dans le → programme pièce ou dans un bloc de données (données utilisateur globales). Une déclaration de variable contient une indication de type de données et le nom de la variable. Voir → Variable système.

Variable système

Variable existant indépendamment de l'intervention du programmeur d'un → programme pièce. Elle est définie par un type de données et par un nom de variable commençant par **$**. Voir → Variable définie par l'utilisateur.

Vitesse limite mécanique

Vitesse de rotation maximale/minimale (de la broche) : A travers la détermination des données machine de → l'AP ou des → données de réglage, il est possible de limiter la vitesse de rotation maximale d'une broche.

Vitesse rapide

Vitesse de déplacement la plus rapide d'un axe. Le rapide est utilisé par exemple pour approcher l'outil du → contour de la pièce à partir d'une position de repos ou pour dégager l'outil du contour de la pièce. La vitesse rapide se règle dans un paramètre machine. Le réglage est spécifique à la machine.

Vitesse tangentielle

La vitesse tangentielle maximale programmable dépend de la résolution d'introduction (définition). Dans le cas d'une résolution d'introduction de 0,1 mm par exemple, la vitesse tangentielle maximale programmable vaut 1000 m/min.

WinSCP

WinSCP est un programme Open Source pour Windows, qui est disponible gratuitement et qui est utilisé pour le transfert de fichiers.

Zone de protection

Espace tridimensionnel à l'intérieur de l'→ espace de travail dans lequel la pointe de l'outil ne doit pas pénétrer.

Zone de travail

Zone tridimensionnelle dans laquelle la pointe d'outil peut se trouver, compte tenu de la structure de la machine-outil. Voir → Volume de protection.

Zone TOA

La zone TOA comprend toutes les données d'outils et de magasins. Par défaut, la zone relative à la portée des données coïncide avec la zone du → canal. Par contre, des paramètres machine permettent de déterminer que plusieurs canaux partagent une → unité TOA de sorte que des données de gestion d'outil communes soient ensuite disponibles pour ces canaux.

Indice

"

"GET" automatique, 124

$

$AA_ATOL, 493
$AA_COUP_ACT, 459, 500, 525
$AA_LEAD_SP, 525
$AA_LEAD_SV, 525
$AA_MOTEND, 276
$AA_TOFF[ ], 596
$AC_ACT_PROG_NET_TIME, 693
$AC_ACTUAL_PARTS, 697
$AC_BLOCKTYPE, 573
$AC_BLOCKTYPEINFO, 573
$AC_CTOL, 493
$AC_CUT_INV, 450
$AC_CUTMOD, 450
$AC_CUTMOD_ANG, 450
$AC_CUTTING_TIME, 693
$AC_CYCLE_TIME, 693
$AC_FIFO1, 571
$AC_MARKER, 566
$AC_OLD_PROG_NET_TIME, 693
$AC_OLD_PROG_NET_TIME_COUNT, 693
$AC_OPERATING_TIME, 693
$AC_OTOL, 493
$AC_PARAM, 566
$AC_PROG_NET_TIME_TRIGGER, 694
$AC_REQUIRED_PARTS, 697
$AC_SMAXVELO, 489
$AC_SMAXVELO_INFO, 489
$AC_SPECIAL_PARTS, 697
$AC_SPLITBLOCK, 573
$AC_STOLF, 496
$AC_TIMER, 570
$AC_TOTAL_PARTS, 697
$AN_POWERON_TIME, 692
$AN_SETUP_TIME, 692
$MC_COMPESS_VELO_TOL, 464
$P_AD, 451
$P_CTOL, 494
$P_CUT_INV, 450
$P_CUTMOD, 450
$P_CUTMOD_ANG, 450
$P_OTOL, 494
$P_STOLF, 496
$P_SUBPAR, 151
$P_TECCYCLE, 629
$PA_ATOL, 494
$R, 567
$Rn, 567
$SA_LEAD_TYPE, 524, 525
$SC_PA_ACTIV_IMMED, 222
$SN_PA_ACTIV_IMMED, 222
$TC_CARR1...14, 434
$TC_CARR18[m], 434, 438
$TC_DP1, 390
$TC_DP10, 390
$TC_DP11, 390
$TC_DP12, 390
$TC_DP13, 390
$TC_DP14, 390
$TC_DP15, 390
$TC_DP16, 390
$TC_DP17, 390
$TC_DP18, 390
$TC_DP19, 390
$TC_DP2, 390
$TC_DP20, 390
$TC_DP21, 390
$TC_DP22, 390
$TC_DP23, 390
$TC_DP24, 390
$TC_DP25, 390
$TC_DP3, 390
$TC_DP4, 390
$TC_DP5, 390
$TC_DP6, 390
$TC_DP7, 390
$TC_DP8, 390
$TC_DP9, 390
$TC_ECPxy, 394
$TC_SCPxy, 394
$TC_TPG1 ... 9, 667, 668

*****

* (fonction de calcul), 61

/

/ (fonction de calcul), 61

+

+ (fonction de calcul), 61

<

< (opérateur de comparaison), 64
<<, 70
<< (opérateur de concaténation), 75
<= (opérateur de comparaison), 64
<> (opérateur de comparaison), 64

=

== (opérateur de comparaison), 64

>

> (opérateur de comparaison), 64
>= (opérateur de comparaison), 64

A

A1, A2, 434
A2, 323
A3, 323
A4, 323, 330
A5, 323, 330
A6, 335
A7, 335
AA, 607
ABS, 61
ACC, 540
Accostage au point de contour suivant, 483
Accostage de butée, 620
ACOS, 61
ACTBLOCNO, 164
ACTFRAME, 281
Action synchrone
  Action, 557
  Condition, 555
  Domaine de validité, 553
  Effacement, 636
  Eléments d'instructions, 552
  interruption, 636
  Positionnement d'un axe, 599
  Syntaxe, 552
Actions synchrones
  Variable d'exécution, 560
  Variables de prétraitement, 560
  Vue d'ensemble des actions, 576
ADISPOSA, 274
Adresses
  Programmation indirecte, 53
Adresses OEM, 272
Affichage de bloc, 190
  Inhibition, 164
Alarme, 699
  Comportement pour les actions synchrones, 639
  numéro, 699
Alarmes de cycle, 699ALF, 115, 117
AND, 64
Angle d'avance, 324
Angle de la tangente à la trajectoire, 623
Angle de rotation, 344
Angle final, 344
Angle latéral, 324
angle rotatif 1, 2, 434
Appel de sous-programme avec indication de chemin et paramètres, 191
Appel modal d'un sous-programme, 185
Apprentissage de compensations, 689
APR, 38
APRB, 38
APRP, 38
APW, 38
APWB, 38
APWP, 38
Arrondir au chiffre supérieur, 144
AS, 201
ASIN, 61
ASPLINE, 233
ASUP, 109
ATAN2, 61
ATOL, 491
Attributs de position
  Programmation indirecte, 57
AV, 535
Avance
  axiale, 607
Avance axiale, 607
avec TCOFRX, 439
avec TCOFRY, 439
AX, 669
AXCTSWE, 677
AXCTSWED, 677
Axe
  Ablocage, 677

Changement, 121
Conjugué, 499
locales, 678
oblique (TRAANG), 371
Prise en charge directe, 121
axe conjugué, 520
axe géométrique
Basculer, 672
Axe Link, 678
Axe oblique, TRAANG, 314
Axe pilote, 520
axeA, 520
AxeA, 453
axeP, 520
AxeP, 453
Axes de commande, 598
Axes d'orientation, 323, 331, 333
Axes FGROUP, 256
Axes géométriques permutables, 672
Axes rotatifs
vecteurs de direction V1, V2, 434
vecteurs distances l1, l2, 434
AXIS, 22
AXNAME, 74, 669
AXSTRING, 669
AXTOCHAN, 126
AXTOSPI, 669

B

B_AND, 64
B_NOT, 64
B_OR, 64
B_XOR, 64
B2, 323
B3, 323
B4, 323, 330
B5, 323, 330
B6, 335
B7, 335
BAUTO, 233
BFRAME, 281
BLOC, 188
Bloc d'arrêt, 474
Bloc par bloc
Inhibition, 158
blocage de l'introduction via l'interface., 580
BLSYNC, 111
BNAT, 233
BOOL, 22
Boucle de comptage, 99
Boucle de programme
Boucle, 98
Boucle de comptage, 99
Boucle IF, 96
Boucle REPEAT, 101
Boucle WHILE, 100
Boucle sans fin, 98
BOUND, 68
Broche
Changement, 121
Broche synchrone, 534
définir la paire, 540
paire, 534
rapport de transmission RT, 541
BSPLINE, 233
BTAN, 233

C

C2, 323
C3, 323
C4, 323, 330
C5, 323, 330
C6, 335
C7, 335
CAC, 231
CACN, 231
CACP, 231
CALCDAT, 715
Calcul d'un frame
MEAFRAME, 298
CALL, 187
CALLPATH, 192, 209
CANCEL, 636
Caractère 0, 72
CASE, 86
CDC, 231
CFINE, 293
Chaîne de caractères
Concaténation, 75
Longueur, 78
Opérations, 72
CHAN, 22
CHANDATA, 210
Changement de poste d'usinage/position, 677
CHAR, 22
Chariotage, 701
CHECKSUM, 142
Chemin de recherche
à l'appel du sous-programme, 148
Appel de sous-programme, 208
Chemin de recherche programmable, 192
CHKDNO, 430

CIC, 231
Cinématique
Décomposée, 438
Cinématique décomposée, 434
CLEARM, 103, 618
CLRINT, 114
CMIRROR, 61, 286
COARSE, 535
COARSEA, 274
Code G
Programmation indirecte, 56
Coefficient du polynôme, 251
COMCAD, 247
Compacteur, 247
Compacteur de bloc CN, 247
COMPCAD, 353
COMPCURV, 247, 353
Compensation des défauts aux transitions entre quadrants
Activation de la procédure d'apprentissage, 689
Désactivation de la procédure d'apprentissage, 689
réapprentissage, 690
COMPLETE, 210
COMPOF, 247, 353
COMPON, 247, 353, 464
Composante de frame
FI, 289
MI, 289
SC, 289
TR, 289
Composante de frameRT, 289
Concaténation
de chaînes de caractères, 75
Conditions marginales lors des transformations, 385
CONTDCON, 708
Conteneur d'axes, 677
Contour
Code, 708
Préparation, 702
Réaccostage, 476
Table, 702, 708
CONTPRON, 702
Contrôle
Structures, 95
Coordination d'axe, 608
Coordination de programmes
Noms des canaux, 105
Numéros de canaux, 105
Correction
résultante, 623
Correction d'outil
En ligne, 404
Mémoire de correcteurs, 389
Système de coordonnées pour valeurs d'usure, 400
Correction de la longueur d'outil en ligne, 443, 596
Correction de rayon de l'outil
décélération aux angles, 273
Fraisage périphérique 3D sans surfaces de délimitation, 419
Correction de rayon d'outil 3D
Angles rentrants/angles saillants, 417
arc de raccordement, 418
Fraisage en bout, 412
Fraisage périphérique, 411
Point d'intersection 3D des équidistantes, 418
Correction de rayon d'outil 3D, 409
Correction de vitesse
courante, 623
Correction d'outil
En ligne, 593
Correction d'outil 3D, 413
Correction sur la trajectoire, 415
Courbure de la trajectoire, 415
Fraisage périphérique:Avec surfaces de délimitation, 419
méthode du point d'intersection, 418
Orientation de l'outil, 423
Profondeur de pénétration, 415
COS, 61
COUPDEF, 535
COUPDEL, 535
coupl., 453
Couplage de deux axes par valeur pilote, 520
Couplage de vitesse, 537
Couplage par valeur de consigne, 537
Couplage par valeur pilote
à partir d'actions synchrones statiques, 521
Couplage par valeur réelle et couplage par valeur de consigne, 520, 524
Synchronisation de l'axe pilote et de l'axe asservi, 523
Couplage par valeur réelle, 537
COUPOF, 535
COUPOFS, 535
COUPON, 535
COUPONC, 535
COUPRES, 535
CP, 376
CPROT, 219
CPROTDEF, 215
Critère de fin de déplacement
programmable, 274
CROT, 61, 286
CSCALE, 61, 286

CSPLINE, 233
CT, 677
CTAB, 513
CTABDEF, 502
CTABDEL, 508
CTABEND, 502
CTABEXISTS, 508
CTABFNO, 518
CTABFPOL, 518
CTABFSEG, 518
CTABID, 511
CTABINV, 513
CTABISLOCK, 511
CTABLOCK, 510
CTABMEMTYP, 511
CTABMPOL, 518
CTABMSEG, 518
CTABNO, 518
CTABNOMEM, 518
CTABPERIOD, 511
CTABPOL, 518
CTABPOLID, 518
CTABSEG, 518
CTABSEGID, 518
CTABSEV, 513
CTABSSV, 513
CTABTEP, 513
CTABTEV, 513
CTABTMAX, 513
CTABTMIN, 513
CTABTSP, 513
CTABTSV, 513
CTABUNLOCK, 510
CTOL, 491
CTRANS, 61, 286, 293
CUT3DC, 409, 415
CUT3DCC, 419
CUT3DCCD, 419
CUT3DF, 409
CUT3DFF, 409
CUT3DFS, 409
CUTMOD, 446
Cycles
  paramétrer des cycles utilisateur, 197
Cycles SIEMENS, 699
Cycles technologiques, 626
  Cascadages, 632
  dans des actions synchrones non modales, 632
  Instructions de saut GOTOP, GOTOF, GOTOB, 633
  Paramètres par défaut avec valeurs initiales, 630
  Piloter un traitement cyclique ICYCOF, 631
  Sauts inconditionnels, 633
  Structures de contrôle IF, 632

D

Décalage angulaire/incrément angulaire des axes rotatifs, 436
Décalage des axes rotatifs, 436
Décalage d'origine
  décalage d'origine externe, 295
  PRESETON, 296
Décalage d'origine externe, 295
Décalage fin, 293
Décalage grossier, 293
Décalage normal au contour OFFN, 369
Décalage Preset, 296
Décélération à tous les angles, 273
Décélération aux angles intérieurs, 273
Déclenchement des coups de poinçon, 658
DEF, 22, 44, 626
DEFAULT, 86
DEFINE, 626
DEFINE ... AS, 201
Définition de tableau, 44
Définition du polynôme, 584
DELAYFSTOF, 468
DELAYFSTON, 468
DELDL, 395
DELDTG, 582
DELETE, 132
Démarrer/arrêter un axe, 602
Déplacement d'axe individuel, 665
Déplacement de broches, 610
Déplacement PTP cartésien, 314
Déplacements conjugués, 497, 611
  Limitation dynamique, 500
Déplacements de positionnement, 598
Descripteur d'axe par défaut, 565
DISABLE, 113
DISPLOF, 164
DISPLON, 164
Disponibilité
  en fonction du système, 5
DISPR, 476
Distance partielle, 660
Distances partielles, 660
DIV, 61
DL, 392
DO, 557
Données de cercles
  Calcul, 715
d'orientation
  Axes, 336

Interpolation, 337
Durée
Actions synchrones, 624
DV, 535

E

EAUTO, 233
Effacement de la distance restant à parcourir, 268, 582
Effacement de la distance restant à parcourir avec préparation, 582
EG
réducteur électronique, 526
EGDEF, 526
EGDEL, 532
EGOFC, 531
EGOFS, 531
EGON, 528
EGONSYN, 528
EGONSYNE, 528
Elément de contour
Exécuter, 714
ELSE, 96
ENABLE, 113
ENAT, 233
ENDFOR, 99
ENDIF, 96
ENDLOOP, 98
ENDPROC, 591
ENDWHILE, 100
ETAN, 233
État du contrat de mesure, 270
Etat du couplage, 500, 525
Etat du palpeur de mesure, 270
Etiquette, 88
ETIQUETTE_FIN, 88
Evénement déclencheur
Pour la mesure, 267
EVERY, 555
EXECSTRING, 60
EXECTAB, 714
EXECUTE, 215, 717
Exécution
Comportement de structures de contrôle, 96
EXP, 61
EXTCALL, 193
EXTERN, 180

F

F10, 215
F3, 689
FA, 535
facteur de couplage, 497
Facteur de tolérance G0, 495
FALSE, 22
FCTDEF, 404, 584
FCUB, 460
FENDNORM, 273
Fichier
Informations, 139
FIFOCTRL, 465
FILEDATE, 139
FILEINFO, 139
FILESIZE, 139
FILESTAT, 139
FILETIME, 139
FINE, 535
FINEA, 274
Fins de course logiciels, 607
FLIN, 460
FNORM, 460
FOCOF, 620
FOCON, 620
Fonction de mesure étendue, 376
Fonction d'évaluation, 586
Fonctions auxiliaires, 579, 660
Fonctions OEM, 272
FOR, 99
Formes des fraises, 413
FPO, 460
FPR, 533
Fraisage en bout, 409, 412
Fraisage en bout 3D, 329
courbure de la trajectoire par le biais de vecteurs normaux à la surface, 330
Fraisage périphérique, 410, 411
Fraisage périphérique (3D)
Avec surfaces de délimitation, 419
Fraisage périphérique 3D avec surfaces de délimitation, 419
Fraise
point auxiliaire (FH), 416
pointe (FS), 416
Frame
appel, 290
Concaténation de frames, 308
FRAME, 22
Frame de base résultant, 305, 306
Frame programmable courant, 306
Frame réglable courant, 306
Frame résultant courant, 307
Frames

Attribution, 291
Concaténation de frames, 291
Frames de base à définition globale pour NCU, 302
Frames de base courants à définition globale pour NCU, 304
Frames de base courants spécifiques à un canal, 305
Frames réglables à définition globale pour NCU, 302
Frames spécifiques à un canal, 303
Frames système courants, 304
Friction, 689
FROM, 555
FTOC, 593
FTOCOF, 404
FTOCON, 404
FXS, 620
FXST, 620
FXSW, 620

G

G05, 375
G07, 375
G40, 409
G450, 417
G451, 417
G62, 273
G621, 273
GEOAX, 672
GET, 121
GETACTTD, 432
GETD, 121
GETDNO, 431
GOTO, 83
GOTOB, 83
GOTOC, 83
GOTOF, 83
GOTOS, 82
GP, 57
Grignotage, 655, 660
Groupe d'axes à déplacements conjugués, 497
Groupe spline, 245
GUD, 22, 206

I

I1,I2, 434
ICYCOF, 631
ICYCON, 631
ID, 553
IDS, 553
IF, 83, 96
IFRAME, 281
II1,II2, 650
INDEX, 78
Indication du chemin
absolu, 103
relatif, 104
Indice de tableau, 47
INICF, 22
INIPO, 22
INIRE, 22
INIT, 103
INITIAL, 210
INITIAL_INI, 210
Initialisation
de tableaux, 44
Variables de tableaux, 617
Instruction de saut
CASE, 86
Instructions
Liste, 719
Instructions de programmation
Liste, 719
INT, 22
Interpolation d'orientation, 351
Interpolation du vecteur de rotation, 343, 350
Interpolation polynomiale, 250
Polynôme dénominateur, 254
INTERSEC, 712
Inversion de sens
-point, 647
IPOBRKA, 274
IPOENDA, 274
IPOSTOP, 535
IPTRLOCK, 473
IPTRUNLOCK, 473
ISAXIS, 669
ISD, 409, 415
ISFILE, 137
ISNUMBER, 74
ISOCALL, 190
ISVAR, 687

J

JERKLIM, 486
Jeu, 689
Jeu de paramètres servo
programmable, 278

L

L..., 178
l'à-coup,
correction, 486
l'arrêt du prétraitement des blocs,, 581
LEAD, 323
LEADOF, 520
LIFTFAST, 115
Lissage
Du tracé d'orientation, 357
Lissage du tracé d'orientation, 349, 352
LLI, 34
LLIMIT, 584
LN, 61
LOCK, 634
LOOP, 98
LUD, 22

M

M, 436
M17, 169
M30, 169
Macro, 201
Marques d'attente, 618
MASLDEF, 546
MASLDEL, 546
MASLOF, 546
MASLOFS, 546
MASLON, 546
MATCH, 78
MAXVAL, 68
MCALL, 185
MD20800, 169
MD37400, 459
MEAC, 262
MEAFRAME, 298
MEAFRAME, 298
MEAFRAME, 302
MEAS, 259
MEASA, 262
MEAW, 259
MEAWA, 262
Mémoire
Mémoire de programmes, 205
Tampon, 465
Travail, 210
Mémoire de correcteurs, 389
Mémoire de programmes, 205
Répertoires standard, 206
Types de fichiers, 206
Mémoire de travail, 210
Zones de données, 210
Mesure, 616
MINDEX, 78
MINVAL, 68
MIRROR, 281
MMC, 691
MOD, 61
MODAXVAL, 669
Mode de fonctionnement
Pour la mesure, 267
MOV, 602
MPF, 206, 689
MU, 373
MZ, 373

N

NCK, 22
NEWCONF, 128
Niveaux d'imbrication
de structures de contrôle, 95
NOC, 535
NOT, 64
NPROT, 219
NPROTDEF, 215
NUMBER, 74
Numéro D
attribuer librement, 429
Numéro de tranchant, 429
Numéro d'identification, 553
Numéros D
contrôler, 430
renommer, 431
NUT=angle, 335

O

OEMIPO1/2, 272
OFFN, 359, 362
Opérateurs logiques, 64
Opérateurs relationnels, 64
OR, 64
Organe porte-outil, 439
cinématique, 434
effacement/modification/lecture des données, 438
Orientable, 439
Organe porte-outil orientable
Variables système, 435
Organes porte-outil orientables
Numéro de l'organe porte-outil, 436

Organes porte-outil orientables, 434
ORIAXES, 333, 349
ORIC, 423
ORICONCCW, 335, 349
ORICONCW, 335, 349
ORICONIO, 335, 349
ORICONTO, 335, 349
ORICURVE, 339, 349
ORID, 423
Orientation de l'outil, 423
Orientation par rapport à la trajectoire
  Insertion de blocs intermédiaires, 352
  Rotation de l'orientation de l'outil, 349
  Rotation du vecteur d'orientation, 349
  Rotations de l'outil, 348
ORIEULER, 333, 349
ORIMKS, 331, 333
ORIPATH, 348
ORIPATHS, 348, 352
ORIPLANE, 335, 349
ORIRESET(A, B, C), 321
ORIROTA, 343
ORIROTC, 343, 349
ORIROTR, 343
ORIROTT, 343
ORIRPY, 333, 349
ORIRPY2, 333
ORIS, 423
ORISOF, 357
ORISON, 357
ORIVECT, 333, 349
ORIVIRT1, 333, 349
ORIVIRT2, 333, 349
ORIWKS, 331, 333
OS, 641
OSB, 641
OSC, 423
OSCILL, 647, 650
Oscillation
  Asynchrone, 641
  Inhiber le mouvement de pénétration, 650
  Oscillation asynchrone, 641
  Oscillation synchrone, 647
  Pénétration au point d'inversion de sens, 652
  Pénétration partielle, 650
  pilotée par des actions synchrones, 647
  Point d'inversion, 650
  Zone d'inversion, 650
Oscillation asynchrone, 641
Oscillation synchrone
  Actions synchrones, 651
  affectation des axes d'oscillation et de pénétration, 650
  arrêt au point d'inversion de sens, 652
  définir les pénétrations, 650
  Exploitation, période d'appel de l'interpolateur, 653
  Mouvement de pénétration, 652
  Pénétration dans la zone d'inversion de sens, 652
  Pénétration partielle suivante, 653
OSCTRL, 641
OSD, 423
OSE, 641
OSNSC, 641
OSOF, 423
OSP1, 641
OSP2, 641
OSS, 423
OSSE, 423
OST, 423
OST1, 641
OST2, 641
OTOL, 491
Outil
  Correction de longueur, 439
  Correction de rayon, 396
  Corrections additives, 392
  Mémoire de correcteurs, 389
  Orientation en cas de changement de frame, 441
  Orientation, lissage, 357
  Paramètres, 389
  Surveillance spécifique à la rectification, 667
OVRA, 540

P

P..., 183
Paramètre de calcul
  Numéro n, 18, 20
Paramètre de calcul (R), 18, 20
Paramètres
  Changement, 389
  Effectifs, 149
  Formels, 149
  Transfert à l'appel d'un sous-programme, 150, 180
Paramètres Call-by-Value
  Pour cycles technologiques, 630
Paramètres d'actions synchrones, 566
Paramètres R, 567
PCALL, 191
PDELAYOF, 655
PDELAYON, 655
Pénétration
  axe, 648

mouvement, 652
Permutation d'axe, 126
Conditions préalables, 124
libération de l'axe, 124
prise en charge de l'axe, 124
Réglage variable du comportement de la permutation d'axe, 125
sans arrêt du prétraitement, 125
sans synchronisation, 123
Permutation d'axes
demander et débloquer par le biais d'actions synchrones, 603
PFRAME, 281
PHI, 335, 342
PHU, 35
Pièce
Compteurs, 697
Répertoire principal, 206
Répertoires, 207
Pilotage de la puissance d'un laser, 585
PL, 233, 250
PO, 250
PO[PHI], 342, 348
PO[PHI]=(a2, a3, a4, a5), 335
PO[PSI], 342, 348
PO[PSI]=(b2, b3, b4, b5), 335
PO[THT], 342, 348
PO[XH], 342
PO[XH]=(xe, x2, x3, x4, x5), 339
PO[YH], 342
PO[YH]=(ye, y2, y3, y4, y5), 339
PO[ZH], 342
PO[ZH]=(ze, z2, z3, z4, z5), 339
Poinçonnage, 655, 660
Pointeur d'interruption automatique, 475
POLY, 250
Polynôme dénominateur, 254
POLYPATH, 250
PON, 664
PONS, 655
POS, 599
POSFS, 535
Position d'arrêt, 544
Position initiale de l'orientation d'outil ORIRESET, 322
Position minimale/maximale des axes rotatifs, 436
Positionnement d'un axe
Position de référence spécifiée, 601
Positionnement tangentiel, 453
Positions singulières, 332
POSP, 647
POSRANGE, 601
POT, 61
Premier frame de base courant du canal, 305
Premier frame de base du canal, 303
Préparation du contour
Signalisation d'erreur, 717
PREPRO, 168
préréglage des mémoires de valeurs réelles, 609
PRESETON, 296, 609
PRIO, 111, 115
PRLOC, 22
PROC, 153
Profondeur de pénétration, 415
Profondeur de pénétration (ISD), 409
Programmation de l'orientation, 334, 351
Programmation des rotations du vecteur d'orientation par le biais de THETA, 343
Programmation indirecte, 57
d'adresses, 53
des codes G, 56
Programme
Initialisation, 210
Mémoire, 207
Répétition:, 183
Saut, 86
Sauts, 83
Temps d'exécution, 692
Programme d'initialisation, 210
Programmer un axe oblique
G05, G07, 374
Protection
plages, 215
PSI, 335, 342
PTP, 376, 381
PTP avec TRANSMIT, 381
PTPG0, 381
PUD, 22
PUNCHACC, 655
PUTFTOC, 404
PUTFTOCF, 404
PW, 233

Q

QECDAT, 689
QECLRN, 689
QECLRNOF, 689
QECLRNON, 689
QECTEST, 689
QFK, 689

R

R..., 18, 20
RDISABLE, 580
Réaccostage du contour
  accostage avec un nouvel outil, 484
  point de réaccostage, 481
READ, 134
REAL, 22
REDEF, 28
réducteur électronique, 526
Référence angulaire, 542
Référence de trajectoire
  Paramétrable, 256
Refpos, 601
Régulation AC, additive, 587
Régulation AC, multiplicative, 588
Régulation d'écartement, 590
RELEASE, 121
REP, 44, 617
REPEAT, 88, 101
REPEATB, 88
Répétition de sections de programme
  avec programmation indirecte CALL, 188
REPOS, 109
REPOSA, 476
REPOSH, 476
REPOSHA, 476
REPOSL, 476
REPOSQ, 476
REPOSQA, 476
RESET, 634
RET, 170, 171
Retrait rapide du contour, 115
RINDEX, 78
RMB, 476
RME, 476
RMI, 476
RMN, 476
ROUND, 61
ROUNDUP, 144
Routine d'interruption, 109
  Affectation et démarrage, 111
  Delete, 114
  Déplacement de retrait, 117
  mettre hors tension/sous tension, 113
  Nouvelle affectation, 112
  Retrait rapide du contour, 115
  Sauvegarde des fonctions G modales, 110
  sens de déplacement programmable, 116, 117
Routines de conversion, 562

S

S1, S2, 535
Saisie et recherche de plages interdites à la recherche, 474
Saut
  au début du programme, 82
  Condition, 84
  Destination, 83
  Instruction, 83
  Repère, 84, 88
SAVE, 157
SBLOF, 158
SBLON, 158
SC, 453
SCPARA, 278
SD, 233
SD42475, 354
SD42476, 354
SD42477, 354
SD42678, 357
SD42680, 357
SD42900, 398
SD42910, 398
SD42920, 399
SD42930, 399
SD42935, 402
SD42940, 403, 449
SD42984, 447
Section de programme
  Répétition:, 88
SEFORM, 213
Segmentation automatique du déplacement, 660
Segmentation du déplacement dans le cas d'axes à interpolation, 663
Segmentation en distances partielles, 664
Sélection d'un caractère individuel, 80
SET, 44, 617
SETAL, 619, 699
SETDNO, 431
SETINT, 111
SETM, 103, 618
Simulation de valeurs pilotes, 525
SIN, 61
SON, 655, 663, 664
SONS, 655
Sous-programme, 145
  Appel avec transfert de paramètres, 180
  Appel indirect, 187
  Appel modal, 185
  Appel sans transfert de paramètres, 178
  Chemin de recherche programmable, 192
  Nom, 146

Répétition:, 183
Retour paramétrable, 171
SPATH, 256
SPF, 206, 689
SPI, 669
SPIF1, 655
SPIF2, 655
Spline
interpolation, 233
Types, 240
Spline A, 240
Spline B, 241
Spline C, 242
SPLINEPATH, 245
SPN, 660
SPOF, 655
SPOS, 535
SPP, 660
SQRT, 61
START, 103
STARTFIFO, 465
STAT, 376, 381
STOLF, 495
STOPFIFO, 465
STOPRE, 465
STOPREOF, 581
STRING, 22
STRINGIS, 683
STRINGVAR, 80
STRLEN, 78
SUBSTR, 80
Synchronisme
fin, 537
grossier, 537
Synchronisme de position, 535
SYNFCT, 586
SYNR, 22
SYNRW, 22
SYNW, 22
Système
disponibilité en fonction du, 5

T

Tableau, 44
Elément, 44
Tampon
Mémoire, 465
TAN, 61
TANG, 453
TANGDEL, 453
TANGOF, 453
TANGON, 453
TCARR, 439
TCOABS, 439
TCOFR, 439
TCOFRZ, 439
Temps d'usinage, 693
Temps restant
pour une pièce, 695
THETA, 342, 343
TILT, 323
TLIFT, 453
TMOF, 667
TMON, 667
TOFFOF, 443
TOFFOF, 596
TOFFON, 443
TOFFON, 596
Tolérance
pour G0, 495
TOLOWER, 77
Torsion, 689
TOUPPER, 77
TOWBCS, 400
TOWKCS, 400
TOWMCS, 400
TOWSTD, 400
TOWTCS, 400
TOWWCS, 400
TRAANG, 371
TRACON, 387
TRACYL, 362, 369
TRAFOOF, 386
TRAILOF, 497
TRAILON, 497
traitement de l'utilisation, 624
Transformation
Axe oblique, 371
Transformation 5 axes
programmation du vecteur de direction, 328
Transformation avec axe linéaire pivotant, 319
Transformation cinématique TRANSMIT, TRACYL et TRAANG, 314
Transformation de l'orientation TRAORI
Cinématique de machine, 312
Déplacements d'outils et déplacements d'orientation, 312
Programmation d'orientation, 321
Transformation 5/6 axes générique, 313
Variantes de programmation d'orientation, 322
Transformation de surface latérale de cylindre, 362, 363
décalage normal au contour OFFN, 369

Transformation de surfaces cylindriques, 314
Transformation polaire, 314
Transformation TRACYL, 364
Transformation TRANSMIT, 360
Transformation, 5 axes
Programmation de l'orientation de l'outil avec LEAD et TILT, 328
Transformation, cinq axes
Programmation de la courbure de trajectoire en vecteur normaux à la surface, 329
programmation en angles d'Euler, 326
programmation en angles RPY, 327
Programmation par le biais de LEAD/TILT, 323
Transformations
concaténées, 387
Position initiale d'orientation de l'outil indépendante de la cinématique, 310
Transformation d'orientation, 310
Transformation trois, quatre et cinq axes TRAORI, 310
Transformations cinématiques, 311
Transformations concaténées, 311
Transformations trois et quatre axes, 320
TRANSMIT, 359, 362, 381
TRAORI, 317, 320
TRUE, 22
TRUNC, 61
TU, 376, 381
Type de cinématique, 438
Type de cinématique M, 438
type de cinématique P :, 438
type de cinématique T :, 438
Type de couplage, 537
Types de transformation
Fonctionnement général, 309

U

U1,U2, 650
uc.com, cycles utilisateurs, 198
ULI, 34
ULIMIT, 584
UNLOCK, 634
UNTIL, 101
UPATH, 256

V

V1,V2, 434
Valeur de réglage, 394
Valeur d'usure, 394
Valeur pilote
Couplage, 613
VAR, 155
Variable
conversion de type, 71
Variable FIFO, 571
Variable frame, 279
Affectations aux instructions G G54 à G599, 285
affecter des valeurs, 286
Appel de transformations de coordonnées, 279
Décalages d'origine G54 à G599, 285
définition de nouveaux frames, 292
variable frame prédéfinie, 281, 290
Variables
Conversion de type, 73
Définies par l'utilisateur, 22
Définition, 22
Nom, 24, 28
Type, 22
Variables de temporisation, 570
Variables GUD
A action synchrone, 563
Variables mémentos, 566
Variables système, 560
VELOLIM, 487
Vue d’ensemble
Frames actifs dans le canal, 304

W

WAIT, 103
WAITC, 535
WAITE, 103
WAITENC, 681
WAITM, 103
WAITMC, 103
WHEN, 555
WHEN-DO, 651
WHENEVER, 555
WHENEVER-DO, 651
WHILE, 100
Winlimit, 601
WRITE, 129

X

xe, ye, ze, 339
XH YH ZH, 339
xi, yi, zi, 339
XOR, 64

A

α, 371