SIEMENS

SINUMERIK

SINUMERIK 840D sl/828D
작업 계획

프로그래밍 메뉴얼

해당 제품

시스템
SINUMERIK 840D sl / 840DE sl
SINUMERIK 828D

소프트웨어	버전
CNC- 소프트웨어	4.4

머리말

플렉시블 NC 프로그램 작성 1

파일 및 프로그램 관리 2

보호 영역 3

특수 동작 명령 4

좌표 변형 (프레임) 5

변환 6

공구 옵셋 7

경로 이송 동작 8

축 커플링 9

동기 동작 10

오실레이션 11

펀칭 및 니블링 12

연삭 13

추가 기능 14

사용자 선삭 프로그램 15

외부에서 싸이클 프로그래밍 16

테이블 17

부록 A

09/2011
6FC5398-2BP40-2LA0

법률상의 주의

경고사항

본 메뉴얼에는 여러분 자신의 안전과 재산 손실을 방지하기위해 여러분이 지켜야할 주의사항이 담겨있습니다 . 여러분의 안전에 관련된 주의사항은 안전 경고 심볼로 강조되어있으며 , 재산 손실에 관련된 주의사항은 안전 경고 심볼이 없습니다 .

⚠위험
피하지 않으면 사망 또는 심각한 부상을 초래할 수 있는 절박한 위험 상황을 나타냅니다 .

⚠경고
피하지 않으면 사망 또는 심각한 부상을 초래할 수 있는 잠재적인 위험 상황을 나타냅니다 .

⚠주의
피하지 않으면 경미한 부상을 입을 수 있는 잠재적인 위험 상황을 나타내는 안전 경고 심볼로서 사용됩니다 .

주의
피하지 않으면 재산 손실을 초래할 수 있는 잠재적인 위험 상황을 나타내는 안전 경고 심볼로서 사용됩니다 .

유의사항
피하지 않으면 원하지 않는 결과나 상태를 초래할 수 있는 잠재적인 상황을 나타내는 안전 경고 심볼로서 사용됩니다 .

여러 위험 수준이 적용될 때에는 , 항상 가장 높은 레벨 (낮은 번호) 의 알림이 표시됩니다 . 안전 경고 심볼이 인적 손실을 나타내는 경우 , 재산 손실을 경고하는 또 다른 알림이 추가될 수도 있습니다 .

자격을 가진 자

본서가 대상으로 하는 제품 / 시스템은 반드시 자격을 가진 자가 취급하는 것으로 하고 , 각 조작 내용에 관련하는 문서, 특히 안전상의 주의 및 경고가 준수되지 않으면 안됩니다 . 자격을 가진 자란 훈련 내용 및 경험을 토대로 하면서 해당 제품 / 시스템의 취급에 동반하는 위험성을 인식하고 , 발생할 수 있는 위해를 사전에 회피할 수 있는 자를 가리킵니다 .

시멘스 제품의 올바른 사용을 위해

다음에 주의하십시오 :

⚠경고
시멘스 제품은 카탈로그 및 부속의 기술 설명서의 지시에 따라 사용해 주십시오 . 타사의 제품 또는 부품과 함께 사용하는 것은 당사의 권장 또는 허가가 있을 경우에 한합니다 . 제품의 올바르고 안전한 사용을 위해 적절한 운반 , 보관 , 조립 , 설치 , 배선 , 시동 , 조작 , 보수를 시행하고 있습니다 . 사용할 때에는 허용된 범위를 꼭 지켜 주십시오 . 부속의 기술 설명서에 기술되어있는 지시를 엄수해 주십시오 .

상표

® 표시는 Siemens AG 의 등록상표입니다 . 본 문서의 기타 표시는 특정 목적으로 제삼자가 사용하는 경우 , 지적 재산권을 해칠 수 있는 상표입니다 .

책임의 포기

저희는 기술된 하드웨어와 소프트웨어가 본 메뉴얼의 내용물과 일치하는 것을 확인했습니다 . 편차가 발생하는 것을 완전히 배제할 수는 없으므로 , 완전히 동일하다고는 보장할 수 없습니다 . 그렇지만 , 메뉴얼의 데이터는 정기적으로 검토되며 , 필요한 수정은 다음의 수정판에 반영됩니다 . 품질 개선을 위한 의견은 환영합니다 .

Siemens AG
Industry Sector
Postfach 48 48
90026 NÜRNBERG
독일

문서 부품 번호 : 6FC5398-2BP40-2LA0
℗ 01.2011

Copyright © Siemens AG2011.
기술 데이터는 변경될 수 있습니다

머리말

SINUMERIK 매뉴얼

SINUMERIK 매뉴얼은 다음과 같이 구성되어 있습니다 .

- 일반 매뉴얼
- 사용자 매뉴얼
- 장비 제조업체 / 서비스 매뉴얼

추가 정보

www.siemens.com/motioncontrol/docu 웹사이트에서 다음 항목에 대한 정보를 참고할 수 있습니다 .

- 매뉴얼 주문 / 매뉴얼 개요
- 추가 매뉴얼을 다운로드할 수 있는 링크
- 온라인 매뉴얼 사용 (매뉴얼 / 정보 검색)

기술 매뉴얼에 대한 문의점 또는 개선사항이 있거나 오탈자 발견시 다음 주소로 알려 주십시오 .

docu.motioncontrol@siemens.com

My Documentation Manager (MDM)

다음 링크는 Siemens 의 컨텐츠를 기반으로 OEM 별 기계 매뉴얼을 개별적으로 저장하고 관리할 수 있는 정보를 제공합니다 .

www.siemens.com/mdm

교육

다양한 교육 과정 정보는 다음을 참조하십시오 .

- www.siemens.com/sitrain

  SITRAIN - 자동화 기술 관련 제품 , 시스템 및 솔루션에 대한 교육
- www.siemens.com/sinutrain

  SinuTrain - SINUMERIK 교육 소프트웨어

FAQ

자주 묻는 질문 (FAQ) 은 제품 지원의 Service&Support 페이지를 참조하십시오 (http://support.automation.siemens.com).

SINUMERIK

SINUMERIK 에 대한 정보는 다음 링크를 참조하십시오 .

www.siemens.com/sinumerik

대상 그룹

이 문서의 대상 독자는 다음과 같습니다 .

- 프로그래머
- 프로젝트 엔지니어

이점

대상 그룹은 본 프로그래밍 매뉴얼을 사용하여 프로그램 작성 및 수정 그리고 소프트웨어 사용자 인터페이스를 사용할 수 있습니다 .

표준 범위

이 프로그래밍 가이드에는 표준 기능이 지원하는 기능성이 설명되어 있습니다 . 장비 제조회사에서 설정한 기능이나 변경 사항은 해당 장비 제조회사의 매뉴얼을 참조하십시오 .

이 매뉴얼에 수록되지 않은 그 밖의 기능을 제어 시스템에서 지원할 수도 있습니다 . 하지만 이 조항이 새로운 시스템이나 서비스 수행 시 이러한 기능을 제공해야 한다는 것을 의미하지는 않습니다 .

간결함을 위해 본 매뉴얼에서는 모든 유형의 제품에 대한 상세 정보를 제공하지 않으며, 설치, 운전 또는 유지 보수의 경우를 모두 설명하지는 않습니다 .

기술 지원

다른 국가의 고객 지원 전화 번호는 다음 인터넷 사이트에서 확인할 수 있습니다 . http://www.siemens.com/automation/service&support

구조 및 내용 정보

" 기본편 " 및 " 고급편 " 프로그래밍 매뉴얼

NC 프로그래밍에 대한 설명은 다음 두 가지 매뉴얼로 나뉘어져 있습니다 .

1. 기본 사항

   프로그래밍 매뉴얼 " 기본편 " 은 드릴링 , 밀링 및 선삭 작업에 대해 적합한 전문 지식을 갖춘 숙련된 조작자용으로 제작되었습니다 . 또한 DIN 66025 에 따라 정의된 명령 및 명령문을 설명하기 위해 간단한 프로그래밍 예제가 사용되었습니다 .

2. 작업 계획

   프로그래밍 매뉴얼 " 고급편 " 은 기본 프로그래밍에 대한 깊이있는 지식이 있는 사람을 대상으로 합니다 . SINUMERIK 시스템에서 특수 프로그래밍 언어를 사용하여 자유 형상 표면 , 채널 좌표계 등의 여러 공작물을 쉽게 프로그램할 수 있으며 또한 프로그래밍을 하기 위한 복잡한 조작을 단순화 할 수 있습니다 .

설명된 NC 언어 요소의 가용성

이 매뉴얼에 수록된 전체 NC 언어 요소는 SINUMERIK 840D sl 에서 사용할 수 있습니다 . SINUMERIK 828D에서 사용할 수 있는 항목들은 ""의 "작업 SINUMERIK 828D에서 가용성 [쪽 860]" 테이블에서 확인할 수 있습니다 .

목차

머리말 3

1 플렉시블 NC 프로그램 작성 17

1.1 변수 17
1.1.1 변수에 대한 일반 정보 17
1.1.2 시스템 변수 18
1.1.3 사전 정의된 사용자 변수 : 산술 파라미터 (R) 21
1.1.4 사전 정의된 사용자 변수 : 링크 변수 23
1.1.5 사용자 변수 정의 (DEF) 25
1.1.6 시스템 변수 , 사용자 변수 및 NC 언어 명령어의 재 정의 (REDEF) 31
1.1.7 속성 : 초기화 값 : 34
1.1.8 속성 : 한계값 (LLI, ULI) 37
1.1.9 속성 : 물리 단위 (PHU) 39
1.1.10 속성 : 사용 권한 (APR, APW, APRP, APWP, APRB, APWB) 41
1.1.11 정의 가능 속성 및 재정의 가능 속성 개요 46
1.1.12 배열 변수 정의 및 초기화 (DEF, SET, REP): 47
1.1.13 배열 변수 정의 및 초기화 (DEF, SET, REP): 추가 정보 52
1.1.14 데이터 유형 55

1.2 간접 프로그램 56
1.2.1 간접 프로그램 주소 56
1.2.2 G 코드 간접 프로그램 59
1.2.3 위치 특성 간접 프로그램 (BP) 60
1.2.4 가공 프로그램 라인 간접 프로그램 (EXECSTRING) 63

1.3 산술 함수 64

1.4 비교 및 논리 연산자 67

1.5 비교 오류에 대한 정밀도 수정 (TRUNC) 69

1.6 변수 최소 , 최대 및 범위 (MINVAL, MAXVAL 및 BOUND) 71

1.7 연산 우선 순위 73

1.8 가능한 유형 변환 74

1.9 문자열 연산 75
1.9.1 STRING 으로 유형 변환 (AXSTRING) 76
1.9.2 STRING 에서 유형 변환 (NUMBER, ISNUMBER, AXNAME) 77
1.9.3 문자열 연결 (<<) 78
1.9.4 소문자 / 대문자로 변환 (TOLOWER, TOUPPER) 79
1.9.5 문자열 길이 결정 (STRLEN) 80
1.9.6 문자열에서 문자 / 문자열 검색 (INDEX, RINDEX, MINDEX, MATCH) 81
1.9.7 하위 문자열 선택 (SUBSTR) 82
1.9.8 단일 문자 선택 (STRINGVAR, STRINGFELD) 83
1.9.9 문자열 포맷 (SPRINT) 84

1.10 프로그램 점프 및 분기 92
1.10.1 프로그램 시작으로 점프 (GOTOS) 92
1.10.2 점프 마커로 프로그램 점프 (GOTOB, GOTOF, GOTO, GOTOC) 93
1.10.3 프로그램 분기 (CASE ... OF ... DEFAULT ...) 96

1.11 프로그램 섹션 반복 (REPEAT, REPEATB, ENDLABEL, P) 98

1.12 점검 구조 105
1.12.1 대체 블록을 가진 프로그램 루프 (IF, ELSE, ENDIF) 106
1.12.2 연속 프로그램 루프 (LOOP, ENDLOOP) 108
1.12.3 카운트 루프 (FOR ... TO ..., ENDFOR) 109
1.12.4 루프 시작 시 조건을 가진 프로그램 루프 (WHILE, ENDWHILE) 111
1.12.5 루프 끝에서 조건을 가진 프로그램 루프 (REPEAT, UNTIL) 112
1.12.6 중첩된 점검 구조를 가진 프로그램 예 113

1.13 프로그램 코디네이션 (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) 114

1.14 인터럽트 루틴 (ASUB) 119
1.14.1 인터럽트 루틴 기능 119
1.14.2 인터럽트 루틴 생성 120
1.14.3 인터럽트 루틴 지정 및 시작 (SETINT, PRIO, BLSYNC) 121
1.14.4 인터럽트 루틴 할당 비활성화 / 다시 활성화 (DISABLE, ENABLE) 123
1.14.5 인터럽트 루틴 할당 삭제 (CLRINT) 124
1.14.6 형상으로부터 빠른 후퇴 (SETINT LIFTFAST, ALF) 125
1.14.7 형상으로부터 빠른 후퇴에 대한 이송 방향 127
1.14.8 인터럽트 루틴의 이동 순서 130

1.15 축 교체 , 스핀들 교체 (RELEASE, GET, GETD) 131

1.16 다른 채널로 축 이송 (AXTOCHAN) 135

1.17 가공 데이터 활성화 (NEWCONF) 137

1.18 파일 쓰기 (WRITE) 138

1.19 파일 삭제 (DELETE) 143

1.20 파일의 행 읽기 (READ) 145

1.21 파일의 존재 여부 점검 (ISFILE) 149

1.22 파일 정보 읽기 (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) 151

1.23 배열을 이용한 체크섬 계산 (CHECKSUM) 154

1.24 반올림 (ROUNDUP) 156

1.25 서브프로그램 방법 157
1.25.1 일반 정보 157
1.25.1.1 서브프로그램 157
1.25.1.2 서브프로그램 이름 158
1.25.1.3 서브프로그램 중첩 159
1.25.1.4 검색 경로 160
1.25.1.5 유형 파라미터 및 실제 파라미터 160
1.25.1.6 파라미터 전송 161
1.25.2 서브프로그램 정의 163
1.25.2.1 파라미터 전송이 없는 서브프로그램 163
1.25.2.2 값별 호출 파라미터 전송을 포함한 서브프로그램 (PROC) 164
1.25.2.3 기준별 호출 파라미터 전송을 포함한 서브프로그램 (PROC, VAR) 165
1.25.2.4 모달 G 코드 저장 (SAVE) 167
1.25.2.5 싱글 블록 실행 억제 (SBLOF, SBLON) 168
1.25.2.6 현재 블록 디스플레이 억제 (DISPLOF, DISPLON, ACTBLOCNO) 174
1.25.2.7 준비 과정에서 서브프로그램 식별 (PREPRO) 177
1.25.2.8 서브프로그램 복귀 M17 178

1.25.2.9 RET 서브프로그램 복귀 ... 179
1.25.2.10 파라미터 지정 가능한 서브프로그램 리턴 점프 (RET ...) ... 180
1.25.3 서브프로그램 호출 ... 186
1.25.3.1 파라미터 전송 없는 서브프로그램 호출 ... 186
1.25.3.2 파라미터 전송을 포함한 서브프로그램 호출 (EXTERN) ... 188
1.25.3.3 프로그램 반복 횟수 (P) ... 190
1.25.3.4 모달 서브프로그램 호출 (MCALL) ... 192
1.25.3.5 간접 서브프로그램 호출 (CALL) ... 194
1.25.3.6 호출 프로그램 일부를 지정하여 서브프로그램 간접 호출 (CALL BLOCK ... TO ...) ... 195
1.25.3.7 ISO 언어로 프로그래밍된 프로그램의 간접 호출 (ISOCALL) ... 196
1.25.3.8 경로 지정 및 파라미터를 통한 서브루틴 호출 (PCALL) ... 198
1.25.3.9 서브프로그램 호출에 대한 확장 검색 경로 (CALLPATH) ... 199
1.25.3.10 외부 서브루틴 실행 (EXTCALL) ... 201
1.25.4 싸이클 ... 205
1.25.4.1 사용자 싸이클 파라미터화 ... 205

1.26 매크로 방법 (DEFINE ... AS) ... 209

2 파일 및 프로그램 관리 ... 213

2.1 프로그램 메모리 ... 213

2.2 작업 메모리 (CHANDATA, COMPLETE, INITIAL) ... 218

2.3 단계 편집기에서 구조화 명령 (SEFORM) ... 221

3 보호 영역 ... 223

3.1 보호 영역의 정의 (CPROTDEF, NPROTDEF) ... 223

3.2 보호 영역 활성화 / 비활성화 (CPROT, NPROT) ... 227

3.3 보호 영역 위반 , 작업 영역 제한 및 소프트웨어 제한 점검 (CALCPOSI) ... 231

4 특수 동작 명령 ... 239

4.1 코드 위치 복귀 (CAC, CIC, CDC, CACP, CACN) ... 239

4.2 스플라인 보간 (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) ... 240

4.3 스플라인 그룹화 (SPLINEPATH) ... 251

4.4 NC 블록 압축 (COMPON, COMPCURV, COMPCAD, COMPOF) ... 253

4.5 폴리노미널 보간 (POLY, POLYPATH, PO, PL) ... 256

4.6 설정 가능한 경로 기준 (SPATH, UPATH) ... 262

4.7 터치 프로브를 통한 측정 (MEAS, MEAW) ... 265

4.8 확장된 측정 기능 (MEASA, MEAWA, MEAC) (옵션) ... 268

4.9 OEM 사용자를 위한 특수 기능 (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829) ... 276

4.10 코너 감속을 사용한 피드 감소 (FENDNORM, G62, G621) ... 277

4.11 프로그래밍된 이동 종료 기준 (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) ... 278

4.12 프로그램 가능한 서보 파라미터 세트 (SCPARA) ... 281

5 좌표 변형 (프레임) ... 283

5.1 프레임 변수를 통한 좌표 변환 ... 283

5.1.1 미리 정의된 프레임 변수 ($P_BFRAME, $P_IFRAME, $P_PFRAME, $P_ACTFRAME) 285

5.2 프레임 변수 / 프레임에 할당 291
5.2.1 직접 값 할당 (축 값 , 각도 , 배율) 291
5.2.2 프레임 구성 요소 읽기 및 변경 (TR, FI, RT, SC, MI) 294
5.2.3 전체 프레임 연결 295
5.2.4 새 프레임 정의 (DEF FRAME) 297

5.3 일반 및 미세조정 옵셋 (CFINE, CTRANS) 298

5.4 외부 워크 옵셋 300

5.5 사전 설정 옵셋 (PRESETON) 301

5.6 공간의 세 개 측정 지점에서 프레임 계산 (MEAFRAME) 302

5.7 NCU 전역 프레임 306
5.7.1 채널 관련 프레임 ($P_CHBFR, $P_UBFR) 307
5.7.2 채널에서 프레임 활성화 308

6 변환 313

6.1 일반적인 변환 유형 프로그래밍 313
6.1.1 변환을 위한 방향 이동 316
6.1.2 방향 변환 개요 TRAORI 319

6.2 3 축 , 4 축 및 5 축 변환 (TRAORI) 321
6.2.1 유니버셜 공구 헤드의 일반적인 관계 321
6.2.2 3 축 , 4 축 및 5 축 변환 (TRAORI) 324
6.2.3 오리엔테이션 프로그래밍 변수 및 초기 설정 (ORIRESET) 326
6.2.4 공구 방향 프로그램 (A..., B..., C..., LEAD, TILT) 327
6.2.5 페이스 밀링 (3D 밀링 A4, B4, C4, A5, B5, C5) 335
6.2.6 방향 축 기준 (ORIWKS, ORIMKS) 336
6.2.7 오리엔테이션 축 프로그래밍 (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) 338
6.2.8 테이퍼의 주변 표면을 따라 오리엔테이션 프로그래밍 (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) 340
6.2.9 두 개의 접점 방향 지정 (ORICURVE, PO[XH]=, PO[YH]=, PO[ZH]=) 344

6.3 방향 다항식 (PO[angle], PO[coordinate]) 346

6.4 공구 오리엔테이션 회전 (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) 348

6.5 경로 기준 방향 351
6.5.1 경로 기준 방향 유형 351
6.5.2 경로 기준 공구 방향 회전 (ORIPATH, ORIPATHS, 회전 각도) 353
6.5.3 경로 기준으로 공구 회전 보간 (ORIROTC, THETA) 354
6.5.4 방향 특성 유연화 (ORIPATHS A8=, B8=, C8=) 356

6.6 방향 압축 (COMPON, COMPCURV, COMPCAD) 358

6.7 오리엔테이션 특성 스무딩 (ORISON, ORISOF) 361

6.8 운동학적 변형 363
6.8.1 선삭 가공된 부품에 대한 밀링 (TRANSMIT) 363
6.8.2 원통 표면 변환 (TRACYL) 367
6.8.3 경사 축 (TRAANG) 375
6.8.4 경사 축 프로그래밍 (G05, G07) 378

6.9 직교 PTP 이송 380

6.9.1 TRANSMIT 에 대한 PTP 385
6.10 변환 선택 시 제한사항 389
6.11 변환 선택 취소 (TRAFOOF) 390
6.12 연결된 변환 (TRACON, TRAFOOF) 391
7 공구 옵셋 393
7.1 옵셋 메모리 393
7.2 추가 옵셋 396
7.2.1 추가 옵셋 선택 (DL) 396
7.2.2 마모값 및 설정값을 지정 ($TC_SCPxy[t,d], $TC_ECPxy[t,d]) 398
7.2.3 추가 옵셋 삭제 (DELDL) 399
7.3 공구 옵셋 특수 처리 400
7.3.1 공구 길이 미러링 402
7.3.2 마모 기호 평가 403
7.3.3 활성화된 가공 작업의 좌표계(TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) 404
7.3.4 공구 길이 및 평면 변경 407
7.4 온라인 공구 옵셋 (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) 408
7.5 3D 공구 옵셋 활성 (CUT3DC..., CUT3DF...) 413
7.5.1 3D 공구 옵셋 활성화 (CUT3DC, CUT3DF, CUT3DFS, CUT3DFF, ISD) 413
7.5.2 3D 공구 옵셋 주변 밀링 , 페이스 밀링 415
7.5.3 3D 공구 옵셋 페이스 밀링을 위한 공구 형태 및 공구 데이터 417
7.5.4 3D 공구 옵셋 경로 , 경로 곡률 , 삽입 깊이 (CUT3DC, ISD) 의 보정 419
7.5.5 3D 공구 옵셋 내부 / 외부 코너 및 교차 절차 (G450/G451) 421
7.5.6 3D 공구 옵셋 한계 표면을 사용한 3D 원주 밀링 422
7.5.7 3D 공구 옵셋 한계 표면 고려 (CUT3DCC, CUT3DCCD) 423
7.6 공구 오리엔테이션 (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) 427
7.7 D 번호 , 절삭날 번호의 자유 할당 433
7.7.1 D 번호 , 절삭날 번호의 자유 할당 (CE 어드레스) 433
7.7.2 D 번호의 자유 할당 : D 번호 확인 (CHKDNO) 433
7.7.3 D 번호의 자유 할당 : D 번호 이름 변경 (GETDNO, SETDNO) 434
7.7.4 D 번호의 자유 할당 : 지정된 D 번호의 T 번호를 결정 (GETACTTD) 435
7.7.5 D 번호의 자유 할당 : 무효화된 D 번호 (DZERO) 436
7.8 공구 홀더 구조 437
7.9 방향 조정 가능한 공구 홀더에 대한 공구 길이 보정 (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) 443
7.10 온라인 공구 길이 보정 (TOFFON, TOFFOF) 446
7.11 회전 가능한 공구에 대한 절삭 데이터 수정 (CUTMOD) 449
8 경로 이송 동작 455
8.1 접선 제어 (TANG, TANGON, TANGOF, TLIFT, TANGDEL) 455
8.2 이송 속도 응답 (FNORM, FLIN, FCUB, FPO) 462
8.3 전처리 메모리를 사용한 프로그램 순서 (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE) ... 468
8.4 조건에 따라 중단 가능한 프로그램 섹션 (DELAYFSTON, DELAYFSTOF) 471

8.5 SERUPRO (IPTRLOCK, IPTRUNLOCK) 에 대한 방지 프로그램 위치 476

8.6 형상에 대해 위치 변경 (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) 478

8.7 모션 제어에 영향 487
8.7.1 백분율 저크 수정 (JERKLIM) 487
8.7.2 백분율 속도 수정 (VELOLIM) 488
8.7.3 JERKLIM 및 VELOLIM 에 대한 프로그램의 예 491

8.8 프로그램 가능한 형상 / 오리엔테이션 공차 (CTOL, OTOL, ATOL) 492

8.9 G0 모션 공차 (STOLF) 495

9 축 커플링 497

9.1 결합 작동 (TRAILON, TRAILOF) 497

9.2 커브 테이블 (CTAB) 501
9.2.1 커브 테이블 정의 (CTABDEF, CATBEND) 502
9.2.2 커브 테이블의 존재 검사 (CTABEXISTS) 508
9.2.3 커브 테이블 삭제 (CTABDEL) 509
9.2.4 삭제 및 덮어쓰기 방지를 위한 커브 테이블 잠금 (CTABLOCK, CTABUNLOCK) 510
9.2.5 커브 테이블 : 테이블 속성 결정 (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD) .. 511
9.2.6 커브 테이블 값 읽기 (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) 513
9.2.7 커브 테이블 : 리소스 사용 확인 (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) 519

9.3 축 관련 리딩 값 결합 (LEADON, LEADOF) 521

9.4 전자 기어 (EG) 527
9.4.1 전자 기어 정의 (EGDEF) 527
9.4.2 전자 기어박스 활성화 (EGON, EGONSYN, EGONSYNE) 529
9.4.3 전자 기어박스 활성화 (EGOFS, EGOFC) 532
9.4.4 전자 기어 정의 삭제 (EGDEL) 533
9.4.5 회전 이송 속도 (G95)/ 전자 기어 (FPR) 533

9.5 동기 스핀들 534
9.5.1 동기식 스핀들 : 프로그래밍 (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) 535

9.6 마스터 / 슬레이브 그룹 (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) 546

10 동기 동작 551

10.1 기본원리 551
10.1.1 적용 범위 및 가공 순서 (ID, IDS) 553
10.1.2 주기적으로 조건 검사 (WHEN, WHENEVER, FROM, EVERY) 555
10.1.3 동작 (DO) 557

10.2 조건 및 동작 관련 연산자 558

10.3 동기 동작에 사용되는 주 실행 변수 560
10.3.1 시스템 변수 560
10.3.2 함축적 유형 변환 562
10.3.3 GUD 변수 563
10.3.4 기본 설정 축 식별자 (NO_AXIS) 565
10.3.5 동기화 동작 마커 ($AC_MARKER[n]) 566
10.3.6 동기화 동작 파라미터 ($AC_PARAM[n]) 567

10.3.7 산술 파라미터 ($R[n]) 567
10.3.8 NC 기계 데이터 및 NC 설정 데이터 판독 및 기록 568
10.3.9 타이머 변수 ($AC_Timer[n]) 570
10.3.10 FIFO 변수 ($AC_FIFO1[n] ... $AC_FIFO10[n]) 571
10.3.11 보간자의 블록 유형에 관한 정보 ($AC_BLOCKTYPE, $AC_BLOCKTYPEINFO, $AC_SPLITBLOCK) 573

10.4 동기 동작의 동작들 576
10.4.1 동기화 동작에서 가능한 동작 576
10.4.2 보조 기능 출력 578
10.4.3 리드인 비활성 설정 (RDISABLE) 579
10.4.4 전처리 중지 취소 (STOPREOF) 580
10.4.5 이동할 거리 삭제 (DELDTG) 581
10.4.6 폴리노미널 정의 (FCTDEF) 582
10.4.7 동기화 기능 (SYNFCT) 585
10.4.8 폐 - 루프 공차 센서의 제한 교정 ($AA_OFF_MODE) 588
10.4.9 온라인 공구 옵셋 (FTOC) 591
10.4.10 온라인 공구 길이 보정 ($AA_TOFF) 594
10.4.11 이동 위치 지정 596
10.4.12 축 위치 지정 (POS) 597
10.4.13 지정된 참조 범위에서 위치 지정 (POSRANGE) 599
10.4.14 축 시작 / 정지 (MOV) 600
10.4.15 축 대체 (RELEASE, GET) 601
10.4.16 축 관련 이송 (FA) 605
10.4.17 소프트웨어 제한 스위치 605
10.4.18 축 좌표 606
10.4.19 실제값 설정 (PRESETON) 607
10.4.20 축 컨테이너 회전에 대한 인에이블 취소 (AXCTSWEC) 608
10.4.21 스핀들 이동 611
10.4.22 결합 동작 (TRAILON, TRAILOF) 612
10.4.23 리딩 값 결합 (LEADON, LEADOF) 614
10.4.24 측정 (MEAWA, MEAC) 616
10.4.25 배열 변수 초기화 (SET, REP) 617
10.4.26 대기 마커 설정 / 삭제 (SETM, CLEARM) 618
10.4.27 고장 응답 (SETAL) 619
10.4.28 고정 정지점으로 이동 (FXS, FXST, FXSW, FOCON, FOCOF) 620
10.4.29 동기화 동작에서 경로 접선 산정 623
10.4.30 현재 오버라이드 산정 623
10.4.31 동기화 동작 사용 점검 624

10.5 공정 싸이클 626
10.5.1 문맥 변수 ($P_TECCYCLE) 629
10.5.2 값에 따른 호출 파라미터 630
10.5.3 기본 파라미터 초기화 630
10.5.4 공정 싸이클을 제어 처리 (ICYCOF, ICYCON) 631
10.5.5 계단식 공정 싸이클 632
10.5.6 비모달식 동기화 동작의 공정 싸이클 632
10.5.7 점검 구조 (IF) 632
10.5.8 점프 명령 (GOTO, GOTOF, GOTOB) 633
10.5.9 해제 , 인에이블 , 리셋 (LOCK, UNLOCK, RESET) 633

10.6 동기화 동작 삭제 (CANCEL) 635

10.7 특정 동작 상태에서의 제어 동작 636

11 오실레이션 639

11.1 비동기 오실레이션 (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) 639

11.2 동기 동작에 의한 오실레이션 제어 (OSCILL) 645

12 펀칭 및 니블링 653

12.1 활성 , 비활성 653
12.1.1 펀칭 및 니블링 활성화 / 비활성화 (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) 653

12.2 자동 경로 분할 658
12.2.1 경로 축을 대상으로 경로 분할 661
12.2.2 단일 축을 대상으로 경로 분할 663

13 연삭 665

13.1 가공 프로그램에서 연삭 관련 도구 모니터링 (TMON, TMOF) 665

14 추가 기능 667

14.1 축 기능 (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) 667

14.2 대체 가능한 기하 축 (GEOAX) 670

14.3 축 컨테이너 (AXCTSWE, AXCTSWED, AXCTSWEC) 675

14.4 유효 축 위치 대기 (WAITENC) 681

14.5 NC 언어 (STRINGIS) 의 범위 확인 682

14.6 함수 호출 ISVAR 과 판독 기계 데이터 배열 색인 686

14.7 보정 특성 배우기 (QECLRNON, QECLRNOF) 688

14.8 가공 프로그램 (MMC) 에서 창 양방향 호출 690

14.9 프로그램 실행 시간 / 가공 카운터 691
14.9.1 프로그램 실행 시간 / 가공 카운터 (전체 보기) 691
14.9.2 프로그램 실행 시간 691
14.9.3 공작물 카운터 695

14.10 외부 장치 / 파일로 출력 (EXTOPEN, WRITE, EXTCLOSE) 696

14.11 알람 (SETAL) 706

14.12 드라이브 통합 확장 정지 및 후퇴 (ESR) 708
14.12.1 드라이브 통합 정지 (ESRS) 설정 708
14.12.2 드라이브 통합 후퇴 (ESRS) 설정 709

15 사용자 선삭 프로그램 711

15.1 스톡 제거 기능 지원 711

15.2 형상 테이블 생성 (CONTPRON) 712

15.3 코드화 형상 테이블의 생성 (CONTDCON) 718

15.4 형상 요소 2 개 사이의 교차점을 결정 (INTERSEC) 722

15.5 테이블의 형상 요소를 블록별로 실행 (EXECTAB) 724

15.6 원 데이터 계산 (CALCDAT) 725

15.7 형상 준비 종료 (EXECUTE) 727

16 외부에서 싸이클 프로그래밍 729

16.1 공정 싸이클 729
16.1.1 소개 729
16.1.2 드릴링 , 센터링 - CYCLE81 731
16.1.3 드릴링 , 카운터보링 - CYCLE82 732
16.1.4 리밍 - CYCLE85 733
16.1.5 심공 드릴링 - CYCLE83 734
16.1.6 보링 - CYCLE86 736
16.1.7 동기 태핑 - CYCLE84 737
16.1.8 태핑 척을 사용한 태핑 - CYCLE840 740
16.1.9 나사 밀링 - CYCLE78 742
16.1.10 자유 프로그래밍 위치 - CYCLE802 744
16.1.11 여러 구멍으로 이루어진 행 – HOLES1 746
16.1.12 그리드 또는 프레임 - CYCLE801 747
16.1.13 여러 개의 구멍으로 형성된 원 - HOLES2 748
16.1.14 페이스 밀링 - CYCLE61 749
16.1.15 사각 포켓 밀링 - POCKET3 751
16.1.16 원호 포켓 밀링 – POCKET4 754
16.1.17 사각 스피곳 밀링 - CYCLE76 756
16.1.18 원주 밀링 - CYCLE77 758
16.1.19 다각형 - CYCLE79 760
16.1.20 일반 슬롯 - SLOT1 762
16.1.21 원호 슬롯 - SLOT2 765
16.1.22 밀링 개방형 슬롯 - CYCLE899 767
16.1.23 연장된 홀 - LONGHOLE 769
16.1.24 나사 밀링 - CYCLE70 771
16.1.25 조각 싸이클 - CYCLE60 773
16.1.26 형상 호출 - CYCLE62 776
16.1.27 경로 밀링 - CYCLE72 777
16.1.28 형상 포켓 사전 드릴링 - CYCLE64 780
16.1.29 형상 포켓 밀링 - CYCLE63 782
16.1.30 선삭 - CYCLE951 784
16.1.31 홈 - CYCLE930 787
16.1.32 언더컷 형상 - CYCLE940 790
16.1.33 나사 선삭 - CYCLE99 793
16.1.34 나사 체인 - CYCLE98 796
16.1.35 절단 - CYCLE92 799
16.1.36 형상 홈 가공 - CYCLE952 801
16.1.37 스위블 - CYCLE800 805
16.1.38 고속 가공 설정 - CYCLE832 808
16.1.39 고속 절삭 (HSC) - CYCLE_HSC 809

17 테이블 811

17.1 작업 811

17.2 작업 SINUMERIK 828D 에서 가용성 860

17.3 현재 HMI 에서 설정된 언어 882

A 부록 883

A.1 약어 목록 883

A.2 개요 888

용어 색인.....891

플렉시블 NC 프로그램 작성 1

1.1 변수

1.1.1 변수에 대한 일반 정보

변수를 사용하면 , 특히 산술 함수 및 점검 구조와 연계해 사용하면 가공 프로그램 및 사이클을 매우 유연하게 설정할 수 있습니다 . 이런 목적에 맞게 시스템은 세 가지 유형의 변수를 제공합니다 .

- 시스템 변수

  시스템 변수는 사전 정의된 고정적 의미를 갖는 변수입니다 . 시스템 변수는 시스템에서 정의해 사용자가 사용할 수 있게 만든 변수입니다 . 또한 이 변수는 시스템 소프트웨어로 읽거나 쓸 수 있습니다 . 예제 : 머신 데이터

  시스템 변수의 의미는 대부분의 경우 시스템에 의해 사전 정의되며 변수의 속성도 고정되어 있습니다 . 하지만 사용자는 다시 정의하는 방식을 통해 시스템 변수의 속성을 일부 수정할 수 있습니다 . " 시스템 변수 , 사용자 변수 및 NC 언어 명령어의 재 정의 (REDEF) [쪽 31]" 를 참조하십시오 .

- 사용자 변수

  사용자 변수는 시스템에서 그 의미를 알지 못하는 변수이며 시스템이 평가하지도 않습니다 . 사용자 변수의 의미는 전적으로 사용자가 정의합니다 .

  사용자 변수는 다시 다음과 같이 나뉩니다 .

  - 사전 정의된 사용자 변수

    사전 정의된 사용자 변수란 시스템에서 이미 정의를 했고 사용자는 특정 머신 데이터를 통해 간단히 번호만 파라미터화하면 되는 변수를 말합니다 . 이 변수의 속성은 사용자가 대폭 수정할 수 있습니다 . " 시스템 변수 , 사용자 변수 및 NC 언어 명령어의 재 정의 (REDEF) [쪽 31]" 를 참조하십시오 .

  - 사용자 정의 변수

    사용자 정의 변수란 전적으로 사용자에 의해 정의되며 런타임 전에는 시스템에서 생성되지 않는 변수를 말합니다 . 이 변수의 번호 , 데이터 유형 , 가시성 및 기타 모든 속성은 전적으로 사용자에 의해 정의됩니다 .

    " 사용자 변수 정의 (DEF) [쪽 25]" 를 참조하십시오 .

도 참조

시스템 변수 시스템 변수 [쪽 18]

사전 정의된 사용자 변수 : 산술 파라미터 (R) 사전 정의된 사용자 변수 : 산술 파라미터 (R) [쪽 21]

사전 정의된 사용자 변수 : 링크 변수 사전 정의된 사용자 변수 : 링크 변수 [쪽 23]

속성 : 초기화 값 : 속성 : 초기화 값 : [쪽 34]

속성 : 한계값 (LLI, ULI) 속성 : 한계값 (LLI, ULI) [쪽 37]

속성 : 물리 단위 (PHU) 속성 : 물리 단위 (PHU) [쪽 39]

속성 : 사용 권한 (APR, APW, APRP, APWP, APRB, APWB) 속성 : 사용 권한 (APR, APW, APRP, APWP, APRB, APWB) [쪽 41]

정의 가능 속성 및 재정의 가능 속성 개요 정의 가능 속성 및 재정의 가능 속성 개요 [쪽 46]

배열 변수 정의 및 초기화 (DEF, SET, REP): 배열 변수 정의 및 초기화 (DEF, SET, REP): [쪽 47]

데이터 유형 데이터 유형 [쪽 55]

1.1.2 시스템 변수

시스템 변수란 시스템에 사전 정의되어 있고 가공 프로그램 및 싸이클의 제어 시스템 , 기계 , 처리 상태 등과 관련한 실제 파라미터 설정에 대한 액세스를 제공하는 변수를 말합니다 .

전처리 변수

전처리 변수란 전처리 컨텍스트에 따라 , 다시 말해 이 시스템 변수가 프로그래밍되어 있는 가공 프로그램 블록이 해석될 때 읽고 쓰여지는 시스템 변수를 말합니다 . 전처리 변수는 전처리 정지를 실행시키지 않습니다 .

메인 실행 변수

메인 실행 변수란 메인 실행의 컨텍스트에 따라 , 다시 말해 이 시스템 변수가 프로그래밍되어 있는 가공 프로그램 블록이 실행될 때 읽고 쓰여지는 시스템 변수를 말합니다 . 다음과 같은 메인 실행 변수가 있습니다 .

- 동기 동작에 프로그램 가능한 시스템 변수 (읽기 / 쓰기)
- 가공 프로그램에 프로그램 가능하며 전처리 정지를 실행시키는 시스템 변수 (읽기 / 쓰기)
- 가공 프로그램에 프로그램 가능하고 전처리 중에 값이 계산되지만 메인 실행 전에는 쓰여지지 않는 시스템 변수 (메인 실행 동기화 : 쓰기 전용)

접두어 시스템

시스템 변수는 구체적으로 식별이 가능해야 하기 때문에 보통 $ 부호 다음에 1~2 개의 글자와 1 개의 밑줄로 이루어진 접두어를 시스템 변수의 이름 앞에 붙입니다 .

$ + 1. 글자	의미 : 데이터 유형
전처리 중에 읽고 쓰여지는 시스템 변수	
$M	머신 데이터 [1]
$S	셋팅 데이터 , 보호 영역 [1]
$T	공구 관리 데이터
$P	프로그램된 값
$C	ISO 인벨로프 싸이클의 싸이클 변수
$O	옵션 데이터
R	R- 파라미터 (산술 파라미터) [2]
메인 실행 중 읽고 쓰여지는 시스템 변수	
$$M	머신 데이터 [1]
$$S	셋팅 데이터 [1]
$A	최신 메인 실행 데이터
$V	서보 데이터
$R	R- 파라미터 (산술 파라미터) [2]

[1] 머신 데이터 및 셋팅 데이터가 가공 프로그램 / 싸이클에서 전처리 변수로 사용되는 경우 1 개의 $ 부호가 접두어로 붙습니다 . 이 데이터가 동기 동작에서 메인 실행 변수로 사용되는 경우 2 개의 $ 부호가 접두어로 붙습니다 .

[2] R- 파라미터가 가공 프로그램 / 싸이클에서 전처리 변수로 사용되는 경우 접두어를 생략합니다 (예 : R10). R- 파라미터가 동기 동작에서 메인 실행 변수로 사용되는 경우 1 개의 $ 부호가 접두어로 붙습니다 (예 : $R10).

2. 글자	의미 : 가시성
N	NCK 전역 변수 (**N**CK)
C	채널별 변수 (**C**hannel)
A	축별 변수 (**A**xis)

일반 조건

접두어 체계의 예외

다음 변수 체계는 위에 명시한 접두어 체계와 다릅니다 .

- $TC_...: 여기서 두 번째 글자인 C 는 채널별 시스템 변수가 아닌 공구 홀더별 시스템 변수 (TC= 공구 캐리어) 를 의미합니다 .
- $P_ ...: 채널별 시스템 변수

동기 동작에서 머신 데이터와 셋팅 데이터 사용

머신 데이터와 셋팅 데이터를 동기 동작에서 사용할 때는 머신 데이터와 셋팅 데이터의 읽기 / 쓰기가 전처리 실행 또는 메인 실행 중 어디에 동기화되도록 할지를 접두어를 사용하여 정의할 수 있습니다 .

가공 중 데이터가 변경되지 않는 경우 전처리 실행과 동기적으로 읽을 수 있습니다 . 이를 위해 머신 데이터 또는 셋팅 데이터에 1 개의 $ 부호를 접두어로 붙입니다 .

프로그램 코드

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

가공 중 데이터가 변경되는 경우 메인 실행과 동기적으로 읽거나 써야 합니다 . 이를 위해 머신 데이터 또는 셋팅 데이터에 2 개의 $ 부호를 접두어로 붙입니다 .

프로그램 코드

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

주

머신 데이터 쓰기

머신 데이터 또는 셋팅 데이터 항목을 쓰기 위해서는 가공 프로그램 / 싸이클 실행 시 활성화된 사용 권한이 쓰기 액세스를 허용해야 하고 해당 데이터가 "즉시 적용" 효과를 갖도록 설정해야 합니다 .

참고 자료

모든 시스템 변수의 속성은 다음 문서에 목록으로 정리되어 있습니다 .

목록 메뉴얼 , 시스템 변수

도 참조

변수에 대한 일반 정보 변수에 대한 일반 정보 [쪽 17]

1.1.3 사전 정의된 사용자 변수 : 산술 파라미터 (R)

기능

산술 파라미터 또는 R- 파라미터는 REAL 데이터 유형의 배열을 의미하는 R 이 표시된 사전 정의된 사용자 변수입니다 . 역사적인 이유 때문에 R- 파라미터는 배열 인덱스를 표기할 수도 있고 (예 : R[10]) 배열 인덱스를 표기하지 않을 수도 (예 : R10) 있습니다 .

동기 동작을 사용하는 경우 R- 파라미터에 $ 부호를 접두어로 붙여야 합니다 (예 : $R10).

구문

전처리 변수로 사용되는 경우 :
R<n>
R[< 표현식 >]

메인 실행 변수로 사용되는 경우 :
$R<n>
$R[< 표현식 >]

의미

R:	가공 프로그램 등에서 전처리 변수로 사용됨을 표시하는 식별자	
$R:	동기 동작 등에서 메인 실행 변수로 사용됨을 표시하는 식별자	
	유형 :	REAL
	범위 :	비 - 지수 표기 ± (0.000 0001 ~ 9999 9999) **참고 :** 최대 8 개의 소수 자리 수가 허용됩니다 . 지수 표기 : ± ($1*10^{-300}$ ~ $1*10^{+300}$) **참고 :** • 표기법 : < 가수 >EX< 지수 > e.g. 8.2EX-3 • 최대 10 개의 문자가 허용됩니다 (부호와 소수점 포함).

<n>: R- 파라미터 번호

유형 : INT

범위 : 0 - MAX_INDEX

참고

MAX_INDEX 는 설정된 R- 파라미터의 개수에 의해 계산됩니다 .

MAX_INDEX = (MD28050 $MN_MM_NUM_R_PARAM) - 1

< 표현식 >: 배열 인덱스

어떤 표현식이든 그 결과를 INT 데이터 유형 (INT, REAL, BOOL, CHAR) 으로 변환할 수만 있다면 배열 인덱스로 사용할 수 있습니다 .

예제

산술 함수로 R- 파라미터 지정 및 사용 :

프로그램 코드		설명
R0=3.5678	;	전처리에 지정
R[1]=-37.3	;	전처리에 지정
R3=-7	;	전처리에 지정
$R4=-0.1EX-5	;	메인 실행에 지정 : R4 = -0.1 * 10^-5
$R[6]=1.874EX8	;	메인 실행에 지정 : R6 = 1.874 * 10^8
R7=SIN(25.3)	;	전처리에 지정
R[R2]=R10	;	R- 파라미터를 사용해 간접 어드레스 지정
R[(R1+R2)*R3]=5	;	산술 표현식을 사용해 간접 어드레스 지정
X=(R1+R2)	;	축 X 를 R1 과 R2 의 합에서 나온 위치로 이송
Z=SQRT(R1*R1+R2*R2)	;	축 Z 를 제곱근 위치로 이송 (R1^2 + R2^2)

도 참조

변수에 대한 일반 정보 변수에 대한 일반 정보 [쪽 17]

1.1.4 사전 정의된 사용자 변수 : 링크 변수

기능

링크 변수는 네트워크를 통해 서로 링크되어 있는 NCU 들이 사이클 데이터를 교환하는 데 쓰이는 "NCU-링크" 기능에 사용할 수 있습니다. 링크 변수는 링크 변수 메모리에 대한 데이터-형식별 액세스를 가능하게 합니다 . 링크 변수 메모리의 크기와 데이터 구조는 사용자 / 장비 제조업체가 시스템별로 정의합니다 .

링크 변수는 링크 통신이 설정된 경우 링크에 연관된 모든 NCU 가 가공 프로그램 및 사이클에서 읽고 쓸 수 있는 시스템 전역 사용자 변수입니다 . 전역 사용자 변수 (GUD) 와 달리 링크 변수는 동기 동작에도 사용할 수 있습니다 .

활성 NCU 링크가 없는 시스템인 경우 제어 시스템에서 로컬로 전역 사용자 변수 (GUD) 와 함께 추가적인 전역 사용자 변수로 링크 변수를 사용할 수 있습니다 .

구문

$A_DLB[< 인덱스 >]
$A_DLW[< 인덱스 >]
$A_DLD[< 인덱스 >]
$A_DLD[< 인덱스 >]

의미

$A_DLB:	BYTE 데이터 형식을 위한 링크 변수 (1 바이트)	
	데이터 유형 :	UINT
	범위 :	0 ~ 255
$A_DLW:	WORD 데이터 형식을 위한 링크 변수 (2 바이트)	
	데이터 유형 :	INT
	범위 :	-32768 ~ 32767
$A_DLD:	DWORD 데이터 형식을 위한 링크 변수 (4 바이트)	
	데이터 유형 :	INT
	범위 :	-2147483648 ~ 2147483647
$A_DLR:	REAL 데이터 형식을 위한 링크 변수 (8 바이트)	
	데이터 유형 :	REAL
	범위 :	$\pm(2{,}2*10^{-308} \ldots 1{,}8*10^{+308})$

< 인덱스 >: 바이트 단위 어드레스 인덱스 (링크 변수 메모리의 시작에서 계산)

데이터 유형 : INT

범위 : 0 - MAX_INDEX

참고

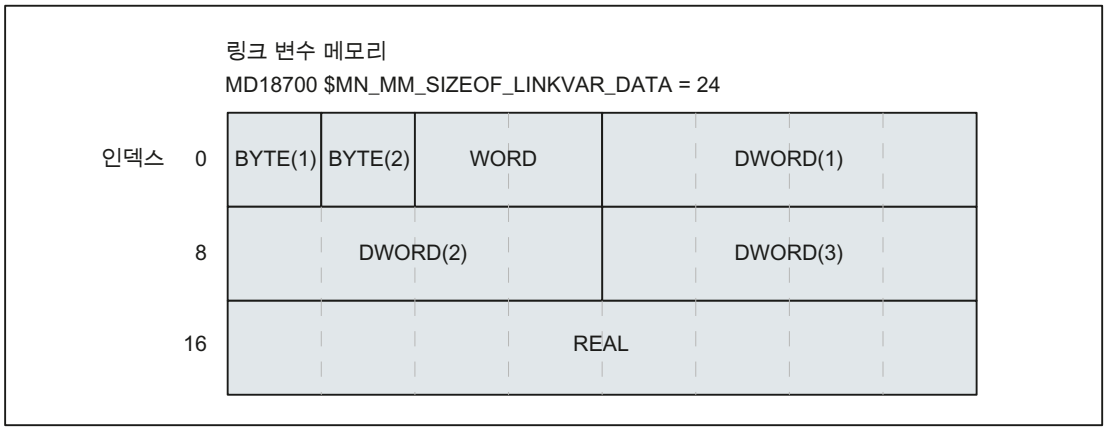
- MAX_INDEX 는 파라미터화된 링크 변수 메모리의 크기에서 계산됩니다 . MAX_INDEX = (MD18700 $MN_MM_SIZEOF_LINKVAR_DATA) - 1
- 인덱스만 프로그래밍이 가능하기 때문에 링크 변수 메모리에 지정된 바이트는 데이터 형식 한계 상에 위치합니다 . ⇒
  인덱스 = n * 바이트 . 여기서 n = 0, 1, 2, ...
  - $A_DLB[i]: i = 0, 1, 2, ...
  - $A_DLW[i]: i = 0, 2, 4, ...
  - $A_DLD[i]: i = 0, 4, 8, ...
  - $A_DLR[i]: i = 0, 8, 16, ...

예제

한 자동화 시스템에 2 개의 NCU (NCU 1 및 NCU 2) 가 있습니다 . 기계 축 AX2 는 NCU1 에 연결되어 있습니다 . AX2 는 NCU2 의 링크 축으로 취급되어 이송됩니다 .

NCU1은 주기적으로 축 AX2의 실제 현재 값 ($VA_CURR) 을 링크 변수 메모리에 기록합니다. NCU2는 링크 통신을 통해 전송되는 실제 현재 값을 주기적으로 읽고 이 값이 한계값을 초과한 경우 알람 61000 을 표시합니다 .

다음 그림은 링크 변수 메모리의 데이터 구조를 나타낸 것입니다 . 실제 현재 값은 REAL 값 형식으로 전송됩니다 .



NCU1

NCU1 은 스태틱 동기 동작의 보간 사이클에서 링크 변수 $A_DLR[16] 를 사용하여 축 AX2 의 실제 현재 값을 주기적으로 링크 변수 메모리에 기록합니다 .

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

NCU2
NCU2 는 스태틱 동기 동작의 보간 사이클 동안 링크 변수 $A_DLR[16] 를 사용하여 링크 변수 메모리에서 축 AX2 의 실제 현재 값을 주기적으로 읽습니다 . 읽은 실제 현재 값이 23.0 A 보다 큰 경우 알람 61000 이 표시됩니다 .

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

도 참조

변수에 대한 일반 정보 변수에 대한 일반 정보 [쪽 17]

1.1.5 사용자 변수 정의 (DEF)

기능

DEF 명령어는 사용자 정의 변수를 정의하고 해당 변수에 값을 지정하는 데 사용됩니다 . 시스템 변수와 구분하기 위해 이 변수를 사용자 정의 변수 또는 사용자 변수 (사용자 데이터) 라고 부릅니다 .

유효성 범위, 즉 변수가 눈에 보이는 범위를 기준으로 할 때 사용자 변수는 다음과 같은 범주로 나뉩니다 .

- 로컬 사용자 변수 (LUD)

  로컬 사용자 변수 (LUD) 는 실행 시점을 기준으로 메인 프로그램이 아닌 가공 프로그램에서 정의된 변수입니다 . 이 변수는 가공 프로그램이 호출될 때 생성되고 가공 프로그램이 종료되거나 NC 가 리셋될 때 삭제됩니다 . 로컬 사용자 변수는 해당 변수가 정의된 가공 프로그램 내에서만 액세스할 수 있습니다 .

- 프로그램 전역 사용자 변수 (PUD)

  프로그램 전역 사용자 변수 (PUD) 는 메인 프로그램으로 사용되는 가공 프로그램에서 정의된 사용자 변수입니다 . 이 변수는 가공 프로그램이 시작될 때 만들어지고 가공 프로그램이 종료되거나 NC 가 리셋될 때 삭제됩니다 . PUD 는 메인 프로그램과 메인 프로그램의 모든 하부 프로그램에서 액세스할 수 있습니다 .

- 전역 사용자 변수 (GUD)

  전역 사용자 변수 (GUD) 는 데이터 블록 (SGUD, MGUD, UGUD, GUD4~GUD9) 에서 정의된 NC 또는 채널 전역 변수이며 가동정지 및 재시작 후에도 계속 유지됩니다 . GUD 는 모든 가공 프로그램에서 액세스할 수 있습니다 .

사용자 변수를 사용하기 위해서는 먼저 정의 (읽기 / 쓰기) 를 해야 합니다 . 사용자 변수를 정의할 때 다음 규칙을 준수해야 합니다 .

- GUD 는 정의 파일 (예 : _N_DEF_DIR/_M_SGUD_DEF) 에 정의해야 합니다 .
- PUD 및 LUD 는 가공 프로그램의 정의 섹션에 정의해야 합니다 .
- 데이터는 지정된 블록에 정의해야 합니다 .
- 각 데이터 정의에 단 하나의 데이터 유형만 사용할 수 있습니다 .
- 각 데이터 정의에 데이터 유형이 동일한 변수를 여러 개 정의할 수 있습니다 .

구문

DEF < 범위> < 유형> <PP_ 정지> < 초기화 _ 시간> < 물리_ 단위> < 한계_ 값> < 액세스 _ 권한> < 이름 >[< 값 _1>,< 값 _2>,< 값 _3>]=< 초기화 _ 값 >

중요도

DEF:	사용자 변수 GUD, PUD 및 LUD 를 정의하는 명령어 .	
< 범위 >:	유효 범위 (GUD 만 해당):	
	NCK:	NC 전역 사용자 변수
	CHAN:	채널 전역 사용자 변수
< 유형 >:	데이터 유형 :	
	INT:	부호가 있는 정수
	REAL:	실수 (LONG REAL ~ IEEE)
	BOOL:	참 값 TRUE (1) / FALSE (0)
	CHAR:	ASCII 문자
	STRING[< 최대 길이 >]:	길이가 정의된 문자열
	AXIS:	축 이름 / 스핀들 이름
	FRAME:	스태틱 좌표 변환을 위한 기하 데이터
	" 데이터 유형 [쪽 55]" 를 참조하십시오 .	
<PP_ 정지 >:	전처리 정지 (GUD 에만 해당하는 선택사항)	
	SYNR:	읽는 동안 전처리 정지
	SYNW:	쓰는 동안 전처리 정지
	SYNRW:	읽거나 쓰는 동안 전처리 정지

< 초기화 _ 시간 >: 변수가 다시 초기화되는 시간 (선택사항)

INIPO: 전원 ON/OFF

INIRE: 메인 프로그램의 끝 , NC 리셋 또는 전원 ON/OFF

INICF: 새 설정 , 메인 프로그램의 끝 , NC 리셋 또는 전원 ON/OFF

PRLOC: 메인 프로그램의 끝 , 로컬 변경 후 NC 리셋 또는 전원 ON/OFF

" 속성 : 초기화 값 : [쪽 34]" 를 참조하십시오 .

< 물리 _ 단위 >: 물리 단위 (선택사항)

PHU < 단위 >:

" 속성 : 물리 단위 (PHU) [쪽 39]" 를 참조하십시오 .

< 한계값 >: 상한값 및 하한값 (선택사항)

LLI < 한계값 >: 하한값

ULI < 한계값 >: 상한값

" 속성 : 한계값 (LLI, ULI) [쪽 37]" 를 참조하십시오 .

< 사용 권한 >: 가공 프로그램 또는 OPI 를 통한 GUD 읽기 / 쓰기 사용 권한 (선택사항)

APRP < 사용 권한 >: 읽기 : 가공 프로그램

APWP < 사용 권한 >: 쓰기 : 가공 프로그램

APRB < 사용 권한 >: 읽기 : OPI

APWB < 사용 권한 >: 쓰기 : OPI

사용 권한 값 범위 : 0 ... 7

"속성: 사용 권한 (APR, APW, APRP, APWP, APRB, APWB) [쪽 41]" 를 참조하십시오 .

< 이름 >: 변수 이름

참고

- 최대 31 자
- 첫 두 문자는 글자 및 / 또는 밑줄이어야 합니다 .
- $ 부호는 시스템 변수를 위한 것이므로 사용하지 않도록 합니다 .

[< 값 _1>,
< 값 _2>,
< 값 _3>]: 1 차원부터 최대 3 차원 배열 변수의 배열 크기 지정 (선택사항)

< 초기화 _ 값 >: 초기화 값 (선택사항)

" 속성 : 초기화 값 : [쪽 34]" 를 참조하십시오 .

배열 변수의 초기화 :

" 배열 변수 정의 및 초기화 (DEF, SET, REP): [쪽 47]" 를 참조하십시오 .

예제

예제 1: 장비 제조업체를 위한 사용자 변수 정의

프로그램 코드

```
%_N_MGUD_DEF                                                    ;   GUD 블록 : 장비 제조업체
$PATH=/_N_DEF_DIR
DEF CHAN REAL PHU 24 LLI 0 ULI 10 STROM_1, STROM_2
;   설명
;   2 개의 GUD 항목 정의 : STROM_1, STROM_2
;   유효 범위 : 채널 전체
;   데이터 유형 : REAL
;   PP 정지 : 프로그램 안됨 => 디폴트 값 = PP 정지 없음
;   물리 단위 : 24 = [A]
;   한계값 : 하한값 = 0.0, 상한값 = 10.0
;   사용 권한 : 프로그램 안됨 => 디폴트 값 = 7 = 키 스위치 위치 0
;   초기화 값 : 프로그램 안됨 => 디폴트 값 = 0.0

DEF NCK REAL PHU 13 LLI 10 APWP 3 APRP 3 APWB 0 APRB 2 ZEIT_1=12, ZEIT_2=45
;   설명
;   2 개의 GUD 항목 정의 : ZEIT_1, ZEIT_2
;   유효 범위 : NCK 전체
;   데이터 유형 : REAL
;   PP 정지 : 프로그램 안됨 => 디폴트 값 = PP 정지 없음
;   물리 단위 : 13 = [s]
;   한계값 : 하한값 = 10.0, 상한값 = 프로그램 안됨 => 정의 범위 상한값
;   사용 권한 :
;       가공 프로그램 : 읽기 / 쓰기 = 3 = 최종 사용자
;       OPI: 쓰기 = 0 = Siemens, 읽기 = 3 = 최종 사용자
;   초기화 값 : ZEIT_1 = 12.0, ZEIT_2 = 45.0

DEF NCK APWP 3 APRP 3 APWB 0 APRB 3 STRING[5] GUD5_NAME = "COUNTER"
;   설명
;   1 개의 GUD 항목 정의 : GUD5_NAME
;   유효 범위 : NCK 전체
;   데이터 유형 : 문자열 ( 최대 5 자 )
;   PP 정지 : 프로그램 안됨 => 디폴트 값 = PP 정지 없음
;   물리 단위 : 프로그램 안됨 => 디폴트 값 = 0 = 물리 단위 없음
;   한계값 : 프로그램 안됨 => 정의 범위 한계 : 하한값 = 0, 상한값 = 255
;   사용 권한 :
;       가공 프로그램 : 읽기 / 쓰기 = 3 = 최종 사용자
;       OPI: 쓰기 = 0 = Siemens, 읽기 = 3 = 최종 사용자
;   초기화 값 : "COUNTER"
M30
```

예제 2: 프로그램 전역 사용자 변수 및 프로그램 로컬 사용자 변수 (PUD/LUD)

프로그램 코드	코멘트
PROC MAIN	; 메인 프로그램
DEF INT VAR1	; PUD 정의
...	
SUB2	; 서브프로그램 호출
...	
M30	

프로그램 코드	코멘트
PROC SUB2	; 서브프로그램 SUB2
DEF INT VAR2	; LUD DEFINITION
...	
IF (VAR1==1)	; PUD 읽기
VAR1=VAR1+1	; PUD 읽기 및 쓰기
VAR2=1	; LUD 쓰기
ENDIF	
SUB3	; 서브프로그램 호출
...	
M17	

프로그램 코드	코멘트
PROC SUB3	; 서브프로그램 SUB3
...	
IF (VAR1==1)	; PUD 읽기
VAR1=VAR1+1	; PUD 읽기 및 쓰기
VAR2=1	; 오류 : 알 수 없는 SUB2 의 LUD
ENDIF	
...	
M17	

예제 3: AXIS 데이터 유형의 사용자 변수 정의 및 사용

프로그램 코드	코멘트
DEF AXIS ABSCISSA	; 1. 기하 축
DEF AXIS SPINDLE	; 스핀들
...	
IF ISAXIS(1) == FALSE GOTOF CONTINUE	
ABSCISSA = $P_AXN1	
CONTINUE:	
...	
SPINDLE=(S1)	1. 스핀들
OVRA[SPINDLE]=80	; 스핀들 오버라이드 = 80%
SPINDLE=(S3)	3. 스핀들

일반 조건

전역 사용자 변수 (GUD)

전역 사용자 변수 정의 시 다음의 머신 데이터를 고려해야 합니다 .

번호	식별자 : $MN_	중요도
11140	GUD_AREA_ SAVE_TAB	GUD 블록을 위한 추가 저장
18118 [1]	MM_NUM_GUD_MODULES	활성 파일 시스템 내 GUD 파일의 개수
18120 [1]	MM_NUM_GUD_NAMES_NCK	전역 GUD 이름의 개수
18130 [1]	MM_NUM_GUD_NAMES_CHAN	채널별 GUD 이름 수
18140 [1]	MM_NUM_GUD_NAMES_AXIS	축별 GUD 이름 수
18150 [1]	MM_GUD_VALUES_MEM	전역 GUD 값의 메모리 위치
18660 [1]	MM_NUM_SYNACT_GUD_REAL	REAL 데이터 유형의 설정 가능한 GUD 개수
18661 [1]	MM_NUM_SYNACT_GUD_INT	INT 데이터 유형의 설정 가능한 GUD 개수
18662 [1]	MM_NUM_SYNACT_GUD_BOOL	BOOL 데이터 유형의 설정 가능한 GUD 개수
18663 [1]	MM_NUM_SYNACT_GUD_AXIS	AXIS 데이터 유형의 설정 가능한 GUD 개수
18664 [1]	MM_NUM_SYNACT_GUD_CHAR	CHAR 데이터 유형의 설정 가능한 GUD 개수
18665 [1]	MM_NUM_SYNACT_GUD_STRING	STRING 데이터 유형의 설정 가능한 GUD 개수

[1] SINUMERIK 828D 의 경우 MD 만 읽을 수 있습니다 !

프로그램 전역 사용자 변수 (PUD)

유의사항
프로그램 로컬 사용자 변수 (PUD) 의 표시 머신 데이터가 다음과 같이 설정된 경우 메인 프로그램에 정의된 프로그램 로컬 사용자 변수 (PUD) 는 서브프로그램에서만 볼 수 있습니다 . MD11120 $MN_LUD_EXTENDED_SCOPE = 1 MD11120 = 0 이면 메인 프로그램에 정의된 프로그램 로컬 사용자 변수 (PUD) 는 서브프로그램에서만 볼 수 있습니다 .

AXIS 데이터 유형의 NCK 전역 사용자 변수를 여러 채널에 사용

데이터 블록에 정의할 때 축 이름으로 초기화한 AXIS 데이터 유형의 NCK 전역 사용자 변수는 해당 축이 다른 여러 NC 채널에서 동일한 채널 축 번호를 갖고 있는 경우에만 이 NC 채널에서 사용할 수 있습니다 .

그렇지 않은 경우 해당 변수는 가공 프로그램의 시작에서 로드되거나 다음 예의 경우처럼 AXNAME(...) 기능 (" 참조 ") 을 사용해야 합니다 .

프로그램 코드	코멘트
DEF NCK STRING[5] ACHSE="X"	; 데이터 블록에 정의
N100 AX[AXNAME(ACHSE)]=111 G00	; 가공 프로그램에 사용

도 참조

변수에 대한 일반 정보 변수에 대한 일반 정보 [쪽 17]

1.1.6 시스템 변수 , 사용자 변수 및 NC 언어 명령어의 재 정의 (REDEF)

기능

REDEF 명령을 사용해 시스템 변수 , 사용자 변수 및 NC 언어 명령어의 속성을 변경할 수 있습니다 . 재 정의를 위해서는 기본적으로 해당 정의보다 재 정의 날짜가 더 늦어야 합니다 .

재 정의 중에 여러 속성을 동시에 변경할 수 없습니다. 변경할 속성마다 별도의 REDEF 연산을 프로그래밍해야 합니다 .

두 개 이상의 속성을 동시에 변경하도록 프로그래밍한 경우 항상 마지막 변경이 활성화됩니다.

재 정의 가능한 속성

" 정의 가능 속성 및 재정의 가능 속성 개요 [쪽 46]" 를 참조하십시오 .

로컬 사용자 변수 (PUD/LUD)

로컬 사용자 변수 (PUD/LUD) 는 재 정의가 허용되지 않습니다 .

구문

REDEF < 이름 > <PP_ 정지 >

REDEF < 이름 > < 물리 _ 단위 >

REDEF < 이름 > < 한계 _ 값 >

REDEF < 이름 > < 사용 _ 권한 >

REDEF < 이름 > < 초기화 _ 시간 >

REDEF < 이름 > < 초기화 _ 시간 > < 초기화 _ 값 >

의미

REDEF: 특정 시스템 변수 , 사용자 변수 및 NC 언어 명령어 속성의 재 정의를 위한 명령어

< 이름 >: 사전 정의된 변수 또는 NC 언어 명령어의 이름

<PP 정지 >: 전처리 정지

SYNR: 읽는 동안 전처리 정지

SYNW: 쓰는 동안 전처리 정지

SYNRW: 읽거나 쓰는 동안 전처리 정지

< 물리 _ 단위 >: 물리 단위

PHU < 단위 >:

" 속성 : 물리 단위 (PHU) [쪽 39]" 를 참조하십시오 .

참고
다음은 재 정의할 수 없습니다 .
- 시스템 변수
- 전역 사용자 데이터 (GUD)
- 데이터 유형 : BOOL, AXIS, STRING, FRAME

< 한계값 >: 하한값 및 / 또는 상한값

LLI < 한계값 >: 하한값

ULI < 한계값 >: 상한값

" 속성 : 한계값 (LLI, ULI) [쪽 37]" 를 참조하십시오 .

참고
다음은 재 정의할 수 없습니다 .
- 시스템 변수
- 전역 사용자 데이터 (GUD)
- 데이터 유형 : BOOL, AXIS, STRING, FRAME

< 사용 권한 >: 가공 프로그램 또는 OPI 를 통한 읽기 / 쓰기 사용 권한

APX < 사용 권한 >: 실행 : NC 언어 요소

APRP < 사용 권한 >: 읽기 : 가공 프로그램

APWP < 사용 권한 >: 쓰기 : 가공 프로그램

APRB < 사용 권한 >: 읽기 : OPI

APWB < 사용 권한 >: 쓰기 : OPI

사용 권한 범위 : 0 ~ 7

" 속성 : 사용 권한 (APR, APW, APRP, APWP, APRB, APWB) [쪽 41]" 를 참조하십시오 .

< 초기화 _ 시간 >: 변수가 다시 초기화되는 시점

INIPO: POWER ON 시

INIRE: 메인 프로그램의 끝 , NC 리셋 또는 POWER ON 시

INICF: 새 설정, 메인 프로그램의 끝, NC 리셋 또는 POWER ON 시

PRLOC: 메인 프로그램의 끝 , 로컬 변경 후 NC 리셋 또는 POWER ON 시

" 속성 : 초기화 값 : [쪽 34]" 를 참조하십시오 .

< 초기화 _ 값 >: 초기화 값 :

초기화 값을 재 정의할 때 항상 초기화 시간을 함께 지정해야 합니다 (< 초기화 _ 시간 > 참조).

" 속성 : 초기화 값 : [쪽 34]" 를 참조하십시오 .

배열 변수의 초기화 :

" 배열 변수 정의 및 초기화 (DEF, SET, REP): [쪽 47]" 를 참조하십시오 .

참고

다음은 재 정의할 수 없습니다 .

- 시스템 변수 (셋팅 데이터는 예외)

예제

장비 제조업체를 위한 데이터 블록에 시스템 변수 $TC_DPC1 재 정의

프로그램 코드

```
%_N_MGUD_DEF                                    ;  GUD 블록 : 장비 제조업체
$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)
;  설명
;  쓰기 액세스 권한 : OPI = 사용 권한 2, 가공 프로그램 = 사용 권한 3
;  참고
;  ACCESS 파일을 사용하는 경우 사용 권한의 재 정의는
;  _N_MGUD_DEF 에서 _N_MACCESS_DEF 로 이동해야 합니다 .
;  물리 단위 = [ % ]
;  한계값 : 하한값 = 0, 상한값 = 200
;  배열 변수는 POWER ON 시 4 개의 값으로 초기화합니다 .
M30
```

일반 조건

입도

재 정의는 항상 그 이름으로 식별되는 전체 변수에 적용됩니다 . 예를 들어 배열 변수를 사용해서 개별 배열 요소에 다른 속성을 지정할 수는 없습니다 .

도 참조

변수에 대한 일반 정보 변수에 대한 일반 정보 [쪽 17]

1.1.7 속성 : 초기화 값 :

사용자 변수의 정의 (DEF)

정의 과정에서 다음의 사용자 변수에 대해 초기화 값을 미리 지정할 수 있습니다 .

- 전역 사용자 변수 (GUD)
- 프로그램 전역 사용자 변수 (PUD)
- 로컬 사용자 변수 (LUD)

시스템 변수 및 사용자 변수의 재 정의 (REDEF)

재 정의 과정에서 다음의 변수에 대해 초기화 값을 미리 지정할 수 있습니다 .

- 시스템 데이터
  - 셋팅 데이터
- 사용자 데이터
  - R- 파라미터
  - 동기 동작 변수 ($AC_MARKER, $AC_PARAM, $AC_TIMER)
  - 동기 동작 GUD (SYG_xy[ ], 여기서 x=R, I, B, A, C, S, y=S, M, U, 4~9)
  - EPS 파라미터
  - 공구 데이터 OEM
  - 메거진 데이터 OEM
  - 전역 사용자 변수 (GUD)

재초기화 시간
변수를 재 정의할 때 해당 변수를 다시 초기화 (즉 , 초기화 값으로 리셋) 해야 하는 시간을 지정할 수 있습니다 .

- INIPO(POWER ON)

  해당 변수는 POWER ON 시 다시 초기화 됩니다 .

- INIRE (리셋)

  해당 변수는 NC 리셋, 모드 그룹 리셋, 가공 프로그램의 끝 (M02/M30) 또는 POWER ON 시 다시 초기화 됩니다 .

- INICF (새 설정)

  해당 변수는 HMI 상의 새 설정 소프트 키를 누르거나 가공 프로그램 명령어 NEWCONFIG 를 사용해 다시 초기화합니다 . 또는 NC 리셋 , 모드 그룹 리셋 , 가공 프로그램의 끝 (M02/M30), 또는 POWER ON 시 다시 초기화됩니다 .

- PRLOC (프로그램 로컬 변경)

  해당 변수는 현재 가공 프로그램에서 변경된 경우 NC 리셋 , 모드 그룹 리셋 또는 가공 프로그램의 끝 (M02/M30) 에서만 다시 초기화 됩니다 .

  PRLOC 속성은 프로그램 가능한 셋팅 데이터를 통해서만 변경이 가능합니다 (아래 표 참조).

도표 1-1 프로그램 가능한 셋팅 데이터

번호	식별자	G 명령어 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

도표 1-1 프로그램 가능한 셋팅 데이터

번호	식별자	G 명령어 1)
43780	$SA_OSCILL_IS_ACTIVE	OS
43790	$SA_OSCILL_START_POS	OSB
1) 이 G 명령어는 셋팅 데이터 어드레스 지정에 사용됩니다 .		

일반 조건

초기화 값 : 전역 사용자 변수 (GUD)

- NCK 유효 범위의 전역 사용자 변수 (GUD) 인 경우 INIPO (POWER ON) 만 초기화 시간으로 정의할 수 있습니다 .
- INIPO (POWER ON) 이외에도 CHAN 유효 범위의 전역 사용자 변수 (GUD) 인 경우 INIRE (리셋) 또는 INICF (새 설정) 를 초기화 시간으로 정의할 수 있습니다 .
- INIRE (리셋) 또는 INICF (새 설정) 초기화 시간 그리고 CHAN 유효범위를 가진 전역 사용자 변수 (GUD) 의 경우 NC 리셋, 모드 그룹 리셋 및 새 설정이 시작된 채널에서만 변수가 다시 초기화 됩니다 .

초기화 값 : FRAME 데이터 유형

FRAME 데이터 유형의 변수인 경우 초기화 값을 지정할 수 없습니다 . FRAME 데이터 유형의 변수는 암시적으로 , 항상 디폴트 프레임에 의해 초기화됩니다 .

초기화 값 : CHAR 데이터 유형

CHAR 데이터 유형의 변수인 경우 ASCII 코드 (0~255) 대신 해당하는 ASCII 문자를 큰 따옴표 안에 프로그래밍할 수 있습니다 (예 : "A").

초기화 값 : STRING 데이터 유형

STRING 데이터 유형의 변수인 경우 문자열은 큰 따옴표 안에 있어야 합니다 (예 : ...= "MACHINE_1").

초기화 값 : AXIS 데이터 유형

AXIS 데이터 유형의 변수인 경우 확장 어드레스를 지정할 때 축 이름은 괄호 안에 있어야 합니다 (예 : ...=(X3)).

초기화 값 : 시스템 변수

시스템 변수의 경우 재 정의를 통해 사용자 지정 초기화 값을 정의할 수 없습니다 . 시스템 변수의 초기화 값은 시스템이 지정하기 때문에 변경할 수 없습니다 . 하지만 재 정의를 통해 시스템 변수가 다시 초기화되는 시점 (INIRE, INICF) 은 변경할 수 있습니다 .

암시적 초기화 값 : AXIS 데이터 유형

AXIS 데이터 유형의 변수에는 다음과 같은 암시적 초기화 값이 사용됩니다 .

- 시스템 데이터 : " 첫 번째 기하 축 "
- 동기 동작 GUD (지정 : SYG_A*), PUD, LUD:
  머신 데이터의 축 이름 : MD20082 $MC_AXCONF_CHANAX_DEFAULT_NAME

암시적 초기화 값 : 공구 및 메거진 데이터

공구 및 메거진 데이터의 초기화 값은 다음의 머신 데이터를 사용해 정의할 수 있습니다 .
MD17520 $MN_TOOL_DEFAULT_DATA_MASK

유의사항
동기화 전역 변수를 다른 위치에서 읽어올 때 전역 변수의 재초기화를 실행하는 이벤트를 동기화하는 것은 전적으로 사용자 / 장비 제조업체의 책임입니다 .

도 참조

변수에 대한 일반 정보 변수에 대한 일반 정보 [쪽 17]

1.1.8 속성 : 한계값 (LLI, ULI)

정의 범위의 상한값 및 하한값은 다음의 데이터 유형에 대해서만 정의할 수 있습니다 .

- INT
- REAL
- CHAR

사용자 변수의 정의 (DEF) 한계값 및 암시적 초기화 값

위와 같은 데이터 유형의 사용자 변수를 정의할 때 명시적 초기화 값이 정의되어 있지 않은 경우 , 변수는 해당 데이터 유형의 암시적 초기화 값으로 설정됩니다 .

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

암시적 초기화 값이 프로그래밍한 한계값의 정의 범위를 벗어나는 경우 , 해당 변수는 암시적 초기화 값에 가장 가까운 한계값으로 초기화됩니다 .

- 암시적 초기화 값 < 하한값 (LLI) ⇒
  초기화 값 = 하한값
- 암시적 초기화 값 > 상한값 (ULI) ⇒
  초기화 값 = 상한값

예제 :

프로그램 코드	설명
DEF REAL GUD1	; 하한값 = 정의 범위 한계 ; 상한값 = 정의 범위 한계 ; 프로그래밍된 초기화 값 없음 ; => 암시적 초기화 값 = 0.0
DEF REAL LLI 5.0 GUD2	; 하한값 = 5.0 ; 상한값 = 정의 범위 한계 ; => 초기화 값 = 5.0
DEF REAL ULI –5 GUD3	; 하한값 = 정의 범위 한계 ; 상한값 = -5.0 ; => 초기화 값 = -5.0

사용자 변수의 재 정의 (REDEF) 한계값 및 현재 실제 값

사용자 변수의 한계값을 다시 정의할 때 현재 실제 값이 새 정의 범위를 벗어나게 정의하면 알람이 발생하고 한계값이 거부됩니다 .

주

사용자 변수의 재 정의 (REDEF)

사용자 변수의 한계값을 다시 정의할 때 반드시 다음의 값이 일관되게 변경되도록 주의해야 합니다 .

- 한계값
- 실제값
- INIPO, INIRE 또는 INICF 를 기준으로 한 재 정의 및 자동 재초기화의 초기화 값

도 참조

변수에 대한 일반 정보 변수에 대한 일반 정보 [쪽 17]

1.1.9 속성 : 물리 단위 (PHU)

다음과 같은 데이터 유형의 변수만 물리 단위로 지정할 수 있습니다 .

- INT
- REAL

프로그램 가능한 물리 단위 (PHU)

물리 단위는 고정된 자연수로 정의합니다 . PHU < 단위 >

다음과 같은 물리 단위를 프로그래밍할 수 있습니다 .

< 단위 >	의미	물리 단위
0	물리 단위 아님	-
1	직선위치 또는 각위치 1)2)	[mm], [inch], [도]
2	직선위치 2)	[mm], [inch]
3	각위치	[도]
4	직선속도 또는 각속도 1)2)	[mm/min], [inch/min], [rpm]
5	직선속도 2)	[mm/min]
6	각속도	[rpm]
7	직선가속 또는 각가속 1)2)	[m/s^2], [$inch/s^2$], [rev/s^2]
8	직선가속 2)	[m/s^2], [$inch/s^2$]
9	각가속	[rev/s^2]
10	직선저크 또는 각저크 1)2)	[m/s^3], [$inch/s^3$], [rev/s^3]
11	직선저크 2)	[m/s^3], [$inch/s^3$]
12	각저크	[rev/s^3]
13	시간	[s]
14	위치 제어기 게인	[16.667/s]
15	회전당 이송 속도 2)	[mm/rev], [inch/rev]
16	온도 보정 1)2)	[mm], [inch]
18	힘	[N]
19	질량	[kg]
20	관성 모멘트 3)	[kgm^2]
21	퍼센트	[%]
22	주파수	[Hz]
23	전압	[V]
24	전류	[A]
25	온도	[°C]
26	각도	[°]
27	KV	[1000/min]
28	직선위치 또는 각위치 3)	[mm], [inch], [도]
29	절삭 속도 2)	[m/min], [feet/min]

< 단위 >	의미	물리 단위
30	원주 속도 2)	[m/s], [feet/s]
31	저항	[ohm]
32	유도 계수	[mH]
33	토크 3)	[Nm]
34	토크 상수 3)	[Nm/A]
35	전류 제어기 게인	[V/A]
36	속도 제어기 게인 3)	[Nm/(rad*s)]
37	속도	[rpm]
42	전원	[kW]
43	미세 전류	[µA]
46	미세 토크 3)	[µNm]
48	밀 당	-
49	-	[Hz/s]
65	유속	[l/min]
66	압력	[bar]
67	부피 3	[cm^3]
68	시스템 게인 3)	[mm/(V*min)]
69	힘 제어기로 시스템 게인	[N/V]
155	나사 리드 3)	[mm/rev], [inch/rev]
156	나사 리드 변경 3)	[mm/rev / rev], [inch/rev / rev]

1) 물리 단위는 축의 유형 , 즉 직선 축 또는 로터리 축에 따라 달라집니다 .

2) 단위 체계 전환

G70/G71(인치 / 미터)
길이와 관련된 시스템 변수 및 사용자 변수에 읽기 / 쓰기 작업을 하기 위해 베이직 시스템 (MD10240 $MN_SCALING_SYSTEM_IS_METRIC) 을 G70/G71 로 전환한 후에는 값이 변환되지 **않습니다** (실제 값 , 디폴트 값 , 한계값).

G700/G710(인치 / 미터)
길이와 관련된 시스템 변수 및 사용자 변수에 읽기 / 쓰기 작업을 하기 위해 베이직 시스템 (MD10240 $MN_SCALING_SYSTEM_IS_METRIC) 을 G700/G710 로 전환한 후에 값이 **변환됩니다** (실제 값 , 디폴트 값 , 한계값).

3) 변수가 현재 NC 측정 시스템 (인치 / 미터) 에 따라 자동으로 변환되지 **않습니다** . 측정 방식 전환은 전적으로 사용자 / 장비 제조업체의 책임입니다 .

주

형식 변환에 따른 레벨 오버플로우

길이를 표시하는 물리 단위를 포함하는 모든 사용자 변수 (GUD/PUD/LUD) 는 내부적으로 미터 형식으로 저장됩니다 . NCK 메인 실행 (예 : 동기 동작) 에 이런 형식의 변수를 지나치게 사용하면 측정 방식 변환으로 인해 보간 단계에서 CPU 타임 오버플로우가 야기되고 4240 알람이 발생합니다 .

유의사항
단위 호환성 변수 (지정 , 비교 , 계산 등) 를 사용할 때 관련 단위의 호환성은 점검하지 않습니다 . 변환이 필요한 경우 호환 여부는 전적으로 사용자 / 장비 제조업체의 책임입니다 .

도 참조

변수에 대한 일반 정보 변수에 대한 일반 정보 [쪽 17]

1.1.10 속성 : 사용 권한 (APR, APW, APRP, APWP, APRB, APWB)

다음의 사용 권한은 프로그래밍할 때 지정되어야 하며 그에 상응하는 사용 권한이 있습니다 .

사용 권한	사용 권한 레벨
시스템 암호	0
장비 제조업체 암호	1
서비스 암호	2
최종 사용자 암호	3
키 스위치 위치 3	4
키 스위치 위치 2	5
키 스위치 위치 1	6
키 스위치 위치 0	7

사용자 변수의 정의 (DEF)

다음의 변수에 대해 사용 권한 (APR.../APW...) 을 정의할 수 있습니다 .

- 전역 사용자 데이터 (GUD)

시스템 변수 및 사용자 변수의 재 정의 (REDEF)

다음의 변수에 대해 사용 권한 (APR.../APW...) 을 재 정의할 수 있습니다 .

- 시스템 데이터
  - 머신 데이터
  - 셋팅 데이터
  - FRAME
  - 공정 데이터
  - 리드스크류 에러 보정 (LEC) 데이터
  - 처짐 보정 (CEC)
  - 상한 돌기 보정 (QEC)
  - 메거진 데이터
  - 공구 데이터
  - 보호 영역
  - 오리엔테이션 조정 가능한 공구 홀더
  - 운동학적 체인
  - 3D 보호 영역
  - 작업 영역 제한
  - ISO 공구 데이터
- 사용자 데이터
  - R 파라미터
  - 동기 동작 변수 ($AC_MARKER, $AC_PARAM, $AC_TIMER)
  - 동기 동작 GUD (SYG_xy[ ], 여기서 x=R, I, B, A, C, S, y=S, M, U, 4~9)
  - EPS 파라미터
  - 공구 데이터 OEM
  - 메거진 데이터 OEM
  - 전역 사용자 변수 (GUD)

주

해당 변수를 재 정의할 때 최저 사용 권한 레벨 7 부터 전용 사용 권한 레벨 1 (장비 제조업체) 사이의 사용 권한을 자유롭게 변수에 지정할 수 있습니다 .

NC 언어 명령 재 정의 (REDEF)

다음의 NC 언어 명령에 대해 사용 권한 또는 실행 권한 (APX) 을 재 정의할 수 있습니다 .

- G 코드 / 준비 기능

  참고 자료 :

  /PG/ 프로그래밍 매뉴얼 , 기본 사항 ; 장 : G 코드 / 준비 기능

- 사전 정의된 기능

  참고 자료 :

  /PG/ 프로그래밍 매뉴얼 , 기본 사항 ; 장 : 사전 정의된 기능

- 사전 정의된 서브프로그램 호출

  참고 자료 :

  /PG/ 프로그래밍 매뉴얼 , 기본 사항 ; 장 : 사전 정의된 서브프로그램 호출

- 동기 동작으로 수행하는 D0 작업
- 싸이클 프로그램 식별자

  싸이클은 싸이클 디렉토리 중 하나에 저장되어야 하며 PROC 연산을 포함해야 합니다 .

가공 프로그램 및 싸이클과 관련된 사용 권한 (APRP, APWP)

사용 권한을 여러 등급으로 구분하면 다음과 같은 설정이 가능해지므로 가공 프로그램이나 싸이클에 대한 액세스를 제한할 수 있습니다 .

- APRP 0/APWP 0
  - 가공 프로그램 처리 중에 시스템 암호를 설정해야 합니다 .
  - 해당 싸이클은 _N_CST_DIR 디렉토리 (시스템) 에 저장되어야 합니다 .
  - _N_CST_DIR 디렉토리의 경우 , MD11160 $MN_ACCESS_EXEC_CST 에서 시스템 레벨로 사용 권한이 설정되어야 합니다 .
- APRP 1/APWP 1 또는 APRP 2/APWP 2
  - 가공 프로그램 처리 중에 장비 제조업체 암호 또는 서비스 암호를 설정해야 합니다 .
  - 해당 싸이클은 _N_CMA_DIR (장비 제조업체) 또는 _N_CST_DIR (시스템) 디렉토리에 저장되어야 합니다 .
  - _N_CMA_DIR 또는 _N_CST_DIR 디렉토리의 경우 , MD11161 $MN_ACCESS_EXEC_CMA 또는 MD11160 $MN_ACCESS_EXEC_CST 에서 최소한 장비 제조업체 레벨로 사용 권한이 설정되어야 합니다 .

- APRP 3/APWP 3
  - 가공 프로그램 처리 중에 최종 사용자 암호를 설정해야 합니다 .
  - 해당 싸이클은 _N_CUS_DIR (사용자), _N_CMA_DIR 또는 _N_CST_DIR 디렉토리에 저장해야 합니다 .
  - _N_CUS_DIR, _N_CMA_DIR 또는 _N_CST_DIR 디렉토리의 경우 , MD11162 $MN_ACCESS_EXEC_CUS, MD11161 $MN_ACCESS_EXEC_CMA 또는 MD11160 $MN_ACCESS_EXEC_CST 에서 최소한 최종 사용자 레벨로 사용 권한이 설정되어야 합니다 .
- APRP 4...7/APWP 4...7
  - 가공 프로그램 처리 중에 키 스위치를 3 ... 0 으로 설정해야 합니다 .
  - 해당 싸이클은 _N_CUS_DIR, _N_CMA_DIR 또는 _N_CST_DIR 디렉토리에 저장해야 합니다 .
  - _N_CUS_DIR, _N_CMA_DIR 또는 _N_CST_DIR 디렉토리의 경우 , MD11162 $MN_ACCESS_EXEC_CUS, MD11161 $MN_ACCESS_EXEC_CMA 또는 MD11160 $MN_ACCESS_EXEC_CST 에서 최소한 관련 키 스위치 위치에 맞는 사용 권한이 설정되어야 합니다 .

OPI (APRB, APWB) 와 관련된 사용 권한

사용 권한 (APRB, APWB) 은 OPI는 물론 모든 시스템 구성 요소 (HMI, PLC, 외부 컴퓨터, EPS 서비스 등) 를 통한 시스템 및 사용자 변수 액세스를 제한합니다 .

주

로컬 HMI 사용 권한

시스템 데이터 상의 사용 권한을 변경할 때 변경 사항이 반드시 HMI 메커니즘을 통해 정의한 사용 권한과 일치하도록 해야 합니다 .

APR/APW 액세스 속성

호환성을 위해 속성 APR 및 APW 가 암시적으로 속성 APRP / APRB 및 APWP / APWB 에 매핑됩니다 .

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

ACCESS 파일을 사용해 사용 권한 설정

ACCESS 파일을 사용해 사용 권한을 지정하면 시스템 데이터 , 사용자 데이터 및 NC 언어 명령에 대한 사용 권한은 앞으로도 해당 ACCESS 파일 내에서만 다시 정의가 가능합니다 . 전역 사용자 데이터 (GUD) 는 예외입니다 . GUD 의 경우 필요하다면 계속 해당 정의 파일에서 사용 권한을 다시 정의해야 합니다 .

지속적인 액세스 보호를 위해서는 해당 디렉토리의 실행 권한 및 액세스 보호 관련 머신 데이터를 지속적으로 수정해야 합니다 .

원칙적으로 액세스 보호 절차는 다음과 같습니다 .

- 필요한 정의 파일 생성 :
  - _N_DEF_DIR/_N_SACCESS_DEF
  - _N_DEF_DIR/_N_MACCESS_DEF
  - _N_DEF_DIR/_N_UACCESS_DEF
- 재 정의를 위해 필요한 값에 정의 파일에 대한 쓰기 권한 설정
  - MD11170 $MN_ACCESS_WRITE_SACCESS
  - MD11171 $MN_ACCESS_WRITE_MACCESS
  - MD11172 $MN_ACCESS_WRITE_UACCESS
- 싸이클에서 보호 요소에 액세스하기 위해서는 싸이클 디렉토리 _N_CST_DIR, _N_CMA_DIR 및 _N_CST_DIR 의 실행 권한 및 쓰기 권한을 수정해야 합니다 .

  실행 권한
  - MD11160 $MN_ACCESS_EXEC_CST
  - MD11161 $MN_ACCESS_EXEC_CMA
  - MD11162 $MN_ACCESS_EXEC_CUS

  쓰기 권한
  - MD11165 $MN_ACCESS_WRITE_CST
  - MD11166 $MN_ACCESS_WRITE_CMA
  - MD11167 MN_ACCESS_WRITE_CUS

  실행 권한은 적어도 사용되는 요소의 최고 사용 권한과 동일한 레벨의 사용 권한으로 설정되어야 합니다 .

  쓰기 권한은 적어도 실행 권한과 동일한 레벨의 사용 권한으로 설정되어야 합니다 .
- 로컬 HMI 싸이클 디렉토리의 쓰기 권한은 로컬 NC 싸이클 디렉토리와 동일한 레벨의 사용 권한으로 설정되어야 합니다 .

  참고 자료

  /BAD/ 조작 매뉴얼 , HMI 고급 ,

  장 : 서비스 운영 영역 > 데이터 운영 > 속성 변경

ACCESS 파일에 서브프로그램 호출

액세스 보호를 추가적으로 설정하기 위해 ACCESS 파일에 서브프로그램 (SPF 또는 MPF 식별자) 을 호출할 수 있습니다 . 서브프로그램은 호출하는 ACCESS 파일의 실행 권한을 그대로 물려 받습니다 .

주

ACCESS 파일에서는 사용 권한만 다시 정의할 수 있습니다 . 다른 모든 속성은 계속해서 해당 정의 파일에서 프로그래밍하거나 다시 정의해야 합니다 .

도 참조

변수에 대한 일반 정보 변수에 대한 일반 정보 [쪽 17]

1.1.11 정의 가능 속성 및 재정의 가능 속성 개요

다음 표는 어떤 데이터 유형의 어떤 속성을 정의 (DEF) 및 / 또는 재정의 (REDEF) 할 수 있는지 보여줍니다 .

시스템 데이터

데이터 유형	초기값	한계값	물리 단위	사용 권한
머신 데이터	---	---	---	REDEF
셋팅 데이터	REDEF	---	---	REDEF
FRAME 데이터	---	---	---	REDEF
공정 데이터	---	---	---	REDEF
리드스크류 에러 보정 (EEC)	---	---	---	REDEF
처짐 보정 (CEC)	---	---	---	REDEF
상한 돌기 보정 (QEC)	---	---	---	REDEF
매거진 데이터	---	---	---	REDEF
공구 데이터	---	---	---	REDEF
보호 영역	---	---	---	REDEF
오리엔테이션 조정 가능한 공구 홀더	---	---	---	REDEF
운동학적 체인	---	---	---	REDEF
3D 보호 영역	---	---	---	REDEF
작업 영역 제한	---	---	---	REDEF
ISO 공구 데이터	---	---	---	REDEF

사용자 데이터

데이터 유형	초기값	한계값	물리 단위	사용 권한
R 파라미터	REDEF	REDEF	REDEF	REDEF
동기 동작 변수 ($AC_...)	REDEF	REDEF	REDEF	REDEF
동기 동작 GUD (SYG_...)	REDEF	REDEF	REDEF	REDEF
EPS 파라미터	REDEF	REDEF	REDEF	REDEF
공구 데이터 OEM	REDEF	REDEF	REDEF	REDEF
매거진 데이터 OEM	REDEF	REDEF	REDEF	REDEF
전역 사용자 변수 (GUD)	DEF/REDEF	DEF	DEF	DEF/REDEF
로컬 사용자 변수 (PUD/LUD)	DEF	DEF	DEF	---

도 참조

변수에 대한 일반 정보 변수에 대한 일반 정보 [쪽 17]

1.1.12 배열 변수 정의 및 초기화 (DEF, SET, REP):

기능

사용자 변수는 1 차원 배열부터 최대 3 차원 배열로 정의할 수 있습니다 .

- 1 차원 : DEF <data type> <variable name>[<n>]
- 2 차원 : DEF <data type> <variable name>[<n>,<m>]
- 3 차원 : DEF <data type> <variable name>[<n>,<m>,<o>]

주

STRING 데이터 유형의 사용자 변수는 최대 2 차원 배열로 정의할 수 있습니다 .

데이터 유형

다음과 같은 데이터 유형의 사용자 변수를 배열로 정의할 수 있습니다 . BOOL, CHAR, INT, REAL, STRING, AXIS, FRAME

배열 요소에 값 지정

다음과 같은 시점에 배열 요소에 값을 지정할 수 있습니다 .

- 배열 정의 중 (초기화 값)
- 프로그램 실행 중

값은 다음과 같은 방법으로 지정할 수 있습니다 .

- 배열 요소를 명시적으로 지정
- 배열 요소를 시작 요소로 명시적으로 지정하고 값 목록 (SET) 지정
- 배열 요소를 시작 요소로 명시적으로 지정하고 값과 해당 값의 반복 주기를 지정 (REP)

주

FRAME 데이터 유형의 사용자 변수는 초기화 값으로 지정할 수 없습니다 .

구문 (DEF)

DEF <data type> <variable name>[<n>,<m>,<o>]
DEF STRING[< 문자열 길이 >] < 변수 이름 >[<n>,<m>]

구문 (DEF...=SET...)

값 목록 사용 :

- 정의 중 :

  DEF < 데이터 유형 > < 변수 이름 >[<n>,<m>,<o>]=SET(< 값 1>,< 값 2>,...)

  해당값 :

  DEF < 데이터 유형 > < 변수 이름 >[<n>,<m>,<o>]=(<value1>,<value2>,...)

주

값 목록을 통해 초기화하는 경우 SET 을 지정할 필요가 없습니다 .

- 값 지정 중 :

  < 변수 이름 >[<n>,<m>,<o>]=SET(<VALUE1>,< 값 2>,...)

구문 (DEF...=REP...)

값을 반복 사용

- 정의 중 :

  DEF <data type> <variable name>[<n>,<m>,<o>]=REP(<value>)

  DEF < 데이터 유형 > < 변수 이름 >[<n>,<m>,<o>]=REP(< 값 >, < 번호 _ 배열 _ 요소 >)

- 값 지정 중 :

  <variable name>[<n>,<m>,<o>]=REP(<value>)

  DEF <data type> <variable name>[<n>,<m>,<o>]=REP(<value>,<number_array_elements>)

의미

DEF:	변수를 정의하기 위한 명령
< 데이터 유형 >:	변수의 데이터 유형 범위 : • 시스템 변수의 경우 : BOOL, CHAR, INT, REAL, STRING, AXIS • GUD 또는 LUD 변수의 경우 : BOOL, CHAR, INT, REAL, STRING, AXIS, FRAME
< 문자열 길이 >:	STRING 데이터 유형의 문자에 대한 최대값
< 변수 이름 >:	변수 이름 .
[<n>,<m>,<o>]:	배열 크기 또는 배열 인덱스
<n>:	1 차원의 배열 크기 또는 배열 인덱스 유형 : INT (시스템 변수 및 AXIS 의 경우) 범위 : 최대 배열 크기 : 65535 배열 인덱스 : 0 ≤ n ≤ 65534
<m>:	2 차원의 배열 크기 또는 배열 인덱스 유형 : INT (시스템 변수 및 AXIS 의 경우) 범위 : 최대 배열 크기 : 65535 배열 인덱스 : 0 ≤ m ≤ 65534
<o>:	3 차원의 배열 크기 또는 배열 인덱스 유형 : INT (시스템 변수 및 AXIS 의 경우) 범위 : 최대 배열 크기 : 65535 배열 인덱스 : 0 ≤ o ≤ 65534
SET:	지정된 값 목록을 이용해 값 지정
(< 값 1>,< 값 2>,...):	값 목록
REP:	지정된 < 값 > 을 이용해 값 지정
< 값 >:	값 , REP 로 초기화하는 경우 배열 요소 값이 작성됩니다 .
< 번호 _ 배열 _ 요소 >:	지정된 < 값 > 이 쓰여져야 할 배열 요소의 수 나머지 배열 요소에는 시점에 따라 다음이 적용됩니다 . • 배열 정의 시 초기화 : → 나머지 배열 요소에 0 이 쓰여집니다 . • 프로그램 실행 중 지정 : → 배열 요소의 실제 값이 그대로 유지됩니다 . 이 파라미터를 프로그래밍하지 않을 경우 모든 배열 요소에 < 값 > 이 쓰여집니다 . 파라미터가 0 인 경우 시점에 따라 다음이 적용됩니다 . • 배열 정의 시 초기화 : 모든 요소가 미리 지정된 0 입니다 . • 프로그램 실행 중 지정 : → 배열 요소의 실제 값이 변경되지 않은 상태를 유지합니다 .

배열 인덱스

배열 요소의 암시적인 순서는 배열 인덱스의 반복 때문에 왼쪽부터 오른쪽 순서로 지정됩니다 (예 : SET 또는 REP 를 사용해 값을 지정하는 경우).

예제 : 3 차원 배열을 24 개 배열 요소로 초기화 :

```
DEF INT FELD[2,3,4] = REP(1,24)
   FELD[0,0,0] = 1          첫 번째 배열 요소
   FELD[0,0,1] = 1          두 번째 배열 요소
   FELD[0,0,2] = 1          세 번째 배열 요소
   FELD[0,0,3] = 1          네 번째 배열 요소
   ...
   FELD[0,1,0] = 1          다섯 번째 배열 요소
   FELD[0,1,1] = 1          여섯 번째 배열 요소
   ...
   FELD[0,2,3] = 1          열두 번째 배열 요소
   FELD[1,0,0] = 1          열세 번째 배열 요소
   FELD[1,0,1] = 1          열네 번째 배열 요소
   ...
   FELD[1,2,3] = 1          스물네 번째 배열 요소
```

해당값 :

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

예제 : 전체 변수 배열 초기화

실제 지정의 경우 다이어그램을 참조하십시오 .

프로그램 코드
N10 DEF REAL FELD1[10,3]=SET(0,0,0,10,11,12,20,20,20,30,30,30,40,40,40,)
N20 ARRAY1[0,0] = REP(100)
N30 ARRAY1[5,0] = REP(-100)
N40 FELD1[0,0]=SET(0,1,2,-10,-11,-12,-20,-20,-20,-30, , , ,-40,-40,-50,-60,-70)
N50 FELD1[8,1]=SET(8.1,8.2,9.0,9.1,9.2)

배열 색인 2

[1,2]	N10: 정의를 위한 초기화			N20/N30: 동일한 값으로 초기화			N40/N50: 다양한 값으로 초기화		
	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
	배열 요소 [5.0]에서 [9.2]까지는 기본값(0.0)으로 초기화 되었습니다.						배열 요소 [3,1]에서 [4,0]까지는 기본값(0.0)으로 초기화 되었습니다. 배열 요소 [6,0]에서 [8,0]까지는 변경되지 않았습니다.		

1

도 참조

배열 변수 정의 및 초기화 (DEF, SET, REP): 추가 정보 배열 변수 정의 및 초기화 (DEF, SET, REP): 추가 정보 [쪽 52]

변수에 대한 일반 정보 변수에 대한 일반 정보 [쪽 17]

1.1.13 배열 변수 정의 및 초기화 (DEF, SET, REP): 추가 정보

추가 정보 (SET)

정의 중 초기화

- 첫 번째 배열 요소부터 시작해 값 목록에 프로그래밍된 배열 요소의 수만큼 값 목록에 있는 값을 지정합니다 .
- 값 목록에 명시적으로 선언된 값이 없는 경우 배열 요소에 값 0 이 지정됩니다 (값 목록에서의 갭).
- AXIS 데이터 유형의 변수인 경우 값 목록에서의 갭이 허용되지 없습니다 .
- 값 목록에 정의된 배열 요소보다 더 많은 값이 포함된 경우 경고가 표시됩니다 .

프로그램 실행 시 값 지정

프로그램 실행 시 값을 지정하는 경우 앞서 설명한 정의 관련 규칙이 적용됩니다 . 또한 다음과 같은 옵션을 지원합니다 .

- 표현식도 값 목록에서 요소로 허용됩니다 .
- 값 지정은 프로그래밍된 배열 인덱스부터 시작합니다 . 값을 하위 배열에 선택적으로 지정할 수 있습니다 .

예제 :

프로그램 코드	설명
DEF INT ARRAY[5,5]	; 배열 정의
ARRAY[0,0]=SET(1,2,3,4,5)	; 처음 5 개 배열 요소 [0,0] ~ [0,4] 에 값 지정
FELD[0,0]=SET(1,2, , ,5)	; 처음 5 개 배열 요소 [0,0] ~ [0,4] 에 갭을 포함한 값을 지정 , 배열 요소 [0,2] 및 [0,3] = 0
ARRAY[2,3]=SET(VARIABLE,4*5.6)	; 배열 인덱스 [2,3]: [2,3] = VARIABLE [2,4] = 4 * 5.6 = 22.4 에서 시작하는 변수 및 표현식을 포함한 값 지정

추가 정보 (REP)

정의 중 초기화

- 모든 배열 요소 또는 선택적으로 지정한 배열 요소는 지정된 값 (상수) 으로 초기화합니다 .
- FRAME 데이터 유형의 변수는 초기화할 수 없습니다 .

예제 :

프로그램 코드	설명
DEF REAL varName[10]=REP(3.5,4)	; 배열 정의 및 배열 요소 [0] ~ [3] 을 값 3.5 로 초기화 .

프로그램 실행 시 값 지정

프로그램 실행 시 값을 지정하는 경우 앞서 설명한 정의 관련 규칙이 적용됩니다 . 또한 다음과 같은 옵션을 지원합니다 .

- 표현식도 값 목록에서 요소로 허용됩니다 .
- 값 지정은 프로그래밍된 배열 인덱스부터 시작합니다 . 값을 하위 배열에 선택적으로 지정할 수 있습니다 .

예제 :

프로그램 코드	설명
DEF REAL varName[10]	; 배열 정의
varName[5]=REP(4.5,3)	; 배열 요소 [5] ~ [7] = 4.5
R10=REP(2.4,3)	; R- 파라미터 R10 ~ R12 = 2.4
DEF FRAME FRM[10]	; 배열 정의
FRM[5] = REP(CTRANS (X,5))	; 배열 요소 [5] ~ [9] = CTRANS(X,5)

추가 정보 (일반)

축 머신 데이터에 값 지정

원칙적으로 축 머신 데이터는 AXIS 데이터 유형의 배열 인덱스를 갖습니다 . 머신 데이터의 축 항목에 SET 또는 REP 를 사용해 값을 지정하면 해당 배열 인덱스가 무시되거나 처리되지 않습니다 .

예제 : 머신 데이터 MD36200 $MA_AX_VELO_LIMIT 에 값 지정

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

다음에 해당합니다 .

$MA_AX_VELO_LIMIT[1,AX1] = 1.1

$MA_AX_VELO_LIMIT[2,AX1] = 2.2

$MA_AX_VELO_LIMIT[3,AX1] = 3.3

유의사항
축 머신 데이터에 값 지정 SET 또는 REP를 사용해 축 머신 데이터에 값을 지정하면 AXIS 데이터 유형의 배열 인덱스가 무시되거나 처리되지 않습니다 .

필요 메모리

데이터 유형	요소당 필요 메모리
BOOL	1 바이트
CHAR	1 바이트
INT	4 바이트
REAL	8 바이트
STRING	(문자열 길이 + 1) 바이트
FRAME	축 수에 따라 ~ 400 바이트
AXIS	4 바이트

도 참조

배열 변수 정의 및 초기화 (DEF, SET, REP): 배열 변수 정의 및 초기화 (DEF, SET, REP): [쪽 47]

1.1.14 데이터 유형

NC 에는 다음과 같은 데이터 유형을 사용할 수 있습니다 .

데이터 유형	의미	값 범위
INT	부호가 있는 정수	-2147483648 ... +2147483647
REAL	실수 (LONG REAL ~ IEEE)	±(~ $2,2*10^{-308}$ … ~ $1,8*10^{+308}$)
BOOL	참 값 TRUE (1) / FALSE (0)	1, 0
CHAR	ASCII 문자	ASCII 코드 0 ~ 255
STRING	정의된 길이의 문자열	최대 200 문자기호 (특수 문자기호가 아님)
AXIS	축 이름 / 스핀들 이름	채널 축 이름
FRAME	스태틱 좌표 변환을 위한 기하 파라미터 (변환, 회전, 스케일링 , 미러링)	---

암시적 데이터 유형 변환

다음과 같은 데이터 유형 변환이 가능하며 변환은 지정 및 파라미터 전송 중 암시적으로 실행됩니다 .

↓ 에서 → 로 변환	REAL	INT	BOOL
REAL	x	o	&
INT	x	x	&
BOOL	x	x	x
x: 제한 없이 사용 가능 o: 값 범위를 초과한 데이터 유실 가능 ⇒ 알람 ; 반올림 : 소수 자리 값 ≥ 0.5 ⇒ 반올림 , 소수 자리 값 < 0.5 ⇒ 버림 &: 값 ≠ 0 ⇒ TRUE, 값 == 0 ⇒ FALSE			

도 참조

변수에 대한 일반 정보 변수에 대한 일반 정보 [쪽 17]

1.2 간접 프로그램

1.2.1 간접 프로그램 주소

기능

간접적으로 어드레스를 프로그래밍하는 경우 확장된 어드레스 (인덱스) 가 적절한 유형의 변수로 교체됩니다 .

주

다음의 경우 간접 프로그램 어드레스를 사용할 수 없습니다 .

- N (블록 번호)
- L (서브프로그램)
- 설정 가능한 어드레스
  (예를 들어 X1 대신 X[1] 은 허용되지 않습니다 .)

구문

<ADDRESS>[<Index>]

의미

<ADDRESS>[...]:	확장자 (인덱스) 를 가진 고정 어드레스
< 인덱스 >:	인덱스 변수 , 예를 들어 스핀들 번호 , 축 , ...

예제

예제 1: 스핀들 번호 간접 프로그램

직접 프로그램 :

프로그램 코드	설명
S1=300	; 스핀들 번호 1 에 대한 속도 (rpm).

간접 프로그램 :

프로그램 코드	설명
DEF INT SPINU=1	; 변수 , 유형 INT 및 값 지정 정의 .
S[SPINU]=300	; 스핀들에 대한 속도 300rpm, 번호는 SPINU 변수에 저장됩니다 (이 예제 1 에서는 스핀들 번호 1).

예제 2: 축 간접 프로그램

직접 프로그램 :

프로그램 코드	설명
FA[U]=300	; 축 "U" 에 대해 피드 속도 300

간접 프로그램 :

프로그램 코드	설명
DEF AXIS AXVAR2=U	; 변수 , 유형 AXIS 및 값 지정 정의 .
FA[AXVAR2]=300	; 어드레스 이름이 AXVAR2 이름의 변수에 저장되는 축에 대한 피드 속도 300.

예제 3: 축 간접 프로그램

직접 프로그램 :

프로그래밍	설명
$AA_MM[X]	; 프로브가 "X" 축의 값 (MCS) 을 측정했습니다 .

간접 프로그램 :

프로그램 코드	설명
DEF AXIS AXVAR3=X	; 변수 , 유형 AXIS 및 값 지정 정의 .
$AA_MM[AXVAR3]	; 프로브가 이름이 변수 AXVAR3 에 저장된 값 (MCS) 을 측정했습니다 .

예제 4: 축 간접 프로그램

직접 프로그램 :

프로그램 코드
X1=100 X2=200

간접 프로그램 :

프로그램 코드	설명
DEF AXIS AXVAR1 AXVAR2	; 두 가지 유형의 AXIS 변수 정의 .
AXVAR1=(X1) AXVAR2=(X2)	; 축 이름 지정 .
AX[AXVAR1]=100 AX[AXVAR2]=200	; 어드레스 이름이 AXVAR1 및 AXVAR2 이름을 가진 변수에 저장된 축 이송

예제 5: 축 간접 프로그램

직접 프로그램 :

프로그램 코드
G2 X100 I20

간접 프로그램 :

프로그램 코드	설명
DEF AXIS AXVAR1=X	; 변수 , 유형 AXIS 및 값 지정 정의 .
G2 X100 IP[AXVAR1]=20	; 축의 중심점 데이터 간접 프로그램 , 어드레스 이름이 AXVAR1 이름의 변수에 저장 .

예제 6: 배열 요소 간접 프로그램

직접 프로그램 :

프로그램 코드	설명
DEF INT ARRAY1[4,5]	; 배열 1 정의

간접 프로그램 :

프로그램 코드	설명
DEFINE DIM1 AS 4	; 배열 치수의 경우 배열 크기를 고정값으로 지정해야 합니다 .
DEFINE DIM2 AS 5	
DEF INT ARRAY[DIM1,DIM2]	
ARRAY[DIM1-1,DIM2-1]=5	

예제 7: 간접 서브프로그램 호출

프로그램 코드	설명
CALL "L" << R10	; 프로그램 호출 , 값이 R10 에 위치 (단계별 문자열).

1.2.2 G 코드 간접 프로그램

기능

G 코드의 간접 프로그램을 통해 싸이클을 효율적으로 프로그래밍할 수 있습니다 .

구문

G[<group>]=<number>

의미

G[...]:	확장자 (인덱스) 를 가진 G 명령
< 그룹 >:	인덱스 파라미터 : G 코드 그룹 유형 : INT
< 번호 >:	G 코드 번호에 대한 변수 유형 : INT 또는 REAL

주

일반적으로 구문을 결정하지 않는 G 코드만 간접적으로 프로그래밍할 수 있습니다 .

구문을 결정하는 G 코드 중 가운데 G 코드 그룹 1 만 프로그래밍이 가능합니다 .
구문을 결정하는 G 코드 가운데 G 코드 그룹 2, 3 및 4 는 불가능합니다 .

주

간접 G 코드 프로그래밍에서 산술 함수는 허용되지 않습니다 . G 코드 값을 계산해야 하는 경우 간접 G 코드 프로그램을 수행하기 전에 개별 가공 프로그램 라인에서 계산해야 합니다 .

예제

예제 1: 셋터블 제로 옵셋 (G 코드 그룹 8)

프로그램 코드	설명
N1010 DEF INT INT_VAR	
N1020 INT_VAR=2	
...	
N1090 G[8]=INT_VAR G1 X0 Y0	; G54
N1100 INT_VAR=INT_VAR+1	; G 코드 계산
N1110 G[8]=INT_VAR G1 X0 Y0	; G55

예제 2: 레벨 선택 (G 코드 그룹 6)

프로그램 코드	설명
N2010 R10=$P_GG[6]	; G 코드 그룹 6 의 활성 G 코드 읽기
...	
N2090 G[6]=R10	

참고 자료

G 코드 그룹에 대한 자세한 내용은
프로그래밍 매뉴얼 , 기본 사항의 "G 코드 / 준비 기능 목록 " 장을 참조하십시오 .

1.2.3 위치 특성 간접 프로그램 (BP)

기능

예를 들어 축 위치의 증분치 또는 절대치 프로그래밍과 같이 위치 특성을 키워드 BP 와 연계하여 변수로 간접적으로 프로그램할 수 있습니다 .

적용

위치 특성의 간접 프로그램은 **교체 싸이클**에서 사용되며 , 이 경우처럼 위치 특성을 키워드 (예를 들어 IC, AC, ...) 로 프로그래밍할 경우 다음과 같은 이점이 있습니다 .

변수로 간접 프로그래밍할 경우 결과적으로 CASE 명령이 필요 **없으며** 그렇지 않을 경우 모든 가능한 위치 특성에 대해 분기됩니다 .

구문

<POSITIONING COMMAND>[<axis/spindle>]=
BP(<position>,<position attribute)
<axis/spindle>=BP(<position>,<position attribute)

의미

<POSITIONING COMMAND>[]:	다음 포지셔닝 명령을 키워드 BP 와 함께 프로그래밍할 수 있습니다 . POS, POSA,SPOS, SPOSA 또한 다음도 가능합니다 . • 해당 채널에 전체 축 및 스핀들 이름이 있습니다 . <axis/spindle> • 변수 축 / 스핀들 식별자 AX
< 축 / 스핀들 >:	포지셔닝할 축 / 스핀들
BP():	포지셔닝을 위한 키워드
< 위치 >:	파라미터 1 상수 또는 변수인 축 / 스핀들 위치
< 위치 속성 >:	파라미터 2 위치 특성 (예를 들어 변수 (예 : $P_SUB_SPOSMODE) 또는 키워드 (IC, AC, ...) 인 위치 복귀 모드)

변수에서 제공된 값에는 다음 의미가 포함됩니다 .

값	의미	허용 가능한 값 :
0	위치 특성에 대해 변경 사항 없음	
1	AC	POS, POSA,SPOS, SPOSA,AX, 축 어드레스
2	IC	POS, POSA,SPOS, SPOSA,AX, 축 어드레스
3	DC	POS, POSA,SPOS, SPOSA,AX, 축 어드레스
4	ACP	POS, POSA,SPOS, SPOSA,AX, 축 어드레스
5	ACN	POS, POSA,SPOS, SPOSA,AX, 축 어드레스
6	OC	-
7	PC	-
8	DAC	POS, POSA,AX, 축 어드레스
9	DIC	POS, POSA,AX, 축 어드레스
10	RAC	POS, POSA,AX, 축 어드레스
11	RIC	POS, POSA,AX, 축 어드레스
12	CAC	POS, POSA
13	CIC	POS, POSA
14	CDC	POS, POSA
15	CACP	POS, POSA
16	CACN	POS, POSA

예제

1 차 스핀들 S1 및 2 차 스핀들 S2 사이 활성 동기 스핀들 커플링의 경우 , 스핀들 위치를 지정하기 위한 다음 교체 싸이클이 메인 프로그램의 SPOS 명령을 사용하여 호출됩니다 .

포지셔닝은 N2230 의 명령을 사용하여 실행됩니다 .
SPOS[1]=GP($P_SUB_SPOSIT,$P_SUB_SPOSMODE)
SPOS[2]=GP($P_SUB_SPOSIT,$P_SUB_SPOSMODE)

복귀할 위치는 시스템 변수 $P_SUB_SPOSIT 로부터 판독되고 위치 복귀 모드는 시스템 변수 $P_SUB_SPOSMODE 로부터 판독됩니다 .

프로그램 코드	설명
N1000 PROC LANG_SUB DISPLOF SBLOF	
...	
N2100 IF($P_SUB_AXFCT==2)	
N2110	; 활성 동기 스핀들 커플링에 대한 SPOS/SPOSA/ M19 명령 교체
N2185 DELAYFSTON	; 시작 정지 지연 영역
N2190 COUPOF(S2,S1)	; 동기 스핀들 커플링 해제
N2200	; 1 차 및 2 차 스핀들 포지셔닝
N2210 IF($P_SUB_SPOS==TRUE) OR ($P_SUB_SPOSA==TRUE)	
N2220	; SPOS 를 사용하여 스핀들 포지셔닝 :
N2230 SPOS[1]=GP($P_SUB_SPOSIT,$P_SUB_SPOSMODE)	
SPOS[2]=GP($P_SUB_SPOSIT,$P_SUB_SPOSMODE)	
N2250 ELSE	
N2260	; M19 를 사용하여 스핀들 포지셔닝 :
N2270 M1=19 M2=19	; 1 차 및 2 차 스핀들 포지셔닝
N2280 ENDIF	
N2285 DELAYFSTOF	; 정지 지연 영역의 끝
N2290 COUPON(S2,S1)	; 동기 스핀들 커플링 활성화
N2410 ELSE	
N2420	; 추가 교체에 대한 쿼리
...	
N3300 ENDIF	
...	
N9999 RET	

보충 조건

- 위치 특성 간접 프로그램은 동기 동작에서 수행할 수 없습니다 .

참고 자료

기능 매뉴얼의 기본 기능 ; BAG, 채널 , 프로그램 작동 , 리셋 응답 (K1),
장 : 서브프로그램에 의한 NC 함수 교체

1.2.4 가공 프로그램 라인 간접 프로그램 (EXECSTRING)

기능

가공 프로그램 명령어 EXECSTRING을 사용하면 기존에 생성한 문자열 변수를 가공 프로그램 라인으로 실행할 수 있습니다 .

구문

EXECSTRING 은 개별 가공 프로그램 라인에서 프로그래밍됩니다 .
EXECSTRING (<string_variable>)

의미

EXECSTRING:	문자열 변수를 가공 프로그램 라인으로 실행시키는 명령어
< 문자열 변수 >:	실행할 실제 가공 프로그램 라인이 포함된 STRING 변수를 입력하십시오 .

주

EXECSTRING 을 사용하면 가공 프로그램의 프로그램 섹션에서 프로그래밍할 수 있는 모든 가공 프로그램 구문을 추출할 수 있습니다 . 다시 말해 PROC 및 DEF 명령은 제외됨을 의미하여 , 또한 INI 및 DEF 파일에서도 일반적으로 제외됩니다 .

예제

프로그램 코드	설명
N100 DEF STRING[100] BLOCK	; 실행할 가공 프로그램 라인을 적용할 문자열 변수 정의
N110 DEF STRING[10] MFCT1="M7"	
...	
N200 EXECSTRING(MFCT1 << "M4711")	; 가공 프로그램 라인 "M7 M4711" 실행
...	
N300 R10=1	
N310 BLOCK="M3"	
N320 IF(R10)	
N330 BLOCK = BLOCK << MFCT1	
N340 ENDIF	
N350 EXECSTRING(BLOCK)	; 가공 프로그램 라인 "M3 M7" 실행

1.3 산술 함수

기능

산술 함수는 주로 R 파라미터 및 REAL 유형의 변수 (또는 상수 및 함수) 에 사용됩니다 . 또한 유형 INT 및 CHAR 도 허용됩니다 .

연산자 / 산술 함수	의미
+	덧셈
-	뺄셈
*	곱셈
/	나눗셈 **주의** : (type INT)/(type INT)=(type REAL); 예 : 3/4 = 0.75
DIV	나눗셈 , 변수 유형 INT 및 REAL 의 경우 **주의** : (type INT)DIV(type INT)=(type INT); 예 : 3 DIV 4 = 0
MOD	모듈로 나눗셈 (유형 INT 에만 해당) 은 INT 나눗셈의 나머지를 제공합니다 . 예제 : 3 MOD 4 = 3
:	체인 연산자 (FRAME 변수)
Sin( )	사인
COS( )	코사인
TAN( )	접선
ASIN( )	아크 사인
ACOS( )	아크 코사인
ATAN2(,)	아크 탄젠트 2
SQRT( )	제곱근
ABS( )	절대치
POT( )	2. 거듭제곱
TRUNC( )	정수로 자름 비교 명령의 정확도는 TRUNC 를 사용하여 설정할 수 있습니다 (" 비교 오류에 대한 정밀도 수정 (TRUNC) [쪽 69]" 참조).
ROUND( )	정수로 반올림
LN( )	자연 로그
EXP( )	지수 함수
MINVAL ( )	두 변수 중 낮은 값 ("변수 최소, 최대 및 범위(MINVAL, MAXVAL 및 BOUND) [쪽 71]" 참조)
MAXVAL ( )	두 변수 중 높은 값 ("변수 최소, 최대 및 범위(MINVAL, MAXVAL 및 BOUND) [쪽 71]" 참조)

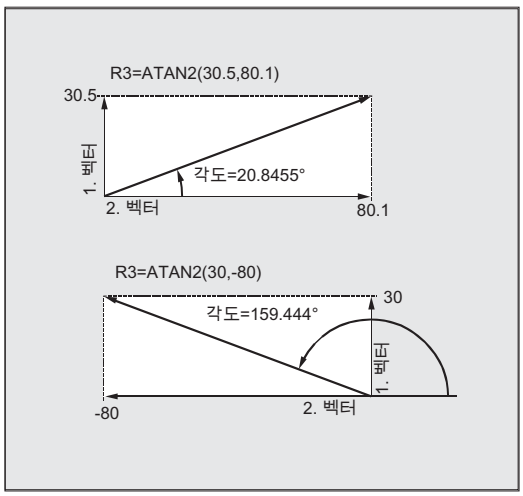
BOUND ( )	정의된 값 범위 내에서 변수 값 ("변수 최소, 최대 및 범위(MINVAL, MAXVAL 및 BOUND) [쪽 71]" 참조)
CTRANS( )	변환
CROT ( )	회전
CSCALE( )	축척 변경
CMIRROR( )	미러링

프로그래밍

산술 함수에는 일반적인 수학적 표기법이 사용됩니다 . 실행 우선 순위는 괄호 안에 표시됩니다 . 각도는 삼각 함수 및 역 함수로 지정됩니다 (직각 = 90°).

예제

예제 1: ATAN2



산술 함수 ATAN2 는 서로 직각을 이루는 두 개의 벡터에서 총 벡터의 각도를 계산합니다 .
결과는 네 개의 4 분원 중 하나에 위치합니다 (-180° < 0 < +180°).
항상 각도 참조는 양의 방향으로 두 번째 값에 기반합니다 .

예제 2: 전체 변수 배열 초기화

프로그램 코드	설명
R1=R1+1	; New R1 = old R1 +1
R1=R2+R3 R4=R5-R6 R7=R8*R9	
R10=R11/R12 R13=SIN(25.3)	
R14=R1*R2+R3	; 곱셈 또는 나눗셈은 덧셈 또는 뺄셈보다 우선합니다 .
R14=(R1+R2)*R3	; 괄호를 먼저 계산합니다 .
R15=SQRT(POT(R1)+POT(R2))	; 괄호 안을 먼저 계산합니다 . R15 = (R1+R2) 의 제곱근
RESFRAME=FRAME1:FRAME2 FRAME3=CTRANS(…):CROT(…)	; 연결 연산자는 프레임을 연결하여 결과 프레임을 구성하거나 값을 프레임 구성 요소에 지정합니다 .

1.4 비교 및 논리 연산자

작동

비교 연산은 예를 들어 점프 조건을 공식으로 작성하는 데 사용할 수 있습니다 . 복잡한 표현식도 비교할 수 있습니다 .

비교 연산은 CHAR, INT, REAL 및 BOOL 유형의 변수에 적용 가능합니다 . 코드 값을 CHAR 유형과 비교합니다 .
STRING, AXIS 및 FRAME 유형의 경우 다음이 가능합니다. == 및 <>, 동기식 작동을 포함하여 STRING 유형 연산에서 사용할 수 있습니다 .

비교 연산의 결과는 항상 BOOL 유형입니다 .

논리 연산은 참 / 거짓 값을 연결하는 데 사용됩니다 .

논리 연산은 유형 BOOL 변수에만 적용할 수 있습니다 . 그러나 또한 내부 유형 변환을 통해 CHAR, INT 및 REAL 데이터 유형에도 적용할 수 있습니다 .

논리 (부울) 연산의 경우 다음이 BOOL, CHAR,INT 및 REAL 데이터 유형에 적용됩니다 .

- 0: FALSE
- 0 이외 값 : TRUE

비트 논리 연산자

또한 논리 연산을 단일 비트의 CHAR 및 INT 유형에 적용할 수 있습니다 . 유형 변환은 자동으로 수행됩니다 .

프로그래밍

관계 연산자	의미
==	같음
<>	같지 않음
>	보다 큼
<	보다 작음
>=	보다 크거나 같음
<=	작거나 같음

논리 연산자	의미
AND	AND
또는	또는
NOT	부정
XOR	OR 제외

비트별 논리 연산자	의미
B_AND	Bit-serial AND
B_OR	Bit-serial OR
B_NOT	Bit-serial 부정
B_XOR	Bit-serial OR 제외

주

산술 표현식에서 모든 연산자의 실행 순서는 괄호로 지정하여 일반 우선 순위 규칙을 재지정할 수 있습니다 .

주

부울 피연산자와 연산자 사이에 공백이 있어야 합니다 .

주

연산자 B_NOT 만 단일 피연산자를 의미합니다 . 이 피연산자는 연산자 뒤에 위치합니다 .

예

예제 1: 관계 연산자

```
IF R10>=100 GOTOF DEST
```

또는

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

R10>=100 비교 결과가 먼저 R11 에 버퍼링됩니다 .

예제 2: 논리 연산자

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

또는

```
IF NOT R10 GOTOB START
```

NOT 만 단일 피연산자를 의미합니다 .

예제 3: 비트별 논리 연산자

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

1.5 비교 오류에 대한 정밀도 수정 (TRUNC)

기능

TRUNC 명령을 수행하면 정밀도 인수에 의해 곱해진 피연산자를 자릅니다 .

비교 명령에 대해 설정 가능한 정밀도

유형 REAL 의 프로그램 데이터가 내부적으로 64 비트의 IEEE 형식으로 표시됩니다 . 이 표시 형식을 통해 십진수 값이 부정확하게 표시될 수 있으며 이로 인해 이론적으로 계산된 값과 비교할 때 예기치 않은 결과가 나올 수 있습니다 .

상대 등식

프로그램 흐름을 간섭하는 표시 형식으로 인해 발생되는 부정확도를 방지하기 위해 절대 등식이 아닌 상대 등식을 사용하여 비교합니다 .

구문

비교 에러에 대한 정밀도 수정

TRUNC (R1*1000)

의미

TRUNC: 십진수 자리를 자릅니다 .

다음을 고려한 10^{-12} 의 상대 품질

- 등식 : (==)
- 부등식 : (<>)
- 보다 크거나 같음 : (>=)
- 보다 작거나 같음 : (<=)
- 보다 크거나 / 작음 : 절대 품질을 가진 (><)
- 보다 큼 : (>)
- 보다 작음: (<)

호환성

호환성의 이유로 (>) 및 (<) 의 상대 품질 확인은 머신 데이터 MD10280 $MN_ PROG_FUNCTION_MASK Bit0 = 1 을 설정하여 비활성화할 수 있습니다 .

주

유형 REAL 데이터와의 비교는 위에 설명한 이유로 인해 특정 부정확도가 발생됩니다. 적용할 수 없는 편차가 발생할 경우 피연산자를 정밀도 인자와 곱하고 TRUNC 로 자르는 INTEGER 계산을 사용합니다 .

동기 동작

비교 명령에 대해 설명한 응답을 동기 동작에도 적용합니다 .

예제

예제 1: 정밀도 고려 사항

프로그램 코드	설명
N40 R1=61.01 R2=61.02 R3=0.01	; 초기값 지정
N41 IF ABS(R2-R1) > R3 GOTOF ERROR	; 지금까지 점프가 실행되었음
N42 M30	; 프로그램 끝
N43 ERROR: SETAL(66000)	;
R1=61.01 R2=61.02 R3=0.01	; 초기값 지정
R11=TRUNC(R1*1000) R12=TRUNC(R2*1000) R13=TRUNC(R3*1000)	; 정확도 수정
IF ABS(R12-R11) > R13 GOTOF ERROR	; 점프가 더 이상 실행되지 않음
M30	; 프로그램 끝
ERROR: SETAL(66000)	;

예제 2: 두 피연산자의 몫 계산 및 평가

프로그램 코드	설명
R1=61.01 R2=61.02 R3=0.01	; 초기값 지정
IF ABS((R2-R1)/R3)-1) > 10EX-5 GOTOF ERROR	; 점프가 실행되지 않음
M30	; 프로그램 끝
ERROR: SETAL(66000)	;

1.6 변수 최소 , 최대 및 범위 (MINVAL, MAXVAL 및 BOUND)

기능

MINVAL 및 MAXVAL 명령은 2 개 변수의 값을 비교할 때 사용합니다. MINVAL 의 경우 더 작은 값이 , MAXVAL 의 경우 더 큰 값이 결과로 출력됩니다 .

BOUND 명령은 테스트 변수의 값이 정의된 값 범위 내에 있는지 테스트하는 데 사용됩니다 .

구문

< 더 작은 값 >=MINVAL(< 변수 1>,< 변수 2>)
< 더 큰 값 >=MAXVAL(< 변수 1>,< 변수 2>)
< 반환 값 >=<BOUND>(< 최소 >,< 최대 >,< 테스트 변수 >)

의미

MINVAL:	2 개의 변수 (< 변수 1>, < 변수 2>) 중 **더 작은 값**을 가져옵니다 .
< 더 작은 값 >:	MINVAL 명령의 결과 변수 더 작은 변수 값으로 설정합니다 .
MAXVAL:	2 개의 변수 (< 변수 1>, < 변수 2>) 중 **더 큰 값**을 가져옵니다 .
< 더 큰 값 >:	MAXVAL 명령의 결과 변수 더 큰 변수 값으로 설정합니다 .
BOUND:	변수 (<테스트 변수) 가 정의된 값 범위 내에 있는지 테스트합니다.
< 최소 >:	값 범위의 최소값을 정의하는 변수 .
< 최대 >:	값 범위의 최대값을 정의하는 변수 .
< 반환값 >:	BOUND 명령의 결과 변수 테스트 변수의 값이 정의된 값 범위 내에 있는 경우 테스트 변수의 값이 결과 변수로 설정됩니다 . 테스트 변수의 값이 정의된 값 범위보다 클 경우 정의된 값 범위의 최대값이 결과 변수로 설정됩니다 . 테스트 변수의 값이 정의된 값 범위보다 작을 경우 정의된 값 범위의 최소값이 결과 변수로 설정됩니다 .

주

MINVAL, MAXVAL 및 BOUND 는 동기 동작에도 프로그래밍할 수 있습니다 .

주

값이 같은 경우 , 동작

두 값이 같은 경우 MINVAL/MAXVAL이 모두 이 값으로 설정됩니다. BOUND의 경우 테스트할 변수의 값이 다시 반환됩니다 .

예제

프로그램 코드	설명
DEF REAL rVar1=10.5, rVar2=33.7, rVar3, rVar4, rVar5, rValMin, rValMax, rRetVar	
rValMin=MINVAL(rVar1,rVar2)	; rValMin 을 값 10.5 로 설정합니다 .
rValMax=MAXVAL(rVar1,rVar2)	; rValMax 을 값 33.7 로 설정합니다 .
rVar3=19.7	
rRetVar=BOUND(rVar1,rVar2,rVar3)	; rVar3 이 범위 내에 있습니다 . rRetVar 이 19.7 로 설정됩니다 .
rVar3=1.8	
rRetVar=BOUND(rVar1,rVar2,rVar3)	; rVar3 이 최소값보다 작습니다 . rRetVar 는 10.5 로 설정됩니다 .
rVar3=45.2	
rRetVar=BOUND(rVar1,rVar2,rVar3)	; rVar3 이 최대값보다 큽니다 . rRetVar 는 33.7 로 설정됩니다 .

1.7 연산 우선 순위

기능

각 연산자에게 우선 순위가 할당됩니다 . 표현식을 평가할 때 최상위 우선 순위를 가진 연산자가 항상 먼저 적용됩니다 . 연산자 간에 동일한우선 순위를 가진 경우 왼쪽부터 오른쪽으로 평가가 이루어집니다 .

산술 표현식에서 , 모든 연산자의 실행 순서는 괄호로 지정하여 일반 우선 순위 규칙을 재지정할 수 있습니다 .

연산자 순서

최상위부터 최하위 우선 순위

1.	NOT, B_NOT	부정 , Bit-serial 부정
2.	*, /, DIV, MOD	곱셈 , 나눗셈
3.	+, –	덧셈 , 뺄셈
4.	B_AND	Bit AND
5.	B_XOR	Bit-serial 익스클루시브 OR
6.	B_OR	Bit-serial OR
7.	AND	AND
8.	XOR	익스클루시브 OR
9.	또는	또는
10.	<<	문자열 연결 , 결과 유형 STRING
11.	==, <>, >, <, >=, <=	비교 연산자

주

프레임의 연결 연산자 ":"는 다른 연산자와 동일한 표현식에서 사용할 수 없습니다. 따라서 이 연산자의 경우 우선 순위가 필요 없습니다 .

예제 : IF 문

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

1.8 가능한 유형 변환

작동

할당 시 유형 변환

일정한 숫자 값, 변수 또는 변수에 할당된 표현식은 변수 유형과 비교 가능해야 합니다. 이러한 경우 유형은 값 할당 시 자동으로 변환됩니다 .

가능한 유형 변환

to	REAL	INT	BOOL	CHAR	STRING	AXIS	FRAME
변경 전							
REAL	yes	예 *	예 1)	예 *	–	–	–
INT	yes	yes	예 1)	예 2)	–	–	–
BOOL	yes	yes	yes	yes	yes	–	–
CHAR	yes	yes	예 1)	yes	yes	–	–
STRING	–	–	예 4)	예 3)	yes	–	–
AXIS	–	–	–	–	–	yes	–
FRAME	–	–	–	–	–	–	yes

설명

* REAL 에서 INT 유형으로 변환 시 >=0.5 의 분수 값은 반올림되고 그 이외 값은 반내림됩니다 (ROUND 함수 참조).

1) 값 <> 0 은 TRUE 에 해당하고 값 == 0 은 FALSE 에 해당

2) 값이 허용 가능한 범위 내에 있는 경우

3) 1 문자인 경우

4) 문자열 길이 0 = >FALSE, 그렇지 않은 경우 TRUE

주

변환으로 인해 대상 범위보다 큰 값이 산출된 경우 오류 메시지가 표시됩니다 .

표현식에 혼합된 유형이 있는 경우 유형 변환이 자동으로 이루어집니다 . 또한 동기식 작동에서도 유형 변환이 가능합니다 . " 이동 - 동기식 작동 , 함축된 유형 변환 " 장을 참조하십시오 .

1.9 문자열 연산

문자열 연산

" 할당 " 및 " 비교 " 이외에 다음과 같은 문자열 동작을 사용할 수 있습니다

- STRING 으로 유형 변환 (AXSTRING) [쪽 76]
- STRING 에서 유형 변환 (NUMBER, ISNUMBER, AXNAME) [쪽 77]
- 문자열 연결 (<<) [쪽 78]
- 소문자 / 대문자로 변환 (TOLOWER, TOUPPER) [쪽 79]
- 문자열 길이 결정 (STRLEN) [쪽 80]
- 문자열에서 문자 / 문자열 검색 (INDEX, RINDEX, MINDEX, MATCH) [쪽 81]
- 하위 문자열 선택 (SUBSTR) [쪽 82]
- 단일 문자 선택 (STRINGVAR, STRINGFELD) [쪽 83]
- 문자열 포맷 (SPRINT) [쪽 84]

0 문자의 특별한 의미

내부적으로 0 문자는 문자열의 끝 식별자로 해석됩니다 . 특정 문자를 0 으로 바꿀 경우 문자열이 잘립니다 .

예제 :

프로그램 코드	코멘트
DEF STRING[20] STRG="axis . stationary"	
STRG[6]="X"	
MSG(STRG)	; 메시지 "axis X stationary" 가 표시됩니다 .
STRG[6]=0	
MSG(STRG)	; 메시지 "axis" 가 표시됩니다 .

1.9.1 STRING 으로 유형 변환 (AXSTRING)

기능

"STRING 으로 형식 변환 " 기능을 사용하여 다른 형식의 변수를 메시지 (MSG) 명령의 컴포넌트로 사용할 수 있습니다 .

연산자 <<를 사용하는 경우 암시적으로 데이터 유형 INT, REAL, CHAR 및 BOOL에 대해 유형 변환이 수행됩니다 (" 문자열 연결 (<<) [쪽 78]" 참조).

INT 값은 일반 읽기 가능한 형식으로 변환됩니다 . REAL 값은 최대 10 개의 십진수 자리로 변환됩니다 .

Type AXIS 변수는 AXSTRING 명령을 사용하여 STRING 으로 변환할 수 있습니다 .

구문

<STRING_ERG> = << < 모든 _ 유형 >
<STRING_ERG> = AXSTRING(< 축 이름 >)

의미

<STRING_ERG>:	유형 변환 결과에 대한 변수 유형 : STRING
< 모든 _ 유형 >:	변수 유형 INT, REAL, CHAR, STRING 및 BOOL
AXSTRING:	AXSTRING 명령을 실행하면 지정된 축 식별자가 문자열로 제공됩니다 .
< 축 이름 >:	축 식별자의 변수 유형 : AXIS

주

FRAME 변수는 변환할 수 없습니다 .

예제

예제 1:
MSG("Position:"<<$AA_IM[X])

예제 2: AXSTRING

프로그램 코드	설명
DEF STRING[32] STRING_ERG	
STRING_ERG=AXSTRING(X)	; STRING_ERG == "X"

1.9.2 STRING 에서 유형 변환 (NUMBER, ISNUMBER, AXNAME)

기능

NUMBER 명령을 사용하여 STRING 에서 REAL 로 변환이 이루어집니다 . 변환 가능 여부는 ISNUMBER 명령을 사용하여 확인할 수 있습니다 .

문자열은 AXNAME 명령을 사용하여 축 데이터 유형으로 변환됩니다 .

구문

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

의미

NUMBER:	NUMBER 명령을 실행하면 <string> 으로 표현된 값을 REAL 값으로 반환합니다 .
< 문자열 >:	변환할 STRING 유형 변수
<REAL_ERG>:	NUMBER 를 사용한 유형 변환 결과에 대한 변수 유형 : REAL
ISNUMBER:	ISNUMBER 명령을 사용하여 <string>을 유효한 값으로 변환할 수 있는지 여부를 확인할 수 있습니다 .
<BOOL_ERG>:	ISNUMBER 를 사용한 조회 결과에 대한 변수 유형 : BOOL 값 : TRUE — ISNUMBER는 <string>이 언어 규칙을 준수하여 유효한 REAL 값을 표시하는 경우 TRUE 값을 반환합니다 . FALSE — ISNUMBER는 동일한 <string>으로 NUMBER를 호출할 경우 FALSE 값을 반환하고 알람이 실행됩니다 .
AXNAME:	AXNAME 명령을 실행하면 지정된 <string> 이 축 식별자로 변환됩니다 . **참고 :** <string> 에 구성된 축 식별자를 할당할 수 없는 경우 알람이 실행됩니다 .
<AXIS_ERG>:	AXNAME 를 사용한 유형 변환 결과에 대한 변수 유형 : AXIS

예제

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

1.9.3 문자열 연결 (<<)

기능

" 문자열 연결 " 기능을 사용하여 개별 요소로부터 문자열을 구성할 수 있습니다 .

연결은 연산자 "<<" 를 사용하여 수행됩니다 . 이 연산자에는 기본 유형 CHAR, BOOL, INT, REAL 및 STRING 의 모든 조합에 대한 대상 유형으로 STRING 이 포함되어 있습니다 . 필요한 경우 변환이 기존 규칙에 따라 수행됩니다 .

구문

<any_type> << <any_type>

의미

< 모든 _ 유형 >: 변수 , 유형 CHAR, BOOL, INT, REAL 또는 STRING

<< : 문자열 (유형 STRING) 을 구성하기 위한 체인 변수 (<any_type>) 에 대한 연산자입니다 .
또한 이 연산자는 소위 "unary" 변수로 독립적으로 사용할 수 있습니다 . 이 연산자는 STRING (FRAME 및 AXIS 의 경우 해당 없음) 에 대한 명시적 유형 변환기에 사용할 수 있습니다 .
<< <any_type>

예를 들어 , 문자 목록 및 파라미터로부터 구성될 수 있는 메시지 또는 명령을 삽입할 수 있습니다 (예를 들어 블록 이름).
MSG(STRG_TAB[LOAD_IDX]<<BAUSTEIN_NAME)

주의
문자열 연결의 중간 결과가 최대 문자열 길이를 초과하지 않아야 합니다 .

주

FRAME 및 AXIS 유형을 연산자 "<<" 와 함께 사용할 수 없습니다 .

예제

예제 1: 문자열 연결

프로그램 코드	설명
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<<"/value:"<<VALUE)	; 표시 : "Index:2/value:9.654"
NO_MSG:	

예제 2: << 을 통한 명시적 유형 변환

프로그램 코드	설명
DEF REAL VALUE=3.5	
<<VALUE	; 지정된 유형 REAL 변수가 STRING 유형으로 변환됩니다 .

1.9.4 소문자 / 대문자로 변환 (TOLOWER, TOUPPER)

기능

"소문자/대문자로 변환" 기능을 통해 문자열의 모든 문자를 표준 표현으로 변환할 수 있습니다.

구문

<STRING_ERG>=TOUPPER("<string>")
<STRING_ERG>=TOLOWER("<string>")

의미

TOUPPER:	TOUPPER 명령을 사용하여 문자열의 모든 문자를 **대문자**로 변환합니다 .
TOLOWER:	TOLOWER 명령을 사용하여 문자열의 모든 문자를 **소문자**로 변환합니다 .
< 문자열 >:	변환할 문자열 유형 : STRING
<STRING_ERG>:	변환 결과에 대한 변수 유형 : STRING

예제

사용자가 운영자 인터페이스에서 입력할 수 있기 때문에 이러한 입력 항목이 주어진 표준 문자 (대문자 또는 소문자) 일 수 있습니다 .

프로그램 코드

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

1.9.5 문자열 길이 결정 (STRLEN)

기능

STRLEN 명령을 사용하여 문자열 길이를 결정할 수 있습니다 .

구문

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

의미

STRLEN:	STRLEN 명령은 지정된 문자열 길이를 결정합니다 . 0 이 아닌 문자 수가 문자열 시작부터 계산하여 반환됩니다 .
< 문자열 >:	길이를 결정할 문자열 유형 : STRING
<INT_ERG>:	결정 결과에 대한 변수 유형 : INT

예제

단일 문자 액세스와 함께 이 함수를 사용하여 문자열 길이를 결정할 수 있습니다 .

프로그램 코드

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

1.9.6 문자열에서 문자 / 문자열 검색 (INDEX, RINDEX, MINDEX, MATCH)

기능

이 기능을 통해 문자열 내에서 단일 문자 또는 문자열을 검색합니다 . 기능 결과에서 검색된 문자열에 문자 / 문자열이 배치되는 위치를 지정합니다 .

구문

INT_ERG=INDEX(STRING,CHAR) ; **결과 유형 : INT**

INT_ERG=RINDEX(STRING,CHAR) ; **결과 유형 : INT**

INT_ERG=MINDEX(STRING,STRING) ; **결과 유형 : INT**

INT_ERG=MATCH(STRING,STRING) ; **결과 유형 : INT**

기호론

검색 기능 : 검색을 성공적으로 수행한 문자열 (첫 번째 파라미터) 의 위치를 제공합니다 . 문자 / 문자열을 찾을 수 없는 경우 값 -1 이 반환됩니다 . 첫 번째 문자의 위치는 0 입니다 .

의미

INDEX: 첫 번째 파라미터에서 , 두 번째 파라미터로 지정된 문자를 시작부터 검색합니다 .

RINDEX: 첫 번째 파라미터에서 , 두 번째 파라미터로 지정된 문자를 끝에서 검색합니다 .

MINDEX: 첫 번째로 찾은 문자의 인덱스가 반환되는 문자 목록이 문자열로 전달되는 경우를 제외하고 INDEX 함수에 해당합니다 .

MATCH: 문자열에서 문자열을 검색합니다 .

이를 통해 예를 들어 비어 있거나 경로 구분자 ("/") 가 있는 위치처럼 특정 기준에 따라 문자열을 분할할 수 있습니다 .

예제

입력을 경로 및 블록 이름으로 분할

프로그램 코드		설명
DEF INT PFADIDX, PROGIDX		
DEF STRING[26] INPUT		
DEF INT LISTIDX		
INPUT = "/_N_MPF_DIR/_N_EXECUTE_MPF"		
LISTIDX = MINDEX (INPUT, "M,N,O,P") + 1	;	"N"은 처음부터 시작하는 선택 목록의 파라미터 INPUT에서 첫 번째 문자이기 때문에 LISTIDX 에 반환되는 값은 3 입니다 .
PFADIDX = INDEX (INPUT, "/") +1	;	따라서 다음이 적용됩니다 . PFADIDX = 1
PROGIDX = RINDEX (INPUT, "/") +1	;	따라서 다음이 적용됩니다 . PROGIDX = 12
		다음 섹션에서 소개하는 SUBSTR 기능을 사용하여 구성 요소 " 경로 " 및 " 모듈 " 의 변수 INPUT 을 분할할 수 있습니다 .
VARIABLE = SUBSTR (INPUT, PFADIDX, PROGIDX-PFADIDX-1)	;	"_N_MPF_DIR" 을 반환합니다 .
VARIABLE = SUBSTR (INPUT, PROGIDX)	;	"_N_EXECUTE_MPF" 를 반환합니다 .

1.9.7 하위 문자열 선택 (SUBSTR)

기능

이 기능으로 문자열에서 하위 문자열을 추출합니다 . 이를 위해 첫 번째 문자의 인덱스 및 원하는 문자열 길이 (해당하는 경우) 가 지정됩니다 . 길이 정보를 지정하지 않을 경우 문자열 데이터가 남아 있는 문자열을 의미합니다 .

구문

STRING_ERG = SUBSTR (STRING,INT) ; **결과 유형 : INT**

STRING_ERG = SUBSTR(STRING,INT, INT) ; **결과 유형 : INT**

기호론

첫 번째 경우에서 , 두 번째 파라미터가 지정한 위치의 하위 문자열이 문자열 끝까지 반환됩니다 .

두 번째 경우에서 , 결과 문자열은 세 번째 파라미터가 지정한 최대 길이로 제한됩니다 .

초기 위치가 문자열 끝 다음에 오는 경우 빈 문자열 (" ") 이 반환됩니다 .

초기 위치 또는 길이가 음수인 경우 알람이 출력됩니다 .

예제

프로그램 코드		설명
DEF STRING[29] ERG		
ERG = SUBSTR ("ACK:10 to 99", 10, 2)	;	따라서 다음이 적용됩니다 . ERG == "10"

1.9.8 단일 문자 선택 (STRINGVAR, STRINGFELD)

기능

이 기능으로 문자열에서 단일 문자를 선택합니다 . 이 기능은 읽기 액세스 및 쓰기 액세스 작업에 모두 적용됩니다 .

구문

CHAR_ERG = STRINGVAR [IDX] ; **결과 유형 : CHAR**

CHAR_ERG = STRINGFELD [IDX_FELD, IDX_CHAR] ; **결과 유형 : CHAR**

기호론

문자열 내에서 지정된 위치의 문자를 읽거나 씁니다. 위치 파라미터가 음수이거나 문자열보다 클 경우 알람이 출력됩니다 .

메시지 예 :

축 식별자를 준비된 문자열에 삽입 .

프로그램 코드	설명
DEF STRING [50] MESSAGE = "Axis n has reached position"	
MESSAGE [6] = "X"	
MSG (MESSAGE)	; "axis X has reached position" 메시지를 반환합니다 .

파라미터

사용자 정의 변수 (LUD, GUD 및 PUD 데이터) 에만 단일 문자 액세스가 가능합니다 .

또한 이 유형의 액세스는 서브프로그램 호출 시 " 값별 호출 " 유형 파라미터에만 적용됩니다 .

예제

예제 1: 시스템 데이터 , 머신 데이터 등에 대한 단일 문자 액세스

프로그램 코드	설명
DEF STRING [50] STRG	
DEF CHAR ACK	
…	
STRG = $P_MMCA	
ACK = STRG [0]	; 인지 구성 요소 평가

예제 2: " 기준별 호출 " 파라미터로 단일 문자 액세스

프로그램 코드	설명
DEF STRING [50] STRG	
DEF CHAR CHR1	
EXTERN UP_CALL (VAR CHAR1)	; 기준별 호출 파라미터 !
…	
CHR1 = STRG [5]	
UP_CALL (CHR1)	; 기준별 호출
STRG [5] = CHR1	

1.9.9 문자열 포맷 (SPRINT)

기능

미리 정의된 SPRINT 기능을 사용하여 문자열을 포맷하고 외부 장치 출력을 준비할 수 있습니다 (" 외부 장치 / 파일로 출력 (EXTOPEN, WRITE, EXTCLOSE) [쪽 696]" 참조).

구문

"<Result_string>"=SPRINT("<Format_string>",<value_1>,<value_2>,..., <value_n>)

중요도

SPRINT:	STRING 유형의 값을 제공하는 사전 정의된 기능에 대한 식별자
"<Format_String>":	고정 요소 및 가변 요소를 포함한 문자열 . 가변 요소는 포맷 제어 문자 % 및 그 뒤에 나오는 포맷 설명을 사용하여 정의됩니다 .
<value_1>,<value_2>,…,<value_n>:	상수 또는 NC 변수 형식의 값 . 이 값은 n 번째 포맷 제어 문자 % 의 위치에 삽입되며 <format_string> 에 나오는 포맷 설명에 해당합니다 .
"<result_string>":	포맷된 문자열 (최대 400 바이트)

사용 가능한 포맷 설명

%B:	변환할 값이 다음과 같은 경우 "TRUE" 문자열로 변환 : • 0 과 같지 않은 경우 • 빈 문자열이 아닌 경우 (문자열 값의 경우) 변환할 값이 다음과 같은 경우 "FALSE" 문자열로 변환 : • 0 과 같은 경우 • 빈 문자열인 경우 **예 :** N10 DEF BOOL BOOL_VAR=1 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF BOOL_VAR:%B", BOOL_VAR) 결과 : 문자열 "CONTENT OF BOOL_VAR:TRUE" 가 RESULT 문자열 변수에 쓰여집니다 .
%C:	ASCII 문자로 변환 **예 :** N10 DEF CHAR CHAR_VAR="X" N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF CHAR_VAR:%C",CHAR_VAR) 결과: 문자열 "CONTENT OF CHAR_VAR:X"가 RESULT 문자열 변수에 쓰여집니다.
%D:	정수 값 (INTEGER) 을 포함한 문자열로 변환 **예 :** N10 DEF INT INT_VAR=123 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF INT_VAR:%D",INT_VAR) 결과: 문자열 "CONTENT OF INT_VAR:123" 이 RESULT 문자열 변수에 쓰여집니다.
%<m>D:	정수 값 (INTEGER) 을 포함한 문자열로 변환 문자열의 최소 길이는 <m> 글자입니다 . 누락된 자리는 공백으로 채워지고 왼쪽으로 정렬됩니다 . **예 :** N10 DEF INT INT_VAR=-123 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF INT_VAR:%6D",INT_VAR) 결과 : 문자열 "CONTENT OF INT_VAR:xx-123" 이 RESULT 문자열 변수에 쓰여집니다 . (예제의 "x" 는 공백을 나타냅니다 .)
%F:	소수 부분이 6 자리인 소수를 포함한 문자열로 변환 . 적절한 경우에는 소수 부분이 반올림되거나 0 으로 채워집니다 . **예 :** N10 DEF REAL REAL_VAR=-1.2341234EX+03 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%F",REAL_VAR) 결과: 문자열 "CONTENT OF REAL_VAR:-1234.123400" 이 RESULT 문자열 변수에 쓰여집니다 .
%<m>F:	소수 부분이 6 자리이고 총 길이가 <m> 자 이상인 소수를 포함한 문자열로 변환 . 적절한 경우에는 소수 부분이 반올림되거나 0 으로 채워집니다 . 누락된 글자는 공백을 사용하여 총 길이 <m> 까지 채워지며 , 좌측으로 정렬됩니다 . **예 :** N10 DEF REAL REAL_VAR=-1.23412345678EX+03 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%15F",REAL_VAR) 결과 : 문자열 "CONTENT OF REAL_VAR: xx-1234.123457" 이 RESULT 문자열 변수에 쓰여집니다 . (예제의 "x" 는 공백을 나타냅니다)

%.<n>F:	소수 부분이 <n>자리인 소수를 포함한 문자열로 변환. 적절한 경우에는 소수 부분이 반올림되거나 0 으로 채워집니다 . **예 :** N10 DEF REAL REAL_VAR=-1.2345678EX+03 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%.3F",REAL_VAR) 결과 : 문자열 "CONTENT OF REAL_VAR:-1234.568" 이 RESULT 문자열 변수에 쓰여집니다 .
%<m>.<n>F:	소수 부분이 <n> 자리이고 총 길이가 <m> 자 이상인 소수를 포함한 문자열로 변환 . 적절한 경우에는 소수 부분이 반올림되거나 0 으로 채워집니다 . 누락된 글자는 공백을 사용하여 총 길이 <m> 까지 채워지며 , 좌측으로 정렬됩니다 . **예 :** N10 DEF REAL REAL_VAR=-1.2341234567890EX+03 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%10.2F",REAL_VAR) 결과 : 문자열 "CONTENT OF REAL_VAR:xx-1234.12" 가 RESULT 문자열 변수에 쓰여집니다 . (예제의 "x" 는 공백을 나타냅니다 .)
%E:	지수로 표현되는 소수를 포함한 문자열로 변환. 가수는 정수 부분 1자리와 소수 부분 6자리로 정규화되어 저장됩니다. 적절한 경우에는 소수 부분이 반올림되거나 0으로 채워집니다 . 지수는 키워드 "EX" 로 시작됩니다 . 그 다음에는 부호 ("+" 또는 "-") 및 2 자리 수 또는 3 자리 수가 옵니다 . **예 :** N10 DEF REAL REAL_VAR=-1234.567890 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%E",REAL_VAR) 결과 : 문자열 "CONTENT OF REAL_VAR:-1.234568EX+03" 이 RESULT 문자열 변수에 쓰여집니다 .
%<m>E:	지수로 표현되고 총 길이가 <m> 자 이상인 소수를 포함한 문자열로 변환 . 누락된 글자는 공백으로 채워지고 왼쪽으로 정렬됩니다 . 가수는 정수 부분 1 자리와 소수 부분 6 자리로 정규화되어 저장됩니다 . 적절한 경우에는 소수 부분이 반올림되거나 0 으로 채워집니다 . 지수는 키워드 "EX" 로 시작됩니다 . 그 다음에는 부호 ("+" 또는 "-") 및 2 자리 수 또는 3 자리 수가 옵니다 . **예 :** N10 DEF REAL REAL_VAR=-1234.5 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%20E",REAL_VAR) 결과 : 문자열 "CONTENT OF REAL_VAR:xxxxxx-1.234500EX+03" 이 RESULT 문자열 변수에 쓰여집니다 . (예제의 "x" 는 공백을 나타냅니다 .)
%.<n>E:	지수로 표현되는 소수를 포함한 문자열로 변환. 가수는 정수 부분 1자리와 소수 부분 <n> 자리로 정규화되어 저장됩니다 . 적절한 경우에는 소수 부분이 반올림되거나 0 으로 채워집니다 . 지수는 키워드 "EX" 로 시작됩니다 . 그 다음에는 부호 ("+" 또는 "-") 및 2 자리 수 또는 3 자리 수가 옵니다 . **예 :** N10 DEF REAL REAL_VAR=-1234.5678 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%.2E",REAL_VAR) 결과 : 문자열 "CONTENT OF REAL_VAR:-1.23EX+03" 이 RESULT 문자열 변수에 쓰여집니다 .

%<m>.<n>E:	지수로 표현되고 총 길이가 <m> 자 이상인 소수를 포함한 문자열로 변환 . 누락된 글자는 공백으로 채워지고 왼쪽으로 정렬됩니다 . 가수는 정수 부분 1 자리와 소수 부분 <n> 자리로 정규화되어 저장됩니다 . 적절한 경우에는 소수 부분이 반올림되거나 0 으로 채워집니다 . 지수는 키워드 "EX" 로 시작됩니다 . 그 다음에는 부호 ("+" 또는 "-") 및 2 자리 수 또는 3 자리 수가 옵니다 . **예 :** N10 DEF REAL REAL_VAR=-1234.5678 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%12.2E", REAL_VAR) 결과: 문자열 "CONTENT OF REAL_VAR:xx-1.23EX+03"이 RESULT 문자열 변수에 쓰여집니다 . (예제의 "x" 는 공백을 나타냅니다 .)
%G:	값 범위에 따라 소수 또는 지수로 표현되는 소수를 포함한 문자열로 변환 . 표현할 절대값이 1.0EX-04 보다 작거나 1.0EX+06 보다 크거나 같으면 지수 표기법이 선택되고 , 그렇지 않으면 소수 표기법이 선택됩니다 . 최대 6 개 유효 자릿수가 표시되거나 , 필요한 경우에는 반올림됩니다 . **소수 표기의 예 :** N10 DEF REAL REAL_VAR=1.234567890123456EX-04 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%G",REAL_VAR) 결과 : 문자열 "CONTENT OF REAL_VAR:0.000123457" 이 RESULT 문자열 변수에 쓰여집니다 . **지수 표기의 예 :** N10 DEF REAL REAL_VAR=1.234567890123456EX+06 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%G",REAL_VAR) 결과 : 문자열 "CONTENT OF REAL_VAR:1.23457EX+06" 이 RESULT 문자열 변수에 쓰여집니다 .
%<m>G:	값 범위에 따라 소수 또는 지수로 표현되는 소수를 포함한 문자열로 변환 (예 : %G). 문자열의 최소 길이는 <m> 글자입니다 . 누락된 글자는 공백으로 채워지고 왼쪽으로 정렬됩니다 . **소수 표기의 예 :** N10 DEF REAL REAL_VAR=1.234567890123456EX-04 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%15G",REAL_VAR) 결과 : 문자열 "CONTENT OF REAL_VAR:xxxx0.000123457" 이 RESULT 문자열 변수에 쓰여집니다 . (예제의 "x" 는 공백을 나타냅니다 .) **지수 표기의 예 :** N10 DEF REAL REAL_VAR=1.234567890123456EX+06 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%15G",REAL_VAR) 결과 : 문자열 "CONTENT OF REAL_VAR:xxx1.23457EX+06" 이 RESULT 문자열 변수에 쓰여집니다 . (예제의 "x" 는 공백을 나타냅니다 .)

%.<n>G:	값 범위에 따라 소수 또는 지수로 표현되는 소수를 포함한 문자열로 변환 . 최대 <n> 개 유효 자릿수가 표시되거나 , 필요한 경우에는 반올림됩니다 . 표현할 절대값이 1.0EX-04 보다 작거나 1.0EX(+<n>) 보다 크거나 같으면 지수 표기법이 선택되고 , 그렇지 않으면 소수 표기법이 선택됩니다 . **소수 표기의 예 :** N10 DEF REAL REAL_VAR=1.234567890123456EX-04 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%.3G",REAL_VAR) 결과 : 문자열 "CONTENT OF REAL_VAR:0.000123" 이 RESULT 문자열 변수에 쓰여집니다 . **지수 표기의 예 :** N10 DEF REAL REAL_VAR=1.234567890123456EX+03 N20 DEF STRING[80] RESULT N30 RESULT = SPRINT("CONTENT OF REAL_VAR:%.3G",REAL_VAR) 결과 : 문자열 "CONTENT OF REAL_VAR:1.23EX+03" 이 RESULT 문자열 변수에 쓰여집니다 .
%<m>.<n>G:	값 범위에 따라 소수 또는 지수로 표현되는 소수를 포함한 문자열로 변환 (예 : %.<n>G). 문자열의 최소 길이는 <m>글자입니다. 누락된 글자는 공백으로 채워지고 왼쪽으로 정렬됩니다 . **소수 표기의 예 :** N10 DEF REAL REAL_VAR=1.234567890123456EX-04 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%12.4G",REAL_VAR) 결과: 문자열 "CONTENT OF REAL_VAR:xxx0.0001235"가 RESULT 문자열 변수에 쓰여집니다 . (예제의 "x" 는 공백을 나타냅니다 .) **지수 표기의 예 :** N10 DEF REAL REAL_VAR=1.234567890123456EX+04 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%12.4G",REAL_VAR) 결과 : 문자열 "CONTENT OF REAL_VAR:xx1.235EX+06" 이 RESULT 문자열 변수에 쓰여집니다 . (예제의 "x" 는 공백을 나타냅니다 .)

%.<n>P:	REAL 값을 <n> 자리의 소수 부분을 고려한 INTEGER 값으로 변환 . INTEGER 값은 32 비트 이진수로 출력됩니다 . 변환할 값을 32 비트로 표현할 수 없는 경우에는 처리가 중단되고 알람이 표시됩니다 . 포맷 명령 %.<n>P를 사용하여 생성된 바이트 순서도 이진수 0을 포함할 수 있으므로 이 방법으로 생성되는 전체 문자열은 더 이상 NC 데이터 유형 STRING 의 규칙을 따르지 않습니다 . 결과적으로 이러한 문자열은 변수 유형 STRING 에 저장할 수도 없고 NC 언어의 문자열 명령을 사용하여 처리할 수도 없습니다 . 유일한 용도는 파라미터를 WRITE 명령으로 전송하여 적절한 외부 장치에서 출력하는 것입니다 (다음 예제 참조). <Format_String> 이 %P 유형의 포맷 설명을 포함하게 되면 %.<n>P 를 사용하여 생성된 이진수를 제외한 전체 문자열은 MD10750 $MN_SPRINT_FORMAT_P_CODE 에 따라 ASCII 문자 코드인 ISO(DIN6024) 또는 EIA(RS244) 의 형식에 맞게 출력됩니다 . 변환할 수 없는 문자가 프로그래밍되면 처리가 중단되고 알람이 표시됩니다 . **예 :** N10 DEF REAL REAL_VAR=123.45 N20 DEF INT ERROR N30 DEF STRING[20] EXT_DEVICE="/ext/dev/1" ... N100 EXTOPEN(ERROR,EXT_DEVICE) N110 IF ERROR <> 0 ... ; 에러 처리 N200 WRITE(ERROR,EXT_DEVICE,SPRINT("INTEGER BINARY CODED:%.3P",REAL_VAR) N210 IF ERROR <> 0 … ; 에러 처리 결과 : 문자열 "INTEGER BINARY CODED: 'H0001E23A'" 가 출력 장치 /ext/dev/1 로 전송됩니다 . 16 진수 값 0x0001E23A 는 십진수 값 123450 에 해당합니다 .

%<m>.<n>P:	머신 데이터 MD10751 $MN_SPRINT_FORMAT_P_DECIMAL 의 설정에 해당하는 REAL 값을 다음 숫자를 포함하는 문자열로 변환 • <m> + <n> 자리의 정수 **또는** • 최대 <m> 자리의 정수 부분과 <n> 자리의 소수 부분으로 이루어진 소수 포맷 설명 %.<n>P 의 경우와 마찬가지로 전체 문자열이 MD10750 $MN_SPRINT_FORMAT_P_CODE 에 의해 정의된 문자 코드로 저장됩니다 . **MD10751 = 0 인 경우의 변환 :** REAL 값은 <m> + <n> 자리의 정수를 포함한 문자열로 변환됩니다 . 필요한 경우 소수 부분은 <n> 자리로 반올림되거나 0 으로 채워집니다 . 누락된 정수 자리는 공백으로 채워집니다. 마이너스 부호가 첨부되며 왼쪽으로 정렬됩니다. 플러스 부호 대신 공백이 입력됩니다 . **예 :** N10 DEF REAL REAL_VAR=-123.45 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("PUNCHED TAPE FORMAT:%5.3P",REAL_VAR) 결과 : 문자열 "PUNCHED TAPE FORMAT:-xx123450" 이 RESULT 문자열 변수에 쓰여집니다 . (예제의 "x" 는 공백을 나타냅니다 .) **MD10751 = 1 인 경우의 변환 :** REAL 값은 최대 <m> 자리의 정수 부분과 <n> 자리의 소수 부분을 갖는 소수를 포함한 문자열로 변환됩니다 . 필요한 경우에는 정수 부분이 잘리고 소수 부분이 반올림되거나 0 으로 채워집니다 . <n> 이 0 과 같으면 소수점이 생략됩니다 . **예 :** N10 DEF REAL REAL_VAR1=-123.45 N20 DEF REAL REAL_VAR2=123.45 N30 DEF STRING[80] RESULT N40 RESULT=SPRINT("PUNCHED TAPE FORMAT:%5.3P VAR2:%2.0P", REAL_VAR1,REAL_VAR2) 결과 : 문자열 "PUNCHED TAPE FORMAT:-123.450 VAR2:23" 이 RESULT 문자열 변수에 쓰여집니다 .
%S:	문자열 삽입 **예 :** N10 DEF STRING[16] STRING_VAR="ABCDEFG" N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF STRING_VAR:%S",STRING_VAR) 결과 : 문자열 "CONTENT OF STRING_VAR:ABCDEFG" 가 RESULT 문자열 변수에 쓰여집니다 .
%<m>S:	최소 <m> 글자를 포함한 문자열 삽입 . 누락된 자리는 공백으로 채워집니다 . **예 :** N10 DEF STRING[16] STRING_VAR="ABCDEFG" N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF STRING_VAR:%10S",STRING_VAR) 결과 : 문자열 "CONTENT OF STRING_VAR:xxxABCDEFG" 가 RESULT 문자열 변수에 쓰여집니다 . (예제의 "x" 는 공백을 나타냅니다 .)
%.<n>S:	첫 번째 글자부터 시작하여 문자열의 <n> 글자 삽입 **예 :** N10 DEF STRING[16] STRING_VAR="ABCDEFG" N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF STRING_VAR:%.3S",STRING_VAR) 결과 : 문자열 "CONTENT OF STRING_VAR:ABC" 가 RESULT 문자열 변수에 쓰여집니다 .

%<m>.<n>S:	첫 번째 글자부터 시작하여 문자열의 <n> 글자 삽입 생성된 문자열의 총 길이는 <m> 글자 이상입니다 . 누락된 자리는 공백으로 채워집니다 . **예 :** N10 DEF STRING[16] STRING_VAR="ABCDEFG" N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF STRING_VAR:%10.5S", STRING_VAR) 결과 : 문자열 "CONTENT OF STRING_VAR:xxxxxABCDE" 가 RESULT 문자열 변수에 쓰여집니다 . (예제의 "x" 는 공백을 나타냅니다 .)
%X:	INTEGER 값을 16 진수 표기법을 사용한 문자열로 변환 **예 :** N10 DEF INT INT_VAR='HA5B8' N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("INTEGER HEXADECIMAL:%X",INT_VAR) 결과 : 문자열 "INTEGER HEXADECIMAL:A5B8" 이 RESULT 문자열 변수에 쓰여집니다 .

주

포맷 설명에는 식별자와 키워드에서 대문자와 소문자를 구분하지 않는 NC 언어의 속성도 적용됩니다 . 결과적으로 대문자와 소문자 중 어느 것을 사용하더라도 기능 차이 없이 프로그램할 수 있습니다 .

조합 옵션

다음 테이블은 NC 데이터 유형과 조합할 수 있는 포맷 설명에 대한 정보를 제공합니다. 암묵적 데이터 유형 변환 규칙이 적용됩니다 (" 데이터 유형 [쪽 55]" 참조).

	NC 데이터 유형						
	BOOL	CHAR	INT	REAL	STRING	AXIS	FRAME
%B	+	+	+	+	+	-	-
%C	-	+	-	-	+	-	-
%D	+	+	+	+	-	-	-
%F	-	-	+	+	-	-	-
%E	-	-	+	+	-	-	-
%G	-	-	+	+	-	-	-
%S	-	+	-	-	+	-	-
%X	+	+	+	-	-	-	-
%P	-	-	+	+	-	-	-

주

테이블은 NC 데이터 유형 AXIS 및 FRAME 을 SPRINT 기능에서 직접 사용할 수 없음을 나타냅니다 . 그러나 다음은 가능합니다 .

- AXIS 데이터 유형을 AXSTRING 기능을 사용하여 SPRINT 로 처리할 수 있는 문자열로 변환할 수 있습니다 .
- 프레임 콤포넌트에 액세스할 때마다 FRAME 데이터 유형의 개별 값을 읽을 수 있습니다 . 결과적으로 REAL 데이터 유형을 이용하여 SPRINT 로 처리할 수 있습니다 .

1.10 프로그램 점프 및 분기

1.10.1 프로그램 시작으로 점프 (GOTOS)

기능

프로그램을 반복하기 위해 GOTOS 명령을 사용하여 메인 또는 서브 프로그램의 시작으로 점프할 수 있습니다 .

머신 데이터를 사용하여 프로그램 시작으로 리턴 점프하도록 설정할 수 있습니다 .

- 프로그램 런타임을 "0" 으로 설정합니다 .
- 공작물 계산은 값 "1" 단위로 증분됩니다 .

구문

GOTOS

의미

GOTOS: 대상 위치가 프로그램 시작인 점프 명령 .

실행은 NC/PLC 인터페이스 신호를 통해 제어됩니다 .

DB21 에서 DBX384.0 로 (제어 프로그램 분기)

값 :	의미 :
0	프로그램 시작으로 리턴 점프하지 않습니다 . GOTOS 이후 다음 가공 프로그램 블록에서 프로그램이 다시 실행됩니다 .
1	프로그램 시작으로 리턴 점프합니다. 가공 프로그램이 반복됩니다.

보충 조건

- GOTOS 는 내부적으로 STOPRE (사전 프로세싱 중단) 를 실행합니다 .
- GOTOS 를 통해 데이터 정의 (LUD 변수) 가 있는 서브프로그램의 경우 , 정의 섹션 이후 첫 번째 프로그램 블록으로 점프합니다 . 예를 들어 데이터 정의는 다시 실행되지 않습니다 . 이것은 정의된 변수가 GOTOS 블록에서 도달된 값을 유지하고 정의 섹션에서 프로그래밍된 표준 값으로 리셋되지 않기 때문입니다 .
- GOTOS 명령은 동기 동작 및 고정 사이클에서 사용할 수 없습니다 .

예제

프로그램 코드	설명
N10 ...	; 프로그램 시작
...	
N90 GOTOS	; 프로그램 시작으로 점프
...	

1.10.2 점프 마커로 프로그램 점프 (GOTOB, GOTOF, GOTO, GOTOC)

기능

점프 마커 (레이블) 는 명령 GOTOF, GOTOB, GOTO 또는 GOTOC 를 사용하여 동일한 프로그램 내에서 다른 위치로 점프할 수 있도록 프로그램에서 설정할 수 있습니다 . 대상 마커 이후의 명령으로 프로그램 실행이 이 즉시 재개됩니다 . 즉 , 프로그램 내에서 분기를 실현할 수 있습니다 .

점프 마커 이외에 메인 및 서브 블록 번호를 점프 대상으로 사용할 수 있습니다 .

점프 명령 앞에 점프 조건 (IF...) 을 지정하면 프로그램 점프는 점프 조건을 충족하는 경우에만 실행됩니다 .

구문

GOTOB <jump destination>
IF <jump condition> = TRUE GOTOB <jump destination>
GOTOF <jump destination>
IF <jump condition> = TRUE GOTOF <jump destination>
GOTO <jump destination>
IF <jump condition> = TRUE GOTO <jump destination>
GOTOC <jump destination>
IF <jump condition> = TRUE GOTOC <jump destination>

의미

GOTOB:	프로그램 진행 방향과 반대 방향으로 점프 ..
GOTOF:	프로그램 진행 방향과 동일한 방향으로 점프 .
GOTO:	점프 대상 검색을 통한 점프 명령 . 검색은 먼저 프로그램 진행 방향으로 수행된 다음 프로그램 진행 반대 방향으로 수행됩니다 .
GOTOC:	알람 14080 "점프 대상을 찾을 수 없습니다"가 발생되지 않는다는 점을 제외하고 GOTO 와 동일한 효과 . 따라서 점프 대상 검색이 실패할 경우 프로그램 실행이 중단되지 않지만 GOTOC 명령에 따라 프로그램 행을 계속 수행합니다 .

< 점프 대상 >: 점프 대상 파라미터
가능한 데이터는 다음과 같습니다 .

< 점프 마커 >:	점프 대상은 사용자 정의 이름을 가진 프로그램에서 설정된 점프 마커 (레이블) 입니다 . <jump marker>:
< 블록 번호 >:	점프 대상이 메인 블록 또는 서브 블록 번호 (예: 200, N300)
STRING 유형 변수 :	변수 점프 대상 . 변수는 점프 마커 또는 블록 번호를 의미합니다 .

IF: 점프 조건을 작성하기 위한 키워드 .
점프 조건은 모든 비교 및 논리 연산을 허용합니다 (결과 : TRUE 또는 FALSE). 프로그램 점프는 이 연산 결과가 TRUE 인 경우 실행됩니다 .

주

점프 마커 (레이블)

점프 마커 (레이블) 는 항상 블록 시작에 위치합니다. 프로그램 번호가 있는 경우 점프 마커는 블록 번호 바로 다음에 위치합니다 .

점프 마커 이름을 지정하는 경우 다음 규칙이 적용됩니다 .

- 문자 개수 :
  - 최소 2 개
  - 최대 32 개
- 허용된 문자 :
  - 문자
  - 숫자
  - 밑줄 기호
- 처음 2 자는 글자 또는 밑줄이어야 합니다 .
- 점프 마커 이름 뒤에 콜론 (":") 이 위치합니다 .

보충 조건

- 점프 대상은 프로그램 **내에** 위치한 점프 마커 또는 블록 번호가 있는 블록만 가능합니다 .
- 점프 조건이 없는 점프 대상은 개별 블록에서 프로그래밍해야 합니다 . 조건 부 점프 명령에는 이 제한 사항이 적용되지 않습니다 . 이 경우 여러 점프 명령을 블록에 구성할 수 있습니다 .
- 점프 조건이 없고 점프 명령이 있는 프로그램의 경우 프로그램 M2/M30 의 끝이 반드시 프로그램 끝에 있을 필요는 없습니다 .

예제

예제 1: 점프 마커로 점프

프로그램 코드		설명
N10 …		
N20 GOTOF Label_1	;	프로그램 끝에서 점프 마커 "Label_1" 로 점프합니다 .
N30 …		
N40 Label_0: R1=R2+R3	;	점프 마커 "Label_0" 이 설정됩니다 .
N50 …		
N60 Label_1:	;	점프 마커 "Label_1" 이 설정됩니다 .
N70 …		
N80 GOTOB Label_0	;	프로그램 시작 방향에서 점프 마커 "Label_0" 으로 점프합니다 .
N90 …		

예제 2: 블록 번호로 간접 점프

프로그램 코드		설명
N5 R10=100		
N10 GOTOF "N"<<R10	;	블록 번호가 R10 에 위치한 블록으로 점프합니다 .
...		
N90 ...		
N100 ...	;	점프 대상
N110 ...		
...		

예제 3: 변수 점프 대상으로 점프

프로그램 코드		설명
DEF STRING[20] DESTINATION		
DESTINATION = "Marker2"		
GOTOF DESTINATION	;	프로그램 끝 방향에서 변수 점프 대상 DESTINATION 으로 점프합니다 .
Marker 1: T="Drill1"		
...		
Marker 2: T="Drill2"	;	점프 대상
...		

예제 4: 점프 조건을 통한 점프

프로그램 코드		설명
N40 R1=30 R2=60 R3=10 R4=11 R5=50 R6=20	;	초기값 할당 .
N41 LA1: G0 X=R2*COS(R1)+R5 Y=R2*SIN(R1)+R6	;	점프 마커 LA1 이 설정됩니다 .
N42 R1=R1+R3 R4=R4-1		

프로그램 코드	설명
N43 IF R4>0 GOTOB LA1	; 점프 조건을 충족하는 경우 프로그램 진행 방향과 반대 방향으로 점프 마커 LA1 로 점프합니다 .
N44 M30	; 프로그램 끝

1.10.3 프로그램 분기 (CASE ... OF ... DEFAULT ...)

기능

CASE 기능을 사용하면 변수 또는 산술 함수의 실제 값 (유형 : INT) 을 확인하고 결과에 따라 프로그램의 다른 위치로 점프할 수 있습니다 .

구문

CASE(<expression>) OF <constant_1> GOTOF <jump destination_1> <constant_2> GOTOF <jump destination_2> ... DEFAULT GOTOF <jump destination_n>

의미

CASE:	점프 문구
< 표현식 >:	변수 또는 산술 함수
OF:	조건 부 프로그램 분기를 구성하기 위한 키워드 .
< 상수 _1>:	변수 또는 산술 함수에 대해 지정된 첫 번째 상수 값 유형 : INT
< 상수 _2>:	변수 또는 산술 함수에 대해 지정된 두 번째 상수 값 유형 : INT
DEFAULT:	변수 또는 산술 함수가 지정된 상수 값을 적용하지 않는 경우 분기 대상을 결정하기 위해 DEFAULT 명령을 사용할 수 있습니다 . **참고 :** DEFAULT 명령이 프로그래밍되지 않은 경우 CASE 명령 다음에 오는 블록이 점프 대상입니다 .
GOTOF:	프로그램 진행 방향과 동일한 방향으로 점프 . GOTOF 대신 다른 모든 GOTO 명령을 프로그래밍할 수 있습니다 (제목 " 프로그램이 점프 마커로 점프 " 참조).

< 점프 대상 _1>: 변수 또는 산술 함수 값이 첫 번째 지정된 상수에 해당하는 경우 이 점프 대상에 대해 분기가 적용됩니다 .

점프 대상은 다음과 같이 지정할 수 있습니다 .

< 점프 마커 >: 점프 대상은 사용자 정의 이름을 가진 프로그램에서 설정된 점프 마커 (레이블) 입니다 . <jump marker>:

< 블록 번호 >: 점프 대상이 메인 블록 또는 서브블록 번호 (예 : 200, N300)

STRING 유형 변수 : 변수 점프 대상 . 변수는 점프 마커 또는 블록 번호를 의미합니다 .

< 점프 대상 _2>: 변수 또는 산술 함수 값이 두 번째 지정된 상수에 해당하는 경우 이 점프 대상에 대해 분기가 적용됩니다 .

< 점프 대상 _n>: 변수 값으로 지정된 상수 값을 적용하지 않는 경우 이 점프 대상에 대해 분기가 적용됩니다 .

예제

프로그램 코드

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

N30 의 CASE 명령은 다음 프로그램 분기 방법을 정의합니다 .

1. 산술 함수의 값이 VAR1+VAR2-VAR3 = 7 이면 점프 마커 정의 "Label_1" (→ N40) 를 가진 블록으로 점프합니다 .
2. 산술 함수의 값이 VAR1+VAR2-VAR3 = 9 이면 점프 마커 정의 "Label_2" (→ N50) 를 가진 블록으로 점프합니다 .
3. 산술 함수의 값이 VAR1+VAR2-VAR3 = 7 또는 9 가 아니면 점프 마커 정의 "Label_3" (→ N60) 을 가진 블록으로 점프합니다 .

1.11 프로그램 섹션 반복 (REPEAT, REPEATB, ENDLABEL, P)

기능

프로그램 섹션 반복을 통해 프로그램 내에서 기존 프로그램 섹션을 특정 순서대로 반복할 수 있습니다 .

반복할 프로그램 라인 또는 프로그램 섹션을 점프 마커 (레이블) 로 표시합니다 .

주

점프 마커 (레이블)

점프 마커는 항상 블록 시작에 위치합니다. 프로그램 번호가 있는 경우 점프 마커는 블록 번호 바로 다음에 위치합니다 .

점프 마커 이름을 지정하는 경우 다음 규칙이 적용됩니다 .

- 문자 개수 :
  - 최소 2
  - 최대 32
- 아래 문자들이 허용됩니다 .
  - 문자
  - 숫자
  - 밑줄
- 처음 2 자는 글자 또는 밑줄이어야 합니다 .
- 점프 마커 이름 뒤에 콜론 (":") 이 위치합니다 .

구문

1. 개별 프로그램 라인 반복 :

```
< 점프 마커 >: ...
...
REPEATB < 점프 마커 > P=<n>
...
```

2. 점프 마커와 REPEAT 문 사이의 프로그램 섹션 반복 :

```
< 점프 마커 >: ...
...
REPEAT < 점프 마커 > P=<n>
...
```

3. 두 점프 마커 사이의 섹션 반복 :

```
< 시작 점프 마커 >: ...
...
< 종료 점프 마커 >: ...
...
REPEAT < 시작 점프 마커 > < 종료 점프 마커 > P=<n>
...
```

주

REPEAT 문과 2 개의 점프 마커를 괄호로 함께 묶을 수 없습니다 . < 시작 점프 마커 > 는 REPEAT 문 앞에 있지만 < 종료 점프 마커 > 가 REPEAT 문 앞에 없으면 < 시작 점프 마커 > 와 REPEAT 문 사이의 섹션이 반복됩니다 .

4. 점프 마커와 ENDLABEL 사이의 섹션 반복 :

```
< 점프 마커 >: ...
...
ENDLABEL: ...
...
REPEAT < 점프 마커 > P=<n>
...
```

주

REPEAT 문을 < 점프 마커 > 및 ENDLABEL 과 함께 괄호 안에 묶을 수 없습니다 . < 점프 마커 > 가 REPEAT 문 앞에 있고 ENDLABEL 이 REPEAT 문 앞에 없으면 < 점프 마커 > 와 REPEAT 문 사이의 섹션이 반복됩니다 .

의미

REPEATB: 프로그램 라인 반복을 위한 명령어

REPEAT: 프로그램 섹션 반복을 위한 명령어

< 점프 마커 >: < 점프 마커 > 는 다음을 표시합니다 .

- 반복할 프로그램 라인 (REPEATB 의 경우)

  또는

- 반복할 프로그램 섹션의 시작 부분 (REPEAT 의 경우)

<점프 마커>로 표시된 프로그램 라인은 REPEAT/REPEATB 문의 앞 또는 뒤에 위치할 수 있습니다 . 처음에 프로그램 시작 방향으로 검색이 시작됩니다 . 이 방향에서 점프 마커를 찾을 수 없는 경우 프로그램 끝 방향으로 검색이 계속됩니다 .

예외 :
점프 마커와 REPEAT 문 사이의 프로그램 섹션을 반복해야 하는 경우 (2 항 구문 참조), < 점프 마커 > 로 표시한 프로그램 라인이 반드시 REPEAT 문 **앞에** 위치해야 합니다 . 그 이유는 검색이 프로그램 시작 **방향으로만** 진행되기 때문입니다 .

< 점프 마커 > 로 표시된 라인에 추가적인 연산이 있는 경우 라인이 반복될 때마다 이 연산도 다시 실행됩니다 .

ENDLABEL: 반복할 프로그램 섹션의 끝을 표시하는 키워드 .

ENDLABEL 로 표시된 라인에 추가적인 연산이 있는 경우 라인이 반복될 때마다 이 연산도 다시 실행됩니다 .

ENDLABEL 은 프로그램에서 1 회 이상 사용할 수 있습니다 .

P: 반복 횟수를 지정하는 어드레스

<n>: 프로그램 섹션 반복 횟수

유형 : INT

반복 대상인 프로그램 섹션을 <n> 회 반복합니다 . 마지막 반복 후 프로그램은 REPEAT/REPEATB 라인 다음 라인부터 다시 시작됩니다 .

참고 :
P=<n> 에 반복 횟수가 지정되지 않은 경우 프로그램 섹션은 단 1 회만 반복됩니다 .

예제

예제 1: 개별 프로그램 라인 반복

프로그램 코드	설명
N10 POSITION1: X10 Y20	
N20 POSITION2: CYCLE(0,,9,8)	; 위치 사이클
N30	
N40 REPEATB POSITION1 P=5	; BLOCK N10 을 5 회 실행 .
N50 REPEATB POSITION2	; 블록 N20 을 1 회 실행 .
N60 ...	
N70 M30	

예제 2: 점프 마커와 REPEAT 문 사이의 프로그램 섹션 반복

프로그램 코드	설명
N5 R10=15	
N10 Begin: R10=R10+1	; 너비
N20 Z=10-R10	
N30 G1 X=R10 F200	
N40 Y=R10	
N50 X=-R10	
N60 Y=-R10	
N70 Z=10+R10	
N80 REPEAT BEGIN P=4	; N10 부터 N70 까지의 섹션 4 회 실행 .
N90 Z10	
N100 M30	

예제 3: 두 점프 마커 사이의 섹션 반복

프로그램 코드	설명
N5 R10=15	
N10 Begin: R10=R10+1	; 너비
N20 Z=10-R10	
N30 G1 X=R10 F200	
N40 Y=R10	
N50 X=-R10	
N60 Y=-R10	
N70 END: Z=10	
N80 Z10	
N90 CYCLE(10,20,30)	
N100 REPEAT BEGIN END P=3	; N10 부터 N70 까지의 섹션 3 회 실행 .
N110 Z10	
N120 M30	

예제 4: 점프 마커와 ENDLABEL 사이의 섹션 반복

프로그램 코드	설명
N10 G1 F300 Z-10	
N20 BEGIN1:	
N30 X10	
N40 Y10	
N50 BEGIN2:	
N60 X20	
N70 Y30	
N80 ENDLABEL: Z10	
N90 X0 Y0 Z0	
N100 Z-10	
N110 BEGIN3: X20	
N120 Y30	
N130 REPEAT BEGIN3 P=3	; N110 부터 N120 까지의 섹션 3 회 실행 .
N140 REPEAT BEGIN2 P=2	; N50 부터 N80 까지의 섹션 2 회 실행 .
N150 M100	
N160 REPEAT BEGIN1 P=2	; N20 부터 N80 까지의 섹션 2 회 실행 .
N170 Z10	
N180 X0 Y0	
N190 M30	

예제 5: 밀링에서 각종 기술로 드릴 위치 가공

프로그램 코드	설명
N10 CENTER DRILL()	; 센터링 드릴 로드
N20 POS_1:	; 드릴링 위치 1
N30 X1 Y1	
N40 X2	
N50 Y2	
N60 X3 Y3	
N70 ENDLABEL:	
N80 POS_2:	; 드릴링 위치 2
N90 X10 Y5	
N100 X9 Y-5	
N110 X3 Y3	
N120 ENDLABEL:	
N130 DRILL()	; 드릴 및 드릴링 사이클 변경
N140 THREAD(6)	; 탭 M6 및 나사 사이클 로드
N150 REPEAT POS_1	; POS_1 에서 ENDLABEL 까지 프로그램 섹션 1 회 반복
N160 DRILL()	; 드릴 및 드릴링 사이클 변경 .
N170 THREAD(8)	; 탭 M8 및 스레딩 사이클 로드 .
N180 REPEAT POS_2	; POS_2 에서 ENDLABEL 까지 프로그램 섹션 1 회 반복
N190 M30	

추가 정보

- 프로그램 섹션 반복을 중첩할 수 있습니다 . 각 호출에서 서브프로그램 레벨을 사용합니다 .
- 프로그램 섹션 반복을 처리하는 중에 M17 또는 RET 이 프로그래밍되면 반복이 중단됩니다 . 해당 프로그램은 REPEAT 라인 다음 블록에서 다시 시작됩니다 .
- 실제 프로그램 디스플레이에서 프로그램 섹션 반복은 개별 서브프로그램 레벨로 표시됩니다 .
- 프로그램 섹션 반복 동안 레벨이 취소되면 프로그램은 프로그램 섹션 반복 호출 이후 지점에서 재개됩니다 .

  예제 :

프로그램 코드	설명
N5 R10=15	
N10 BEGIN: R10=R10+1	; 너비
N20 Z=10-R10	
N30 G1 X=R10 F200	
N40 Y=R10	; 중단 레벨
N50 X=-R10	
N60 Y=-R10	
N70 END: Z10	
N80 Z10	
N90 CYCLE(10,20,30)	
N100 REPEAT BEGIN END P=3	
N120 Z10	; 프로그램 실행 재개 .
N130 M30	

- 점검 구조 및 프로그램 섹션 반복을 조합하여 사용할 수 있습니다. 그러나 서로 간에 겹치지 않아야 합니다 . 프로그램 섹션 반복이 점검 구조 분기 내에 나타나거나 점검 구조가 프로그램 섹션 반복 내에 나타나야 합니다 .

- 점프 및 프로그램 섹션 반복이 혼합된 경우 블록은 정확하게 순서대로 실행됩니다 . 예를 들어 점프가 프로그램 섹션 반복에서 수행되는 경우 프로그램 섹션의 프로그램된 끝이 검색될 때까지 프로세스가 계속됩니다 .

  예제 :

프로그램 코드

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

주

REPEAT 문은 이송 블록 다음에 위치해야 합니다 .

1.12 점검 구조

기능

시스템은 프로그램 순서대로 NC 블록을 처리합니다 .

이 순서는 대체 프로그램 블록 및 프로그램 루프 프로그래밍에 따라 다를 수 있습니다 . 이러한 점검 구조는 점검 구조 요소 (키워드) IF...ELSE, LOOP, FOR, WHILE 및 REPEAT를 사용하여 프로그래밍됩니다 .

⚠ 주의

점검 구조는 프로그램의 명령문 섹션에만 삽입될 수 있습니다 . 프로그램 헤더 정의는 조건적 또는 반복적으로 실행될 수 없습니다 .

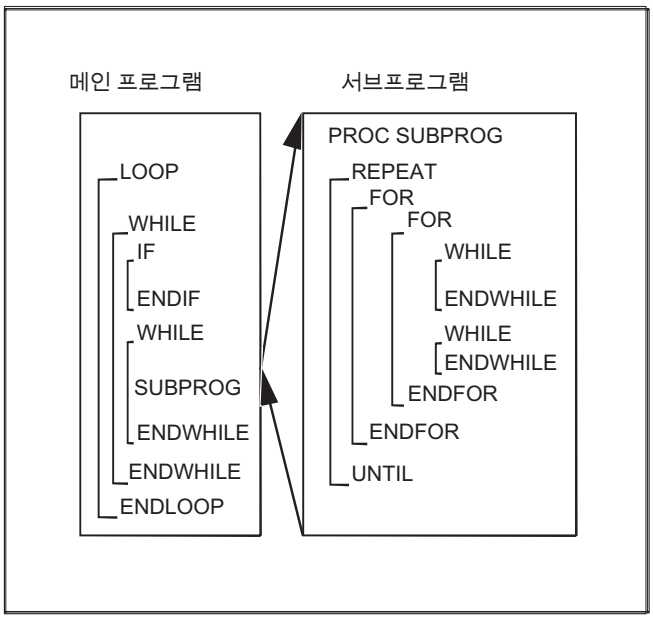
점검 구조의 키워드 또는 분기 대상에 대한 매크로를 적용할 수 없습니다 . 매크로를 정의할 경우 이러한 점검이 수행되지 않습니다 .

효율성

점검 구조는 프로그램 내에서 일부 적용됩니다 .

중첩 깊이

최대 16 개의 점검 구조에 대한 중첩 깊이를 각 서브프로그램 레벨에서 설정할 수 있습니다 .



런타임 응답

해석 모드 (표준으로 활성) 에서 점검 구조에서 가져올 수 있는 것보다 많은 프로그램 분기를 사용하여 , 프로그램 처리 시간을 상당히 효율적으로 줄일 수 있습니다 .

미리 컴파일된 사이클 내에서 프로그램 분기 및 점검 구조 사이에 차이가 없습니다 .

보충 조건

- 점검 구조 요소를 가진 블록을 억제할 수 없습니다 .
- 점검 구조 요소 블록에서 점프 마커 (라벨) 는 허용되지 않습니다 .
- 점검 구조는 분석적으로 처리됩니다 . 루프 끝이 감지되면 루프 시작을 검색하여 프로세스에 있는 점검 구조를 찾을 수 있습니다 . 이러한 이유로 프로그램의 블록 구조는 해석 모드에서 완전히 점검되지 않습니다 .
- 일반적으로 점검 구조와 프로그램 분기를 혼합하여 사용하지 않는 것이 바람직합니다 .
- 점검을 통해 사이클 사전 처리 시 , 점검 구조가 중첩되었는지 확인할 수 있습니다 .

1.12.1 대체 블록을 가진 프로그램 루프 (IF, ELSE, ENDIF)

기능

프로그램 루프에 대체 프로그램 블록이 포함되어야 하는 경우 IF 및 ELSE 로 구성된 명령문이 사용됩니다 . IF 조건을 충족하는 경우 IF 다음에 오는 프로그램 블록이 실행됩니다 . IF 조건을 충족하지 **않는 경우 ELSE** 다음에 오는 대체 프로그램 블록이 실행됩니다 .

주

대체 블록이 필요 없는 경우 IF 루프도 ELSE 명령 및 ELSE 다음에 오는 프로그램 블록 없이 프로그래밍할 수 있습니다 .

구문

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

의미

IF:	IF 루프를 적용합니다 .
ELSE:	대체 프로그램 블록을 적용합니다 .
ENDIF:	IF 루프 끝을 표시하고 루프 시작으로 리턴 점프합니다 .
< 조건 >:	프로그램 블록을 실행하는지 확인하는 조건 .

예제

공구 교환 서브프로그램

프로그램 코드	설명
PROC L6	; 공구 교환 루틴
N500 DEF INT TNR_AKTUELL	; 활성 T 번호에 대한 변수
N510 DEF INT TNR_VORWAHL	; 미리 선택한 T 번호에 대한 변수
	; 현재 공구 확인
N520 STOPRE	
N530 IF $P_ISTEST	; 프로그램 테스트 모드에서 ...
N540 TNR_AKTUELL = $P_TOOLNO	; ... " 현재 " 공구가 프로그램 컨텍스트에서 판독됩니다 .
N550 ELSE	; 그렇지 않으면 ...
N560 TNR_AKTUELL = $TC_MPP6[9998,1]	; ... 스핀들 공구가 판독됩니다 .
N570 ENDIF	
N580 GETSELT(TNR_VORWAHL)	; 스핀들에서 미리 선택한 공구의 T 번호가 판독됩니다 .
N590 IF TNR_AKTUELL <> TNR_VORWAHL	; 미리 선택한 공구가 계속 현재 공구가 아닌 경우 ...
N600 G0 G40 G60 G90 SUPA X450 Y300 Z300 D0	; ... 공구 교환 위치로 이동 ...
N610 M206	; ... 공구 교환을 실행합니다 .
N620 ENDIF	
N630 M17	

1.12.2 연속 프로그램 루프 (LOOP, ENDLOOP)

기능

연속 루프는 연속 프로그램에서 사용됩니다 . 루프의 끝에는 항상 시작으로 돌아가는 분기가 있습니다 .

구문

```
LOOP
...
ENDLOOP
```

의미

LOOP: 연속 루프를 시작합니다 .

ENDLOOP: 루프 끝을 표시하고 루프 시작으로 리턴 점프합니다 .

예제

프로그램 코드

```
...
LOOP
MSG ("no tool cutting edge active")
M0
STOPRE
ENDLOOP
...
```

1.12.3 카운트 루프 (FOR ... TO ..., ENDFOR)

기능

카운트 루프는 고정된 실행 횟수만큼 작업을 반복해야 하는 경우 사용됩니다 .

구문

```
FOR <variable> = <initial value> TO <end value>
...
ENDFOR
```

의미

FOR:	카운트 루프를 시작합니다 .
ENDFOR:	카운트 최종 값에 도달할 때까지 루프 끝을 표시하고 루프 시작으로 리턴 점프합니다 .
< 변수 >:	카운트 변수이며 처음부터 끝 값까지 증분되며 실행 시마다 "1" 값 단위로 증가합니다 . 유형 INT 또는 REAL **참고 :** REAL 유형은 예를 들어 카운트 루프에 대해 R 파라미터를 프로그래밍한 경우 사용됩니다 . 카운트 변수가 REAL 유형인 경우 해당 값은 정수로 반올림됩니다 .
< 초기값 >:	카운트 초기 값 조건 : 시작값은 반드시 최종값보다 작아야 합니다 .
< 최고 배율 값 >:	카운트 최종 값

예제

예제 1: 카운터 변수인 INTEGER 변수 또는 R 파라미터

카운터 변수인 INTEGER 변수 :

프로그램 코드	설명
DEF INT iVARIABLE1	
R10=R12-R20*R1 R11=6	
FOR iVARIABLE1 = R10 TO R11	; 카운트 변수 = INTEGER 변수
R20=R21*R22+R33	
ENDFOR	
M30	

카운터 변수인 R 파라미터 :

프로그램 코드	설명
R11=6	
FOR R10=R12-R20*R1 TO R11	; 카운터 변수 = R 파라미터 (Real 변수)
R20=R21*R22+R33	
ENDFOR	
M30	

예제 2: 지정된 수량 가공

프로그램 코드	설명
DEF INT WKPCCOUNT	; "WKPCCOUNT" 이름을 가진 유형 INT 변수를 정의합니다 .
FOR WKPCCOUNT = 0 TO 100	; 카운트 루프를 시작합니다 . "WKPCCOUNT" 변수는 초기값 "0" 부터 최종값 "100" 까지 증분합니다 .
G01 …	
ENDFOR	; 카운트 루프의 끝
M30	

1.12.4 루프 시작 시 조건을 가진 프로그램 루프 (WHILE, ENDWHILE)

기능

WHILE 루프의 경우 조건은 루프 시작에 위치합니다 . 조건을 충족하는 한 WHILE 루프가 실행됩니다 .

구문

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

의미

WHILE:	프로그램 루프를 시작합니다 .
ENDWHILE:	루프 끝을 표시하고 루프 시작으로 리턴 점프합니다 .
< 조건 >:	WHILE 루프가 실행되도록 조건을 충족해야 합니다 .

예제

프로그램 코드	설명
...	
WHILE $AA_IW[DRILL_AXIS] > -10	; 다음 조건 하에서 WHILE 루프를 호출합니다 . 드릴링 축에 대해 실제 WCS 지령치는 -10 이상이어야 합니다 .
G1 G91 F250 AX[DRILL_AXIS] = -1	
ENDWHILE	
...	

1.12.5 루프 끝에서 조건을 가진 프로그램 루프 (REPEAT, UNTIL)

기능

REPEAT 루프의 경우 조건은 루프 끝에 위치합니다. REPEAT 루프는 한번만 실행되고 조건을 충족할 때까지 계속 반복합니다 .

구문

```
REPEAT
...
UNTIL <significance>
```

의미

REPEAT:	프로그램 루프를 시작합니다 .
UNTIL:	루프 끝을 표시하고 루프 시작으로 리턴 점프합니다 .
< 조건 >:	REPEAT 루프가 더 이상 실행되지 않도록 충족해야 하는 조건입니다 .

예제

프로그램 코드	설명
...	
REPEAT	; REPEAT 루프를 호출합니다 .
...	
UNTIL ...	; 조건을 충족하는지 여부를 확인합니다 .
...	

1.12.6 중첩된 점검 구조를 가진 프로그램 예

프로그램 코드	비고
LOOP	
IF NOT $P_SEARCH	; 블록 검색 없음
G01 G90 X0 Z10 F1000	
WHILE $AA_IM[X] <= 100	
G1 G91 X10 F500	; 심공 드릴링 템플릿
Z–F100	
Z5	
ENDWHILE	
Z10	
ELSE	
MSG("No drilling during block search")	
ENDIF	
$A_OUT[1] = 1	; 다음 드릴링 플레이트
G4 F2	
ENDLOOP	
M30	

1.13 프로그램 코디네이션 (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM)

기능

채널

채널은 다른 채널과 독립적으로 프로그램을 처리할 수 있습니다. 프로그램을 통해 일시적으로 할당된 축 및 스핀들을 제어할 수 있습니다 .

시운전 시 제어를 위한 두 개 이상의 채널을 설정할 수 있습니다 .

프로그램 코디네이션

공작물 가공에 여러 채널이 관련된 경우 프로그램을 동기화해야 합니다 .

이 프로그램 코디네이션을 위한 특별한 명령문이 있습니다. 각 명령문은 블록에서 개별적으로 프로그래밍됩니다 .

주

또한 프로그램 코디네이션은 해당 채널에서 수행할 수 있습니다 .

프로그램 코디네이션 명령문

- 절대 경로를 통한 지정

	절대 경로는 다음 규칙에 따라 프로그래밍됩니다 .
INIT (n,"/_HUGO_DIR/_N_name_MPF") 또는	- 현재 디렉토리 /_N_name_MPF " 현재 디렉토리 " 는 선택한 공작물 디렉토리 또는 표준 디렉토리 /_N_MPF_DIR 을 의미합니다 .
INIT (n,"/_N_MPF_DIR/_N_name_MPF")	- 특정 채널에서 실행할 특정 프로그램을 선택합니다 . n: 채널 개수는 시스템 구성에 따라 다릅니다 . - 전체 프로그램 이름
예제 :	**최대 SW 3:**
INIT(2,"/_N_WKS_DIR/_DRESS_MPF") G01F0.1 START	**INIT** 명령 (동기화 없음) 및 **NC 시작** 사이에 최소 하나의 실행 가능한 블록을 프로그래밍해야 합니다 . 서브프로그램 호출 "_SPF" 를 경로에 추가해야 합니다 .
INIT (2,"/_N_WKS_DIR/ _N_UNDER_1_SPF")	

- 상대 경로 지정

예제 :	프로그램 호출처럼 동일한 규칙을 상대 경로 정의에 적용합니다 .
INIT(2,"DRESS")	
INIT(3,"UNDER_1_SPF")	서브프로그램 호출 "_SPF" 를 프로그램 이름에 추가해야 합니다 .

파라미터

프로그램 간 데이터 교환을 위해 모든 채널에서 액세스할 수 있는 변수 (NCK 별 전역 변수) 를 사용할 수 있습니다 . 그렇지 않을 경우 각 채널에 대해 개별 프로그램을 작성해야 합니다 .

INIT(n, 경로 이름 , 인지 모드)	채널에서 실행하기 위한 명령입니다 . 절대 또는 상대 경로 이름을 가진 특정 프로그램을 선택합니다 .
START (n, n)	선택한 프로그램을 다른 채널에서 시작합니다 . n,n: 채널 번호 열거 : 값은 제어 구성에 따라 다릅니다 .
WAITM (marker no., n, n, ...)	동일한 채널에 마커 "marker no." 를 설정합니다 . 정위치 정지를 통해 이전 블록을 종료합니다. 지정된 채널 "n" (현재 채널은 지정할 필요 없음) 에서 마커가 동일한 "marker no." 가 될 때까지 대기합니다 . 동기화 이후 마커가 삭제됩니다 . 채널당 10 개의 마커를 동시에 설정할 수 있습니다 .
WAITMC (marker No., n, n,	동일한 채널에 마커 "marker no." 를 설정합니다 . 정위치 정지는 다른 채널이 마커에 아직 도달하지 않은 경우에만 시작됩니다. 지정된 채널 "n" (현재 채널은 지정할 필요 없음) 에서 마커가 동일한 "marker no." 가 될 때까지 대기합니다 . 지정된 채널에서 마커 "marker no."에 도달하는 즉시 정위치 정지로 전환 없이 계속됩니다 .
WAITE (n, n, ...)	지정된 채널 (현재 채널 지정되지 않음) 의 프로그램이 종료될 때까지 기다립니다 . 예 : 시작 명령 이후 지연 시간 프로그램 . N30 START(2) N31 G4 F0.01 N40 WAITE(2)
SETM (marker No., marker No.,	현재 프로세스에 영향을 주지 않도록 동일한 채널에서 마커 "marker no." 를 설정합니다 . SETM() 은 RESET 및 NC START 이후에도 유효합니다 .
CLEARM (marker No., marker No.,	현재 프로세스에 영향을 주지 않도록 동일한 채널에서 마커 "marker no."를 삭제합니다. CLEARM() 을 통해 모든 마커를 삭제할 수 있습니다. CLEARM (0) 은 마커 "0"을 삭제합니다. CLEARM() 은 RESET 및 NC START 이후에도 유효합니다 .
n	해당 채널 번호 또는 채널 이름

주

위의 모든 명령은 개별 블록에서 프로그래밍해야 합니다 .

마커 수는 사용되는 CPU 에 따라 다릅니다 .

채널 수

최대 10 개의 채널을 코디네이션이 필요한 채널의 채널 수 (정수 값) 로 지정할 수 있습니다 .

채널 이름

채널 이름은 변수를 사용하여 숫자로 변환되어야 합니다 ("변수 및 산술 파라미터" 참조). 또는 $MC_CHAN_NAME 을 사용하여 정의된 채널 이름 (식별자 또는 키워드) 도 채널 번호 이외의 이름으로 프로그래밍할 수 있습니다. 정의된 이름은 NC 명명 표기법을 준수해야 합니다. 예를 들어 첫 번째 두 개의 글자는 문자 또는 밑줄이어야 합니다 .

⚠ 주의
실수로 변경되지 않도록 숫자 지정을 보호합니다 . 예를 들어 키워드 , 명령 , 축 이름 등과 같이 NC 에 다른 의미를 가진 이름이 없어야 합니다 .

SETM() 및 CLEARM()

SETM() 및 CLEARM() 도 동기화된 동작과는 독립적으로 프로그래밍할 수 있습니다 . " 대기 라벨 설정 / 삭제 : SETM CLEARM " 장을 참조하십시오 .

예제

"MACHINE" 이라는 채널의 채널 번호가 1 이고 ,

"LOADER" 이라는 채널의 채널 번호가 2 인 경우 :

DEF INT MACHINE=1, LOADER=2

변수는 채널과 동일한 이름이 지정됩니다 .

따라서 START 문은 다음과 같습니다 .

START(MACHINE)

예제 : 프로그램 코디네이션

채널 1:

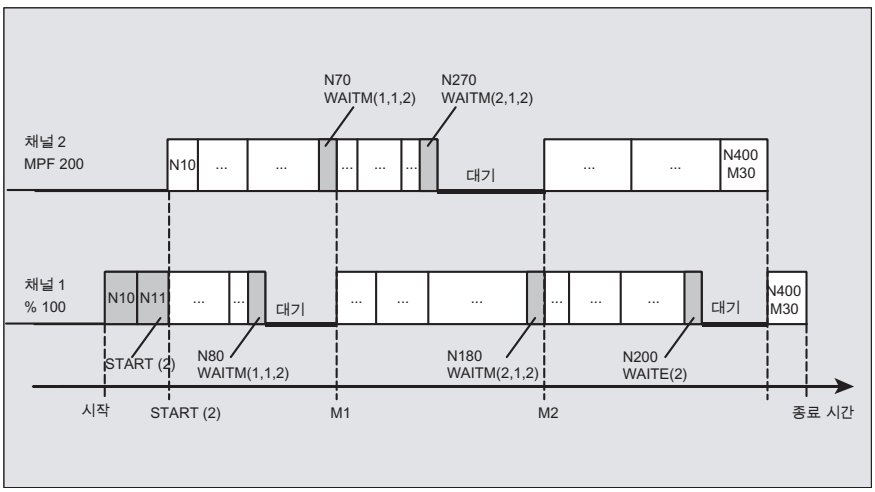
_N_MPF100_MPF

프로그램 코드	설명
N10 INIT(2,"MPF200")	
N11 START(2)	; 채널 2 의 프로세싱
...	
N80 WAITM(1,1,2)	; 채널 1 및 채널 2 에서 WAIT 마커 1 대기 , 채널 1 에서 추가 처리
...	
N180 WAITM(2,1,2)	; 채널 1 및 채널 2 에서 WAIT 마커 2 대기 , 채널 2 에서 추가 처리
...	
N200 WAITE(2)	; 채널 2 의 프로그램 종료 대기
N201 M30	; 프로그램 채널 1 종료 , 전체 종료
...	

채널 2:

_N_MPF200_MPF

프로그램 코드	설명
;$PATH=/_N_MPF_DIR	
	; 채널 2 의 프로세스
N70 WAITM(1,1,2)	; 채널 1 및 채널 2 에서 WAIT 마커 2 대기 , 채널 1 에서 추가 처리
...	
N270 WAITM(2,1,2)	; 채널 1 및 채널 2 에서 WAIT 마커 2 대기 , 채널 2 에서 추가 처리
...	
N400 M30	; 채널 2 에서 프로그램 종료



예제 : 공작물 프로그램

프로그램 코드
N10 INIT(2,"/_N_WKS_DIR/_N_SHAFT1_WPD/_N_CUT1_MPF")

예제 : 상대 경로가 지정된 INIT 명령

채널 1 에서 프로그램 /_N_MPF_DIR/_N_MAIN_MPF 선택

프로그램 코드	설명
N10 INIT(2,"MYPROG")	; 채널 2 에서 프로그램 /_N_MPF_DIR/_N_MYPROG_MPF 선택

예제 : 정수 변수를 가진 채널 이름 및 채널 번호

$MC_CHAN_NAME[0]= "CHAN_X" ; 첫 번째 채널 이름
$MC_CHAN_NAME[1]= "CHAN_Y" ; 두 번째 채널 이름

프로그램 코드	설명
START(1, 2)	; 첫 번째 및 두 번째 채널에서 실행 시작

마찬가지로 채널 식별자를 통한 프로그램 :

프로그램 코드	설명
START(CHAN_X, CHAN_Y)	; 첫 번째 및 두 번째 채널에서 시작 실행 ; $MC_CHAN_NAME 머신 데이터에 의한 내부 채널 번호 1 및 2 또한 첫 번째 및 두 번째 채널에서 실행 시작

정수 변수를 통한 프로그램 :

프로그램 코드	설명
DEF INT chanNo1, chanNo2)	; 채널 번호 정의
chanNo1=CHAN_X chanNo2=CHAN_Y	
START(chanNo1, chanNo2)	

1.14 인터럽트 루틴 (ASUB)

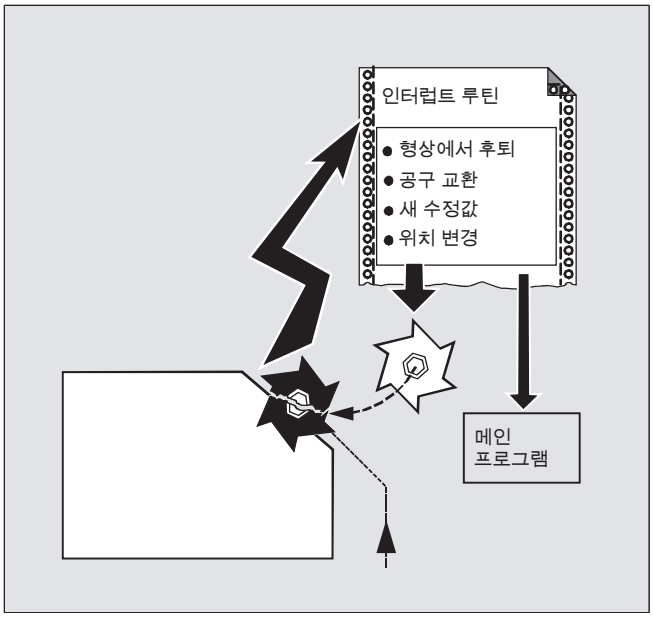
1.14.1 인터럽트 루틴 기능

주

" 비동기 서브프로그램 (ASUB)" 및 " 인터럽트 루틴 " 이란 용어는 여기서 동일한 기능을 설명하는 것이며 서로 번갈아 사용됩니다 .

기능

전형적인 예는 인터럽트 루틴의 기능에 대해 명확하게 설명해야 합니다 .



가공하는 동안 공구가 파손되었습니다 . 이로 인해 현재 가공 프로세스를 중단하는 동시에 인터럽트 루틴이라는 서브프로그램을 시작한다는 신호가 실행됩니다 . 인터럽트 루틴에는 이 경우에 실행해야 할 모든 명령문이 포함되어 있습니다 .

인터럽트 루틴이 실행을 완료하고 기계가 작동을 계속할 준비를 마치면 제어 시스템은 메인 프로그램으로 다시 이동하여 REPOS 명령에 따라 중단된 지점에서 가공을 계속합니다 (" 형상에 대해 위치 변경 (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [쪽 478]" 참조).

⚠ 주의

REPOS 명령이 서브프로그램에서 프로그래밍되지 않은 경우 제어 시스템은 중단된 블록과 이어지는 블록 끝 지점으로 이동합니다 .

참고 자료

기능 매뉴얼, 기본 기능; BAG, 채널, 프로그램 조작, 리셋 응답 (K1), 장: "비동기 서브프로그램 (ASUB), 인터럽트 루틴 "

1.14.2 인터럽트 루틴 생성

인터럽트 루틴을 서브프로그램으로 생성

인터럽트 루틴은 정의 시 서브프로그램으로 식별됩니다 .

예제 :

프로그램 코드	설명
PROC LIFT_Z	; 프로그램 이름 "ABHEB_Z"
N10 ...	; NC 블록이 계속됨
...	
N50 M17	; 최종적으로 프로그램이 종료되고 메인 프로그램으로 복귀합니다 .

모달 G 코드 저장 (SAVE)

인터럽트 루틴은 SAVE 를 정의하여 지정할 수 있습니다 .

SAVE 특성은 인터럽트 루틴을 호출하기 전에 활성화된 모달 G 코드를 저장하고 인터럽트 루틴을 종료한 후 다시 활성화됨을 의미합니다 (" 모달 G 코드 저장 (SAVE) [쪽 167]" 참조).

따라서 인터럽트 루틴을 완료한 후 중단 지점에서 프로세스를 다시 시작할 수 있습니다 .

예제 :

프로그램 코드
PROC LIFT_Z SAVE
N10 ...
...
N50 M17

추가 인터럽트 루틴 할당 (SETINT)

SETINT 명령을 인터럽트 루틴 내에서 프로그래밍할 수 있으며 (인터럽트 루틴 지정 및 시작 (SETINT, PRIO, BLSYNC) [쪽 121]) 따라서 추가 인터럽트 루틴을 활성화합니다 . 이 작업은 입력을 통해 실행됩니다 .

참고 자료

서브프로그램 생성 방법에 대한 자세한 내용은 " 서브프로그램 , 매크로 " 장을 참조하십시오 .

1.14.3 인터럽트 루틴 지정 및 시작 (SETINT, PRIO, BLSYNC)

기능

제어 시스템은 실행 중인 프로그램을 중단하고 해당 인터럽트 루틴을 시작할 수 있음을 나타내는 신호 (입력 1...8) 를 실행합니다 .

입력에 대한 할당으로 가공 프로그램에서 SETINT 명령을 사용하여 프로그램이 실행되기 시작합니다 .

여러 SETINT 명령이 가공 프로그램에 있고 따라서 여러 신호를 동시에 수신할 수 있는 경우 할당된 인터럽트 루틴이 실행되는 순서를 정의한 우선 순위를 인터럽트 루틴에 할당해야 합니다 . PRIO=<value>

인터럽트 루틴을 실행하는 동안 새 신호를 수신하면 현재 인터럽트 루틴은 높은 우선 순위의 루틴에 의해 중단됩니다 .

구문

SETINT(<n>) PRIO=<value> <NAME>
SETINT(<n>) PRIO=< 값 > <NAME> BLSYNC
SETINT(<n>) PRIO=< 값 > <NAME> LIFTFAST

의미

SETINT(<n>):	명령 : 입력 <n> 을 인터럽트 루틴에 할당합니다 . 할당된 인터럽트 루틴은 입력 <n> 이 전환될 때 시작됩니다 . **참고 :** 이미 할당된 입력이 새 루틴에 할당될 경우 기존 할당은 자동으로 취소됩니다 .
<n>:	파라미터 : 입력 번호 유형 : INT 범위 : 1 ... 8
PRIO= :	명령 : 우선 순위 정의
< 값 >:	우선 순위 값 유형 : INT 범위 : 1 ... 128 우선 순위 1 은 최상위 우선 순위에 해당합니다 .
<NAME>:	실행해야 하는 서브프로그램 (인터럽트 루틴) 의 이름
BLSYNC:	SETINT 연산이 BLSYNC 와 함께 프로그래밍된 경우 , 인터럽트 신호가 수신되어도 진행 중인 프로그램 블록은 계속해서 처리됩니다 . 이 블록의 처리가 완료되어야만 인터럽트 루틴이 시작됩니다 .
LIFTFAST:	SETINT 연산을 LIFTFAST 와 함께 프로그래밍한 경우 , 인터럽트 신호가 수신되면 인터럽트 루틴을 시작하기 전에 "형상에서 공구의 고속 후퇴"를 먼저 실행합니다 ("형상으로부터 빠른 후퇴 (SETINT LIFTFAST, ALF) [쪽 125]" 참조).

예제

예제 1: 인터럽트 루틴 할당 및 우선 순위 정의

프로그램 코드	설명
...	
N20 SETINT(3) PRIO=1 ABHEB_Z	; 입력 3이 ON되면 인터럽트 루틴 "ABHEB_Z"가 시작되어야 합니다 .
N30 SETINT(2) PRIO=2 ABHEB_X	; 입력 2가 ON되면 인터럽트 루틴 "ABHEB_X"가 시작되어야 합니다 .
...	

입력을 동시에 사용할 수 있는 경우 (동시에 활성화되는 경우) 인터럽트 루틴은 우선 순위 값의 순서에 따라 실행됩니다 . 먼저 "ABHEB_Z" 가 실행된 다음 "ABHEB_X" 가 실행됩니다 .

예제 2: 새로운 인터럽트 루틴 할당

프로그램 코드	설명
...	
N20 SETINT(3) PRIO=2 ABHEB_Z	; 입력 3이 ON되면 인터럽트 루틴 "ABHEB_Z"가 시작되어야 합니다 .
…	
N120 SETINT(3) PRIO=1 ABHEB_X	; 입력 3이 새 인터럽트 루틴에 할당됩니다. 입력 3이 ON되면 "ABHEB_Z" 대신 "ABHEB_X" 가 시작되어야 합니다 .

1.14.4 인터럽트 루틴 할당 비활성화 / 다시 활성화 (DISABLE, ENABLE)

기능

SETINT 명령은 DISABLE을 통해 비활성화할 수 있으며 입력 → 인터럽트 루틴 할당 손실 없이 ENABLE 을 통해 다시 활성화할 수 있습니다 .

구문

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

의미

DISABLE(<n>):	명령 : 입력 <n> 의 인터럽트 루틴 할당 **비활성화**	
ENABLE(<n>):	명령 : 입력 <n> 의 인터럽트 루틴 할당 **다시 활성화**	
<n>:	파라미터 : 입력 번호	
	유형 :	INT
	범위 :	1 ... 8

예제

프로그램 코드	설명
...	
N20 SETINT(3) PRIO=1 ABHEB_Z	; 입력 3 이 ON 되면 인터럽트 루틴 "ABHEB_Z" 가 시작되어야 합니다 .
...	
N90 DISABLE(3)	; N20 에서 SETINT 명령이 비활성화됩니다 .
...	
N130 ENABLE(3)	; N20 에서 SETINT 명령이 다시 활성화됩니다 .
...	

1.14.5 인터럽트 루틴 할당 삭제 (CLRINT)

기능

입력 → SETINT 를 사용하여 인터럽트 루틴 정의 , CLRINT 로 삭제할 수 있습니다 .

구문

CLRINT(<n>)

의미

CLRINT(<n>):	명령 : 입력 <n> 의 인터럽트 루틴 할당 삭제	
<n>:	파라미터 : 입력 번호	
	유형 :	INT
	범위 :	1 ... 8

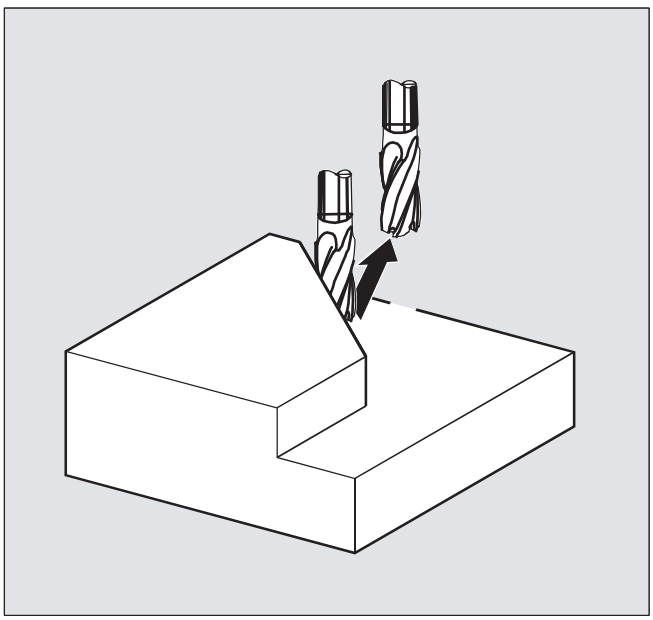
예제

프로그램 코드	설명
...	
N20 SETINT(3) PRIO=2 ABHEB_Z	;
...	
N50 CLRINT(3)	; 입력 "3" 및 인터럽트 루틴 "ABHEB_Z" 사이 할당이 삭제됩니다 .
...	

1.14.6 형상으로부터 빠른 후퇴 (SETINT LIFTFAST, ALF)

기능

LIFTFAST 가 있는 SETINT 명령의 경우 입력이 ON 되면 공구가 공작물 형상으로부터 빠르게 후퇴합니다 .



추가 시퀀스는 SETINT 명령에 LIFTFAST 이외의 인터럽트 루틴이 포함되었는지 여부에 따라 다릅니다 .

인터럽트 루틴이 포함된 경우 :	고속 후퇴 **이후**에 인터럽트 루틴이 실행됩니다 .
인터럽트 루틴이 없는 경우 :	고속 후퇴 이후 가공이 중단되고 알람이 출력됩니다 .

구문

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

의미

SETINT(<n>):	명령 : 입력 <n> 을 인터럽트 루틴에 할당합니다 . 할당된 인터럽트 루틴은 입력 <n> 이 ON 될 때 시작됩니다 .	
<n>:	파라미터 : 입력 번호	
	유형 :	INT
	범위 :	1 ... 8
PRIO= :	우선 순위 정의	

< 값 >:	우선 순위 값 범위 : 1 ... 128 우선 순위 1 은 최상위 우선 순위에 해당합니다 .
<NAME>:	실행할 서브프로그램 (인터럽트 루틴) 의 이름 .
LIFTFAST:	명령 : 형상으로부터 고속 후퇴
ALF=… :	명령 : 프로그램 가능한 이송 방향 (이동 블록) ALF 를 통한 프로그램 가능성에 대한 내용은 제목 " 형상으로부터 빠른 후퇴에 대한 이송 방향 [쪽 127]" 을 참조하십시오 .

보충 조건

미러링을 통한 활성 프레임 동작

상승 방향을 결정할 때 미러를 가진 프레임이 활성 상태인지 확인합니다 . 이 경우 후퇴 방향에 대해, 오른쪽 및 왼쪽이 서로 교체됩니다. 공구 방향의 방향 구성 요소는 미러링되지 않습니다. 이 동작은 MD 설정을 사용하여 활성화됩니다 .

MD21202 $MC_LIFTFAST_WITH_MIRROR = TRUE

예제

파손된 공구는 자동으로 대체 공구로 교체되어야 합니다 . 그런 다음 새 공구로 가공이 계속됩니다 .

메인 프로그램 :

메인 프로그램	설명
N10 SETINT(1) PRIO=1 W_WECHS LIFTFAST	; 입력 1 이 ON 되면 공구는 즉시 형상으로부터 빠르게 상승하여 후퇴합니다 (공구 반경 보정 G41 의 경우 코드 번호 7). 그런 다음 인터럽트 루틴 "W_WECHS" 가 실행됩니다 .
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	

서브프로그램 :

서브프로그램	설명
PROC W_CHANGE SAVE	; 실제 작동 상태가 저장되는 서브프로그램
N10 G0 Z100 M5	; 공구 교환 위치 , 스핀들 정지

서브프로그램	설명
N20 T11 M6 D1 G41	; 공구 교환
N30 REPOSL RMB M3	; 형상으로 재 위치하고 메인 프로그램으로 리턴 점프합니다 . (이 작업은 한 블록 내에 프로그래밍되어 있어야 합니다 .)

1.14.7 형상으로부터 빠른 후퇴에 대한 이송 방향

후퇴 이동

다음 G 코드는 후퇴 이동 평면을 정의합니다 .

- LFTXT

  후퇴 이동 평면은 경로 접선 및 공구 방향에 의해 정의됩니다 (기본 설정).

- LFWP

  후퇴 이동 평면은 G 코드 G17, G18 또는 G19 를 사용하여 선택된 활성 작업 / 가공 평면입니다 . 후퇴 이동 방향은 접선 경로에 종속되지 않습니다 . 따라서 축과 평행한 방향으로 빠른 상승을 프로그래밍할 수 있습니다 .

- LFPOS

  POLF 로 프로그래밍한 절대 축 위치로 POLFMASK/POLFMLIN 를 이용해 축의 후퇴를 선언합니다 .

  직선 시스템 내에 여러 축이 있는 경우 ALF 는 상승 방향에 영향을 주지 않습니다 .

  참고 자료 :
  프로그래밍 매뉴얼 , 기본 사항 . 장 : " 나사 절삭 시 고속 후퇴 "

프로그램 가능한 이송 방향 (ALF=...)

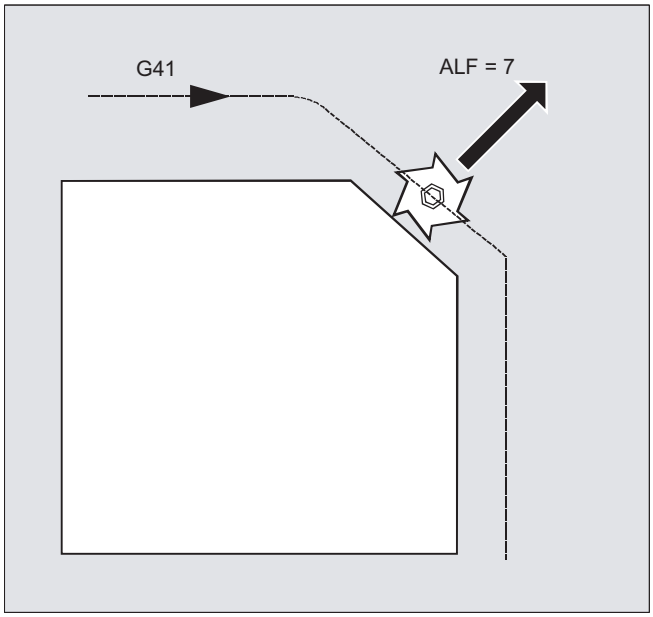
방향은 후퇴 이동 평면에서 ALF 를 통해 45 도 단위씩 프로그래밍됩니다 .

가능한 이송 방향은 제어 시스템의 특수 코드 번호로 저장되며 이러한 번호를 사용하여 호출할 수 있습니다 .

예제 :

프로그램 코드
N10 SETINT(2) PRIO=1 LIFT_Z LIFTFAST
ALF=7

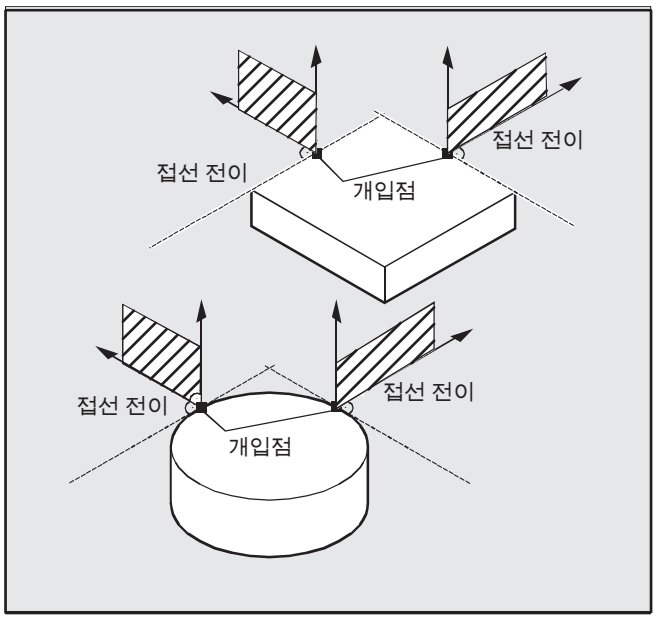
G41 이 활성화된 상태에서 (가공 방향이 형상 왼쪽 방향) 공구는 형상으로부터 수직 방향으로 후퇴합니다 .



LFTXT 의 이송 방향을 정의하기 위한 기준 평면

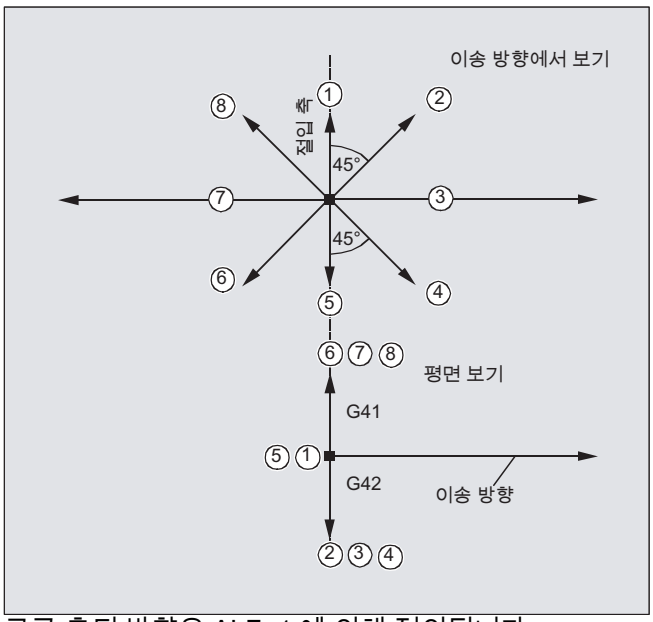
공구가 프로그래밍된 형상에 적용하는 시점에서 , 공구는 해당 코드 번호를 통해 상승 이동을 지정하기 위한 기준으로 사용되는 평면에서 고정됩니다 .

기준 평면은 수직 공구 축 (절입 방향) 및 이 축과 수직 위치에 있는 벡터로부터 파생되며 공구 적용 지점의 접선 방향과 수직을 이룹니다 .



LFTXT 의 이송 방향이 적용된 코드 번호

기준 평면부터 시작하여 다음 그림에서 이송 방향을 가진 코드 번호를 찾습니다 .



공구 후퇴 방향은 ALF=1 에 의해 정의됩니다 .

" 빠른 후퇴 " 동작은 ALF=0 에 의해 비활성화됩니다 .

⚠ 주의
공구 반경 보정이 활성화될 경우 : • G41 코드 2, 3, 4 • G42 코드 6, 7, 8 이러한 경우 , 공구가 형상으로 이동하여 공작물과 충돌하기 때문에 사용하지 **않아야 합니다** .

LFWP 의 이송 방향이 적용된 코드 번호

LFWP 의 경우 , 작업 / 가공 평면 방향은 다음과 같이 할당됩니다 .

- G17: X/Y 평면

  ALF=1: X 방향으로 후퇴

  ALF=3: Y 방향으로 후퇴
- G18: Z/X 평면

  ALF=1: Z 방향으로 후퇴

  ALF=3: X 방향으로 후퇴
- G19: Y/Z 평면

  ALF=1: Y 방향으로 후퇴

  ALF=3: Z 방향으로 후퇴

1.14.8 인터럽트 루틴의 이동 순서

LIFTFAST 없이 인터럽트 루틴

축 이동은 경로 아래를 따라 제동되어 정지합니다 (0 속도). 그런 다음 인터럽트 루틴이 시작됩니다 .

정지 위치는 인터럽트 위치로 저장되고 RMI 를 통해 REPOS 에 대한 인터럽트 루틴 끝에 접근합니다 .

LIFTFAST 에 의한 인터럽트 루틴

축 이동은 경로를 따라 제동됩니다 . LIFTFAST 이동은 중복된 이동으로 동시에 실행됩니다 . 경로 이동 및 LIFTFAST 이동이 정지하면 (0 속도) 인터럽트 루틴이 시작됩니다 .

형상 위치는 LIFTFAST 이동이 시작되고 그에 따라 경로만 남겨진 인터럽트 위치로 저장됩니다 .

LIFTFAST 및 ALF=0 에 의한 인터럽트 루틴은 LIFTFAST 없는 인터럽트 루틴과 동일한 방법으로 정밀하게 동작합니다 .

주

형상으로부터 신속하게 후퇴할 때 형상 축을 이동하는 절대값은 가공 데이터를 사용하여 설정할 수 있습니다 .

1.15 축 교체 , 스핀들 교체 (RELEASE, GET, GETD)

기능

한 개의 채널에서 하나 이상의 축 또는 스핀들 만 보간됩니다. 축을 두 개의 서로 다른 채널 (예: 펠릿 충전기) 간에 번갈아 사용해야 하는 경우 먼저 현재 채널에서 활성화한 다음 다른 채널로 이송해야 합니다 . 채널 간에 축 교체가 유효합니다 .

축 교체 확장

사전 처리 중지 및 사전 처리와 기본 실행 사이 또는 사전 처리 중지 없이 축 / 스핀들을 교체할 수 있습니다 . 또한 축 교체는 다음을 통해 수행할 수 있습니다 .

- 함축적 GET/GETD 를 사용한 축 컨테이너 회전 AXCTSWE 또는 AXCTWED
- 이 프로세스에서 해당 축이 다른 축과 연결된 경우 회전 상태의 프레임
- 동기 동작에 대해서는 이동 동기화 작업 , " 축 교체 RELEASE, GET" 을 참조하십시오 .

장비 제조업체

장비 제조업체의 설명서를 참조하십시오. 축을 교체하는 경우 구성 가능한 머신 데이터의 모든 채널에서 한 축을 고유하게 정의해야 하며 또한 가공 데이터를 사용하여 축 교체 특성을 설정할 수 있습니다 .

구문

RELEASE (축 이름 , 축 이름 , ...) or RELEASE (S1)

GET (축 이름 , 축 이름 , ...) or GET (S2)

GETD (축 이름 , 축 이름 , ...) or GETD(S3)

GETD (GET Directly) 를 통해 축을 다른 채널에서 직접 가져옵니다 . 따라서 다른 채널에서 이 GETD 에 대해 RELEASE 를 프로그래밍하지 않아야 합니다 . 또한 다른 채널과의 통신을 연결해야 합니다 (예 : 대기 마커).

의미

RELEASE (축 이름 , 축 이름 , …):	축을 분리합니다 .
GET (축 이름 , 축 이름 , …):	축을 적용합니다 .
GETD (축 이름 , 축 이름 , …):	직접 축을 적용합니다 .
축 이름	시스템에서 축 할당 : AX1, AX2, ... 또는 가공 축 이름을 지정합니다 .
RELEASE(S1):	스핀들 S1, S2, ... 를 분리합니다 .
GET(S2):	스핀들 S1, S2, ... 를 적용합니다 .
GETD(S3):	직접 스핀들 S1, S2, ... 를 적용합니다 .

사전 처리 중지 없이 GET 요청

사전 처리 중지 **없이** GET 요청 이후 축이 RELEASE(axis) 또는 WAITP(axis) 를 통해 다시 활성화된 경우 후속 GET 은 사전 처리 중지를 **통한** GET 으로 유도됩니다 .

⚠ 주의

GET 를 통해 적용된 축 또는 스핀들은 키 또는 프로그램 리셋 이후에도 이 채널에 할당된 상태를 유지합니다 .

프로그램을 다시 시작할 때 축이 원래 채널에서 필요한 경우 교체된 축 또는 스핀들을 프로그램에서 다시 할당해야 합니다 .

전원을 켠 상태에서 머신 데이터에서 정의된 채널에 할당됩니다 .

예제

예제 1: 두 채널 간 축 교체

6 개 축 중에서 채널 1 에서 가공하는 경우 다음이 사용됩니다 . 1., 2., 3. 번째 및 4 번째 축 채널 2 의 5 번째 및 6 번째 축은 공작물 교체 시 사용됩니다 .

축 2 는 두 채널 간에 교체되어야 하고 전원을 켠 후 채널 1 에 할당될 수 있습니다 .

채널 1 의 메인 프로그램 :

프로그램 코드	설명
INIT (2,"TRANSFER2")	; 채널 2 에서 프로그램 TRANSFER2 를 선택합니다 .
N… START (2)	; 채널 2 에서 프로그램을 시작합니다 .
N… GET (AX2)	; 축 AX2 를 적용합니다 .
...	
N… RELEASE (AX2)	; 축 AX2 를 분리합니다 .
N… WAITM (1,1,2)	; 채널 1 및 2 에서 동기화를 위해 두 채널에서 WAIT 마커에 도달할 때까지 대기합니다 .
...	; 축 교체 이후 나머지 프로그램 .
N… M30	

채널 2 의 TRANSFER2 프로그램 :

프로그래밍	설명
N… RELEASE (AX2)	
N160 WAITM(1,1,2)	; 채널 1 및 2 에서 동기화를 위해 두 채널에서 WAIT 마커에 도달할 때까지 대기합니다 .
N150 GET(AX2)	; 축 AX2 를 적용합니다 .
...	; 축 교체 이후 나머지 프로그램 .
N… M30	

예제 2: 동기화 없이 축 교체

축을 동기화할 필요가 없는 경우 GET 에 의해 사전 처리 중지가 생성되지 않습니다 .

프로그래밍	설명
N01 G0 X0	
N02 RELEASE(AX5)	
N03 G64 X10	
N04 X20	
N05 GET(AX5)	; 동기화가 필요 없는 경우 실행할 수 있는 블록이 아닙니다 .
N06 G01 F5000	; 실행할 수 없는 블록입니다 .
N07 X20	; N04 의 X 위치처럼 실행할 수 없는 블록입니다 .
N08 X30	; N05 이후 실행할 수 있는 첫 번째 블록입니다 .
...	

예제 3: 전처리 정지 없이 축 교체 실행

사전 조건 : 사전 처리 중지 없이 축 교체는 머신 데이터를 통해 구성해야 합니다 .

프로그래밍	설명
N010 M4 S100	
N011 G4 F2	
N020 M5	
N021 SPOS=0	
N022 POS[B]=1	
N023 WAITP(B)	; B 축이 중립 축이 됩니다 .
N030 X1 F10	
N031 X100 F500	
N032 X200	
N040 M3 S500	; 축에서 사전 처리 중지 /REORG 가 실행되지 않습니다 .
N041 G4 F2	
N050 M5	
N099 M30	

스핀들 또는 축 B 가 180 도 각도로 이송된 다음 **PLC 축**인 블록 N023 이후 즉시 1 도 각도로 복귀하는 경우 이 축은 중립 상태로 복귀되고 블록 N40 에서 사전 처리 중지를 실행하지 않습니다 .

요구 사항

축 교체를 위한 사전 조건

- 축은 가공 데이터의 축을 사용하는 모든 채널에서 정의되어야 합니다 .
- **축**별 가공 데이터에서 전원을 켠 후 축이 할당되는 채널을 정의해야 합니다 .

설명

축 분리 : RELEASE

축을 활성화하는 경우 다음 사항에 유의하십시오 .

1. 축을 변환하지 않아야 합니다 .
2. 축 링크 (접선 제어) 에 관련된 모든 축을 활성화해야 합니다 .
3. 이 경우 동시 위치 지정 축을 교체할 수 없습니다 .
4. 갠트리 마스터 축의 다음 모든 축은 마스터를 통해 이송됩니다 .
5. 결합된 축 (결합 이동 , 마스터 값 결합 , 전자식 기어) 에서는 그룹의 선행 축만 활성화할 수 있습니다 .

축 적용 : GET

실제 축 교체는 이 명령으로 수행됩니다 . 명령이 프로그래밍되는 채널에서 해당 축의 모든 작업이 수행됩니다 .

GET 효과 :

동기화를 통한 축 교체 :

다른 채널 또는 PLC 에 할당되었고 "WAITP", G74 를 통해 다시 동기화되지 않은 경우 축을 항상 동기화해야 하거나 GET 이전 이동 거리를 삭제해야 합니다 .

- 사전 처리 중지가 수행됩니다 (STOPRE 의 경우).
- 교체가 완료될 때까지 실행이 중단됩니다 .

자동 "GET"

기본적으로 축을 채널에서 사용 가능하지만 현재 " 채널 축 " 으로 정의되지 않은 경우 GET 가 자동으로 실행됩니다 . 축이 이미 동기화된 경우 사전 처리 중지가 실행되지 않습니다 .

다양한 축 교체 동작

가공 데이터를 사용하여 다음과 같이 축 이송 점을 설정할 수 있습니다 .

- 또한 축이 WAITP 에 의해 중립 상태가 되면 (이전처럼 응답) 두 채널 간 자동 축 교체가 실행됩니다 .
- 축 컨테이너 회전을 요청할 경우 실행 중인 채널에 할당할 수 있는 축 컨테이너의 모든 축이 함축적인 GET 또는 GETD 를 사용하여 채널로 이송됩니다 . 후속 축 교체는 축 컨테이너 회전을 완료한 경우 다시 한 번만 허용됩니다 .
- 중간 블록이 기본 실행 중에 삽입되면 재구성이 필요한지 여부를 확인합니다 . 재구성은 이 블록의 축 상태가 현재 축 상태와 일치하지 **않는** 경우에만 필요합니다 .
- 사전 처리 중지 및 사전 처리와 기본 실행 간 동기화를 통한 GET 블록 대신 사전 처리 중지 없이 축을 교체할 수 있습니다 . 이 경우 중간 블록은 간단히 GET 요청에 의해 생성됩니다 . 기본 실행 중에 이 블록을 실행하면 시스템은 블록의 축 상태가 현재 축 상태와 일치하는지 여부를 확인합니다 .

축 또는 스핀들 교체 작업 방법에 대한 자세한 내용은
/FB2/ 작동 설명서 , 확장 기능 , 모드 그룹 , 채널 , 축 교체 (K5) 를 참조하십시오 .

1.16 다른 채널로 축 이송 (AXTOCHAN)

기능

AXTOCHAN 언어 명령은 축을 다른 채널로 이동하기 위해 축을 요청하는 데 사용할 수 있습니다 . 축은 NC 가공 프로그램 및 동기 동작에서 해당 채널로 이동할 수 있습니다 .

구문

AXTOCHAN(axis name,channel number[,axis name,channel number[,...]])

의미

AXTOCHAN:	특정 채널의 축 요청
축 이름	시스템에서 축 할당 : X, Y, … 또는 관련 가공 축 이름 입력 . 실행 중인 채널이 같은 채널 또는 현재 축을 보간 처리 중인 채널일 필요가 없습니다 .
채널 번호 :	축을 할당할 채널 이름

주

경쟁 포지셔닝 축 및 PLC 제어 축

PLC 축은 채널을 경쟁 포지셔닝 축으로 교체할 수 없습니다 . PLC 에 의해서만 제어되는 축은 NC 프로그램에 할당할 수 없습니다 .

참고 자료 :
기능 매뉴얼 , 확장 기능 ; 포지셔닝 축 (P2)

예제

NC 프로그램에서 AXTOCHAN

X 및 Y 축이 첫 번째 및 두 번째 채널에서 선언되었습니다 . 현재 채널 1 이 우측 보간 상태이며 다음 프로그램이 해당 채널에서 시작됩니다 .

프로그램 코드	설명
N110 AXTOCHAN(Y,2)	; Y 축을 두 번째 채널로 이동합니다 .
N111 M0	
N120 AXTOCHAN(Y,1)	; Y 축을 복귀시킵니다 (중립).
N121 M0	
N130 AXTOCHAN(Y,2,X,2)	; Y 축 및 X 축을 두 번째 채널로 이동합니다 (축이 중립 상태).
N131 M0	
N140 AXTOCHAN(Y,2)	; Y 축을 두 번째 채널로 이동합니다 (NC 프로그램).
N141 M0	

추가 정보

NC 프로그램에서 AXTOCHAN

GET 은 동일한 채널에서 NC 프로그램에 대해 축을 요청하는 경우에만 실행됩니다 . 따라서 시스템은 상태가 실제로 변경될 때까지 대기합니다 . 다른 채널에 대해 축을 요청하거나 축이 동일한 채널에서 중립 축이 되어야 하는 경우 그에 따라 요청이 전달됩니다 .

동기 동작의 AXTOCHAN

동일한 채널에 대해 축을 요청하는 경우 동기 동작의 AXTOCHAN 은 동기 동작의 GET 으로 매핑됩니다 . 이 경우 축은 동일한 채널에 대해 첫 번째 요청 시 중립 축이 됩니다 . 두 번째 요청 시 축은 NC 프로그램에서 GET 요청과 동일한 방법으로 NC 프로그램에 할당됩니다 . 동기 동작의 GET 요청에 대한 자세한 내용은 " 동기화 동작 " 을 참조하십시오 .

1.17 가공 데이터 활성화 (NEWCONF)

기능

NEWCONF 명령은 "NEW_CONFIG" 적용 레벨의 모든 머신 데이터를 활성 상태로 설정하는 데 사용됩니다 . 또한 "MD 데이터 활성 " 소프트 키를 눌러 HMI 사용자 인터페이스에서 기능을 활성화할 수 있습니다 .

NEWCONF 기능이 실행되면 함축적인 전처리 정지가 실행됩니다 . 다시 말해 경로 이동이 중단됩니다 .

구문

NEWCONF

의미

NEWCONF :	"NEW_CONFIG" 효율성 레벨의 모든 머신 데이터를 활성 상태로 설정하는 명령어 .

가공 프로그램에서 채널 간 NEWCONF 실행

가공 프로그램에서 축 머신 데이터를 변경한 후 NEWCONF 로 활성화한 경우 , NEWCONF 는 가공 프로그램 채널에 영향을 주는 변경 내용이 포함된 머신 데이터만 활성화합니다 .

주

모든 변경 내용이 적용되도록 하려면 머신 데이터의 변경 내용에 의해 영향을 받는 축 또는 기능을 계산하는 모든 채널에서 NEWCONF 명령을 실행해야 합니다 .

NEWCONF 에 대한 축 머신 데이터가 유효하지 않습니다 .

PLC 에 의해 제어되는 축은 반드시 축 RESET 을 수행해야 합니다 .

예제

밀링 : 서로 다른 방법으로 드릴 가공

프로그램 코드	설명
N10 $MA_CONTOUR_TOL[AX]=1.0	; 머신 데이터 변경 .
N20 NEWCONF	; 머신 데이터 활성화 .
...	

1.18 파일 쓰기 (WRITE)

기능

WRITE 명령을 사용하면 NC 프로그램의 세트 / 데이터를 패시브 파일 시스템의 지정된 파일 (로그 파일) 끝에 쓸 수 있습니다 . 이것은 현재 실행 중인 프로그램일 수도 있습니다 .

주

해당 파일이 NC 에 없는 경우 새 파일이 생성되고 WRITE 명령을 사용해 그 파일에 쓸 수 있습니다 .

이 파일은 스태틱 NC 메모리에 저장됩니다 . SINUMERIK 840D sl 의 경우 CompactFlash Card에 저장됩니다. SINUMERIK 840D와 달리 스태틱 NC 메모리에 저장하면 WRITE 명령의 런타임이 약 75ms 증가합니다 .

동일한 이름을 가진 파일이 하드 디스크에 있을 경우 NC 에서 파일을 닫은 후 덮어씁니다 . 해결 방법 (HMI Advanced): "Properties (속성)" 소프트 키를 사용하여 " 서비스 " 작동 영역 아래 NC 에서 이름을 변경합니다 .

또한 WRITE 명령을 사용하면 NC 프로그램의 데이터를 외부 장치 / 외부 파일에 쓸 수도 있습니다 (" 외부 장치 / 파일로 출력 (EXTOPEN, WRITE, EXTCLOSE) [쪽 696]" 참조).

조건

현재 설정된 보호 레벨은 파일의 WRITE 권한과 동일하거나 커야 합니다 . 그렇지 않을 경우 에러 메시지 (에러 변수의 리턴 값 = 13) 와 함께 액세스가 거부됩니다 .

구문

```
DEF INT < 에러 >
...
WRITE(<error>,"<file name>"/"<ExtG>","<set/data>")
```

중요도

WRITE: 지정된 파일의 끝에 블록 또는 데이터를 추가하는 명령어 .

< 에러 >: **파라미터 1:** 에러 값을 반환하기 위한 변수

유형 . INT

값 :

0	오류 없음
1	경로 허용 안됨
2	경로 찾을 수 없음
3	파일 찾을 수 없음
4	잘못된 파일 유형
10	파일이 가득 참
11	파일 사용 중
12	사용 가능한 리소스 없음
13	액세스 권한 없음
14	외부 장치가 할당되거나 개방되지 않음
15	외부 장치로 쓸 때 발생한 에러
16	잘못된 외부 경로가 프로그래밍됨

< 파일 이름 >: **파라미터 2:** 지정된 블록 또는 데이터가 추가될 패시브 파일 시스템의 파일 이름 .

유형 : STRING

파일 이름을 지정할 때 다음 사항을 고려해야 합니다 .

- 파일 이름에 공백 또는 제어 문자 (32 자 이하의 ASCII 코드를 가진 문자) 가 포함되면 에러 코드 1 " 경로 허용 안됨 " 이 표시되면서 WRITE 명령이 취소됩니다 .
- 경로 데이터 및 파일 식별자로 파일 이름을 지정할 수 있습니다 .
  - 경로 데이터

    경로 데이터는 절대값이어야 하므로 "/" 로 시작합니다 .

    지정된 경로가 없는 경우 파일은 현재 디렉토리 (선택한 프로그램의 디렉토리) 에 저장됩니다 .
  - 파일 식별자

    파일 이름에 도메인 이름 ("_N_") 이 없는 경우 추가됩니다 .

    파일 이름의 끝에서 네 번째 문자가 밑줄 ("_") 인 경우 그 다음 세 문자가 파일 식별자로 해석됩니다 . STRING 유형의 변수를 이용해 모든 파일 명령에 동일한 파일 이름을 사용하는 경우 _SPF 및 _MPF 만 파일 식별자로 사용할 수 있습니다 .

    "_MPF" 또는 "_SPF" 와 같은 파일 식별자가 없는 경우 파일 이름은 자동적으로 _MPF 로 끝나게 됩니다 .
- 파일 이름 길이는 최대 32 바이트이며 경로 길이는 최대 128 바이트입니다 .

예제 :

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

<ExtG>: 데이터를 외부 장치 / 파일로 출력하려는 경우에는 열고자 하는 외부 장치 / 파일의 심볼 식별자를 파일 이름 대신 지정해야 합니다 .

유형 : STRING

자세한 내용은 " 외부 장치 / 파일로 출력 (EXTOPEN, WRITE, EXTCLOSE) [쪽 696]" 을 참조하십시오 .

참고 :
식별자는 EXTOPEN 명령에서 지정한 식별자와 동일해야 합니다 .

<블록 / 데이터>: 지정된 파일에 추가할 블록 또는 데이터 .

유형 : STRING

주

NCK 의 패시브 파일 시스템에 쓰는 경우 WRITE 명령은 출력 문자열 끝에 "LF" 문자 (LINE FEED = 새 라인) 를 암묵적으로 삽입합니다 .

이 동작은 외부 장치 / 파일에 출력하는 경우에는 적용되지 않습니다 . "LF" 도 출력하는 경우에는 출력 문자열에 명시적으로 지정해야 합니다 .

예제 3: 암묵적 / 명시적 "LF" 를 참조하십시오 !

일반 조건

- **최대 파일 크기 (→ 장비 제조업체)**

  패시브 파일 시스템에서 가능한 로그 파일의 최대 파일 크기는 머신 데이터로 설정합니다 .

  MD11420 $MN_LEN_PROTOCOL_FILE

  최대 파일 길이는 패시브 파일 시스템에서 WRITE 명령을 사용해 생성한 모든 파일에 적용됩니다 . 최대 파일 길이를 초과하면 에러 메시지가 출력되고 해당 블록 또는 데이터가 저장되지 않습니다 . 사용 가능한 메모리가 충분할 경우 새 파일을 생성할 수 있습니다 .

예제

예제 1: 절대 경로 데이터가 없는 패시브 파일 시스템에 WRITE 명령 적용

프로그램 코드	코멘트
N10 DEF INT ERROR	; 에러 변수 정의 .
N20 WRITE(ERROR,"PROT","LOG FROM 7.2.97")	; "LOG FROM 7.2.97" 의 텍스트를 _N_PROT_MPF 파일에 씁니다 .
N30 IF ERROR	; 오류 평가 .
N40 MSG ("Error with WRITE command:" <<ERROR)	
N50 M0	
N60 ENDIF	
...	

예제 2: 절대 경로 데이터를 포함한 패시브 파일 시스템에 WRITE 명령 적용

프로그램 코드

```
...
WRITE(ERROR,"/_N_WKS_DIR/_N_PROT_WPD/_N_PROT_MPF","LOG FROM 7.2.97")
...
```

예제 3: 암묵적 / 명시적 "LF"

a, 암묵적으로 생성된 "LF" 를 포함하여 패시브 파일 시스템에 쓰기

프로그램 코드

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

출력 결과 :

MY_STRING

MY_STRING

b, 암묵적으로 생성된 "LF" 를 포함하지 않고 외부 파일에 쓰기

프로그램 코드

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

출력 결과 :

MY_STRINGMY_STRING

c, 명시적으로 생성된 "LF" 를 포함하여 외부 파일에 쓰기

a 에서와 동일한 결과를 얻으려면 다음을 프로그래밍해야 합니다 .

프로그램 코드

```
...
N200 DEF STRING[30] DEV_1
N210 DEF INT ERROR
N220 DEV_1="LOCAL_DRIVE/myprotfile.mpf"
```

프로그램 코드

```
N230 EXTOPEN(ERROR,DEV_1)
N240 WRITE(ERROR,DEV_1,"MY_STRING'H0A'")
N250 WRITE(ERROR,DEV_1,"MY_STRING'H0A'")
N260 EXTCLOSE(ERROR,DEV_1)
N270 M30
```

출력 결과 :

MY_STRING

MY_STRING

도 참조

외부 장치/파일로 출력 (EXTOPEN, WRITE, EXTCLOSE) 외부 장치/파일로 출력 (EXTOPEN, WRITE, EXTCLOSE) [쪽 696]

1.19 파일 삭제 (DELETE)

기능

파일을 WRITE 명령을 통해 생성했는지 여부와 관계 없이 DELETE 명령을 이용해 모든 파일을 삭제할 수 있습니다 . 또한 더 높은 레벨의 액세스 권한을 사용하여 생성된 파일도 DELETE 를 이용해 삭제할 수 있습니다 .

구문

DEF INT < 에러 >
DELETE(< 에러 >,"<file name(파일 이름)>")

의미

DELETE:	지정된 파일을 삭제하는 명령어 .		
< 에러 >:	에러 값을 리턴하기 위한 변수 .		
	유형	INT	
	값 :	0	에러 없음
		1	경로 허용 안됨
		2	경로 찾을 수 없음
		3	파일 찾을 수 없음
		4	잘못된 파일 유형
		11	파일 사용 중
		12	사용 가능한 리소스 없음
		20	기타 에러

< 파일 이름 >: 삭제할 파일 이름

유형 : STRING

파일 이름을 지정할 때 다음 사항을 고려해야 합니다 .

- 파일 이름에 공백 또는 제어 문자 (32 자 이하의 ASCII 코드를 가진 문자) 가 포함되면 에러 코드 1 "경로 허용 안됨"이 표시되면서 DELETE 명령이 취소됩니다 .
- 경로 데이터 및 파일 식별자로 파일 이름을 지정할 수 있습니다 .
  - 경로 데이터

    경로 데이터는 절대값이어야 하므로 "/" 로 시작합니다 .

    지정된 경로가 없는 경우 현재 디렉토리 (선택한 프로그램의 디렉토리) 에서 해당 파일을 검색합니다 .
  - 파일 식별자

    파일 이름에 도메인 이름 ("_N_") 이 없는 경우 추가됩니다 .

    파일 이름의 끝에서 네 번째 문자가 밑줄 ("_") 인 경우 그 다음 세 문자가 파일 식별자로 해석됩니다 . STRING 유형의 변수를 이용해 모든 파일 명령에 동일한 파일 이름을 사용하는 경우 _SPF 및 _MPF 만 파일 식별자로 사용할 수 있습니다 .

    "_MPF" 또는 "_SPF" 와 같은 파일 식별자가 없는 경우 파일 이름은 자동적으로 _MPF 로 끝나게 됩니다 .
- 파일 이름 길이는 최대 32 바이트이며 경로 길이는 최대 128 바이트입니다 .

예제 :

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

예제

프로그램 코드	설명
N10 DEF INT ERROR	; 에러 변수 정의 .
N15 STOPRE	; 전처리 정지 .
N20 DELETE(ERROR,"/_N_SPF_DIR/_N_TEST1_SPF")	; 서브프로그램 디렉토리에서 파일 TEST1 을 삭제합니다 .
N30 IF ERROR	; 오류 평가 .
N40 MSG("error for DELETE command:" <<ERROR)	
N50 M0	
N60 ENDIF	

1.20 파일의 행 읽기 (READ)

기능

READ 명령은 지정된 파일에서 하나 또는 여러 라인을 읽고 STRING 유형의 배열에 읽은 정보를 저장합니다 . 이 배열에서 각각의 읽은 행에 배열 요소가 포함됩니다 .

주

정보를 읽을 파일은 NCK 의 스태틱 유저 메모리 (패시브 파일 시스템) 에 저장되어 있어야 합니다 .

조건

현재 설정된 보호 레벨은 파일의 READ 권한과 동일하거나 커야 합니다 . 그렇지 않을 경우 에러 메시지 (에러 변수의 리턴 값 = 13) 와 함께 액세스가 거부됩니다 .

구문

DEF INT < 에러 >
DEF STRING[< 문자열 길이 >] < 결과 >[<n>,<m>]
READ(< 에러 >,"<file name(파일 이름)>",< 시작 라인 >,< 라인의 수 >,< 결과 >)

의미

READ: 지정된 파일에서 라인을 읽고 해당 라인을 변수 배열에 저장하도록 하는 명령어 .

< 에러 >: 에러 값 리턴을 위한 변수 (기준별 호출 파라미터)

유형 . INT

값 :

0	에러 없음
1	경로 허용 안됨
2	경로 찾을 수 없음
3	파일 찾을 수 없음
4	잘못된 파일 유형
13	액세스 권한 부족
21	라인이 존재하지 않음 (<시작 라인> 또는 <라인의 수 > 파라미터가 지정된 파일의 라인 수를 초과한 경우).
22	결과 변수 (< 결과 >) 의 필드 길이가 너무 작음 .
23	라인의 범위가 너무 큼 (선택한 < 라인의 수 > 파라미터가 너무 커서 읽기가 파일 끝을 초과하여 진행).

< 파일 이름 >: 읽을 파일의 이름 (값별 호출 파라미터)

유형 : STRING

파일 이름을 지정할 때 다음 사항을 고려해야 합니다 .

- 파일 이름에 공백 또는 제어 문자 (32 자 이하의 ASCII 코드를 가진 문자) 가 포함되면 에러 코드 1 "경로 허용 안됨"이 표시되면서 READ 명령이 취소됩니다 .
- 경로 및 파일 식별자로 파일 이름을 지정할 수 있습니다 .
  - 경로 데이터

    경로 데이터는 절대값이어야 하므로 "/" 로 시작합니다 .

    지정된 경로가 없는 경우 현재 디렉토리 (선택한 프로그램의 디렉토리) 에서 해당 파일을 검색합니다 .
  - 파일 식별자

    파일 이름에 도메인 이름 ("_N_") 이 없는 경우 추가됩니다 .

    파일 이름의 끝에서 네 번째 문자가 밑줄 ("_") 인 경우 그 다음 세 문자가 파일 식별자로 해석됩니다 . STRING 유형의 변수를 이용해 모든 파일 명령에 동일한 파일 이름을 사용하는 경우 _SPF 및 _MPF 만 파일 식별자로 사용할 수 있습니다 .

    "_MPF" 또는 "_SPF" 와 같은 파일 식별자가 없는 경우 파일 이름은 자동적으로 _MPF 로 끝나게 됩니다 .
- 파일 이름 길이는 최대 32 바이트이며 경로 길이는 최대 128 바이트입니다 .

예제 :

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

< 시작 라인 >: 읽을 파일 섹션의 시작 라인 (값별 호출 파라미터)

유형 : INT

값 : 0 < 라인의 수 > 파라미터가 지정한 수의 라인을 파일 끝 전까지 읽습니다 .

1 ~ n 읽을 첫 번째 라인의 번호 .

< 라인의 수 >: 읽을 라인의 개수 (값별 호출 파라미터)

유형 : INT

< 결과 >: 결과 변수 (기준별 호출 파라미터)
읽은 텍스트가 저장된 변수 배열 .

유형 : STRING (최대 길이 : 255)

< 라인의 수 > 파라미터가 지정한 라인의 개수가 결과 변수의 배열 크기 [<n>,<m>] 보다 작은 경우 나머지 배열 요소는 수정되지 않습니다 .

제어 문자 "LF" (줄 바꿈) 또는 "CR LF" (캐리지 리턴 줄 바꿈) 를 사용해 라인을 종료하면 결과 변수에 저장되지 **않습니다** .

읽은 라인이 지정된 문자열 길이보다 길면 초과한 부분이 잘립니다 . 오류 메시지가 출력되지 않습니다 .

주

이진 파일은 읽어올 수 없습니다 . 읽기를 시도하면 " 잘못된 데이터 유형 " 에러가 출력됩니다 (에러 변수의 리턴 값 = 4). 다음 유형의 파일을 읽을 수 없습니다 . _BIN, _EXE, _OBJ, _LIB, _BOT, _TRC, _ACC, _CYC, _NCK.

예제

프로그램 코드	설명
N10 DEF INT ERROR	; 에러 변수 정의 .
N20 DEF STRING[255] RESULT[5]	; 결과 변수 정의 .
N30 READ(ERROR,"/_N_CST_DIR/_N_TESTFILE_MPF",1,5,RESULT)	; 도메인 , 파일 식별자 및 경로 지정이 있는 파일 이름 .
N40 IF ERROR <>0	; 오류 평가 .
N50 MSG("ERROR"<<ERROR<<"ON READ COMMAND")	
N60 M0	
N70 ENDIF	
...	

1.21 파일의 존재 여부 점검 (ISFILE)

기능

ISFILE 명령은 NCK 의 스태틱 유저 메모리 (패시브 파일 시스템) 에 특정 파일이 존재하는지 여부를 확인하는 데 사용됩니다 .

구문

< 결과 >=ISFILE("<file name(파일 이름 >")

의미

ISFILE: 지정된 파일이 패시브 파일 시스템에 존재하는지 여부를 확인하는 명령어 .

< 파일 이름 >: 패시브 파일 시스템에 존재하는지 여부를 확인할 파일의 이름 .

유형 : STRING

파일 이름을 지정할 때 다음 사항을 고려해야 합니다 .

- 파일 이름에 공백 또는 제어 문자 (32 자 이하의 ASCII 코드를 가진 문자) 가 포함되면 안됩니다 .
- 경로 및 파일 식별자로 파일 이름을 지정할 수 있습니다 .
  - 경로 데이터

    경로 데이터는 절대값이어야 하므로 "/" 로 시작합니다 .

    지정된 경로가 없는 경우 현재 디렉토리 (선택한 프로그램의 디렉토리) 에서 해당 파일을 검색합니다 .
  - 파일 식별자

    파일 이름에 도메인 이름 ("_N_") 이 없는 경우 추가됩니다 .

    파일 이름의 끝에서 네 번째 문자가 밑줄 ("_") 인 경우 그 다음 세 문자가 파일 식별자로 해석됩니다 . STRING 유형의 변수를 이용해 모든 파일 명령에 동일한 파일 이름을 사용하는 경우 _SPF 및 _MPF 만 파일 식별자로 사용할 수 있습니다 .

    "_MPF" 또는 "_SPF" 와 같은 파일 식별자가 없는 경우 파일 이름은 자동적으로 _MPF 로 끝나게 됩니다 .
- 파일 이름 길이는 최대 32 바이트이며 경로 길이는 최대 128 바이트입니다 .

예제 :

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

< 결과 >: 점검 결과를 지정할 결과 변수 .

유형 . BOOL

값 :		
	TRUE	파일이 있음
	FALSE	파일이 없음

예제

프로그램 코드	설명
N10 DEF BOOL RESULT	; 결과 변수 정의 .
N20 RESULT=ISFILE("TESTFILE")	
N30 IF(RESULT==FALSE)	
N40 MSG("FILE DOES NOT EXIST")	
N50 M0	
N60 ENDIF	
...	

or:

프로그램 코드	설명
N10 DEF BOOL RESULT	; 결과 변수 정의 .
N20 RESULT=ISFILE("TESTFILE")	
N30 IF(NOT ISFILE("TESTFILE"))	
N40 MSG("FILE DOES NOT EXIST")	
N50 M0	
N60 ENDIF	
...	

1.22 파일 정보 읽기 (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO)

기능

FILEDATE, FILETIME, FILESIZE, FILESTAT 및 FILEINFO 명령은 최종 쓰기 액세스 날짜 및 시간 , 해당 파일 크기 및 / 또는 파일 상태와 같은 특정 파일 정보를 읽어오는 데 사용합니다 .

주

정보를 읽을 파일은 NCK 의 스태틱 유저 메모리 (패시브 파일 시스템) 에 저장되어 있어야 합니다 .

조건

현재 설정된 보호 레벨은 파일의 상위 디렉토리의 표시 권한과 동일하거나 커야 합니다 .
그렇지 않을 경우 에러 메시지 (에러 변수의 반환값 = 13) 와 함께 액세스가 거부됩니다 .

구문

DEF INT < 에러 >
DEF STRING[< 문자열 길이 >] < 결과 >
FILE....(< 에러 >,"<file name(파일 이름)>",< 결과 >)

의미

FILEDATE:	FILEDATE 명령은 지정된 파일에 마지막으로 쓰기 액세스한 **날짜**를 리턴합니다 .
FILETIME:	FILETIME 명령은 지정된 파일에 마지막으로 쓰기 액세스한 **시간**을 리턴합니다 .
FILESIZE:	FILESIZE 명령은 지정된 파일의 **현재 크기**를 리턴합니다 .
FILESTAT:	FILESTAT 명령은 지정된 파일의 읽기 권한 , 쓰기 권한 및 실행 권한과 관련된 **상태**를 리턴합니다 .
FILEINFO:	FILEINFO 명령은 FILEDATE, FILETIME, FILESIZE 및 FILESTAT 명령으로 읽어올 수 있는 특정 파일에 대한 **모든 파일 정보**를 리턴합니다 .
< 에러 >:	에러 값 리턴을 위한 변수 (기준 파라미터에 의한 호출)

유형 .	INT	
값 :	0	오류 없음
	1	경로 허용 안됨
	2	경로 찾을 수 없음
	3	파일 찾을 수 없음
	4	잘못된 파일 유형
	13	액세스 권한 부족
	22	결과 변수 (< 결과 >) 의 문자열 길이가 너무 작음 .

< 파일 이름 >: 파일 정보를 읽어올 파일의 이름 .

유형 : STRING

파일 이름을 지정할 때 다음 사항을 고려해야 합니다 .

- 파일 이름에 공백 또는 제어 문자 (32 자 이하의 ASCII 코드를 가진 문자) 가 포함되면 에러 코드 1 "경로 허용 안 됨"이 표시되면서 FILE... 명령이 취소됩니다 .
- 경로 및 파일 식별자로 파일 이름을 지정할 수 있습니다 .
  - 경로 데이터

    경로 데이터는 절대값이어야 하므로 "/" 로 시작합니다 .

    지정된 경로가 없는 경우 현재 디렉토리 (선택한 프로그램의 디렉토리) 에서 해당 파일을 검색합니다 .
  - 파일 식별자

    파일 이름에 도메인 이름 ("_N_") 이 없는 경우 추가됩니다 .

    파일 이름의 끝에서 네 번째 문자가 밑줄 ("_") 인 경우 그 다음 세 문자가 파일 식별자로 해석됩니다 . STRING 유형의 변수를 이용해 모든 파일 명령에 동일한 파일 이름을 사용하는 경우 _SPF 및 _MPF 만 파일 식별자로 사용할 수 있습니다 .

    "_MPF" 또는 "_SPF" 와 같은 파일 식별자가 없는 경우 파일 이름은 자동적으로 _MPF 로 끝나게 됩니다 .
- 파일 이름 길이는 최대 32 바이트이며 경로 길이는 최대 128 바이트입니다 .

예제 :

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

< 결과 >: 결과 변수 (기준별 호출 파라미터)
요청한 파일 정보를 저장하는 변수 .

유형 : STRING 대상 : FILEDATE
유형 : "dd.mm.yy"
⇒ 문자열 길이는 반드시 8 이어야 합니다 .

FILETIME
유형 : "hh:mm.ss"
⇒ 문자열 길이는 반드시 8 이어야 합니다 .

FILESTAT
유형 : "rwxsd"
(r: read, w: write, x: execute, s: show, d: delete)
⇒ 문자열 길이는 반드시 5 여야 합니다 .

FILEINFO
유형 : "rwxsd nnnnnnnn dd.mm.yy hh:mm:ss"
⇒ 문자열 길이는 반드시 32 여야 합니다 .

INT 대상 : FILESIZE
파일 크기는 바이트로 출력됩니다 .

예제

프로그램 코드	설명
N10 DEF INT ERROR	; 에러 변수 정의 .
N20 STRING[32] RESULT	; 결과 변수 정의 .
N30 FILEINFO(ERROR,"/_N_MPF_DIR/_N_TESTFILE_MPF",RESULT)	; 도메인 , 파일 식별자 및 경로 지정이 있는 파일 이름 .
N40 IF ERROR <>0	; 오류 평가
N50 MSG("ERROR"<<ERROR<<"ON FILEINFO COMMAND")	
N60 M0	
N70 ENDIF	
...	

예를 들어 RESULT 결과 변수 형식으로 다음과 같은 결과를 반환할 수 있습니다 .

"77777 12345678 26.05.00 13:51:30"

1.23 배열을 이용한 체크섬 계산 (CHECKSUM)

기능

CHECKSUM 명령은 배열을 이용해 체크섬을 계산하는 데 사용됩니다 . 이 체크섬을 이전 체크섬 계산과 비교하여 배열 데이터의 변경 여부를 확인할 수 있습니다 .

적용

스톡 제거 동안 초기 형상이 변경되었는지 여부를 확인합니다 .

구문

```
DEF INT < 에러 >
DEF STRING[< 문자열 길이 >] < 체크섬 >
DEF ... < 배열 >[<n>,<m>,<o>]
< 에러 >=CHECKSUM(< 체크섬 >,"<array( 배열 )>"[,< 시작 열 >,< 종료 열 >])
```

의미

CHECKSUM: 배열을 이용해 체크섬을 계산하는 명령어 .

< 에러 >: 에러 값을 반환하기 위한 변수 .

유형 . INT

값 :

0	에러 없음
1	기호를 찾을 수 없음
2	배열 없음
3	인덱스 1 이 너무 큼
4	인덱스 2 가 너무 큼
5	잘못된 데이터 유형
10	체크섬 오버플로우

< 체크섬 >: 체크섬 계산 결과를 기록할 결과 변수 (기준별 호출 파라미터)

유형 : STRING

필요한 문자열 길이 : 16

체크섬은 16 진수 숫자의 문자 문자열로 표시됩니다 . 그러나 형식 문자는 표시되지 않습니다 .

예제 : "A6FC3404E534047C"

< 배열 >:	체크섬을 생성하는 데 사용할 배열의 이름 (파라미터에 의한 호출) 유형 : STRING 최대 문자열 길이 : 32 다음과 같은 유형의 1 ~ 3 차원 배열을 사용할 수 있습니다 . BOOL, CHAR, INT, REAL, STRING **참고 :** 머신 데이터의 배열은 사용할 수 없습니다 .
< 시작 열 >:	체크섬 계산에 이용할 배열의 시작 열 번호 (옵션 : 파라미터)
< 종료 열 >:	체크섬 계산에 이용할 배열의 종료 열 번호 (옵션 : 파라미터)

주

< 시작 열 > 및 < 종료 열 > 파라미터 지정 여부는 옵션입니다 . 열 인덱스가 표시되지 않는 경우 체크섬은 전체 배열을 통해 구성됩니다 .

체크섬 결과는 항상 제한적입니다 . 배열 요소가 변경되면 결과 문자열도 변경됩니다 .

예제

프로그램 코드	설명
N10 DEF INT ERROR	; 에러 변수 정의 .
N20 DEF STRING[16] MY_CHECKSUM	; 결과 변수 정의 .
N30 DEF INT MY_VAR[4,4]	; 배열 정의 .
N40 MY_VAR=...	
N50 ERROR=CHECKSUM(MY_CHECKSUM,"MY_VAR",0,2)	
...	

예의 경우 MY_CHECKSUM 결과 변수 형식으로 다음과 같은 결과를 반환할 수 있습니다 .

"A6FC3404E534047C"

1.24 반올림 (ROUNDUP)

기능

실수 (REAL) 유형의 입력 값 (소수점을 가진 실수) 은 "ROUNDUP" 기능을 사용하여 한 단계 높은 정수로 반올림할 수 있습니다 .

구문

ROUNDUP(<value>)

의미

ROUNDUP:	입력 값을 반올림하기 위한 명령
< 값 >:	유형 입력 값

주

정수 (INTEGER) 형식의 입력값은 변경되지 않은 상태로 리턴됩니다 .

예제

예제 1: 다양한 입력 값 및 반올림 결과

예제	반올림 결과
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

예제 2: NC 프로그램에서 ROUNDUP

프로그램 코드

```
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 서브프로그램 방법

1.25.1 일반 정보

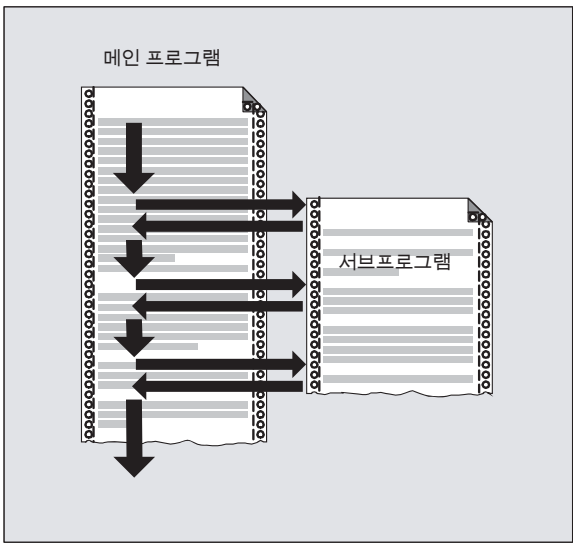
1.25.1.1 서브프로그램

기능

" 서브프로그램 " 은 가공 프로그램을 메인 프로그램과 서브프로그램으로 나누던 시절에 만들어진 용어입니다 . 그 당시 메인 프로그램은 제어 시스템 상에서의 처리를 위해 선택하고 실행시키는 가공 프로그램이었습니다 . 서브프로그램은 메인 프로그램 내에서 호출하는 가공 프로그램이었습니다 .

오늘날의 SINUMERIK NC 언어에서는 이처럼 엄격하게 두 프로그램을 구분하지 않습니다 . 원칙적으로 각 가공 프로그램은 메인 프로그램으로 선택이 가능하며 다른 메인 프로그램에서 서브프로그램으로 실행 또는 호출할 수도 있습니다 .

따라서 다른 가공 프로그램에서 호출한 가공 프로그램을 서브프로그램이라고 부를 수 있습니다 .



적용

모든 고급 프로그래밍 언어가 그렇듯이 NC 언어에서도 1 회 이상 사용된 프로그램 섹션을 독립적인 자급자족식 프로그램으로 바꾸는 데 서브프로그램을 사용합니다 .

서브프로그램을 사용하면 다음과 같은 이점이 있습니다 .

- 프로그램을 읽기 쉽고 간단하게 할 수 있습니다 .
- 검증된 프로그램 일부를 재사용함으로써 품질을 향상할 수 있습니다
- 특정 가공 라이브러리 생성 기능을 제공합니다 .
- 메모리 공간 절약

1.25.1.2 서브프로그램 이름

이름 지정 규칙

서브프로그램의 이름을 지정할 때 다음 규칙을 반드시 준수해야 합니다 .

- 첫 두 문자는 글자 (A~Z, a~z) 여야 합니다 .
- 그 다음 문자는 글자 , 숫자 (0~9) 및 밑줄 ("_") 을 조합해 지정할 수 있습니다 .
- 최대 31 자까지 사용할 수 있습니다 .

주

SINUMERIK NC 언어는 대문자와 소문자를 구분하지 **않습니다** .

프로그램 이름 확장

프로그램이 생성될 때 지정된 프로그램 이름을 제어 시스템에서 접두어 및 접미어를 추가해 확장할 수 있습니다 .

- 접두어 : _N_
- 접미어 :
  - 메인 프로그램 : _MPF
  - 서브프로그램 : _SPF

프로그램 이름 사용

서브프로그램 호출 컨텍스트에서 프로그램 이름을 사용할 때 접두어 , 프로그램 이름 및 접미어를 어떤 식으로든 조합할 수 있습니다 .

예제 :

다음의 호출을 사용해 프로그램 이름이 "SUB_PROG" 인 서브프로그램을 시작할 수 있습니다 .

1. SUB_PROG
2. _N_SUB_PROG
3. SUB_PROG_SPF
4. _N_SUB_PROG_SPF

주

이름이 같은 메인 프로그램과 서브프로그램

메인 프로그램 (.MPF) 과 서브프로그램 (.SPF) 의 이름이 같은 경우 , 이 프로그램 이름을 가공 프로그램에서 사용할 때 어느 프로그램인지 분명히 식별할 수 있도록 해당 접미어를 반드시 지정해야 합니다 .

1.25.1.3 서브프로그램 중첩

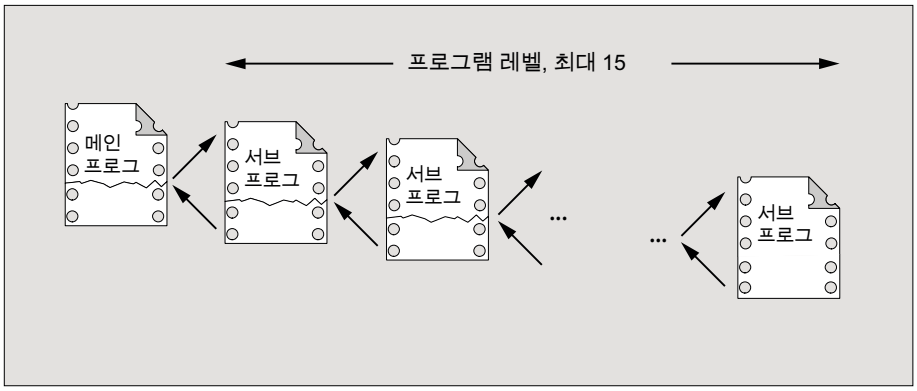
메인 프로그램은 서브프로그램을 호출할 수 있고 , 호출된 서브프로그램은 다시 더 많은 서브프로그램을 호출할 수 있습니다 . 따라서 여러 프로그램의 순서가 중첩됩니다 . 각 프로그램은 전용 프로그램 레벨에서 실행됩니다 .

중첩 깊이

NC 언어는 현재 16 개의 프로그램 레벨을 제공합니다 . 메인 프로그램은 항상 최상위 프로그램 레벨인 0 에서 실행됩니다 . 서브프로그램은 호출한 프로그램의 레벨 바로 아래 프로그램 레벨에서 실행됩니다 . 따라서 첫 번째 서브프로그램의 레벨은 프로그램 레벨 1 이 됩니다 .

프로그램 레벨 구분 :

- 프로그램 레벨 0: 메인 프로그램 레벨
- 프로그램 레벨 1 ~ 15: 서브프로그램 레벨 1 ~ 15



인터럽트 루틴 (ASUB)

서브프로그램이 인터럽트 루틴 컨텍스트에서 호출된 경우 , 해당 서브프로그램은 채널 내에 현재 활성화된 프로그램 레벨 (n) 에서 실행되지 않고 그 아래 프로그램 레벨 (n+1) 에서 실행됩니다 . 가장 낮은 프로그램 레벨에서도 이런 식으로 실행될 가능성이 있기 때문에 인터럽트 루틴에서는 2 개의 프로그램 레벨 (레벨 16 및 17) 을 추가로 사용할 수 있습니다 .

2 개 이상의 프로그램 레벨이 필요한 경우 채널에서 실행되는 가공 프로그램을 설정할 때 명시적으로 지정해야 합니다 . 즉 , 지정된 최대 프로그램 레벨까지만 사용할 수 있도록 해서 인터럽트 처리에 사용할 프로그램 레벨을 충분히 남겨두어야 합니다 .

예를 들어 인터럽트 처리에 4 개의 프로그램 레벨이 필요한 경우 최대 프로그램 레벨 13 까지 사용할 수 있도록 가공 프로그램을 설정해야 합니다 . 그래야만 인터럽트 발생 시 인터럽트 처리에 필요한 4 개 프로그램 레벨 (레벨 14~17) 을 가공 프로그램이 사용할 수 있습니다 .

Siemens 사이클

Siemens 사이클은 3 개의 프로그램 레벨을 필요로 합니다 . 따라서 Siemens 사이클은 마지막으로 호출되어야 합니다 .

- 가공 프로그램 처리 : 프로그램 레벨 12
- 인터럽트 루틴 : 프로그램 레벨 14

1.25.1.4 검색 경로

경로를 지정하지 않고 서브프로그램을 호출하면 제어 시스템은 표시된 순서대로 다음의 디렉토리를 검색합니다 .

순서	디렉토리	설명
1.	현재 디렉토리	호출 프로그램이 위치한 디렉토리
2.	/_N_SPF_DIR /	전역 서브프로그램 디렉토리
3.	/_N_CUS_DIR /	사용자 사이클
4.	/_N_CMA_DIR /	제조업체 사이클
5.	/_N_CST_DIR /	표준 사이클

1.25.1.5 유형 파라미터 및 실제 파라미터

유형 파라미터 및 실제 파라미터는 파라미터 전송이 포함된 서브프로그램의 정의 및 호출과 관련된 파라미터입니다 .

유형 파라미터

서브프로그램을 정의할 때 서브프로그램으로 전송될 파라미터 (유형 파라미터) 를 해당 파라미터의 유형 및 파라미터 이름과 함께 정의해야 합니다 .

즉 , 유형 파라미터가 서브프로그램의 인터페이스를 정의하게 됩니다 .

예제 :

프로그램 코드	설명
PROC CONTOUR (REAL X, REAL Y)	; 유형 파라미터 : X 및 Y 는 실수 형식
N20 X1=X Y1=Y	; 축 X1 을 X 위치로 축 Y1 을 Y 위치로 이송
...	
N100 RET	

실제 파라미터

서브프로그램을 호출할 때 절대값 혹은 절대 변수 (실제 파라미터) 를 해당 서브프로그램에 전송해야 합니다 .

그에 따라 실제 파라미터는 서브프로그램이 호출될 때 해당 서브프로그램의 인터페이스에 가장 최근의 값을 지정합니다 .

예제 :

프로그램 코드	설명
N10 DEF REAL WIDTH	; 변수 정의
N20 WIDTH=20.0	; 변수 할당
N30 CONTOUR(5.5, WIDTH)	; 실제 파라미터를 포함한 서브프로그램 호출 : 5.5 및 WIDTH
...	
N100 M30	

1.25.1.6 파라미터 전송

파라미터 전송이 포함된 서브프로그램 정의

파라미터 전송이 포함된 서브프로그램은 PROC 키워드와 서브프로그램에서 요구하는 모든 파라미터가 포함된 완전한 파라미터 목록을 사용해 정의합니다 .

불완전한 파라미터 전송

서브프로그램이 호출된 경우 해당 서브프로그램 인터페이스에 정의된 모든 파라미터를 명시적으로 전송할 필요는 없습니다 . 전송에서 제외된 파라미터 대신 기본값 "0" 이 전송됩니다 .

단 , 파라미터 순서는 분명하게 식별이 가능해야 하기 때문에 쉼표를 파라미터 구분자로 항상 포함시켜야 합니다 . 마지막 파라미터는 예외입니다 . 마지막 파라미터가 호출에서 제외되면 마지막 쉼표 역시 전송되지 않습니다 .

예제 :

서브프로그램 :

프로그램 코드	설명
PROC SUB_PROG (REAL X, REAL Y, REAL Z)	; 유형 파라미터 : X, Y 및 Z
...	
N100 RET	

메인 프로그램 :

프로그램 코드	설명
PROC MAIN_PROG	
...	
N30 SUB_PROG(1.0,2.0,3.0)	; 완전한 파라미터 전송이 포함된 서브프로그램 호출 : X=1.0, Y=2.0, Z=3.0
…	
N100 M30	

N30 에서 불완전한 파라미터 전송을 포함한 서브프로그램을 호출한 예 :

```
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
```

주의

기준별 호출 파라미터 전송

기준별 호출을 통해 전송되는 파라미터는 서브프로그램 호출 시 함께 전송되어야 합니다 .

주의
AXIS 데이터 유형 AXIS 데이터 유형의 파라미터는 서브프로그램 호출 시 함께 전송되어야 합니다 .

전송 파라미터 점검

시스템 변수 $P_SUBPAR [n] (여기서 n = 1, 2,...) 을 사용해 파라미터가 명시적으로 전송되었는지 아니면 서브프로그램에 남겨졌는지 점검할 수 있습니다 . 인덱스 n 은 유형 파라미터의 순서를 나타냅니다 . 인덱스 n = 1 은 첫 번째 유형 파라미터를 , 인덱스 n = 2 는 두 번째 유형 파라미터를 나타내며 그 다음도 이런 식으로 계속됩니다 .

다음 예는 프로그램의 일부를 발췌한 것으로 첫 번째 유형 파라미터를 기준으로 어떻게 점검이 실행되는지 보여줍니다 .

프로그래밍	설명
PROC SUB_PROG (REAL X, REAL Y, REAL Z)	; 유형 파라미터 : X, Y 및 Z
N20 IF $P_SUBPAR[1]==TRUE	; 첫 번째 유형 파라미터 X 를 점검합니다 .
...	; 유형 파라미터 X 가 명시적으로 전송된 경우 이 동작들이 실행됩니다 .
N40 ELSE	
...	; 유형 파라미터 X 가 명시적으로 전송되지 않은 경우 이 동작들이 실행됩니다 .
N60 ENDIF	
...	; 일반 동작
N100 RET	

1.25.2 서브프로그램 정의

1.25.2.1 파라미터 전송이 없는 서브프로그램

기능

파라미터 전송이 없는 서브프로그램을 정의할 때는 프로그램의 시작에서 정의 라인을 생략할 수 있습니다 .

구문

```
[PROC < 프로그램 이름 >]
...
```

의미

PROC:	프로그램 시작에 정의 연산
< 프로그램 이름 >:	프로그램 이름

예제

예제 1: PROC 연산을 포함한 서브프로그램

프로그램 코드	설명
PROC SUB_PROG	; 정의 라인
N10 G01 G90 G64 F1000	
N20 X10 Y20	
...	
N100 RET	; 서브프로그램 복귀

예제 2: PROC 연산이 없는 서브프로그램

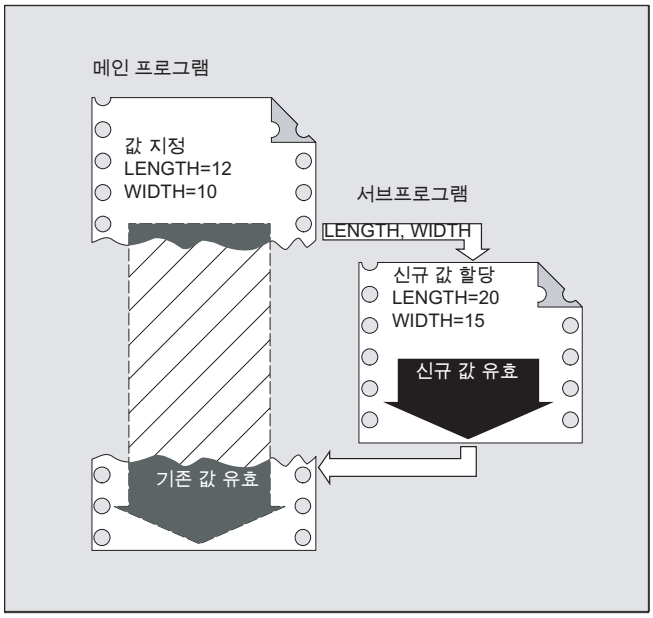
프로그램 코드	설명
N10 G01 G90 G64 F1000	
N20 X10 Y20	
...	
N100 RET	; 서브프로그램 복귀

1.25.2.2 값별 호출 파라미터 전송을 포함한 서브프로그램 (PROC)

기능

값별 호출 파라미터 전송이 포함된 서브프로그램은 PROC 키워드 , 프로그램 이름 , 서브프로그램에서 요구하는 모든 파라미터와 그 유형 및 이름이 포함된 완전한 파라미터 목록 순서로 정의합니다 . 정의 연산은 첫 번째 프로그램 라인에 위치해야 합니다 .

값별 호출 파라미터 전송은 호출 프로그램에 영향을 미치지 않습니다 . 호출 프로그램은 실제 파라미터의 값만 서브프로그램에 전송합니다 .



주

최대 127 개의 파라미터를 전송할 수 있습니다 .

구문

PROC < 프로그램 이름 > (< 파라미터 유형 > < 파라미터 이름 > 등)

의미

PROC:	프로그램 시작에 정의 연산
< 프로그램 이름 >:	프로그램 이름
< 파라미터 유형 >:	파라미터의 데이터 유형 (예 : REAL, INT, BOOL)
< 파라미터 이름 >:	파라미터 이름

유의사항
PROC 키워드 다음에 지정된 프로그램 이름은 사용자 인터페이스 상에서 지정된 프로그램 이름과 일치해야 합니다 .

예제

2 개의 REAL 유형 파라미터가 포함된 서브프로그램 정의

프로그램 코드	설명
PROC SUB_PROG (REAL LENGTH, REAL WIDTH)	; 파라미터 1: 유형 : REAL, 이름 : LENGTH 파라미터 2: 유형 : REAL, 이름 : WIDTH
...	
N100 RET	; 서브프로그램 복귀

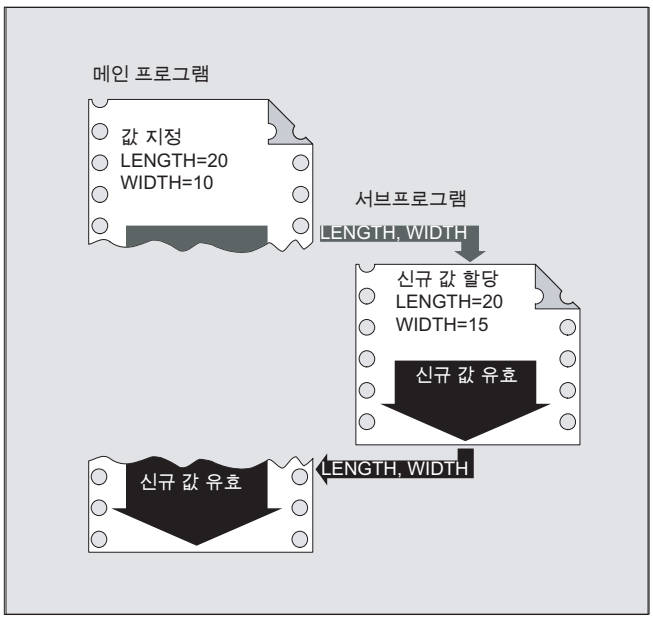
1.25.2.3 기준별 호출 파라미터 전송을 포함한 서브프로그램 (PROC, VAR)

기능

기준별 호출 파라미터 전송이 포함된 서브프로그램은 PROC 키워드 , 프로그램 이름 , 완전한 파라미터 목록 순서로 정의합니다 . 파라미터 목록에는 서브프로그램에서 요구하는 모든 파라미터 및 VAR 키워드와 함께 해당 파라미터의 유형 및 이름이 포함됩니다 . 정의 연산은 첫 번째 프로그램 라인에 위치해야 합니다 .

기준별 호출 파라미터 전송을 통해 배열 기준도 전송할 수 있습니다 .

기준별 호출 파라미터 전송은 호출 프로그램에 영향을 미칩니다 . 호출 프로그램은 실제 파라미터에 대한 기준을 서브프로그램에 전송하며 따라서 서브프로그램에 해당 변수에 대한 직접 액세스를 제공합니다 .



주

최대 127 개의 파라미터를 전송할 수 있습니다 .

주

전송할 변수가 호출 프로그램에 정의되어 있는 경우 (LUD) 에만 기준별 호출 파라미터를 전송합니다 . 채널 전역 변수 또는 NC 전역 변수는 서브프로그램 내에서 직접 액세스할 수 없기 때문에 전송할 필요가 없습니다 .

구문

PROC < 프로그램 이름 > (VAR < 파라미터 유형 > < 파라미터 이름 > 등)
PROC < 프로그램 이름 > (VAR < 배열 유형 > < 배열 이름 >, [<m>,<n>,<o>] 등)

의미

PROC:	프로그램 시작에 정의 연산
VAR:	기준별 파라미터 전송 키워드
< 프로그램 이름 >:	프로그램 이름
< 파라미터 유형 >:	파라미터의 데이터 유형 (예 : REAL, INT, BOOL)
< 파라미터 이름 >:	파라미터 이름
< 배열 유형 >:	배열 요소의 데이터 유형 (예 : REAL, INT, BOOL)
< 배열 이름 >:	배열 이름
[<m>,<n>,<o>]:	배열 크기 현재 최대 3 차원 배열까지 사용 가능 <m>: 1 차원 배열 크기 <n>: 2 차원 배열 크기 <o>: 3 차원 배열 크기

유의사항
PROC 키워드 다음에 지정된 프로그램 이름은 사용자 인터페이스 상에서 지정된 프로그램 이름과 일치해야 합니다 .

주

정의되지 않은 배열 길이의 배열이 있는 경우 서브프로그램은 변수 길이의 배열을 유형 파라미터로 처리할 수 있습니다 . 예를 들어 2 차원 배열을 유형 파라미터로 정의하는 경우 1 차원 배열의 길이가 지정되지 않습니다 . 이 경우 쉼표를 대신 표시해야 합니다 .

예제 : PROC < 프로그램 이름 > (VAR REAL ARRAY[,5])

예제

REAL 유형의 파라미터 2 개를 이용한 서브프로그램을 정의 :

프로그램 코드	설명
PROC SUB_PROG(VAR REAL LENGTH, VAR REAL WIDTH)	; 파라미터 1: 유형 기준 : REAL, 이름 : LENGTH 파라미터 2: 유형 기준 : REAL, 이름 : WIDTH
...	
N100 RET	

1.25.2.4 모달 G 코드 저장 (SAVE)

작동

SAVE 특성은 서브프로그램 호출 이전에 활성 모달 G 코드가 저장되고 서브프로그램 종료 후 다시 활성화됨을 의미합니다 .

주의
연속 경로 모드 중단 연속 경로 모드를 실행 중인 상태에서 SAVE 속성으로 서브프로그램을 호출하면 서브프로그램 끝에서 연속 경로 모드가 중단됩니다 (리턴 점프).

구문

PROC <subprogram name> SAVE

의미

SAVE: 서브프로그램 호출 이전에 모달 G 코드를 저장하고 서브프로그램 종료 후 복원합니다 .

예제

모달 G 코드 G91 은 CONTOUR 서브프로그램에서 유효합니다 (증분 치수). 모달 G 코드 G90 은 메인 프로그램에서 유효합니다 (절대 치수). SAVE 를 정의한 서브프로그램으로 인해 서브 프로그램 종료 후 메인 프로그램에서 다시 G90 이 유효합니다 .

서브프로그램 정의 :

프로그램 코드	설명
PROC CONTOUR (REAL VALUE1) SAVE	; SAVE 파라미터를 통한 서브프로그램 정의
N10 G91 ...	; 모달 G 코드 G91: 증분 치수
N100 M17	; 서브프로그램 종료

메인 프로그램 :

프로그램 코드	설명
N10 G0 X... Y... G90	; 모달 G 코드 G90: 절대 좌표
N20 ...	
...	
N50 CONTOUR (12.4)	; 서브프로그램 호출
N60 X... Y...	; 모달 G 코드 G90 이 SAVE 를 사용하여 재활성화됨

일반 조건

프레임

SAVE 특성을 통한 서브프로그램과 관련된 프레임 동작은 프레임 시간에 따라 다르며 머신 데이터를 사용하여 설정할 수 있습니다 .

참고 자료

기능 매뉴얼 . 기본 기능 ; 축 , 좌표계 , 프레임 (K2),
장 : "SAVE 로 서브프로그램 반환 "

1.25.2.5 싱글 블록 실행 억제 (SBLOF, SBLON)

기능

전체 프로그램을 위한 싱글 블록 억제

예를 들어 SBLOF 가 지정된 프로그램은 싱글 블록 실행이 활성화되어도 한 블록처럼 실행됩니다 , 즉 싱글 블록이 무시됩니다 .

SBLOF 는 PROC 라인에 위치하며 서브프로그램이 종료되거나 중단될 때까지 유효합니다 . 리턴 명령을 통해 서브프로그램 종료 시 중단할지 여부를 결정합니다 .

M17 을 통한 리턴 점프 :	서브프로그램 종료 시 중단
RET 를 통한 리턴 점프 :	서브프로그램 종료 시 중단 없음

프로그램 내에서 싱글 블록 억제

SBLOF 는 한 블록에 사용해야 합니다 . 싱글 블록은 다음 시점까지 이 블록 이후에 비활성화됩니다 .

- 다음 SBLON 까지

  또는

- 활성 서브프로그램 레벨 종료 시까지

구문

전체 프로그램을 위한 싱글 블록 억제 :
PROC ... SBLOF

프로그램 내에서 싱글 블록 억제 :

```
SBLOF
...
SBLON
```

의미

PROC:	프로그램의 첫 번째 명령
SBLOF:	싱글 블록 실행 비활성화를 위한 명령 SBLOF 는 PROC 블록 또는 특정 블록에만 기록할 수 있습니다 .
SBLON:	싱글 블록 실행 활성화를 위한 명령 SBLON 은 개별 블록에 있어야 합니다 .

제한사항

- **싱글 블록 억제 및 블록 디스플레이**

  현재 블록 디스플레이는 DISPLOF 를 사용하여 사이클 / 서브프로그램에서 억제할 수 있습니다 . DISPLOF 가 SBLOF 와 함께 프로그래밍되는 경우 사이클 / 서브프로그램 호출이 사이클 / 서브프로그램 내에서 싱글 블록 정지 시 계속 표시됩니다 .

- **시스템 ASUB 또는 사용자 ASUB 에서 싱글 블록 억제**

  시스템 또는 사용자 ASUB 에서 싱글 블록 정지가 머신 데이터 MD10702 $MN_IGNORE_SINGLEBLOCK_MASK (비트 0 = 1 또는 비트 1 = 1) 의 설정을 사용하여 억제된 경우 ASUB 에서 SBLON 을 프로그래밍하여 싱글 블록 정지를 다시 활성화할 수 있습니다 .

  사용자 ASUB 에서의 싱글 블록 정지가 머신 데이터 MD20117 $MC_IGNORE_SINGLEBLOCK_ASUP 의 설정을 사용하여 억제된 경우 ASUB 에서 SBLON 을 프로그래밍하여 싱글 블록 정지를 다시 활성화할 수 **없습니다** .

- **다양한 싱글 블록 실행 유형의 싱글 블록 억제에 대한 특별한 기능**

  싱글 블록 실행 SBL2 가 활성화된 경우 (각 가공 프로그램 블록 이후 정지) MD10702 $MN_IGNORE_SINGLEBLOCK_MASK 에서 비트 12 가 "1" 로 설정된 경우 SBLON 블록에서 실행 정지는 **없습니다** .

  싱글 블록 실행 SBL3 이 활성화되면 (각 가공 프로그램 블록 이후 정지 및 사이클 내에서 정지) SBLOF 명령이 억제됩니다 .

예제

예제 1: 프로그램 내에서 싱글 블록 억제

프로그램 코드	설명
N10 G1 X100 F1000	
N20 SBLOF	; 싱글 블록 비활성화
N30 Y20	
N40 M100	
N50 R10=90	
N60 SBLON	; 싱글 블록 다시 활성화
N70 M110	
N80 ...	

N20 및 N60 사이 영역이 싱글 블록 모드에서 하나의 단계로 실행됩니다 .

예제 2: 사이클은 사용자를 위한 하나의 명령으로 실행

메인 프로그램 :

프로그램 코드
N10 G1 X10 G90 F200
N20 X-4 Y6
N30 CYCLE1
N40 G1 X0
N50 M30

사이클 CYCLE1:

프로그램 코드	설명
N100 PROC CYCLE1 DISPLOF SBLOF	; 싱글 블록 억제
N110 R10=3*SIN(R20)+5	
N120 IF (R11 <= 0)	
N130 SETAL(61000)	
N140 ENDIF	
N150 G1 G91 Z=R10 F=R11	
N160 M17	

CYCLE1 은 활성 싱글 블록 실행을 위해 수행됩니다 . 예를 들어 CYCLE1 을 실행하려면 시작 키를 눌러야 합니다 .

예제 3:

수정된 워크 옵셋 및 공구 옵셋을 활성화하기 위해 PLC 에 의해 시작된 ASUB 는 보이지 않게 실행되어야 합니다 .

프로그램 코드

```
N100 PROC ZO 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
```

예제 4: MD10702 비트 12 = 1 상태에서 정지되지 않음

초기 상황 :

- 싱글 블록 실행이 활성화됩니다 .
- MD10702 $MN_IGNORE_SINGLEBLOCK_MASK Bit12 = 1

메인 프로그램 :

프로그램 코드	설명
N10 G0 X0	; 이 가공 프로그램 라인에서 정지됩니다 .
N20 X10	; 이 가공 프로그램 라인에서 정지됩니다 .
N30 CYCLE	; 사이클에 의해 생성된 이송 블록 .
N50 G90 X20	; 이 가공 프로그램 라인에서 정지됩니다 .
M30	

사이클 CYCLE:

프로그램 코드	설명
PROC CYCLE SBLOF	; 싱글 블록 정지 억제 :
N100 R0 = 1	
N110 SBLON	; MD10702 bit12=1 에 의해 가공 프로그램 라인에서 실행이 정지되지 않습니다 .
N120 X1	; 이 가공 프로그램 라인에서 실행이 정지됩니다 .
N140 SBLOF	
N150 R0 = 2	
RET	

예제 5: 프로그램 중첩에 대해 싱글 블록 억제

초기 상황 :

싱글 블록 실행이 활성화됩니다 .

프로그램 중첩 :

프로그램 코드	설명
N10 X0 F1000	; 이 블록에서 실행이 정지됩니다 .
N20 UP1(0)	
PROC UP1(INT _NR) SBLOF	; 싱글 블록 정지를 억제합니다 .
N100 X10	
N110 UP2(0)	
PROC UP2(INT _NR)	
N200 X20	
N210 SBLON	; 싱글 블록 정지를 활성화합니다 .
N220 X22	; 이 블록에서 실행이 정지됩니다 .
N230 UP3(0)	
PROC UP3(INT _NR)	
N300 SBLOF	; 싱글 블록 정지를 억제합니다 .
N305 X30	
N310 SBLON	; 싱글 블록 정지를 활성화합니다 .
N320 X32	; 이 블록에서 실행이 정지됩니다 .
N330 SBLOF	; 싱글 블록 정지를 억제합니다 .
N340 X34	
N350 M17	; SBLOF 가 활성화됩니다 .
N240 X24	; 이 블록에서 실행이 정지됩니다 . SBLON 이 활성화됩니다 .
N250 M17	; 이 블록에서 실행이 정지됩니다 . SBLON 이 활성화됩니다 .
N120 X12	
N130 M17	; 이 리턴 점프 블록에서 실행이 정지됩니다 . PROC 명령의 SBLOF 가 활성화됩니다 .
N30 X0	; 이 블록에서 실행이 정지됩니다 .
N40 M30	; 이 블록에서 실행이 정지됩니다 .

그 밖의 정보

비동기화된 서브프로그램에 대해 단일 블록 비활성화

ASUB 를 한 번에 실행하려면 PROC 명령을 ASUB 에서 SBLOF 를 통해 프로그래밍해야 합니다 . 또한 이 작업은 작동 " 편집 가능한 시스템 ASUB"(MD11610 $MN_ASUP_EDITABLE) 에도 적용됩니다 .

편집 가능한 시스템 ASUB 의 예 :

프로그램 코드	비고
N10 PROC ASUB1 SBLOF DISPLOF	
N20 IF $AC_ASUP==’H200’	
N30 RET	; 모드 변경에 대해 REPOS 없음 .
N40 ELSE	
N50 REPOSA	; 다른 모든 경우 REPOS.
N60 ENDIF	

단일 블록 모드에서 프로그램 제어

단일 블록 실행 작동 상태에서 사용자는 블록별로 가공 프로그램을 실행할 수 있습니다 . 설정 유형은 다음과 같습니다 :

- SBL1: 각 가공 작동 블록 이후 IPO 단일 블록 정지 .
- SBL2: 각 블록 이후 단일 블록 정지 .
- SBL3: 싸이클 내에서 정지 (SBLOF 명령은 SBL3 을 선택하여 억제).

프로그램 중첩에 대해 단일 블록 억제

SBLOF 가 서브프로그램의 PROC 명령에서 프로그래밍된 경우 M17 을 통한 서브프로그램 리콜 시점에서 실행이 정지됩니다 . 이를 통해 호출 프로그램의 다음 블록이 실행되는 것을 방지합니다 . SBLOF 없이 SBLOF 가 서브프로그램의 PROC 명령에서 프로그래밍되는 경우 단일 블록 억제가 활성화되고 호출 프로그램의 다음 가공 작동 블록 이후에만 실행이 정지됩니다 . 이러한 작동을 원하지 않는 경우 리콜 (M17) 이전에 서브프로그램에서 SBLON 을 프로그래밍해야 합니다 . RET 을 통해 높은 수준의 프로그램에 대한 리콜의 경우 실행이 정지되지 않습니다 .

1.25.2.6 현재 블록 디스플레이 억제 (DISPLOF, DISPLON, ACTBLOCNO)

기능

현재 프로그램 블록이 블록 디스플레이에 표준으로 표시됩니다 . DISPLOF 명령을 사용해 사이클 및 서브프로그램에서 해당 블록의 디스플레이를 억제할 수 있습니다 . 현재 블록 대신 사이클 또는 서브프로그램의 호출이 표시됩니다 . DISPLON 명령을 사용해 블록 디스플레이 억제를 취소할 수 있습니다 .

DISPLOF 및 DISPLON 은 PROC 연산으로 프로그램 라인에 프로그래밍합니다 . 또한 두 명령은 서브프로그램 전체는 물론 암시적으로 해당 서브프로그램이 호출한 다른 서브프로그램 중 DISPLON 또는 DISPLOF 명령을 포함하지 않은 모든 서브프로그램에도 적용됩니다 . 이 사항은 모든 ASUB 에 적용됩니다 .

구문

PROC … DISPLOF
PROC … DISPLOF ACTBLOCNO
PROC … DISPLON

의미

DISPLOF: 현재 블록 표시를 억제하기 위한 명령 .

위치 : PROC 연산이 포함된 프로그램 라인의 끝

적용 : 서브프로그램 혹은 프로그램 끝에서 리턴 점프까지

참고 :
DISPLOF 명령을 사용하는 서브프로그램이 다른 서브프로그램을 호출한 경우 호출된 서브프로그램 중 DISPLON 이 명시적으로 프로그래밍된 경우를 제외한 모든 서브프로그램에서도 해당 블록 디스플레이가 억제됩니다 .

DISPLON: 해당 블록 디스플레이 억제를 취소하는 명령어 .

위치 : PROC 연산이 포함된 프로그램 라인의 끝

적용 : 서브프로그램 혹은 프로그램 끝에서 리턴 점프까지

참고 :
DISPLON 명령을 사용하는 서브프로그램이 다른 서브프로그램을 호출한 경우 호출된 서브프로그램 중 DISPLOF 가 명시적으로 프로그래밍되어 있는 경우를 제외한 모든 서브프로그램에 해당 블록이 표시됩니다 .

ACTBLOCNO: DISPLOF 에 ACTBLOCNO 속성이 포함되어 있으면 알람이 발생하는 경우 해당 알람이 발생한 실제 블록의 번호가 출력됩니다 . 또한 DISPLOF 가 낮은 수준의 프로그램 레벨에서 프로그래밍되는 경우에만 적용됩니다 .

반면에 ACTBLOCNO 가 없는 DISPLOF 의 경우 DISPLOF 에서 지정하지 않은 마지막 프로그램 레벨의 사이클 또는 서브프로그램 호출에 대한 블록 번호가 표시됩니다 .

예제

예제 1: 사이클에서 현재 블록 표시 억제

프로그램 코드	설명
PROC CYCLE (AXIS TOMOV, REAL POSITION) SAVE DISPLOF	; 현재 블록 표시가 억제되고 대신 사이클 호출이 표시되어야 합니다 . 예를 들면 : CYCLE(X,100.0)
DEF REAL DIFF	; 사이클 컨텐츠
G01 ...	
...	
RET	; 서브프로그램 리턴 점프 . 사이클 호출 다음 블록이 블록 디스플레이에 표시됩니다 .

예제 2: 알람 출력에 대한 블록 표시

서브프로그램 SUBPROG1 (ACTBLOCNO 사용):

프로그램 코드	설명
PROC SUBPROG1 DISPLOF ACTBLOCNO	
N8000 R10 = R33 + R44	
...	
N9040 R10 = 66 X100	; 알람 12080 출력
...	
N10000 M17	

서브프로그램 SUBPROG2 (ACTBLOCNO 사용 안 함):

프로그램 코드	설명
PROC SUBPROG2 DISPLOF	
N5000 R10 = R33 + R44	
...	
N6040 R10 = 66 X100	; 알람 12080 출력
...	
N7000 M17	

메인 프로그램 :

프로그램 코드	설명
N1000 G0 X0 Y0 Z0	
N1010 ...	
...	
N2050 SUBPROG1	; 알람 출력 = "문자 R10= 에 대한 12080 채널 K1 블록 N9040 구문 오류"
N2060 ...	
N2350 SUBPROG2	; 알람 출력 = "문자 R10= 에 대한 12080 채널 K1 블록 N2350 구문 오류"
...	
N3000 M30	

예제 3: 현재 블록 디스플레이 억제 취소

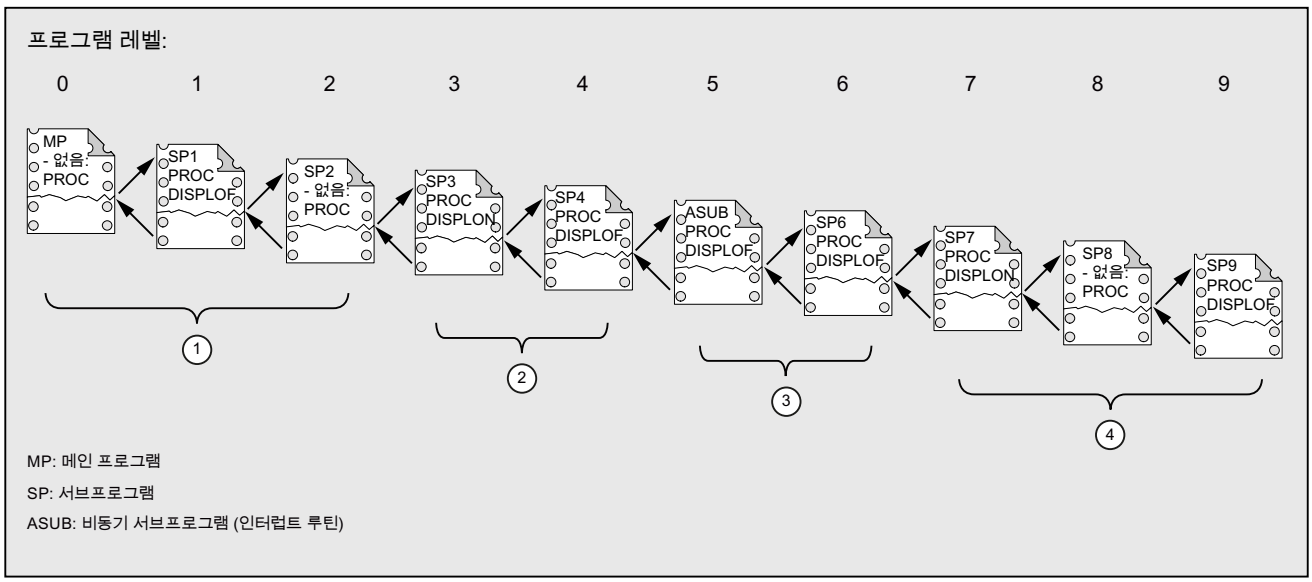
억제가 포함된 서브프로그램 SUB1:

프로그램 코드	설명
PROC SUB1 DISPLOF	; SUB1 서브프로그램에서 해당 블록 디스플레이를 억제합니다 . 하지만 SUB1 이 호출되면 현재 블록이 표시됩니다 .
...	
N300 SUB2	; 서브프로그램 SUB2 를 호출합니다 .
...	
N500 M17	

억제가 없는 서브프로그램 SUB2:

프로그램 코드	설명
PROC SUB2 DISPLON	; 서브프로그램 SUB2 에서 현재 블록 디스플레이 억제를 취소합니다 .
...	
N200 M17	; 서브프로그램 SUB1 로 복귀합니다 . SUB1 에 현재 블록 디스플레이 억제가 복구됩니다 .

예제 4: 서로 다른 DISPLON/DISPLOF 조합에 대한 디스플레이 응답



① 프로그램 레벨 0 의 가공 프로그램 라인은 현재 블록 디스플레이에 표시됩니다 .

② 프로그램 레벨 3 의 가공 프로그램 라인은 현재 블록 디스플레이에 표시됩니다 .

③ 프로그램 레벨 3 의 가공 프로그램 라인은 현재 블록 디스플레이에 표시됩니다 .

④ 프로그램 레벨 7/8 의 가공 프로그램 라인은 현재 블록 디스플레이에 표시됩니다 .

1.25.2.7 준비 과정에서 서브프로그램 식별 (PREPRO)

기능

모든 파일은 전원을 켜는 동안 PROC 연산 끝에 있는 PREPRO 키워드를 통해 식별할 수 있습니다 .

주

이 프로그램 준비 유형은 설정된 관련 머신 데이터에 따라 다릅니다 . 장비 제조업체의 설명서를 참조하십시오 .

참고 자료 :
기능 매뉴얼 , 특수 기능 , 전처리 (V2)

구문

PROC … PREPRO

의미

PREPRO: 전원을 켜는 동안 준비된 사이클 디렉토리에 저장된 NC 프로그램의 모든 파일을 식별하기 위한 키워드

서브프로그램 호출과 준비를 통한 서브프로그램 읽기

사이클 디렉토리는 전원을 켜는 동안 및 서브프로그램을 호출하는 동안 파라미터로 전처리한 서브프로그램에 대해 동일한 순서로 처리됩니다 .

1. _N_CUS_DIR 사용자 사이클
2. _N_CMA_DIR 제조업체 사이클
3. _N_CST_DIR 표준 사이클

동일한 이름을 가진 NC 프로그램이 서로 다른 특성을 가진 경우, 검색된 첫 번째 PROC 연산이 활성화되고 다른 PROC 연산은 알람 메시지 없이 무시됩니다 .

1.25.2.8 서브프로그램 복귀 M17

기능

복귀 명령 M17 (또는 가공 프로그램 종료 명령 M30) 은 서브프로그램의 끝에 위치합니다 . 이 명령은 서브프로그램 호출 시 가공 프로그램 블록에 있는 호출 프로그램으로 복귀하도록 처리합니다 .

주

M17 및 M30 은 NC 언어에서 동일하게 취급됩니다 .

구문

```
PROC < 프로그램 이름 >
...
M17/M30
```

일반 조건

서브프로그램 복귀가 연속 이송 모드에 미치는 영향

M17 (또는 M30) 이 가공 프로그램 블록에 독립적으로 존재하는 경우 채널 내에서 사용 중인 연속 이송 모드가 인터럽트됩니다 .

연속 이송 모드가 인터럽트되는 것을 방지하려면 M17 (또는 M30) 을 마지막 이송 블록에 포함시켜야 합니다 . 또한 다음 머신 데이터를 "0" 으로 설정해야 합니다 .

MD20800 $MC_SPF_END_TO_VDI = 0 (NC/PLC 인터페이스에 M30/M17 출력 안 함)

예제

1. 개별 블록에 M17 이 포함된 서브프로그램

프로그램 코드	설명
N10 G64 F2000 G91 X10 Y10	
N20 X10 Z10	
N30 M17	; 연속 이송 모드를 인터럽트하고 리턴 점프합니다 .

2. 마지막 이송 블록에 M17 이 포함된 서브프로그램

프로그램 코드	설명
N10 G64 F2000 G91 X10 Y10	
N20 X10 Z10 M17	; 연속 이송 모드를 인터럽트하지 않고 리턴 점프합니다 .

1.25.2.9 RET 서브프로그램 복귀

기능

서브프로그램에서 RET 명령을 M17 리턴 점프 명령 대신 사용할 수도 있습니다 . RET 는 개별 가공 프로그램 블록에 프로그래밍되어야 합니다 . M17 과 마찬가지로 RET 는 서브프로그램을 호출한 다음 가공 프로그램 블록에서 호출한 프로그램으로 복귀합니다 .

주

파라미터를 사용하여 RET 의 리턴 점프 방식을 변경할 수 있습니다 (" 파라미터 지정 가능한 서브프로그램 리턴 점프 (RET ...) [쪽 180]" 참조).

적용

리턴 점프가 발생 되더라도 G64 연속 경로 모드 (G641 ~ G645) 를 중단하지 않으려면 RET 명령을 사용해야 합니다 .

사전 조건

RET 명령어는 SAVE 속성으로 정의되지 않은 서브프로그램에만 사용할 수 있습니다 .

구문

```
PROC < 프로그램 이름 >
...
RET
```

예제

메인 프로그램 :

프로그램 코드	설명
PROC MAIN_PROGRAM	; 프로그램 시작
...	
N50 SUB_PROG	; 서브프로그램 호출 : SUB_PROG
N60 ...	
...	
N100 M30	; 프로그램 종료

서브프로그램 :

프로그램 코드	설명
PROC SUB_PROG	
...	
N100 RET	; 메인 프로그램의 블록 N60 으로 리턴 점프를 실행합니다 .

1.25.2.10 파라미터 지정 가능한 서브프로그램 리턴 점프 (RET ...)

기능

일반적으로 RET 또는 M17 서브프로그램 종료는 서브프로그램이 호출된 프로그램으로 복귀하여 서브프로그램 호출 이후 프로그램 라인에서 계속 프로세스를 진행합니다 .

그러나 반면에 , 다른 위치에서 프로그램을 다시 시작해야 하는 경우도 있습니다 . 예를 들면 ,

- ISO 언어 모드에서 황삭 사이클을 호출한 후 (형상 설명 이후) 프로그램 실행을 재개하는 경우 .
- 오류 처리를 위해 서브프로그램 레벨에서 메인 프로그램으로 복귀하는 경우 (ASUB 이후 포함).
- 컴파일 사이클 및 ISO 언어 모드에서 특정 적용 분야에 대해 여러 프로그램 레벨을 통한 리턴 점프 .

이러한 경우 RET 명령은 " 리턴 점프 파라미터 " 와 함께 프로그래밍됩니다 .

구문

RET("<destination block>")
RET("<destination block>",<block after destination block>)
RET("<destination block>",<block after destination block> <number of return jump levels>)
RET("<destination block>", ,<number of return jump levels>)
RET("<destination block>",<block after destination block>,<number of return jump levels>, <return jump to the beginning of the program>)
RET(, ,<number of return jump levels>,<return jump to the beginning of the program>)

의미

RET: 서브프로그램 종료 (M17 대신 사용)

< 대상 블록 >: 리턴 점프 파라미터 1

점프 대상으로 프로그램 실행을 재개해야 하는 블록을 선언합니다 .

리턴 점프 파라미터 3 을 프로그래밍하지 않을 경우 점프 대상은 현재 서브프로그램을 호출한 프로그램에 위치합니다 .

가능한 데이터는 다음과 같습니다 .

"<block number>"	대상 블록 번호
"<jump marker>"	대상 블록에 설정해야 하는 점프 마커
"<character string>"	프로그램에서 인식해야 하는 문자열 (예 : 프로그램 또는 변수 이름). 다음 규칙은 문자열을 프로그래밍할 때 적용됩니다 . • **종료 시 공란** (점프 마커와 반대이며 끝에 ":" 을 사용하여 지정). • **문자열 앞**에만 하나의 블록 번호 및 / 또는 점프 마커를 설정할 수 있으며 , **프로그램 명령이 없습니다** .

< 대상 블록 다음 블록 >: 리턴 점프 파라미터 2

리턴 점프 파라미터 1 을 참조하십시오 .

유형 : INT

값 :	0	리턴 점프 파라미터 1 을 사용하여 지정된 블록으로 리턴 점프됩니다 .
	> 0	리턴 점프 파라미터로 지정된 블록 다음에 위치한 블록으로 리턴 점프됩니다 .

< 리턴 점프 레벨의 번호 >: 리턴 점프 파라미터 3

프로그램 실행을 계속해야 하는 프로그램 레벨에 도달하기 위해 뒤로 점프해야 하는 레벨 수를 지정합니다 .

유형 : INT

값 :	1	프로그램이 "현재 프로그램 레벨 - 1"에서 재개됩니다 (파라미터 없는 RET 와 비슷함).
	2	프로그램이 "현재 프로그램 레벨 - 2"에서 재개됩니다 . 예를 들면 한 레벨을 건너뜁니다 .
	3	프로그램이 "현재 프로그램 레벨 - 3"에서 재개됩니다 . 예를 들면 두 레벨을 건너뜁니다 .
	...	

값 범위 :

1 ... 15

< 프로그램 시작으로 리턴 점프 >:	리턴 점프 파라미터 4		
	유형 :	BOOL	
	값 :	1	메인 프로그램으로 리턴 점프되고 거기에서 **ISO 언어 모드**가 활성화되면 프로그램은 프로그램의 시작 위치로 분기됩니다 .

주

대상 블록 검색을 지정하기 위해 문자열로 서브프로그램을 리턴 점프하는 경우 처음에는 항상 호출 프로그램의 점프 마커를 검색합니다 .

문자열을 사용하여 점프 대상을 고유하게 정의해야 하는 경우 문자열이 점프 마커의 이름과 일치하지 않아야 합니다 . 일치할 경우 서브프로그램이 항상 문자열이 아닌 점프 마커로 리턴 점프됩니다 (예제 2 참조).

일반 조건

여러 프로그램 레벨을 통해 리턴 점프하는 경우 개별 프로그램 레벨에 대해 SAVE 명령이 평가됩니다 .

여러 프로그램 레벨을 통한 리턴 점프에 대해 모달 서브프로그램이 활성 상태이고 건너뛴 프로그램 중 하나에서 선택 취소 명령 MCALL 이 모달 서브프로그램에 대해 프로그래밍되면 모달 서브프로그램은 활성 상태를 유지합니다 .

⚠ 주의

프로그래머는 여러 프로그램 레벨을 통한 리턴 점프에 대해 프로그램이 계속해서 올바른 모달 설정을 수행하는지 주의 깊게 확인해야 합니다 . 이 작업은 적절한 메인 블록을 프로그래밍하여 수행할 수 있습니다 .

예제

예제 1: ASUB 실행 이후 메인 프로그램에서 다시 시작

프로그래밍	설명
N10010 CALL "UP1"	; 프로그램 레벨 0 (메인 프로그램)
N11000 PROC UP1	; 프로그램 레벨 1
N11010 CALL "UP2"	
N12000 PROC UP2	; 프로그램 레벨 2
...	
N19000 PROC ASUB	; 프로그램 레벨 3 (ASUB 실행)
...	
N19100 RET("N10900", ,$P_STACK)	; 서브프로그램 복귀 .
N10900	; 메인 프로그램에서 다시 시작 .
N10910 MCALL	; 모달 서브프로그램 비활성화 .
N10920 G0 G60 G40 M5	; 추가 모달 설정 변경 .

예제 2: 대상 블록 검색을 지정하기 위한 문자열 (string>)

메인 프로그램 :

프로그램 코드	설명
PROC MAIN_PROGRAM	
N1000 DEF INT iVar1=1, iVar2=4	
N1010 ...	
N1200 subProg1	; 서브프로그램 "subProg1" 호출
N1210 M2 S1000 X10 F1000	
N1220	
N1400 subProg2	; 서브프로그램 "subProg2" 호출
N1410 M3 S500 Y20	
N1420 ..	
N1500 lab1: iVar1=R10*44	
N1510 F500 X5	
N1520 ...	
N1550 subprog1: G1 X30	; 여기서 "subProg1" 은 점프 마커로 정의됩니다 .
N1560 ...	
N1600 subProg3	서브프로그램 "subProg3" 호출
N1610 ...	
N1900 M30	

서브프로그램 subProg1:

프로그램 코드	설명
PROC subProg1	
N2000 R10=R20+100	
N2010 ...	
N2200 RET("subProg2")	; 블록 N1400 에서 메인 프로그램으로 리턴 점프

서브프로그램 subProg2:

프로그램 코드	설명
PROC subProg2	
N2000 R10=R20+100	
N2010 ...	
N2200 RET("iVar1")	; 블록 N1500 에서 메인 프로그램으로 리턴 점프

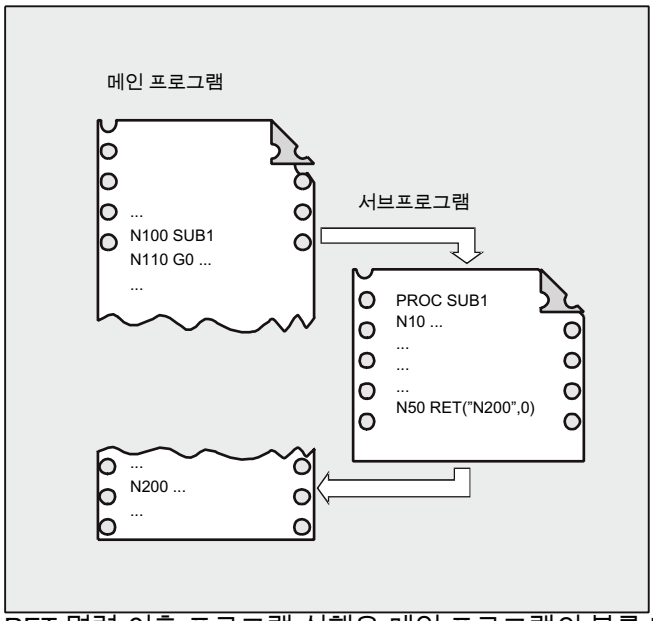
서브프로그램 subProg3:

프로그램 코드	설명
PROC subProg3	
N2000 R10=R20+100	
N2010 ...	
N2200 RET("subProg1")	; 블록 N1550 에서 메인 프로그램으로 리턴 점프

추가 정보

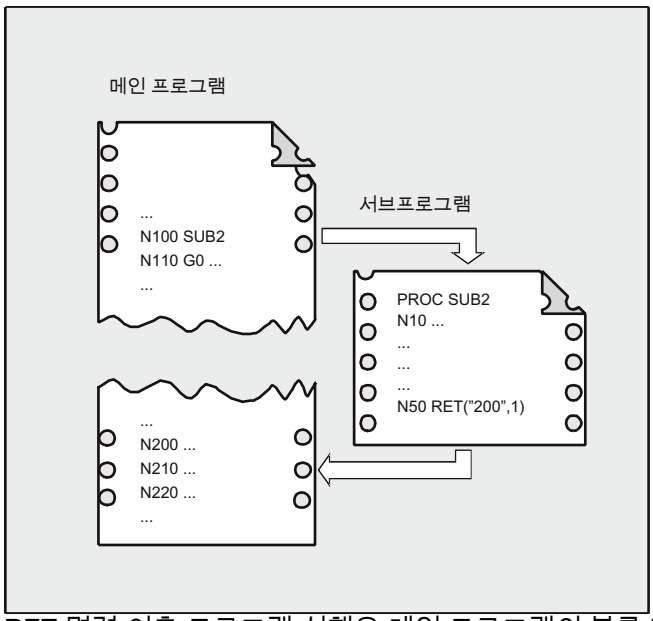
리턴 점프 파라미터 1-3 의 영향에 대해서는 다음 그림에서 설명합니다 .

첫 번째 리턴 점프 파라미터 1 - "N200", 리턴 점프 파라미터 2 = 0



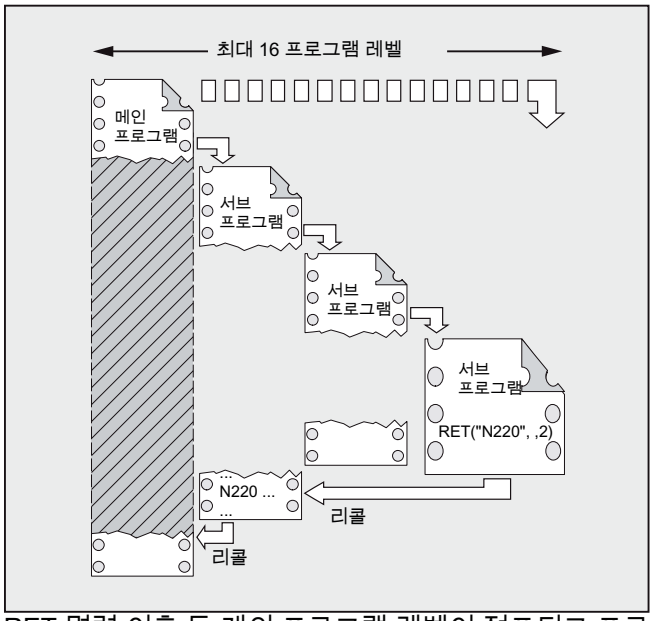
RET 명령 이후 프로그램 실행은 메인 프로그램의 블록 N200 에서 계속됩니다 .

두 번째 리턴 점프 파라미터 1 - "N200", 리턴 점프 파라미터 2 = 1



RET 명령 이후 프로그램 실행은 메인 프로그램의 블록 N200 다음에 오는 블록 (N210) 에서 계속됩니다 .

세 번째 리턴 점프 파라미터 1 - "N220", 리턴 점프 파라미터 3 = 2



RET 명령 이후 두 개의 프로그램 레벨이 점프되고 프로그램 실행은 블록 N220 에서 계속됩니다 .

1.25.3 서브프로그램 호출

1.25.3.1 파라미터 전송 없는 서브프로그램 호출

기능

서브프로그램은 어드레스 L 및 서브프로그램 번호를 사용해 호출하거나 프로그램 이름을 지정해 호출합니다 .

메인 프로그램을 서브프로그램으로 호출할 수도 있습니다 . 이 경우 메인 프로그램에 설정된 프로그램 종료 M2 또는 M30 이 M17 (프로그램 종료 시 호출 프로그램으로 복귀) 로 평가됩니다 .

주

따라서 서브프로그램을 메인 프로그램처럼 시작할 수도 있습니다 .

시스템의 탐색 방식 :

*_MPF 포함 ?

*_SPF 포함 ?

결과 : 호출할 서브프로그램의 이름이 메인 프로그램의 이름과 동일한 경우 호출을 실행한 메인 프로그램이 다시 호출됩니다 . 이는 보통의 경우 바람직하지 않은 현상이므로 서브프로그램 및 메인 프로그램에 고유한 이름을 지정하여 이런 현상을 방지해야 합니다 .

주

파라미터 전송을 요구하지 않는 서브프로그램은 초기화 파일에서 호출할 수도 있습니다 .

구문

L< 번호 >/< 프로그램 이름 >

주

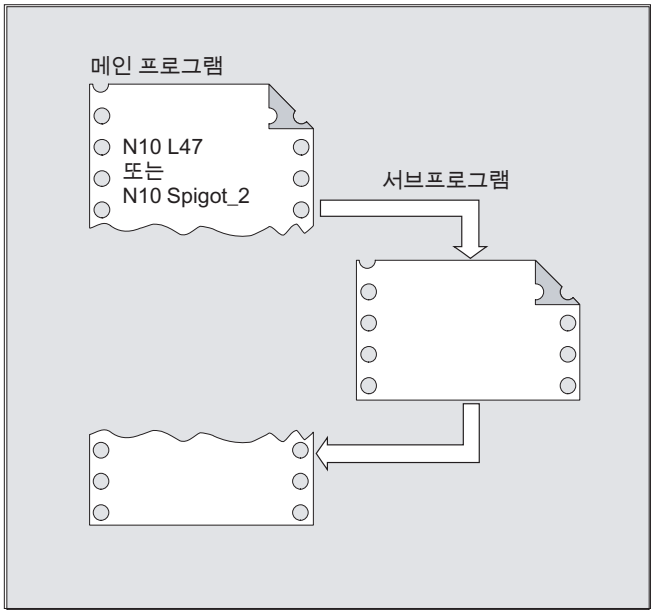
서브프로그램 호출은 항상 개별 NC 블록에 프로그래밍해야 합니다 .

의미

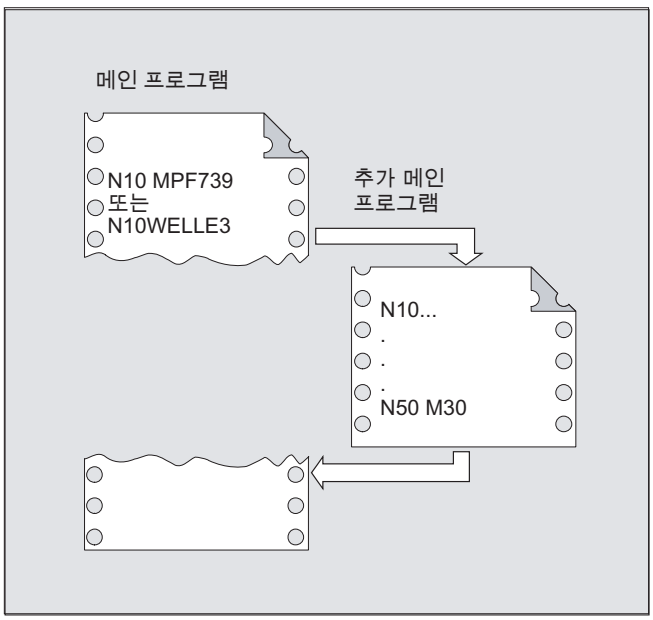
L:	서브프로그램 호출 어드레스	
< 번호 >:	서브프로그램 이름	
	유형 :	INT
	값 :	소수점 최대 7 개 **주의 :** 선행 0 은 이름에서 중요합니다 (⇒ L123, L0123 및 L00123 은 세 개의 서로 다른 서브프로그램).
< 프로그램 이름 >:	서브프로그램 이름 (또는 메인 프로그램 이름)	

예제

예제 1: 파라미터 전송 없이 서브프로그램 호출



예제 2: 메인 프로그램을 서브프로그램으로 호출



1.25.3.2 파라미터 전송을 포함한 서브프로그램 호출 (EXTERN)

기능

파라미터 전송을 포함한 서브프로그램이 호출되는 경우 변수 또는 값을 직접 전송할 수 있습니다 (VAR 파라미터 제외).

파라미터 전송을 통한 서브프로그램은 메인 프로그램에서 호출하기 전에 (예를 들어 프로그램 시작) 메인 프로그램에서 EXTERNAL 을 통해 선언할 수 있습니다 . 서브프로그램의 이름 및 변수 유형은 전송되는 순서대로 지정됩니다 .

⚠ 주의
변수 유형 및 전송 순서는 모두 서브프로그램의 PROC 로 선언된 정의와 일치해야 합니다 . 파라미터 이름은 메인 프로그램과 서브프로그램에서 서로 다를 수 있습니다 .

구문

```
EXTERNAL <program name>(<type_Par1>,<type_Par2>,<type_Par3>)
...
<program name>(<value_Par1>,<value_Par2>,<value_Par3>)
```

⚠ 주의
서브프로그램 호출은 항상 개별 NC 블록에서 프로그램해야 합니다 .

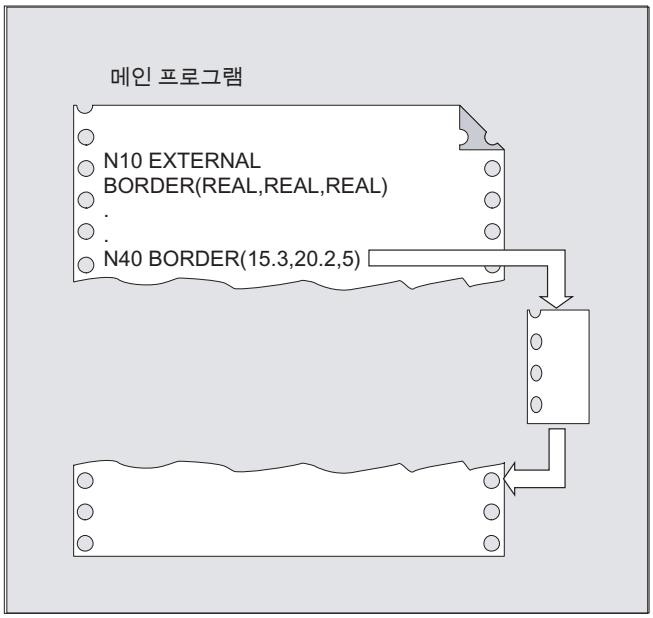
의미

< 프로그램 이름 >:	서브프로그램 이름
EXTERN:	파라미터 전송을 통해 서브프로그램을 선언할 키워드입니다 . **참고 :** 서브프로그램이 공작물 또는 전역 서브프로그램 디렉토리에 있는 경우에만 EXTERNAL 을 지정해야 합니다. 사이클은 EXTERN으로 선언할 필요가 없습니다 .
<유형_파라미터1>,<유형_파라미터2>,<유형_파라미터 3>:	전송 순서대로 전송할 파라미터의 변수 유형
<값_파라미터1>,<값_파라미터2>,<값_파라미터3> :	전송할 파라미터의 변수 값

예제

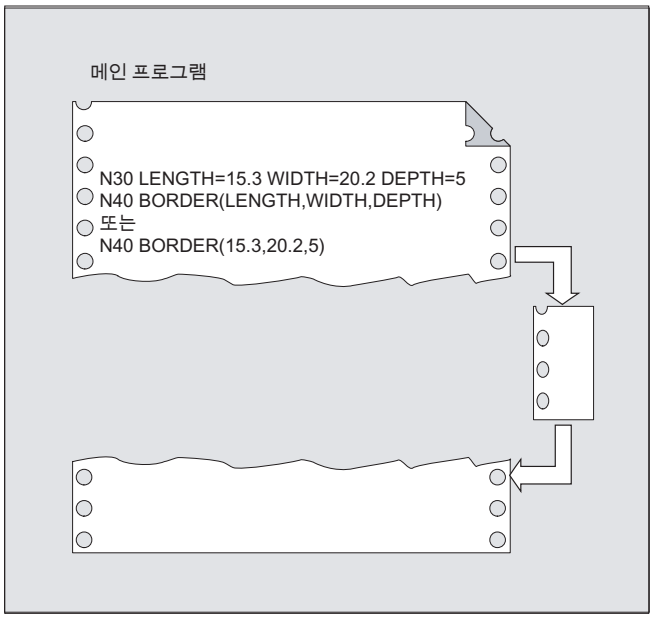
예제 1: 선언 후 서브프로그램 호출

프로그램 코드	설명
N10 EXTERNAL BORDERS(REAL,REAL,REAL)	; 서브프로그램을 지정합니다 .
...	
N40 BORDER(15.3,20.2,5)	; 파라미터 전송을 통해 서브프로그램을 호출합니다 .



예제 2: 선언 없이 서브프로그램 호출

프로그램 코드	설명
N10 DEF REAL LENGTH, WIDTH, DEPTH	
N20…	
N30 LENGTH=15.3 WIDTH=20.2 DEPTH=5	
N40 BORDER(LENGTH,WIDTH,DEPTH)	; or: N40 BORDER(15.3,20.2,5)



1.25.3.3 프로그램 반복 횟수 (P)

기능

서브프로그램을 연속적으로 여러 번 실행해야 하는 경우 서브프로그램 호출을 통해 블록의 어드레스 P 에서 원하는 프로그램 반복 횟수를 입력할 수 있습니다 .

> ⚠ 주의
>
> **프로그램 반복 및 파라미터 전송을 통한 서브프로그램 호출**
>
> 예를 들어 파라미터는 첫 번째 실행에서와 같이 프로그램을 호출하는 경우에만 전송됩니다 . 나머지 반복 동안 파라미터는 변경되지 않습니다 . 프로그램을 반복하는 동안 파라미터를 반복하려면 서브프로그램에 적절한 조건을 만들어야 합니다 .

구문

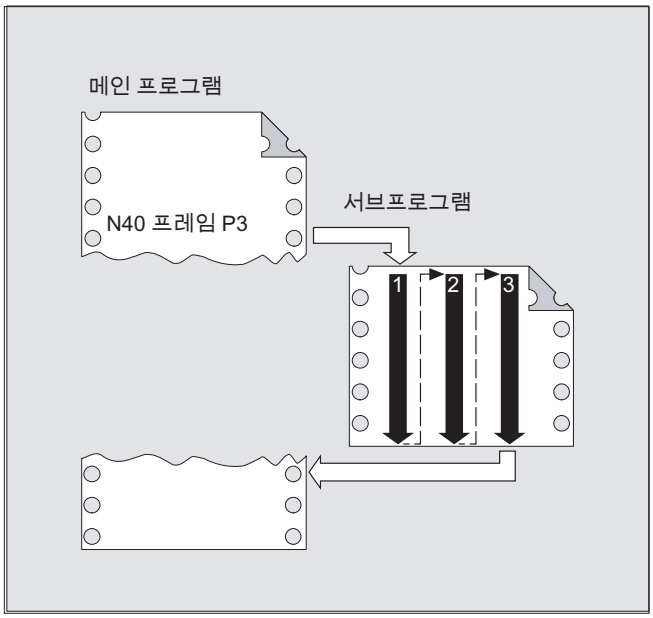
< 프로그램 이름 > P< 값 >

의미

< 프로그램 이름 >:	서브루틴 호출	
P:	프로그램 반복을 프로그래밍할 어드레스	
< 값 >:	프로그램 반복 횟수	
	유형 :	INT
	값 범위 :	1 … 9999 (서명 안 함)

예제

프로그램 코드	설명
...	
N40 FRAME P3	; BORDER 서브프로그램은 차례대로 3 회 실행해야 합니다 .
...	



1.25.3.4 모달 서브프로그램 호출 (MCALL)

기능

MCALL 을 통한 모달 서브프로그램 호출의 경우 , 각 블록의 경로 이동 후 서브프로그램이 자동으로 호출되어 실행됩니다. 이를 통해 서브프로그램 호출이 자동화되고 따라서 서로 다른 공작물 위치에서 실행됩니다 (예를 들어 드릴링 패턴 생성을 위해).

MCALL은 서브프로그램 호출 없이 또는 새 서브프로그램에 대해 새 모달 서브프로그램 호출을 프로그래밍하여 기능을 비활성화하는 데 사용됩니다 .

⚠ 주의

프로그램 실행 시 한 번에 하나의 MCALL 호출만 유효합니다 . 파라미터는 MCALL 호출에 대해 한 번만 전송됩니다 .

다음 경우에도 서브프로그램 모달은 이동 프로그램 없이 호출됩니다 :

- G0 또는 G1 이 활성 상태인 경우 프로그램 어드레스 S 및 F.
- G0/G1 이 블록 또는 추가 G 코드를 통해 단독으로 프로그래밍된 경우 .

구문

MCALL < 프로그램 이름 >

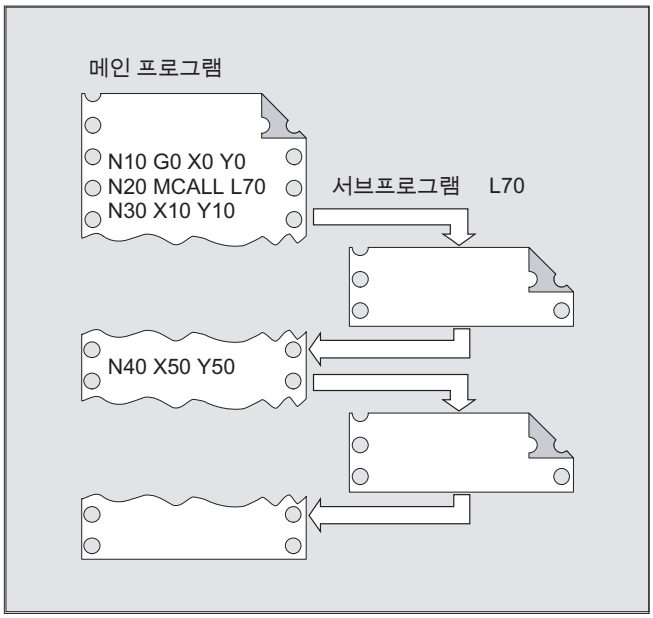
의미

MCALL:	서브프로그램 모달 호출을 위한 명령어 .
< 프로그램 이름 >:	서브프로그램 이름

예제

예제 1:

프로그램 코드	설명
N10 G0 X0 Y0	
N20 MCALL L70	; 모달 서브루틴 호출 .
N30 X10 Y10	; 프로그래밍된 위치에 도달한 후 서브프로그램 L70 을 실행합니다 .
N40 X50 Y50	; 프로그래밍된 위치에 도달한 후 서브프로그램 L70 을 실행합니다 .



예제 2:

프로그램 코드
N10 G0 X0 Y0
N20 MCALL L70
N30 L80

이 예제에서 프로그래밍된 경로 축을 가진 다음 NC 블록이 서브프로그램 L80 에 위치합니다 . L70 은 L80 에 의해 호출됩니다 .

1.25.3.5 간접 서브프로그램 호출 (CALL)

기능

프로그램의 특정 지점에서 사용 중인 조건에 따라 서로 다른 서브프로그램을 호출할 수 있습니다 . 서브프로그램의 이름은 유형 STRING 의 변수에 저장됩니다 . 서브프로그램 호출은 CALL 및 변수 이름을 통해 실행됩니다 .

⚠ 주의
간접 서브프로그램 호출은 파라미터 전송 없는 서브프로그램의 경우에만 가능합니다 . 직접 서브프로그램 호출의 경우 STRING 상수에 이름을 저장합니다 .

구문

CALL <program name>

의미

CALL:	간접 서브프로그램 호출을 위한 명령어 .
< 프로그램 이름 >:	서브프로그램 이름 (변수 또는 상수)
	유형 : STRING

예제

STRING 상수로 직접 호출 :

프로그램 코드	설명
…	
CALL "/_N_WKS_DIR/_N_SUBPROG_WPD/_N_PART1_SPF"	; CALL을 통해 서브프로그램 PART1 직접 호출 .
…	

변수를 통한 간접 호출 :

프로그램 코드	설명
…	
DEF STRING[100] PROGNAME	; 변수를 정의합니다 .
PROGNAME="/_N_WKS_DIR/_N_SUBPROG_WPD/_N_PART1_SPF"	; 서브프로그램 PART1 을 PROGNAME 변수에 지정합니다 .
CALL PROGNAME	; CALL 및 PROGNAME 변수를 통해 서브프로그램 PART1 을 간접 호출합니다 .
…	

1.25.3.6 호출 프로그램 일부를 지정하여 서브프로그램 간접 호출 (CALL BLOCK ... TO ...)

기능

CALL 및 키워드 조합 BLOCK ... TO 를 사용하여 서브프로그램을 간접 호출하고 시작 레이블 및 종료 레이블로 지정한 프로그램의 일부를 실행합니다 .

구문

CALL <program name> BLOCK <start label> TO <end label>
CALL BLOCK <start label> TO <end label>

의미

CALL:	간접 서브프로그램 호출을 위한 명령어 .
< 프로그램 이름 >:	실행할 프로그램 일부가 포함된 서브프로그램 이름 (변수 또는 상수) (**지정**은 **옵션**임). 유형 : STRING **참고 :** < 프로그램 이름 > 이 프로그래밍되지 않은 경우 해당 프로그램에서 < 시작 레이블 > 및 < 종료 레이블 > 로 지정한 프로그램의 일부를 검색해 실행합니다 .
BLOCK ... TO ... :	프로그램 일부 간접 실행을 위한 키워드 조합
< 시작 레이블 >:	실행할 프로그램 일부의 시작을 의미하는 변수 . 유형 : STRING
< 종료 레이블 >:	실행할 프로그램 일부의 종료를 의미하는 변수 . 유형 : STRING

예제

메인 프로그램 :

프로그램 코드	설명
...	
DEF STRING[20] STARTLABEL, ENDLABEL	; 시작 레이블 및 종료 레이블의 변수 정의
STARTLABEL="LABEL_1"	
ENDLABEL="LABEL_2"	
...	
CALL "CONTUR_1" BLOCK STARTLABEL TO ENDLABEL	; 간접 서브프로그램 호출 및 호출 프로그램 일부를 표시하는 식별자 .
...	

서브프로그램 :

프로그램 코드	설명
PROC CONTUR_1 ...	
LABEL_1	; 시작 레이블 : 프로그램 일부의 실행을 시작합니다 .
N1000 G1 ...	
...	
LABEL_2	; 종료 레이블 : 프로그램 일부의 실행을 종료합니다 .
...	

1.25.3.7 ISO 언어로 프로그래밍된 프로그램의 간접 호출 (ISOCALL)

기능

ISO 언어로 프로그래밍된 프로그램은 간접 프로그램 호출 ISOCALL 을 사용하여 호출할 수 있습니다. 머신 데이터에서 설정된 ISO 모드가 활성화됩니다. 원래 실행 모드는 프로그램 종료 시 다시 적용됩니다 . ISO 모드를 머신 데이터에서 설정하지 않으면 서브프로그램이 Siemens 모드에서 호출됩니다 .

ISO 모드에 대한 자세한 내용은
참고 자료 :
ISO 언어 기능 설명서를 참조하십시오 .

구문

ISOCALL <program_name>

의미

ISOCALL:	머신 데이터에서 설정된 ISO 모드가 활성화되는 간접 서브프로그램 호출 키워드 .
< 프로그램 이름 >:	ISO 언어로 프로그래밍된 프로그램 이름 (변수 또는 상수 , 유형 STRING)

예제 : ISO 모드에서 프로그래밍된 사이클을 통한 형상 호출

프로그램 코드	설명
0122_SPF	; ISO 모드에서 형상 설명
N1010 G1 X10 Z20	
N1020 X30 R5	
N1030 Z50 C10	
N1040 X50	
N1050 M99	
N0010 DEF STRING[5] PROGNAME = “0122“	; Siemens 가공 프로그램 (사이클)
...	
N2000 R11 = $AA_IW[X]	
N2010 ISOCALL PROGNAME	
N2020 R10 = R10+1	; ISO 모드에서 프로그램 0122.spf 실행
...	
N2400 M30	

1.25.3.8 경로 지정 및 파라미터를 통한 서브루틴 호출 (PCALL)

기능

PCALL 을 사용하여 절대 경로 및 파라미터 전송을 통해 서브프로그램을 호출할 수 있습니다 .

구문

PCALL <path/program name>(<parameter 1>,…,<parameter n>)

의미

PCALL:	절대 경로 이름을 통한 서브프로그램 호출을 위한 키워드
< 경로 / 프로그램 이름 >:	서브프로그램 이름을 포함하여 "/" 로 시작하는 절대 경로 이름 절대 경로 이름을 지정하지 않을 경우 PCALL 은 프로그램 식별자를 통한 표준 서브프로그램 호출과 유사하게 동작합니다 . 프로그램 식별자는 선행 _N_ 및 확장자 없이 기록됩니다 . 선행 문자 및 확장자를 사용하여 프로그램 이름을 프로그래밍하는 경우 EXTERN 명령을 사용하여 선행 문자 및 확장자를 명시적으로 선언해야 합니다 .
< 파라미터 1>, ...:	서브프로그램의 PROC 연산 결과를 따르는 실제 파라미터.

예제

프로그램 코드
PCALL/_N_WKS_DIR/_N_WELLE_WPD/WELLE(parameter1,parameter2,…)

1.25.3.9 서브프로그램 호출에 대한 확장 검색 경로 (CALLPATH)

기능

서브프로그램 호출에 대한 검색 경로는 CALLPATH 명령을 사용하여 확장할 수 있습니다 .

따라서 서브프로그램의 전체 절대 경로 이름을 지정할 필요 없이 선택하지 않은 공작물 디렉토리에서도 서브프로그램을 호출할 수 있습니다 .

검색 경로 확장은 사용자 사이클 항목 앞에 적용됩니다 (_N_CUS_DIR).

검색 경로 확장은 다음 이벤트가 발생할 경우 다시 선택 취소됩니다 .

- 공란을 포함한 CALLPATH
- 파라미터가 없는 CALLPATH
- 가공 프로그램 끝
- 리셋

구문

CALLPATH("<path name>")

의미

CALLPATH:	프로그램 가능한 검색 경로 확장 키워드 . 개별 가공 프로그램 행에 프로그래밍됩니다 .
< 경로 이름 >:	상수 또는 변수 , 유형 STRING. 검색 경로를 확장해야 하는 디렉토리의 절대 경로 이름이 포함되어 있습니다 . 경로 이름은 "/" 로 시작합니다 . 접두사 및 접미사를 사용하여 경로를 완벽하게 지정해야 합니다 . 최대 경로 길이는 128 바이트입니다 . <path name> 에 공란이 있거나 파라미터 없이 CALLPATH 를 호출하면 검색 경로 명령이 다시 리셋됩니다 .

예제

프로그램 코드
CALLPATH ("/_N_WKS_DIR/_N_MYWPD_WPD")

다음 검색 경로가 설정됩니다 (위치 5. 는 새로운 경로):

1. 현재 디렉토리 / 서브프로그램 이름
2. 현재 디렉토리 / 서브프로그램 식별자 _SPF
3. 현재 디렉토리 / 서브프로그램 식별자 _MPF
4. /_N_SPF_DIR/ 서브프로그램 식별자 _SPF
5. **/_N_WKS_DIR/_N_MYWPD/ 서브프로그램 식별자 _SPF**
6. /N_CUS_DIR/_N_MYWPD/ 서브프로그램 식별자 _SPF
7. /_N_CMA_DIR/ 서브프로그램 식별자 _SPF
8. /_N_CST_DIR/ 서브프로그램 식별자 _SPF

보충 조건

- CALLPATH는 프로그래밍된 경로 이름이 실제 존재하는지 여부를 확인합니다. 잘못된 일부 프로그램을 실행할 경우 수정 블록 알람 14009 와 함께 중단됩니다 .
- 또한 CALLPATH 에서 INI 파일을 프로그래밍할 수 있습니다 . INI 파일을 처리하는 데 걸리는 시간 동안에만 유효합니다 (WPD-INI 파일 또는 NC 활성 데이터의 초기화 프로그램 , 예를 들어 첫 번째 채널 _N_CH1_UFR_INI 의 프레임). 검색 경로가 다시 리셋됩니다 .

1.25.3.10 외부 서브루틴 실행 (EXTCALL)

기능

EXTCALL 명령어를 사용하면 외부 프로그램 메모리 (로컬 드라이브, 네트워크 드라이브, USB 드라이브) 에서 가공 프로그램을 다운로드하여 서브프로그램으로 실행할 수 있습니다 .

외부 프로그램 디렉토리의 경로는 셋팅 데이터를 사용하여 사전 설정할 수 있습니다 .

SD42700 $SC_EXT_PROG_PATH

이것은 EXTCALL 호출로 지정된 프로그램 경로 또는 식별자와 함께 호출할 프로그램의 전체 경로를 지정합니다 .

주

점프 대상

점프 명령 (GOTOF, GOTOB, CASE, FOR, LOOP, WHILE, REPEAT, IF, ELSE, ENDIF 등) 을 포함한 외부 프로그램의 경우에는 점프 대상이 포스트 로딩 메모리 안에 있어야 합니다 .
포스트 로딩 메모리 크기는 다음을 통해 설정합니다 .

MD18360 MM_EXT_PROG_BUFFER_SIZE

파라미터

외부 프로그램을 호출할 때는 다음 파라미터 중 어떤 것도 전송할 수 없습니다 .

구문

EXTCALL("< 경로 />< 프로그램 이름 >")

중요도

EXTCALL:	외부 서브프로그램 호출을 위한 명령어 .	
"< 경로 />< 프로그램 이름 >":	STRING 유형의 상수 / 변수	
	< 경로 >:	절대 또는 상대 경로 데이터 (**옵션**)
	< 프로그램 이름 >:	프로그램 이름은 접두어 "_N_" 없이 지정합니다 . "_" 또는 "." 문자를 사용하여 프로그램 이름에 파일 확장자 ("MPF", "SPF") 를 추가할 수 있습니다 (**옵션**). 예제 : "SHAFT" 또는 "WELLE_SPF" 또는 "WELLE.SPF"

주

경로 지정 : 단축 명칭

경로를 지정할 때 다음과 같은 단축 명칭을 사용할 수 있습니다 .

- **LOCAL_DRIVE:** 로컬 드라이브
- **CF_CARD:** 콤팩트 플래시 카드
- **USB:** OP 전면의 USB

CF_CARD: 및 **LOCAL_DRIVE:** 교대로 사용할 수 있습니다 .

주

USB 드라이브를 통해 외부 소스에서 실행

외부 가공 프로그램을 USB 인터페이스를 통해 외부 USB 드라이브에서 전송해야 하는 경우 X203 (이름 "TCU_1") 을 통한 인터페이스만 사용할 수 있습니다 .

유의사항
USB 드라이브를 통해 외부 소스에서 실행 (USB 전면 연결) 전면 USB 포트에 삽입한 USB 드라이브에서 직접 실행하는 것은 좋지 않습니다 . 그것은 가공 프로그램을 실행하는 동안 USB 드라이브 연결이 끊기면 (접촉 문제 , USB 드라이브 고장 , 중단 또는 예기치 않은 충격이나 전원 차단으로 인한) 즉시 가공이 중단되기 때문입니다. 이로 인해 공구 및 / 또는 공작물이 손상될 수 있습니다 .

예제

로컬 드라이브에서 실행

메인 프로그램 :

프로그램 코드

```
N010 PROC MAIN
N020 ...
N030 EXTCALL ("ROUGHING")
N040 ...
N050 M30
```

외부 서브프로그램 :

프로그램 코드

```
N010 PROC ROUGHING
N020 G1 F1000
N030 X= ... Y= ... Z= ...
N040 ...
...
...
N999999 M17
```

"MAIN.MPF" 메인 프로그램이 NC 메모리에 저장되고 실행을 위해 선택됩니다 .

이어서 "SCHRUPPEN.SPF" 또는 "SCHRUPPEN.MPF" 서브프로그램이 로컬 드라이브의 "/user/sinumerik/data/prog/WKS.DIR/WST1.WPD" 디렉토리에 로드됩니다 .

서브프로그램 경로는 SD42700 에 사전 설정됩니다 .

SD42700 $SC_EXT_PROG_PATH = "LOCAL_DRIVE:WKS.DIR/WST1.WPD"

주

SD42700 에 경로가 지정되어 있지 않으면 본 예제의 EXTCALL 명령을 다음과 같이 프로그래밍해야 합니다 .

EXTCALL("LOCAL_DRIVE:WKS.DIR/WST1.WPD/SCHRUPPEN")

추가 정보

절대 경로 이름을 사용한 EXTCALL 호출

지정된 경로에 서브프로그램이 존재할 경우 EXTCALL 호출 후에 실행됩니다 . 존재하지 않을 경우에는 프로그램 실행이 취소됩니다 .

상대 경로 이름 또는 경로 이름 없이 EXTCALL 호출

상대 경로 이름 또는 경로 이름 없이 EXTCALL 호출을 수행하는 경우, 가용 프로그램 메모리는 다음과 같이 검색됩니다 .

- 경로 이름이 SD42700 $SC_EXT_PROG_PATH 에 사전 설정된 경우 EXTCALL 호출에 지정된 데이터 (프로그램 이름 또는 상대 경로 이름) 가 이 경로에서 먼저 검색됩니다. 절대 경로는 다음 문자를 연결하여 생성됩니다 :
  - SD42700 에 사전 설정된 경로 이름
  - 구분자인 "/" 문자
  - EXTCALL 에 프로그램된 서브프로그램 경로 또는 식별자
- 호출된 서브프로그램이 사전 설정된 경로에 없을 경우 , EXTCALL 호출에 지정된 데이터가 사용자 - 메모리 디렉토리에서 검색됩니다 .
- 검색은 서브프로그램을 처음으로 찾았을 때 끝이 납니다 . 검색에서 아무 것도 찾을 수 없는 경우 프로그램이 취소됩니다 .

조정 가능한 리로드 메모리 (FIFO 버퍼)

" 외부 소스 실행 " 모드에서 프로그램 (메인 프로그램 또는 서브프로그램) 을 실행하려면 NCK에 리로드 메모리가 필요합니다. 리로드 메모리의 크기는 30 Kb로 사전 설정되어 있으며, 다른 모든 메모리 관련 머신 데이터와 마찬가지로 장비 제조업체만이 요구조건을 충족시키기 위해 변경할 수 있습니다 .

프로그램 (메인 프로그램 또는 서브프로그램) 이 " 외부 소스 실행 " 모드로 동시에 실행되기 위해서는 각 프로그램당 하나의 리로드 메모리가 설정되어야 합니다 .

RESET, POWER ON

RESET 과 POWER ON 은 외부 서브프로그램 호출을 중단하고 관련 로드 메모리를 삭제합니다 .

" 외부 소스에서 실행 " 을 위해 선택한 서브프로그램은 RESET/ 일부 프로그램 종료 이후 " 외부 소스에서 실행 " 을 위해 선택됩니다 . POWER ON 은 선택 사항을 삭제합니다 .

참고 자료

" 외부 소스에서 실행 " 에 대한 자세한 내용은 다음을 참조하십시오 .

기능 매뉴얼 , 기본 기능 , 모드 그룹 , 채널 , 프로그램 동작 , 리셋 절차 (K1)

1.25.4 싸이클

1.25.4.1 사용자 싸이클 파라미터화

기능

cov.com 및 uc.com 파일을 사용하여 자체 싸이클을 파라미터화할 수 있습니다 .

cov.com	싸이클 개요
uc.com	싸이클 호출 설명

cov.com 파일은 제품 제공 시 표준 싸이클에 포함되어 있으며 그에 따라 확장되어야 합니다 . uc.com 파일은 사용자가 만들어야 합니다 .

두 파일 모두 "User cycles" 디렉토리의 패시브 파일 시스템에 로드해야 하거나 프로그램에서 적절한 경로를 할당해야 합니다 .

;$PATH=/_N_CUS_DIR

cov.com 파일 (싸이클 개요) 수정

표준 싸이클에서 제공하는 cov.com 파일의 구조는 다음과 같습니다 .

%_N_COV_COM	파일 이름
;$PATH=/_N_CST_DIR	경로
;Vxxx 11.12.95 Sca cycle overview	코멘트 라인
C1(CYCLE81) drilling, centering	첫 번째 싸이클 호출
C2(CYCLE82) drilling, counterboring	두 번째 싸이클 호출
...	
C24(CYCLE98) chaining of threads	마지막 싸이클 호출
M17	확장자

새롭게 추가된 각 싸이클에 대해 다음 구문을 라인에 추가해야 합니다 .
C< 번호 > (< 싸이클 _ 이름 >) < 코멘트 >

여기서

< 번호 >:	파일에서 아직 사용하지 않은 정수
< 싸이클 이름 >:	통합할 싸이클의 프로그램 이름
< 코멘트 >:	싸이클의 설명 텍스트 (옵션)

예 :
C25 (MY_CYCLE_1) usercycle_1
C26 (SPECIAL CYCLE)

uc.com 파일의 사용자 싸이클 설명

각 싸이클의 헤더 라인 :

cov.com 파일 앞에 "//" 위치 :
//C < 번호 > (< 싸이클 이름 >) < 코멘트 >

예 :
//C25 (MY_CYCLE_1) user cycle_

각 파라미터의 설명 라인 :
(< 데이터 유형 식별자 > / < 최소값 > < 최대값 > / < 기본값 > /< 코멘트 >)

여기서

< 데이터 유형 식별자 >:	R: 실수 값 I: 정수 값 C: 문자 (1 자) S: 문자열
< 최소값 > < 최대값 >:	값 범위 정의 (생략 가능) 입력할 때 검사되는 입력 값의 한계 . 이 범위를 벗어나는 값은 입력할 수 없습니다 . 카운트 값을 지정하여 토클 키로 처리할 수 있습니다 . 이 값은 "*" 기호로 시작하여 카운트되며, 다른 값은 허용되지 않습니다. 예 : (I/*123456/1/Machining type) 문자열 및 문자 유형에 대한 제한이 없습니다 .
< 기본값 >:	싸이클을 호출할 때 해당 화면 양식에 기본값으로 설정되는 값 (생략 가능). 기본값은 조작자의 입력에 따라 달라질 수 있습니다 .
< 코멘트 >:	입력 필드 앞에 싸이클의 호출 화면 양식에 표시되는 설명 텍스트 (최대 50 자).

예

다음 두 가지 싸이클에서 싸이클 파라미터를 새롭게 작성해야 합니다 .

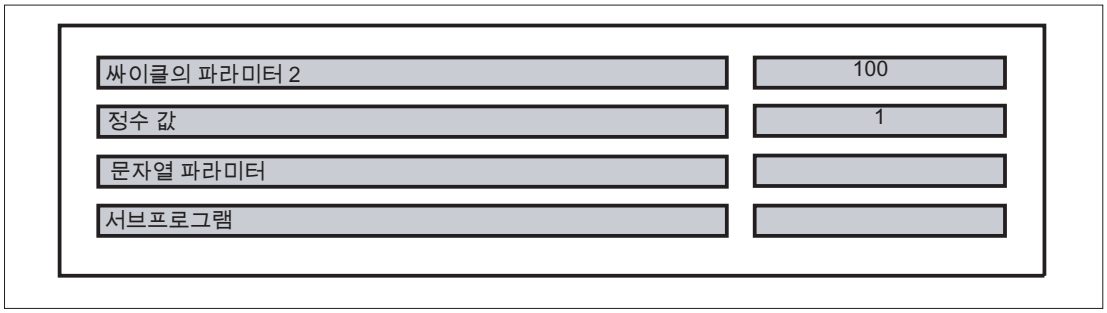
```
PROC MY_CYCLE_1 (REAL PAR1, INT PAR2, CHAR PAR3, STRING[10] PAR4)
싸이클에는 다음과 같은 전송 파라미터가 있습니다 .
PAR1:          ;  –1000.001 <= PAR2 <= 123.456 범위의 실수 값 , 기본값 100
PAR2:          ;  0 <= PAR3 <= 999999 사이 양의 정수 값 , 기본값 0
PAR3:          ;  ASCII 문자 1 개
PAR4:          ;  서브프로그램 이름의 문자열 길이 10
...
M17
```

```
PROC SPECIAL CYCLE (REAL VALUE1, INT VALUE2)
싸이클에는 다음과 같은 전송 파라미터가 있습니다 .
VALUE1:        ;  값 범위 제한 및 기본값 없는 실수 값
VALUE2:        ;  값 범위 제한 및 기본값 없는 정수 값
...
M17
```

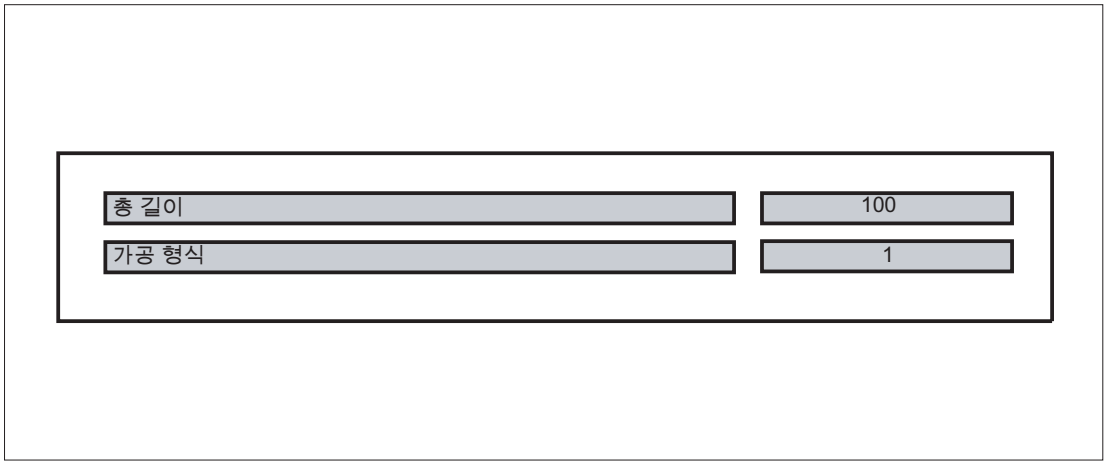
관련 파일 uc.com:

```
%_N_UC_COM
;$PATH=/_N_CUS_DIR
//C25(MY_CYCLE_1) usercycle_1
(R/-1000.001 123.456 / 100 /Parameter_2 of cycle)
(I/0 999999 / 1 / Integer value)
(C//"A" / Character parameter)
(S///Subprogram name)
//C26(SPECIALCYCLE)
(R///Entire length)
(I/*123456/3/Machining type)
M17
```

싸이클 MY_CYCLE_1 에 대한 표시 화면 양식



싸이클 SPECIAL CYCLE 에 대한 표시 화면 양식



1.26 매크로 방법 (DEFINE ... AS)

⚠ 주의
매크로를 사용할 경우 제어 시스템의 프로그래밍 언어가 상당히 변경될 수 있습니다 ! 따라서 매크로 사용 시 주의해야 합니다 .

기능

매크로는 일련의 개별 명령문이며 해당 이름이 함께 할당됩니다 . G, M 및 H 코드 또는 L 서브프로그램 이름도 매크로로 사용할 수 있습니다 . 프로그램 실행 동안 매크로를 호출하면 프로그램 이름 하에 프로그래밍된 명령문이 차례로 실행됩니다 .

적용

반복되는 명령 순서는 매크로가 전용 매크로 블록 (매크로 파일) 에 있거나 프로그램 시작 시 한 번만 프로그래밍됩니다 . 매크로는 메인 프로그램 또는 서브프로그램에서 호출하여 실행할 수 있습니다 .

활성화

NC 프로그램에서 매크로 파일의 매크로를 사용하려면 매크로 파일을 NC 로 다운로드해야 합니다 .

구문

매크로 정의 :
DEFINE <Macro name> AS <instruction 1> <instruction 2> ...

NC 프로그램에서의 호출 :
<macro name>

의미

DEFINE ... AS :	매크로를 정의하기 위한 키워드 조합
< 매크로 이름 >:	매크로 이름 식별자만 매크로 이름으로 허용됩니다 . 매크로는 매크로 이름을 사용하여 NC 프로그램에서 호출됩니다 .
< 연산 >:	매크로에 포함되어야 할 프로그래밍 명령 .

매크로 정의 시 규칙

- 식별자 , G, M, H 코드 및 L 프로그램 이름을 매크로에서 정의할 수 있습니다 .
- 또한 매크로를 NC 프로그램에서 정의할 수 있습니다 .
- G 코드 매크로는 전체 제어 시스템 (모달) 의 매크로 모듈에서만 전역적으로 정의할 수 있습니다 .
- H 및 L 코드는 2 자리로 프로그래밍할 수 있습니다 .
- M 및 G 코드는 3 자리를 사용하여 프로그래밍할 수 있습니다 .

⚠ 주의
키워드 및 예약된 이름은 매크로를 사용하여 다시 정의할 수 없습니다 .

일반 조건

매크로는 중첩될 수 없습니다 .

예제

예제 1: 프로그램 시작 시 매크로 정의

프로그램 코드	설명
DEFINE LINE AS G1 G94 F300	; 매크로 정의
...	
...	
N70 LINE X10 Y20	; 매크로 호출
...	

예제 2: 매크로 파일에서 매크로 정의

프로그램 코드	설명
DEFINE M6 AS L6	; 공구 교환 시 서브프로그램이 호출되어 필요한 데이터 전송을 수행합니다 . 실제 M 코드는 서브프로그램에서 출력됩니다 (예 : M106).
DEFINE G81 AS DRILL(81)	; DIN-G 코드 열거 .
DEFINE G33 AS M333 G333	; 나사 절삭 동안 PLC 에서 동기화가 요청됩니다 . 원래 G 코드 G33 은 머신 데이터에 의해 G333 으로 이름이 변경되었으며 사용자의 프로그래밍 방법은 동일합니다 .

예제 3: 외부 매크로 파일

외부 매크로 파일을 제어 시스템으로 판독한 후 매크로 파일을 NC 에 다운로드해야 합니다 . NC 프로그램에서만 매크로를 사용할 수 있습니다 .

프로그램 코드	설명
%_N_UMAC_DEF	
;$PATH=/_N_DEF_DIR	; 고객별 매크로
DEFINE PI AS 3.14	
DEFINE TC1 AS M3 S1000	
DEFINE M13 AS M3 M7	; CW 방향 스핀들 회전 , ON
DEFINE M14 AS M4 M7	; CCW 방향 스핀들 회전 , ON
DEFINE M15 AS M5 M9	; 스핀들 정지 , OFF
DEFINE M6 AS L6	; 공구 교환 프로그램 호출
DEFINE G80 AS MCALL	; 드릴링 사이클 선택 취소
M30	

파일 및 프로그램 관리

2

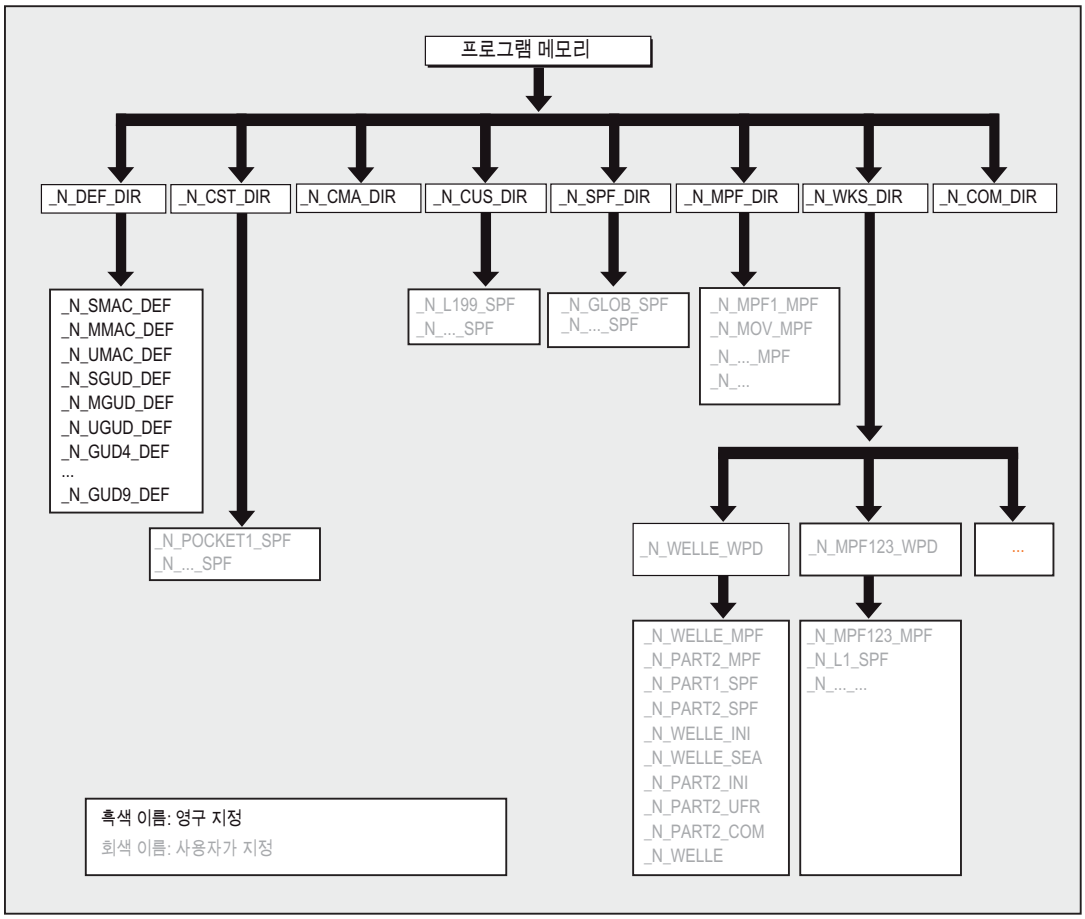
2.1 프로그램 메모리

기능

파일 및 프로그램 (예 : 메인 프로그램 및 서브프로그램 , 매크로 정의) 은 비활성 프로그램 메모리에 저장됩니다 (→ 패시브 파일 시스템).

참고 자료 :
기능 매뉴얼 , 확장 기능 ; 메모리 환경설정 (S7)

또한 다양한 파일 유형을 일시적으로 이 메모리에 저장하고 필요에 따라 작업 메모리로 전송할 수 있습니다 (예 : 특정 공작물 가공 시 초기화를 위해 전송).



표준 디렉토리

표준 디렉토리는 다음과 같습니다 .

폴더	내용
_N_DEF_DIR	데이터 모듈 및 매크로 모듈
_N_CST_DIR	표준 사이클
_N_CMA_DIR	제조업체 사이클
_N_CUS_DIR	사용자 사이클
_N_WKS_DIR	공작물
_N_SPF_DIR	글로벌 서브프로그램
_N_MPF_DIR	메인 프로그램
_N_COM_DIR	설명

파일 유형

메인 메모리에 다음과 같은 파일 유형을 저장할 수 있습니다 :

파일 유형	설명
이름 _MPF	메인 프로그램
이름 _SPF	서브프로그램
name_TEA	머신 데이터
name_SEA	셋팅 데이터
name_TOA	공구 옵셋
name_UFR	워크 옵셋 / 프레임
name_INI	초기화 파일
name_GUD	전역 사용자 데이터
name_RPA	R 파라미터
name_COM	설명
name_DEF	글로벌 사용자 데이터 및 매크로 정의

공작물 메인 디렉토리 (_N_WKS_DIR)

공작물 메인 디렉토리는 `_N_WKS_DIR` 이름으로 프로그램 메모리의 표준 설정에 있습니다 . 공작물 메인 디렉토리에는 프로그래밍된 공작물의 모든 공작물 디렉토리가 포함되어 있습니다 .

공작물 디렉토리 (..._WPD)

데이터 및 프로그램을 보다 유연하게 처리하기 위해 특정 데이터 및 프로그램을 함께 그룹화하거나 개별 공작물 디렉토리에 저장할 수 있습니다 .

공작물 디렉토리에는 공작물을 가공하는 데 필요한 모든 파일이 포함되어 있습니다 . 이러한 파일로 메인 프로그램 , 서브프로그램 , 초기화 프로그램 및 설명 파일이 포함될 수 있습니다 .

처음으로 가공 프로그램을 시작하면 선택한 프로그램에 따라 기계 데이터 MD11280 $MN_WPD_INI_MODE 에 의해 초기화 프로그램이 한 번 실행됩니다 .

예제 :

SHAFT 공작물에 대해 생성된 공작물 디렉토리 _N_WELLE_WPD 에는 다음 파일이 포함되어 있습니다 .

파일	의미
_N_SHAFT_MPF	메인 프로그램
_N_PART2_MPF	메인 프로그램
_N_PART1_SPF	서브프로그램
_N_PART2_SPF	서브프로그램
_N_SHAFT_INI	공작물 데이터에 대한 일반 초기화 프로그램
_N_SHAFT_SEA	데이터 초기화 프로그램 설정
_N_PART2_INI	가공 2 프로그램 데이터에 대한 일반 초기화 프로그램
_N_PART2_UFR	가공 2 프로그램 데이터에 대한 초기화 프로그램
_N_SHAFT_COM	설명 파일

외부 PC 에서 공작물 디렉토리 생성

아래 설명된 단계를 외부 데이터 스테이션에서 수행합니다 . 제어 시스템에서 직접 (PC 에서 제어 시스템) 사용 설명서의 파일 및 프로그램 관리를 참조하십시오 .

경로 이름 ($PATH=...) 을 가진 공작물 디렉토리 생성

대상 경로 `$PATH=...` 는 파일의 두 번째 라인에서 지정됩니다 . 그런 다음 파일이 지정된 경로에 저장됩니다 .

예 :

프로그램 코드

```
%_N_SHAFT_MPF
;$PATH=/_N_WKS_DIR/_N_SHAFT_WPD
N40 G0 X... Z...
...
M2
```

파일 _N_WELLE_MPF 가 디렉토리 /_N_WKS_DIR/_N_WELLE_WPD 에 저장됩니다 .

경로 이름 ($PATH=...) 없이 공작물 디렉토리 생성

경로 이름이 누락되면 _SPF 확장자를 가진 파일이 디렉토리 /_N_SPF_DIR 에 저장되고 _INI 확장자를 가진 파일은 RAM 에 저장되며 다른 모든 파일은 디렉토리 /_N_MPF_DIR 에 저장됩니다 .

예 :

프로그램 코드

```
%_N_SHAFT_SPF
...
M17
```

파일 _N_WELLE_SPF 가 디렉토리 /_N_SPF_DIR 에 저장됩니다 .

가공할 공작물 선택

채널에서 실행할 공작물 디렉토리를 선택할 수 있습니다 . **동일한 이름**을 가진 메인 프로그램 또는 단일 메인 프로그램 (_MPF) 을 이 디렉토리에 저장하면 자동으로 실행할 디렉토리가 선택됩니다 .

참조 문서 :
/BAD/ 작동 설명서 HMI Advanced; " 작업 목록 " 및 " 실행할 프로그램 선택 "

서브프로그램 호출의 검색 경로

서브프로그램 (또는 초기화 파일) 호출 시 가공 프로그램에서 명시적으로 검색 경로를 지정하지 않은 경우 호출 프로그램은 고정된 검색 경로에서 검색합니다 .

절대 경로를 통한 서브프로그램 호출

예제

프로그램 코드

```
...
CALL"/_N_CST_DIR/_N_CYCLE1_SPF"
...
```

절대 경로 없이 서브프로그램 호출

일반적으로 프로그램이 경로 지정 없이 호출됩니다 .

예제

프로그램 코드

```
...
CYCLE1
...
```

지정된 순서로 디렉토리에서 호출된 프로그램을 검색합니다 .

번호	폴더	의미
1	현재 디렉토리 / *이름*	공작물 메인 디렉토리 또는 표준 디렉토리 _N_MPF_DIR
2	현재 디렉토리 /*name_SPF*	
3	현재 디렉토리 /*name_MPF*	
4	/_N_SPF_DIR/*name_SPF*	글로벌 서브프로그램
5	/_N_CUS_DIR/*name_SPF*	사용자 싸이클
6	/_N_CMA_DIR/*name_SPF*	제조업체 싸이클
7	/_N_CST_DIR/*name_SPF*	표준 싸이클

서브프로그램 호출에 대한 검색 경로 프로그래밍 (CALLPATH)

CALLPATH 가공 프로그램 명령을 사용하여 서브프로그램 호출의 검색 경로를 확장합니다 .

예 :

프로그램 코드

```
CALLPATH ("/_N_WKS_DIR/_N_MYWPD_WPD")
...
```

검색 경로는 지정된 프로그램에 따라 위치 5(사용자 싸이클) 의 앞에 저장됩니다 .

CALLPATH 를 통한 서브프로그램 호출의 프로그램 가능한 검색 경로에 대한 자세한 내용은 "CALLPATH 를 통한 서브프로그램 호출의 검색 경로 확장 " 장을 참조하십시오 .

2.2 작업 메모리 (CHANDATA, COMPLETE, INITIAL)

기능

작업 메모리에는 제어 시스템에서 작동되는 현재 시스템 및 사용자 데이터가 포함되어 있습니다 (활성 파일 시스템). 예 :

- 활성 머신 데이터
- 공구 옵셋 데이터
- 워크 옵셋
- ...

초기화 프로그램

작업 메모리 데이터를 초기화하는 프로그램입니다 . 다음 파일 유형을 사용할 수 있습니다 .

파일 유형	의미
name_TEA	머신 데이터
name_SEA	셋팅 데이터
name_TOA	공구 옵셋
name_UFR	워크 옵셋 / 프레임
name_INI	초기화 파일
name_GUD	전역 사용자 데이터
name_RPA	R 파라미터

사용 설명서에서 조작자 인터페이스에 대한 모든 파일 유형 정보를 참조하십시오 .

데이터 영역

적용할 데이터를 서로 다른 영역에 구성할 수 있습니다. 예를 들어 한 제어 시스템이 여러 개의 채널을 갖거나 흔히 그렇듯이 여러 개의 축을 가질 수도 있습니다 .

다음이 존재합니다 :

식별자	데이터 영역
NCK	NCK 관련 데이터
CH<n>	채널 관련 데이터 (<n> 은 채널 이름)
AX<n>	축 관련 데이터 (<n> 은 가공 축 번호)
TO	공구 데이터
COMPLETE	모든 데이터

외부 PC 에서 초기화 프로그램 생성

데이터 영역 식별자 및 데이터 유형 식별자를 사용하여 영역을 결정할 수 있으며 데이터가 저장될 때 하나의 단위로 취급되어야 합니다 .

_N_AX5_TEA_INI	축 5 의 머신 데이터
_N_CH2_UFR_INI	채널 2 의 프레임
_N_COMPLETE_TEA_INI	모든 머신 데이터

제어 시스템을 처음으로 시작하면 데이터 세트가 자동으로 로드되어 제어 시스템을 적절하게 작동합니다 .

다중 채널 제어 절차 (CHANDATA)

여러 채널에 대한 `CHANDATA(<channel number>)` 는 파일 _N_INITIAL_INI 에서만 허용됩니다 . 이 파일은 제어 시스템의 모든 데이터를 초기화하는 테스트 파일입니다 .

프로그램 코드	설명
%_N_INITIAL_INI	
CHANDATA(1)	
	; 기계 축 할당 , 채널 1:
$MC_AXCONF_MACHAX_USED[0]=1	
$MC_AXCONF_MACHAX_USED[1]=2	
$MC_AXCONF_MACHAX_USED[2]=3	
CHANDATA(2)	
	; 기계 축 할당 , 채널 2:
$MC_AXCONF_MACHAX_USED[0]=4	
$MC_AXCONF_MACHAX_USED[1]=5	
CHANDATA(1)	
	; 축 관련 머신 데이터 :
	; 일반 정위치 정지 창 :
$MA_STOP_LIMIT_COARSE[AX1]=0.2	; 축 1
$MA_STOP_LIMIT_COARSE[AX2]=0.2	; 축 2
	; ; 미세 정위치 정지 창 :
$MA_STOP_LIMIT_FINE[AX1]=0.01	; 축 1
$MA_STOP_LIMIT_FINE[AX1]=0.01	; 축 2

⚠ 주의

CHANDATA 문

가공 프로그램에서 `CHANDATA` 명령은 NC 프로그램을 실행하는 해당 채널에 대해서만 설정할 수 있습니다 . 예를 들어 이 명령을 사용하여 잘못된 채널에서는 실행되지 않도록 NC 프로그램을 보호할 수 있습니다 .

오류가 발생할 경우 진행 중인 프로그램이 중단됩니다 .

주

작업 목록의 INI 파일에는 CHANDATA 명령이 없습니다 .

초기화 프로그램 저장 (COMPLETE, INITIAL)

작업 메모리의 파일을 외부 PC 에 저장한 다음 읽을 수 있습니다 .

- 파일은 COMPLETE 를 통해 저장됩니다 .
- INITIAL 을 사용하여 모든 영역에서 INI 파일 (_N_INITIAL_INI) 을 생성합니다 .

초기화 프로그램 읽기

유의사항
"INITIAL_INI" 이름을 가진 파일을 읽으면 파일에서 제공하지 않는 모든 데이터는 표준 데이터를 사용하여 초기화됩니다 . 머신 데이터만 예외입니다 . 따라서 **설정 데이터 , 공구 데이터 , ZO, GUD 값 , ...** 은 표준 데이터와 함께 제공됩니다 (일반적으로 "ZERO").

예를 들어 개별 머신 데이터를 읽는 경우 COMPLETE_TEA_INI 파일이 적합합니다 . 제어 시스템은 이 파일에서 머신 데이터만 필요합니다 . 이 경우에서 다른 데이터 영역은 영향을 받지 않기 때문입니다 .

초기화 프로그램 로딩

또한 한 채널의 데이터만 사용하는 경우 INI 프로그램을 선택하여 가공 프로그램으로 호출할 수 있습니다 . 따라서 프로그램 제어 데이터를 초기화할 수도 있습니다 .

2.3 단계 편집기에서 구조화 명령 (SEFORM)

작동

구조화 명령 SEFORM 이 단계 편집기에서 평가되어 (편집기 기반 프로그램 지원) HMI Advanced 에 대한 단계 뷰를 생성합니다. 단계 뷰는 NC 서브프로그램의 판독성을 향상시키는 데 사용됩니다 .

구문

```
SEFORM(<section name>,<level>,<icon>)
```

의미

SEFORM()	파라미터 <section name>, <level> 및 <icon> 을 통한 구조화 명령의 기능 호출		
<section name>	작업 단계의 식별자		
	유형 :	STRING	
<level>	메인 또는 서브 레벨의 인덱스		
	유형 :	INT	
	값 :	0	메인 레벨
		1, ..., <n>	하위 레벨 1, ... , 하위 레벨 <n>
<icon>	이 섹션에 대해 표시된 아이콘 이름 .		
	유형 :	STRING	

주

SEFORM 명령은 단계 편집기에서 생성됩니다 .

<section name> 파라미터를 통해 전송된 문자열은 MSG 명령과 비슷한 방법으로 OPI 변수에 메인 - 런 - 동기식으로 저장됩니다 . 정보는 다음 SEFORM 명령에 의해 덮어쓸 때까지 유지됩니다 . 리셋 및 가공 프로그램 종류 시 내용이 지워집니다 .

가공 프로그램 실행 동안 NCK 는 파라미터 <level> 및 <icon> 을 확인하지만 추가로 처리하지 않습니다 .

참조 문서

편집기 기반 프로그램 지원에 대한 자세한 내용은
작동 설명서 HMI Advanced 를 참조하십시오 .

보호 영역

3

3.1 보호 영역의 정의 (CPROTDEF, NPROTDEF)

기능

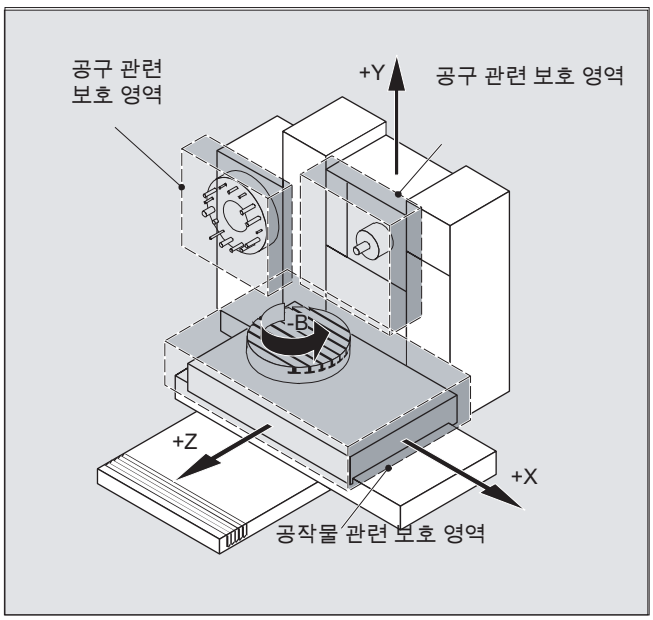
보호 영역을 사용하여 기계의 다양한 요소 , 구성 요소 및 공작물을 잘못된 이동으로부터 보호합니다 .

공구 방향 보호 영역 :

공구에 속한 부품의 경우 (예 : 공구 , 공구 홀더)

공작물 방향 보호 영역 :

공작물에 속한 부품의 경우 (예 : 공작물 부품 , 클램핑 테이블 , 클램핑 개수 , 스핀들 척 , 테일스톡).



구문

```
DEF INT NOT_USED
G17/G18/G19
CPROTDEF/NPROTDEF(<n>,<t>,<applim>,<applus>,<appminus>)
G0/G1/... X/Y/Z...
...
EXECUTE(NOT_USED)
```

의미

DEF INT NOT_USED:	INTERGER 데이터 유형의 로컬 변수 정의 (' 동기 동작 [쪽 551]' 장 참조)
G17/G18/G19:	필요한 평면은 G17/G18/G19 을 사용하여 CPROTDEF 또는 NPROTDEF 이전에 선택하고 EXECUTE 이전에는 변경하지 않아야 합니다 . CPROTDEF 또는 NPROTDEF 및 EXECUTE 사이에 직교 좌표을 프로그래밍하지 않아야 합니다 .
CPROTDEF:	채널별 보호 영역 정의 (NCU 572/573 만 해당)
NPROTDEF:	기계 관련 보호 영역 정의
G0/G1/... X/Y/Z... ... :	보호 영역의 형상은 선택한 평면에서 최대 11 개의 이송으로 지정됩니다 . 첫 번째 이송은 형상 이동입니다 . 유효한 보호 영역은 형상의 왼쪽 영역입니다 . **참고 :** CPROTDEF 또는 NPROTDEF 와 EXECUTE 사이에 프로그래밍된 이송 모션은 실행되는 것이 아니라 보호 영역만 정의합니다 .
EXECUTE:	정의 종료
<n>:	정의된 보호 영역 수
<t>:	보호 영역의 유형 TRUE: **공구** 관련 보호 영역 FALSE: **공작물** 관련 보호 영역
<applim>:	3 차원에서의 제한 유형 0: 제한 없음 1: 양의 방향 한계 2: 마이너스 방향으로 제한 3: 플러스 방향 및 마이너스 방향으로 제한
<applus>:	양의 방향 및 3 차원에서의 제한 값
<appminus>:	마이너스 방향 및 3 차원에서의 제한 값
NOT_USED:	에러 변수는 EXECUTE 를 통한 보호 영역에 적용되지 않습니다 .

일반 조건

보호 영역의 정의 도중

- 커터 또는 공구 노즈의 모든 공구 반경 보정이 비활성화되어야 합니다 .
- 실행 중인 변환이 없어야 합니다 .
- 실행 중인 프레임이 없어야 합니다 .

원점 복귀 (G74), 고정 정지점 복귀 (G75), 블록 전처리 정지 또는 프로그램 종료를 프로그래밍해서는 안됩니다 .

추가 정보

보호 영역 정의

보호 영역 정의는 다음과 같습니다 .

- 채널 관련 보호 영역의 경우 CPROTDEF
- 기계 관련 보호 영역의 경우 NPROTDEF
- 보호 영역에 대한 형상 설명
- EXECUTE 를 통한 정의 종료

보호 영역이 NC 가공 프로그램에서 활성화되면 보호 영역의 기준점에 대한 상대 옵셋을 지정할 수 있습니다 .

형상 설명 기준점

공작물 방향 보호 영역은 기본 좌표계에 정의됩니다 .

공구 방향 보호 영역은 공구 케리어 기준점 F 를 기준으로 정의됩니다 .

허용 가능한 형상 요소

보호 영역의 형상을 정의하는데 허용되는 요소는 다음과 같습니다 .

- 직선 형상 요소의 경우 G0, G1
- CW 방향 원호 세그먼트의 경우 G2 (공작물 방향 보호 영역에서만)
- CCW 방향 원호 세그먼트의 경우 G3

주

동심원으로 보호 영역을 설명하는 경우 두 개의 반원으로 나누어야 합니다 . G2, G3 또는 G3, G2 의 연속적인 순서는 허용되지 않습니다 . 필요한 경우 짧은 G1 블록을 삽입해야 합니다 .

형상 설명의 마지막 점이 첫 번째 점과 일치해야 합니다 .

외부 보호 영역

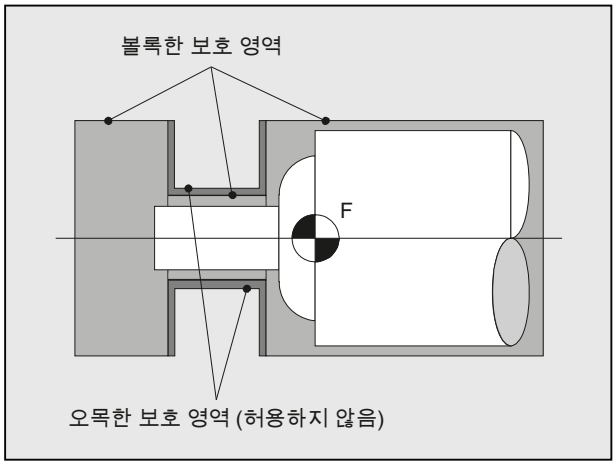
외부 보호 영역 (공작물 관련 보호 영역에서만 가능) 은 CW 방향으로 정의해야 합니다 .

회전 중심에서 대칭되는 보호 영역

축 또는 회전 중심에서 대칭되는 보호 영역 (예: 스핀들 척) 의 경우 회전 중심까지가 아닌 형상 전체를 지정해야 합니다 !

공구 관련 보호 영역

공구 관련 보호 영역은 항상 볼록해야 합니다 . 오목한 보호 영역을 원하는 경우 여러 개의 볼록한 보호 영역으로 나누어야 합니다 .



3.2 보호 영역 활성화 / 비활성화 (CPROT, NPROT)

기능

충돌 모니터링 및 보호 영역 비활성화를 위해 이전에 정의된 보호 영역의 활성화 및 사전 활성화 .

동일한 채널에서 동시에 활성화할 수 있는 최대 보호 영역 수는 머신 데이터에서 정의됩니다 .

공구 관련 보호 영역이 활성화되지 않으면 공작물 관련 보호 영역에 대한 공구 경로를 확인합니다 .

주

공작물 관련 보호 영역이 활성화되지 않으면 보호 영역 모니터링은 수행되지 않습니다 .

구문

CPROT(<n>,< 상태 >,<xMov>,<yMov>,<zMov>)
NPROT(<n>,< 상태 >,<xMov>,<yMov>,<zMov>)

의미

CPROT:	채널 관련 보호 영역 호출 (NCU 572/573 만 해당)
NPROT:	기계 관련 보호 영역 호출
<n>:	보호 영역의 개수
< 상태 >:	상태 파라미터
	0: 보호 영역 비활성화
	1: 보호 영역 사전 활성화
	2: 보호 영역 활성화
	3: 조건부 정지를 통한 보호 영역 사전 활성화
<xMov>,<yMov>,<zMov>:	형상 축에서 정의된 보호 영역 이동

일반 조건

공구 반경 보정이 활성인 경우 보호 영역 모니터링

공구 반경 보정이 활성인 경우 공구 반경 보정의 평면이 보호 영역 정의 평면과 일치하는 경우에만 보호 영역을 모니터링할 수 있습니다 .

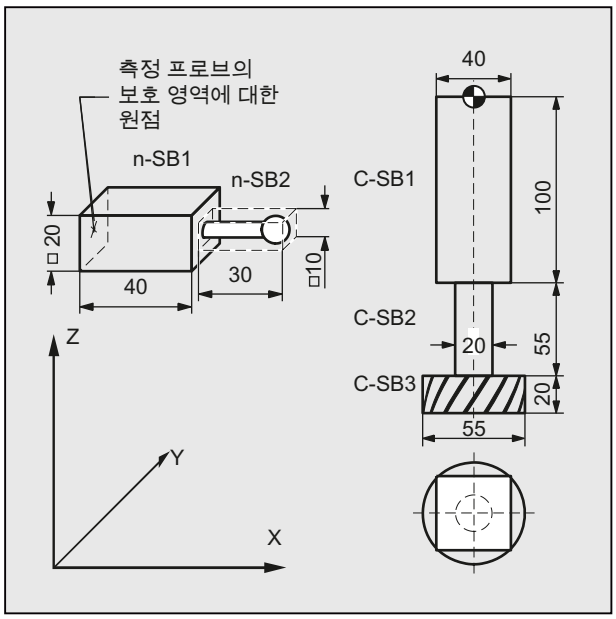
예제

측정 프로브를 통해 밀링 커터의 충돌 가능성이 밀링 기계에서 모니터링되어야 합니다 . 측정 프로브의 위치는 기능 작동 시 옵셋에 의해 정의되어야 합니다 . 이를 위해 다음과 같은 보호 영역이 정의됩니다 .

- 측정 프로브 홀더 (n-SB1) 및 측정 프로브 자체 (n-SB2) 에 대한 기계 관련 및 공작물 관련 보호 영역 .
- 밀링 커터 홀더 (c-SB1), 커터 직경 (c-SB2) 및 밀링 커터 자체 (c-SB3) 에 대한 채널 관련 및 공구 방향 보호 영역 .

모든 보호 영역의 오리엔테이션은 Z 방향입니다 .

기능 작동 시 측정 프로브의 원점 위치는 X = -120, Y = 60 및 Z = 80 이어야 합니다 .



프로그램 코드	설명
DEF INT PROTECTB	; 도움말 변수 정의
보호 영역 G17 정의	; 설정 오리엔테이션
NPROTDEF(1,FALSE,3,10,–10)G01 X0 Y–10 X40 Y10 X0 Y–10 EXECUTE(PROTECTB)	; 보호 영역 n–SB1
NPROTDEF(2,FALSE,3,5,–5) G01 X40 Y–5 X70 Y5 X40 Y-5 EXECUTE(PROTECTB)	; 보호 영역 n–SB2

프로그램 코드	설명
CPROTDEF(1,TRUE,3,0,–100) G01 X–20 Y–20 X20 Y20 X–20 Y-20 EXECUTE(PROTECTB)	; 보호 영역 c–SB1
CPROTDEF(2,TRUE,3,–100,–150) G01 X0 Y–10 G03 X0 Y10 J10 X0 Y–10 J–10 EXECUTE(PROTECTB)	; 보호 영역 c–SB2
CPROTDEF(3,TRUE,3,–150,–170) G01 X0 Y-27.5 G03 X0 Y27,5 J27,5 X0 Y27,5 J–27,5 EXECUTE(PROTECTB)	; 보호 영역 c–SB3
보호 영역 활성화 :	
NPROT(1,2,–120,60,80)	; 옵셋을 통한 보호 영역 n–SB1 활성화
NPROT(2.2,–120,60,80)	; 옵셋을 통한 보호 영역 n–SB2 활성화
CPROT(1,2,0,0,0)	; 옵셋을 통한 보호 영역 c–SB1 활성화
CPROT(2,2,0,0,0)	; 옵셋을 통한 보호 영역 c–SB2 활성화
CPROT(3,2,0,0,0)	; 옵셋을 통한 보호 영역 c–SB3 활성화

추가 정보

활성화 상태 (< 상태 >)

- **< 상태 >=2**

  보호 영역은 일반적으로 상태 = 2 의 가공 프로그램에서 활성화됩니다 .

  상태는 기계 방향 보호 영역에서도 항상 채널과 관련됩니다 .

- **< 상태 >=1**

  PLC 사용자 프로그램에서 PLC 사용자 프로그램에 의해 보호 영역이 효율적으로 설정되도록 하는 경우 필요한 사전 활성화는 상태 = 1 에서 구현됩니다 .

- **< 상태 >=3**

  조건부 정지를 통한 사전 활성화의 경우 , 시스템은 위반한 사전 활성화 보호 영역 앞에서는 절대로 정지하지 않습니다 . 정지는 보호 영역이 활성화된 상태에서만 가능합니다 . 정지 기능을 사용하면 특별한 경우에 보호 영역이 활성화된 상태에서 중단없는 프로세싱을 가능하게 합니다 . 주의 : 시스템은 모션 바로 전에 보호 영역이 활성화된 상태에서 정지 동작의 결과로 보호 영역으로 이동할 수 있습니다 .

  조건부 정지를 통한 사전 활성화에는 상태 = 3 을 사용합니다 .

- **< 상태 >=0**

  상태 = 0 으로 설정하면 보호 영역이 비활성화됩니다 . 변경할 필요는 없습니다 .

활성화를 위한 보호 영역의 이동

옵셋을 1 차원 , 2 차원 또는 3 차원에서 수행할 수 있습니다 . 옵셋은 다음을 의미합니다 .

- 공작물 관련 보호 영역의 기계 제로 .
- 공구 관련 보호 영역에서 공구 캐리어 기준점 F.

부팅 이후 상태

부팅 후 바로 보호 영역을 활성화할 수 있으며 후속적으로 기준점에 복귀합니다 . 이를 위해 시스템 변수 $SN_PA_ACTIV_IMMED[<n>] 또는 $SC_PA_ACTIV_IMMED[<n>] 를 TRUE 로 설정해야 합니다 . 이러한 변수는 항상 상태 = 2 로 활성화되며 옵셋이 없습니다 .

보호 영역의 다중 활성화

여러 채널에서 보호 영역을 동시에 활성화할 수 있습니다 (예 : 두 개의 반대 측면이 있는 테일스톡). 모든 형상 축이 원점 복귀된 경우에만 보호 영역이 모니터링됩니다 .

다음이 적용됩니다 .

- 단일 채널에서 서로 다른 옵셋 상태에서 보호 영역을 동시에 활성화할 수 없습니다 .
- 기계 방향 보호 영역은 두 채널에서 동일한 오리엔테이션을 가져야 합니다 .

3.3 보호 영역 위반 , 작업 영역 제한 및 소프트웨어 제한 점검 (CALCPOSI)

작동

CALCPOSI 함수는 정의된 시작점부터 시작하여 형상 축이 축 제한 (소프트웨어 제한), 작업 영역 제한 또는 보호 영역 위반 없이 정의된 경로로 이송할 수 있는지 여부를 점검하기 위한 함수입니다 .

정의된 경로로 이송할 수 없는 경우 최대 허용 가능한 경로가 반환됩니다 .

CALCPOSI 함수는 미리 정의된 서브프로그램입니다 . 블록에 단독으로 있어야 합니다 .

구문

Status=CALCPOSI(_STARTPOS, _MOVDIST, _DLIMIT, _MAXDIST, _BASE_SYS, _TESTLIM)

의미

상태

0: 기능 정상 ,
지정된 경로를 완벽하게 이송할 수 있습니다 .

–: 최소한 하나의 구성 요소가 _DLIMIT 에서 음수

–: 변환 계산 시 오류 발생

지정된 경로를 완벽하게 이송할 수 없는 경우 양의 십진수 값이 반환됩니다 .

단위 숫자 (위반한 제한 유형):

1: 소프트웨어 제한은 이송 경로를 제한합니다 .

2: 작업 영역 제한은 이송 경로를 제한합니다 .

3: 소프트웨어 제한은 이송 경로를 제한합니다 .

여러 제한을 동시에 위반할 경우 (예를 들어 소프트웨어 제한 및 보호 영역) 지정된 이송 경로에서 가장 중요한 제한이 되는 제한이 단위 숫자로 표시됩니다 .

10 자리 숫자

10:

초기값이 제한을 위반합니다 .

20:

지정된 직선이 제한을 위반합니다 . 또한 이 값은 끝점이 제한 자체를 위반하지 않지만 시작점부터 끝점 사이 경로에서 제한값을 위반하는 경우 반환됩니다 (예를 들어 보호 영역을 통과하여 비선형 변환에 대해 WCS 의 곡선 소프트웨어 제한 , 또는 전송).

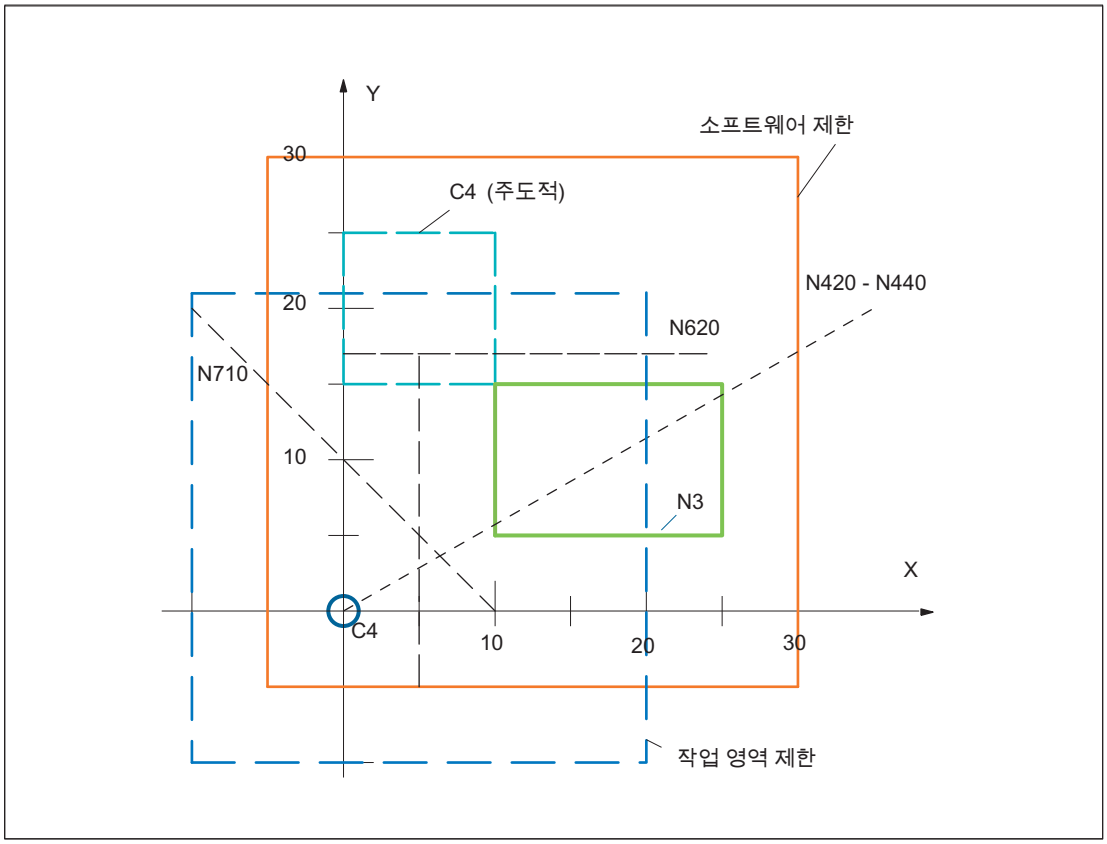
	100 자리 숫자 100: 양의 제한값이 위반됩니다 (단위 숫자가 1 또는 2 인 경우에만 해당 , 예를 들어 소프트웨어 제한 및 작업 영역 제한). 100: NCK 보호 영역이 위반됩니다 (단위 숫자가 3 인 경우에만 해당). 200: 음의 제한값이 위반됩니다 (단위 숫자가 1 또는 2 인 경우에만 해당 , 예를 들어 소프트웨어 제한 및 작업 영역 제한). 200: 채널 관련 보호 영역이 위반됩니다 (단위 숫자가 3 인 경우에만 해당). **1000 자리 숫자** 1000: 제한을 위반하는 축 개수를 인자와 곱합니다 (단위 숫자가 1 또는 2 인 경우에만 해당 , 예를 들어 소프트웨어 제한 및 작업 영역 제한). 축 계산은 1 부터 시작하고 위반된 소프트웨어 제한 (단위 숫자 = 1) 의 경우 기계 축을 참조하고 위반된 작업 영역 제한 (단위 숫자 = 2) 의 경우 형상 축을 참조합니다 . 1000: 위반된 보호 영역의 개수를 인자와 곱합니다 (단위 숫자가 3 인 경우에만 해당). 여러 보호 영역이 위반된 경우 최대 이송 경로 제한이 적용된 제한이 100 및 1000 자리의 보호 영역으로 표시됩니다 .
_STARTPOS	(WCS) 에서 시작값이 가로축인 경우 [0], 세로축인 경우 [1] 및 직교축인 경우 [2]
_MOVEDIST	증분 경로 정의가 가로축인 경우 [0], 세로축인 경우 [1] 및 직교축인 경우 [2]
_DLIMIT	[0] - [2]: 형상 축에 할당된 최소 간극 . [3]: 형상 축을 고유하게 할당할 수 없는 경우 비선형 변환에 대해 선형 기계 축에 할당된 최소 간극 . [4]: 형상 축을 고유하게 할당할 수 없는 경우 비선형 변환에 대해 회전 기계 축에 할당된 최소 간극 . SW 제한을 모니터링해야 하는 경우 특별한 변환에만 해당합니다 .
_MAXDIST	반환값인 배열 [0] - [2]. 관련 기계 축에서 축 제한에 대해 정의된 최소 간극을 위반하지 않는 세 가지 모든 형상 축의 증분 경로 . 이송 경로가 제한되지 않을 경우 이 반환 파라미터의 내용은 _MOVDIST 의 내용과 동일합니다 .

_BASE_SYS	FALSE 또는 파라미터가 지정되지 않음 : 위치 및 길이 데이터를 평가하는 경우 그룹 13 의 G 코드 (G70, G71, G700, G710; 인치 / 미터) 가 평가됩니다 . G70 이 활성 상태이고 기본 시스템이 미터 단위인 경우 또는 G71 이 활성 상태이고 인치 단위인 경우 WCS 시스템 변수 $AA_IW[X] 및 $AA_MW[X]) 가 기본 시스템에서 제공되며 필요한 경우 CALCPOSI 함수를 사용하여 다시 변환해야 합니다 . TRUE : 위치 및 길이 데이터를 평가하는 경우 그룹 13 의 활성 G 코드 값과 관계 없이 항상 제어 시스템의 기본 시스템이 사용됩니다 .
_TESTLIM	점검할 제한 사항 (이진 코드): 1: 소프트웨어 제한 모니터 2: 작업 영역 제한 모니터 3: 활성화된 보호 영역 모니터 4: 사전 활성화된 보호 영역 모니터 값을 추가하여 조합합니다 . 기본값 : 15; 모두 확인합니다 .

예제

예제에서(그림 참조) X 소프트웨어 제한 및 작업 영역 제한이 표시되어 있습니다. 또한 두 개의 채널 관련 보호 영역 C2 및 C4 와 NCK 보호 영역 N3 의 세 가지 보호 영역이 정의되어 있습니다 . C2 는 반지름 2mm 인 원형의 활성 공구 관련 보호 영역입니다 . C4 는 측면 길이 10mm 인 사각형의 사전 활성화된 공작물 관련 보호 영역이며 N3 는 측면 길이 10mm 및 15mm 인 직사각형의 활성 보호 영역입니다 . 다음 NC 에서 처음에는 보호 영역 및 작업 필드

제한이 스케치로 정의되고 CALCPOSI 함수가 다양한 파라미터 지정과 함께 호출됩니다 . 각 개별 CALCPOSI 호출의 결과는 예제의 끝에 있는 표에 요약되어 있습니다 .



프로그램 코드	비고
N10 def real _STARTPOS[3]	
N20 def real _MOVDIST[3]	
N30 def real _DLIMIT[5]	
N40 def real _MAXDIST[3]	
N50 def int _SB	
N60 def int _STATUS	
N70 cprotdef(2, true, 0)	; 공구 관련 보호 영역
N80 g17 g1 x–y0	
N90 g3 i2 x2	
N100 i–x–	
N110 execute(_SB)	

프로그램 코드	비고
N120 cprotdef(4, false, 0)	; 공작물 관련 보호 영역
N130 g17 g1 x0 y15	
N140 x10	
N150 y25	
N160 x0	
N170 y15	
N180 execute(_SB)	
N190 nprotdef(3, false, 0)	; 기계 관련 보호 영역
N200 g17 g1 x10 y5	
N210 x25	
N220 y15	
N230 x10	
N240 y5	
N250 execute(_SB)	
N260 cprot(2,2,0, 0, 0)	; 보호 영역 활성화 또는 사전 활성화
N270 cprot(4,1,0, 0, 0)	
N280 nprot(3,2,0, 0, 0)	
N290 g25 XX=–YY=–	; 작업 영역 제한 그룹 정의
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.	
; 다양한 함수 호출	; 기타 시작점
N420 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST)	
N430 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,,3)	
N440 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,,1)	
N450 _STARTPOS[0] = 5.	; 기타 대상
N460 _STARTPOS[1] = 17.	
N470 _STARTPOS[2] = 0.	

프로그램 코드	비고
N480 _MOVDIST[0] = 0.	
N490 _MOVDIST[1] =–.	
N500 _MOVDIST[2] = 0.	
; 다양한 함수 호출	
N510 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,,14)	
N520 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,, 6)	
N530 _DLIMIT[1] = 2.	
N540 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,, 6)	
N550 _STARTPOS[0] = 27.	
N560 _STARTPOS[1] = 17.1	
N570 _STARTPOS[2] = 0.	
N580 _MOVDIST[0] =–.	
N590 _MOVDIST[1] = 0.	
N600 _MOVDIST[2] = 0.	
N610 _DLIMIT[3] = 2.	
N620 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,, 12)	
N630 _STARTPOS[0] = 0.	
N640 _STARTPOS[1] = 0.	
N650 _STARTPOS[2] = 0.	
N660 _MOVDIST[0] = 0.	
N670 _MOVDIST[1] = 30.	
N680 _MOVDIST[2] = 0.	
N690 trans x10	
N700 arot z45	
N710 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST)	
N720 M30	

예제에서 테스트 결과 :

Block No. N...	_STATUS	_MAXDIST [0] (= X)	_MAXDIST [1] (= Y)	추가 설명
420	3123	8.040	4.594	보호 영역 SB N3 이 위반되었습니다 .
430	1122	20.000	11.429	보호 영역 모니터링이 없으며 작업 영역 제한이 위반되었습니다 .
440	1121	30.000	17.143	이제 소프트웨어 제한 모니터링만 활성화됩니다 .
510	4213	0.000	0.000	시작점이 보호 C4 를 위반합니다 .
520	0000	0.000	–.000	사전 활성화된 보호 영역 C4 가 모니터링되지 않습니다 . 정의된 경로로 완벽하게 이송할 수 있습니다 .
540	2222	0.000	–.000	_DLIMIT[1]=2 로 인해 이송 경로가 작업 영역 제한에 의해 제한 받습니다 .
620	4223	–.000	0.000	C4에 대한 간극은 C2 및 _DLIMIT[3]으로 인해 총 4mm 입니다 . 0.1mm 의 C2 – N3 간극이 이송 경로를 제한하지 않습니다 .
710	1221	0.000	21.213	전환 및 회전 활성 상태인 프레임 . _MOVDIST 에서 허용 가능한 이송 경로가 전환 및 회전 좌표계 (WCS) 에서 적용됩니다 .

특별한 경우 및 추가 세부 정보

활성 G 코드 "DIAMON" 을 가진 페이싱 축의 경우에도 항상 모든 경로 데이터가 반경으로 입력됩니다. 관련 축 중 특정 축 일부를 완전히 이송할 수 없는 경우 결과 끝점이 지정된 경로에 위치하도록 다른 축의 경로도 _MAXDIST 반환값에 따라 감소됩니다 .

하나 이상의 관련 축에 대해 소프트웨어 제한 , 조작 범위 제한 또는 보호 영역이 정의되지 않을 수 있습니다 . 관련 축이 원점 복귀하는 경우에만 모든 제한이 모니터링됩니다 . 관련 로타리 축은 모듈로 축이 아닌 경우에만 모니터링됩니다 .

일반적인 이송 작동 시 소프트웨어 제한 및 조작 범위 제한에 대한 모니터링은 실제 설정에 따라 다릅니다 (소프트웨어 제한 1 또는 소프트웨어 제한 2 를 선택한 경우 인터페이스 신호 , 조작 범위 제한의 특정 활성화 및 활성 공구의 반경을 조작 범위 제한 모니터링에 대해 고려할지 여부를 지정하는 경우 GWALIMON/WALIMOF 셋팅 데이터).

특정 운동학적 변형 (예 : TRANSMIT) 의 경우 공작물 좌표계 (WCS) 의 위치로부터 기계 축의 위치를 고유하게 결정할 수 없습니다 . 일반적인 이송 작동 시 고유성은 일반적으로 이전 이력 및 WCS 에서의 연속적인 이동은 기계 축의 연속적인 이동에 해당해야 한다는 조건으로부터 실현됩니다. CALCPOSI 함수를 사용하여 소프트웨어 제한을 모니터링할 때 현재 기계 위치를 사용하여 고유하지 않은 결정을 확인합니다 . 필요한 경우 함수에 대해 유효한 기계 축 위치를 입력하기 위해 CALCPOSI 이전에 **STOPRE** 를 프로그래밍해야 합니다 .

_DLIMIT[3] 에서 지정된 보호 영역에 대한 구분은 지정된 이송 경로에서 이동하는 경우 유지될 수 없는 경우도 있습니다. 따라서 _MOVDIST에서 반환된 끝점이 이 거리만큼 길어지면 직선이 보호 영역에 매우 가까이 통과할 수 있더라도 보호 영역이 위반되지 않습니다 .

주

작업 영역 제한에 대한 자세한 내용은
/PG/ 기본 프로그래밍 설명서 ,

/FB1/ 작동 설명서 , 기본 기능 ; 축 모니터링 , 보호 영역 (A3) 의 소프트웨어 제한을 참조하십시오 .

특수 동작 명령 4

4.1 코드 위치 복귀 (CAC, CIC, CDC, CACP, CACN)

작동

다음 명령을 사용하여 위치 번호를 통해 선형 및 로타리 축을 기계 데이터 테이블에 저장된 고정 축 위치로 이송할 수 있습니다 . 이러한 유형의 프로그램을 " 코드 위치 복귀 " 라고 합니다 .

구문

CAC(<n>)
CIC(<n>)
CACP(<n>)
CACN(<n>)

의미

CAC(<n>)	위치 수 n 에서 코드 위치로 복귀합니다 .
CIC(<n>)	실제 위치 수부터 시작하여 코드 위치를 이전(+n) 또는 뒤로(-n) n 위치로 복귀합니다 .
CDC(<n>)	최단 경로를 따라 위치 수 n 에서 해당 위치로 복귀합니다 (로타리 축에만 해당).
CACP(<n>)	양의 방향으로 위치 수 n 에서 코드 위치로 복귀합니다 (로타리 축에만 해당).
CACN(<n>)	음의 방향으로 위치 수 n 에서 코드 위치로 복귀합니다 (로타리 축에만 해당).
<n>	기계 데이터 테이블 내의 위치 수 값 범위 : 0, 1, … (테이블 위치 최대값 - 1)

예 : 위치 지정 축의 코드 위치 복귀

프로그램 코드	비고
N10 FA[B]=300	; 위치 지정 축 B 의 피드 속도
N20 POS[B]=CAC(10)	; 위치 수 10 에서 코드 위치로 복귀합니다 .
N30 POS[B]=CIC(-4)	; " 현재 위치 수 " -4 에서 코드 위치로 복귀합니다 .

참조 문서

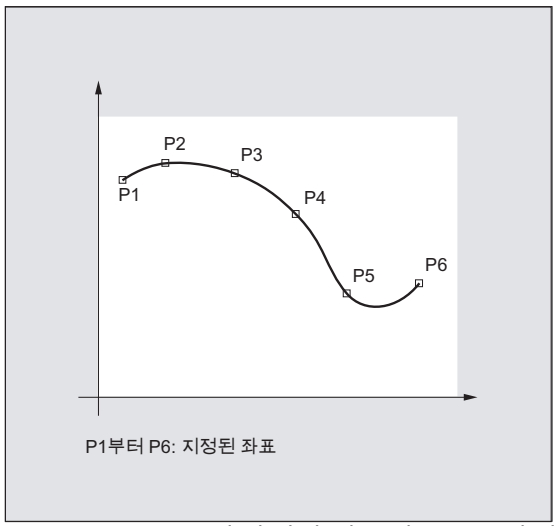
- 작동 설명서 확장 기능 ; 인덱싱 축 (T1)
- 작동 설명서 , 동기화된 작동

4.2 스플라인 보간 (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL)

작동

랜덤 곡선 공작물 형상은 분석 양식에서 정밀하게 정의할 수 없습니다 . 이러한 유형의 형상은 표면을 디지털화하는 경우와 같이 곡선을 따라 제한된 수의 점을 사용하여 근사치를 사용하기 때문입니다 . 공작물의 디지털화된 표면을 생성하려면 곡선을 따라 점을 연결하여 형상을 정의해야 합니다 . 스플라인 보간은 이 작업을 허용합니다 .

스플라인은 2 차 또는 3 차 다항식을 통해 산출된 곡선을 정의합니다 . 스플라인 곡선을 따라 형성된 점의 특성은 **사용 중인 스플라인 유형에 따라** 정의할 수 있습니다 .



SINUMERIK 솔루션 라인의 경우 다음 스플라인 유형을 사용할 수 있습니다 :

- A 스플라인
- B 스플라인
- C 스플라인

구문

일반 :
ASPLINE X... Y... Z... A... B... C...
BSPLINE X... Y... Z... A... B... C...
CSPLINE X... Y... Z... A... B... C...

B 스플라인의 경우 다음을 추가로 프로그래밍할 수 있습니다 :
PW=<n>
SD=2
PL=<value>

A 및 C 스플라인의 경우 다음을 추가로 프로그래밍할 수 있습니다 :

BAUTO / BNAT / BTAN

EAUTO / ENAT / ETAN

의미

스플라인 보간 유형 :

ASPLINE	A 스플라인 보간을 활성화하기 위한 명령
BSPLINE	B 스플라인 보간을 활성화하기 위한 명령
CSPLINE	C 스플라인 보간을 활성화하기 위한 명령
	ASPLINE, BSPLINE 및 CSPLINE 명령은 모달식으로 적용되며 이동 명령 그룹에 속합니다 .

곡선을 따라 점 및 확인 점 :

X... Y... Z... A... B... C...	직교 좌표계에서의 위치

점 가중치 (B 스플라인만 해당):

PW	PW 명령을 사용하면 소위 곡선을 따라 모든 점에 대해 " 점 가중치 " 를 프로그래밍할 수 있습니다 .		
<n>	" 점 가중치 "		
	값의 범위 :	0 ≤ n ≤ 3	
	증분 :	0.0001	
	결과 :	n > 1	확인 점은 곡선을 상당히 많이 끌어당깁니다 .
		n < 1	확인 점은 곡선을 상당히 적게 끌어당깁니다 .

스플라인 차수 (B 스플라인만 해당):

SD	3 차 다각형이 표준으로 사용되지만 SD=2 를 프로그래밍하면 2 차 다각형도 사용할 수 있습니다 .

노드 사이 거리 (B 스플라인만 해당):

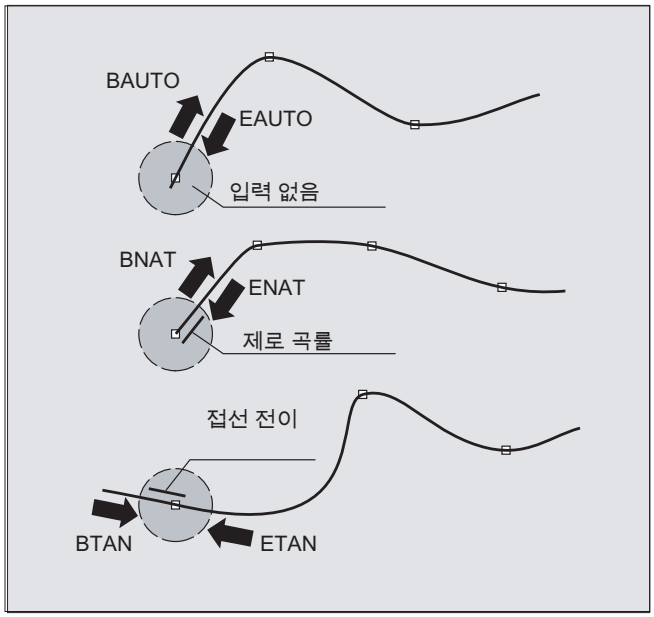
PL	노드 사이 거리는 내부적으로 적절하게 계산됩니다 . 또한 제어 시스템은 PL 명령을 사용하여 소위 파라미터-간격-길이에서 지정된 미리 정의된 노드 간극을 가공할 수 있습니다 .	
<value>	파라미터 간격 길이	
	값의 범위 :	경로 치수

스플라인 곡선 시작 위치에서 진행 동작 (A 또는 C 스플라인만 해당):

BAUTO	진행 동작을 지정하지 않습니다 . 시작은 첫 번째 점의 위치에 의해 결정됩니다 .
BNAT	제로 곡률
BTAN	이전 블록으로 접선 방향 진행 (삭제 위치)

스플라인 곡선 끝 위치에서 진행 동작 (A 또는 C 스플라인만 해당):

EAUTO	진행 동작을 지정하지 않습니다 . 끝은 마지막 점의 위치에 의해 결정됩니다 .
ENAT	제로 곡률
ETAN	이전 블록으로 접선 방향 진행 (삭제 위치)



주

프로그램 가능한 진행 동작은 B 스플라인에는 영향을 주지 않습니다 . B 스플라인은 항상 시작점 및 끝점에서 확인 다각형에 대해 접선 방향입니다 .

보충 조건

- 공구 반경 보정을 사용할 수 있습니다 .
- 평면 돌출부에서 충돌 모니터링이 실행됩니다 .

예

예제 1: B 스플라인

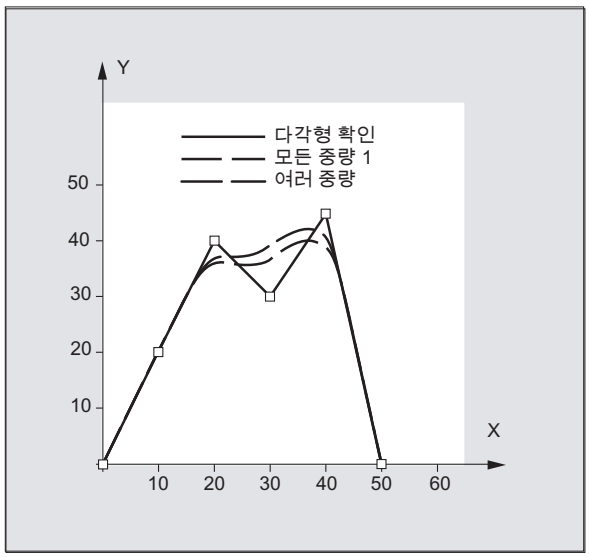
프로그램 코드 1(모두 가중치 1)

```
N10 G1 X0 Y0 F300 G64
N20 BSPLINE
N30 X10 Y20
N40 X20 Y40
N50 X30 Y30
N60 X40 Y45
N70 X50 Y0
```

프로그램 코드 2(가중치가 서로 다름)

```
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
```

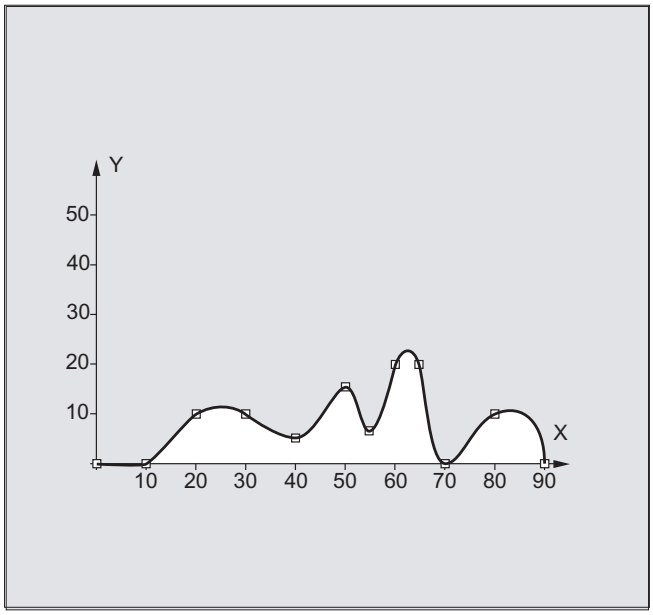
프로그램 코드 3(확인 다각형)	비고
N10 G1 X0 Y0 F300 G64	
N20	; 해당 없음
N30 X10 Y20	
N40 X20 Y40	
N50 X30 Y30	
N60 X40 Y45	
N70 X50 Y0	



예제 2: C 스플라인 , 시작 및 끝 위치에서 제로 곡률

프로그램 코드

```
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
```



예제 3: 스플라인 보간 (A 스플라인) 및 좌표 변형 (ROT)

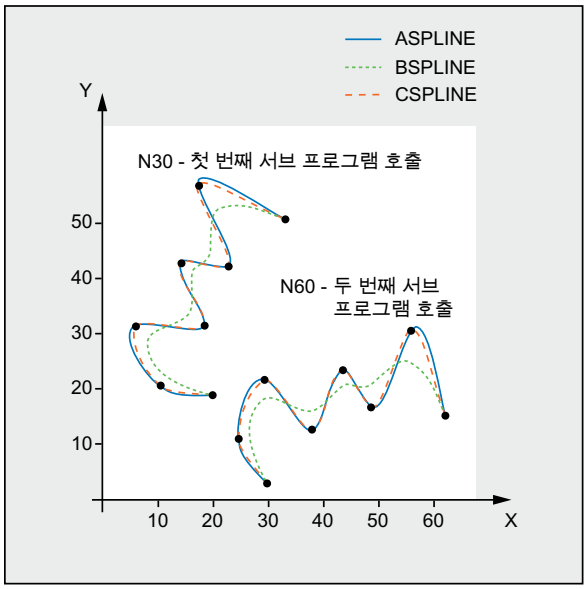
메인 프로그램 :

프로그램 코드	비고
N10 G00 X20 Y18 F300 G64	; 시작점으로 복귀
N20 ASPLINE	; 보간 유형 A 스플라인을 활성화합니다 .
N30 CONTOUR	; 첫 번째 서브프로그램 호출 .
N40 ROT Z-45	; 좌표 변환 : Z 축 주위를 -45° 각도로 WCS 회전 .
N50 G00 X20 Y18	; 형상 시작점으로 복귀 .
N60 CONTOUR	; 두 번째 서브프로그램 호출 .
N70 M30	; 프로그램 끝

서브프로그램 " 형상 "(곡선을 따라 점에 대한 좌표 포함):

프로그램 코드
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

스플라인 곡선 이외에 예제 프로그램 (ASPLINE) 을 실행하면 , 다음 그림과 같이 B 또는 C 스플라인 보간 (BSPLINE, CSPLINE) 을 활성화할 때 가져온 스플라인 곡선도 포함됩니다 .



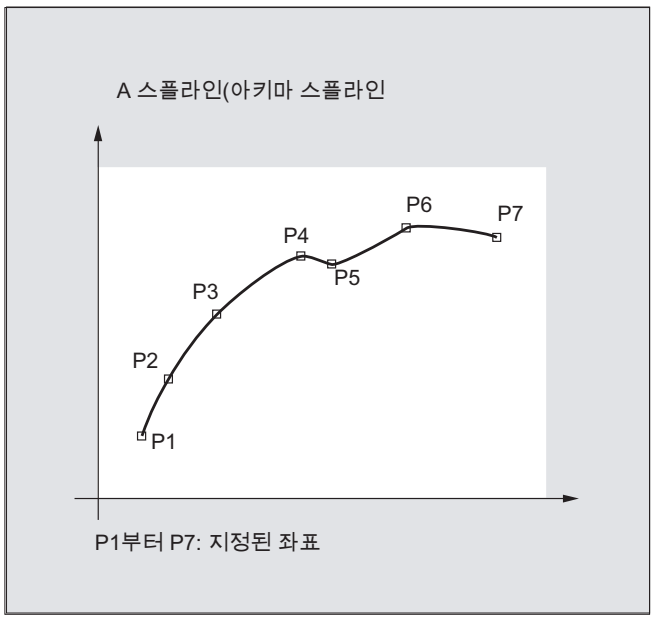
그 밖의 정보

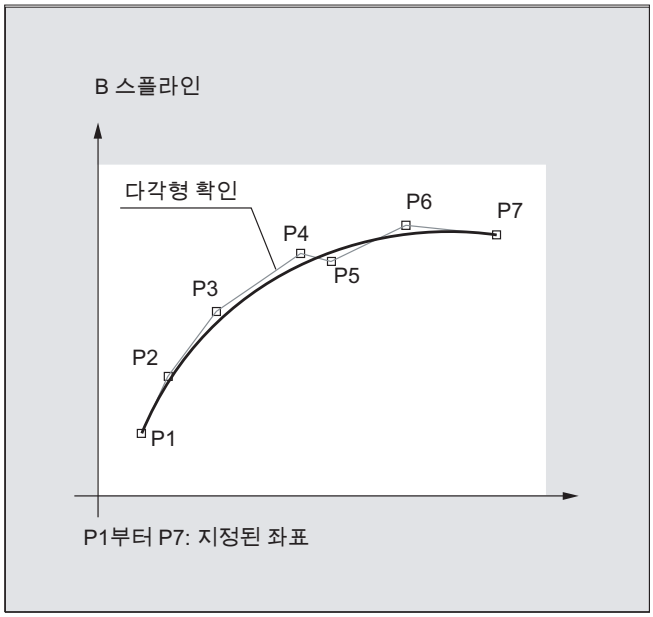
스플라인 보간의 이점

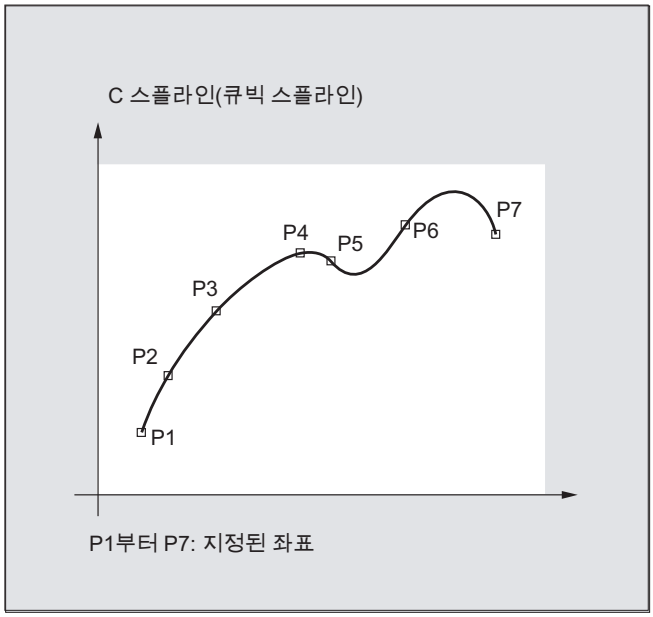
스플라인 보간을 사용하면 직선 블록 G01 을 사용하는 것과 반대로 다음과 같은 이점을 얻을 수 있습니다 .

- 형상을 설명하는 데 필요한 가공 프로그램 블록 수 감소 .
- 가공 프로그램 블록 간의 이동 시 기계 시스템의 응력을 줄이는 부드러운 곡선 특성 .

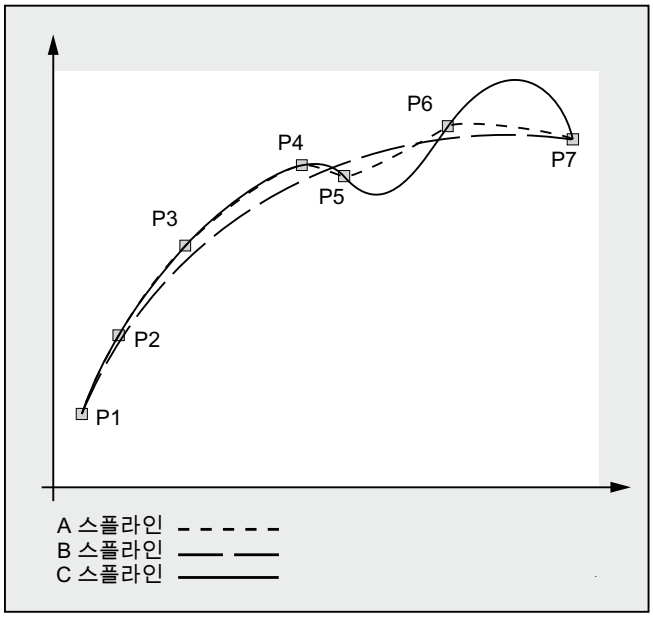
속성 및 다양한 스플라인 유형 사용

스플라인 유형	속성 및 사용
A 스플라인	 **속성 :** • 곡선을 따라 지정된 중간 지점들을 정확하게 통과합니다 . • 곡선 특성은 접선이지만 연속 곡률이 없습니다 . • 원치 않는 오실레이션이 거의 발생하지 않습니다 . • 곡선을 따라 중간 지점에 대한 변경에 영향을 받는 영역은 제한적입니다 . 즉 , 곡선을 따라 중간 지점에 대한 변경은 최대 6 개의 인접한 중간 지점에만 영향을 줍니다 . **용도 :** A 스플라인은 특히 기울기가 상당히 변경되는 보간 곡선에 적합합니다(예를 들어 계단식 곡선 및 특성).

스플라인 유형	속성 및 사용
B 스플라인	 **특징 :** • 곡선을 따라 지정된 중간 지점을 통해 진행하지 말고 근접한 위치에서 진행하십시오. 중간 지점은 곡선을 끌어 당기지 않습니다. 곡선 특성은 인자를 사용하여 중간점에 가중치를 부과하는 방법으로 추가로 영향을 줄 수 있습니다 . • 곡선 특성은 연속 곡률을 통해 접선 방향입니다 . • 원하지 않는 오실레이션을 생성하지 않습니다 . • 곡선을 따라 중간점에 대한 변경에 영향을 받는 영역은 제한적입니다 . 즉 , 곡선을 따라 중간점에 대한 변경은 최대 6 개의 인접한 중간점에만 적용됩니다 . **용도 :** B 스플라인은 주로 CAD 시스템에 대한 인터페이스로 사용됩니다 .

스플라인 유형	속성 및 사용
C 스플라인	 **특징 :** • 곡선을 따라 지정된 중간점들을 정확하게 통과합니다 . • 곡선 특성은 연속 곡률을 통해 접선 방향입니다 . • 특히 기울기가 상당히 변경되는 위치에서 원하지 않는 오실레이션이 자주 생성됩니다 . • 중간점에 대한 변경에 대해 전체 영역이 영향을 받습니다 . 즉 , 중간점이 변경되면 전체 곡선 특성에 영향을 줍니다 . **용도 :** 중간점이 분석적으로 계산되어 정의할 수 있는 곡선(원, 포물선, 쌍곡선)에 위치한 경우 C 스플라인을 사용할 수 있습니다 .

동일한 보간 지점을 가진 세 가지 스플라인 유형의 비교



최소 스플라인 블록 수

G 코드 ASPLINE, BSPLINE 및 CSPLINE 은 블록 끝점을 스플라인과 연결합니다 . 이를 위해 일련의 블록 (끝점) 이 동시에 계산되어야 합니다 . 계산을 위한 버퍼 크기의 기본값은 10 개 블록입니다 . 스플라인 끝점은 블록 정보의 일부일 뿐입니다 . 그러나 제어기에서 10 개 블록 모두에 대해 특정 수의 스플라인 끝점 블록이 필요합니다 .

스플라인 유형	최소 스플라인 블록 수
A 스플라인 :	10 개 블록 중 최소한 **4** 개 블록이 스플라인 블록이어야 합니다 . 여기서 설명 블록 또는 파라미터 계산은 제외됩니다 .
B 스플라인 :	10 개 블록 중 최소한 **6** 개 블록이 스플라인 블록이어야 합니다 . 여기서 설명 블록 또는 파라미터 계산은 제외됩니다 .
C 스플라인 :	필요한 최소 스플라인 블록 수는 다음 합계와 같습니다 . MD20160 $MC_CUBIC_SPLINE_BLOCKS + 1 의 값 스플라인 세그먼트를 계산하는 데 사용되는 점 수가 MD20160 에 입력됩니다 . 기본 설정값은 8 입니다 . 표준으로 10 개 블록 중 최소 **9** 개 블록이 스플라인 블록이어야 합니다 .

주

허용 오차 값이 더 작고 스플라인의 관련 축 중 하나가 위치 지정 축으로 프로그래밍된 경우 알람이 출력됩니다 .

짧은 스플라인 블록 조합

스플라인 보간을 통해 짧은 스플라인 블록을 만들 수 있으며 이로 인해 불필요한 경로 속도를 줄입니다. "짧은 스플라인 블록 조합" 기능을 통해 결과 블록 길이가 충분하도록 이러한 블록을 조합할 수 있으며 경로 속도가 줄어 들지 않습니다 .

이 기능은 채널 관련 기계 데이터를 통해 활성화됩니다 .

MD20488 $MC_SPLINE_MODE(스플라인 보간 설정).

참조 문서 :
작동 설명서 , 기본 기능 ; 연속 경로 모드 , 정위치 정지 , 록 어헤드 (B1),
장 : 짧은 스플라인 블록 조합

4.3 스플라인 그룹화 (SPLINEPATH)

작동

스플라인 그룹에서 보간할 축은 SPLINEPATH 명령을 사용하여 선택합니다 . 스플라인 보간 그룹화에 최대 8 개의 축을 사용할 수 있습니다 .

주

SPLINEPATH 가 명시적으로 프로그래밍되지 않을 경우 채널의 세 개 축 중 첫 번째 축이 스플라인 그룹으로 이송됩니다 .

구문

스플라인 그룹이 개별 블록에 정의됩니다 :

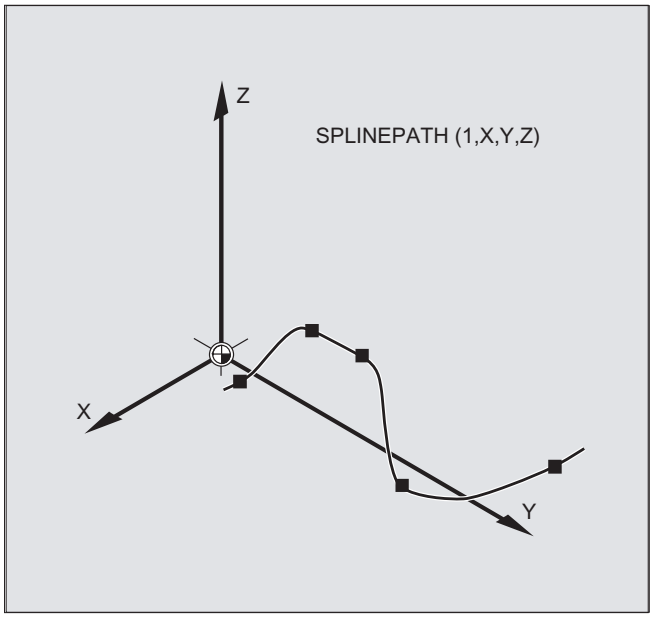
SPLINEPATH(n,X,Y,Z,…)

의미

SPLINEPATH	스플라인 그룹을 정의하기 위한 명령
n	=1 (고정값)
X,Y,Z,…	스플라인 그룹에서 보간할 경로 축의 식별자

예 : 세 개의 경로 축을 가진 스플라인 그룹

프로그램 코드	비고
N10 G1 X10 Y20 Z30 A40 B50 F350	
N11 SPLINEPATH(1,X,Y,Z)	; 스플라인 그룹
N13 CSPLINE BAUTO EAUTO X20 Y30 Z40 A50 B60	; C 스플라인
N14 X30 Y40 Z50 A60 B70	; 중간점
…	
N100 G1 X… Y…	; 스플라인 보간 선택 취소



4.4 NC 블록 압축 (COMPON, COMPCURV, COMPCAD, COMPOF)

기능

CAD/CAM 시스템은 일반적으로 지정된 공차에 따르는 직선 블록을 제공합니다 . 복잡한 형상의 경우 데이터 및 짧은 경로 부분이 많아질 수 있습니다 . 짧은 경로 부분은 처리 속도를 제한합니다 .

컴프레서 기능을 통해 직선 블록을 사용하여 지정된 형상이 폴리노미얼 블록으로 처리됩니다. 이것은 다음과 같은 장점이 있습니다 :

- 공작물 형상을 설정하기 위해 필요한 가공 프로그램 블록 수 감소
- 연속 블록 트렌지션
- 최대 경로 속도 증가

다음 컴프레서 기능을 사용할 수 있습니다 .

- COMPON

  블록 트렌지션의 **속도**는 일정하지만 관련 축의 가속도는 블록 트렌지션 시 급격하게 상승할 수 있습니다 .

- COMPCURV

  블록 트렌지션이 **연속 가속**을 가집니다. 이로 인해 블록 트렌지션 시 모든 축의 원활한 속도 및 가속이 보장됩니다 .

- COMPCAD

  표면 품질 및 속도와 관련하여 복잡한 계산 시간과 메모리 공간을 최적화하여 사용합니다 . COMPCAD 는 CAD/CAM 프로그램에서 미리 표면을 향상시키기 위한 조치가 이루어질 경우에만 사용할 수 있습니다 .

COMPOF 는 컴프레서 기능을 종료합니다 .

구문

COMPON
COMPCURV
COMPCAD
COMPOF

의미

COMPON:	컴프레서 기능 COMPON 을 활성화하기 위한 명령 적용 : 모달
COMPCURV:	컴프레서 기능 COMPCURV 를 활성화하기 위한 명령 . 적용 : 모달
COMPCAD:	컴프레서 기능 COMPCAD 를 활성화하기 위한 명령 . 적용 : 모달
COMPOF :	현재 활성 상태인 컴프레서 기능 비활성화하기 위한 명령

주

라운딩 기능 G642 및 저크 제한 SOFT를 사용하여 표면 품질을 추가로 향상시킬 수 있습니다. 이러한 명령은 프로그램 시작 시 기록해야 합니다 .

일반 조건

- NC 블록 압축은 일반적으로 직선 블록 (G1) 에만 실행할 수 있습니다 .
- 단순 구문을 준수하는 블록만 압축됩니다 :

  N... G1X... Y... Z... F... ; 설명

  다른 모든 블록은 변경되지 않은 상태로 실행됩니다 (압축 없음).
- 또한 C=100 또는 A=AC(100) 와 같은 확장된 어드레스를 가진 이동 블록도 압축됩니다 .
- 위치 값을 직접 프로그래밍할 필요는 없지만 예를 들어 X=R1*(R2+R3) 과 같이 파라미터 지정을 사용하여 간접적으로 지정할 수도 있습니다 .
- 옵션 "오리엔테이션 변환"을 사용할 수 있는 경우 방향 벡터를 사용하여 프로그래밍된 공구 오리엔테이션 (공구 회전 포함) 의 NC 블록도 압축할 수 있습니다 (" 방향 압축 (COMPON, COMPCURV, COMPCAD) [쪽 358]" 참조).
- 예를 들어 보조 기능 출력과 같이 다른 유형의 NC 명령에 의해 중단됩니다 .

예제

예제 1: COMPON

프로그램 코드	설명
N10 COMPON	; 컴프레서 기능 COMPON 을 실행합니다 .
N11 G1 X0.37 Y2.9 F600	; 종점 이전의 G1 및 피드 .
N12 X16.87 Y–.698	
N13 X16.865 Y–.72	
N14 X16.91 Y–.799	
…	
N1037 COMPOF	; 컴프레서 기능 OFF
…	

예제 2: COMPCAD

프로그램 코드	설명
G00 X30 Y6 Z40	
G1 F10000 G642	; 라운딩 기능 G642 를 실행합니다 .
SOFT	; 저크 제한 SOFT 를 실행합니다 .
COMPCAD	; 컴프레서 기능 COMPCAD 를 실행합니다 .
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	; 컴프레서 기능 OFF
G00 Z50	
M30	

참고 자료

기능 매뉴얼 , 기본 기능 ; 연속 경로 모드 , 정위치 정지 , 선행 제어 (B1),
장 : "NC 블록 압축 "

4.5 폴리노미널 보간 (POLY, POLYPATH, PO, PL)

기능

실제로 스플라인 보간이 아닌 폴리노미얼 보간 (PLLY) 이 사용됩니다 . 폴리노미얼 보간의 주요 목적은 스플라인 섹션을 직접 프로그래밍할 수 있도록 외부에서 생성된 스플라인 곡선을 프로그래밍하기 위한 인터페이스 역할을 수행하는 것입니다 .

이 보간 모드는 폴리노미얼 계수를 계산하는 NC 작업의 부담을 덜어 줍니다 . 계수가 CAD 시스템 또는 다음 프로세서에 의해 직접 제공되는 경우 최적화될 수 있습니다 .

구문

3 차 다항식 :
POLY PO[X]=(xe,a2,a3) PO[Y]=(ye,b2,b3) PO[Z]=(ze,c2,c3) PL=n

5 차 다항식 및 새로운 폴리노미얼 구문 :
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")

주

NC 블록에 프로그래밍된 축과 폴리노미얼 계수의 합은 블록당 허용된 최대 축 수를 초과하지 않아야 합니다 .

의미

POLY :	POLY가 포함된 블록을 통한 폴리노미얼 보간 활성화 .
POLYPATH :	AXIS 축 그룹 그리고 VECT 축 그룹 모두에 폴리노미얼 보간 선택 가능
PO[축 이름 / 변수] :	끝점 및 폴리노미얼 계수
X, Y, Z :	축 이름
xe, ye, ze :	특정 축의 끝 위치 지정 ; 경로 차원에 대한 값 범위
a2, a3, a4, a5 :	계수 a_2, a_3, a_4 및 a_5 는 경로 차원에 대한 값 범위에서 해당 값이 기록됩니다 . 각 경우에서 마지막 계수가 0 인 경우 생략될 수 있습니다 .
PL :	폴리노미얼이 정의된 파라미터 간격 길이 (함수 f(p) 의 정의 범위). 간격은 항상 0부터 시작하며 p는 0-PL 사이의 값으로 간주할 수 있습니다 . PL 의 이론 값 범위 : 0,0001 … 99 999,9999 **참고 :** PL 값은 위치한 블록에 적용됩니다 . PL 을 프로그래밍하지 않을 경우 PL=1 이 적용됩니다 .

폴리노미얼 보간 활성화 / 비활성화

POLX G 명령을 사용하여 가공 프로그램에서 폴리노미얼 보간을 활성화합니다 .

POLY G 명령은 G0, G1, G2, G3, ASPLINE, BSPLINE 및 CSPLINE 과 함께 그룹 1 에 속합니다 .

이름과 종점만 프로그래밍된 축 (예 : X10) 은 직선으로 움직입니다 . NC 블록의 모든 축이 이렇게 프로그래밍되면 제어 시스템은 G1 의 경우와 동일하게 동작합니다 .

첫 번째 G 그룹 (G0, G1) 에 다른 명령을 프로그래밍하면 폴리노미얼 보간이 암시적으로 다시 비활성화됩니다 .

폴리노미얼 계수

PO 값 (PO[]=) 또는 ...=PO(...) 는 축의 모든 폴리노미얼 계수를 지정합니다 . 값이 여러 개인 경우 다항식 차수에 해당하는 쉼표로 구분되어 지정됩니다 . 한 블록 내에서 여러 축에 대해 서로 다른 다항식 차수가 가능합니다 .

POLYPATH 서브프로그램

POLYPATH(...) 를 사용하면 특정 축 그룹에 대한 폴리노미얼 보간을 선택적으로 해제할 수 있습니다 .

경로 축 및 보조 축만 해제하는 경우 :	POLYPATH("AXES")
오리엔테이션 축만 해제하는 경우 : (오리엔테이션 변환을 적용하여 이동할 때)	POLYPATH("VECT")

해제되지 않은 축은 직선으로 움직입니다 .

폴리노미얼 보간은 기본적으로 축 그룹 모두에 활성화되어 있습니다 .

POLYPATH( ) 파라미터없이 프로그래밍하면 모든 축에 대한 폴리노미얼 보간을 비활성화할 수 있습니다 .

예제

프로그램 코드	설명
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	; 폴리노미얼 보간 실행
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	; 축의 데이터 혼합
N27 PO[X]=(10,2.5) PO[Y]=(2.3)	; PL 프로그램 없음 ; PL=1 적용
N30 G1 X… Y… Z.	; 폴리노미얼 보간 비활성화
…	

예 : 새로운 폴리노미얼 구문

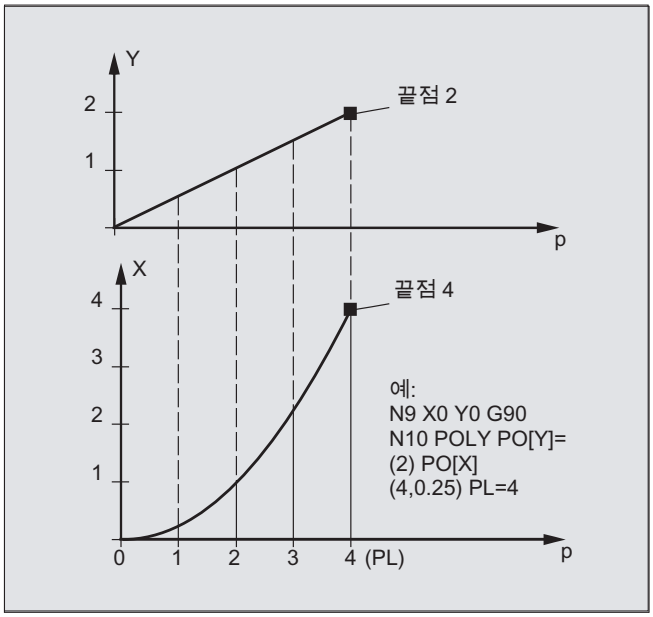
아직 유효한 폴리노미얼 구문	새로운 폴리노미얼 구문
PO[axis identifier]=(.. , ..)	Axis identifier=PO(.. , ..)
PO[PHI]=(.. , ..)	PHI=PO(.. , ..)
PO[PSI]=(.. , ..)	PSI=PO(.. , ..)
PO[THT]=(.. , ..)	THT=PO(.. , ..)
PO[]=(.. , ..)	PO(.. , ..)
PO[variable]=IC(.. , ..)	variable=PO IC(.. , ..)

예 : X/Y 평면의 곡선 .

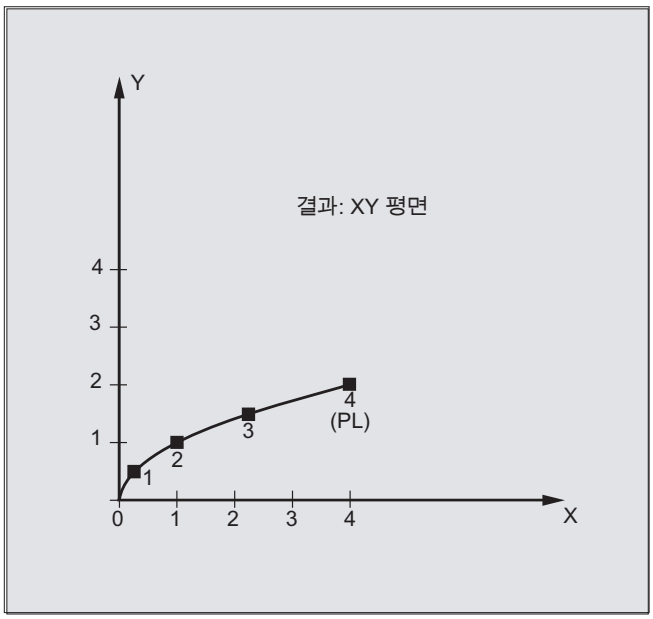
프로그래밍

프로그램 코드
N9 X0 Y0 G90 F100
N10 POLY PO[Y]=(2) PO[X]=(4,0.25) PL=4

곡선 X(p) 및 Y(p) 의 형태



XY 평면의 곡선 형태



설명

다항식 함수를 표현하는 데 사용되는 방정식은 일반적으로 다음과 같습니다 .

$f(p)= a_0 + a_1p + a_2p^2 +. . . + a_np^n$

여기서 a_n: 상수 계수
p: 파라미터

제어 시스템에 최대 5 차 다항식까지 프로그래밍할 수 있습니다 .

$f(p)= a_0 + a_1p + a_2p^2 + a_3p^3 + a_4p^4 + a_5p^5$

특정 값을 이러한 계수에 할당하면 선, 포물선 및 제곱 함수와 같은 다양한 곡선 형태를 만들 수 있습니다 .

직선을 만들려면 $a_2 = a_3 = a_4 = a_5 = 0$ 를 사용합니다 .

$f(p) = a_0 + a_1p$

다음 내용이 그대로 적용됩니다 .

$a_{0:}$ 선행 블록의 끝에서 축 위치

p = PL

$a_1 = (x_E - a_0 - a_2*p^2 - a_3*p^3)/p$

G 명령 POLY 를 사용하면 폴리노미얼 보간을 활성화하지 **않고** 폴리노미얼을 프로그램할 수 있습니다 . 이 경우 프로그래밍된 폴리노미얼은 보간되지 않는 대신 축에 프로그래밍된 종점은 직선으로 복귀합니다 (G1). 프로그래밍된 폴리노미얼은 가공 프로그램 (POLY) 에서 폴리노미얼 보간을 명시적으로 활성화한 후에만 이동합니다 .

특징 : 분모 다항식

명령 PO[]=(…) 를 사용하여 축 이름을 지정하지 않고 기하 축의 공통 분모 다항식을 프로그래밍할 수 있습니다 . 예를 들어 형상 축은 두 다항식의 몫으로 보간됩니다 .

이 프로그래밍 옵션을 통해 원뿔 곡선론 (원 , 타원 , 포물선 , 쌍곡선) 과 같은 형태를 정확하게 표현할 수 있습니다 .

예제 :

프로그램 코드	설명
POLY G90 X10 Y0 F100	; 형상 축이 X10 Y0 위치로 직선으로 이송됩니다 .
PO[X]=(0,–10) PO[Y]=(10) PO[]=(2,1)	; 형상 축이 4 분원을 따라 X0 Y10 으로 이송됩니다 .

분모 다항식의 상수 계수 (a_0) 는 항상 1 로 간주되며 프로그래밍된 종점은 G90/G91 에 종속되지 않습니다 .

다음과 같이 프로그래밍된 값에서 X(p) 및 Y(p) 를 계산합니다 .

$X(p) = (10 - 10 * p^2) / (1 + p^2)$

$Y(p) = 20 * p / (1 + p^2)$

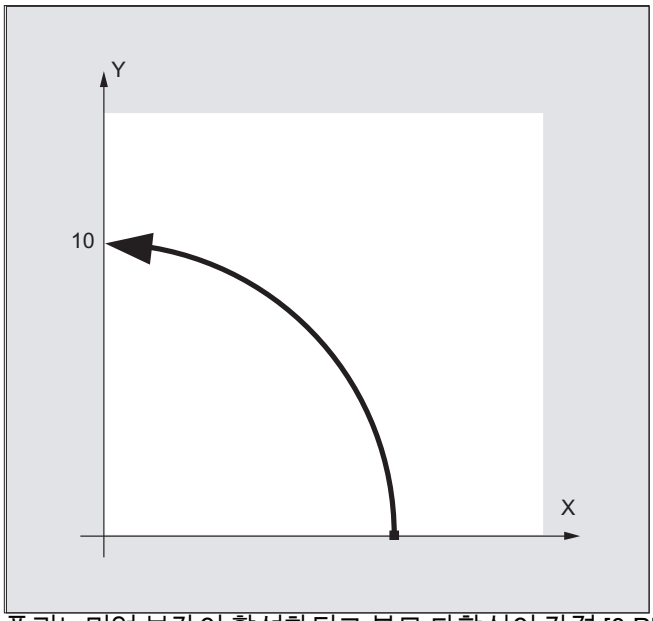
$0 \leq p \leq 1$

프로그래밍된 시작점 , 종점 , 계수 a_2 및 PL=1 을 적용한 중간 값은 다음과 같습니다 .

분자 (X) = $10 + 0 * p - 10 * p^2$

분자 (Y) = $0 + 20 * p + 0 * p^2$

분모 = $1 + p^2$



폴리노미얼 보간이 활성화되고 분모 다항식이 간격 [0,PL] 내에서 0으로 프로그래밍된 경우 이 조건은 거부되고 알람이 출력됩니다 . 분모 다항식은 특수 축의 이동에 영향을 주지 않습니다 .

주

공구 반경 보정은 폴리노미얼 보간과 함께 G41, G42 를 통해 활성화할 수 있으며 선형 또는 원호 보간 모드와 동일한 방법으로 적용할 수 있습니다 .

4.6 설정 가능한 경로 기준 (SPATH, UPATH)

작동

폴리노미얼 보간이 작동하는 동안 사용자는 속도 결정 FGROUP 축과 다른 경로 축의 관계를 두 가지로 다르게 지정해야 합니다. 즉, 경로 축을 경로 S와 동기화하거나 FGROUP 축의 곡선 파라미터 U 와 동기화해야 합니다 .

경로 보간 방식 두 개는 적용하는 곳이 다르고 , G 코드 그룹 45 에 포함된 두 모달 언어 명령 SPATH 및 UPATH 를 사용하여 설정 / 전환할 수 있습니다 .

구문

SPATH
UPATH

의미

SPATH:	FGROUP 축의 경로 기준은 호 길이입니다 .
UPATH:	FGROUP 축의 경로 기준은 곡선 파라미터입니다 .

주

또한 UPATH 및 SPATH 는 경로 이동을 통해 F 워드 폴리노미얼의 상호 관계를 정의합니다 (FPOLY, FCUB, FLIN).

일반 조건

경로 기준 설정은 다음 사항에 영향을 주지 않습니다 .

- 직선 보간 및 원호 보간
- 나사 블록
- 모든 경로 축이 FGROUP 에 포함된 경우

예제

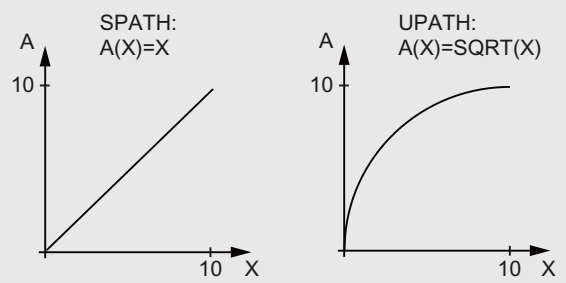
예제 1:

다음 예에서는 측면 길이가 20 mm 이고 G643 으로 모서리가 라운딩을 사각형을 보여 줍니다 . 축 머신 데이터 MD33100 $MA_COMPRESS_POS_TOL[<n>] 을 사용하여 각 축에 정밀 형상의 최대 편차를 정의합니다 .

프로그램 코드	설명
N10 G1 X… Y… Z… F500	
N20 G643	; G643 을 사용하여 블록 내부 코너 라운딩
N30 XO Y0	
N40 X20 Y0	; 축의 모시리 길이 (mm)
N50 X20 Y20	
N60 X0 Y20	
N70 X0 Y0	
N100 M30	

예제 2:

다음 예에서는 두 가지 이동 제어 유형 간의 차이점을 보여 줍니다 . 디폴트 설정 FGROUP(X,Y,Z) 이 활성화됩니다 .



SPATH와 UPATH에 대해 각각 다른 축 간 기하 관계 적용

프로그램 코드
N10 G1 X0 A0 F1000 SPATH
N20 POLY PO[X]=(10,10) A10

or:

프로그램 코드
N10 G1 X0 F1000 UPATH
N20 POLY PO[X]=(10,10) A10

블록 N20 에서 FGROUP 축의 경로 S 는 곡선 파라미터 U 의 제곱에 따라 달라집니다 . 따라서 SPATH 또는 UPATH 활성화 여부에 따라 경로 X 에 있는 동기 축 A 의 위치가 달라집니다 .

추가 정보

폴리노미얼 보간 중 모든 경로 축 i 의 위치는 폴리노미얼 파이 (U) 에 지정됩니다 폴리노미얼 보간에는 항상 엄격한 의미의 폴리노미얼 보간 (POLY), 모든 스플라인 보간 유형 (ASPLINE, BSPLINE, CSPLINE) 및 압축 함수 (COMPON, COMPCURV) 가 포함된 모든 직선 보간이 포함됩니다 . 곡선 파라미터 U 는 NC 블록 내에서 0 에서 1 로 이동하기 때문에 표준화됩니다 .

프로그래밍된 경로 피드와 관련된 축은 언어 명령 FGROUP 를 통해 경로 축에서 선택할 수 있습니다 . 그러나 폴리노미얼 보간 동안 이러한 축의 경로에서 일정한 속도를 가진 보간은 일반적으로 곡선 파라미터 U 의 일정하지 않은 변경을 의미합니다 .

리셋 및 기계 / 옵션 데이터에 대한 제어 동작

MD20150 $MC_GCODE_RESET_VALUES[44] (G 코드 그룹 45) 에 의해 정의된 G 코드는 리셋한 후에 적용됩니다 . 기존 설치와의 호환성을 유지하기 위해 SPATH 를 디폴트 값으로 설정합니다 .

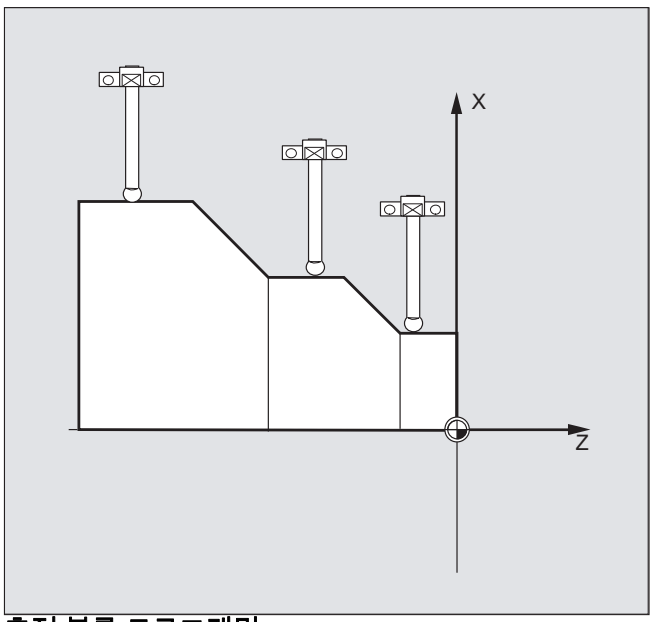
스무딩 유형의 초기 상태는 MD20150 $MC_GCODE_RESET_VALUES[9] (G 코드 그룹 10) 에 의해 정의됩니다 .

축 머신 데이터 MD33100 $MA_COMPRESS_POS_TOL[<n>] 은 확장된 의미를 지니고 있습니다 . 이 데이터는 컴프레서 기능 및 G642 을 사용한 스무딩에 대한 공차를 포함합니다 .

4.7 터치 프로브를 통한 측정 (MEAS, MEAW)

기능

' 터치 - 트리거 프로브를 사용한 측정 ' 은 공작물 위의 실제 위치에 접근하는 데 사용됩니다 . 프로브의 전환 에지에서는 측정 블록에 프로그래밍된 모든 축의 위치가 측정되어 각 축의 해당 메모리 셀에 기록됩니다 .



측정 블록 프로그래밍

측정 블록을 프로그래밍할 때 다음 두 가지 명령을 사용할 수 있습니다 .

- MEAS

  MEAS 명령은 실제 위치와 지령치 위치 사이의 이동할 거리를 삭제하는 데 사용됩니다 .

- MEAW

  MEAW 명령은 프로그래밍된 위치로 항상 접근해야 하는 측정 작업에 사용됩니다 .

MEAS 및 MEAW 는 넌모달 명령으로서 , 모션 작업과 함께 프로그래밍됩니다 . 축의 개수를 비롯하여 이송 속도 및 보간 유형 (G0, G1 등) 을 측정할 작업에 맞게 선택해야 합니다 .

측정 결과 읽기

프로브를 통해 가져온 축에 대한 측정 결과를 다음의 변수에서 사용할 수 있습니다 .

- $AA_MM[< 축 >]

  기계 좌표계에서의 측정 결과

- $AA_MW[< 축 >]

  공작물 좌표계에서의 측정 결과

이러한 변수를 읽을 때 내부 전처리 정지는 생성되지 않습니다 .

주

전처리 정지는 프로그램의 해당 위치에서 STOPRE를 통해 프로그래밍되어야 합니다. 그렇지 않을 경우 시스템이 잘못된 값을 읽습니다 .

구문

MEAS=<TE> G... X... Y... Z...
MEAW=<TE> G... X... Y... Z...

의미

MEAS	명령 : 이동할 거리를 **삭제하고** 측정 적용 : 넌모달
MEAW	명령 : 이동할 거리를 **삭제하지 않고** 측정 적용 : 넌모달
<TE>	측정을 시작하는 트리거 이벤트 유형 : INT 범위 : -2, -1, 1, 2 **참고 :** 구성 레벨에 따라 최대 2 개의 프로브가 있습니다 . 의미 : (+)1 프로브 1 의 상승 에지 (측정 입력 1) -1 프로브 1 의 하강 에지 (측정 입력 1) (+)2 프로브 2 의 상승 에지 (측정 입력 2) -2 프로브 2 의 하강 에지 (측정 입력 2) **참고 :** 구성 레벨에 따라 최대 2 개의 프로브가 있습니다 .
G...	보간 유형 (예 : G0, G1, G2 또는 G3)
X... Y... Z...	직교 좌표의 종점

예제

프로그램 코드	설명
N10 MEAS=1 G1 F1000 X100 Y730 Z40	; 첫 번째 측정 입력 및 직선 보간에서 프로브를 통한 측정 블록 . 전처리 정지가 자동으로 생성됩니다 .
...	

추가 정보

작업 상태 측정

프로브가 시작되었는지 여부와 관련하여 프로그램에서 평가가 필요한 경우 상태 변수 $AC_MEA[n] (n 은 측정 프로브의 번호) 를 조회할 수 있습니다 .

값	의미
0	측정 작업이 완료되지 않음 .
1	측정 작업이 성공적으로 완료됨 (프로브가 시작됨).

주

프로그램에서 오류가 발생하면 변수가 1 로 설정됩니다 . 이 변수는 측정 블록의 시작에서 자동으로 프로브의 초기 상태로 설정됩니다 .

측정값 읽기

블록의 모든 이송 경로 및 포지셔닝 축에 대한 위치가 판독됩니다 (제어 구성에 따라 최대 축 수). MEAS 의 경우에는 프로브가 시작되면 정의된 방식에 따라 모션이 감속됩니다 .

주

GEO 축이 측정 블록에 프로그래밍될 경우 모든 현재 GEO 축에 대해 측정값이 저장됩니다 .

변환에 참여하는 축이 측정 블록에 프로그래밍될 경우 이 변환에 참여하는 모든 축에 대한 측정값이 기록됩니다 .

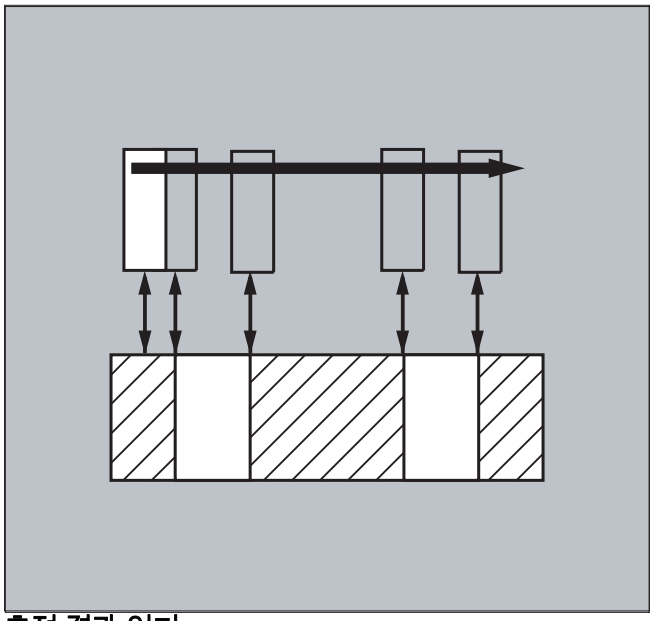
4.8 확장된 측정 기능 (MEASA, MEAWA, MEAC) (옵션)

기능

축 측정을 위해 여러 개의 프로브 및 여러 개의 측정 시스템을 사용할 수 있습니다 .

MEASA 또는 MEAWA 명령은 프로그래밍된 각각의 축에 대해 최대 4개의 측정값을 구하는 데 사용할 수 있습니다 . 그런 다음 이 값들은 트리거 이벤트에 따라 시스템 변수에 저장됩니다 .

MEAC 명령을 사용하여 연속 측정 작업을 실행할 수 있습니다 . 이 경우 측정 결과는 FIFO 변수에 저장됩니다 . 측정당 최대 측정값 수는 MEAC 의 경우도 4 개입니다 .



측정 결과 읽기

측정 결과를 다음의 변수에서 사용할 수 있습니다 .

- $AA_MM1...4[< 축 >]

  기계 좌표계에서의 측정 결과

- $AA_MW1...4[< 축 >]

  공작물 좌표계에서의 측정 결과

구문

MEASA[< 축 >]=(< 모드 >,<TE1>,...,<TE4>)
MEAWA[< 축 >]=(< 모드 >,<TE1>,...,<TE4>)
MEAC[< 축 >]=(< 모드 >,< 측정 메모리 >,<TE1>,...,<TE4>)

주

MEASA 및 MEAWA 는 넌모달이므로 한 블록에 함께 프로그래밍할 수 있습니다 . 그러나 MEASA/MEAWA 를 동일한 블록에 MEAS/MEAW 와 함께 프로그래밍하면 에러 메시지가 출력됩니다 .

의미

MEASA	명령 : 이동할 거리를 삭제하고 축 측정 적용 : 넌모달
MEAWA	명령 : 이동할 거리를 삭제하지 않고 축 측정 적용 : 넌모달
MEAC	명령 : 이동할 거리를 삭제하지 않고 연속 축 측정 적용 : 넌모달
<axis>	측정에 사용된 채널 축 이름
<mode>	운전 모드를 나타내는 두 자리 숫자 (측정 모드 및 측정 시스템) **측정 모드** (10 단위): 0 측정 작업을 취소합니다 . 1 최대 4 개의 서로 다른 트리거 이벤트를 동시에 실행할 수 있습니다 . 2 최대 4 개의 서로 다른 트리거 이벤트를 연속적으로 실행할 수 있습니다 . 3 최대 4 개의 트리거 이벤트를 연속적으로 실행할 수 있지만 START 에서 트리거 이벤트 1 을 모니터링하지 않습니다 (알람 21700/21703 은 억제됨). **참고 :** MEAC 에서는 이 모드가 지원되지 않습니다 . **측정 시스템** (10 단위): 0 (또는 데이터 없음) 활성 측정 시스템 1 측정 시스템 1 2 측정 시스템 2 3 두 측정 시스템
<TE>	측정을 시작하는 트리거 이벤트 유형 : INT 범위 : -2, -1, 1, 2 의미 : (+)1 프로브 1 의 상승 에지 -1 프로브 1 의 하강 에지 (+)2 프로브 2 의 상승 에지 -2 프로브 2 의 하강 에지
< 측정 메모리 >	FIFO 수 (저장소 순환)

예제

예제 1: 모드 1 에서 이동할 거리를 삭제하고 축 측정 (시간 순서대로 평가)

a) 1 개 측정 시스템 사용

프로그램 코드	설명
...	
N100 MEASA[X]=(1,1,-1) G01 X100 F100	; 활성 측정 시스템으로 모드 1 에서 측정 . 이송 경로의 프로브 1 에서 X=100 까지의 상승 / 하강 에지에서 측정 신호 대기 .
N110 STOPRE	; 전처리 정지
N120 IF $AC_MEA[1]==FALSE GOTOF END	; 측정이 성공했는지 점검합니다 .
N130 R10=$AA_MM1[X]	; 첫 번째 프로그래밍된 트리거 이벤트 (상승 에지) 에서 가져온 측정값을 저장합니다 .
N140 R11=$AA_MM2[X]	; 두 번째 프로그래밍된 트리거 이벤트 (하강 에지) 에서 가져온 측정값을 저장합니다 .
N150 END:	

b) 2 개 측정 시스템 사용

프로그램 코드	설명
...	
N200 MEASA[X]=(31,1,-1) G01 X100 F100	; 두 측정 시스템으로 모드 1 에서 측정 . 이송 경로의 프로브 1 에서 X=100 까지의 상승 / 하강 에지에서 측정 신호 대기 .
N210 STOPRE	; 전처리 정지
N220 IF $AC_MEA[1]==FALSE GOTOF END	; 측정이 성공했는지 점검합니다 .
N230 R10=$AA_MM1[X]	; 상승 에지에서 측정 시스템 1 의 측정값을 저장합니다 .
N240 R11=$AA_MM2[X]	; 상승 에지에서 측정 시스템 2 의 측정값을 저장합니다 .
N250 R12=$AA_MM3[X]	; 하강 에지에서 측정 시스템 1 의 측정값을 저장합니다 .
N260 R13=$AA_MM4[X]	; 하강 에지에서 측정 시스템 2 의 측정값을 저장합니다 .
N270 END:	

예제 2: 모드 2 에서 이동할 거리를 삭제하고 축 측정 (프로그래밍된 순서대로 평가)

프로그램 코드	설명
...	
N100 MEASA[X]=(2,1,-1,2,-2) G01 X100 F100	; 활성 측정 시스템으로 모드 2 에서 측정 . X=100 으로 경로를 이송하면서 프로브 1 상승 에지 , 프로브 1 하강 에지 , 프로브 2 상승 에지 , 프로브 2 하강 에지의 순서로 측정 신호를 기다림
N110 STOPRE	; 전처리 정지
N120 IF $AC_MEA[1]==FALSE GOTOF PROBE2	; 프로브 1 의 측정이 성공했는지 점검합니다 .
N130 R10=$AA_MM1[X]	; 첫 번째 프로그래밍된 트리거 이벤트 (프로브 1 상승 에지) 에서 가져온 측정값 저장
N140 R11=$AA_MM2[X]	; 두 번째 프로그래밍된 트리거 이벤트 (프로브 1 상승 에지) 에서 가져온 측정값 저장
N150 PROBE2:	
N160 IF $AC_MEA[2]==FALSE GOTOF END	; 프로브 2 의 측정이 성공했는지 점검합니다 .
N170 R12=$AA_MM3[X]	; 세 번째 프로그래밍된 트리거 이벤트 (프로브 2 상승 에지) 에서 가져온 측정값 저장
N180 R13=$AA_MM4[X]	; 네 번째 프로그래밍된 트리거 이벤트 (프로브 2 상승 에지) 에서 가져온 측정값 저장
N190 END:	

예제 3: 모드 1 에서 연속 축 측정 (시간 순서대로 평가)

a) 최대 100 개의 측정값 측정

프로그램 코드	설명
...	
N110 DEF REAL MEASVALUE[100]	
N120 DEF INT loop=0	
N130 MEAC[X]=(1,1,-1) G01 X1000 F100	; 활성 측정 시스템의 모드에서 측정 , $AC_FIFO1 에서 측정값 저장 , X=1000 으로 경로를 이송하며 프로브 1 하강 에지의 측정 신호를 기다림
N135 STOPRE	
N140 MEAC[X]=(0)	; 축 위치에 도달할 경우 측정을 종료합니다 .
N150 R1=$AC_FIFO1[4]	; 파라미터 R1 에 축적된 측정값 수를 저장합니다 .
N160 FOR loop=0 TO R1-1	
N170 MEASURED VALUE[loop]=$AC_FIFO1[0]	; $AC_FIFO1 에서 측정값을 읽고 저장합니다 .
N180 ENDFOR	

b) 10 개 측정값 이후 이동할 거리를 삭제하면서 측정

프로그램 코드	설명
...	
N10 WHEN $AC_FIFO1[4]>=10 DO MEAC[x]=(0) DELDTG(x)	; 이동할 거리 삭제
N20 MEAC[x]=(1,1,1,-1) G01 X100 F500	
N30 MEAC [X]=(0)	
N40 R1 = $AC_FIFO1[4]	; 측정값의 개수
...	

추가 정보

측정 작업

측정 작업을 가공 프로그램 또는 동기 동작에서 프로그래밍할 수 있습니다 ('모션 동기화 동작' 참조). 각 축에 대해 주어진 시간에 하나의 측정 작업만 실행할 수 있습니다 .

주

이송 속도는 직접 측정 작업에 맞도록 조정해야 합니다 .

MEASA 및 MEAWA 의 경우 각 위치 제어기 사이클에서 동일한 유형의 단일 트리거 이벤트와 서로 다른 유형의 4개 이하 트리거 이벤트가 발생하는 이송 속도에서만 정확한 결과를 얻을 수 있습니다 .

MEAC를 통한 연속 측정의 경우 보간 사이클 및 위치 제어 사이클 간의 비율은 8:1을 초과하지 않아야 합니다 .

트리거 이벤트

트리거 이벤트는 프로브 수 및 측정 신호의 트리거 기준 (상승 또는 하강 에지) 으로 구성됩니다 .

지정된 프로브에 대해 최대 4 개의 트리거 이벤트를 각 측정마다 처리할 수 있습니다 . 즉 , 각각 두 개의 측정 신호 에지를 가진 최대 2 개의 프로브가 지정됩니다 . 트리거 이벤트의 처리 순서 및 최대 개수는 선택한 모드에 따라 다릅니다 .

주

다음은 모드 1 측정에 적용됩니다 . 동일한 트리거 이벤트는 한 측정 작업에서 한 번만 프로그래밍할 수 있습니다 .

운전 모드

운전 모드의 첫 번째 숫자 (십진수의 첫째 자릿수) 는 필수 측정 시스템을 선택합니다 . 하나의 측정 시스템만 설치되었지만 두 번째 프로그래밍된 경우 설치된 시스템이 자동으로 선택됩니다 .

두 번째 숫자 (십진수의 둘째 자릿수) 는 필수 측정 모드를 선택합니다 . 따라서 측정 과정은 관련 제어기에서 지원하는 옵션에 맞추어집니다 .

- **모드 1**

  트리거 이벤트는 발생된 시간 순서대로 평가됩니다 . 이 모드를 선택하면 하나의 트리거 이벤트만 6 개의 축 모듈에 대해 프로그래밍할 수 있습니다 . 둘 이상의 트리거 이벤트를 지정하면 모드 선택이 자동으로 모드 2 로 전환됩니다 (메시지 없음).

- **모드 2**

  트리거 이벤트는 프로그래밍된 순서대로 평가됩니다 .

- **모드 3**

  트리거 이벤트는 프로그래밍된 순서대로 평가되지만 시작 시 트리거 이벤트 1 을 모니터링하지 않습니다 .

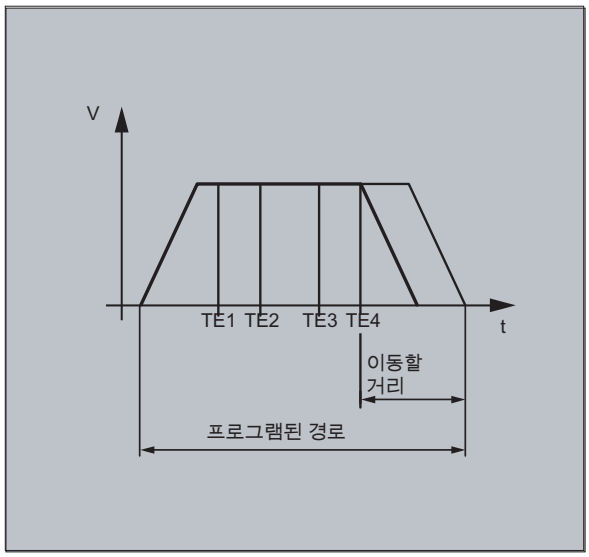
  주

  두 개의 측정 시스템을 사용 중인 경우 두 개의 트리거 이벤트만 프로그래밍할 수 있습니다 .

이동할 거리를 삭제하거나 삭제하지 않고 측정

명령 MEASA를 프로그래밍하면 이동할 거리는 모든 필요한 측정값을 기록할 때까지 삭제되지 않습니다 .

MEAWA 기능은 프로그래밍된 위치로 반드시 복귀해야 하는 특수한 측정 작업의 경우에 사용됩니다 .



주

MEASA는 동기 동작에서 프로그래밍할 수 없습니다. 대체 방법으로, MEAWA 및 이동할 거리 삭제를 동기 동작으로 프로그래밍할 수 있습니다 .

MEAWA 를 통한 측정 작업을 동기 동작에서 시작하는 경우 측정값은 기계 좌표계에서만 사용할 수 있습니다 .

MEASA, MEAWA 의 측정 결과

측정 결과는 다음 시스템 변수에서 사용할 수 있습니다 .

- 기계 좌표계에서 :

$AA_MM1[< 축 >]	트리거 이벤트 1 에서 프로그래밍된 측정 시스템의의 측정값
...	...
$AA_MM4[< 축 >]	트리거 이벤트 4 에서 프로그래밍된 측정 시스템의의 측정값

- 공작물 좌표계에서 :

$AA_WM1[< 축 >]	트리거 이벤트 1 에서 프로그래밍된 측정 시스템의 측정값
...	...
$AA_WM4[< 축 >]	트리거 이벤트 4 에서 프로그래밍된 측정 시스템의의 측정값

주

이러한 변수를 읽을 때 내부 전처리 정지는 생성되지 않습니다 . 전처리 정지는 프로그램의 적절한 위치에서 STOPRE 를 통해 프로그래밍되어야 합니다 . 그렇지 않을 경우 잘못된 값을 읽을 수 있습니다 .

기하 축 / 변환

형상 축에 대해 축 측정을 시작하는 경우 나머지 모든 형상 축에 대해 동일한 측정 작업을 프로그래밍해야 합니다 . 변환에 관련된 축에 대해 동일하게 적용됩니다 .

예제 :
N10 MEASA[Z]=(1,1) MEASA[Y]=(1,1) MEASA[X]=(1,1) G0 Z100

또는
N10 MEASA[Z]=(1,1) POS[Z]=100

두 개의 측정 시스템을 통한 측정 작업

두 개의 측정 시스템에서 측정 작업을 실행하는 경우 관련 축의 두 측정 시스템에 가능한 두 개의 트리거 이벤트를 가져옵니다 . 따라서 예약된 변수 할당은 미리 설정됩니다 .

$AA_MM1[< 축 >]	또는	$AA_MW1[< 축 >]	트리거 이벤트 1 의 측정 시스템 1 에서 측정한 값
$AA_MM2[< 축 >]	또는	$AA_MW2[< 축 >]	트리거 이벤트 1 의 측정 시스템 2 에서 측정한 값
$AA_MM3[< 축 >]	또는	$AA_MW3[< 축 >]	트리거 이벤트 2 의 측정 시스템 1 에서 측정한 값
$AA_MM4[< 축 >]	또는	$AA_MW4[< 축 >]	트리거 이벤트 2 의 측정 시스템 2 에서 측정한 값

프로브 상태

프로브 상태는 다음의 시스템 변수에서 사용할 수 있습니다 .

$A_PROBE[<n>]

<n>= 프로브

값	의미
1	프로브 작동 오류
0	프로브 정상 작동

MEASA, MEAWA 의 측정 작업 상태

프로그램에 평가가 필요한 경우에는 $AC_MEA[n] 를 통해 측정 작업 상태를 검사할 수 있습니다 (여기서 <n> = 프로브 번호). 블록에서 프로그래밍된 프로브 <n>의 트리거 이벤트가 모두 발생하면 이 변수는 '1' 의 값을 반환합니다 . 그렇지 않으면 그 값은 0 입니다 .

주

측정이 동기 동작에서 시작될 경우 $AC_MEA는 더 이상 업데이트되지 않습니다. 이 경우에는 새 PLC 상태 신호인 DB31, … DBX62.3 또는 해당 변수 $AA_MEAACT[< 축 >] 를 검사해야 합니다 .

의미 :

$AA_MEAACT==1: 측정 적용

$AA_MEAACT==0: 측정이 활성화되지 않음

연속 측정 (MEAC)

MEAC 를 위한 측정값은 기계 좌표계에서 사용할 수 있으며 프로그래밍된 FIFO[n] 메모리 (원형 완충 메모리) 에 저장됩니다 . 측정을 위해 두 개의 프로브가 구성된 경우 두 번째 프로브의 측정값은 이러한 용도로 특별히 구성된 FIFO[n+1] 메모리에 별도로 저장됩니다 .

FIFO 메모리는 원형 원리에 따라 측정값이 $AC_FIFO 변수에 기록되는 원형 완충 메모리입니다 . 자세한 내용은 ' 모션 동기화 동작 ' 장을 참조하십시오 .

주

FIFO 내용은 순환 저장소에서 한 번만 읽을 수 있습니다 . 이러한 측정값을 여러 번 사용해야 하는 경우 사용자 데이터에 버퍼링해야 합니다 .

FIFO 메모리의 측정값 수가 머신 데이터에 정의된 최대값을 초과할 경우 측정이 자동으로 종료됩니다 .

측정값을 주기적으로 읽어 연속 측정 프로세스를 실행할 수 있습니다 . 이 경우 새 측정값을 읽을 때 동일한 빈도에서 데이터를 읽어야 합니다 .

인식 프로그래밍 오류

다음 프로그래밍 오류가 감지되어 표시됩니다 .

- MEASA/MEAWA 가 동일한 블록에 MEAS/MEAW 와 함께 프로그래밍되는 경우

  예제 :

  N01 MEAS=1 MEASA[X]=(1,1) G01 F100 POS[X]=100
- MEASA/MEAWA 의 파라미터 개수가 2 보다 작거나 5 보다 큰 경우

  예제 :

  N01 MEAWA[X]=(1) G01 F100 POS[X]=100
- MEASA/MEAWA 의 트리거 이벤트가 1/ -1/ 2/ -2 가 아닌 경우

  예제 :

  N01 MEASA[B]=(1,1,3) B100
- MEASA/MEAWA 가 잘못된 모드를 가진 경우

  예제 :

  N01 MEAWA[B]=(4,1) B100
- MEASA/MEAWA 의 트리거 이벤트가 두 번 프로그래밍된 경우

  예제 :

  N01 MEASA[B]=(1,1,-1,2,-1) B100
- MEASA/MEAWA 를 사용하며 GEO 축을 누락한 경우

  예제 :

  N01 MEASA[X]=(1,1) MEASA[Y]=(1,1) G01 X50 Y50 Z50 F100 ;GEO 축 X/Y/Z
- 측정 작업이 GEO 축과 일치하지 않는 경우

  예제 :

  N01 MEASA[X]=(1,1) MEASA[Y]=(1,1) MEASA[Z]=(1,1,2) G01 X50 Y50 Z50 F100

4.9 OEM 사용자를 위한 특수 기능 (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829)

OEM 주소

OEM 주소의 의미는 OEM 사용자에 의해 결정됩니다 . 해당 기능은 컴파일 싸이클을 통해 조합됩니다. 예비용으로 5개의 OEM 주소 (OMA1 ... OMA5) 가 있습니다. 주소 식별자는 설정 가능합니다 . 블록에서 OEM 주소를 프로그래밍할 수 있습니다 .

예비용 G 코드 호출

OEM 사용자를 위해서는 다음과 같은 G 코드 호출이 예비용으로 제공됩니다 .

- OEMIPO1, OEMIPO2 (G 코드 그룹 1 부터)
- G810 ... G819 (G 코드 그룹 31)
- G820 ... G829 (G 코드 그룹 32)

해당 기능은 컴파일 싸이클을 통해 조합됩니다 .

기능 및 서브루틴

또한 OEM 사용자는 파라미터 전송을 통해 미리 정의된 기능 및 서브루틴을 설정할 수 있습니다 .

주

공작물 시뮬레이션

SW 4.4 까지는 컴파일 싸이클이 지원되지 않으며 , SW 4.4 이상에서는 일부 컴파일 싸이클 (CC) 만 공작물 시뮬레이션을 위해 지원됩니다 .

지원되지 않는 컴파일 싸이클 (OMA1 ... OMA5, OEMIPO1/2, G810 ... G829, 해당 절차 및 기능) 의 가공 프로그램에 사용되는 언어 명령 - 따라서 개별적으로 처리하지 않아도 알람 메시지 생성 및 시뮬레이션 취소가 수행됩니다 .

해결 : 가공 프로그램 ($P_SIM query) 에서 누락된 CC 고유 언어 요소를 개별적으로 처리하십시오 .
예제 :

```
N1 G01 X200 F500
IF (1==$P_SIM)
N5 X300 ; CC 시뮬레이션에서는 활성화되지 않음
ELSE
N5 X300 OMA1=10
ENDIF
```

4.10 코너 감속을 사용한 피드 감소 (FENDNORM, G62, G621)

작동

자동 코너 감속을 사용하여 코너에 도달하기 전에 벨 곡선에 따라 피드가 감소됩니다 . 또한 세팅 데이터를 통한 가공 관련 공구 동작의 범위를 파라메타로 지정할 수 있습니다 . 이것에는 다음이 포함됩니다 :

- 피드 감소의 시작 및 끝
- 피드가 감소되는 상태에서 오버라이드
- 관련 코너의 삭제

관련 코너는 이러한 코너의 내부 각도가 세팅 데이터에서 파라메타로 지정된 코너보다 작습니다 .

기본값 FENDNORM 은 자동 코너 오버라이드의 기능을 비활성화합니다 .

참조 문서 :
/FBFA/ " 작동 설명 ISO 언어 "

구문

FENDNORM

G62 G41

G621

의미

FENDNORM	자동 코너 감속 OFF
G62	공구 반경 옵셋이 활성 상태일 때 내부 코너에서 코너 감속
G621	공구 반경 옵셋이 활성 상태일 때 모든 코너에서 코너 감속

G62 는 다음 상태에서의 내부 코너에만 적용 :

- 활성 공구 반경 옵셋 G41, G42 및
- 활성 연속 경로 모드 G64, G641

다음으로 인해 감소된 피드에서 코너에 접근한다 .

F * (피드 감소에 대한 오버라이드) * 피드 오버라이드

중심 경로를 기준으로 코너에서 공구의 방향을 변경할 정확한 지점에서 최대 가능 피드 감소에 도달한다 .

G621 은 FGROUP **에 의해 정의된 축의 각 코너에 있는 G62 와 동일하게 적용됩니다** .

4.11 프로그래밍된 이동 종료 기준 (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA)

기능

경로 보간 (G601, G602 및 G603) 의 블록 변경 기준과 마찬가지로 , 가공 프로그램이나 명령 / PLC 축의 동기 동작에 단일 축 보간을 위한 모션 종료 기준을 프로그래밍할 수 있습니다 .

이동 종료 기준 설정은 가공 프로그램 블록 및 단일 축 이동의 기술 싸이클 블록이 신속하거나 느리게 완료되는 것에 영향을 줍니다 . FC15/16/18 을 통해 PLC 에 동일하게 적용됩니다 .

구문

FINEA[<axis>]
COARSEA[<axis>]
IPOENDA[<axis>]
IPOBRKA(< 축 >[,< 시간 인스턴트 >])
ADISPOSA[< 축 >]=(< 모드 >,< 범위 크기 >)

중요도

FINEA:	이동 종료 기준 : " 미세 정위치 정지 "	
	적용 :	모달
COARSEA:	이동 종료 기준 : " 일반 정위치 정지 "	
	적용 :	모달
IPOENDA:	이동 종료 기준 : " 보간자 정지 "	
	적용 :	모달
IPOBRKA:	블록 변경 기준 : 정지 동작	
	적용 :	모달
ADISPOSA:	이동 종료 기준에 대한 공차 범위	
	적용 :	모달
< 축 >:	채널 축 이름 (X, Y, ...)	
< 시간 인스턴트 >:	블록 변경 시간 , 정지 동작을 % 로 참조 :	
	• 100% = 정지 동작 시작	
	• 0% = 정지 동작 종료 , IPOENDA 와 동일한 중요도	
	유형 :	REAL
< 모드 >:	공차 범위 참조	
	값 범위 :	0 허용 오차 창 비활성
		1 설정 위치에 따라 허용 오차 창
		2 실제 위치에 따라 허용 오차 창
	유형 :	INT
< 범위 크기 >:	허용 오차 창의 크기	
	유형 :	REAL

예제

예제 1: 이동 종료 기준 : " 보간자 정지 "

프로그램 코드

```
; 포지셔닝 축 X 를 100, 속도 200 m/min, 가속도 90% 로 이송 ,
; 이동 종료 기준 : 보간자 정지
N110 G01 POS[X]=100 FA[X]=200 ACC[X]=90 IPOENDA[X]


; 동기 동작 :
; ALWAYS IF: 입력 1 을 설정
; THEN 포지셔닝 축 X 를 50, 속도 200 m/min, 가속도 140% 이송 ,
; 이동 종료 기준 : 보간자 정지
N120 EVERY $A_IN[1] DO POS[X]=50 FA[X]=200 ACC[X]=140 IPOENDA[X]
```

예제 2: 블록 변경 기준 : " 정지 동작 "

프로그램 코드	코멘트
	; 기본 설정이 적용됩니다 .
N40 POS[X]=100	; X 에서 위치 100 으로 포지셔닝 이동 ; 블록 변경 기준 : 미세 정위치 정지
N20 IPOBRKA(X,100)	; 블록 변경 기준 : " 정지 동작 ", ; 100% = 정지 동작 시작
N30 POS[X]=200	; X 축이 제동되기 시작하는 즉시 블록이 변경됩니다 .
N40 POS[X]=250	; X 축이 더 이상 위치 200 에서 제동되지 않으나 ; 위치 250 으로 이송은 계속합니다 . ; 축이 제동을 시작하자마자 블록이 변경됩니다 .
N50 POS[X]=0	; X 축이 제동되어 위치 0 으로 돌아갑니다 . ; 위치 0 에서 " 미세 정위치 정지 " 로 블록 변경이 이루어집니다 .
N60 X10 F100	; X 축은 경로 축으로 위치 10 으로 이송됩니다 .

추가 정보

이동 종료 기준에 대한 시스템 변수

효과적인 이동 종료 기준은 시스템 변수 $AA_MOTEND 를 사용하여 읽을 수 있습니다 .

참고 자료 : /LIS2sl/ 목록 매뉴얼 , 2 권

블록 변경 기준 : " 정지 동작 " (IPOBRKA)

블록 변경 기준 "정지 동작"을 활성화할 때 옵션 블록 변경 시간 인스턴트 값이 프로그래밍되는 경우 , 그 값은 다음 포지셔닝 모션에 적용되며 메인 실행에 동기화된 셋팅 데이터로 쓰여집니다 . 블록 변경 시간 인스턴트 값을 지정하지 않으면 셋팅 데이터의 실제 값이 사용됩니다 .

SD43600 $SA_IPOBRAKE_BLOCK_EXCHANGE

다음에 축 이동 종료 기준 (FINEA, COARSEA , IPOENDA) 을 축에 프로그래밍하면 해당 액세스에 대한 IPOBRKA 가 비활성화됩니다 .

추가 블록 변경 기준 : " 공차 범위 " (ADISPOSA)

ADISPOSA 를 사용하면 블록 끝 (실제 위치 또는 지령치 위치로서) 에 있는 공차 범위를 추가 블록 변경 기준으로 정의할 수 있습니다 . 그런 경우에는 블록 변경을 위한 두 가지 조건을 만족해야 합니다 .

- 블록 변경 기준 : " 정지 동작 "
- 블록 변경 기준 : " 공차 범위 "

참고 자료

포지셔닝 축의 블록 변경 기준과 관련한 자세한 정보는

- 기능 매뉴얼 , 확장 기능 ; 포지셔닝 축 (P2)
- 프로그래밍 매뉴얼 , 기본 사항 ; ' 이송 속도 제어 ' 장을 참조하십시오 .

4.12 프로그램 가능한 서보 파라미터 세트 (SCPARA)

작동

가공 프로그램 및 동기화된 작동의 파라미터 세트 (MD 구성) 는 SCPARA 를 사용하여 프로그래밍할 수 있습니다 (현재까지는 PLC 를 통해서만 가능).

DB3n DBB9 bit3

PLC 및 NCK 간에 충돌이 없도록 하기 위해 PLC → NCK 인터페이스에서 추가 비트가 정의됩니다 :

DB3n DBB9 bit3 "SCPARA 에 의한 파라미터 세트 선택 비활성화 ".

SCPARA 의 파라미터 세트 입력이 금지되면 이 설정을 프로그래밍하더라도 오류 메시지가 출력되지 않습니다 .

구문

SCPARA[<axis>]=<value>

의미

SCPARA	파라미터 블록 정의
<axis>	채널 축 이름 (X, Y, ...)
<value>	원하는 파라미터 블록 (1<= 값 <=6)

주

실제 파라미터 세트는 시스템 변수 $AA_SCPAR[<axis>] 를 사용하여 조회할 수 있습니다 .

G33, G331 또는 G332 의 경우 제어 시스템에서 가장 적합한 파라미터 세트가 선택됩니다 .

서보 파라미터 세트를 가공 프로그램 , 동기화된 작동 또는 PLC 에서 **변경**해야 하는 경우 PLC 사용자 프로그램을 확장해야 합니다 .

참조 문서 :
/FB1/ 작동 설명서 기본 기능 ; 피드 속도 (V1),
" 피드 속도 영량 " 섹션 .

예제

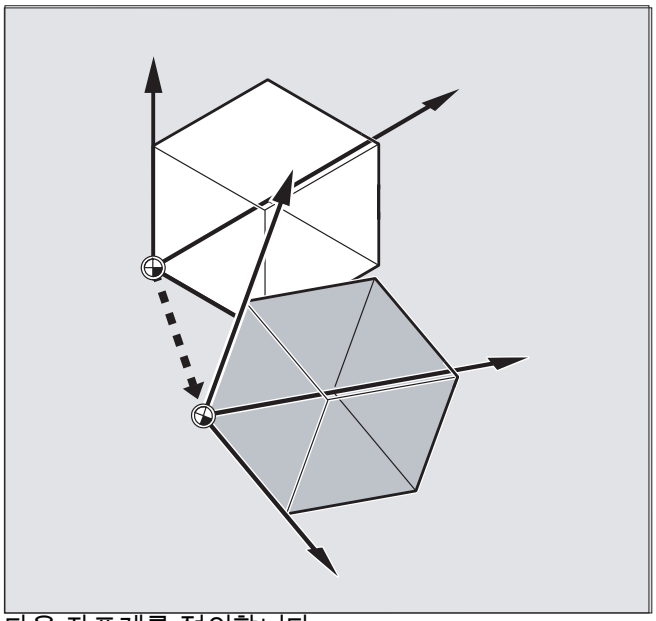
프로그램 코드	비고
...	
N110 SCPARA[X]= 3	; X 축에 대해 세 번째 파라미터 세트가 선택됩니다 .
...	

좌표 변형 (프레임) 5

5.1 프레임 변수를 통한 좌표 변환

작동

프로그래밍 설명서 " 기본 사항 " 에 이미 설명된 프로그래밍 옵션 이외에 미리 정의된 프레임 변수로 좌표계를 정의할 수 있습니다 .



다음 좌표계를 정의합니다 .

MCS: 기계 좌표계

BCS: 기본 좌표계

BZS: 기본 원점 좌표계

SZS: 설정 가능한 제로 좌표계

WCS: 공작물 좌표계

미리 정의된 프레임 변수

미리 정의된 프레임 변수는 제어 언어에 이미 정의되어 사용 및 적용되는 키워드이며 NC 프로그램에서 처리될 수 있습니다 .

가능한 프레임 변수 :

- 기본 프레임 (기본 옵셋)
- 설정 가능한 프레임
- 프로그램 가능한 프레임

값 할당 및 실제 값 읽기

프레임 변수 / 프레임 관계

프레임 값을 프레임 변수에 할당하여 좌표 변환을 실행할 수 있습니다 .

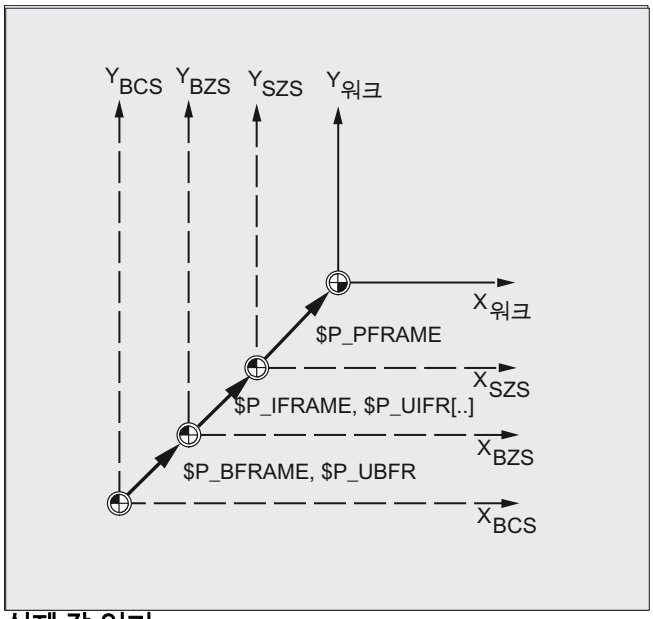
예 : $P_PFRAME=CTRANS(X,10)

프레임 변수 :

$P_PFRAME: 현재 프로그램 가능한 프레임입니다 .

프레임 :

CTRANS(X,10): 10mm 단위의 프로그램 가능한 X 축의 워크 옵셋입니다 .



실제 값 읽기

좌표계의 현재 실제 값은 가공 프로그램의 미리 정의된 변수를 통해 읽을 수 있습니다 .

$AA_IM[axis]: MCS 에서 실제 값 읽기

$AA_IB[axis]: BCS 에서 실제 값 읽기

$AA_IBN[axis]: BOS 에서 실제 값 읽기

$AA_IEN[axis]: SZS 에서 실제 값 읽기

$AA_IW[axis]: WCS 에서 실제 값 읽기

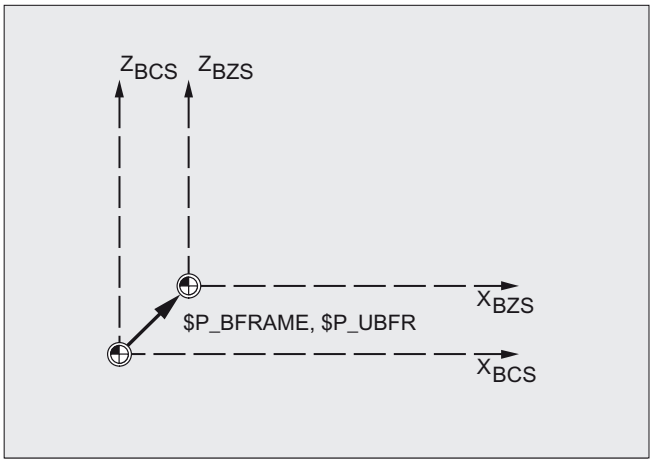
5.1.1 미리 정의된 프레임 변수 ($P_BFRAME, $P_IFRAME, $P_PFRAME, $P_ACTFRAME)

$P_BFRAME

기본 좌표계 (BCS) 및 기본 원점 좌표계 (BOS) 사이 기준을 구성하는 현재 기본 프레임 변수입니다 .

프로그램에서 즉시 활성화하기 위해 $P_UBFR 을 통해 설명된 기본 프레임의 경우 ,

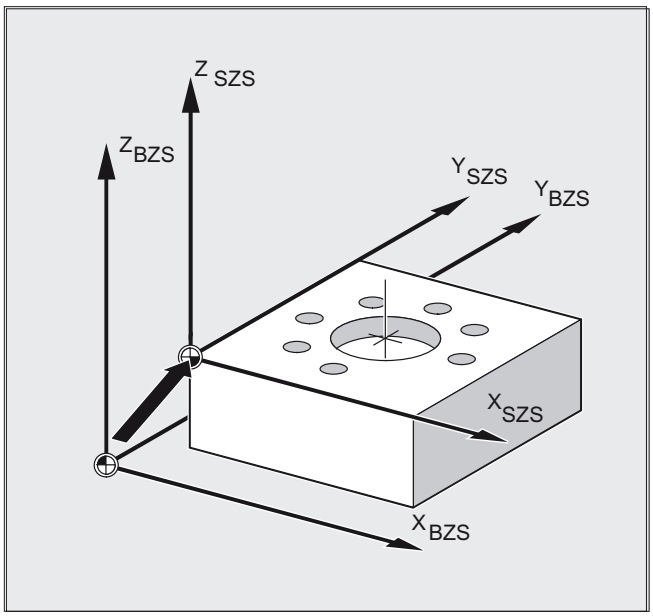
- G500, G54...G599 를 프로그래밍해야 하거나 또는
- $P_UBFR 을 통해 $P_BFRAME 을 설명해야 합니다 .



$P_IFRAME

기본 원점 좌표계 (BOS) 및 공작물 제로 시스템 (SZS) 사이 기준을 구성하는 현재 셋터블 프레임 변수입니다 .

- $P_IFRAME 은 $P_UIFR[$P_IFRNUM] 에 해당합니다 .
- 예를 들어 G54 를 프로그래밍한 후 $P_IFRAME 에는 G54 에 의해 정의된 변환 , 회전 , 배율 및 미러링이 포함되어 있습니다 .

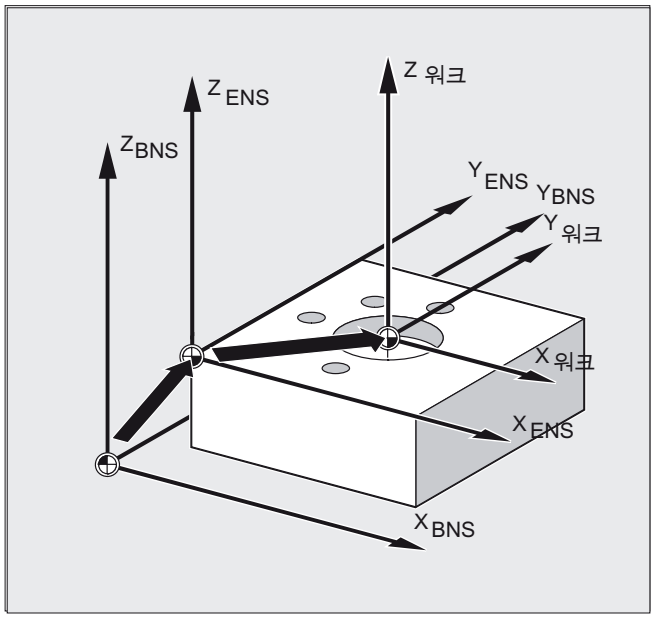


$P_PFRAME

공작물 제로 시스템 (SZS) 및 공작물 좌표계 (WCS) 사이 기준을 구성하는 현재 프로그램 가능한 프레임 변수입니다 .

아래의 값을 포함합니다 .

- TRANS/ATRANS, ROT/AROT, SCALE/ASCALE, MIRROR/AMIRROR 의 **프로그래밍** 또는
- CROT, CMIRROR, CSCALE 의 **할당**



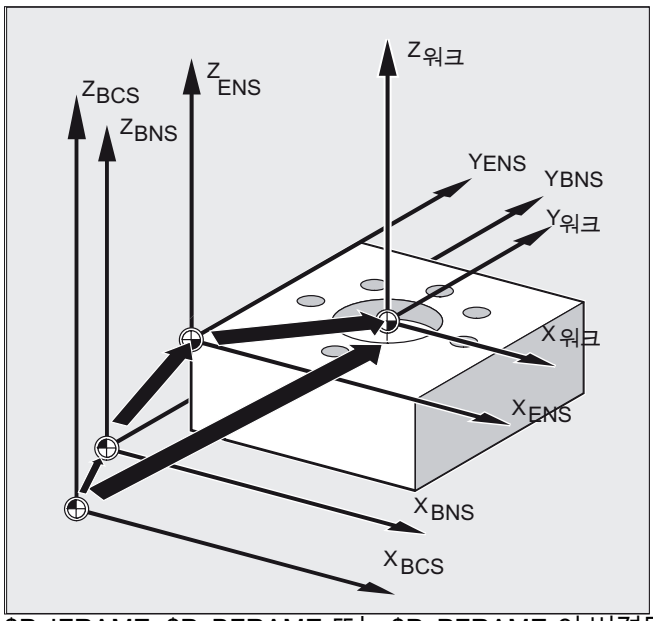
$P_ACTFRAME

아래의 내용을 포함하는 전체 프레임을 표시합니다 .

- 현재 기본 프레임 변수 $P_BFRAME 과의 연결 ,
- 현재 셋터블 프레임 변수 $P_IFRAME 과 시스템 프레임 연결 및
- 현재 프로그램 가능한 프레임 변수 $P_IFRAME 과 시스템 프레임 연결 .

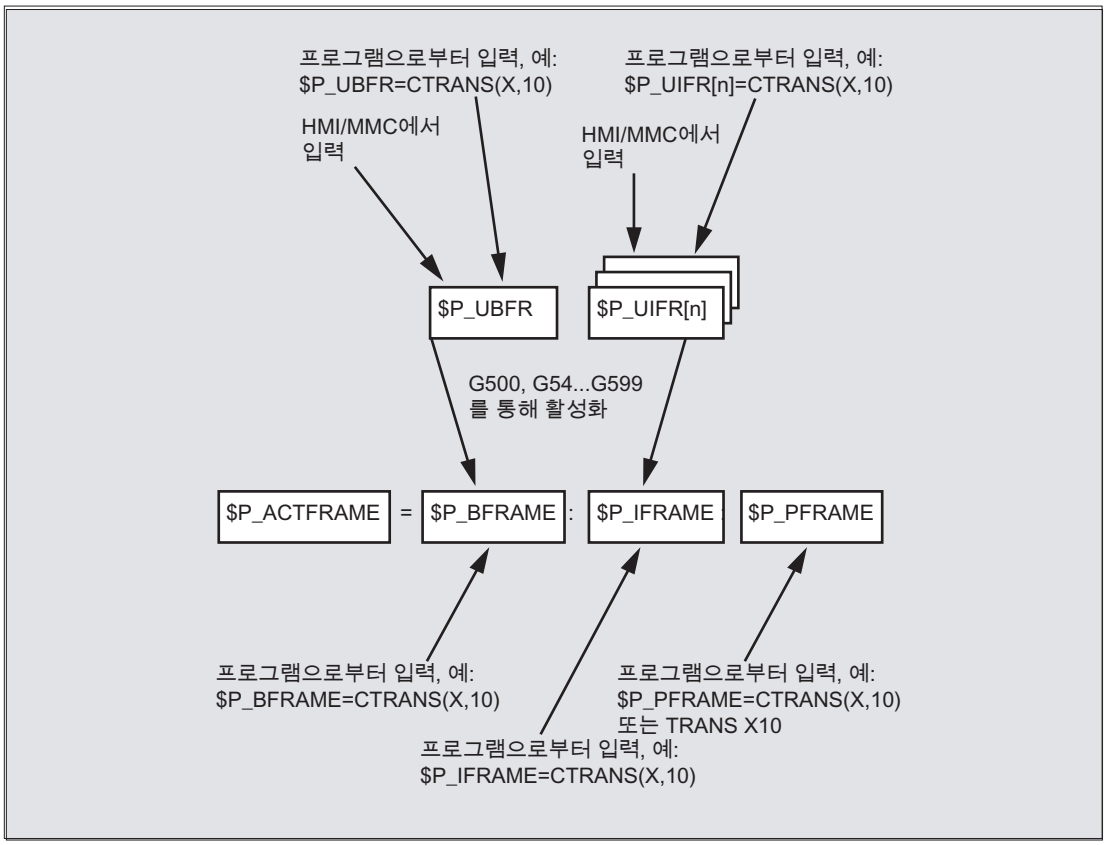
시스템 프레임에 대해서는 " 채널에서 작동하는 프레임 " 섹션을 참조하십시오 .

$P_ACTFRAME 은 현재 유효한 공작물 제로 점과 관계됩니다 .



$P_IFRAME, $P_BFRAME 또는 $P_PFRAME 이 변경된 경우 $P_ACTFRAME 이 다시 계산됩니다 .

$P_ACTFRAME 은 $P_BFRAME:$P_IFRAME:$P_PFRAME 에 해당합니다 .



기본 프레임 및 셋터블 프레임은 MD 20110 RESET_MODE_MASK 를 다음과 같인 설정할 경우 리셋 이후 적용됩니다 .

Bit0=1, bit14=1 --> $P_UBFR (기본 프레임) acts

Bit0=1, bit5=1 --> $P_UIFR [$P_UIFRNUM](셋터블 프레임) acts

셋터블 프레임

기본 프레임은 $P_UBFR 을 통해 프로그래밍되지만 동시에 가공 프로그램에서 활성화되지 않습니다 . $P_UBFR 을 통해 프로그래밍된 기본 프레임은 다음과 같은 경우 계산에 포함됩니다 .

- 리셋되고 비트 0 및 14 가 MD RESET_MODE_MASK 에 설정된 경우 및
- G500,G54...G599 명령문이 실행된 경우 .

$P_UIFR[n] 셋터블 프레임

미리 정의된 프레임 변수 $P_UIFR[n] 을 사용하여 가공 프로그램의 셋터블 제로 옵셋 G54 - G599 를 읽거나 쓸 수 있습니다 .

이러한 변수는 $P_UIFR[n] 이라는 프레임 유형의 1 차원 배열을 제공합니다 .

G 명령에 할당

5 개의 $P_UIFR[0]... $P_UIFR[4] 셋터블 프레임 또는 5 개의 동등한 G 명령 (G500 및 G54 - G57) 을 저장할 수 있습니다 .

$P_IFRAME=$P_UIFR[0] 은 G500 에 해당합니다 .

$P_IFRAME=$P_UIFR[1] 은 G54 에 해당합니다 .

$P_IFRAME=$P_UIFR[2] 은 G55 에 해당합니다 .

$P_IFRAME=$P_UIFR[3] 은 G56 에 해당합니다 .

$P_IFRAME=$P_UIFR[4] 은 G57 에 해당합니다 .

머신 데이터를 통해 프레임 수를 변경할 수 있습니다 .

$P_IFRAME=$P_UIFR[5] 은 G505 에 해당합니다 .

... ...

$P_IFRAME=$P_UIFR[99] 은 G599 에 해당합니다 .

주

최대 100 개의 좌표계를 생성할 수 있으며 예를 들어 다양한 치구의 원점을 서로 다른 프로그램에서 전역적으로 호출할 수 있습니다 .

⚠ 주의

프레임 변수는 NC 프로그램의 개별 NC 블록에서 프로그래밍해야 합니다 . **예외:**G54, G55, ... 를 통한 셋터블 프레임의 프로그래밍

5.2 프레임 변수 / 프레임에 할당

5.2.1 직접 값 할당 (축 값 , 각도 , 배율)

작동

NC 프로그램의 프레임 또는 프레임 변수에 값을 직접 할당할 수 있습니다 .

구문

$P_PFRAME=CTRANS (X, axis value, Y, axis value, Z, axis value, …)

$P_PFRAME=CROT (X, angle, Y, angle, Z, angle, …)

$P_UIFR[..]=CROT (X, angle, Y, angle, Z, angle, …)

$P_PFRAME=CSCALE (X, scale, Y, scale, Z, scale, …)

$P_PFRAME=CMIRROR (X, Y, Z)

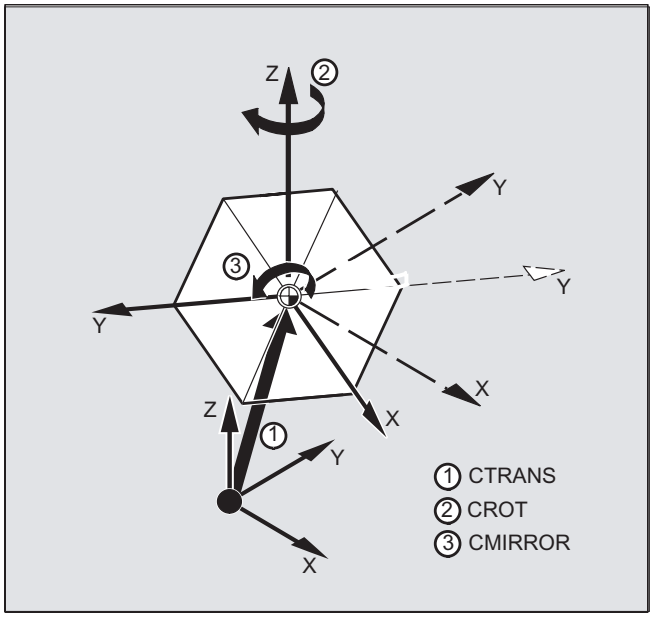
$P_BFRAME 프로그램은 $P_PFRAME 에 대해 아날로그 방식으로 수행됩니다 .

의미

CTRANS	지정된 축의 변환
CROT	지정된 축 주위 회전
CSCALE	지정된 축에 대한 배율 변경
CMIRROR	지정된 축의 방향 전환
X Y Z	지정된 형상 축의 방향으로 옵셋 값
축 값	옵셋의 축 값 할당
각도	지정된 축 주위 회전 각도 할당
배율	배율 변경

예제

현재 프로그램 가능한 프레임에 값을 할당하여 변환 , 회전 및 미러링을 수행합니다 .



N10 $P_PFRAME=CTRANS(X,10,Y,20,Z,5):CROT(Z,45):CMIRROR(Y)

프레임 적색 구성 요소는 미리 할당한 다른 값입니다 .

CROT 를 통해 세 가지 모든 UIFR 구성 요소에 값을 미리 할당합니다 .

프로그램 코드	비고
$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	

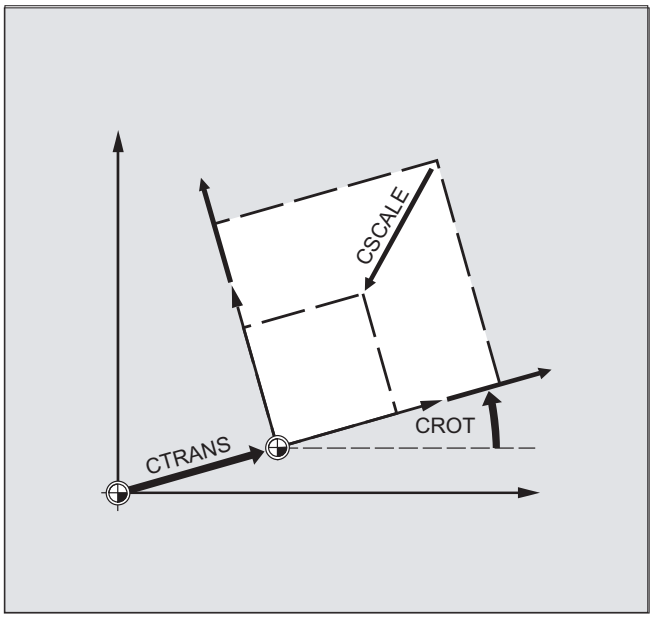
설명

여러 산술 규칙을 연속적으로 프로그래밍할 수 있습니다 .

예 :

$P_PFRAME=CTRANS(…):CROT(…):CSCALE…

명령은 콜론 연결 연산자에 의해 연결되어야 합니다 : (...):(...). 이렇게 하면 프로그래밍된 순서로 첫 번째 명령이 연결되고 두 번째 명령이 추가로 실행됩니다 .



주

위의 명령으로 프로그래밍된 값이 프레임에 할당되고 저장됩니다 .

값은 활성 프레임 변수 $P_BFRAME 또는 $P_PFRAME 의 프레임에 할당될 때까지 활성화되지 않습니다 .

5.2.2 프레임 구성 요소 읽기 및 변경 (TR, FI, RT, SC, MI)

작동

이 기능을 통해 예를 들어 특정 옵셋 값 또는 회전 각도와 같은 프레임의 **개별** 데이터에 액세스할 수 있습니다 . 이러한 값을 수정하거나 다른 변수에 할당할 수 있습니다 .

구문

R10=$P_UIFR[$P_UIFNUM,X,RT]	현재 유효한 설정 가능한 워크 옵셋 $P_UIFRNUM 의 X 축 주위 회전 각도 RT 를 변수 R10 에 할당합니다 .
R12=$P_UIFR[25,Z,TR]	설정 프레임 No. 25 데이터 레코드의 Z 에서 옵셋 값 TR 을 변수 R12 에 할당합니다 .
R15=$P_PFRAME[Y,TR]	현재 프로그램 가능한 프레임의 Y 에서 옵셋 값 TR 을 변수 R15 에 할당합니다 .
$P_PFRAME[X,TR] = 25	현재 프로그램 가능한 프레임의 X 에서 옵셋 값 TR 을 수정합니다 . X25 가 즉시 적용됩니다 .

의미

$P_UIFRNUM	이 명령으로 현재 유효한 설정 가능한 워크 옵셋에 대한 기준이 자동으로 구성됩니다 .
P_UIFR[n,…,…]	설정 가능한 프레임 수 n 에 액세스할 프레임 수 n 을 지정합니다 .
	읽거나 수정할 구성 요소를 지정합니다 :
TR	TR 변환
FI	FI 미세 변환
RT	RT 회전
SC	SC 배율 수정
MI	MI 미러링
X Y Z	해당 축 X, Y, Z 도 지정됩니다 (예제 참조).

RT 회전 값 범위

첫 번째 형상 축 주위 회전 :	-180° ~ +180°
두 번째 형상 축 주위 회전 :	-90° ~ +90°
세 번째 형상 축 주위 회전 :	-180° ~ +180°

설명

프레임 호출

시스템 변수 $P_UIFRNUM 을 지정하여 $P_UIFR or G54, G55, ... 를 통해 현재 워크 옵셋 설정에 액세스할 수 있습니다
($P_UIFRNUM 에는 현재 설정된 프레임 수 포함).

다른 모든 저장된 설정 가능한 $P_UIFR 프레임은 해당 번호 $P_UIFR[n] 을 지정하여 호출합니다 .

미리 정의된 프레임 변수 및 사용자 정의 프레임의 경우 예를 들어 $P_IFRAME 과 같은 이름을 지정합니다 .

데이터 호출

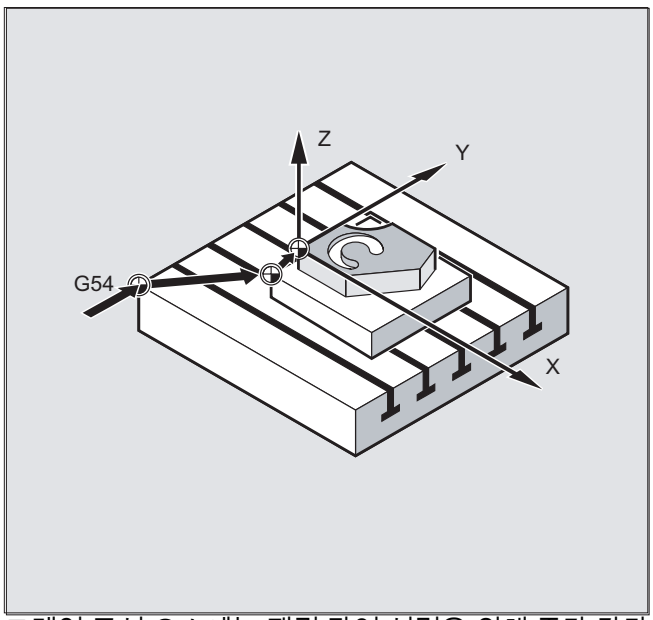
액세스 또는 수정할 값의 축 이름 및 프레임 구성 요소는 예를 들어 [X, RT] 또는 [Z, MI] 와 같이 각괄호 내에 기록됩니다 .

5.2.3 전체 프레임 연결

작동

전체 프레임을 다른 프레임에 할당하거나 프레임을 NC 프로그램에서 서로 연결할 수 있습니다 .

프레인 연결은 팰릿에 정렬된 여러 공작물의 설명에 적합하여 동일한 프로세스에서 가공되어야 합니다 .



프레임 구성 요소에는 팰릿 작업 설명을 위해 중간 값만 포함될 수 있습니다 . 이러한 구성 요소는 다양한 공작물 제로 생성을 위해 연결됩니다 .

구문

프레임 할당

```
DEF FRAME SETTING1
SETTING1=CTRANS(X,10)
$P_PFRAME=SETTING1

DEF FRAME SETTING4
SETTING4=$P_PFRAME
$P_PFRAME=SETTING4
```

사용자 프레임 SETTING1 의 값을 현재 프로그램 가능한 프레임에 할당합니다 .

현재 프로그램 가능한 프레임이 일시적으로 저장되고 다시 호출 가능합니다 .

프레임 체인

프레임은 프로그램된 순서대로 연결됩니다 . 프레임 구성 요소 (변환 , 회전 등) 는 연속적으로 추가 실행됩니다 .

$P_IFRAME=$P_UIFR[15]:$P_UIFR[16]	예를 들어 $P_UIFR[15] 에는 워크 옵셋 데이터가 포함됩니다 . 예를 들어 회전 데이터와 같은 $P_UIFR[16]의 데이터가 후속적으로 추가 처리됩니다 .
$P_UIFR[3]=$P_UIFR[4]:$P_UIFR[5]	설정 가능한 프레임 3 은 설정 가능한 프레임 4 및 5 를 연결하여 생성됩니다 .

주

프레임은 각각 연결 콜론 **:** 을 사용하여 연결되어야 합니다 . .

5.2.4 새 프레임 정의 (DEF FRAME)

작동

위에서 설명한 미리 정의된 설정 가능한 프레임 이외에 새 프레임을 생성하기 위한 옵션이 제공됩니다 . 이 작업은 FRAME 유형의 변수를 생성하여 수행하며 선택한 프레임의 이름을 할당할 수 있습니다 .

함수 CTRANS, CROT, CSCALE 및 CMIRROR 를 사용하여 값을 NC 프로그램의 프레임에 할당할 수 있습니다 .

구문

DEF FRAME PALETTE1

PALETTE1=CTRANS(…):CROT(…)…

의미

DEF FRAME	새 프레임을 생성합니다 .
PALETTE1	새 프레임의 이름
=CTRANS(...): CROT(...)...	값을 가능한 함수에 할당합니다 .

5.3 일반 및 미세조정 옵셋 (CFINE, CTRANS)

작동

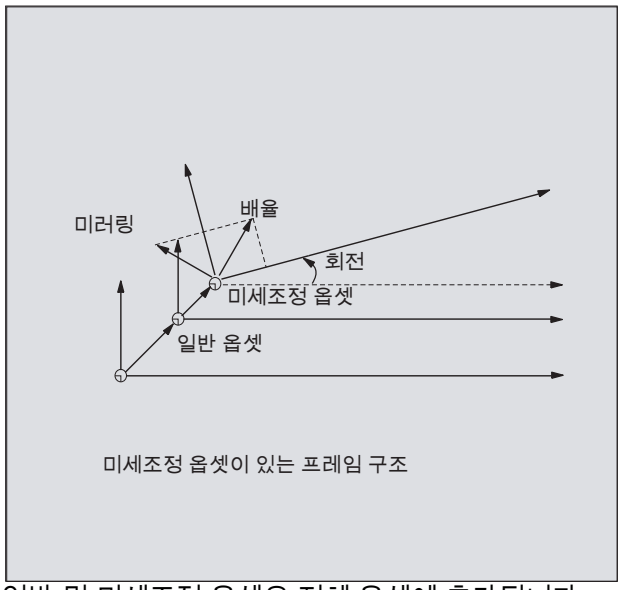
미세조정 옵셋

기본 프레임 및 기타 모든 설정 가능한 프레임의 미세조정 옵셋은 명령 CFINE (X, ..,Y, ...) 을 통해 프로그래밍할 수 있습니다 .

MD18600 $MN_MM_FRAME_FINE_TRANS=1 인 경우 미세조정 옵셋만 수행할 수 있습니다 .

일반 옵셋

일반 옵셋은 CTRANS(...) 를 통해 정의됩니다 .



일반 및 미세조정 옵셋은 전체 옵셋에 추가됩니다 .

구문

$P_UBFR=CTRANS(x, 10) : CFINE(x, 0.1) : CROT(x, 45)	; 옵셋 연결 , ; 미세조정 옵셋 및 회전
$P_UIFR[1]=CFINE(x, 0.5 y, 1.0, z, 0.1)	전체 프레임은 일반 옵셋 i 을 포함하여 CFINE 을 통해 덮어씁니다 .

미세조정 옵셋의 개별 구성 요소에 대한 액세스는 구성 요소 사양 FI(미세 변환) 를 통해 수행됩니다 .

DEF REAL FINEX	;FINEX 변수의 정의
FINEX=$P_UIFR[$P_UIFNUM, x, FI]	;FINEX 변수를 사용하여 미세조정 옵셋 가져오기
FINEX=$P_UIFR[3, x, FI]$P	;FINEX 변수를 사용하여 세 번째 프레임에서 X 축의 미세조정 옵셋 가져오기

의미

CFINE(x, value, y, value, z, value)	여러 축에 대한 미세조정 옵셋 추가 옵셋 (변환)
CTRANS(x, value, y, value, z, value)	여러 축에 대한 일반 옵셋 절대 옵셋 (변환).
x y z	축의 제로 이동 (최대 8)
값	변환 부분

장비 제조업체

미세조정 옵셋은 MD18600 $MN_ MM_FRAME_FINE_TRANS 를 사용하여 다음 버전에서 구성할 수 있습니다 .

0:
미세조정 옵셋을 입력 또는 프로그래밍할 수 없습니다 . G58 및 G59 를 수행할 수 없습니다 .

1:
설정 가능한 프레임 , 기본 프레임 , 프로그램 가능한 프레임 , G58 및 G59 의 미세조정 옵셋을 입력 / 프로그래밍할 수 있습니다 .

설명

HMI 조작을 통해 변경된 미세조정 옵셋은 예를 들어 G500, G54...G599 를 통한 활성화와 같이 해당 프레임의 활성화가 완료될 때까지 적용되지 않습니다 . 활성화가 완료되면 프레임의 미세조정 옵셋은 프레임이 활성 상태인 전체 시간 동안 활성 상태를 유지합니다 .

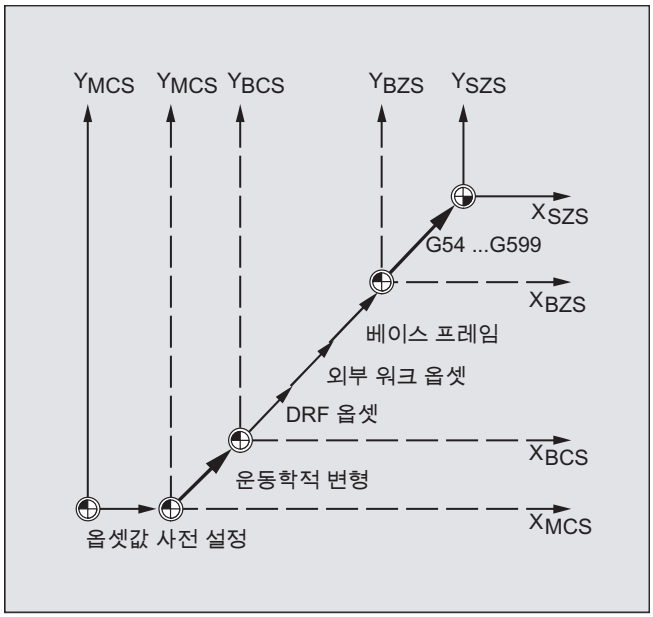
프로그램 가능한 프레임에는 미세조정 옵셋이 없습니다 . 프로그램 가능한 프레임에 미세조정 옵셋이 적용된 프레임이 할당될 경우 전체 옵셋은 일반 및 미세조정 옵셋을 추가하여 구성됩니다 . 프로그램 가능한 프레임을 읽을 때 미세조정 옵셋은 항상 0 입니다 .

5.4 외부 워크 옵셋

작동

이 방법은 기본 및 공작물 좌표계 간 영점 이동을 위한 다른 방법입니다 .

외부 워크 옵셋에서는 선형 변환만 프로그래밍할 수 있습니다 .



프로그래밍

$AA_ETRANS 옵셋 값은 축 관련 시스템 변수를 할당하여 프로그래밍됩니다 .

옵셋 값 할당

$AA_ETRANS[axis]=RI

RI 는 새 값이 포함된 REAL 유형의 산술 변수입니다 .

외부 옵셋은 일나적으로 PLC 에 의해 설정되지만 가공 프로그램에는 지정되지 않습니다 .

주

가공 프로그램에 입력되는 값은 해당 신호가 VDI 인터페이스 (NCU-PLC 인터페이스) 에서 활성화되는 경우에만
활성화됩니다 .

5.5 사전 설정 옵셋 (PRESETON)

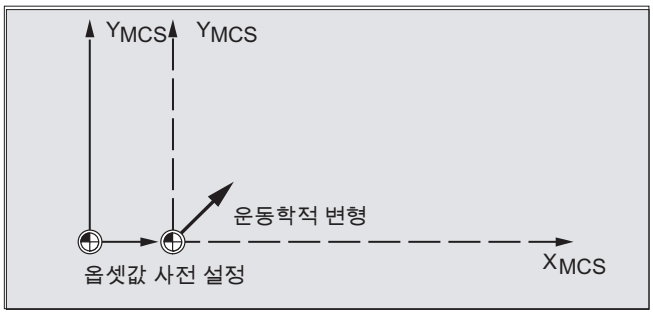
기능

경우에 따라 PRESETON 을 이용해 이미 원점을 참조한 기계 축에 실제 값을 새로 할당해야 할 수도 있습니다 . 이것은 기계 좌표계에서 워크 옵셋에 해당합니다 .

⚠ 주의

PRESETON 을 실행하면 기계 축은 '원점 복귀 안됨' 상태가 됩니다. 따라서 원점 복귀가 필요 없는 축에 대해서만 이 기능을 사용하도록 권장합니다 . 원래 기계 좌표계를 복구하려면 기계 축을 G74 (원점 복귀) 등을 이용해 다시 원점 복귀하도록 해야 합니다 .

참고 자료 : 프로그래밍 매뉴얼 기본편 . 보충 명령 , 원점 복귀 (G74)



구문

PRESETON(< 축 >, < 값 >, ...)

의미

PRESETON	사전 설정 실제 값 메모리
< 축 >	기계 축 이름
< 값 >	기계 좌표계에서 기계 축의 새로운 실제 값

주

동기 동작을 통해 실제 값을 설정할 때는 WHEN 또는 EVERY 키워드만 사용해야 합니다 .

예제

기하 축 : A, 관련 기계 축 : X1

프로그램 코드	코멘트
N10 G0 A100	; 축 A 가 위치 100 으로 이송됩니다 .
N20 PRESETON(X1,50)	; 위치 100 에서 기계 축 X1 은 새로운 실제 값 50 을 할당받습니다 . 새로운 실제 값은 다음과 같이 표시됩니다 . - 축 X1, MCS: 50 - 축 A, WCS: 50
N30 A100	; 축 A 가 위치 100 으로 50 mm 이송됩니다 .

5.6 공간의 세 개 측정 지점에서 프레임 계산 (MEAFRAME)

작동

MEAFRAME 은 측정 싸이클을 지원하기 위해 사용되는 840D 언어의 확장입니다 .

함수 MEAFRAME 은 세 개의 이상적인 해당 측정 지점에서 프레임을 계산합니다 .

공작물이 가공을 위한 위치가 지정되면 일반적으로 직교 기계 좌표계 기준 위치가 이상적인 위치를 참조하여 이동 및 회전합니다 . 정확한 가공 또는 측정을 위해 비용이 드는 물리적 부품 조정이 필요하거나 가공 프로그램에 정의된 이동을 변경해야 합니다 .

이상 위치가 알려진 공간에서 세 개 지점을 샘플링하여 프레임을 정의할 수 있습니다 . 터치 프로브 또는 광학 센서는 지지 플레이트 또는 프로브 볼에 고정된 특수 구멍을 정밀하게 접촉하는 샘플링에 사용됩니다 .

구문

MEAFRAME IDEAL_POINT,MEAS_POINT,FIT_QUALITY)

의미

MEAFRAME	공간의 세 개 측정 지점에서 프레임 계산	
IDEAL_POINT	이상 지점의 세 가지 좌표가 포함된 실제 데이터 배열	
MEAS_POINT	측정 지점의 세 가지 좌표가 포함된 실제 데이터 배열	
FIT_QUALITY	REAL 변수 ,	다음 정보 반환 :
	-1:	이상 지점은 거의 직선 위에 있습니다 . 프레임은 계산할 수 없습니다 . 반환된 프레임 변수에는 중립 프레임이 포함됩니다 .
	-2:	측정 지점은 거의 직선 위에 있습니다 . 프레임은 계산할 수 없습니다 . 반환된 프레임 변수에는 중립 프레임이 포함됩니다 .
	-4:	회전 매트릭스 계산은 다른 이유로 실패했습니다 .
	양의 값 :	비틀림 합계 (점 사이 거리) 이며 측정된 삼각형을 이상적인 삼각형에 일치하는 삼각형으로 변환해야 합니다 .

주

측정 품질

측정된 좌표를 회전 및 변환을 사용하여 이상적인 좌표로 매핑하려면 측정 지점에 의해 형성된 삼각형을 이상적인 삼각형과 일치해야 합니다 . 이 작업은 측정된 삼각형을 이상적인 삼각형으로 다시 구성하는 데 필요한 제곱 편차 합계를 최소화하는 보정 알고리즘을 통해 수행됩니다 .

유효 비틀림을 사용하여 측정 품질을 판단할 수 있기 때문에 MEAFRAME을 통해 추가 변수로 반환됩니다 .

주

MEAFRAME 에 의해 생성된 프레임은 SW 6.3 이후부터 ADDFRAME 함수를 프레임 체인의 다른 프레임에 적용하여 변환할 수 있습니다 .
예제 : "ADDFRAME 을 통한 프레임 연결 ".

ADDFRAME(FRAME, STRING) 의 파라미터에 대한 자세한 내용은
/FB1/ 작동 설명서 기본 작동 ; 축 , 좌표계 , 프레임 (K2), " 프레임 연결 " 을 참조하십시오 .

예제

프로그램 코드	비고
	; 가공 프로그램 1
DEF FRAME CORR_FRAME	

측정 지점 설정

프로그래밍	비고
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)	; 테스트용
DEF REAL FIT_QUALITY = 0	
DEF REAL ROT_FRAME_LIMIT = 5	; 부품 위치에 대해 최대 5도 회전이 허용됩니다.
DEF REAL FIT_QUALITY_LIMIT = 3	; 이상 및 측정 삼각형 간에 최대 3mm 옵셋이 허용됩니다 .
DEF REAL SHOW_MCS_POS1[3]	
DEF REAL SHOW_MCS_POS2[3]	
DEF REAL SHOW_MCS_POS3[3]	

프로그램 코드	비고
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	; 첫 번째 RPY 각도 제한
SETAL(65020) GOTOF NO_FRAME ENDIF	
N260 IF CORR_FRAME[Y,RT] > ROT_FRAME_LIMIT	; 두 번째 RPY 각도 제한
SETAL(65021) GOTOF NO_FRAME ENDIF	
N270 IF CORR_FRAME[Z,RT] > ROT_FRAME_LIMIT	; 세 번째 RPY 각도 제한
SETAL(65022) GOTOF NO_FRAME ENDIF	
N300 $P_IFRAME=CORR_FRAME	; 설정 가능한 프레임을 통한 샘플 프레임 활성화
	; 형상 축을 이상적인 지점 위치에 지정하여 프레임 점검
N400 X=IDEAL_POINT[0,0] Y=IDEAL_POINT[0,1] Z=IDEAL_POINT[0,2]	
N410 SHOW_MCS_POS1[0]=$AA_IM[X]	
N410 SHOW_MCS_POS1[1]=$AA_IM[X]	
N430 SHOW_MCS_POS1[2]=$AA_IM[Z]	
N500 X=IDEAL_POINT[1,0] Y=IDEAL_POINT[1,1] Z=IDEAL_POINT[1,2]	
N510 SHOW_MCS_POS2[0]=$AA_IM[X]	
N520 SHOW_MCS_POS2[1]=$AA_IM[Y]	
N530 SHOW_MCS_POS2[2]=$AA_IM[Z]	
N600 X=IDEAL_POINT[2,0] Y=IDEAL_POINT[2,1] Z=IDEAL_POINT[2,2]	
N610 SHOW_MCS_POS3[0]=$AA_IM[X]	
N620 SHOW_MCS_POS3[1]=$AA_IM[Y]	
N630 SHOW_MCS_POS3[2]=$AA_IM[Z]	
N700 G500	; 설정 가능한 프레임을 제로 프레임으로 비활성화합니다 (입력된 값 없음 , 사전 할당).
No_FRAME	; 설정 가능한 프레임을 제로 프레임으로 사전 할당하여 비활성화합니다 (입력된 값 없음).
M0	
M30	

프레임 연결 예

옵셋을 위한 MEAFRAME 연결

MEAFRAME( ) 함수는 옵셋 프레임을 제공합니다 . 이 옵셋 프레임은 예를 들어 G54 와 같은 함수를 호출한 경우 활성화되는 설정 프레임 $P_UIFR[1] 과 연결될 경우 절차 또는 가공에 대한 추가 변환을 위해 설정 가능한 프레임을 받습니다 .

ADDFRAME 을 통한 연결

프레임 체인에서 이 옵셋 프레임을 다른 위치에 적용하거나 다른 프레임을 설정 프레임 이전에 활성화할 경우 ADDFRAME( ) 함수를 사용하여 채널 기본 프레임 중 하나 또는 시스템 프레임에 연결할 수 있습니다 .

다음은 프레임에서 활성화하지 않아야 합니다 .

- MIRROR 를 통한 미러링
- SCALE 을 통한 배율 지정

지령치 및 실제 값에 대한 입력 파라미터는 공작물 좌표계입니다 . 이러한 좌표는 항상 미터법 또는

- 인치 (G71/G70) 및
- 반경 (DIAMOF) 을 기준으로

제어기의 기본 시스템에서 지정해야 합니다 .

5.7 NCU 전역 프레임

작동

각 NCU 의 모든 채널에 대해 한 세트의 NCU 전역 프레임만 사용됩니다 . NCU 전역 프레임을 모든 채널에서 읽고 쓸 수 있습니다 . NCU 전역 프레임은 각 채널에서 활성화됩니다 .

옵셋이 있는 **채널 축 및 기계 축**은 전역 프레임을 통해 배율을 지정하고 미러링할 수 있습니다 .

형상 관계 및 프레임 체인

전역 프레임의 경우 축 간에 형상 관계가 없습니다 . 따라서 회전을 수행하거나 형상 축 식별자를 프로그래밍할 수 없습니다 .

- 전역 프레임에서 회전을 사용할 수 없습니다 . 회전 프로그래밍은 알람 : "18310 채널 %1 블록 %2 프레임 : 회전은 허용되지 않습니다 ." 가 표시되면서 거부됩니다 .
- 전역 프레임 및 채널 관련 프레임을 연결할 수 있습니다. 결과 프레임에는 모든 축의 회전을 포함하여 모든 프레임 구성 요소가 포함되어 있습니다 . 전역 프레임에 회전 구성 요소를 가진 프레임 할당은 알람 " 프레임 : 회전이 허용되지 않습니다 ." 와 함께 거부됩니다 .

NCU 전역 프레임

NCU 전역 기본 프레임 $P_NCBFR[n]

최대 8 개의 NCU 전역 기본 프레임을 구성할 수 있습니다 .

채널 관련 기본 프레임도 사용할 수 있습니다 .

전역 프레임을 NCU의 모든 채널에서 읽고 쓸 수 있습니다. 전역 프레임을 쓸 때 사용자는 채널 코디네이션을 확인해야 합니다 . 이 확인 작업은 예를 들어 대기 마커 (WAITMC) 를 사용하여 수행할 수 있습니다 .

장비 제조업체

전역 기본 프레임 수는 기계 데이터를 사용하여 구성되며
/FB1/ 작동 설명서 기본 작동 ; 축 , 좌표계 , 프레임 (K2) 을 참조하십시오 .

NCU 전역 설정 가능한 프레임 $P_UIFR[n]

모든 설정 가능한 프레임 G500, G54...G599 을 NCU 전역적으로 또는 채널 관련별로 구성할 수 있습니다 .

장비 제조업체

모든 설정 가능한 프레임은 기계 데이터 $MN_MM_NUM_GLOBAL_USER_FRAMES 를 통해 전역 프레임으로 다시 구성할 수 있습니다 .

채널 축 식별자 및 기계 축 식별자는 프레임 프로그램 명령에서 축 식별자를 사용할 수 있습니다 . 형상 식별자의 프로그래밍은 알람과 함께 거부됩니다 .

5.7.1 채널 관련 프레임 ($P_CHBFR, $P_UBFR)

작동

설정 가능한 프레임 또는 기본 프레임은 조작자 작업 또는 PLC 에서 다음을 통해 읽고 쓸 수 있습니다 .

- 가공 프로그램 또는
- 조작자 패널 인터페이스 .

미세조정 옵셋도 전역 프레임에 사용할 수 있습니다 . 채널 관련 프레임의 경우 G53, G153, SUPA 및 G500 을 통해 전역 프레임 억제도 발생합니다 .

장비 제조업체

기본 프레임 수는 MD 28081 MM_NUM_BASE_FRAMES 를 통해 채널에서 구성할 수 있습니다. 표준 구성은 채널당 최소 하나의 기본 프레임에 대해 설계됩니다. 채널당 최대 8개의 기본 프레임이 지원됩니다 . 8 개의 기본 프레임 이외에 채널에 8 개의 NCU 전역 기본 프레임도 추가할 수 있습니다 .

채널 관련 프레임

$P_CHBFR[n]

시스템 변수 $P_CHBFR[n] 를 사용하여 기본 프레임을 읽고 쓸 수 있습니다 . 기본 프레임을 쓰는 경우 연결된 전체 기본 프레임은 G500, G54...G599 명령을 실행할 때까지 활성화되지 않습니다 . 변수는 주로 HMI 또는 PLC 에서 기본 프레임에 대한 쓰기 작업을 저장하는 데 사용됩니다 . 이러한 프레임 변수는 데이터 백업에 의해 저장됩니다 .

채널에서 첫 번째 기본 프레임

필드 장치 0 의 기본 프레임은 미리 정의된 $P_UBFR 변수에 쓸 때 동시에 활성화되지 않고 G500, G54...G599 명령만 실행됩니다 . 또한 프로그램에서 변수를 읽고 쓸 수 있습니다 .

$P_UBFR

$P_UBFR 은 $P_CHBFR[0] 과 동일합니다 . 기본적으로 채널에는 항상 하나의 기본 프레임만 존재하기 때문에 시스템 변수는 이전 버전과 호환됩니다 . 채널 관련 기본 프레임이 없는 경우 읽기 / 쓰기를 수행할 때 " 프레임 : 명령이 허용되지 않습니다 ." 알람이 출력됩니다 .

5.7.2 채널에서 프레임 활성화

작동

채널에서 프레임 활성화는 이러한 프레임과 관련된 시스템 변수를 통해 가공 프로그램에서 전환됩니다. 또한 시스템 프레임도 여기에 속합니다. 현재 시스템 프레임은 가공 프로그램에서 이러한 시스템 변수를 통해 읽고 쓸 수 있습니다 .

채널에서 프레임 활성화

개요

현재 시스템 프레임	다음의 경우 :
$P_PARTFRAME	TCARR 및 PAROT
$P_SETFRAME	PRESET 및 스크래칭
$P_EXTFRAME	외부 워크 옵셋
$P_NCBFRAME[n]	현재 NCU 전역 기본 프레임
$P_CHBFRAME[n]	현재 채널 기본 프레임
$P_BFRAME	채널에서 현재 첫 번째 기본 프레임
$P_ACTBFRAME	전체 기본 프레임
$P_CHBFRMASK 및 $P_NCBFRMASK	전체 기본 프레임
$P_IFRAME	현재 설정 가능한 프레임
현재 시스템 프레임	다음의 경우 :
$P_TOOLFRAME	TOROT 및 TOFRAME
$P_WPFRAME	공작물 원점
$P_TRAFRAME	변환
$P_PFRAME	현재 프로그램 가능한 프레임
현재 시스템 프레임	다음의 경우 :
$P_CYCFRAME	싸이클
P_ACTFRAME	현재 전체 프레임
FRAME 체인	현재 프레임은 전체 기본 프레임으로 구성됩니다 .

$P_NCBFRAME[n] 현재 NCU 전역 기본 프레임

시스템 변수 $P_NCBFRAME[n] 을 사용하여 현재 전역 기본 프레임 필드 요소를 읽고 쓸 수 있습니다 . 결과 전체 기본 프레임은 채널에서 쓰기 프로세스를 통해 계산됩니다 .

수정된 프레임은 프레임을 프로그래밍한 채널에서만 활성화됩니다. NCU 의 모든 채널에 대해 프레임을 수정해야 하는 경우 $P_NCBFR[n] 및 $P_NCBFRAME[n] 을 동시에 써야 합니다 . 다른 채널은 예를 들어 G54 를 통해 프레임을 활성화해야 합니다 . 기본 프레임을 쓸 때마다 전체 기본 프레임이 다시 계산됩니다 .

$P_CHBFRAME[n] 현재 채널 기본 프레임

시스템 변수 $P_CHBFRAME[n] 을 사용하여 현재 채널 기본 프레임 필드 요소를 읽고 쓸 수 있습니다 . 결과 전체 기본 프레임은 쓰기 작업의 결과로 채널에서 계산됩니다 . 기본 프레임을 쓸 때마다 전체 기본 프레임이 다시 계산됩니다 .

$P_BFRAME 채널에서 현재 첫 번째 기본 프레임

미리 정의된 프레임 변수 $P_BFRAME 을 사용하여 가공 프로그램의 채널에서 유효한 필드 장치 0 인 현재 기본 프레임을 읽고 쓸 수 있습니다 . 기록된 기본 프레임이 즉시 계산에 포함됩니다 .

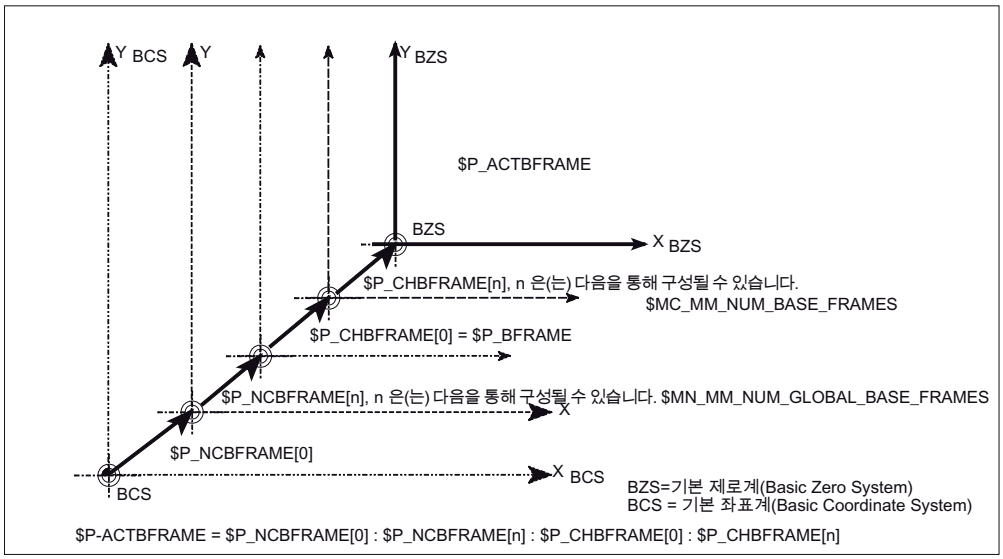
$P_UBFR 은 $P_CHBFR[0] 과 동일합니다 . 시스템 변수에는 항상 유효한 기본값이 포함됩니다 . 채널 관련 기본 프레임이 없는 경우 읽기 / 쓰기를 수행할 때 " 프레임 : 명령이 허용되지 않습니다 ." 알람이 출력됩니다 .

$P_ACTBFRAME 전체 기본 프레임

$P_ACTFRAME 변수는 연결된 전체 기본 프레임을 결정합니다 . 변수는 읽기 전용입니다 .

$P_ACTFRAME 은 다음에 해당합니다 .

$P_NCBFRAME[0] : ... : $P_NCBFRAME[n] : $P_CHBFRAME[0] : ... : $P_CHBFRAME[n].



$P_CHBFRMASK 및 $P_NCBFRMASK 전체 기본 프레임

시스템 변수 $P_CHBFRMASK 및 $P_NCBFRMASK를 사용하여 "전체" 기본 프레임의 계산에 포함될 기본 프레임을 선택할 수 있습니다 . 변수는 프로그램에서만 프로그래밍되고 조작자 패널 인터페이스를 통해 읽을 수 있습니다 . 변수 값은 비트 마스크로 해석되며 계산에 포함될 $P_ACTFRAME 의 기본 프레임 필드 요소를 결정합니다 .

$P_CHBFRMASK 를 사용하여 포함될 채널 관련 기본 프레임을 정의할 수 있으며 $P_NCBFRMASK 를 사용하여 계산에 포함될 NCU 전역 기본 프레임을 정의할 수 있습니다 .

변수가 프로그램되면 전체 기본 프레임 및 전체 프레임이 다시 계산됩니다 . 리셋 이후 기본 설정에서 다음의 값

$P_CHBFRMASK = $MC_CHBFRAME_RESET_MASK 및

$P_NCBFRMASK = $MC_CHBFRAME_RESET_MASK.

예를 들어 ,

$P_NCBFRMASK = 'H81' ;$P_NCBFRAME[0] : $P_NCBFRAME[7]

$P_CHBFRMASK = 'H11' ;$P_CHBFRAME[0] : $P_CHBFRAME[4]

$P_IFRAME 현재 설정 가능한 프레임

미리 정의된 프레임 변수 $P_IFRAME 을 사용하여 가공 프로그램의 채널에서 현재 설정 가능한 프레임을 읽고 쓸 수 있습니다 . 기록된 설정 가능한 프레임이 즉시 계산에 포함됩니다 .

NCU 전역 설정 가능한 프레임의 경우 수정된 프레임만 프레임이 프로그래밍된 채널에서 작동합니다 . NCU 의 모든 채널에 대해 프레임을 수정해야 하는 경우 $P_UIFR[n] 및 $P_IFRAME 을 동시에 써야 합니다 . 다른 채널은 예를 들어 G54 를 통해 해당 프레임을 활성화해야 합니다 .

$P_PFRAME 현재 프로그램 가능한 프레임

$P_PFRAME 은 TRANS/ATRANS, G58/G59, ROT/AROT, SCALE/ASCALE, MIRROR/AMIRROR 의 프로그래밍 또는 CTRANS, CROT, CMIRROR, CSCALE 을 프로그래밍된 프레임에 할당을 통해 프로그래밍된 프레임입니다 .

설정 가능한 다음 사이에서 기준을 구성하는 현재 프로그램 가능한 프레임 변수

- 제로 시스템 (SZS) 및
- 공작물 좌표계 (WCS)

.

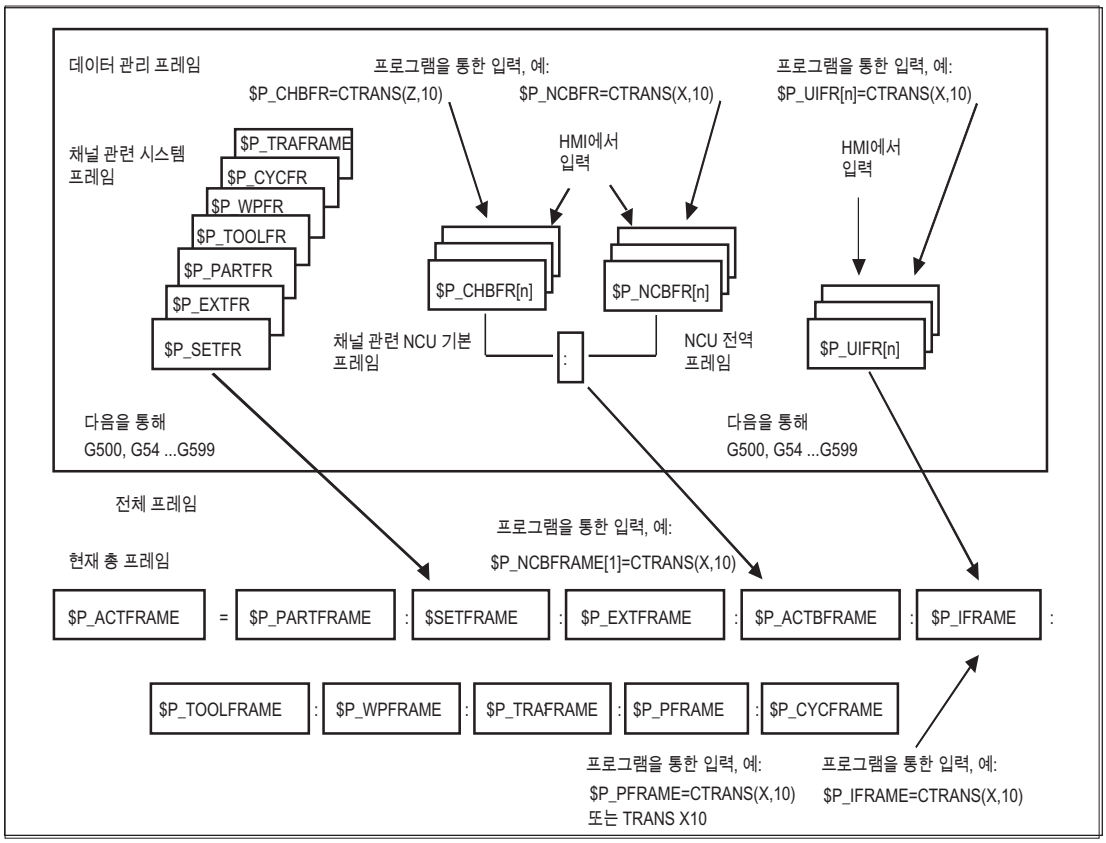
P_ACTFRAME 현재 전체 프레임

결과 현재 전체 프레임 $P_ACTFRAME 은 모든 기본 프레임 , 현재 설정 가능한 프레임 및 프로그램 가능한 프레임의 체인입니다. 프레임 구성 요소가 변경될 때마다 항상 현재 프레임이 업데이트됩니다 .

$P_ACTFRAME 은 다음에 해당합니다 .

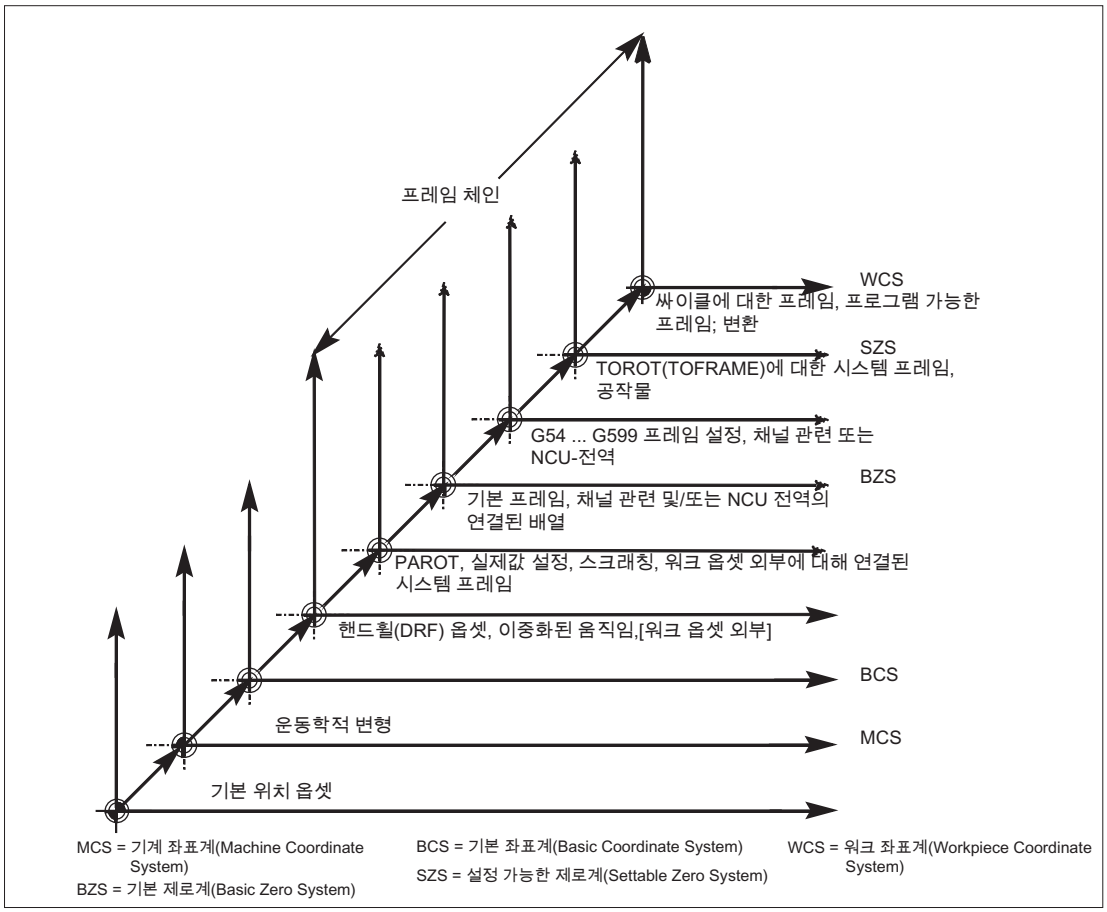
$P_PARTFRAME : $P_SETFRAME : $P_EXTFRAME : $P_ACTBFRAME : $P_IFRAME :

$P_TOOLFRAME : $P_WPFRAME : $P_TRAFRAME : $P_PFRAME : $P_CYCFRAME



FRAME 체인

현재 프레임은 위에서 언급한 현재 전체 프레임에 따라 전체 기본 프레임, 설정 가능한 프레임, 시스템 프레임 및 프로그램 가능한 프레임으로 구성됩니다 .



변환 6

6.1 일반적인 변환 유형 프로그래밍

일반 작동

제어 시스템을 다양한 기계 운동에 적용하기 위해 적절한 파라미터로 변환 유형을 프로그램하도록 선택할 수 있습니다 . 이러한 파라미터를 사용하여 선택한 변환에 따라 공간 상의 공구 방향 및 로타리 축의 방향 이동 모두를 선언할 수 있습니다 .

3 축 , 4 축 및 5 축 변환에서 프로그래밍된 위치 데이터는 항상 공구의 팁과 관련되며 공간 상의 가공된 표면에 대해 직각 방향으로 추적됩니다 . 직교 좌표는 기본 좌표계에서 기계 좌표계로 변환되어 형상 축과 관련됩니다 . 이러한 좌표에서 작동 지점을 설명합니다 . 가상의 로타리 축은 공간 상의 공구 방향을 설명하고 TRAORI 를 통해 프로그래밍됩니다 .

운동학적 변환의 경우 직교 좌표계에서 위치를 프로그래밍할 수 있습니다 . 제어 시스템은 TRANSMIT, TRACYL 및 TRAANG 을 통해 프로그래밍된 직교 좌표계 이송 이동을 실제 기계 축의 이송 이동에 매핑합니다 .

프로그래밍

3 축 , 4 축 및 5 축 변환 (TRAORI)

선언된 방향 변형은 TRAORI 명령과 변환 수 , 방향 벡터 및 로타리 축 옵셋에 대해 가능한 세 가지 파라미터를 통해 활성화됩니다 .

TRAORI(변환 수 , 방향 벡터 , 로타리 축 옵셋)

운동학적 변환

TRANSMIT(변환 수) 선언된 변환은 운동학적 변형의 예입니다 .

TRACYL(작동 직경 , 변환 수)

TRAANG(옵셋 축의 각도 , 변환 수)

활성 변환 비활성화

TRAFOOF 를 사용하여 현재 활성 변환을 비활성화할 수 있습니다 .

방향 변환

3 축 , 4 축 및 5 축 변환 (TRAORI)

기계의 작동 영역에서 공간 상에 구성되는 표면 가공의 최적화를 위해 기계 공구는 세 가지 선형 축 X, Y 및 Z 이외에 다른 축이 필요합니다 . 추가 축은 공간 상의 방향을 설명하며 이후 섹션에서 방향 축이라고 합니다 . 이러한 추가 축은 다양한 운동성을 가진 기계의 네 가지 유형에서 로타리 축으로 사용할 수 있습니다 .

1. 2 축 스위블 헤드 , 예를 들어 고정 공구 테이블에서 선형 축과 평행한 하나의 로타리 축을 가진 cardanic 공구 헤드 .
2. 2 축 로타리 테이블 , 예를 들어 두 개의 축에 대해 회전할 수 있는 공구 테이블을 가진 고정 스위블 헤드 .
3. 단일 축 스위블 헤드 및 단일 축 로타리 테이블 , 예를 들어 공구 테이블에 대해 회전된 공구를 가진 하나의 회전 가능한 스위블 헤드로써 , 하나의 축에 대해 회전 가능합니다 .
4. 2 축 스위블 헤드 및 단일 축 로타리 테이블 , 예를 들어 공구 테이블에서 한 축에 대해 회전 가능하고 회전 가능한 공구를 가진 하나의 회전 가능한 스위블 헤드 .

3 축 및 4 축 변환은 5 축 변환의 특별한 유형이며 5 축 변환과 동일한 방법으로 프로그래밍됩니다 .

"**일반 3-/4-/5-/6-축 변환**"의 기능상 범위는 다른 모든 방향 변환과 같이, 직교 로타리 축에 대한 변환 및 범용 밀링 헤드에 대한 변환 모두에 적합하며 또한 TRAORI 를 통해 이러한 네 가지 기계 유형에 대해 활성화할 수 있습니다 . 일반 5-/6- 축 변환에서 , 공구 방향은 추가로 3 도의 자유도가 있기 때문에 공간 상에서 필요에 따라 방향을 지정할 수 있도록 공구 방향을 기준으로 축에 대해 공구를 회전시킬 수 있습니다 .

참조 문서 : /FB3/ 작동 설명서 , 특별한 기능 ; 3- ~ 5- 축 변환 (F2)

운동성과 관계 없이 초기 공구 방향 설정

ORIRESET

방향 변환이 TRAORI 를 사용하여 활성화된 경우 ORIRESET 을 사용하여 옵션 파라미터 A, B, C 를 사용하여 최대 3 개 방향 축에 대한 초기 설정을 지정할 수 있습니다 . 프로그래밍된 파라미터가 라운드 축에 할당되는 순서는 변환에 의해 정의된 방향 축 순서에 따라 다릅니다 . ORIRESET(A, B, C) 을 프로그래밍하면 방향 축이 선형적이고 동기식으로 현재 위치에서 지정된 초기 설정 위치로 이동합니다 .

운동학적 변환

TRANSMIT 및 TRACYL

선삭 가공 기계에서 밀링하는 경우

1. TRANSMIT 를 통해 선삭 가공 클램프에서 표면 가공 또는
2. TRACYL 을 통해 원통형 본체의 경로를 따라 홈 가공

위의 작업을 선언된 변환에 대해 프로그래밍할 수 있습니다 .

TRAANG

절입 축의 경사 절입 설정이 필요한 경우 (예를 들어 연삭 가공) TRAANG 을 사용하여 선언된 변환에 대한 구성 가능한 각도를 프로그래밍할 수 있습니다 .

직교 PTP 이송

운동학적 변환에는 최대 8 개의 서로 다르게 연결된 조인트 위치 STAT= 를 프로그래밍할 수 있는 소위 " 직교 PTP 이송 " 이 포함되어 있습니다 . 직교 좌표계에서 위치를 프로그래밍하더라도 기계 좌표에서 기계 이동이 발생합니다 .

참조 문서 :
/FB2/ 기능 설명 확장 기능 ; 운동학적 변환 (M1)

연결된 변환

두 개의 변환을 차례대로 전환할 수 있습니다. 여기서 두 번째로 연결된 변환의 경우 축의 이동 부분을 첫 번째 변환에서 가져옵니다 .

첫 번째 변환은 다음과 같습니다 .

- 방향 변환 TRAORI
- 극 변환 TRANSMIT
- 원통 변환 TRACYL
- 경사 축 변환 TRAANG

두 번째 변환은 경사 축에 대해 TRAANG 유형 변환이어야 합니다 .

6.1.1 변환을 위한 방향 이동

이송 및 오리엔테이션 이동

프로그래밍된 오리엔테이션으로 이송은 주로 기계 유형에 의해 결정됩니다 . TRAORI 를 통한 3축, 4축 및 5축 유형 변환의 경우 로터리 축 또는 피봇 직선 축은 공구의 오리엔테이션 이동을 설명합니다 .

오리엔테이션을 변환할 로터리 축의 위치를 변경하면 나머지 기계 축에 대해 이동 보정이 유도됩니다 . 공구 팁의 위치는 고정되어 있습니다 .

공구의 오리엔테이션 이동은 Euler 또는 RPY 각도 , 방향 또는 표면 법선 벡터 , 테이퍼의 로터리 축 또는 테이퍼 주변 표면의 중간 오리엔테이션에 대해 법선 방향 벡터를 입력하여 해당 분야에 적절한 수직 축의 로터리 축 식별자 A..., B..., C... 를 사용하여 프로그래밍할 수 있습니다 .

TRANSMIT, TRACYL 및 TRAANG 을 통한 좌표계 변환의 경우 제어 시스템은 프로그래밍된 직교 좌표계 이송을 실제 기계 축의 이송에 매핑합니다 .

3 축 , 4 축 및 5 축 변환에 대한 기계 운동 (TRAORI)

최대 2 개의 로터리 축을 사용하여 공구 또는 공구 테이블을 회전할 수 있습니다 . 스위블 헤드 및 로터리 테이블 (각각의 경우 단일 축) 조합도 가능합니다 .

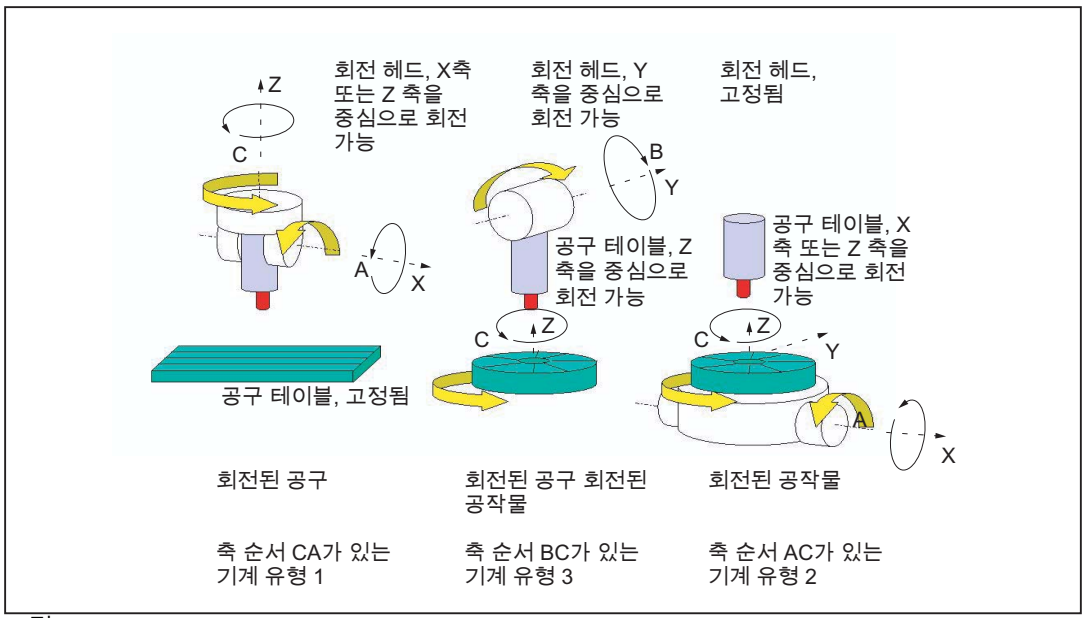


그림 6-1

기계 유형	오리엔테이션 프로그래밍
3 축 변환 기계 유형 1 및 2	로터리 축과 수직 방향인 평면에서만 공구 오리엔테이션 프로그래밍이 가능합니다 . **2 개**의 이송 축 (직선 축) 및 **1 개**의 회전 축 (로터리 축)
4 축 변환 기계 유형 1 및 2	로터리 축과 수직 방향인 평면에서만 공구 오리엔테이션 프로그래밍이 가능합니다 . **3 개**의 이송 축 (직선 축) 및 **1 개**의 회전 축 (로터리 축)
5 축 변환 기계 유형 3 단일 축 스위블 헤드 및 단일 축 로터리 테이블	오리엔테이션 변환 프로그래밍 . **3 개**의 직선 축 및 **2 개**의 직교 로터리 축을 가진 구조 . 로터리 축은 세 개의 직선 축 중 2 개와 평행합니다 . 첫 번째 로터리 축은 2개의 직교 직선 축에 의해 이동됩니다. 세 번째 직선 축은 공구와 함께 회전합니다 . 두 번째 로터리 축은 공작물을 회전합니다 .

일반 5/6 축 변환

기계 유형	오리엔테이션 변환 프로그래밍
일반 5/6 축 변환 기계 유형 4 공구 자체 및 단일 축 로터리 테이블 주위를 회전하는 공구를 가진 2 축 스위블 헤드	오리엔테이션 변환 프로그래밍 . **3 개**의 직선 축 및 **3 개**의 직교 로터리 축을 가진 구조 . 로터리 축은 세 개의 직선 축 중 2 개와 평행합니다 . 첫 번째 로터리 축은 2 개의 직교 직선 축에 의해 이동됩니다 . 세 번째 직선 축은 공구와 함께 회전합니다 . 두 번째 로터리 축은 공작물을 회전합니다 . 또한 기본 공구 오리엔테이션도 THETA 로터리 각도로 공구 자체 주위를 회전하는 추가 공구 회전을 통해 프로그래밍할 수 있습니다 .

" 일반 3 축 , 4 축 및 5/6 축 변환 " 을 호출하면 공구의 기본 오리엔테이션도 전달할 수 있습니다 . 로터리 축 방향 측면에서 제한은 더 이상 적용되지 않습니다 . 로터리 축이 서로 간에 정확하게 수직을 이루지 않거나 기존 로터리 축이 직선 축과 정확하게 평행하지 않을 경우 " 일반 5/6 축 변환 " 이 공구 오리엔테이션 측면에서 최상의 결과를 제공할 수 있습니다 .

좌표계 변환 TRANSMIT, TRACYL 및 TRAANG

선반에서 밀링 또는 연삭 가공 동안 경사 절입을 위해 설정 가능한 축의 경우 기본적으로 다음 축 정렬이 선언된 변환에 따라 적용됩니다 .

TRANSMIT	극점 변환 활성화
선삭 가공 클램프에서 표면 가공	로터리 축 회전 축과 수직인 절입 축 회전 축과 평행한 수평 축

TRACYL	원통 표면 변환 활성화
원통형 본체의 경로를 따라 홈 가공	로터리 축 회전 축과 수직인 절입 축 회전 축과 평행한 수평 축

TRAANG	절입 축 변환 활성화
사면 절입 축을 사용하여 가공	로터리 축 각도 설정이 가능한 절입 축 회전 축과 평행한 수평 축

직교 PTP 이송

기계가 기계 좌표에서 이동하고 다음을 통해 프로그래밍됩니다 :

TRAORI	변환 활성화
PTP 포인트 투 포인트 이동	직교 좌표계 (MCS) 로 위치 이동
CP	(BCS) 에서 직교 축의 경로 이동
STAT	연결 조인트의 위치는 변환에 종속됩니다 .
TU	축이 최단 경로를 통해 이송되는 각도

일반 5/6 축 변환을 통한 PTP 이송

기계가 기계 좌표 및 공구 오리엔테이션을 사용하여 이동하고 여기서 이동은 운동 또는 방향 벡터와 관계 없이 라운드 축 위치 및 Euler 및 / 또는 RPY 각도 벡터를 사용하여 프로그래밍할 수 있습니다 .

이러한 경우 라운드 축 보간 , 대형 원 보간을 통한 벡터 보간 또는 테이퍼의 주변 표면에 대한 오리엔테이션 벡터 보간을 수행할 수 있습니다 .

예제 : 범용 밀링 헤드에 대한 3 축 - 5 축 변환

최소 5 개의 축으로 구성된 기계 :

- 작업 영역에서 현재 위치에서 임의의 위치로 3 개의 이송 축이 직선으로 이동합니다 .
- 설정 가능한 각도 (일반적으로 45 도) 로 정렬된 2 개의 로터리 스위블 축을 통해 공구는 45 도로 설정된 공간 상의 위치로 회전할 수 있습니다 .

6.1.2 방향 변환 개요 TRAORI

TRAORI 와 함께 사용 가능한 프로그래밍 유형

기계 유형	활성 변환 TRAORI 를 통한 프로그래밍
기계 유형 1, 2 또는 3 2 축 스위블 헤드 또는 2 축 로타리 테이블 또는 단일 축 스위블 헤드와 단일 축 로타리 테이블의 조합 .	방향 축의 축 순서 및 공구의 방향은 기계 운동 또는 기계 운동과 관계 없이 프로그램 가능한 방향으로 **공작물 관련** 기준에 따라 기계 데이터를 사용하여 **기계 관련** 기준으로 구성할 수 있습니다 . 기준 시스템의 방향 축에 대한 회전 방향은 다음을 통해 프로그래밍됩니다 : - ORIMKS 기준 시스템 = 기계 좌표계 - ORIWKS 기준 시스템 = 공작물 좌표계 기본 설정은 ORIWKS 입니다 . 다음을 사용하여 방향 축 프로그래밍 : 기계 축 위치 A, B, C 는 Euler 각도 (표준) 를 통해 - ORIEULER 에 의해 A2, B2, C2 각도 프로그래밍 가상 축을 향함 . - RPY 각도를 통한 ORIRPY, - 가상 방향 축을 통한 ORIVIRT1, 첫 번째 정의 . - 가상 방향 축을 통한 ORIVIRT2, 두 번째 정의 . 보간 유형 간 차이점 : **선형 보간** - 방향 축 또는 기계 축의 ORIAXES **대형 반경 원 보간** (방향 벡터의 보간) - 방향 축에서 ORIVECT, 벡터 구성 요소 (방향 / 표면 법선) 의 A3, B3, C3 을 지정하여 방향 축 프로그래밍 , 결과 공구 방향 프로그램 , 블록 시작부에서 벡터 표면 법선의 A4, B4, C4, 블록 종단부에서 표면에 수직인 벡터의 A5, B5, C5, 공구 방향에 대한 LEAD 선행 각도 , 공구 방향에 대한 TILT 경사 각도
	테이퍼 주변 표면에 대한 **방향 벡터의 보간** , 보간을 사용하여 **공간 상의 모든 위치에서** 테이퍼 주변 표면에 대한 방향 변경 : - 평면에서 ORIPLANE(대형 반경 원 보간) - 시계 방향으로 테이퍼 주변 표면에 대한 ORICONCW, - 시계 반대 방향으로 테이퍼 주변 표면에 대한 ORICONCCW, A6, B6, C6 방향 벡터 (테이퍼 회전 축) - 다음을 사용하여 테이퍼 주변 표면에 대한 OICONIO 보간 : A7, B7, C7 중간 벡터 (초기 및 최종 방향) 또는 - 테이퍼의 주변 표면에 대한 ORICONTO, 법선 방향 변환 다음을 통해 **경과** 관련 방향 변경 - 최대 5 차의 PO[XH]=(xe, x2, x3, x4, x5) 방향 다항식을 사용하여 2 개의 접점 이동에 대한 ORICURVE 지정 최대 5 차의 PO[YH]=(ye, y2, y3, y4, y5) 방향 다항식 최대 5 차의 PO[ZH]=(ze, z2, z3, z4, z5) 방향 다항식 - 다음에 해당하는 공구의 A8, B8, C8 방향 재지정 단계를 통한 방향 특성의 ORIPATHS 유연화 : 후퇴 이동 동안 공구의 방향 및 경로 길이

기계 유형	활성 변환 TRAORI 를 통한 프로그래밍
기계 유형 1 및 3 공구 자체 주위를 추가로 회전하는 공구의 기타 기계 유형에는 세 번째 로타리 축이 필요합니다 . 예를 들어 일반 6 축 변환과 같은 방향 변환 . 방향 벡터의 회전 .	LEAD 각도를 가진 공구 방향에 대한 **회전** 프로그래밍 , 각도는 표면 법선 벡터 기준 최대 5 차 다항식의 PO[PHI] 프로그래밍 , 경로 법선 (Z 방향) 에 대한 TILT 각도 회전 , 최대 5 차 다항식의 PO[PSI] 프로그래밍 , 회전 THETA 각도 (Z 에서 공구 방향에 대한 회전) THETA= 블록 끝에 도달한 값 THETA=AC(...) 차원에 대한 절대 비모달식 전환 THETA=IC(...) 연결 차원에 대한 비모달식 전환 , THETA=Θ_e 는 프로그래밍된 각도 G90/G91 보간 , PO[THT]=(..) 최대 5 차 다항식의 프로그래밍 , 회전 벡터 프로그래밍 - ORIROTA 회전 , 절대값 - ORIROTR 상대 회전 벡터 - ORIROTT 법선 회전 벡터
방향 경로 기준 방향은 경로에 대해 법선 방향인 로타리 벡터의 경로 또는 회전을 기준으로 변경됩니다 .	다음을 사용하여 **경로 기준** 방향 변경 - ORIPATH 경로 기준 공구 방향 - ORIPATHS 또한 회전 벡터의 방향 특성 프로그래밍에 blip 가 있는 경우 - ORIROTC 법선 회전 벡터 , 경로 법선 방향 회전

6.2 3 축 , 4 축 및 5 축 변환 (TRAORI)

6.2.1 유니버셜 공구 헤드의 일반적인 관계

작동

3D 곡면을 가공할 때 최적의 절삭 조건을 구현하기 위해서는 공구의 설정 각도를 다양하게 바꿀 수 있어야 합니다 .

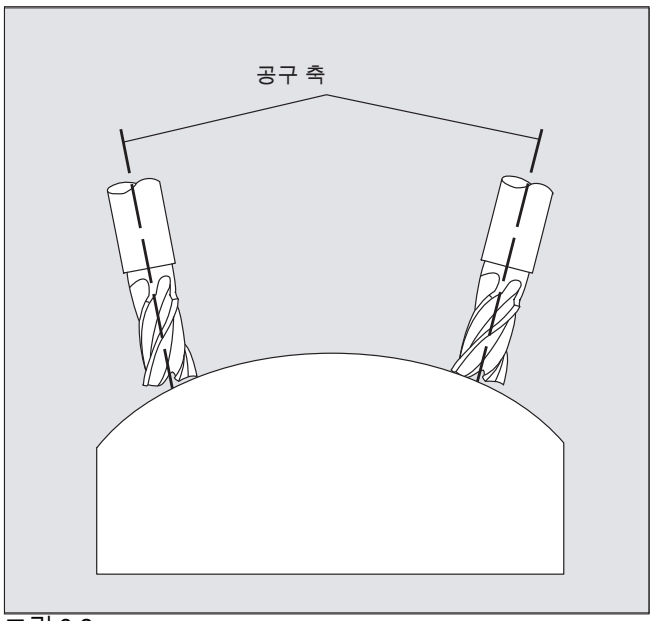


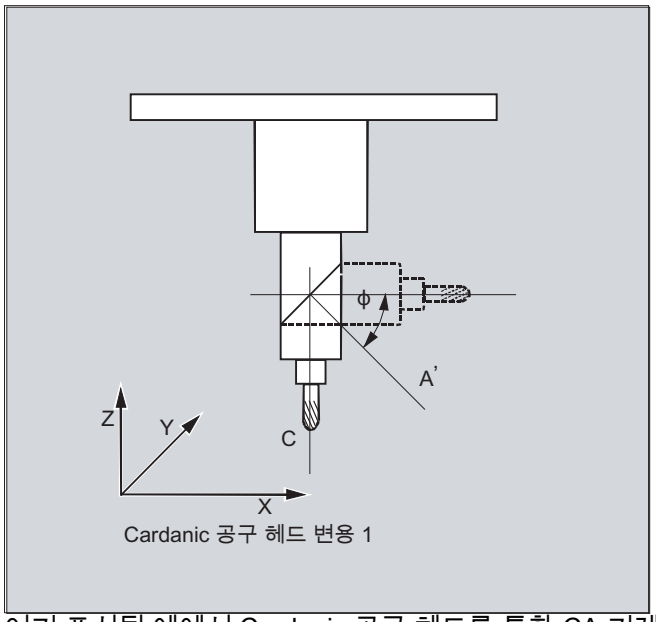
그림 6-2

이를 실행하기 위한 기계 설계가 축 데이터에 저장됩니다 .

5 축 변환

Cardanic 공구 헤드

세 개의 선형 축 (X, Y, Z) 및 두 개의 방향 축 (C, A) 이 여기서 공구의 설정 각도 및 작동 지점을 정의합니다 . 두 개의 방향 축 중 하나는 예제 A 처럼 경사축 (대부분의 경우 45°) 으로 생성됩니다 .

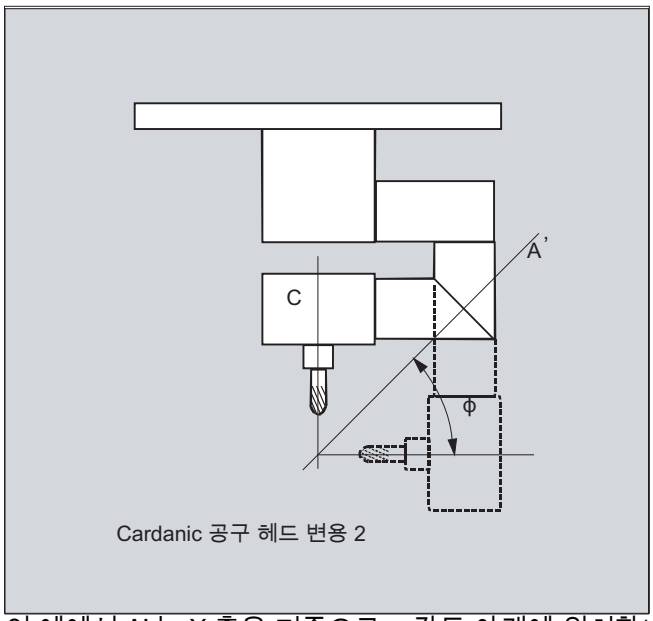


Cardanic 공구 헤드 변용 1

여기 표시된 예에서 Cardanic 공구 헤드를 통한 CA 기계 운동을 그림으로 표현한 정렬을 확인할 수 있습니다 .

장비 제조업체

방향 축의 축 순서 및 공구의 방향은 기계 운동에 따라 기계 데이터를 사용하여 구성할 수 있습니다 .



Cardanic 공구 헤드 변용 2

이 예에서 A' 는 X 축을 기준으로 φ 각도 아래에 위치합니다 .

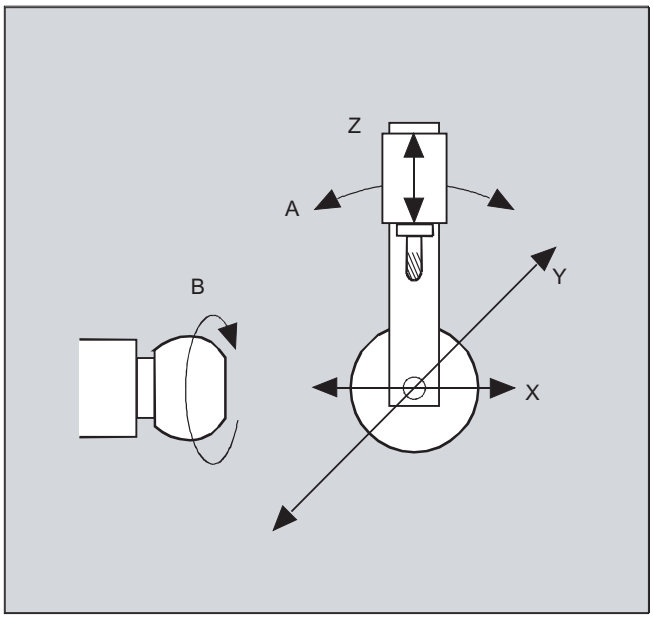
일반적으로 다음과 같은 가능한 관계가 적용됩니다 :

A' 는 다음 축을 기준으로 φ 각도 아래 위치	X 축
B' 는 다음 축을 기준으로 φ 각도 아래 위치	Y 축
C' 는 다음 축을 기준으로 φ 각도 아래 위치	Z 축

φ 각도는 기계 데이터를 사용하여 0° ~ +89° 범위에서 구성할 수 있습니다 .

선형 축 회전

이 동작은 이동하는 공작물 및 이동하는 공구를 정렬하는 것입니다 . 운동 구조는 세 개의 선형 축 (X, Y, Z) 및 두 개의 직교 방향으로 정렬된 로타리 축으로 구성됩니다 . 예를 들어 첫 번째 로타리 축은 두 개의 선형 축의 복합 슬라이드 위에서 이동하고 공구는 세 번째 선형 축에 평행하게 위치합니다. 두 번째 로타리 축은 공작물을 회전합니다. 세 번째 선형 축(회전 축)은 복합 슬라이드 평면에 위치합니다 .



로타리 축의 축 순서 및 공구의 방향은 기계 운동에 따라 기계 데이터를 사용하여 구성할 수 있습니다 .

다음 관계가 가능합니다 :

축 :	**축 순서 :**
1. 로타리 축	A A B B C C
2. 로타리 축	B C A C A B
회전한 선형 축	Z Y Z X Y X

공구 방향에 대해 구성 가능한 축 순서에 대한 자세한 내용은 다음을 참조하십시오 .

참고 문서 :/FB3/ 작동 설명서 , 특별한 기능 ; 3- ~ 5- 축 변환 (F2), 유니버셜 밀링 헤드 섹션 , " 파라미터 설정 ".

6.2.2 3 축 , 4 축 및 5 축 변환 (TRAORI)

기능

사용자가 2개 또는 3개의 변환 축과 하나의 로터리 축을 구성할 수 있습니다. 변환에서 로터리 축은 오리엔테이션 평면에 대해 수직인 것으로 간주합니다 .

공구의 오리엔테이션은 로터리 축과 평행인 평면에서만 가능합니다. 변환은 이동 가능한 공구 및 이동 가능한 공작물을 가진 기계 유형을 지원합니다 .

3 축 및 4 축 변환은 5 축 변환과 동일한 방법으로 구성 및 프로그래밍됩니다 .

참고 자료 :
기능 매뉴얼 , 특수 기능 : 다중 축 변환 (F2)

구문

TRAORI(<n>)

TRAORI(<n>,<X>,<Y>,<Z>,<A>,<B>)

TRAFOOF

의미

TRAORI:	첫 번째 지정된 오리엔테이션 변환을 실행합니다 .
TRAORI(<n>):	n 에 의해 지정된 오리엔테이션 변환을 실행합니다 .
<n>:	변환의 번호 값 : 1 또는 2 예제 : TRAORI(1) 이 오리엔테이션 변환 1 을 활성화합니다 .
<X>,<Y>,<Z>:	공구가 향하고 있는 오리엔테이션 벡터의 구성 요소
<A>,<B>:	로터리 축의 프로그램 옵셋
TRAFOOF:	변환 비활성화

공구 오리엔테이션

공구에 대해 선택한 오리엔테이션에 따라 공구 길이 옵셋이 공구 오리엔테이션에서 작동하도록 활성 작업 평면 (G17, G18, G19) 을 NC 프로그램에서 설정해야 합니다 .

주

변환이 동작되면 위치 데이터 (X, Y, Z)는 항상 공구의 팁을 기준으로 합니다. 변환에서 로터리 축의 위치를 변경하면 나머지 기계 축의 동작이 보정됩니다 . 즉 , 공구 팁의 위치는 변경되지 않습니다 .

오리엔테이션 변환은 항상 공구 팁에서 공구 어댑터를 향합니다 .

오리엔테이션 축 옵셋

오리엔테이션 변환이 활성화되면 방향 축에 대해 직접 추가 옵셋을 프로그래밍할 수 있습니다.

정확한 순서대로 프로그래밍할 경우 파라미터를 생략할 수 있습니다 .

예제 :

TRAORI(, , , ,A,B) ; 옵셋을 하나만 입력하는 경우

직접 프로그래밍에 대한 대안으로 오리엔테이션 축의 추가 옵셋도 현재 활성 워크 옵셋에서 자동으로 전송할 수 있습니다 . 이송은 머신 데이터에서 구성됩니다 .

예제

```
TRAORI(1,0,0,1)      ; 공구의 기본 오리엔테이션은 Z 방향입니다 .

TRAORI(1,0,1,0)      ; 공구의 기본 오리엔테이션은 Z 방향입니다 .

TRAORI(1,0,1,1)      ; 공구의 기본 오리엔테이션은 Y/Z 방향입니다 (-45° 에 해당 ).
```

6.2.3 오리엔테이션 프로그래밍 변수 및 초기 설정 (ORIRESET)

TRAORI 를 통한 공구 오리엔테이션 프로그래밍

프로그램 가능한 TRAORI 오리엔테이션 변환과 연계하여 직선 축 X, Y, Z뿐만 아니라 A..., B..., C... 축을 사용하여 축 위치를 프로그래밍하거나 각도나 벡터 구성 요소를 가진 가상 축을 프로그래밍할 수 있습니다 . 오리엔테이션 및 기계 축에 대해 다양한 보간 유형이 가능합니다 . PO[angle] 오리엔테이션 폴리노미널 및 PO[axis] 축 폴리노미널이 현재 활성 상태인지 여부와 관계 없이 다양한 유형의 폴리노미널을 프로그래밍할 수 있습니다. 여기에는 G1, G2, G3, CIP 또는 POLY 가 포함됩니다 .

일부 경우 공구 오리엔테이션 변경은 오리엔테이션 벡터를 사용하여 프로그래밍할 수도 있습니다 . 이러한 경우 각 블록의 최종 오리엔테이션은 벡터에 대한 직접 프로그래밍 또는 로터리 축 위치 프로그래밍을 통해 설정할 수 있습니다 .

주

3 축 - 5 축 변환에 대한 오리엔테이션 프로그래밍 변수

3 축 - 5 축 변환과 관련하여 다음과 같이 변환할 수 있습니다 :

1. A, B, C 에 기계 축 좌표 직접 입력
2. Euler 각도 또는 RPY 각도를 사용하여 가상 축에 대해 A2, B2, C2 각도 프로그램
3. 벡터 구성 요소에 대해 A3 ,B3, C3 입력
4. 경로 및 표면을 기준으로 리드 및 틸트 각도에 대해 LEAD, TILT 입력
5. 블록 시작 및 블록 끝에서 A4, B4, C4 and A5, B5, C5 표면 법선 벡터
6. 테이퍼 주변 표면에 대한 오리엔테이션 벡터의 A6, B6, C6 및 A7, B7, C7 보간
7. 후퇴 이동을 위한 공구 , 방향 및 경로 길이에 대한 A8, B8, C8 오리엔테이션 재지정

위의 변용은 상호 배타적입니다 .

값을 혼합하여 프로그래밍할 경우 알람 메시지가 출력됩니다 .

초기 공구 오리엔테이션 설정 ORIRESET

ORIRESET (A, B, C) 을 프로그래밍하면 오리엔테이션 축이 직선적이고 동기식으로 현재 위치에서 지정된 초기 설정 위치로 이동합니다 .

축에 대해 초기 설정 위치가 프로그래밍되지 않을 경우 관련 머신 데이터 $MC_TRAFO5_ROT_AX_OFFSET_1/2 에 정의된 위치가 사용됩니다 . 존재할 수 있는 라운드 축의 활성 프레임은 무시됩니다 .

주

오리엔테이션 변환이 TRAORI(...) 를 통해 활성화된 경우에만 운동 구조에 관계 없이 공구 방향에 대한 초기 설정을 알람 14101 없이 ORIRESET(...) 을 사용하여 프로그래밍할 수 있습니다 .

예제

```
1. 기계 운동 CA ( 채널 축 이름 C, A) 의 예제
ORIRESET(90, 45)        ;C 축 = 90 도 , A 축 = 45 도
ORIRESET(, 30)          ;C 축 = $MC_TRAFO5_ROT_AX_OFFSET_1/2[0], A 축 = 30 도
ORIRESET( )             C 축 = $MC_TRAFO5_ROT_AX_OFFSET_1/2[0],
                        A 축 = $MC_TRAFO5_ROT_AX_OFFSET_1/2[1]
2. 기계 운동 CAC ( 채널 축 이름 C, A, B) 의 예제
ORIRESET(90, 45, 90)    C 축 = 90 도 , A 축 = 45 도 , B 축 = 90 도
ORIRESET( )             C 축 = $MC_TRAFO5_ROT_AX_OFFSET_1/2[0],
                        A 축 = $MC_TRAFO5_ROT_AX_OFFSET_1/2[1],
                        B 축 = $MC_TRAFO5_ROT_AX_OFFSET_1/2[2]
```

LEAD, TILT 및 THETA 회전 프로그램

3 축 ~ 5 축 변환과 관련하여 공구 오리엔테이션 회전이 LEAD 및 TILT 각도를 통해 프로그래밍됩니다 .

세 번째 로터리 축의 변환과 관련하여 C2 에 대한 추가 프로그램 설정 (오리엔테이션 벡터의 회전) 은 벡터 구성 요소 및 LEAD, TILT 각도의 항목을 가진 두 방향 모두에 대해 허용됩니다 .

추가 세 번째 로터리 축에서 공구의 자체 회전은 THETA 로터리 각도를 통해 프로그래밍할 수 있습니다 .

6.2.4 공구 방향 프로그램 (A..., B..., C..., LEAD, TILT)

기능

다음 옵션은 공구 오리엔테이션 프로그래밍하는 경우 사용할 수 있습니다 :

1. 로터리 축의 이동 직접 프로그래밍 . 오리엔테이션 변경은 항상 베이직 또는 기계 좌표계에서 수행됩니다 . 오리엔테이션 축은 동기화된 축으로 이송됩니다 .
2. A2, B2, C2 를 사용하여 각도 정의에 따라 Euler 또는 RPY 각도에서 프로그래밍
3. A3, B3, C3 을 사용하여 방향 벡터의 프로그래밍 방향 벡터의 방향은 공구 팁에서 공구 어댑터를 향합니다 .
4. A4, B4, C4 를 통해 블록 시작 및 A5, B5, C5 (페이스 밀링) 를 통해 블록 끝에서 표면 법선 벡터 프로그래밍 .
5. 리드 각도 LEAD 및 틸트 각도 TILT 를 사용한 프로그래밍

6. A6, B6, C6을 사용한 법선 벡터 또는 A7, B7, C7을 사용한 테이퍼의 주변 표면에 대한 중간 오리엔테이션으로 테이퍼의 로터리 축 프로그래밍
   " 테이퍼의 주변 표면을 따라 오리엔테이션 프로그래밍 (ORIPLANE, ORICONxx)" 을 참조하십시오 .
7. A8, B8, C8 을 사용하여 후퇴 이동 동안 오리엔테이션 재지정 , 방향 및 경로 길이 프로그래밍
   " 오리엔테이션 특성 스무딩 (ORIPATHS A8=, B8=, C8=)" 을 참조하십시오 .

주

모든 경우 오리엔테이션 프로그램은 오리엔테이션 변환을 수행이 동작중인 경우에 한해서 허용됩니다 .

이점 : 어떤 형태의 기계라도 이 프로그램을 전송할 수 있습니다 .

G 코드를 통한 공구 오리엔테이션 정의

주

장비 제조업체

머신 데이터를 사용하여 Euler 또는 RPY 각도 간에 전환할 수 있습니다 . 머신 데이터를 설정한 경우 그룹 50 의 활성 G 코드에 따라 또는 이 코드와 관계 없이 전환할 수 있습니다 . 다음과 같은 설정 옵션을 선택할 수 있습니다 .

1. 오리엔테이션 축 정의 및 오리엔테이션 각도 정의에 대한 머신 데이터가 G 코드를 통해 0 으로 설정된 경우 :
   A2, B2, C2를 사용하여 프로그래밍된 각도는 **머신 데이터에 따라 다릅니다**. 오리엔테이션 프로그래밍의 각도 정의는 Euler 또는 RPY 각도로 해석됩니다 .
2. 오리엔테이션 축 정의를 위한 머신 데이터가 G 코드를 통해 1 로 설정되면 전환은 그룹 50 의 활성 G 코드에
   종속됩니다 .
   A2, B2, C2 를 사용하여 프로그래밍된 각도는 활성 G 코드 ORIEULER, ORIRPY, ORIVIRT1, ORIVIRT2, ORIAXPOS 및 ORIPY2 에 따라 해석됩니다 . 또한 오리엔테이션 축으로 프로그래밍된 값은 그룹 50 의 활성 G 코드에 따라 오리엔테이션 각도로 해석됩니다 .
3. 오리엔테이션 각도 정의를 위한 머신 데이터가 G 코드를 통해 1로 설정되고 오리엔테이션 축 정의를 머신 데이터가 G 코드를 통해 0 으로 설정되면 전환은 그룹 50 의 활성 G 코드에 **종속되지 않습니다** .
   A2, B2, C2 를 사용하여 프로그래밍된 각도는 활성 G 코드 ORIEULER, ORIRPY, ORIVIRT1, ORIVIRT2 ORIAXPOS 및 ORIPY2 중 하나에 따라 해석됩니다 . 오리엔테이션 축으로 프로그래밍된 값은 그룹 50 의 활성 G 코드와 관계 없이 항상 라운드 축으로 해석됩니다 .

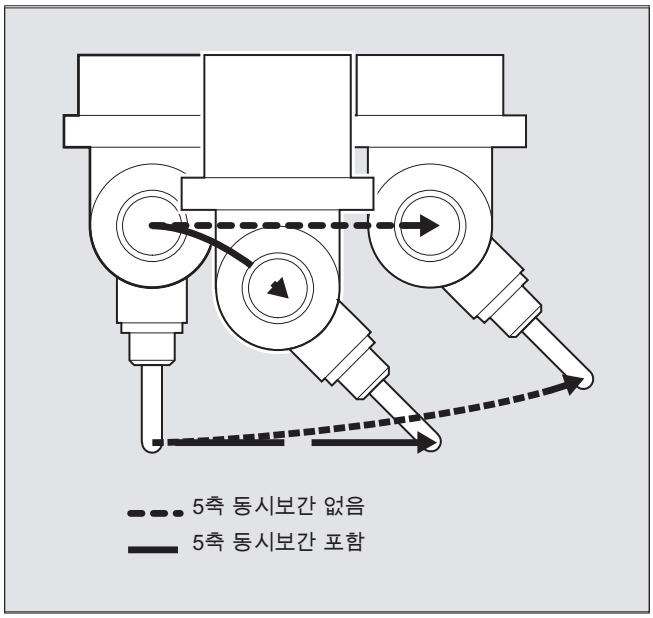
프로그래밍

G1 X Y Z A B C	로터리 축 이동에 대한 프로그래밍
G1 X Y Z A2= B2= C2=	Euler 각도 방식 프로그램
G1 X Y Z A3== B3== C3==	방향 벡터 프로그램
G1 X Y Z A4== B4== C4==	블록 시작에서 표면 법선 벡터 프로그래밍
G1 X Y Z A5== B5== C5==	블록 끝에서 표면 법선 벡터 프로그래밍
LEAD=	공구 오리엔테이션 프로그램에 사용할 리드 각도
TILT=	공구 오리엔테이션 프로그램에 사용할 틸트 각도

파라미터

G....	로터리 축 이동 세부 정보
X Y Z	직선 축 세부 정보
A B C	로터리 축의 기계 축 위치 세부 정보
A2 B2 C2	가상 축 또는 오리엔테이션 축의 각도 프로그래밍 (Euler 또는 RPY 각도)
A3 B3 C3	방향 벡터 구성 요소 세부 정보
A4 B4 C4	예를 들어 페이스 밀링에 대한 세부 정보 , 블록 시작에서 표면 법선 벡터의 구성 요소
A5 B5 C5	예를 들어 페이스 밀링에 대한 세부 정보 , 블록 끝에서 표면 법선 벡터의 구성 요소
LEAD	평면에서 표면 법선 벡터 관련 각도는 경로 접선 및 표면 법선 벡터에 의해 입력됩니다 .
TILT	평면에서의 각도는 표면 법선 벡터를 기준으로 경로 접선에 대해 수직입니다 .

예제 : 5 축 변환 적용 및 미적용 시 비교



설명

일반적으로 5 축 프로그램은 CAD/CAM 시스템에 의해 생성되고 제어 시스템에서 입력되지 않습니다 . 따라서 사후 프로세서의 프로그램에서 주로 다음 설명을 따릅니다 .

방향 프로그램의 유형은 그룹 50 의 G 코드에서 정의됩니다 :

Euler 각도를 통한 ORIEULER
RPY 각도를 통한 ORIRPY(회전 순서 ZYX)
가상 방향 축을 통한 ORIVIRT1(정의 1)
가상 방향 축을 통한 ORIVIRT2(정의 2)
라운드 축 위치를 가진 가상 방향 축을 통한 ORIAXPOS
RPY 각도를 통한 ORIPY2(방향 순서 XYZ)

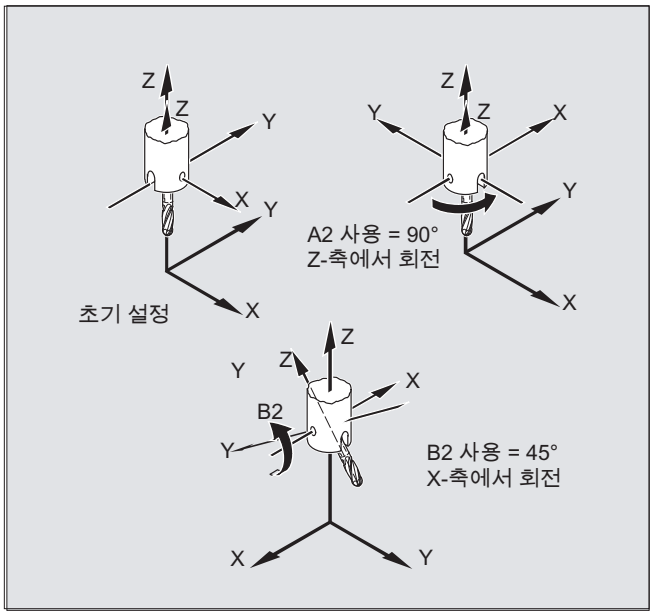
장비 제조업체

장비 제조업체는 기계 데이터를 사용하여 다양한 변용을 정의할 수 있습니다. 장비 제조업체의 설명서를 참조하십시오 .

Euler 각도 방식 프로그램 ORIEULER

A2, B2, C2 를 통해 방향 프로그래밍하는 동안 프로그래밍된 값은 Euler 각도 (도) 방식으로 해석됩니다 .

방향 벡터는 처음에 Z 축 중심으로 A2 를 사용하고 , 다음에 새로운 Y 축을 기준으로 B2 를 사용하고 , 마지막으로 새로운 Z 축을 기준으로 C2 를 사용하여 각 축 방향에서 벡터 회전이 생성됩니다 .



이 경우 C2 의 값 (새 Z 축 중심 회선) 은 의미 없으며 프로그래밍할 필요가 없습니다 .

RPY 각도 방식 프로그램 ORIRPY

RPY 각도 방식으로 방향 프로그램에서 A2, B2, C2 값이 프로그램되었습니다 .

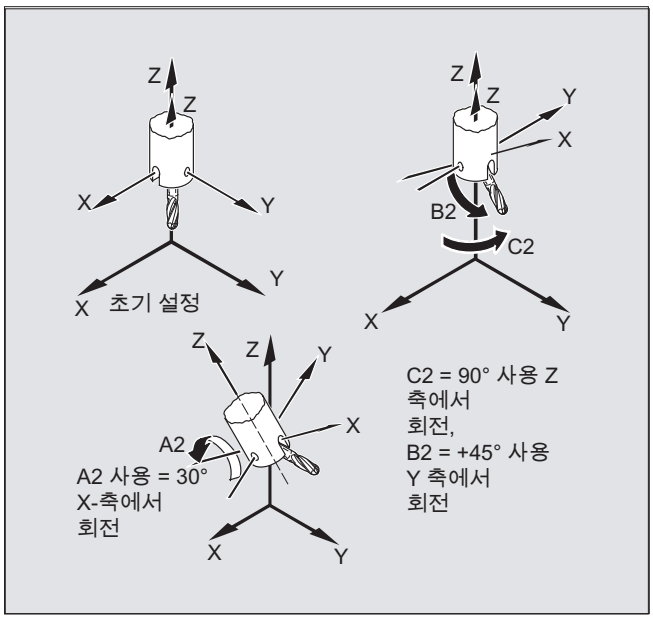
주

Euler 각도 방식의 프로그램과 대조적으로 여기에 사용된 세 값은 모두 방향 벡터에 영향을 미칩니다 .

장비 제조업체

방향 축 $MC_ORI_DEF_WITH_G_CODE = 0 에 대해 RPY 각도를 통해 각도를 방향 각도로 정의하는 경우

방향 벡터는 처음에 Z 축을 중심으로 C2 를 사용하고 , 다음에 새로운 Y 축을 기준으로 B2 를 사용하고 , 마지막으로 새로운 X 축을 기준으로 A2 를 사용하여 각 축 방향에서 벡터 회전이 생성됩니다 .

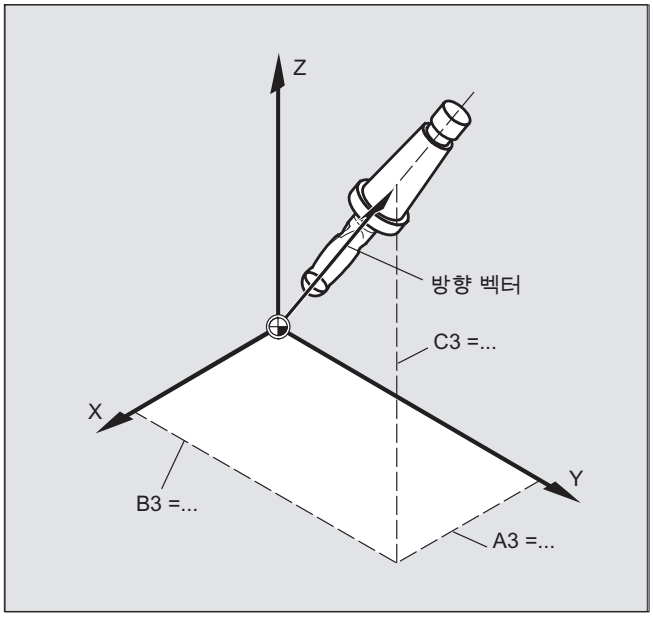


G 코드를 통해 방향 축을 정의하고 기계 데이터 $MC_ORI_DEF_WITH_G_CODE = 1 인 경우 : 방향 벡터는 처음에 Z 축을 중심으로 A2 를 사용하고 , 다음에 새로운 X 축을 기준으로 B2 를 사용하고 , 마지막으로 새로운 Z 축을 기준으로 C2 를 사용하여 각 축 방향에서 벡터 회전이 생성됩니다 .

방향 벡터 프로그램

방향 벡터의 구성 요소는 A3, B3, C3 를 사용하여 프로그램합니다 . 벡터는 공구 어댑터 방향을 향하고 벡터 길이는 중요하지 않습니다 .

프로그램되지 않은 벡터 구성 요소는 '0' 으로 설정됩니다 .



LEAD= 및 TILT= 를 통해 공구 방향 프로그램

다음으로부터 공구 방향이 결정됩니다 :

- 경로 접선
- 블록 A4, B4, C4 의 시작 및
  블록 A5, B6, C5 의 끝에서 표면 법선 벡터
- 경로 접선 및 표면 법선 벡터에 의해 정의된 평면에서 리드 각도 LEAD
- 경로 접선과 수직이고 표면 법선 벡터를 기준으로 블록의 끝에서
  틸트 각도 TILT

내부 모서리에서 동작 (3D 공구 보정의 경우)

블록이 내부 모서리에서 짧아지는 경우 결과 공구 방향도 블록 끝에서 적용됩니다 .

LEAD= 및 TILT= 를 통해 공구 방향 정의

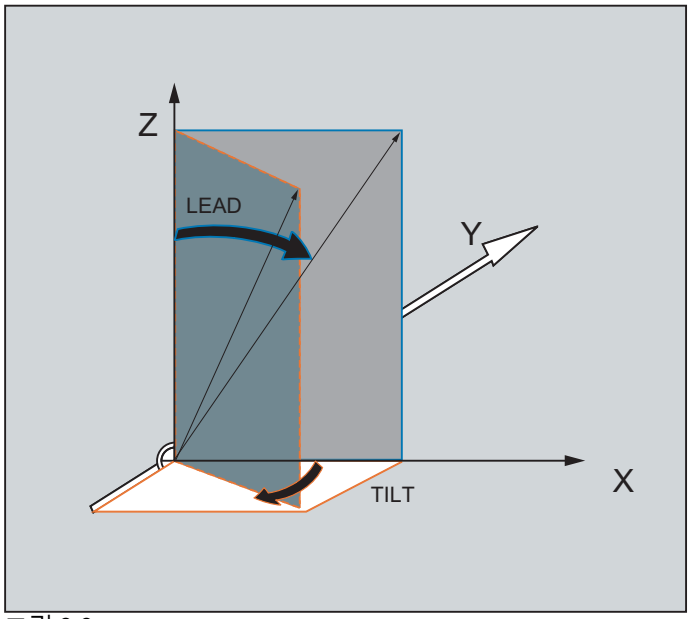
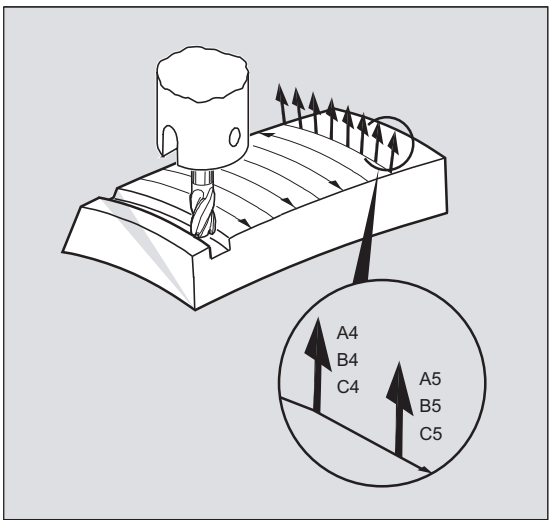


그림 6-3

6.2.5 페이스 밀링 (3D 밀링 A4, B4, C4, A5, B5, C5)

작동

페이스 밀링은 모든 종류의 곡면을 가공하는데 사용됩니다 .



이러한 유형의 3D 밀링의 경우 공작물 표면에 대한 3D 경로의 선 정의가 필요합니다 .

공구 형상 및 치수는 계산 시 고려되며 일반적으로 CAM 에서 수행됩니다 . 완전히 계산된 NC 블록은 사후 프로세서를 통해 제어 시스템에서 판독됩니다 .

경로 곡률 프로그램

표면 설명

경로 곡률은 다음 구성 요소를 통해 표면 법선 벡터에 의해 설명됩니다 .

A4, B4, C4 블록 시작에서 벡터 시작

A4, B5, C5 블록 끝에서 벡터 종료

블록에 시작 벡터만 포함된 경우 표면 법선 벡터는 블록 전반에 걸쳐 일정합니다 . 블록에 종료 벡터만 포함된 경우 보간은 대형 반경 원형 보간을 통해 이전 블록의 종료 값에서 프로그래밍된 종료 값으로 진행됩니다 .

시작 및 종료 벡터가 모두 프로그래밍되면 보간은 대형 반경 원형 보간을 통해 두 방향 사이에서 실행됩니다 . 이를 통해 지속적으로 생성할 경로를 유연화할 수 있습니다 .

초기 설정 시 활성 G17 - G19 레벨과 관계 없이 표면 법선 벡터는 Z 방향을 향합니다 .

벡터 길이는 의미없습니다 .

프로그램되지 않은 벡터 구성 요소는 0 으로 설정됩니다 .

활성 ORIWKS(" 방향 축의 기준 (ORIWKS, ORIMKS)" 참조) 를 통해 표면 법선 벡터는 활성 프레임과 관련되고 프레임 회전 시 회전합니다 .

장비 제조업체

표면 법선 벡터는 기계 데이터를 통해 설정된 제한 값 이내에서 경로 접선와 수직을 이루어야 하며 그렇지 않을 경우 알람이 출력됩니다 .

6.2.6 방향 축 기준 (ORIWKS, ORIMKS)

기능

다음을 사용하여 공작물 좌표계에서 오리엔테이션 프로그래밍

- Euler 또는 RPY 각도 또는
- 오리엔테이션 벡터

회전 모션의 코스는 ORIMKS/ORIWKS 를 사용하여 설정할 수 있습니다 .

주

장비 제조업체

오리엔테이션에 대한 보간 유형은 머신 데이터를 사용하여 지정합니다 .

MD21104 $MC_ORI_IPO_WITH_G_CODE

= FALSE: G 코드 ORIWKS 및 ORIMKS 가 기준을 제공합니다 .

= TRUE: 51 번째 그룹의 G 코드 (ORIAXES, ORIVECT, ORIPLANE 등) 가 기준을 제공합니다 .

구문

ORIMKS=...

ORIWKS=...

의미

ORIMKS	기계 좌표계에서 회전
ORIWKS	공작물 좌표계에서 회전

주

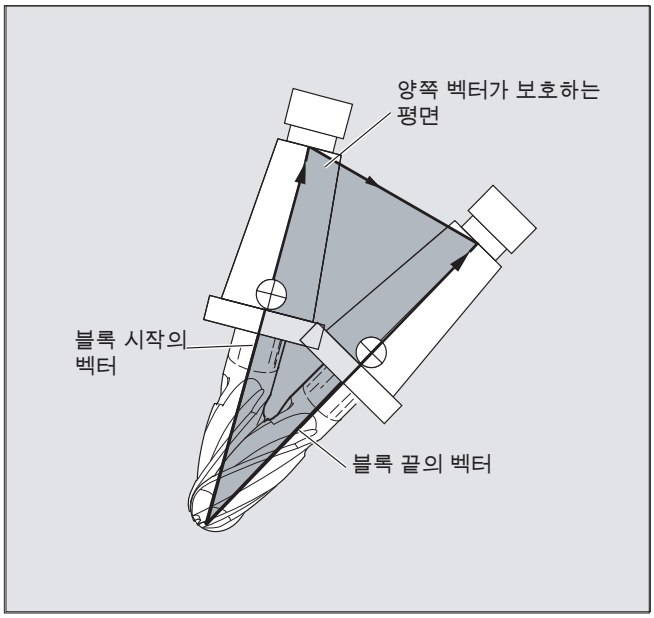
ORIWKS 가 기본 설정입니다 . 5 축 프로그램의 경우 실행할 기계를 즉시 확인할 수 없을 때에는 항상 ORIWKS 를 선택해야 합니다 . 기계가 실제로 실행하는 이동은 기계 운동 구조에 따라 다릅니다 .

ORIMKS 는 기계의 실제 동작을 프로그래밍할 때 (장치 등과의 충돌을 방지하기 위해) 사용할 수 있습니다 .

설명

ORIMKS 의 경우 공구에 의해 수행되는 이동은 기계 운동 구조에 따라 **다릅니다** . 공간 상의 고정 지점에서 공구 팁의 방향을 변경할 경우 선형 보간은 로타리 축 위치 사이에서 발생합니다 .

ORIWKS 의 경우 공구에 의해 수행되는 이동은 기계 운동 구조에 따라 **종속되지 않습니다** . 고정 툴 팁의 방향 변경을 통해 공구는 시작 및 종료 벡터에 의해 설정된 평면에서 이동합니다 .



단일 위치

주

ORIWKS

5 축 기계의 단일 설정 영역에서 방향을 이동하려면 기계 축을 상당히 이동해야 합니다 . (예를 들어 로타리 축으로 C 및 회전 축으로 A 를 가진 로타리 스위블 헤드에서 A = 0 의 모든 위치는 일치합니다 .)

장비 제조업체

기계 축의 과부하를 방지하기 위해 속도 제어 시스템이 단일 위치 근처의 공구 경로 속도를 상당히 감소합니다 .

기계 데이터

$MC_TRAFO5_NON_POLE_LIMIT

$MC_TRAFO5_POLE_LIMIT

극점에 가까운 방향 이동은 극점을 향하고 급속한 가공이 가능하도록 변환을 파라미터화할 수 있습니다 .

단일 위치는 MD $MC_TRAFO5_POLE_LIMIT 에 의해서만 처리됩니다 .

참고 문서 :
/FB3/ 작동 설명서 , 특별한 기능 ; 3- ~ 5- 축 변환 (F2),
" 단일 지점 및 처리 방법 " 절 .

6.2.7 오리엔테이션 축 프로그래밍 (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2)

기능

오리엔테이션 축 기능은 공간 상의 공구 오리엔테이션을 설명하고 로터리 축의 옵셋을 프로그래밍하여 수행됩니다. 또한 공구 자체를 회전하여 3도의 자유도를 확보할 수 있습니다. 이 경우 공구 오리엔테이션은 6 축 변환이 필요한 세 번째 로터리 축을 통해 공간에서 지정됩니다 . 공구 자체 회전은 회전 벡터의 보간 유형에 따라 회전의 THETA 각도를 사용하여 정의됩니다 (ORIROTA/TR/TT, ORIROTC, THETA)").

프로그래밍

축 식별자 A2, B2 및 C2 를 사용하여 오리엔테이션 축을 프로그램합니다 .

N... ORIAXES 또는 ORIVECT	직선 또는 대형 반경 원호 보간
N... G1 X Y Z A B C	
또는	또는
N... ORIPLANE	평면의 오리엔테이션 보간
또는	또는
N.. ORIEULER 또는 ORIRPY/ORIRPY2	오리엔테이션 각도 Euler/RPY 각도
N... G1 X Y Z A2= B2= C2=	가상 축의 각도 프로그램
또는	또는
N... ORIVIRT1 또는 ORIVIRT2	가상 축의 방향 벡터 1 또는 2
N... G1 X Y Z A3= B3= C3=	프로그래밍 정의

오리엔테이션 축의 기타 로터리 축 옵셋은 공간 상의 테이퍼 주변 표면을 따라 오리엔테이션 각도에 대해 프로그래밍할 수 있습니다 . 테이퍼의 주변 표면을 따라 오리엔테이션 프로그램 (ORIPLANE, ORICONxx)" 을 참조하십시오 .

파라미터

ORIAXES	기계 또는 오리엔테이션 축의 직선 보간
ORIVECT	대형 반경 원호 보간 (ORIPLANE 과 동일)
ORIMKS	기계 좌표계에서 회전
ORIWKS	공작물 좌표계에서 회전
	설명은 공구 오리엔테이션 회전 절을 참조하십시오 .
A= B= C=	기계 축 위치 프로그램
ORIEULER	Euler 각도를 통한 오리엔테이션 프로그램
ORIRPY	RPY 각도를 통한 오리엔테이션 프로그램 방향 순서는 XYZ 이고 , A2 는 X 기준 회전 각도 B2 는 Y 기준 회전 각도 C2 는 Z 기준 회전 각도
ORIRPY2	RPY 각도를 통한 오리엔테이션 프로그램 방향 순서는 ZYX 이고 , A2 는 Z 기준 회전 각도 B2 는 Y 기준 회전 각도 C2 는 X 기준 회전 각도
A2= B2= C2=	가상 축의 각도 프로그램
ORIVIRT1	가상 오리엔테이션 축을 사용한 오리엔테이션 프로그램
ORIVIRT2	(정의 1), MD $MC_ORIAX_TURN_TAB_1 에 따라 정의
	(정의 2), MD $MC_ORIAX_TURN_TAB_2 에 따라 정의
A3= B3= C3=	방향 축의 방향 벡터 프로그램

설명

장비 제조업체

MD $MC_ORI_DEF_WITH_G_CODE 는 프로그래밍된 각도 A2, B2, C2 를 정의하는 방법을 지정하는 데 사용합니다 :

정의는 MD $MC_ORIENTATION_IS_EULER(표준) 를 따르거나 G 그룹 50(ORIEULER, ORIRPY, ORIVIRT1, ORIVIRT2) 을 따릅니다 .

MD $MC_ORI_IPO_WITH_G_CODE 는 활성화할 보간 모드 유형을 정의하는 데 사용됩니다 : ORIWKS/ORIMKS 또는 ORIAXES/ORIVECT.

조그 모드

이 모드의 작업에서 방향 각도 보간은 항상 선형적입니다 . 이송 키를 통해 연속 및 증분 이송 동안 하나의 방향 축만 이송할 수 있습니다 . 방향 축은 핸드휠을 사용하여 동시에 이송할 수 있습니다 .

방향 축을 수동 이송하는 경우 채널별 피드가 스위치를 재지정하거나 급속 이송 재지정 시 급속 이송이 스위치 작업을 재지정합니다 .

개별 속도 설정은 다음 기계 데이터를 통해 가능합니다 :

$MC_JOG_VELO_RAPID_GEO

$MC_JOG_VELO_GEO

$MC_JOG_VELO_RAPID_ORI

$MC_JOG_VELO_ORI

주

" 변환 패키지 취급 " 을 통한 SINUMERIK 840D

"Cartesian 수동 이송 " 기능을 사용하면 JOG 모드에서 형상 축의 변환은 기준 시스템 MCS, WCS 및 TCS 에서 서로 간에 개별적으로 설정할 수 있습니다 .

참고 문서 :
/FB2/ 기능 설명 확장 기능 ; 운동학적 변환 (M1)

6.2.8 테이퍼의 주변 표면을 따라 오리엔테이션 프로그래밍 (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO)

기능

확장된 오리엔테이션을 통해 공간 상의 테이퍼의 주변 표면을 따라 오리엔테이션을 변경할 수 있습니다 . 오리엔테이션 벡터는 ORICONxx 모달 명령을 사용하여 테이퍼의 주변 표면에서 보간됩니다 . 공간에서 보간하는 경우 종료 오리엔테이션은 ORIPLANE 을 통해 프로그래밍할 수 있습니다 . 시작 오리엔테이션은 일반적으로 이전 블록에 의해 정의됩니다 .

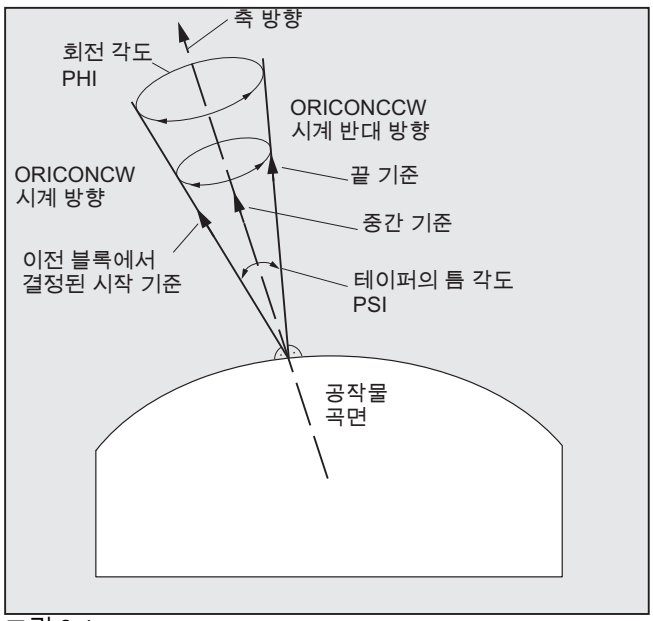


그림 6-4

프로그래밍

종료 오리엔테이션은 A2, B2, C2 를 사용하여 Euler 또는 RPY 각도 방식으로 각도 프로그램을 지정하거나 A, B, C 를 사용하여 로터리 축 위치를 프로그래밍하여 정의됩니다 . 테이퍼의 주변 표면을 따라 오리엔테이션 축에 대해 자세한 프로그램 세부 정보가 필요합니다 .

- A6, B6, C6 벡터를 통한 테이퍼의 로터리 축
- 식별자 NUT 를 가진 개방 각도 PSI
- A7, B7, C7 를 통해 테이퍼를 벗어난 중간 오리엔테이션

주

테이퍼의 로터리 축에 대한 방향 벡터 A6, B6, C6 프로그램

종료 오리엔테이션 프로그램은 경우에 따라 필요 없습니다 . 종료 오리엔테이션을 지정하지 않으면 360 도의 전체 외부 테이퍼가 보간됩니다 .

NUT=angle 로 테이퍼의 개방 각도 프로그램

종료 오리엔테이션을 지정해야 합니다 .

360 도의 전체 외부 테이퍼는 이 방법으로 보간할 수 없습니다 .

테이퍼 외부에서 중간 오리엔테이션 A7, B7, C7 프로그램

종료 오리엔테이션을 지정해야 합니다 . 오리엔테이션 및 회전 방향 변경은 세 가지 벡터인 시작 오리엔테이션 , 종료 오리엔테이션 및 중간 오리엔테이션에 의해 고유하게 정의됩니다. 세 가지 모든 벡터가 서로 달라야 합니다. 프로그래밍된 중간 오리엔테이션이 시작 또는 종료 오리엔테이션과 평행할 경우 오리엔테이션에 대한 직선적인 대형 반경 원호 보간이 시작 및 종료 벡터에 의해 정의된 평면에서 정의됩니다 .

테이퍼의 주변 표면에 대한 확장 오리엔테이션 보간

N... ORICONCW 또는 ORICONCCW	테이퍼 외부에 대한 **보간**
N... A6= B6= C6= A3= B3= C3=	테이퍼의 시계 방향 / 시계 반대 방향으로 방향 벡터 및 종료 오리엔테이션 또는
또는	
N... ORICONTO	종료 오리엔테이션의 접선 트렌지션 및 지정
N... G1 X Y Z A6= B6= C6=	또는 다음을 사용하여
또는	
N... ORICONIO	종료 오리엔테이션 지정 및 테이퍼 외부에서 중간 오리엔테이션
N... G1 X Y Z A7= B7= C7=	
N... PO[PHI]=(a2, a3, a4, a5)	회전 각도 폴리노미널 및
N... PO[PSI]=(b2, b3, b4, b5)	개방 각도 폴리노미널

파라미터

ORIPLANE	평면에서 보간 (대형 반경 원호 보간)
ORICONCW	테이퍼의 주변 표면에서 시계 방향으로 보간됩니다 .
ORICONCCW	테이퍼의 주변 표면에서 시계 반대 방향으로 보간됩니다 .
ORICONTO	테이퍼의 주변 표면에서 접선 트렌지션으로 보간
A6= B6= C6=	테이퍼의 로터리 축 프로그램 (법선 벡터)
NUT=angle	테이퍼의 개방 각도 (도)
NUT=+179	180 도보다 작거나 같은 각도로 이송
NUT=-181	180 도보다 크거나 같은 각도로 이송
ORICONIO	테이퍼의 주변 표면에 대한 보간
A7= B7= C7=	중간 오리엔테이션 (법선 벡터로 프로그래밍)
PHI	테이퍼의 방향 축에 대한 회전 오리엔테이션 각도
PSI	테이퍼의 개방 각도
가능한 폴리노미널 PO[PHI]=(a2, a3, a4, a5) PO[PSI]=(b2, b3, b4, b5)	서로 다른 각도와 관계 없이 최대 5 차 폴리노미널까지 프로그래밍 할 수 있습니다 .

서로 다른 오리엔테이션 변경의 예제

```
…
N10 G1 X0 Y0 F5000
N20 TRAORI(1)                                  ; 오리엔테이션 변환 실행
N30 ORIVECT                                    ; 벡터로 공구 오리엔테이션 보간
…                                              ; 평면에서 공구 오리엔테이션
N40 ORIPLANE                                   ; 대형 반경 원호 보간 선택
N50 A3=0 B3=0 C3=1
N60 A3=0 B3=1 C3=1                             ; Y/Z 평면에서 오리엔테이션은 45 도로 회전합니다 .
                                               ; 오리엔테이션 (0,1, 2,1/ √ 2, 1/ √ 2) 은 블록 끝에
                                               ; 도달합니다 .
…
N70 ORICONCW                                   ; 테이퍼 외부에서 오리엔테이션 프로그래밍
N80 A6=0 B6=0 C6=1 A3=0 B3=0 C3=1              ; 오리엔테이션 벡터는 방향 (0,0,1) 의 테이퍼 외부에서
                                               ; CW 방향으로 오리엔테이션 (1/ √ 2,0,1/ √ 2) 까지
                                               ; 보간되고 , 회전 각도는 270 도입니다 .
N90 A6=0 B6=0 C6=1                             ; 공구 오리엔테이션은 동일한 테이퍼의 외부에서 전체
                                               ; 회전을 통해 적용됩니다 .
```

설명

공간 상의 어디서나 테이퍼의 주변 표면을 따라 방향 변경에 대해 설명해야 하는 경우 회전해야 할 공구 방향에 대한 벡터를 알고 있어야 합니다. 또한 시작 및 종료 방향을 지정해야 합니다 . 시작 방향은 이전 블록으로부터 제공되며 종료 방향은 다른 조건을 통해 프로그램 또는 정의해야 합니다 .

ORIPLANE 평면에서의 프로그래밍은 ORIVECT 에 해당

각도 다항식과 함께 대형 반경 원형 보간 프로그래밍은 형상에 대한 선형적 및 폴리노미널 보간에 해당합니다. 공구 방향은 시작 및 종료 방향에 의해 정의된 평면에서 보간됩니다. 추가 다항식을 프로그래밍하는 경우 방향 벡터를 평면에서 벗어나 기울일 수 있습니다 .

평면 G2/G3, CIP 및 CT 에서 원형 프로그램

확장 방향은 평면에서 원형 보간에 해당합니다 . G2/G3, 중간 지점 CIP 를 통한 원형 및 접선 원형과 같이 중심 또는 반경을 가진 원의 해당 프로그램 옵션의 경우 다음을 참조하십시오 .

참고 문서 : 프로그램 설명서 기본 사항 , " 이동 명령 프로그램 ".

방향 프로그램

테이퍼의 주변 표면에서 방향 벡터 보간 ORICONxx

테이퍼의 주변 표면에 대한 방향 보간을 위해 G 코드 그룹 51 에서 네 가지 서로 다른 유형의 보간을 선택할 수 있습니다 .

1. 종료 방향 , 테이퍼 방향 또는 개방 각도 지정을 통해 시계 방향 ORICONCW 으로 테이퍼 외부에서 보간. 방향 벡터는 식별자 A6, B6, C6 및 간격 0도 - 180도 사이에서 식별자 NUT= value 범위를 가진 테이퍼의 개방 각도를 통해 프로그래밍됩니다 .
2. 종료 방향 , 테이퍼 방향 또는 개방 각도 지정을 통해 시계 반대 방향 ORICONCCW 으로 테이퍼 외부에서 보간 . 방향 벡터는 식별자 A6, B6, C6 및 간격 0 도 - 180 도 사이에서 식별자 NUT= value 범위를 가진 테이퍼의 개방 각도를 통해 프로그래밍됩니다 .
3. 종료 방향 및 중간 방향 지정을 통해 테이퍼 외부에서 보간 ORICONIO, 식별자 A7, B7, C7 를 통해 프로그래밍됩니다 .
4. 접선 전이 및 종료 방향 지정을 통해 테이퍼 외부에서 보간 ORICONTO. 방향 벡터는 식별자 A6, B6, C6 을 통해 프로그래밍됩니다 .

6.2.9 두 개의 접점 방향 지정 (ORICURVE, PO[XH]=, PO[YH]=, PO[ZH]=)

기능

공간 상의 두 번째 곡선을 사용하여 오리엔테이션 변경 프로그램

공간 상의 곡선을 따라 공구 팁을 사용하는 것 이외에 오리엔테이션 변경을 프로그램하는 다른 방법은 ORICURVE 를 사용하여 공구의 두 번째 접점 이동을 프로그래밍하는 것입니다 . 이 방법에서 , 공구 오리엔테이션 변경은 공구 벡터를 프로그램할 때처럼 고유하게 정의할 수 있습니다 .

장비 제조업체

공구의 두 번째 오리엔테이션 경로를 프로그램하는 경우 , 머신 데이터를 통해 설정할 수 있는 축 식별자에 대한 장비 제조업체의 참고 사항을 참조하십시오 .

프로그래밍

이 유형의 보간은 공간 상의 두 개 곡선에 대한 점 (G1 사용) 또는 폴리노미널 (POLY 사용) 을 프로그래밍하는 데 사용할 수 있습니다 . 원형 및 나선은 허용되지 않습니다 . BSPLINE 스플라인 보간 및 " 짧은 스플라인 블록 조합 " 기능도 실행할 수 있습니다 .

참고 자료 :
/FB1/ 기능 매뉴얼, 기본 기능; 연속 이송 모드, 정위치 정지, 선행 제어 (B1), 장: 짧은 스플라인 블록 조합

다른 스플라인 유형 , ASPLINE 및 CSPLINE, 및 COMPON, COMPCURV 또는 COMPCAD 을 사용한 컴프레서 활성화는 허용되지 않습니다 .

공구의 두 개 접점 이동은 좌표에 대한 오리엔테이션 폴리노미널로 프로그램할 때 5 차 폴리노미널로 미리 정의할 수 있습니다 .

공간 상의 추가 곡선 및 좌표 폴리노미널을 통한 확장 오리엔테이션 보간

N... ORICURVE	공구의 두 번째 접점 이동 및 관련 좌표의 추가 폴리노미널 지정
N... PO[XH]=(xe, x2, x3, x4, x5)	
N... PO[YH]=(ye, y2, y3, y4, y5)	
N... PO[ZH]=(ze, z2, z3, z4, z5)	

파라미터

ORICURVE	공구의 두 접점 사이 이동을 지정하는 오리엔테이션 보간
XH YH ZH	공간 상의 곡선으로 추가 형상의 공구에 대한 두 번째 접점의 좌표 식별자
가능한 폴리노미널 PO[XH]=(xe, x2, x3, x4, x5) PO[YH]=(ye, y2, y3, y4, y5) PO[ZH]=(ze, z2, z3, z4, z5)	해당 종점을 사용하는 것 이외에 공간 상의 곡선은 폴리노미널을 사용하여 프로그래밍할 수 있습니다 .
xe, ye, ze	공간 상의 곡선에 대한 종점
xi, yi, zi	최대 5 차 폴리노미널의 계수

주

두 번째 오리엔테이션 경로를 프로그램하기 위한 식별자 XH YH ZH

다른 식별자 또는 직선 축과 충돌하지 않도록 식별자를 선택해야 합니다 .

X Y Z 축

및 다음과 같은 로터리 축

A2 B2 C2 Euler 각도 또는 RPY 각도

A3 B3 C3 방향 벡터

A4 B4 C4 또는 A5 B5 C5 표면 법선 벡터

A6 B6 C6 회전 벡터 또는 A7 B7 C7 중간점 좌표

또는 기타 보간 파라미터 .

6.3 방향 다항식 (PO[angle], PO[coordinate])

작동

현재 활성화된 G 코드 그룹의 다항식 보간과 관계 없이 3 축 ~ 5 축 변환에 대해 두 가지 서로 다른 유형의 최대 5 차 방향 다항식을 프로그래밍할 수 있습니다 .

1. **각도**에 대한 다항식 : 시작 및 종료 방향에 의해 정의된 평면을 기준으로 리드 각도 LEAD, 틸트 각도 TILT.
2. **좌표**에 대한 다항식 : 공구의 기준점에서 공구 방향에 대해 공간 상의 두 번째 곡선의 XH, YH, ZH.

6 축 변환을 통해 회전 벡터 THT 의 회전은 공구 방향에 추가하여 공구 회전에 대해 최대 5 차 다항식으로 프로그래밍할 수 있습니다 .

구문

유형 1 **각도**에 대한 방향 다항식

N… PO[PHI]=(a2, a3, a4, a5)	3 축 ~ 5 축 변환
N… PO[PSI]=(b2, b3, b4, b5)	3 축 ~ 5 축 변환

유형 2 **좌표**에 대한 방향 다항식

N… PO[XH]=(xe, x2, x3, x4, x5) N… PO[YH]=(ye, y2, y3, y4, y5) N… PO[ZH]=(ze, z2, z3, z4, z5)	공구 방향에 대해 두 번째 방향 경로의 좌표에 대한 식별자

두 가지 경우 모두 , 6 축 변환을 통해 다음을 사용하여 **회전**에 대한 다항식을 프로그래밍할 수 있습니다 .

N… PO[THT]=(c2, c3, c4, c5)	경로를 기준으로 회전 보간
또는	
N… PO[THT]=(d2, d3, d4, d5)	방향 변경에 대한 절대 , 상대 및 접선 보간

방향 벡터 . 이것은 변환이 THETA 회전 각도를 사용하여 프로그램 및 보간할 수 있는 옵셋을 가진 회전 벡터를 지원하는 경우 가능합니다 .

의미

PO[PHI]	시작 및 종료 방향 사이 평면에서의 각도
PO[PSI]	시작 및 종료 방향 사이 평면에서 방향의 기울기를 설명하는 각도
PO[THT]	THETA를 사용하여 프로그래밍된 그룹 54의 G 코드 중 하나에 대한 방향 벡터를 회전하여 생성된 회전 각도
PHI	리드 각도 LEAD
PSI	틸트 각도 TILT
THETA	Z 에서 공구 방향에 대한 회전
PO[XH]	공구의 기준점에 대한 X 좌표
PO[YH]	공구의 기준점에 대한 Y 좌표
PO[ZH]	공구의 기준점에 대한 Z 좌표

설명

다음과 같은 경우 방향 다항식을 프로그래밍할 수 없습니다 :

- ASPLINE, BSPLINE, CSPLINE 스플라인 보간이 활성화된 경우 .
  유형 1 방향 각도 다항식은 스플라인 보간을 제외하고 모든 유형의 보간에 가능합니다 . 즉 ,
  급이송 G00 또는 피드 속도 G01 을 통한 선형 보간 ,
  POLY 를 사용한 다항식 보간 및
  원형 / 나선 보간 G02, G03, CIP, CT, INVCW 및 INCCCW
  .
  그러나 유형 2 방향 좌표 다항식은 급이송 G00 또는
  피드 속도 G01 을 통한 선형 보간 또는
  POLY 를 통한 다항식 보간이 활성화된 경우에만 가능합니다 .
- 방향이 ORIAXES 축 보간을 사용하여 보간되는 경우 . 이 경우 다항식은 방향 축 A 및 B 에 대해 PO[A] and PO[B] 를 통해 직접 프로그래밍할 수 있습니다 .

유형 1 ORIVECT, ORIPLANE 및 ORICONxx 를 통한 방향 다항식

유형 1 방향 다항식만 ORIVECT, ORIPLANE 및 ORICONxx 를 통해 대형 반경 원형 보간 및 테이퍼의 외부에서 보간에 대해 가능합니다 .

유형 2 ORICURVE 를 통한 방향 다항식

공간 간의 추가 곡선을 통한 보간 ORICURVE 가 활성화되면 방향 벡터의 Cartesian 구성 요소가 보간되고 유형 2 방향 다항식만 가능합니다 .

6.4 공구 오리엔테이션 회전 (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA)

작동

또한 이동 가능한 공구를 가진 기계 유형에서 공구 방향을 변경하려면 각 블록을 종료 방향으로 프로그래밍합니다 . 기계 운동 구조에 따라 방향 축의 방향 또는 방향 벡터 THETA 회전 방향을 프로그래밍할 수 있습니다 . 이러한 회전 벡터에 대해 서로 다른 보간 유형을 프로그래밍할 수 있습니다 :

- ORIROTA: 절대 회전 방향에 대한 회전 각도 .
- ORIROTR: 시작 및 종료 방향 사이 평면을 기준으로 회전 각도 .
- ORIROTT: 방향 벡터의 변경을 기준으로 회전 각도 .
- ORIROTC: 경로 접선에 대한 접선 회전 각도 .

구문

보간 유형 ORIROTA가 활성화된 경우에만 회전 또는 회전 벡터의 각도는 다음과 같이 네 가지 모든 모드에서 프로그래밍할 수 있습니다 :

1. 로타리 축 위치 A, B, C 로 직접
2. A2, B2, C2 를 통한 Euler 각도 (도)
3. A2, B2, C2 를 통한 RPY 각도 (도)
4. A3, B3, C3 을 통한 방향 벡터 (THETA=value 를 사용한 회전 각도)

ORIROTR 또는 ORIROTT 가 활성화되면 회전 각도는 THETA 를 통해서만 직접 프로그래밍할 수 있습니다 .

또한 방향 변경 없이 개별 블록에서도 회전을 프로그래밍할 수 있습니다 . 이 경우 ORIROTR 및 ORIROTT 는 관계 없습니다 . 이 경우 회전 각도는 항상 절대 방향을 기준으로 보간됩니다 (ORIROTA).

N... ORIROTA	회전 벡터 보간 정의
N... ORIROTR	
N... ORIROTT	
N... ORIROTC	
N... A3= B3= C3= THETA=value	방향 벡터 회전 정의
N... PO[THT]=(d_2, d_3, d_4, d_5)	5 차 다항식을 통한 회전 각도 보간

의미

ORIROTA	절대 회전 방향에 대한 회전 각도 .
ORIROTR	시작 및 종료 방향 사이 평면을 기준으로 회전 각도 .
ORIROTT	방향 변경에 대한 접선 회전 벡터를 통한 회전 각도
ORIROTC	경로 접선에 대한 접선 회전 벡터를 통한 회전 각도
THETA	방향 벡터의 회전
THETA=value	블록 끝에 도달한 회전 각도 (도)
THETA=Θe	회전 벡터의 끝각 Θ_e 를 통한 회전 각도
THETA=AC(...)	절대 치수로의 비모달식 전환
THETA=AC(…)	증분 치수로의 비모달식 전환
Θe	G90 의 절대값 및 G91(증분 치수) 의 상대값 모두에 대한 회전 벡터의 끝각이 활성화됩니다 .
PO[THT]=(....)	회전 각도에 대한 다항식

방향 회전의 예

프로그램 코드	비고
N10 TRAORI	; 방향 변환 실행
N20 G1 X0 Y0 Z0 F5000	; 공구 방향
N30 A3=0 B3=0 C3=1 THETA=0	; 회전 각도 0 의 Z 방향
N40 A3=1 B3=0 C3=0 THETA=90	; X 방향 및 약 90 도 회전
N50 A3=0 B3=1 C3=0 PO[THT]=(180,90)	; 기본 설명
N60 A3=0 B3=1 C3=0 THETA=IC(-90)	; Y 방향 및 약 180 도 회전
N70 ORIROTT	; 90 도로 회전을 일정하게 유지
N80 A3=1 B3=0 C3=0 THETA=30	; 방향 변경을 기준으로 회전 각도
	; X/Y 평면에 대해 30 도 각도의 회전 벡터

블록
N40 을 보간하는 경우 회전 각도가 초기값 0 도에서 최종값 90 도로 선형적으로 보간됩니다 . 블록 N50 에서 회전 각도가

포물선 $\theta(u) = +90u^2$ 에 따라 90 도에서 180 도로 변경됩니다 . N60 에서 방향 변경 없이 회전을 실행할 수 있습니다 .

N80에서 공구 방향은 Y 방향에서 X 방향으로 회전합니다. 방향 변경은 X/Y 평면에서 발생하고 회전 벡터는 이 평면에 대해 30 도 각도를 설명합니다 .

설명

ORIROTA

회전 각도 THETA 는 공간 상의 절대 방향을 기준으로 보간됩니다 . 기본 회전 방향은 기계 데이터에서 정의됩니다 .

ORIROTR

회전 각도 THETA 는 시작 및 종료 방향에 의해 정의된 평면을 기준으로 보간됩니다 .

ORIROTT

회전 각도 THETA 는 방향 변경을 기준으로 보간됩니다 . THETA=0 의 경우 회전 벡터가 방향 변경에 대해 접선 방향으로 보간되며 최소 하나의 다항식이 방향에 대해 " 틸트 각도 PSI" 를 프로그래밍한 경우에만 ORIROTR과 다릅니다. 결과적으로 평면에서 실행되지 않는 방향으로 변경됩니다 . 추가 회전 각도 THETA 는 항상 방향 변경을 기준으로 특정 값을 산출하도록 회전 벡터를 보간하는 데 사용할 수 있습니다 .

ORIROTC

회전 벡터는 THETA 각도를 사용하여 프로그래밍할 수 있는 옵센을 통해 경로 접선을 기준으로 보간됩니다 . 또한 옵셋 각도에 대해 최대 5 차 다항식 PO[THT]=(c2, c3, c4, c5) 을 프로그래밍할 수 있습니다 .

6.5 경로 기준 방향

6.5.1 경로 기준 방향 유형

작동

이 확장 기능을 사용하여 블록 끝 뿐만 아니라 전체 경로 전반에 걸쳐 관련 방향을 설정할 수 있습니다 . 이전 블록에서 적용된 방향이 대형 반경 원형 보간을 사용하여 프로그래밍된 끝 방향으로 이송됩니다 . 기본적으로 경로를 기준으로 원하는 방향을 프로그래밍하는 두 가지 방법은 다음과 같습니다 .

1. 공구 회전처럼 , 공구 방향은 ORIPATH, ORPATHTS 를 사용하여 경로를 기준으로 보간됩니다 .
2. 방향 벡터는 일반적인 방법으로 프로그래밍 및 보간됩니다 . 방향 벡터의 회전은 ORIROTC 를 사용하여 경로 접선을 기준으로 실행됩니다 .

구문

공구의 방향 및 회전 보간 유형은 다음을 사용하여 프로그래밍됩니다 :

N... ORIPATH	경로 기준 방향
N... ORIPATHS	방향 유연화 특성을 통한 경로 기준 방향
N... ORIROTC	경로를 기준으로 회전 벡터 보간

경로의 모서리에서 발생하는 방향 blip 은 ORIPATHS 를 사용하여 유연화할 수 있습니다 . 후퇴 이동의 방향 및 경로 길이는 구성 요소 A8=X, B8=Y C8=Z 를 사용하여 벡터를 통해 프로그래밍됩니다 .

ORIPATH/ORIPATHS 는 세 개의 각도를 통해 경로 접선에 대한 다양한 기준을 프로그래밍하는 데 사용할 수 있습니다 .

- LEAD= 경로 및 표면을 기준으로 리드 각도의 지정
- TILT= 경로 및 표면을 기준으로 틸트 각도의 지정
- THETA= 회전 각도

전체 경로의 경우 . PO[THT]=(...) 를 사용하는 THETA 회전 각도 이외에 최대 5 차 다항식을 프로그래밍할 수 있습니다 .

주

장비 제조업체

장비 제조업체의 설명서를 참조하십시오 . 기타 설정은 구성 가능한 기계 및 셋팅 데이터를 통해 경로 기준 방향에 대해 설정할 수 있습니다 . 자세한 내용은 다음을 참조하십시오 .

참고 문서 :
/FB3/ 작동 설명서 , 특별한 기능 ; 3- ~ 5- 축 변환 (F2)
" 방향 " 장

의미

다양한 설정은 기계 데이터를 통해 보간 각도 LEAD 및 TILT 에 대해 수행할 수 있습니다 :

- LEAD 및 TILT 를 사용하여 프로그래밍된 공구 방향 기준은 전체 블록에 대해 유지됩니다 .
- 리드 각도 LEAD: 접선 및 법선 벡터 TILT 기준 수직 방향 회전 : 법선 벡터 기준 방향 회전 .
- 리드 각도 LEAD: 접선 및 법선 벡터 틸트 각도 TILT 기준 수직 방향 회전 : 경로 접선 방향으로 방향 회전 .
- 회전 각도 THETA=: 6 축 변환에서 방향 축 역할을 수행하는 추가적인 세 번째 로타리 축의 공구 회전 .

주

OSC, OSS, OSSE, OSD 및 OST 와 연동하여 허용되지 않는 경로 기준 방향

경로 기준 방향 보간 , 즉 ORIPATH 또는 ORIPATHS 및 ORIOTC 는 그룹 34 의 G 코드를 통해 방향 특성 유연화와 연동하여 프로그래밍할 수 없습니다 . 이 경우 OSOF 를 활성화해야 합니다 .

6.5.2 경로 기준 공구 방향 회전 (ORIPATH, ORIPATHS, 회전 각도)

기능

6 축 변환의 경우 공구는 공간 상에서 원하는 대로 공구 오리엔테이션을 지정하도록 세 번째 축을 통해 공구를 회전할 수 있습니다. ORIPATH 또는 ORIPATHS를 사용하여 경로 기준 공구 오리엔테이션 회전의 경우 THETA 회전 각도를 통해 추가 회전을 프로그래밍할 수 있습니다 . 또는 공구 방향에 수직인 평면에 위치한 벡터를 사용하여 LEAD 및 TILT 각도를 프로그래밍할 수 있습니다 .

장비 제조업체

장비 제조업체의 설명서를 참조하십시오 . LEAD 및 TILT 각도 보간은 머신 데이터를 사용하여 서로 다르게 설정할 수 있습니다 .

구문

공구 오리엔테이션 및 공구 회전

경로 기준 공구 오리엔테이션 유형은 ORIPATH 또는 ORIPATHS 를 사용하여 활성화됩니다 .

N... ORIPATH	경로 기준 오리엔테이션 유형 활성화
N... ORIPATHS	오리엔테이션 유연화 특성을 통해 경로 기준 오리엔테이션 유형 활성화
회전 가능한 세 가지 각도 활성화 :	
N... LEAD=	표면 법선 벡터 기준 프로그래밍된 오리엔테이션의 각도
N... TILT=	평면에서의 프로그래밍된 오리엔테이션의 각도는 표면 법선 벡터를 기준으로 경로 접선에 대해 수직입니다 .
N... THETA=	세 번째 로터리 축의 공구 오리엔테이션에서 오리엔테이션 변경 기준 회전 각도

블록 끝에서 각도 값은 LEAD=value, TILT=value 또는 THETA=value 를 사용하여 프로그래밍됩니다 . 일정한 각도 이외에 세 가지 모든 각도에 대해 최대 5 차 폴리노미널을 프로그래밍할 수 있습니다 .

N... PO[PHI]=(a2, a3, a4, a5)	리드 각도 LEAD 에 대한 폴리노미널
N... PO[PSI]=(b2, b3, b4, b5)	틸트 각도 TILT 에 대한 폴리노미널
N... PO[THT]=(d2, d3, d4, d5)	회전 각도 THETA 에 대한 폴리노미널

고차 폴리노미널 계수가 0 인 경우 프로그래밍할 때 생략할 수 있습니다 . 예제 : PO[PHI]=a2 의 결과는 LEAD 각도의 포물선입니다 .

의미

경로 기준 공구 오리엔테이션

ORIPATH	경로 기준 공구 오리엔테이션
ORIPATHS	경로 기준 공구 오리엔테이션 ; 오리엔테이션 특성의 blip 가 유연화됩니다 .
LEAD	경로 접선 및 표면 법선 벡터에 의해 정의된 평면에서 표면 법선 벡터 기준 각도
TILT	Z 방향에서 오리엔테이션 회전 또는 경로 접선 기준 회전
THETA	Z 방향으로의 공구 방향에 대한 회전
PO[PHI]	LEAD 각도에 대한 방향 폴리노미널
PO[PSI]	TILT 각도에 대한 오리엔테이션 폴리노미널
PO[THT]	THETA 회전 각도에 대한 오리엔테이션 폴리노미널

주

회전 각도 THETA

자체적으로 오리엔테이션 축 역할을 수행하는 세 번째 로터리 축을 통해 공구를 회전시키려면 6 축 변환이 필요합니다 .

6.5.3 경로 기준으로 공구 회전 보간 (ORIROTC, THETA)

작동

회전 벡터를 통한 보간

경로 접선을 기준으로 ORIROTC 를 통해 프로그래밍된 공구 회전의 회전 벡터도 THETA 회전 각도를 사용하여 프로그래밍할 수 있는 옵셋을 통해 보간될 수 있습니다 . 따라서 PO[THT] 를 사용하여 옵셋 각도에 대해 최대 5 차 다항식을 프로그래밍할 수 있습니다 .

구문

N... ORIROTC	경로 접선 기준 공구 회전 실행
N... A3= B3= C3= THETA=value	방향 벡터 회전 정의
N... A3= B3= C3= PO[THT]=(c2, c3, c4, c5)	최대 5 차 다항식을 통한 옵셋 각도 보간

또한 방향 변경 없이 개별 블록에서도 회전을 프로그래밍할 수 있습니다 .

의미

6 축 변환으로 경로 기준 공구 회전 보간

ORIROTC	경로 접선 기준 접선 회전 벡터 실행
THETA=value	블록 끝에 도달한 회전 각도 (도)
THETA=θe	회전 벡터의 끝각 Θ_e 를 통한 회전 각도
THETA=AC(…)	블록당 절대 치수로 전환
THETA=IC(…)	블록당 증분 치수로 전환
PO[THT]=(c2, c3, c4, c5)	5 차 다항식을 통한 옵셋 각도 보간

주

회전 벡터 보간 ORIROTC

공구 방향과 반대 방향으로 경로 접선 기준 공구 회전 실행은 6 축 변환을 통해서만 가능합니다 .

활성 ORIROTC 사용

회전 벡터 ORIROTA를 프로그래밍할 수 없습니다. 프로그램을 수행할 경우 알람 14128 "활성 ORIROTC 를 통한 공구 회전 절대 프로그래밍 " 이 출력됩니다 .

3 축 ~ 5 축 변환에 대해 공구 회전 방향

공구 방향은 Euler 각도 , RPY 각도 또는 3 축 ~ 5 축 변환을 통한 방향 벡터를 사용하여 프로그래밍할 수 있습니다 . 공간 상의 공구 방향 변경도 대형 반경 원형 보간 ORIVECT, 방향 축 선형 보간 ORIAXES, 테이퍼의 주변 표면에 대한 모든 보간 ORICONxx 및 공구의 두 접점을 가진 공간 상의 곡선에 추가 보간을 프로그래밍하여 수행할 수 있습니다 .

G....	로타리 축 이동 세부 정보
X Y Z	선형 축 세부 정보
ORIAXES	기계 또는 방향 축의 선형 보간
ORIVECT	대형 반경 원형 보간 (ORIPLANE 과 동일)
ORIMKS ORIWKS	기계 좌표계에서 회전 공작물 좌표계에서 회전 설명은 공구 방향 회전 절을 참조하십시오 .
A= B= C=	기계 축 위치 프로그램
ORIEULER	Euler 각도를 통한 방향 프로그램
ORIRPY	RPY 각도를 통한 방향 프로그램
A2= B2= C2=	가상 축의 각도 프로그램
ORIVIRT1 ORIVIRT2	가상 방향 축을 사용한 방향 프로그램 (정의 1), MD $MC_ORIAX_TURN_TAB_1 에 따라 정의 (정의 2), MD $MC_ORIAX_TURN_TAB_2 에 따라 정의
A3= B3= C3=	방향 축의 방향 벡터 프로그램
ORIPLANE	평면에서 보간 (대형 반경 원형 보간)
ORICONCW	테이퍼의 주변 표면에서 시계 방향으로 보간됩니다 .
ORICONCCW	테이퍼의 주변 표면에서 시계 반대 방향으로 보간됩니다 .

ORICONTO	테이퍼의 주변 표면에서 접선 전이로 보간
A6= B6= C6=	테이퍼의 로타리 축 프로그램 (법선 벡터)
NUT=angle	테이퍼의 개방 각도 (도)
NUT=+179	180 도보다 작거나 같은 각도로 이송
NUT=-181	180 도보다 크거나 같은 각도로 이송
ORICONIO	테이퍼의 주변 표면에 대한 보간
A7= B7= C7=	중간 방향 (법선 벡터로 프로그래밍)
ORICURVE XH YH ZH, 예를 들어 다항식 PO[XH]=(xe, x2, x3, x4, x5) 사용	공구의 두 접점 사이 이동을 지정하는 방향 보간 . 끝점 이외에 추가 곡선 다항식도 프로그래밍할 수 있습니다 .

주

활성 ORIAXES 로 공구 방향을 방향 축을 통해 보간하면 회전 각도는 블록 끝에서 경로를 기준으로만 실행됩니다 .

6.5.4 방향 특성 유연화 (ORIPATHS A8=, B8=, C8=)

작동

형상에서 일정한 가속으로 발생하는 방향 변경으로 인해 특히 형상 모서리를 포함하여 경로 이동을 따라 원치 않는 중단이 발생할 수 있습니다 . 방향 특성의 결과 blip 는 개별 중간 블록을 삽입하여 유연화할 수 있습니다. 방향을 다시 지정하는 동안 ORIPATHS를 활성화하면 일정한 가속 상태에서 방향이 변경됩니다 . 이 단계에서 공구를 후퇴할 수 있습니다 .

장비 제조업체

이 기능을 활성화하는 데 사용된 미리 정의된 기계 및 셋팅 데이터에 대해서는 장비 제조업체의 참고 사항을 참조하십시오 .

기계 데이터는 후퇴 벡터를 해석하는 방법을 설정하는 데 사용할 수 있습니다 :

1. TCS 에서 Z 좌표는 공구 방향에 의해 정의됩니다 .
2. WCS 에서 Z 좌표는 활성 평면에 의해 정의됩니다 .

" 경로 기준 방향 " 기능에 대한 자세한 내용은
참고 문서 : /FB3/ 작동 설명서 , 특별한 기능 ; 3- ~ 5- 축 변환 (F2) 을 참조하십시오 .

구문

전반적으로 경로 기준 일정한 공구 방향에 대해 형상 모서리에서 자세한 프로그래밍 세부 정보가 필요합니다 . 이 이동의 방향 및 경로 길이는 구성 요소 A8=X, B8=Y C8=Z 를 사용하여 벡터를 통해 프로그래밍됩니다 .

N... ORIPATHS A8=X B8=Y C8=Z

의미

ORIPATHS	경로 기준 공구 방향 ; 방향 특성의 blip 가 유연화됩니다 .
A8= B8= C8=	방향 및 경로 길이에 대한 벡터 구성 요소
X, Y, Z	공구 방향으로 후퇴 이동

주

방향 벡터 A8, B8, C8 프로그래밍

이 벡터의 길이가 정확하게 0 인 경우 후퇴 이동은 실행되지 않습니다 .

ORIPATHS

경로 기준 공구 방향은 ORIPATHS 를 사용하여 활성화됩니다 . 그렇지 않을 경우 방향은 선형적인 대형 반경 원형 보간을 통해 시작 방향에서 끝 방향으로 이송됩니다 .

6.6 방향 압축 (COMPON, COMPCURV, COMPCAD)

기능

오리엔테이션 변환 (TRAORI) 이 활성화되고 공구 방향이 프로그래밍된 (유형에 관계 없이) NC 프로그램이 지정된 제한 범위 내에 있는 경우 압축할 수 있습니다 .

프로그래밍

공구 방향

오리엔테이션 변환 (TRAORI) 이 활성화되면 5 축 기계의 경우 공구 방향을 다음과 같은 방법으로 프로그래밍할 수 있습니다 (운동 구조와 관계 없음):

- 다음을 통해 방향 **벡터** 프로그래밍 :

  A3=<...> B3=<...> C3=<...>

- 다음을 통해 Euler **각도** 또는 RPY- **각도** 프로그래밍 :

  A2=<...> B2=<...> C2=<...>

공구의 회전

6- 축 기계의 경우 공구 오리엔테이션 이외에 공구 회전을 프로그래밍할 수 있습니다 .

회전 각도는 다음을 통해 프로그래밍됩니다 :

THETA=<...>

"공구 오리엔테이션 회전 (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) [쪽 348]" 을 참조하십시오 .

주

추가 회전이 프로그래밍된 NC 블록은 회전 각도가 **선형적으로** 변경되는 경우에만 압축할 수 있으며 따라서 회전 각도에 대해 PO[THT]=(...) 다항식을 프로그래밍하지 않아야 합니다 .

압축할 NC 블록의 일반적인 구조

압축할 수 있는 NC 블록의 일반적인 구조는 다음과 같을 수 있습니다 :

N... X=<...> Y=<...> Z=<...> A3=<...> B3=<...> C3=<...> THETA=<...> F=<...>

또는

N... X=<...> Y=<...> Z=<...> A2=<...> B2=<...> C2=<...> THETA=<...> F=<...>

주

위치 값은 직접 입력하거나 (예 : X90) 파라미터 설정을 통해 간접적으로 (예 : X=R1*(R2+R3)) 입력할 수 있습니다 .

로터리 축 위치를 사용하여 공구 방향 프로그래밍

또한 로터리 축 위치를 사용하여 공구 방향을 지정할 수 있습니다 . 예를 들면 다음 구조를 사용합니다 :

N... X=<...> Y=<...> Z=<...> A=<...> B=<...> C=<...> THETA=<...> F=<...>

이 경우 압축은 대형 반경 원형 보간 실행 여부에 따라 두 가지 방법으로 실행됩니다. 대형 반경 원형 보간이 실행되지 않을 경우 방향에서 압축된 변경은 로터리 축의 축 다항식을 통해 일반적인 방법으로 표시됩니다 .

형상 정확도

선택한 압축 모드 (MD20482 $MC_COMPRESSOR_MODE) 에 따라 셋팅 데이터를 사용하여 설정된 구성된 축 관련 허용 오차 (MD33100 $MA_COMPRESS_POS_TOL) 또는 다음 채널 관련 허용 오차가 압축을 위한 형상 축 및 방향 축에 적용됩니다 :

SD42475 $SC_COMPRESS_CONTUR_TOL(최대 형상 편차)

SD42476 $SC_COMPRESS_ORI_TOL(공구 방향에 대한 최대 각도 편차)

SD42477 $SC_COMPRESS_ORI_ROT_TOL(공구의 회전 각도에 대한 최대 각도 편차) (6 축 기계에서만 사용 가능)

참고 문서 :
작동 설명서 , 기본 기능 ; 3- ~ 5- 축 변환 (F2)
장 : " 방향 압축 "

활성화 / 비활성화

콤프레서 기능은 모달 G 코드 COMPON, COMPCURV 또는 COMPCAD 를 사용하여 활성화됩니다 .

COMPOF 는 콤프레서 작동을 종료합니다 .

"NC 블록 압축 (COMPON, COMPCURV, COMPCAD, COMPOF) [쪽 253]" 을 참조하십시오 .

주

방향 이동은 대형 반경 원형 보간이 활성화되는 경우에만 압축됩니다 (예를 들어 공구 방향은 시작 및 종료 방향에 의해 결정되는 평면에서 변경됨).

대형 반경 원형 보간은 다음 조건 하에서 실행됩니다 :

- MD21104 $MC_ORI_IPO_WITH_G_CODE = 0,
  ORIWKS 가 활성화되고
  방향이 벡터로 프로그래밍됩니다 (A3, B3, C3 또는 A2, B2, C2 사용).
- MD21104 $MC_ORI_IPO_WITH_G_CODE = 1 및
  ORIVECT 또는 ORIPLANE 이 활성화됩니다 .

  공구 방향은 방향 벡터 또는 로터리 축 위치로 프로그래밍할 수 있습니다 . G 코드 ORICONxx 또는 ORICURVE 중 하나가 활성화되거나 방향 각도 다항식 (PO[PHI] 및 PO[PSI]) 가 프로그래밍되면 대형 원형 보간은 실행되지 않습니다 .

예제

아래 프로그램 예에서 다각형 정의에 의해 구성된 원이 압축됩니다 . 동시에 공구 방향이 테이퍼 외부에서 이동합니다 . 프로그래밍된 방향 변경이 차례대로 실행되지만 불안정하게 콤프레서 작동에 의해 방향 유연화 이동이 생성됩니다 .

프로그래밍	코멘트
DEF INT NUMBER=60	
DEF REAL RADIUS=20	
DEF INT COUNTER	
DEF REAL ANGLE	
N10 G1 X0 Y0 F5000 G64	
$SC_COMPRESS_CONTUR_TOL=0.05	; 형상의 최대 편차 = 0.05mm
$SC_COMPRESS_ORI_TOL=5	; 방향의 최대 편차 = 5 도
TRAORI	
COMPCURV	
	; 이동은 다각형에서 생성된 원을 설명합니다. 방향은 Z축을 따라 45 도의 개방 각도로 테이퍼에서 이동합니다 .
N100 X0 Y0 A3=0 B3=-1 C3=1	
N110 FOR COUNTER=0 TO NUMBER	
N120 ANGLE=360*COUNTER/NUMBER	
N130 X=RADIUS*cos(각도) Y=RADIUS*sin(각도)	
A3=sin(각도) B3=-cos(각도) C3=1	
N140 ENDFOR	

6.7 오리엔테이션 특성 스무딩 (ORISON, ORISOF)

기능

' 오리엔테이션 특성 스무딩 (ORISON)' 기능은 여러 블록에서 오리엔테이션에 영향을 미치는 오실레이션을 스무딩하는 데 사용할 수 있습니다. 목적은 오리엔테이션과 형상 모두의 특성을 평탄하게 다듬는 것입니다 .

조건

' 오리엔테이션 특성 스무딩 (ORISON)' 기능은 5 축 /6 축 변환이 있는 시스템에서만 사용할 수 있습니다 .

구문

```
ORISON
...
ORISOF
```

의미

ORISON:	오리엔테이션 특성 스무딩 ON	
	적용 :	모달
ORISOF:	오리엔테이션 특성 스무딩 OFF	
	적용 :	모달

셋팅 데이터

오리엔테이션 특성은 다음 조건에 따라 스무딩됩니다 .

- 미리 정의된 최대 공차 (공구 오리엔테이션의 최대 각도 편차)

  및

- 미리 정의된 최대 경로 거리

셋팅 데이터에서는 다음의 지정이 정의됩니다 .

- SD42678 $SC_ORISON_TOL (오리엔테이션 특성 스무딩을 위한 공차)
- SD42680 O$SC_ORISON_DIST (오리엔테이션 특성 스무딩을 위한 경로 거리)

예제

프로그램 코드	설명
...	
TRAORI()	; 오리엔테이션 변환 활성화
ORISON	; 오리엔테이션 스무딩 활성화
$SC_ORISON_TOL=1.0	; 오리엔테이션 공차 스무딩 = 1.0 도
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	; 오리엔테이션 스무딩 비활성화
...	

오리엔테이션은 -45 도에서 +45 도까지 XZ 평면에서 90 로 회전합니다 . 오리엔테이션 특성 스무딩으로 인해 오리엔테이션은 더 이상 -45 도나 +45 도의 최대 각도 값에 도달할 수 없습니다 .

추가 정보

블록 개수

오리엔테이션 특성은 머신 데이터 MD28590 $MC_MM_ORISON_BLOCKS 에 설정된 블록 개수에 맞추어 스무딩됩니다 .

주

메모리 블록을 충분히 지정하지 않은 상태로 (MD28590 < 4) ORISON 을 사용하여 오리엔테이션 특성 스무딩을 활성화하면 알람 신호가 출력되고 기능을 실행할 수 없게 됩니다 .

최대 블록 경로 길이

오리엔테이션 특성은 설정된 최대 블록 경로 길이 (MD20178 $MC_ORISON_BLOCK_PATH_LIMIT) 보다 이송 거리가 짧은 블록에서만 스무딩됩니다 . 이송 거리가 더 긴 블록은 스무딩을 중단하며 프로그래밍된 대로 이송됩니다 .

6.8 운동학적 변형

6.8.1 선삭 가공된 부품에 대한 밀링 (TRANSMIT)

기능

TRANSMIT 함수는 다음을 실행할 수 있습니다 .

- 선삭 가공 클램프에서 선삭 가공된 부품에 대한 페이스 밀링 (드릴 구멍 , 형상).
- 이러한 가공 작업을 프로그램하는 데 직교 좌표계가 사용됩니다 .
- 제어 시스템은 직교 좌표계의 프로그램된 이송이 후방 가공 축 (경사 절입 축 , 일반적인 경우) 의 이송에 매핑됩니다 .
  - 로터리 축
  - 로터리 축과 수직 방향인 절입 축
  - 로터리 축과 평행한 수직 방향 축
  - 선형 축은 서로 간에 수직 방향으로 위치합니다 .
- 선삭 가공 중심 기준 공구 중심 옵셋이 허용됩니다 .
- 속도 제어에서는 회전에 대해 정의된 제한값을 고려합니다 .

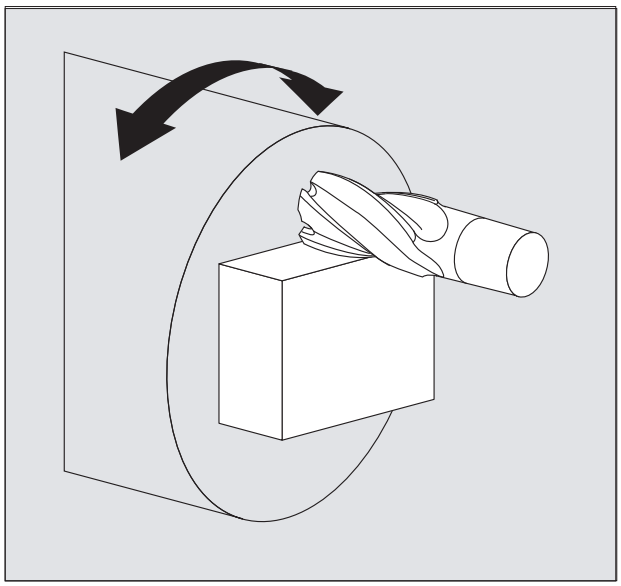


그림 6-5

TRANSMIT 변환 형식

TRANSMIT 가공 작업에는 두 개의 파라미터 형식이 있습니다 :

- TRANSMIT (TRAFO_TYPE_n = 256) 을 통한 표준 사례
- TRANSMIT 추가 Y 선형 축 사용 (TRAFO_TYPE_n = 257)

예를 들어 확장 변환 형식 257 을 사용하여 실제 Y 축에서 공구의 클램핑 보정을 보정할 수 있습니다 .

구문

TRANSMIT 또는 TRANSMIT(n)

TRAFOOF

로터리 축

로터리 축은 형상 축에 의해 둘러 싸여 있기 때문에 프로그래밍할 수 없으며 따라서 채널 축으로 직접 프로그래밍할 수 없습니다 .

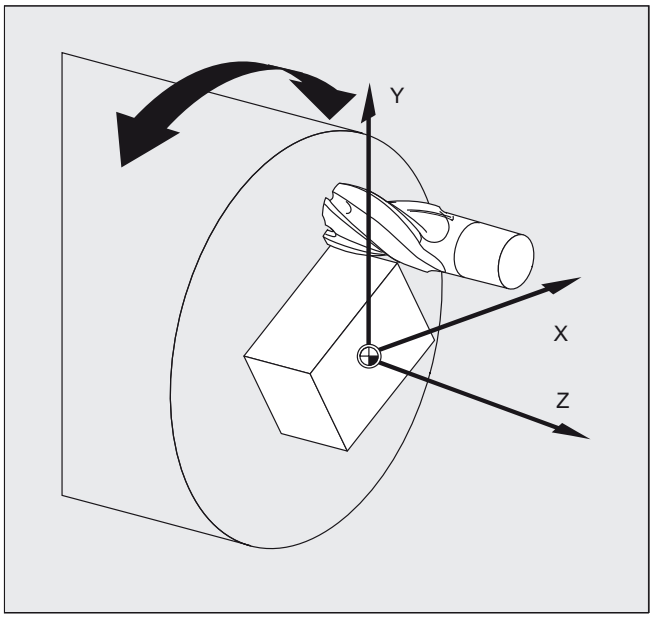
의미

TRANSMIT:	첫 번째로 선언된 TRANSMIT 기능을 활성화합니다 . 이 기능도 극 변환으로 지정됩니다 .
TRANSMIT(n):	n 번째 선언된 TRANSMIT 기능을 활성화합니다 . n 의 최대값은 2 입니다 (TRANSMIT(1) 은 TRANSMIT 와 동일함).
TRAFOOF:	활성 변환을 비활성화합니다 .
OFFN:	옵셋 형상 법선 : 프로그래밍된 기준 형상으로부터 페이스 가공 거리 .

주

활성 TRANSMIT 변환은 다른 변환 중 하나가 관련 채널 (예를 들어 TRACYL, TRAANG, TRAORI) 에서 활성화된 경우 비활성화됩니다 .

예제



프로그램 코드	비고
N10 T1 D1 G54 G17 G90 F5000 G94	; 공구 선택
N20 G0 X20 Z10 SPOS=45	; 시작 위치에 접근
N30 TRANSMIT	; TRANSMIT 기능 활성화
N40 ROT RPL=–45 N50 ATRANS X–2 Y10	; 프레임 설정
N60 G1 X10 Y–10 G41 OFFN=1OFFN N70 X–10 N80 Y10 N90 X10 N100 Y–10	; 평방 황삭 ; 1mm 허용 오차
N110 G0 Z20 G40 OFFN=0 N120 T2 D1 X15 Y–15 N130 Z10 G41	; 공구 교환
N140 G1 X10 Y–10 N150 X–10 N160 Y10 N170 X10 N180 Y–10	; 평방 정삭
N190 Z20 G40 N200 TRANS N210 TRAFOOF	; 프레임 선택 취소
N220 G0 X20 Z10 SPOS=45	; 시작 위치에 접근
N230 M30	

설명

극 좌표

극 좌표를 통과하는 두 가지 방법은 다음과 같습니다 :

- 선형 축을 따라 이송
- 극 좌표로 이송하고 극 좌표에서 로타리 축을 회전하면서 극 좌표에서 이탈

MD 24911 및 24951 을 사용하여 선택합니다 .

추가 Y 선형 축을 통한 TRANSMIT(변환 형식 257):

극 좌표 변환 중 이 변환 형식은 향상된 공구 보정을 수행하기 위해 다른 선형 축에서 기계의 중복성을 사용합니다 . 다음 조건이 두 번째 선형 축에 적용됩니다 :

- 소형 작업 영역 및
- 두 번째 선형 축은 가공 프로그램의 후퇴에서 사용하지 않아야 합니다 .

가공 프로그램 및 BCS 또는 MCS 에서 해당 축 지정에 대해 특정 기계 데이터 설정이 고려됩니다 .

참조 문서

/FB2/ 작동 설명서 , 확장된 기능 ; 운동학적 변환 (M1) 을 참조하십시오 .

6.8.2 원통 표면 변환 (TRACYL)

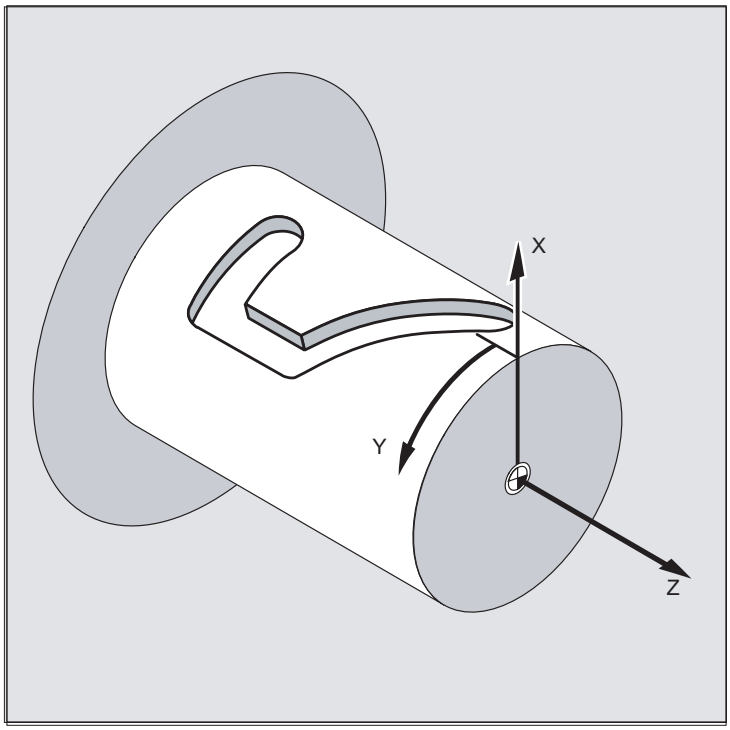
기능

다음을 위해 TRACYL 실린더 표면 변환 기능을 사용할 수 있습니다 :

기계

- 실린더 본체의 수직 홈 ,
- 실린더에 비스듬한 홈을 가공할 때
- 실린더 본체의 임의 경로를 따르는 홈 .

홈의 경로는 실린더 표면을 펼친 상태의 수평 표면을 기준으로 프로그래밍됩니다 .



TRACYL 변환 형식

세 가지 형식의 실린더 표면 좌표 변환은 다음과 같습니다 :

- 홈 벽면 옵셋이 없는 TRACYL(TRAFO_TYPE_n=512)
- 홈 벽면 옵셋이 있는 TRACYL: (TRAFO_TYPE_n=513)
- 추가 직선 축 및 홈 벽면 옵셋이 있는 TRACYL: (TRAFO_TYPE_n=514)
  홈 벽면 옵셋은 세 번째 파라미터를 사용하여 TRACYL 을 통해 설정됩니다 .

홈 측면 보정을 통한 실린더 커브 변환의 경우 프로그래밍된 홈 중심선에 대한 홈 중심이 정삭되도록 보정에 사용된 축이 0 (y=0) 에 위치해야 합니다 .

축 활용

다음 축의 포지셔닝 축 또는 왕복 이동 축으로 사용할 수 없습니다 :

- 실린더 표면의 주변 방향에 있는 기하 축 (Y 축)
- 홈 측면 보정에 대한 추가 직선 축 (Z 축)

구문

TRACYL(d) 또는 TRACYL(d, n) 또는

변환 형식 514 의 경우

TRACYL(d, n, groove side offset)

TRAFOOF

로터리 축

로터리 축이 기하 축으로 되어 있는 한 프로그램 할 수 없으며 , 따라서 채널 축으로 직접 프로그래밍할 수 없습니다 .

의미

TRACYL(d)	채널 머신 데이터에서 지정된 첫 번째 TRACYL 기능을 활성화합니다 . d 는 직경에 대한 파라미터입니다 .
TRACYL (d, n)	채널 머신 데이터에서 지정된 n 번째 TRACYL 기능을 활성화합니다 . n 의 최대값은 2 이고 TRACYL(d,1) 은 TRACYL(d) 에 해당합니다 .
D	직경 값. 직경은 공구 팁 및 선삭 중심 사이 거리의 두 배입니다. 이 직경을 항상 지정해야 하며 1 보다 커야 합니다 .
n	TRACYL 데이터 블록 1 (미리 선택) 또는 2에 대한 두 번째 옵션 파라미터 .
슬롯 측면 보정	TRACYL 에 대한 세 번째 옵션 파라미터의 값은 머신 데이터 모드를 사용하여 미리 선택됩니다 . 값 범위 : 0: 홈 벽면 옵셋이 없는 변환 형식 1: 홈 벽면 옵셋이 있는 변환 형식 514
TRAFOOF	변환 OFF (BCS 및 MCS 는 다시 한번 일치).
OFFN	옵셋 형상 법선: 프로그래밍된 기준 형상으로부터 홈 측면 거리.

주

활성 TRACYL 변환은 채널에서 다른 변환 중 (예를 들어 TRANSMIT, TRAANG, TRAORI) 하나가 활성화된 경우 비활성화됩니다 .

예제 : 공구 정의

다음 예제는 TRACYL 실린더 변환의 설정 테스트에 적합합니다 :

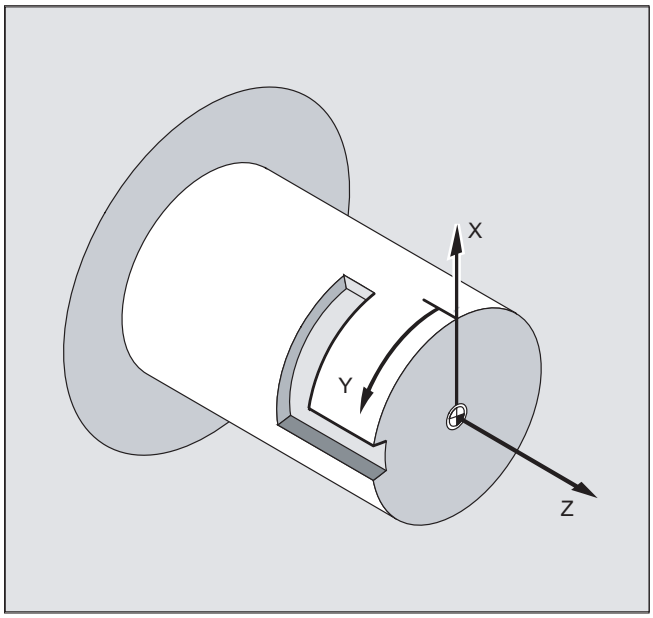
프로그램 코드	설명	
공구 파라미터 번호 (DP)	의미	설명
$TC_DP1[1,1]=120	공구 종류	밀링 공구
$TC_DP2[1,1]=0	공구 끝 위치	선삭 공구에만 해당

프로그램 코드	설명	
기하	길이 보정	
$TC_DP3[1,1]=8.	길이 옵셋 벡터	형식 및 평면에 따른
$TC_DP4[1,1]=9.		계산
$TC_DP5[1,1]=7.		

프로그램 코드	설명	
기하	반경	
$TC_DP6[1,1]=6.	반경	공구 반경
$TC_DP7[1,1]=0	슬롯 톱의 경우 슬롯 폭 b, 밀링 공구의 경우 라운딩 반경	
$TC_DP8[1,1]=0	돌출부 k	슬롯 톱에만 해당
$TC_DP9[1,1]=0		
$TC_DP10[1,1]=0		
$TC_DP11[1,1]=0	테이퍼 밀링 공구 각도	

프로그램 코드	설명	
마모량	공구 길이 및 반경 보정	
$TC_DP12[1,1]=0	$TC_DP24=0 에 대한 나머지 파라미터	공구 베이스 치수 / 어댑터

예제 : 홈 모양의 홈 만들기



실린더 표면 변환 활성화 :

프로그램 코드	설명
N10 T1 D1 G54 G90 F5000 G94	; 공구 선택 , 클램핑 보정 ,
N20 SPOS=0 N30 G0 X25 Y0 Z105 CC=200	; 시작 위치에 접근
N40 TRACYL (40)	; 실린더 표면 변환 활성화
N50 G19	; 평면 선택

홈 모양의 홈 가공 :

프로그램 코드	설명
N60 G1 X20	; 홈 베이스로 공구 절입
N70 OFFN=12	; 홈 중심선을 기준으로 12 mm 홈 측면 공간 정의
N80 G1 Z100 G42	; 홈 우측 면 접근
N90 G1 Z50	; 실린더 축에 평행하게 홈 절삭
N100 G1 Y10	; 원주에 평행하게 홈 절삭
N110 OFFN=4 G42	; 홈 좌측 면 접근 ; 홈 중심선을 기준으로 4mm 홈 측면 공간 정의
N120 G1 Y70	; 원주에 평행하게 홈 절삭
N130 G1 Z100	; 실린더 축에 평행하게 홈 절삭
N140 G1 Z105 G40	; 홈 벽면에서 후진
N150 G1 X25	; 후퇴
N160 TRAFOOF	
N170 G0 X25 Y0 Z105 CC=200	; 시작 위치에 접근
N180 M30	

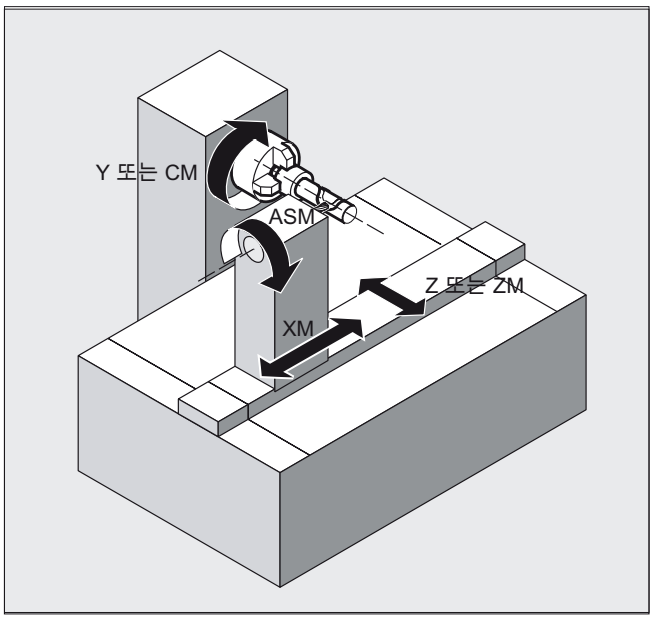
설명

홈 벽면 옵셋이 없는 경우 (변환 형식 512):

시스템이 원통 좌표계의 프로그램된 이송을 실제 기계 축의 이송으로 변환합니다 .

- 로타리 축
- 로타리 축과 수직 방향인 절입 축
- 로타리 축과 평행한 수직 방향 축

선형 축은 서로 간에 수직 방향으로 위치합니다 . 절입 축이 로타리 축을 절삭합니다 .

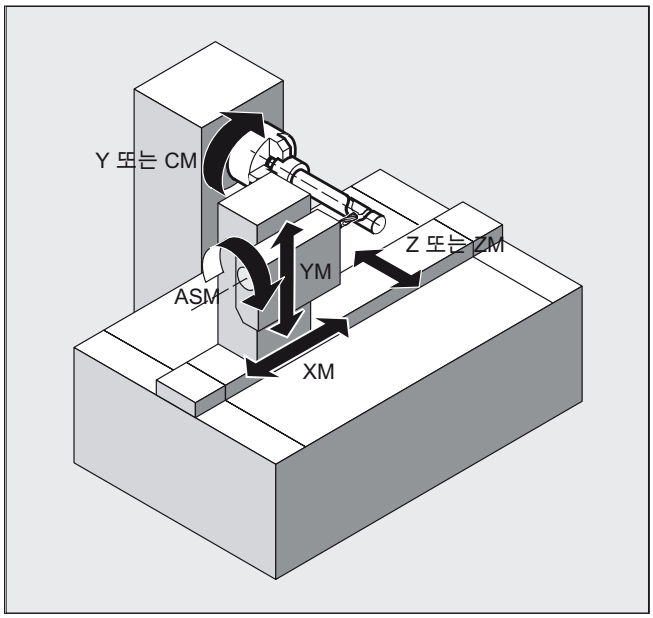


홈 벽면 옵셋이 있는 경우 (변환 형식 513):

위와 동일한 운동 구조를 가졌지만 추가 수직 방향 축이 주변 방향과 평행합니다 .

선형 축은 서로 간에 수직 방향으로 위치합니다 .

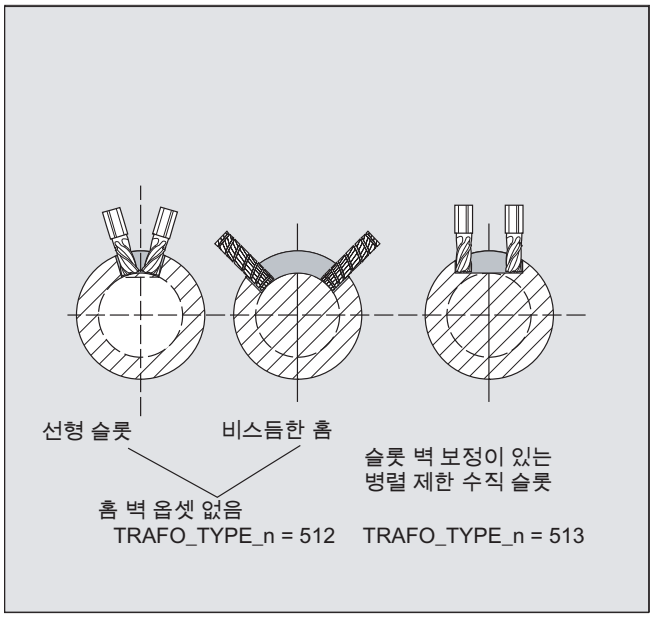
속도 제어에서는 회전에 대해 정의된 제한값을 고려합니다 .



홈 이송 단면

축 구성 1 의 경우 로타리 축을 따라 수직 방향 홈은 홈 너비가 정확하고 공구 반경과 일치하는 경우에만 평행 제한이 적용됩니다 .

주변과 평행한 홈 (이송 홈) 은 시작 및 끝에서 평행하지 않습니다 .



추가 선형 축 및 홈 벽면 옵셋이 있는 경우 (변환 형식 514):

두 번째 선형 축을 가진 기계에서 이 변환 변용은 향상된 공구 보정을 수행하기 위해 중복성을 사용합니다 . 다음 조건이 두 번째 선형 축에 적용됩니다 :

- 소형 작업 영역 및
- 두 번째 선형 축은 가공 프로그램 통과 시 사용하지 않아야 합니다 .

가공 프로그램 및 BCS 또는 MCS 에서 해당 축 지정에 대해 특정 기계 데이터 설정이 고려됩니다 .

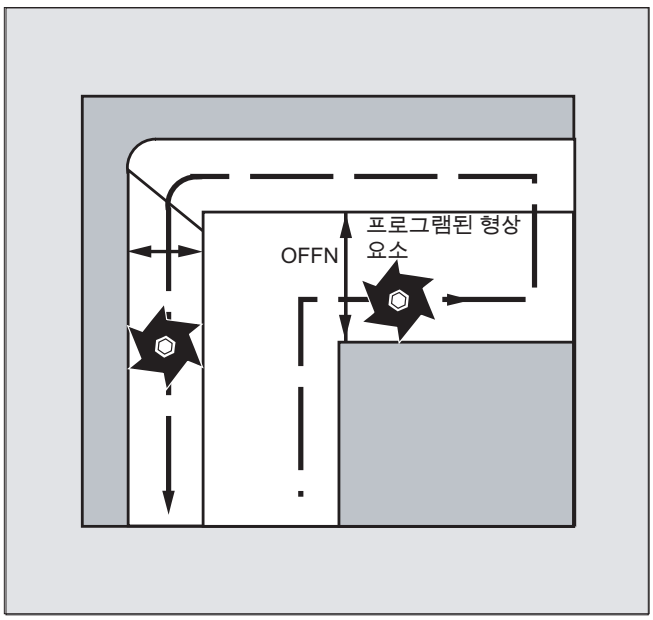
참조 문서

/FB2/ 작동 설명서 , 확장된 기능 ; 운동학적 변환 (M1) 을 참조하십시오 .

옵셋 형상 법선 OFFN(변환 형식 513)

TRACYL 을 통해 홈을 밀링 가공하기 위해 다음이 프로그래밍됩니다 :

- 가공 프로그램에서 홈 중심선 ,
- OFFN 을 사용하여 프로그래밍된 **홈 너비 절반** .



홈 밀링 가공 프로그램은 일반적으로 다음과 같은 단계로 구성됩니다 .

1. 공구 선택
2. TRACYL 을 선택합니다 .
3. 적절한 좌표 옵셋을 선택합니다 (프레임).
4. 위치
5. OFFN 을 프로그램합니다 .
6. TRC 를 선택합니다 .

7. 블록에 접근합니다 (TRC 위치 지정 및 홈 측면 접근).
8. 홈 중심선 형상
9. TRC 의 선택을 취소합니다 .
10. 블록을 후진시킵니다 (TRC 후진 및 홈 측면으로부터 멀리 이동).
11. 위치
12. TRAFOOF
13. 최초 좌표 전환을 다시 선택합니다 (프레임).

특징

- TRC 선택 :

  TRC 는 홈 측면이 아닌 프로그래밍된 홈 중심선을 기준으로 프로그래밍됩니다 . 공구가 홈 측면 좌측으로 이동되는 것을 방지하기 위해 G41 대신 G42 가 입력됩니다 . OFFN 에서 홈 너비를 음수로 입력하면 이것을 방지할 수 있습니다 .
- TRACYL 을 사용하지 않을 때와 TRACYL 을 사용할 때 OFFN 이 다르게 작동합니다 . TRACYL을 사용하지 않더라도 TRC가 활성화된 경우 OFFN이 포함되어 있으므로 OFFN이 TRAFOOF 이후에 0 으로 재설정되어야 합니다 .
- 가공 프로그램 안에서 OFFN 을 변경할 수 있습니다 . 이것은 중심에서 홈 중심선을 이동하는 데 사용할 수 있습니다 (그림 참조).
- 홈 가이딩 :

  TRACYL 은 홈 너비를 형성하는 직경을 가진 공구와 관련되기 때문에 홈 가이딩을 위한 동일한 홈을 만들지 않습니다 . 기본적으로 큰 원통 공구를 사용하기 때문에 작은 원통 공구와 동일한 홈 측면 형상을 만들 수 없습니다 . TRACYL 은 오류를 최소화합니다 . 정확성이 저하되는 문제를 방지하려면 공구 반경이 홈 너비보다 약간만 작아야 합니다 .

주

OFFN 및 TRC

TRAFO_TYPE_n = 512 에서 값은 TRC 의 허용으로 OFFN 에서 적용됩니다 .

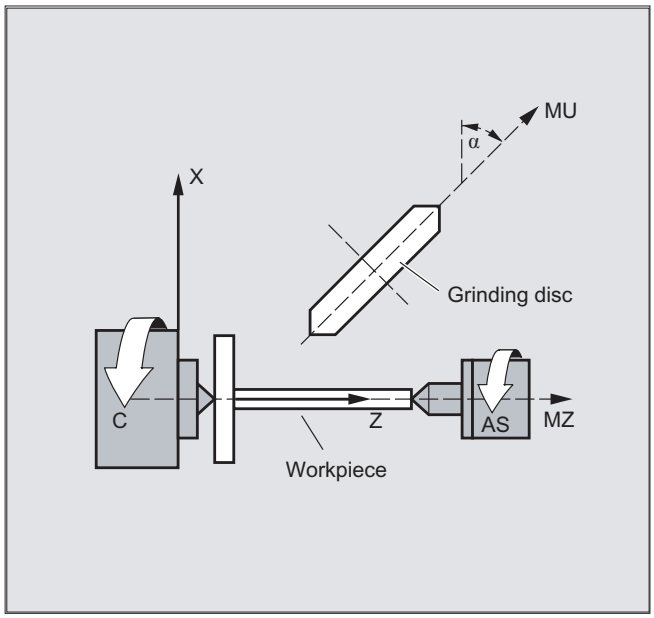
TRAFO_TYPE_n = 513 조건에서 홈 너비의 절반이 OFFN 에 프로그래밍됩니다 . 형상은 OFFN-TRC 를 통해 후퇴합니다 .

6.8.3 경사 축 (TRAANG)

기능

경사 축 기능은 그라인딩에 사용되며 다음 성능에 도움이 됩니다 :

- 사면 절입 축을 사용하여 가공
- 직교 좌표계는 프로그램 목적으로 사용할 수 있습니다 .
- 제어 시스템은 직교 좌표계의 프로그램된 이송이 후방 가공 축 (경사 절입 축 , 일반적인 경우) 의 이송에 매핑됩니다 . 경사 절입 축 .



구문

TRAANG(α) 또는 TRAANG(α, n)

TRAFOOF

의미

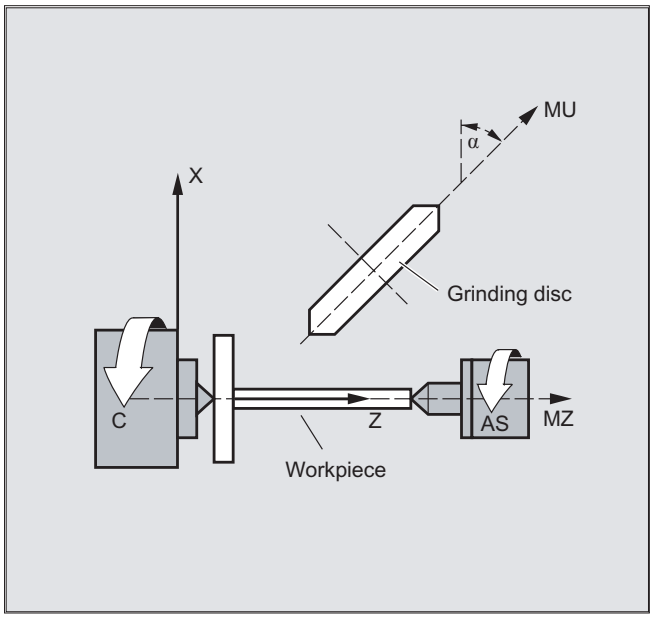
TRAANG( ) 또는 TRAANG(,n)	이전 파라미터 설정을 사용하여 변환을 활성화합니다 .
TRAANG(α)	지정된 첫 번째 경사 축 변환을 활성화합니다 .
TRAANG(α,n)	합의된 n 번째 경사 축 변환을 활성화합니다 . n 의 최대값은 2 입니다 . TRAANG(α,1) 은 TRAANG(α) 에 해당합니다 .
αA	경사 축 각도 사용 가능한 α 값은 -90 도 < α < + 90 도입니다 .
TRAFOOF	변환 해제
n	합의된 변환 개수

각도 α 생략 또는 0

α (각도) 가 생략된 경우 (예 : TRAANG(), TRAANG(, n)), 변환은 이전 파라미터 설정을 통해 활성화됩니다 . 첫 번째 선택에는 머신 데이터에 따라 기본 설정이 적용됩니다 .

각도 α = 0 (예 : TRAANG(0), TRAANG(0,n)) 은 유효한 파라미터 설정이며 이전 버전의 경우처럼 파라미터 생략에 해당하는 것은 없습니다 .

예제



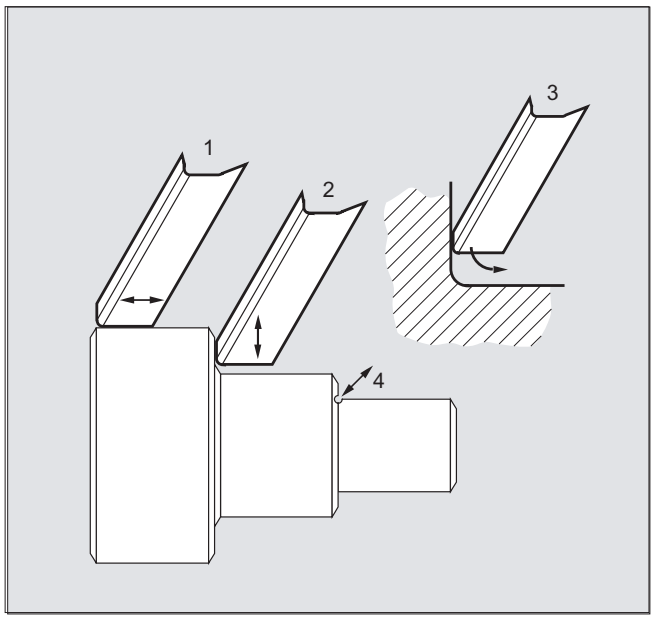
프로그램 코드	설명
N10 G0 G90 Z0 MU=10 G54 F5000 -> -> G18 G64 T1 D1	; 공구 선택 , 클램핑 보정 , 평면 선택
N20 TRAANG(45)	; 경사 축 변환 활성화
N30 G0 Z10 X5	; 시작 위치에 접근
N40 WAITP(Z)	; 오실레이션을 위한 축 분리
N50 OSP[Z]=10 OSP2[Z]=5 OST1[Z]=–2 -> -> OST2[Z]=–2 FA[Z]=5000 N60 OS[Z]=1 N70 POS[X]=4.5 FA[X]=50 N80 OS[Z]=0	; 치수에 도달할 때까지 오실레이션 (오실레이션에 대해서는 " 오실레이션 " 장 참조)
N90 WAITP(Z)	; 포시셔닝 축으로 오실레이션 축 분리
N100 TRAFOOF	; 변환 비활성화
N110 G0 Z10 MU=10	; 후퇴
N120 M30	;

-> 싱글 블록에서 프로그래밍

설명

다음 가공 작업을 수행할 수 있습니다 .

1. 수직 연삭
2. 단면 연삭
3. 특정 형상 연삭
4. 사면 플런지 - 절삭 연삭



장비 제조업체

다음 설정은 머신 데이터에 정의되어 있습니다 .

- 기계 축과 사면 축 사이의 각도
- " 경사 축 " 기능을 통해 지정된 좌표계의 원점에 대한 공구의 영점 위치
- 보정 이동 시 가로 축에 대한 속도는 예약되어 있습니다 .
- 보정 이동 시 가로 축에 대한 축 가속도는 예약되어 있습니다 .

축 구성

직교 좌표계에서 프로그램하려면 이 좌표계와 실제 기존 기계 축 (MU,MZ) 간 상관 관계를 제어 시스템에 알려 주어야 합니다 .

- 형상 축에 이름 지정
- 채널 축에 기하 축 할당
  - 일반 경우 (경사 축이 활성화되지 않음)
  - 활성화된 경사 축
- 기계 축 번호에 채널 축 지정
- 스핀들 식별
- 기계 축 이름 지정 .

이 절차는 " 활성화된 경사 축 " 을 제외하면 일반 축 구성 절차에 해당합니다 .

6.8.4 경사 축 프로그래밍 (G05, G07)

작동

조그 모드에서는 연삭 휠의 이동은 직교 방향이거나 경사 축 방향입니다 (화면은 직교 방향으로 유지됨). 이동되는 부분은 후방 U 축이며 , Z 축 화면이 업데이트됩니다 .

조그 모드에서 직교 좌표를 사용하여 REPOS 옵셋을 이송해야 합니다 .

"PTP 이송 " 이 활성화된
조그 모드에서 직교 조작 범위 제한은 초과 이동에 대해 모니터링되며 관련 축이 먼저 끊깁니다 . "PTP 이송 " 이 활성화되지 않은 경우에는 축을 조작 범위 제한까지 오른쪽으로 이송할 수 있습니다 .

참조 문서

/FB2/ 기능 설명 확장 기능 ; 운동학적 변환 (M1)

구문

G07

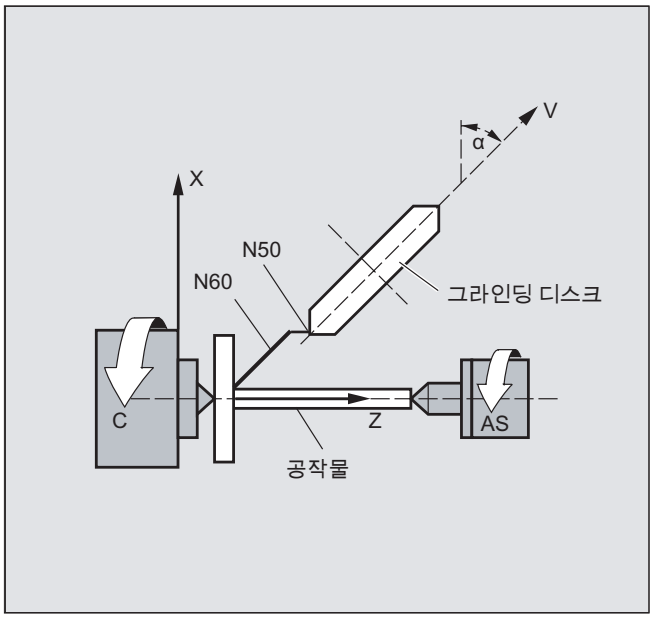
G05

G07/G05 명령을 사용하면 경사 축을 보다 쉽게 프로그램할 수 있습니다. 위치를 프로그램하여 직교 좌표계에 표시할 수 있습니다 . 직교 좌표에 공구 보정 및 제로 옵셋이 포함됩니다 . NC 프로그램에서 경사 축 각도를 프로그램한 후 시작 위치에 접근한 다음 (G07) 사면 플런지 절삭 (G05) 을 수행할 수 있습니다 .

의미

G07	시작 위치로 접근
G05	사면 플런지 절삭을 활성화합니다 .

예제



프로그래밍		비고
N.. G18	;	경사 축 각도 프로그램
N50 G07 X70 Z40 F4000	;	시작 위치로 접근
N60 G05 X70 F100	;	사면 플런지 절삭
N70 ...	;	

6.9 직교 PTP 이송

작동

이 기능을 사용하여 직교 좌표계에서 위치를 프로그래밍할 수 있지만 기계 이동은 기계 좌표에서 발생합니다 . 예를 들어 연결 조인트의 위치를 변경할 때 이동이 단독으로 실행되는 경우 이 기능을 사용할 수 있습니다 .

주

활성 변환과 연동하는 경우에만 함수를 의미있게 사용할 수 있습니다. 또한 "PTP 이송"은 G0 및 G1 과 연동할 경우에만 허용됩니다 .

구문

N... TRAORI

N... STAT='B10' TU='B100' PTP

N... CP

일반 5/6 축 변환을 통한 PTP 이송

PTP 를 통한 활성 일반 5/6 축 변환 동안 기계 좌표계 (ORIMKS) 에서 종단 간 이송이 실행되는 경우 공구 방향은 로타리 축 위치를 통해 프로그래밍할 수 있습니다 .

N... G1 X Y Z A B C

운동 구조와 관계 없이 Euler 및 / 또는 RPY 각도 벡터를 통해

N... ORIEULER 또는 ORIRPY

N... G1 X Y Z A2 B2 C2

또는 방향 벡터

N... G1 X Y Z A3 B3 C3

이 프로그래밍됩니다 . 로타리 축 보간 , 대형 원 보간을 통한 벡터 보간 ORIVECT 또는 테이퍼의 주변 표면에 대한 방향 벡터 보간 ORICONxx 를 수행할 수 있습니다 .

벡터를 통한 방향의 비고유성

벡터를 통해 방향을 프로그래밍하는 경우 사용 가능한 로타리 축 위치가 고유하지 않습니다 . 접근할 로타리 축 위치는 STAT = <...> 프로그래밍을 통해 선택할 수 있습니다 . 만약

STAT = 0 을 프로그래밍하면 (기본 설정)
시작 위치에서 가장 짧은 거리에 있는 위치에 접근합니다 . 만약

STAT = 1 을 프로그래밍하면
시작 위치에서 가장 먼 거리에 있는 위치에 접근합니다 .

의미

PTP 및 CP 명령은 모달식으로 실행됩니다 . CP 는 기본 설정입니다 .

STAT 값 프로그래밍 시 모달이 적용되면 TU 프로그래밍은 = <...> 비모달입니다 .

또 다른 차이점은 STAT 값을 프로그래밍하는 경우에만 벡터 보간 동안 영향을 주는 반면에 TU 프로그래밍은 활성 로타리 축 보간 동안 평가됩니다 .

PTP	point to point(종단 간 이동) 이동은 동기화된 축 이동으로 실행됩니다 . 이동에 관련된 가장 느린 축은 속도를 제어하는 축입니다 .
CP	continuous path(경로 이동) 이동은 직교 경로 이동으로 실행됩니다 .
STAT=	연결 조인트의 위치 ; 이 값은 변환에 종속됩니다 .
TU=	TURN 정보는 블록처럼 작동합니다 . 이를 통해 -360 도 및 +360 도 사이에서 확실하게 축 각도에 접근할 수 있습니다 .

예제

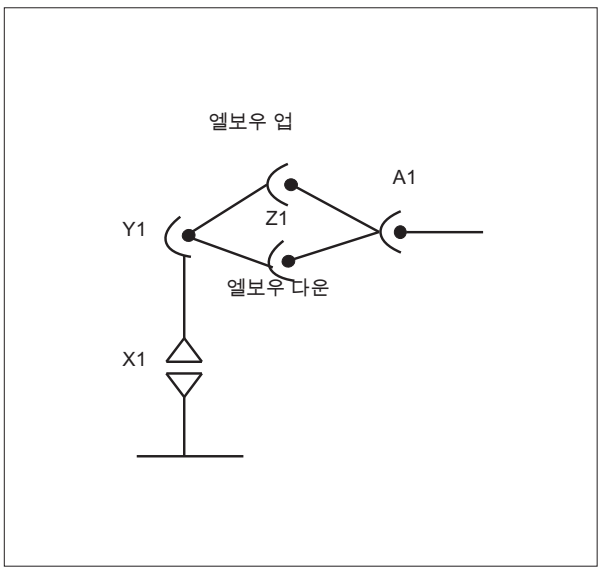


그림 6-6

N10 G0 X0 Y-30 Z60 A-30 F10000	초기 설정 → 엘보우 업
N20 TRAORI(1)	변환 설정
N30 X1000 Y0 Z400 A0	
N40 X1000 Z500 A0 STAT='B10' TU='B100' PTP	변환 없이 방향 재지정 → 엘보우 다운
N50 X1200 Z400 CP	변환 활성 다시 실행
N60 X1000 Z500 A20	
N70 M30	

일반 5 축 변환을 통한 PTP 이송

가정 : 우향 각도 CA 운동에 기반합니다 .

프로그램 코드	비고
TRAORI	; 변환 CA 운동 실행
PTP	; PTP 이송 활성화
N10 A3 = 0 B3 = 0 C3 = 1	; 로타리 축 위치 C = 0 A = 0
N20 A3 = 1 B3 = 0 C3 = 1	; 로타리 축 위치 C = 90 A = 45
N30 A3 = 1 B3 = 0 C3 = 0	; 로타리 축 위치 C = 90 A = 90
N40 A3 = 1 B3 = 0 C3 = 1 STAT = 1	; 로타리 축 위치 C = 270 A = –45

로타리 축 위치에 대한 명확한 접근 위치를 선택합니다 :

블록 N40 에서 STAT = 1 로 프로그래밍하면 로타리 축은 시작 지점 (C=90, A=90) 에서 끝점 (C=270, A=–45) 사이 긴 경로를 이송하고 STAT = 0 으로 프로그래밍할 경우 끝점 (C=90, A=45) 까지의 가장 짧은 경로를 이송합니다 .

설명

명령 PTP 및 CP 는 직교 이송 및 기계 축 이송 간의 전환에 영향을 줍니다 .

일반 5/6 축 변환을 통한 PTP 이송

PTP 이송 동안 5/6 축 변환과 달리 , 일반적으로 TCP 는 방향만 변경되는 경우 정지 상태를 유지하지 않습니다. 모든 변환 축(3개의 선형 축 및 최대 3개의 라운드 축)의 변환된 끝 위치는 실제 변환 활성 없이 선형적으로 접근됩니다 .

PTP 이송은 모달 G 코드 CP 를 프로그래밍하여 비활성화됩니다 .

다음 설명서에서 다양한 변환에 대해 설명합니다 :
/FB3/ 작동 설명서 특별한 기능 ; 변환 패키지 취급 (TE4).

위치 프로그래밍 (STAT=)

기계 위치는 직교 좌표의 위치 데이터 및 공구 방향을 통해서만 고유하게 결정되지 않습니다 . 관련 운동에 따라 8 개의 서로 다른 주요 연결 조인트 위치로 제공될 수 있습니다 . 이러한 위치는 변환과 관련됩니다 . 고유하게 직교 위치를 축 각도로 변환할 수 있으려면 명령 STAT= 를 통해 연결 조인트의 위치를 지정해야 합니다 . "STAT" 명령에는 가능한 각 위치에 대한 비트가 이진 값으로 포함되어 있습니다 .

"STAT" 에 대해 프로그래밍할 셋팅 비트에 대한 자세한 내용은 다음을 참조하십시오 :
/FB2/ 기능 설명 확장 기능 ; 운동학적 변환 (M1), " 직교 PTP 이송 " 절 .

축 각도 프로그래밍 (TU=)

축 각도 < ± 360 도로 확실하게 접근할 수 있으려면 명령 "TU=" 를 사용하여 이 정보를 프로그래밍해야 합니다 .

축은 최단 경로에 의해 이송됩니다 :

- 위치에 대해 TU 가 프로그래밍되지 않은 경우 ,
- 이송 범위 > ±360 도를 가진 축 .

예 :

그림에 표시된 대상 위치에 음 또는 양의 방향으로 접근할 수 있습니다 . 방향은 어드레스 A1 하에서 프로그래밍됩니다 .

A1=225°, TU=Bit 0, → 양의 방향

A1= − 135°, TU=Bit 1, → 음의 방향

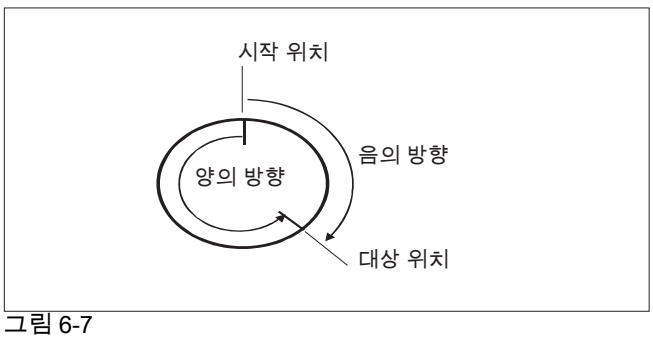


그림 6-7

일반 5/6 축 변환 및 대상 위치에 대한 TU 평가 예

변수 TU 에는 비트가 포함되어 있으며 변환과 관련된 각 축의 이송 방향을 표시합니다 . TU 비트의 할당은 라운드 축의 채널 축 보기와 일치합니다 . TU 정보는 변환에 포함된 최대 3 개의 가능한 라운드 축에 대해서만 평가됩니다 :

Bit0: 축 1, TU bit = 0 : 0 도 <= 라운드 축 각도 < 360 도

Bit1: 축 2, TU bit = 1 : –360 도 < 라운드 축 각도 < 0 도

라운드 축의 시작 위치는 C = 0 입니다 . C = 270 으로 프로그래밍하면 라운드 축이 다음 대상 위치로 이송됩니다 :

C = 270: TU 비트 0, 양의 회전 방향

C = –90: TU 비트 1, 음의 회전 방향

추가 동작

모드 변경

" 직교 PTP 이송 " 기능은 AUTO 및 MDA 작동 모드에서만 유용합니다 . JOG 모드를 변경할 경우 현재 설정이 유지됩니다 .

G 코드 PTP 를 설정하면 축이 MCS 에서 이송됩니다 . G 코드 CP 를 설정하면 축이 WCS 에서 이송됩니다 .

Power On/RESET

전원을 켜거나 리셋한 후 설정은 기계 데이터 $MC_GCODE_REST_VALUES[48] 에 따라 달라집니다 . 기본 이송 모드 설정은 "CP" 입니다 .

REPOS

중단 블록 동안 기능 " 직교 PTP 이송 " 이 설정되면 위치 재지정을 위해 PTP 를 사용할 수 있습니다 .

오버레이 이동

DRF 옵셋 또는 외부 워크 옵셋은 직교 PTP 이송에서 제한된 범위까지만 가능합니다 .
PTP 에서 CP 이동으로 변경하면 BCS 에서 재지정이 없어야 합니다 .

CP 및 PTP 이동 사이 유연화

블록 간의 프로그램 가능한 전이 라운딩은 G641 을 통해 가능합니다 .

라운딩 영역의 크기는 라운딩할 블록 전이를 기준으로 mm 또는 인치 단위의 경로입니다 .
크기를 다음과 같이 지정해야 합니다 :

- ADISPOS 를 통한 G0 블록의 경우
- ADIS 를 통한 다른 모든 이동 명령의 경우 .

경로 계산은 비 G0 블록의 F 어드레스를 고려하여 수행합니다 . 피드는 FGROUP(..) 에서 지정한 축을 기준으로 유지됩니다 .

피드 계산

CP 블록의 경우 기본 좌표계의 직교 축이 계산에 사용됩니다 .

PTP 블록의 경우 기계 좌표계의 해당 축이 계산에 사용됩니다 .

6.9.1 TRANSMIT 에 대한 PTP

작동

TRANSMIT 에 대한 PTP 는 G0 및 G1 블록을 최적의 시간에 접근하는 데 사용할 수 있습니다 . 기본 좌표계의 축을 선형적으로 이송 (CP) 하지 않고 기계 축은 선형적으로 (PTP) 이송됩니다 . 이렇게 하면 극 좌표 근처에서의 기계 축 이동으로 블록 끝점에 훨씬 빨리 접근합니다 .

가공 프로그램은 계속해서 직교 공작물 좌표계에서 기록되고 모든 좌표 옵셋 , 회전 및 프레임 프로그래밍 설정은 유효합니다 . HMI 에서의 시뮬레이션도 직교 공작물 좌표계에 표시됩니다 .

구문

N... TRANSMIT

N... PTPG0

N... G0 ...

...

N... G1 ...

의미

TRANSMIT	첫 번째로 선언된 TRANSMIT 기능을 활성화합니다 . (" 선삭 가공된 부품에 대한 밀링 : TRANSMIT" 절 참조)
PTPG0	Point to Point G0(각 G0 블록에서 종단 간 이동 자동화 및 CP 다시 설정) STAT 및 TU 가 모달식이기 때문에 가장 최근에 프로그래밍된 값이 항상 적용됩니다 .
PTP	point to point(종단 간 이동) TRANSMIT 의 경우 , PTP 는 직교 나선에서 Archimedean 나선이 극 좌표 주위 또는 극 좌표에서 후퇴하는 것을 의미합니다 . 결과적으로 공구 이동은 CP 와 매우 다르게 실행되며 관련 프로그래밍 예에 설명되어 있습니다 .
STAT=	극 좌표와 관련하여 비고유성 해결 .
TU=	TU 는 TRANSMIT 에서 PTP 와 관계 없습니다 .

PTP 및 TRANSMIT 를 통한 극 좌표 주위 이동의 예

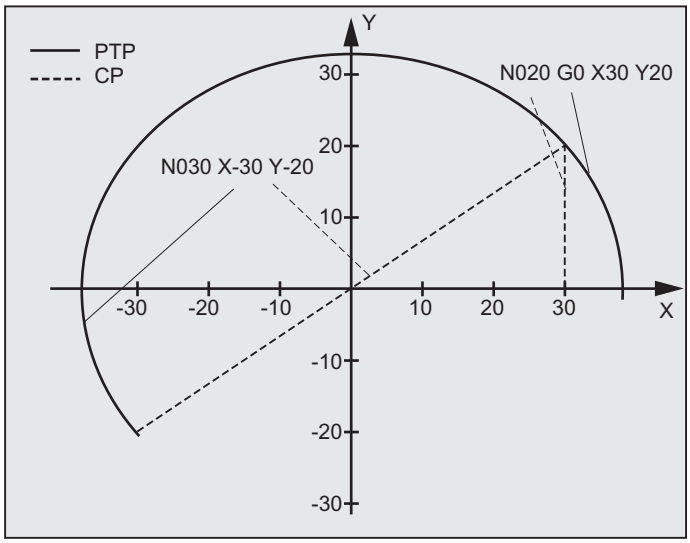


그림 6-8

프로그램 코드	비고
N001 G0 X30 Z0 F10000 T1 D1 G90	; 초기 설정 , 절대 치수
N002 SPOS=0	
N003 TRANSMIT	; 변환 TRANSMIT
N010 PTPG0	; 각 G0 블록에 대해 , 자동으로 CP 다음에 PTP 수행
N020 G0 X30 Y20	
N030 X-30 Y-20	
N120 G1 X30 Y20	
N110 X30 Y0	
M30	

PTP 및 TRANSMIT 를 통한 극 좌표에서 후퇴의 예

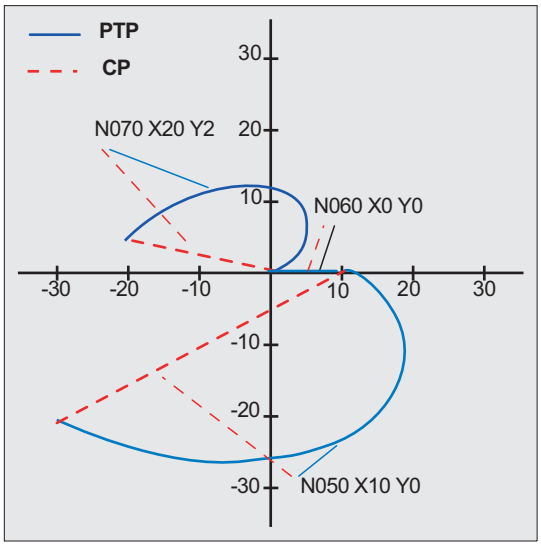


그림 6-9

프로그래밍	비고
N001 G0 X90 Z0 F10000 T1 D1 G90	; 초기 설정
N002 SPOS=0	
N003 TRANSMIT	; 변환 TRANSMIT
N010 PTPG0	; 각 G0 블록에 대해 , 자동으로 CP 다음에 PTP 수행
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	

설명

PTP 및 PTPG0

PTPG0은 PTP를 처리할 수 있는 모든 변환에 대해 고려됩니다. PTPG0은 다른 모든 경우 관련 없습니다 .

G0 블록은 CP 모드에서 처리됩니다 .

PTP 또는 PTPG0 은 가공 프로그램 또는 기계 데이터 $MC_GCODE_RESET_VALUES[48] 에서 CP 를 선택 취소하여 수행됩니다 .

⚠ 주의

제한사항

공구 이동 및 충돌과 관련하여 다음과 같은 많은 제한 사항 및 특정 기능 예외가 적용됩니다 .

공구 반경 보정 (TRC) 은 PTP 를 통해 활성화할 수 없습니다 .

PTPG0 에서 활성 공구 반경 보정 (TRC) 의 경우 CP 에 의해 이송됩니다 .

PTP 는 원활한 접근 및 후퇴 (SAR) 를 허용하지 않습니다 .

PTPG0, CP 이송이 원활한 접근 및 후퇴 (SAR) 에 사용됩니다 .

PTP 는 절삭 싸이클 (CONTPRON, CONTDCON) 을 허용하지 않습니다 .

PTPG0 에서 절삭 싸이클 (CONTPRON, CONTDCON) 은 CP 에 의해 이송됩니다 .

모따기 (CHF, CHR) 및 라운딩 (RND, RNDM) 은 무시됩니다 .

콤프레서는 PTP 와 호환되지 않으며 PTP 블록에서 자동으로 선택 취소됩니다 .

보간에서 중복된 축은 PTP 섹션 동안 변경할 수 없습니다 .

G643 을 지정할 경우 축 정확성을 통해 유연화한 후 자동으로 G642 로 전환됩니다 .

활성 PTP 의 경우 변환 축은 동시에 위치 지정 축이 될 수 없습니다 .

참고 문서 :

/FB2/ 기능 설명 확장 기능 ; 운동학적 변환 (M1),
" 직교 PTP 이송 " 절

TRACON 에 대한 PTP:

또한 PTP 는 첫 번째로 연결된 변환이 PTP 를 지원하는 경우 TRACON 을 통해 사용할 수 있습니다 .

TRANSMIT 의 STAT= 및 TU= 의미

로타리 축을 180 도 회전하거나 CP 형상이 극 좌표를 통과해야 하는 경우 기계 데이터 $MC_TRANSMIT_POLE_SIDE_FIX_1/2 [48] 에 종속된 로타리 축을 시계 방향 또는 시계 반대 방향으로 -/+ 180 도 회전하여 이송할 수 있습니다 . 또한 이송이 극 좌표를 통과하거나 극 좌표 주위 회전을 수행할지 여부를 설정할 수 있습니다 .

6.10 변환 선택 시 제한사항

작동

가공 프로그램 또는 MDA 를 통해 변환을 선택할 수 있습니다 . 주 :

- 중간 이동 블록이 삽입되지 않습니다 (모따기 / 반경).
- 스플라인 블록 순서는 제외되어야 합니다 . 그렇지 않을 경우 메시지가 표시됩니다 .
- 미세 공구 보정은 선택 취소해야 합니다 (FTOCOF). 그렇지 않을 경우 메시지가 표시됩니다 .
- 공구 반경 보정은 선택 취소해야 합니다 (G40). 그렇지 않을 경우 메시지가 표시됩니다 .
- 활성화된 공구 길이 옵셋은 제어 시스템에 의해 변환에 포함됩니다 .
- 제어 시스템은 변환하기 전에 현재 프레임 활성을 선택 취소합니다 .
- 제어 시스템은 변환에 의해 영향을 받은 활성 작동 영역 제한을 선택 취소합니다 (WALIMOF).
- 보호 영역 모니터링이 선택 취소됩니다 .
- 연속 경로 제어 및 라운딩이 중단됩니다 .
- 기계 데이터에서 지정된 모든 축이 블록을 기준으로 동기화되어야 합니다 .
- 교환된 축이 다시 교환됩니다 . 그렇지 않을 경우 메시지가 표시됩니다 .
- 메시지는 종속 축에 대해 출력됩니다 .

공구 변경

공구는 공구 반경 보정 기능을 선택 취소할 경우에만 교환할 수 있습니다 .

공구 길이 옵셋 및 공구 반경 보정 선택 / 선택 취소 변경은 동일한 블록에서 프로그래밍하지 않아야 합니다 .

프레임 변경

기본 좌표계를 전적으로 참조하는 모든 명령이 허용됩니다 (FRAME, 공구 반경 보정). 그러나 비활성 변환과 달리 , G91 을 통한 프레임 변경 (증분 치수) 은 개별적으로 처리되지 않습니다 . 이송할 증분은 이전 블록에 적용된 프레임과 관계 없이 새 프레임의 공작물 좌표계에서 평가됩니다 .

예외

변환이 적용된 축은 다음으로 사용할 수 없습니다 :

- 미리 설정된 축 (알람),
- 확인 점 복귀의 경우 (알람),
- 원점 복귀의 경우 (알람).

6.11 변환 선택 취소 (TRAFOOF)

기능

TRAFOOF 명령은 사용 중인 모든 변환 및 프레임을 비활성화합니다 .

주

이후 필요한 프레임은 갱신된 프로그램에 의해 활성화되어야 합니다 .

참고 :

변환 해제 시 변환 선택에 사용된 동일한 제한 사항이 적용됩니다 ("변환 선택 시 제한사항" 절 참조).

구문

TRAFOOF

의미

TRAFOOF	사용 중인 모든 변환 / 프레임을 비활성화하는 명령

6.12 연결된 변환 (TRACON, TRAFOOF)

기능

첫 번째 변환에서 축의 이동 구성 요소를 연결된 두 번째 변환의 입력 데이터로 사용하도록 **두 개**의 변환을 연결할 수 있습니다 . 두 번째 변환에서 이동 부분은 기계 축에서 동작합니다 .

연결에는 **두 개**의 변환이 포함됩니다 .

주

공구는 항상 연결부의 첫 번째 변환에 할당됩니다 . 그런 다음 활성 공구 길이가 0 인 경우 후속 변환이 수행됩니다 . 머신 데이터 (_BASE_TOOL_) 에서 설정된 베이직 공구 길이만 연결부의 첫 번째 변환에 대해 사용됩니다 .

장비 제조업체

머신 데이터에 의해 미리 정의된 변환에 대해서는 장비 제조업체에 의해 제공된 정보를 참조하십시오 .

변환 및 연결 변환은 옵션입니다 . 현재 카탈로그에서는 항상 특정 제어 시스템의 관련 변환에 대한 가용성 정보를 제공합니다 .

적용

- 예를 들어 공구 연삭 가공과 같이 경사진 연삭 휠을 사용하여 실린더의 측면 선으로 프로그래밍 (TRACYL) 된 형상 연삭
- 경사진 연삭 휠을 사용하여 TRANSMIT 를 통해 생성된 라운드되지 않은 형상의 정삭 .

구문

TRACON(trf,par)	이를 통해 연결 변환을 실행합니다 .
TRAFOOF	

의미

TRACON	이를 통해 연결 변환을 실행합니다 . 이전에 다른 변환을 실행한 경우 TRACON() 을 통해 암시적으로 비활성화됩니다 .
TRAFOOF	가장 최근에 활성화된 (연결된) 변환이 비활성화됩니다 .
trf	연결된 변환 값 : 첫 번째 / 단일 연결 변환에 대해 0 또는 1. 여기서 프로그래밍되지 않을 경우 0 또는 1 값을 지정하는 것과 동일한 의미를 가집니다 . 예를 들어 첫 번째 / 단일 변환이 실행됩니다 . 두 번째 연결 변환의 경우 2. (0-2 이외를 사용한 경우 알람이 발생합니다 .)
par	예를 들어 경사 축의 각도와 같이 파라미터가 필요한 연결부의 변환에 대해 하나 이상의 파라미터가 쉼표로 구분됩니다 . 파라미터를 설정하지 않을 경우 기본값 또는 마지막으로 사용한 파라미터가 적용됩니다 . 이전 파라미터에 대해 기본 설정을 적용해야 하는 경우 지정된 파라미터가 필요한 순서대로 평가되도록 하려면 쉼표를 사용해야 합니다 . 특히 , trf 를 지정할 필요가 없더라도 최소 하나의 파라미터 앞에 쉼표가 필요합니다 . 예를 들면 : TRACON(, 3.7).

요구 사항

두 번째 변환은 **" 경사 축 "**(TRAANG) 이어야 합니다 . 첫 번째 변환은 다음과 같습니다 .

- 범용 밀링 헤드를 포함한 방향 변환 (TRAORI)
- TRANSMIT
- TRACYL
- TRAANG

연결할 개별 변환 및 활성화할 연결된 변환이 기계 데이터에 의해 정의된 연결된 변환에 대해 활성화 명령을 사용하기 위한 조건입니다 .

개별 변환 설명에 표시된 보조 조건 및 특별한 경우도 연결된 변환에서 사용할 수 있습니다 .

변환 기계 데이터 구성에 대한 내용은 다음을 참조하십시오 :

/FB2/ 작동 설명서 확장된 기능 ; 운동학적 변환 (M1) 및

/FB3/ 작동 설명서 , 특별한 기능 ; 3- ~ 5- 축 변환 (F2).

공구 옵셋 7

7.1 옵셋 메모리

작동

옵셋 메모리 구조

모든 데이터 필드는 T 및 D 번호 (" 플랫 D No." 제외) 를 통해 호출할 수 있습니다 . 공구의 형상 데이터 이외에 공구 유형과 같은 기타 정보가 포함되어 있습니다 .

플랫 D 번호 구조

"플랫 D No. 구조"는 NCK 외부에서 공구 관리를 수행하는 경우 사용됩니다. 이 경우 D 번호는 공구에 대한 지정 없이 해당 공구 보정 블록을 통해 생성됩니다 .

T는 가공 프로그램에서 계속해서 프로그래밍할 수 있습니다. 그러나 이 T에는 프로그래밍된 D 번호에 대한 기준이 없습니다 .

사용자 절삭 에지 데이터

사용자 절삭 에지 데이터는 기계 데이터를 통해 구성할 수 있습니다 . 장비 제조업체의 설명서를 참조하십시오 .

공구 파라미터

주

옵셋 메모리의 개별 값

옵셋 메모리 P1 - P25 의 개별 값은 시스템 변수를 통해 프로그램에서 읽고 쓸 수 있습니다 . 다른 모든 파라미터는 보관됩니다 .

공구 파라미터 $TC_DP15 ~ $TC_DP17, $TC_DP19 및 $TC_DP20 뿐만 아니라 $TC_DP6 ~ $TC_DP8, $TC_DP10 및 $TC_DP11 에는 공구 유형에 따라 다른 의미가 있습니다 .

[1] 또한 3D 페이스 밀링의 경우 밀링 공구에 적용

[2] 슬롯 톱의 공구 유형

[3] 아직 사용 안됨 : SINUMERIK 840D 에서 사용되지 않음

공구 파라미터 번호 (DP)	시스템 변수 의미	추가 설명
$TC_DP1	공구 종류	개요에 대해서는 목록 참조
$TC_DP2	공구 끝 위치	선삭 공구에만 해당
형상	**길이 보정**	
$TC_DP3	길이 1	다음에 할당
$TC_DP4	길이 2	유형 및 레벨
$TC_DP5	길이 3	

공구 파라미터 번호 (DP)	시스템 변수 의미	추가 설명
형상	**반경**	
$TC_DP6[1] $TC_DP6[2]	반경 1/ 길이 1 직경 d	밀링 / 선삭 / 연삭 공구 슬롯 톱
$TC_DP7[1] $TC_DP7[2]	길이 2/ 모서리 반경 , 테이퍼 밀링 공구 슬롯 너비 b 모서리 반경	밀링 공구 슬롯 톱
$TC_DP8[1] $TC_DP8[2]	밀링 공구의 라운딩 반경 1 돌출 길이 k	밀링 공구 슬롯 톱
$TC_DP9[1.3]	라운딩 반경 2	아직 사용 안됨
$TC_DP10[1]	각도 1 공구의 페이스 끝	테이퍼 밀링 공구
$TC_DP11[1]	각도 2 공구 수직 방향 축	테이퍼 밀링 공구
마모량	**길이 및 반경 보정**	
$TC_DP12	길이 1	
$TC_DP13	길이 2	
$TC_DP14	길이 3	
$TC_DP15[1] $TC_DP15[2]	반경 1/ 길이 1 직경 d	밀링 / 선삭 / 연삭 공구 슬롯 톱
$TC_DP16[1] $TC_DP16[3]	길이 2/ 모서리 반경 , 테이퍼 밀링 공구 , 슬롯 너비 b 모서리 반경	밀링 공구 슬롯 톱
$TC_DP17[1] $TC_DP17[2]	밀링 공구의 라운딩 반경 1 돌출 길이 k	밀링 /3D 페이스 밀링 슬롯 톱
$TC_DP18[1.3]	라운딩 반경 2	아직 사용 안됨
$TC_DP19[1]	각도 1 공구의 페이스 끝	테이퍼 밀링 공구
$TC_DP20[1]	각도 2 공구 수직 방향 축	테이퍼 밀링 공구
공구 기준 치수 / 어댑터	**공구 길이 옵셋**	
$TC_DP21	길이 1	
$TC_DP22	길이 2	
$TC_DP23	길이 3	
공정		
$TC_DP24	간극 각도	선삭 공구에만 해당
$TC_DP25		아직 사용 안됨

비고

여러 입력 구성 요소는 형상 변수에 대해 사용 가능합니다(예를 들어 길이 1 또는 반경). 이러한 구성 요소는 값을 구하기 위해 함께 추가된 다음 (예를 들어 전체 길이 1, 전체 반경) 계산에 사용됩니다 .

필요 없는 옵셋 값은 0 으로 지정되어야 합니다 .

형상 공구의 공구 파라미터 $TC-DP1 ~ $TC-DP23

주

$TC_DP7 과 같이 테이블에 나열되지 않은 공구 파라미터는 평가되지 않습니다 . 예를 들어 내용이 의미가 없습니다 .

공구 파라미터 번호 (DP)	의미	절삭 Dn		추가 설명
$TC_DP1	공구 종류			400 ... 599
$TC_DP2	공구 끝 위치			
형상	**길이 보정**			
$TC_DP3	길이 1			
$TC_DP4	길이 2			
$TC_DP5	길이 3			
형상	**반경**			
$TC_DP6	반경			
형상	**제한 각도**			
$TC_DP10	최소 제한 각도			
$TC_DP11	최대 제한 각도			
마모량	**길이 및 반경 보정**			
$TC_DP12	마모 길이 1			
$TC_DP13	마모 길이 2			
$TC_DP14	마모 길이 3			
$TC_DP15	마모 반경			
마모량	**제한 각도**			
$TC_DP19	마모 최소 제한 각도			
$TC_DP20	마모 최대 제한 각도			
공구 기준 치수 / 어댑터	**길이 보정**			
$TC_DP21	길이 1			
$TC_DP22	길이 2			
$TC_DP23	길이 3			

기본값 및 마모 값

결과 값은 기본값 및 마모 값의 합계입니다 (예를 들어 반경의 경우 $TC_DP6 + $TC_DP15). 기본 측정값 ($TC_DP21 – $TC_DP23) 도 첫 번째 절삭 에지의 공구 길이에 추가됩니다 . 표준 공구의 유효 공구 길이에 영향을 줄 수 있는 다른 모든 파라미터도 이 공구 길이에 영향을 줍니다 (어댑터 , 방향 공구 홀더 , 셋팅 데이터).

제한 각도 1 및 2

제한 각도 1 및 2 각각은 절삭 에지 원점에 대한 절삭 에지 중심점의 벡터와 관련되며 시계 방향으로 카운트됩니다 .

7.2 추가 옵셋

7.2.1 추가 옵셋 선택 (DL)

작동

추가 옵셋은 가공 과정에서 프로그램될 수 있는 공정 옵셋으로 볼 수도 있습니다 . 이 옵셋은 절삭날의 기하학적 데이터를 참조하기 때문에 공구 절삭 데이터의 요소라고 할 수 있습니다 .

추가 옵셋의 데이터는 DL 번호 (DL: 위치 종속 ; 사용 위치에 관한 옵셋) 를 사용하여 처리되며 조작자 인터페이스를 통해 입력됩니다 .

용도

치수 오류가 발생하면 추가 옵셋을 사용하여 사용 위치를 보정해야 하는 경우가 발생합니다 .

구문

```
DL=<number>
```

중요도

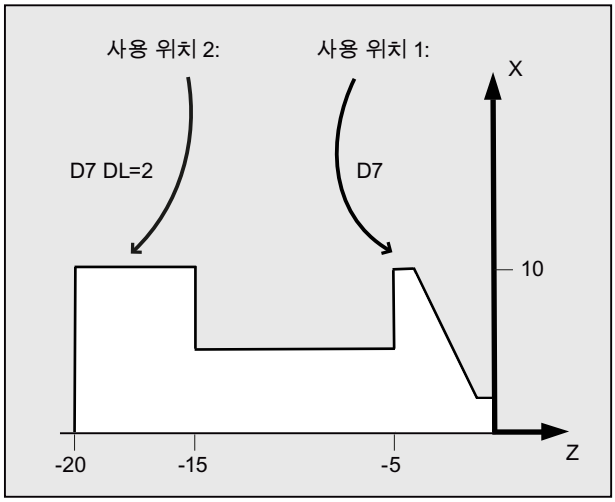
DL	추가 옵셋 활성화 명령
<number>	활성화시킬 추가 공구 옵셋 데이터는 <number> 파라미터를 사용하도록 지정되어 있습니다 .

주

머신 데이터는 추가 옵셋의 번호를 정의하고 이를 활성화하는 데 이용됩니다 (→ 기계 OEM 의 데이터를 주의 깊게 관찰하십시오 !).

예제

2 개의 베어링 시트에 동일한 절삭날 사용 :



프로그램 코드	비고
N110 T7 D7	; 회전 장치는 위치 7에 있습니다. D7 및 DL=1은 다음 블록에서 활성화되어 그 안을 관통합니다 .
N120 G0 X10 Z1	
N130 G1 Z-6	
N140 G0 DL=2 Z-14	; DL=2 는 D7 에 추가되어 활성화되며 다음 블록을 관통합니다 .
N150 G1 Z-21	
N160 G0 X200 Z200	; 공구 교환 위치로 이동 .
...	

7.2.2 마모값 및 설정값을 지정 ($TC_SCPxy[t,d], $TC_ECPxy[t,d])

작동

마모값 및 설정값은 시스템 변수를 사용하여 읽기 및 쓰기를 할 수 있습니다. 이 논리는 공구 및 공구 끝의 시스템 변수와 상응하는 논리에 기반을 두고 있습니다 .

시스템 변수

시스템 변수	중요도
$TC_SCPxy[<t>,<d>]	xy 를 통해 특정 형상 파라미터에 마모값이 할당됩니다 . 이 때 , X 는 마모값의 숫자에 대응되고 , Y 는 형상 파라미터에 대한 참조를 구축합니다 .
$TC_ECPxy[<t>,<d>]	xy 를 통해 특정 형상 파라미터에 설정값에 할당됩니다 . 이 때 , X 는 설정값의 숫자에 대응되고 , Y 는 형상 파라미터에 대한 참조를 구축합니다 .
<t>: 공구의 T 번호 <d>: 공구 절삭날의 D 번호	

주

정의된 마모값 및 설정값은 형상 파라미터 및 기타 옵셋 파라미터에 추가됩니다 (D 번호).

예제

길이 1 의 마모값은 공구 <t> 의 절삭날 <d> 에 대한 1.0 의 값으로 설정됩니다 .

파라미터 : $TC_DP3 (길이 1, 선삭 공구 사용)

마모값 : $TC_SCP13 to $TC_SCP63

설정값 : $TC_ECP13 to $TC_ECP63

$TC_SCP43 [<t>,<d>] = 1.0

7.2.3 추가 옵셋 삭제 (DELDL)

작동

DELDL 명령은 공구의 절삭날에 대한 추가 옵셋을 삭제합니다 (메모리 공간 확보용). 정의된 마모값 및 설정값 모두가 삭제됩니다 .

구문

```
DELDL[<t>,<d>]
DELDL[<t>]
DELDL
<Status>=DELDL[<t>,<d>]
```

중요도

DELDL	추가 옵셋 삭제 명령
<t>	공구의 T 번호
<d>	공구 절삭날의 D 번호
DELDL[<t>,<d>]	공구 <t> 절삭날 <d> 의 추가 옵셋이 모두 삭제됩니다 .
DELDL[<t>]	공구 <t> 절삭날의 추가 옵셋이 모두 삭제됩니다 .
DELDL	TO 장치의 모든 공구 절삭날의 옵셋이 모두 삭제됩니다 (명령이 프로그램되는 채널).
<status>	삭제 상태
	값 : 의미 :
	0 삭제가 성공적으로 완료되었습니다 .
	- 옵셋이 아직 삭제되지 않았거나 (파라미터 설정이 정확하게 단 하나의 공구 날을 지정하고 있는 경우) 또는 완전히 삭제되지 않았습니다 (매개 변수 설정이 몇 개의 절삭날을 지정하고 있는 경우).

주

활성화된 공구의 마모값 및 설정값은 삭제할 수 없습니다 (D 또는 공구 데이터의 삭제 특성과 기본적으로 동일함).

7.3 공구 옵셋 특수 처리

작동

공구 길이 및 마모에 대한 기호의 평가는 설정 데이터 SD42900 에서 SD42960 를 사용하여 제어할 수 있습니다 .

이 사항은 기하축을 미러링하거나 가공 평면을 변경할 때 마모 요소의 특성 및 공구 방향으로 온도 보정을 할 때도 동일하게 적용됩니다 .

마모값 :

다음에서 마모값에 참조가 만들어진 경우 , 이 참조는 실제 마모값의 합계 ($TC_DP12 to $TC_DP20) 및 마모값($SCPX3 to $SCPX11)과 설정값($ECPX3 to $ECPX11)을 사용한 합계 옵셋으로 받아들여야 합니다 .

총합된 옵셋에 대한 자세한 정보를 보시려면
참고문서 :
기능 메뉴얼 , 공구 관리를 참조하십시오 .

셋팅 데이터

셋팅 데이터	중요도
SD42900 $SC_MIRROR_TOOL_LENGTH	공구 길이 요소의 미러링 및 공구 베이스 치수의 요소
SD42910 $SC_MIRROR_TOOL_WEAR	공구 길이 요소의 마모값 미러링
SD42920 $SC_WEAR_SIGN_CUTPOS	마모 요소의 기호를 공구 끝 위치의 기능으로 평가
SD42930 $SC_WEAR_SIGN	마모 치수의 기호를 전환하십시오 .
SD42935 $SC_WEAR_TRANSFORM	마모값의 변환
SD42940 $SC_TOOL_LENGTH_CONST	공구 길이 요소의 기하축 할당
SD42950 $SC_TOOL_LENGTH_TYPE	종류와 상관없이 공구 길이 요소 할당
SD42960 $SC_TOOL_TEMP_COMP	공구 방향으로 온도 보정값 . 또한 공구 방향이 프로그램된 경우에 효과적입니다 .

참조 문서

기능 메뉴얼 기본 기능 ; 공구 옵셋 (W1)

그 밖의 정보

수정된 셋팅 데이터의 활성화

위에서 설명한 세팅 데이터가 수정된 경우 , 다음에 공구날이 선택될 때까지 공구 요소는 재평가되지 않습니다. 이미 공구가 활성화되어있고, 이 공구의 데이터가 재평가될 예정이라면, 공구를 다시 선택해야만 합니다 .

결과 공구 길이가 축의 미러링 상태 변경으로 인해 수정된 경우에도 동일한 사항이 적용됩니다 . 미러링 명령을 사용한 이후에는 수정된 공구 길이 요소를 활성화시키기 위해 공구를 다시 선택해야만 합니다 .

전향적 공구 홀더 및 새로운 세팅 데이터

셋팅 데이터 SD42900 부터 SD42940 은 방향 능력이 있는 활성화 공구 홀더의 요소에 대해 어떤 영향도 없습니다. 하지만 전향적 공구 홀더를 계산할 때는 항상 총 결과 길이(공구 길이 + 마모 + 공구 베이스 수치)를 고려합니다. 셋팅 데이터에 의해 시작된 수정 사항은 모두 결과 총 길이의 계산에 포함되어 있습니다 . 예를 들어 , 전향적 공구 홀더의 벡터는 가공 평면과 상관 없습니다 .

주

전향적 공구 홀더가 사용되었을 때 , 비 미러 기본 시스템에 사용하는 모든 공구를 정의하는 것이 실용적일 때가 많습니다 . 미러링 된 가공에만 사용될 때도 마찬가지입니다 . 미러링된 축을 사용하여 가공할 때 , 공구의 실제 위치가 정확하게 설명되어 공구 홀더가 그만큼 회전합니다. 그러면 모든 공구 길이 요소가 정확한 방향으로 자동으로 작동합니다. 이 때 개별 축의 미러링 상태에 따른 셋팅 데이터를 통해 개별 요소 평가를 제어하지 않아도 됩니다 .

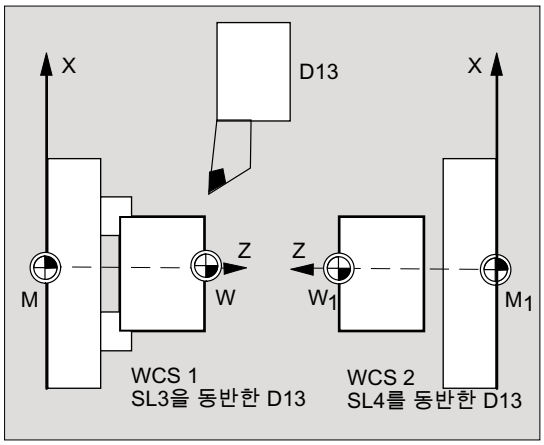
그 밖의 이용 옵션

전향적 공구 홀더 기능을 활용하면 기계의 선삭 공구의 물리적 옵션이 없는 경우에도 유용합니다 . 아무리 다른 방향의 공구가 영구적으로 설치되어 있는 경우라 하더라도 마찬가지입니다 . 공구 치수는 기본 오리엔테이션에서 획일적으로 수행될 수 있습니다 . 이 때 가공에 대한 치수 관련성은 실제 공구 홀더의 회전에 따라 계산됩니다 .

7.3.1 공구 길이 미러링

작동

셋팅 데이터인 SD42900 $SC_MIRROR_TOOL_LENGTH 및 SD42910 $SC_MIRROR_TOOL_WEAR가 0으로 설정되어 있지 않은 경우, 공구 길이 요소 및 마모값 및 해당 축을 사용한 기본 치수의 요소를 미러링할 수 있습니다 .



SD42900 $SC_MIRROR_TOOL_LENGTH

셋팅 데이터가 0 **이 아님** :

공구 길이 요소 ($TC_DP3, $TC_DP4 and $TC_DP5) 및 기본 치수 요소 ($TC_DP21, $TC_DP22 and $TC_DP23) 는 해당 축에 대해 미러링되지 않고 , 기호 전환에 의해 미러링됩니다 .

마모값이 미러링되지 **않습니다** . 마모값도 미러링 되지 않은 경우 , 셋팅 데이터 SD42910 $SC_MIRROR_TOOL_WEAR 를 반드시 설정해야 합니다 .

SD42910 $SC_MIRROR_TOOL_WEAR

셋팅 데이터가 0 **이 아님** :

공구 길이 요소의 마모값 - 해당 축은 미러링 됨 - 역시 기호 전환에 의해 미러링됩니다 .

7.3.2 마모 기호 평가

작동

셋팅 데이터 SD42920 $SC_WEAR_SIGN_CUTPOS and SD42930 $SC_WEAR_SIGN 이 0 으로 설정되지 않는 경우 , 마모 요소의 기호 평가를 전환할 수 있습니다 .

SD42920 $SC_WEAR_SIGN_CUTPOS

셋팅 데이터가 0 **이 아님** :

관련 공구 끝 위치를 사용하는 공구의 경우 (선삭 및 연삭 공구 , 공구 종류 400), 가공 평면에서 마모 요소의 기호 평가는 공구 끝 위치에 달려 있습니다 . 이 셋팅 데이터는 공구 끝 위치와 관련이 없는 공구 종류에 중요하지 않습니다 .

다음 표에서 , SD42920(0 이 아님) 을 사용하여 전환된 기호를 사용하는 치수는 X 를 사용하여 고안되었습니다 .

공구 끝 위치	길이 1	길이 2
1		
2		X
3	X	X
4	X	
5		
6		
7		X
8	X	
9		

주

SD42920 및 SD42910 을 사용한 기호 평가는 서로 연관성이 없습니다 . 만일 , 예를 들어 치수의 기호가 셋팅 데이터를 사용하여 변경되었다면 , 결과 기호는 그대로 남습니다 .

SD42930 $SC_WEAR_SIGN

셋팅 데이터가 0 **이 아님** :

마모 치수의 기호를 모두 전환하십시오 . 이는 공구 길이 및 공구 반경 , 라운드 반경 등과 같은 기타 변수 모두에 영향을 줍니다 .

공구 치수를 양수로 입력한 경우 , 공구는 " 짧고 " " 가늘어 " 집니다 . " 공구 옵셋 , 특정 처리 ", " 변경된 셋팅 데이터 활성화 " 장을 참조하십시오 .

7.3.3 활성화된 가공 작업의 좌표계 (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS)

작동

기계의 구조 또는 전향적 공구 홀더의 가용성에 따라 , 좌표계 중 하나에서 측정된 마모값이 적절한 좌표계로 변환 또는 전환됩니다 .

활성화된 가공 작업의 좌표계

다음 좌표계는 공구 길이 옵셋을 생산할 수 있습니다. 이 옵셋은 공구 길이 요소 "마모"를 그룹 56 의 G 코드를 통해 활성화된 공구에 통합시킬 때 사용될 수 있습니다 .

- 기계 좌표계 (MCS)
- 기본 좌표계 (BCS)
- 공작물 좌표계 (WCS)
- 공구 좌표계 (Tool coordinate system: TCS)
- 구조 전환의 공구 좌표계 (Tool coordinate system of kinematic transformation: KCS)

구문

```
TOWSTD
TOWMCS
TOWWCS
TOWBCS
TOWTCS
TOWKCS
```

중요도

TOWSTD	공구 길이 마모값에서 옵셋에 대한 설정값 시작하기
TOWMCS	MCS 에서 공구 길이의 옵셋
TOWWCS	WCS 에서 공구 길이의 옵셋
TOWBCS	BCS 에서 공구 길이의 옵셋
TOWTCS	공구 홀더 원점에서 공구 길이의 옵셋 (전향적 공구 홀더)
TOWKCS	공구 헤드에서 공구 길이의 옵셋 (구조 전환)

그 밖의 정보

특징적 기능들

가장 중요한 특징적 기능들은 다음 표에 나와 있습니다 .

G 코드	마모값	활성화된 전향적 공구 홀더
TOWSTD	초기값 , 공구 길이	마모값은 회전 대상입니다 .
TOWMCS	MCS 에서 마모값 . 방향을 잡을 수 있는 공구 홀더가 활성화되지 않은 경우 , TOWMCS 는 TOWSTD 와 동일합니다 .	마모는 상관 없이 결과 공구 길이의 벡터만을 회전시킵니다 .
TOWWCS	마모값이 WCS 에서 MCS 로 변환되었습니다 .	공구 벡터는 마모는 고려하지 않고 TOWMCS 에 대해서만 계산되었습니다 .
TOWBCS	마모값이 BCS 에서 MCS 로 변환되었습니다 .	공구 벡터는 마모는 고려하지 않고 TOWMCS 에 대해서만 계산되었습니다 .
TOWTCS	마모값은 공작물 좌표계에서 MCS 로 변환되었습니다 .	공구 벡터는 마모는 고려하지 않고 TOWMCS 에 대해서만 계산되었습니다 .

TOWWCS , TOWBCS, TOWTCS: 공구 벡터에 마모 벡터가 추가되었습니다 .

선형 변환

MCS 가 BSC 에서 선형 변환에 의해 생성된 경우에만 MCS 에서 공구 길이를 의미 있게 정의할 수 있습니다 .

비선형 변환

예를 들어, TRANSMIT 을 사용하는 비선형 변환이 활성화된 경우 , 좌표계의 요청대로 MCS 가 지정되었다면 , BCS 는 자동적으로 사용됩니다 .

구조 전환 및 전향적 공구 홀더 없음

구조 전환 및 전향적 공구 홀더 모두 활성화되어있지 않은 경우 , 그 외 4 개의 좌표계가 모두(WCS에 대한 좌표계 제외) 결합됩니다. 그렇게 되면 WCS만 다른 좌표계와 달라집니다. 공구 길이만 평가되면 되기 때문에 좌표계간의 해석은 무의미합니다 .

참조 문서 :
공구 보정에 관해 자세한 사항을 보시려면 다음을 참조하십시오 .
기능 메뉴얼 기본 기능 ; 공구 옵셋 (WI)

계산에 마모값 포함

셋팅 데이터 **SD42935 $SC_WEAR_TRANSFORM** 는 3 개의 마모 요소를 정의합니다 .

- 마모량
- 미세조정 옵셋 합계
- 일반 옵셋 합계

위 항목들은 다음의 G 코드 중 하나가 활성화되어 있는 경우 , 전향적이 될 수 있는 공구 홀더 또는 어댑터 변환을 사용하여 회전됩니다 .

- TOWSTD 기본 설정

  공구 길이의 옵셋 설정
- TOWMCS 마모값

  기계 좌표계 (MCS)
- TOWWCS 마모값

  공작물 좌표계 (WCS
- TOWBCS 마모값 (BCS)

  기본 좌표계
- TOWTCS 공구 홀더 참조 (T 공구 홀더 참조) 의 공구 좌표계에서 마모값
- TOWKCS 구조 전환에 대한 공구 헤드의 좌표계에서 마모값

주

개별 마모 요소의 평가 (기하축 , 기호 평가에 할당) 는 다음의 영향을 받습니다 .

- 활성 평면
- 어댑터 변환
- 다음의 셋팅 데이터 :
  - 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 공구 길이 및 평면 변경

작동

셋팅 데이터 SD42940 $SC_TOOL_LENGTH_CONST 가 0 으로 설정되지 않은 경우 , 평면을 변경할 때 연삭 및 선삭 공구에 대해 공구 길이 요소 - 길이 , 마모 및 기본 치수 등 - 를 기하축에 할당할 수 있습니다 .

SD42940 $SC_TOOL_LENGTH_CONST

셋팅 데이터가 0 **이 아님** :

공구 길이 요소 (길이 , 마모 및 공구 기본 치수) 에 기하축을 할당하면 가공 평면이 변경되어도 할당이 변경되지 않습니다 (G17 - G19).

다음 표는 연삭 및 선삭 공구(공구 종류 400 - 599)에 대한 기하축에 공구 길이 요소를 할당하는 것을 보여줍니다 .

내용	길이 1	길이 2	길이 3
17	Y	X	Z
*)	X	Z	Y
19	Z	Y	X
-17	X	Y	Z
-18	Z	X	Y
-19	Y	Z	X

*) 나열된 6 개 값 중 하나와 동일하지 않은 0 이 아닌 각 값은 18 의 값으로 평가됩니다 .

다음 표는 다른 모든 공구 (공구 종류 400 초과 또는 599 미만) 에 대한 기하축에 공구 길이 요소를 할당하는 것을 보여줍니다 .

작업 평면	길이 1	길이 2	길이 3
*)	Z	Y	X
18	Y	X	Z
19	X	Z	Y
-17	Z	X	Y
-18	Y	Z	X
-19	X	Y	Z

*) 나열된 6 개 값 중 하나와 동일하지 않은 0 이 아닌 각 값은 17 의 값으로 평가됩니다 .

주

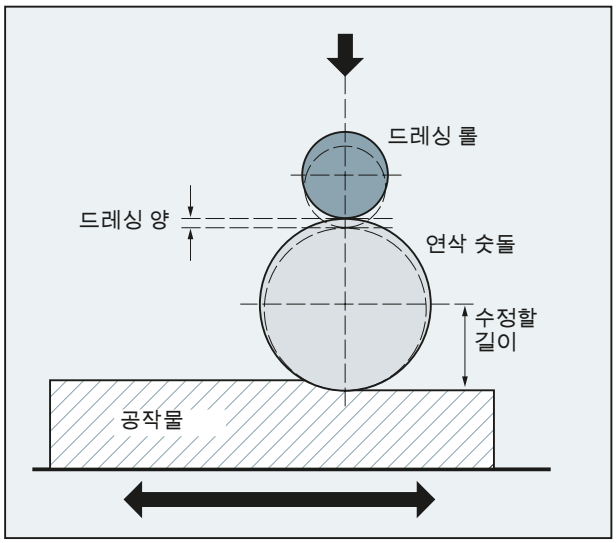
표에 나온 설명에서는 기하 축 1 ~ 3의 이름을 X, Y, Z 축으로 간주합니다. 하지만 축 식별자가 아닌 축 순서가 축에 보정을 할당하는 데 있어 결정적인 요소입니다 .

7.4 온라인 공구 옵셋 (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF)

기능

' 온라인 공구 옵셋 ' 기능이 활성 상태이면 가공을 통해 생성된 공구 길이 옵셋이 즉시 연삭 공구에 적용됩니다 .

연삭 휠이 가공과 평행하게 드레싱되는 CD 드레싱이 그 예입니다 .



공구 길이 옵셋은 가공 채널 또는 병렬 채널 (드레서 채널) 에 의해서 변경될 수 있습니다 .

온라인 공구 옵셋은 드레싱 처리 시간에 따라 다양한 기능을 사용하여 작성됩니다 .

- 연속 작성 , 넌모달 (PUTFTOCF)

  PUTFTOCF 를 사용하면 가공과 동시에 드레싱이 이루어집니다 .

  공구 옵셋은 가공 채널에서 폴리노미널 기능의 1차, 2차, 또는 3차 순서에 따라 지속적으로 변화합니다 . 이 순서는 FCTDEF 를 사용하여 사전에 정의해 두어야 합니다 .

  PUTFTOCF 는 항상 넌모달이므로 후속 이송 블록에 적용됩니다 .
- 연속 작성 , 모달 : ID=1 DO FTOC(' 온라인 공구 옵셋 (FTOC) [쪽 591]' 참조)
- 불연속 작성 (PUTFTOC)

  PUTFTOC 를 사용하면 병렬 채널에서의 가공과 드레싱이 동시에 이루어지지 않습니다 . PUTFTOC 로 지정된 옵셋 값은 대상 채널에 즉시 적용됩니다 .

주

온라인 공구 옵셋은 연삭 공구에만 적용될 수 있습니다 .

구문

대상 채널에서 온라인 공구 옵셋 활성화 / 비활성화 :

```
FTOCON
...
FTOCOF
```

온라인 공구 옵셋 작성 :

- 연속 , 넌모달 :

```
FCTDEF(< 기능 >,< 하한 >,< 상한 >,<a0>,<a1>,<a2>,<a3>)
PUTFTOCF(< 기능 >,< 기준 값 >,< 공구 파라미터 >,< 채널 >,< 스핀들 >)
...
```

- 불연속 :

```
PUTFTOC(< 옵셋 값 >,< 공구 파라미터 >,< 채널 >,< 스핀들 >)
...
```

의미

FTOCON: 온라인 공구 옵셋 활성화
FTOCON 은 온라인 공구 옵셋이 적용될 채널에서 작성해야 합니다 .

FTOCOF: 온라인 공구 옵셋 취소
FTOCOF 를 사용하면 옵셋은 더 이상 적용되지 않지만 , PUTFTOC 를 사용하여 작성 완료한 값은 절삭날 관련 옵셋 데이터에서 수정됩니다 .

참고 :
온라인 공구 옵셋을 확실하게 비활성화하려면 FTOCOF 이후 공구 (T...) 를 선택 / 선택 취소해야 합니다 .

FCTDEF: FCTDEF 는 PUTFTOCF 를 위한 폴리노미널 기능을 정의하는 데 사용됩니다 .

파라미터 :

< 기능 >:	폴리노미널 기능의 번호 유형 : INT
< 하한 >:	하한값 유형 : REAL
< 상한 >:	상한값 유형 : REAL
<a0> ... <a3>:	폴리노미널 기능 계수 유형 : REAL

PUTFTOCF: ' 온라인 공구 옵셋의 연속 넌모달 작성 ' 기능 호출

파라미터 :

< 기능 >: 폴리노미널 기능의 번호

유형 : INT

참고 :
FCTDEF 에 대한 설정과 일치해야 합니다 .

< 기준 값 >: 옵셋이 도출될 변수 기준 값 (예 : 실제 값 변경)

유형 : VAR REAL

< 공구 파라미터 : 옵셋 값이 추가될 마모 파라미터 (길이 1, 2 또는 3) 의 번호

유형 : INT

< 채널 >: 온라인 공구 옵셋이 적용될 채널 번호

유형 : INT

참고 :
사용 중인 채널에 옵셋을 적용시키지 않는 경우에만 채널 번호를 지정해야 합니다 .

< 스핀들 >: 온라인 공구 옵셋이 적용될 스핀들 번호

유형 : INT

참고 :
현재 사용중인 활성 공구가 아닌 비활성 연삭 휠에 옵셋이 적용될 경우에만 스핀들 번호를 지정해야 합니다 .

PUTFTOC: ' 온라인 공구 옵셋의 불연속 작성 ' 기능 호출

파라미터 :

< 옵셋 값 >: 마모 파라미터에 추가할 옵셋 값

유형 : REAL

< 공구 파라미터 >: PUTFTOCF 참조

< 채널 >: 온라인 공구 옵셋이 적용될 채널 번호

유형 : INT

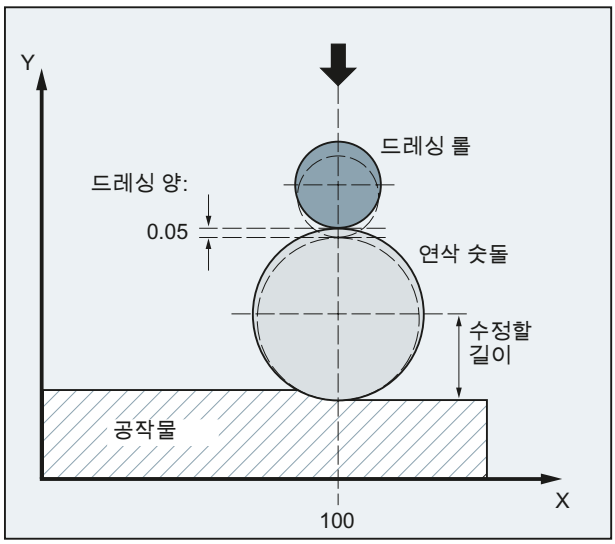
< 스핀들 >: PUTFTOCF 참조

예제

다음 사항을 포함한 표면 연삭 기계 :

- Y: 연삭 휠에 대한 절입 축
- V: 드레싱 롤러에 대한 절입 축
- 가공 채널 : X, Z, Y 축을 사용하는 채널 1
- 드레싱 채널 : V 축을 사용하는 채널 2

연삭 모션이 시작된 후 연삭 휠은 X100에서 0.05만큼 드레싱됩니다. 드레싱 양은 '온라인 옵셋 연속 작성 ' 이 설정된 연삭 공구에 적용됩니다 .



채널 1 에서 가공 프로그램 :

프로그램 코드	설명
…	
N110 G1 G18 F10 G90	; 기본 위치 .
N120 T1 D1	; 현재 공구 선택
N130 S100 M3 X100	; 스핀들 ON, 시작 위치로 이송 .
N140 INIT(2,"DRESS","S")	; 채널 2 에서 드레싱 프로그램 선택
N150 START(2)	; 채널 2 에서 드레싱 프로그램 시작
N160 X200	; 대상 지점으로 이송
N170 FTOCON	; 온라인 옵셋 활성화
N… G1 X100	; 추가 가공 .
N… M30	

채널 2 에서 드레싱 프로그램 :

프로그램 코드	설명
…	
N40 FCTDEF(1,–1000,1000,–$AA_IW[V],1)	; 정의 기능 : 경사도 1 의 직선
N50 PUTFTOCF(1,$AA_IW[V],3,1)	; 온라인 공구 옵셋 연속 작성 : V 축 모션에서 도출되는 활성 연삭 휠의 길이 3 은 채널 1 에서 보정됩니다 .
N60 V–0.05 G1 F0.01 G91	; 드레싱에 대한 전진 이동 , PUTFTOCF 는 이 블록에서만 적용됩니다 .
…	
N… M30	

추가 정보

온라인 TO 에 관한 일반 정보

각 보간 사이클에 대한 연속 작성의 경우 , 평가 기능 활성화에 이어 지령치 점프를 방지하기 위해 마모 메모리에서 각 변경 사항이 추가로 계산됩니다 .

양쪽의 경우 : 온라인 공구 옵셋은 각각의 스핀들 및 마모 파라미터의 길이 1, 2 **또는** 3 에서 동작합니다 .

길이는 현재 작업 평면을 기준한 기하 축에 지정됩니다 .

사용 중인 연삭 휠이 아닌 경우에는 GWPSON 및 TMON 을 포함한 공구 데이터를 사용하는 공구에 스핀들이 지정됩니다 .

옵셋은 항상 마모 파라미터 , 현재 공구 사이드 또는 비활성화된 공구 중 왼손 공구 사이드에 적용됩니다 .

주

여러 공구 사이드에 대한 옵셋이 동일한 경우 그 값은 연결 법칙에 따라 보조 공구 사이드로 자동 전송되어야 합니다 .

가공 채널에 대해 온라인 옵셋이 정의되어 있는 경우에는 이 채널의 해당 공구에 대한 마모 값을 가공 프로그램에서나 조작자 동작을 통해서는 변경할 수 없습니다 .

온라인 공구 옵셋은 공구 모니터링 (TMON) 에 추가하여 연속 연삭 휠 원주 속도 일정 제어 (GWPS) 에도 적용됩니다 .

7.5 3D 공구 옵셋 활성 (CUT3DC..., CUT3DF...)

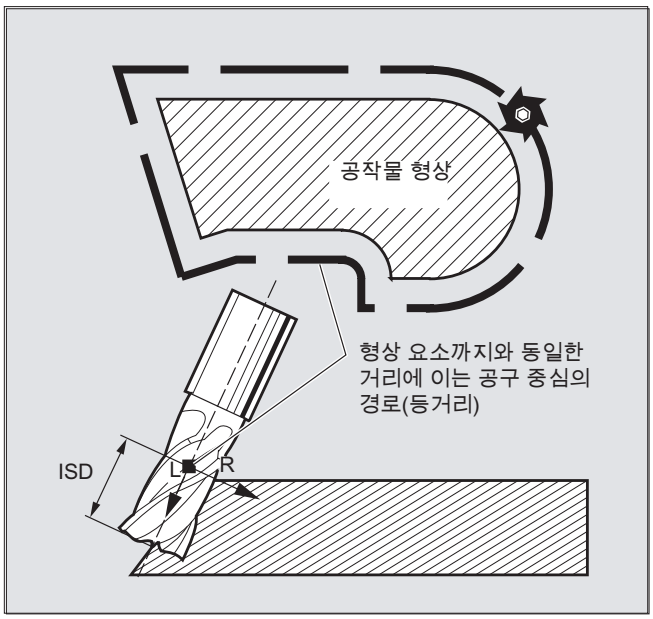
7.5.1 3D 공구 옵셋 활성화 (CUT3DC, CUT3DF, CUT3DFS, CUT3DFF, ISD)

작동

공구 방향 변경은 원통형 공구에 대한 공구 반경 보정을 고려합니다 .

2D 공구 반경 보정과 동일한 프로그래밍 명령이 3D 공구 반경 보정에 적용됩니다 . 좌측 / 우측 옵셋은 G41/G42 를 사용한 이동 방향으로 지정되어 있습니다 . 접근 응답은 항상 NORM 으로 제어됩니다 . 3D 반경 보정은 5 축 동시보간을 선택했을 때만 적용됩니다 .

3D 공구 반경 보정 역시 5D 공구 반경 보정을 호출합니다 . 이 경우에서는 공간에서 공구의 방향에 대해 5 자유도를 사용할 수 있기 때문입니다 .



2 1/2 D 및 3D 공구 반경 보정의 차이점

3D 공구 반경 보정 공구 방향은 변경될 수 있습니다. 2 1/2D 공구 반경 보정을 사용한다는 것은 연속 방향의 공구만을 사용한다는 것으로 간주됩니다 .

구문

```
CUT3DC
CUT3DFS
CUT3DFF
CUT3DF
ISD=<value>
```

중요도

CUT3DC	원주 밀링에 대한 3D 반경 옵셋 활성화
CUT3DFS	연속 방향을 사용하는 페이스 밀링에 대한 3D 공구 옵셋 공구 방향은 G17 - G19에 의해 결정되며, 프레임의 영향은 받지 않습니다 .
CUT3DFF	연속 방향을 사용하는 페이스 밀링에 대한 3D 공구 옵셋 공구 방향은 G17 - G19 에 의해 결정되며 , 프레임에 의해 회전되는 경우도 간혹 있습니다 .
CUT3DF	방향 변경을 사용하는 페이스 밀링에 대한 3D 공구 옵셋 (5 축 변환만 사용).
G40 X... Y... Z...	비활성화 방법 : 기하축을 사용하는 선형 블록 G0/G1
ISD	삽입 깊이

주

이 명령들은 모달식으로 영향을 주며 CUT2D 및 CUT2DF 와 같이 동일한 그룹에 작성됩니다 . 이 명령은 현재 평면에서 다음 이동을 실행할 때까지 선택 취소를 할 수 없습니다 . 이 명령은 항상 G40 을 지원하며 CUT 명령으로부터 독립적입니다 .

중간 블록은 3D 공구 반경 보정을 사용하도록 허용되었습니다. 2 1/2D 공구 반경 보정에 대한 정의를 적용합니다 .

보충 조건

- **G450/G451 및 DISC**

  원형 블록은 항상 외부 코너에 삽입됩니다 . G450/G451 에는 의미가 없습니다 .
  DISC 명령은 평가되지 않습니다 .

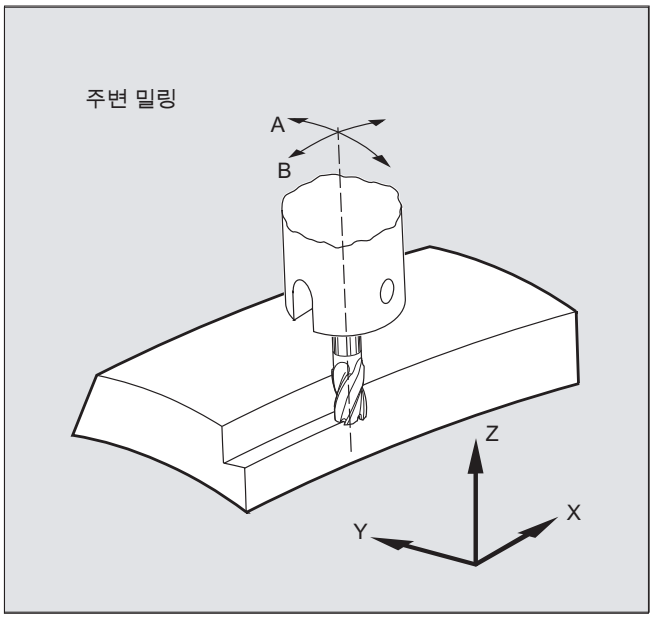
예제

프로그램 코드	비고
N10 A0 B0 X0 Y0 Z0 F5000	
N20 T1 D1	; 공구 호출 , 공구 옵셋값을 호출합니다 .
N30 TRAORI(1)	; 변환 선택
N40 CUT3DC	; 3D 공구 반경 보정 선택
N50 G42 X10 Y10	; 공구 반경 보정 선택
N60 X60	
N70 …	

7.5.2 3D 공구 옵셋 주변 밀링 , 페이스 밀링

원주 밀링

여기에서 사용하는 밀링의 유형은 경로를 정의 (가이드 라인) 하고 방향을 수정하여 구현됩니다 . 이런 유형의 가공에서 , 경로에 있는 공구의 형태는 관련이 없습니다 . 특히 , 공구 개입점의 반경이 중요합니다 .

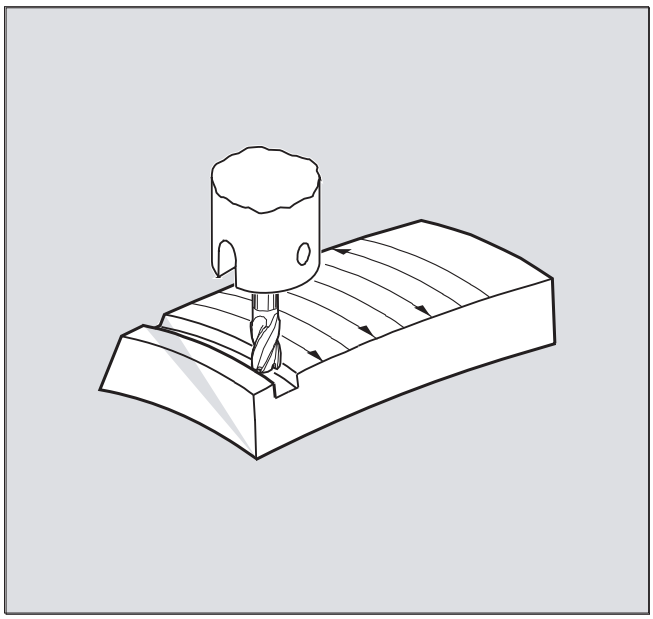


주

기능 3D TRC 는 원통형 공구로 제한됩니다 .

페이스 밀링

이런 유형의 3D 밀링에는 공작물 표면에 있는 3D 경로의 줄별 정의가 필요하게 됩니다 . 공구 형태 및 치수는 보통 CAM 에서 실행되는 계산에서 고려됩니다 . 포스트 프로세서는 가공 프로그램 (NC 블록에 추가) 을 3D 공구 옵셋에 필요한 방향 (활성화된 5 축 변환) 및 G 코드에 작성합니다 . 즉 , 기계 조작자가 NC 경로를 계산할 때 사용했던 공구 대신 약간 작은 공구를 사용할 수도 있다는 의미입니다 .



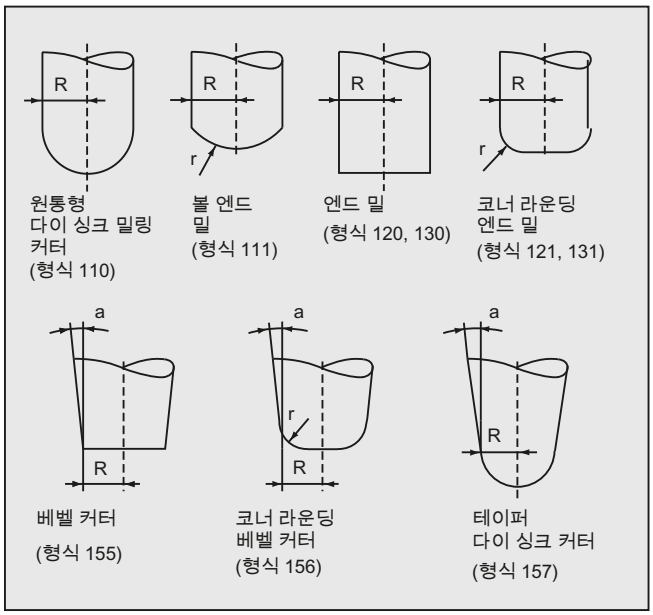
예 :

NC 블록은 10 mm 의 밀링 공구를 사용하여 계산됩니다 . 이 경우 , 직경 9.9 mm 의 밀링 공구는 가공에 사용될 수 있으며 , 그에 따라 수정된 거친 특성이 확장될 수도 있습니다 .

7.5.3 3D 공구 옵셋 페이스 밀링을 위한 공구 형태 및 공구 데이터

밀링 형태 , 공구 데이터

페이스 밀링 작업 및 공구 데이터 제한값에 사용될 수 있는 공구 형태 개요는 다음에 나열되어 있습니다 . 공구 샤프트의 형태는 고려되지 않습니다 . 공구 종류 120 과 156 의 효과는 동일합니다 .



NC 프로그램에서 유형 번호가 테이블에 있는 것과 다르게 지정된 경우 , 시스템은 자동적으로 공구 종류 110 (원통형 다이 싱크 밀링 공구) 을 사용합니다 . 공구 데이터 제한값을 넘어서면 알람이 발생합니다 .

커터 종류	종류 번호	R	r	a
원통형 다이 싱크 밀링 커터	110	> 0	-	-
볼 엔드 밀	111	> 0	>R	-
엔드 밀 , 앵글 헤드 커터	120, 130	> 0	-	-
엔드 밀 , 코너 라운딩을 사용하는 앵글 헤드 커터	121, 131	>r	> 0	-
베벨 커터	155	> 0	-	> 0
코너 라운딩 베벨 커터	156	> 0	> 0	> 0
테이퍼 다이 싱크 커터	157	> 0	-	> 0

R = 샤프트 반경 (공구 반경)

r = 코너 반경

a = 공구 세로 방향 축과 토러스 표면의 상단 사이의 각도

- = 평가되지 않습니다 .

공구 데이터	공구 파라미터	
공구 치수	기하	마모량
R	$TC_DP6	$TC_DP15
r	$TC_DP7	$TC_DP16
a	$TC_DP11	$TC_DP20

공구 길이 옵셋

공구 팁은 길이 옵셋의 원점입니다 (교차 세로 방향 축 / 표면)

3D 공구 옵셋 , 공구 교환

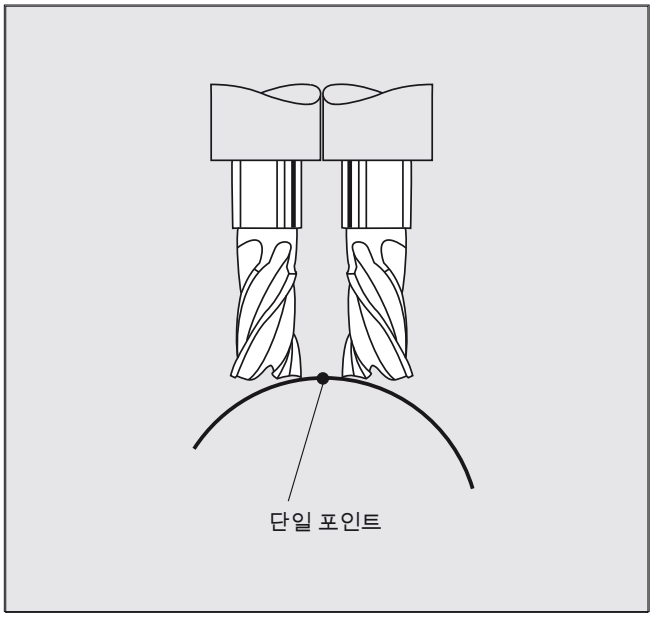
치수가 수정된 새로운 공구 (R, r, a) 또는 다른 샤프트는 G41 또는 G42 의 프로그래밍을 사용해서만 지정될 수 있습니다 (G40 에서 G41 또는 G42 로 이동 , G41 또는 G42 의 재프로그래밍) 이 법칙은 공구 길이 등 이외의 그 어떤 공구 데이터에도 적용되지 않기 때문에 G41 또는 G42 를 재프로그래밍 하지 않고도 공구를 로드할 수 있습니다 .

7.5.4 3D 공구 옵셋 경로 , 경로 곡률 , 삽입 깊이 (CUT3DC, ISD) 의 보정

작동

경로 보정

페이스 밀링과 관련하여 , 볼록한 표면을 수직 위치 공구로 가공한 오른쪽 예제에 나와 있는 것처럼 접점 (Contact Point) 이 공구 표면으로 뛰어오를 때 어떤 일이 발생하는지 조사하는 것이 좋습니다 . 예제에 나와 있는 용도는 경계선 사례로 간주됩니다 .



경계선 사례는 공구와 표면 벡터 사이의 각 접근 이동을 기초로 하여 가공점에서 갑작스런 변화를 감지하는 제어에 의해 모니터링 됩니다 . 제어가 이 위치에 선형 블록을 삽입하기 때문에 움직임이 실행될 수 있습니다 .

이 선형 블록은 머신 데이터에 저장된 사이드 각에 허용된 각범위를 기반으로 계산됩니다 . 머신 데이터에 저장된 제한값을 넘어서는 경우 , 시스템은 알람을 출력합니다 .

경로 곡률

경로 곡률이 모니터링되지 않습니다 . 이런 경우 , 형상을 침범하지 않는 형태의 공구만 사용하는 것이 좋습니다 .

삽입 깊이 (ISD)

삽입 깊이 ISD 는 3D 공구 반경 보정이 활성화되어 있을 때만 평가됩니다 .

프로그램 명령 ISD(삽입 깊이) 는 원주 밀링 작업에 대한 공구 삽입 깊이를 프로그램할 때 사용됩니다 . 이렇게 사용하면 공구의 외부 곡면에 가공점의 위치를 변경할 수 있게 됩니다 .

구문

3D 공구 보정 원주 밀링

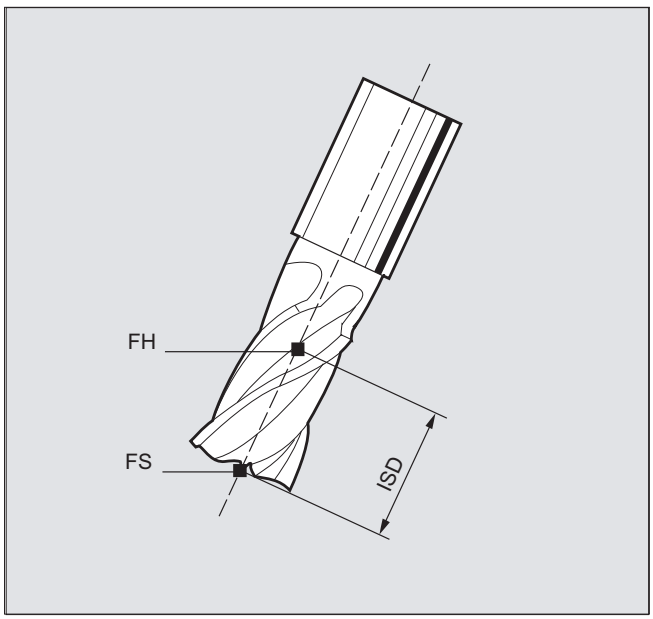
```
CUT3DC
ISD=<value>
```

중요도

CUT3DC	사면벽을 사용한 포켓 밀링 등 원주 밀링에 대한 3D 공구 옵셋을 활성화합니다 .
ISD	밀링 공구 팁 (FS) 및 밀링 공구 구문점 (FH) 사이의 공차 (<value>) 는 명령 ISD 를 사용하도록 지정되어 있습니다 .

밀링 공구 원점

밀링 공구 원점 (FH) 은 공구 축에 프로그램된 가공점을을 프로젝트할 때 얻어집니다 .



그 밖의 정보

CUT3DC 를 사용하는 원주 밀링에 대해 경사면 벽을 사용하는 포켓 밀링

3D 공구 반경 보정에서 , 밀 반경의 편차는 가공하게 될 표면의 표준을 향한 절입에 의해 보정됩니다 . 밀링 공구 페이스가 위치해 있는 평면은 삽입 깊이 ISD 가 동일하게 남아 있는 한 변경되지 않습니다 . 예를 들어 , 표준 공구 반경보다 작은 밀링 공구는 역시 한계 표면인 포켓 베이스에 도달하지 못할 것입니다 . 자동 공구 절입을 하기 위해서는 제어가 이 한계 표면을 알고 있어야 합니다 . " 한계 표면을 사용한 3D 원주 밀링 " 섹션을 참조하십시오 .

충돌 모니터링에 관해 자세한 정보를 보시려면
참조 문서 :
프로그래밍 메뉴얼 기본 ; " 공구 옵셋 " 장을 참조하십시오 .

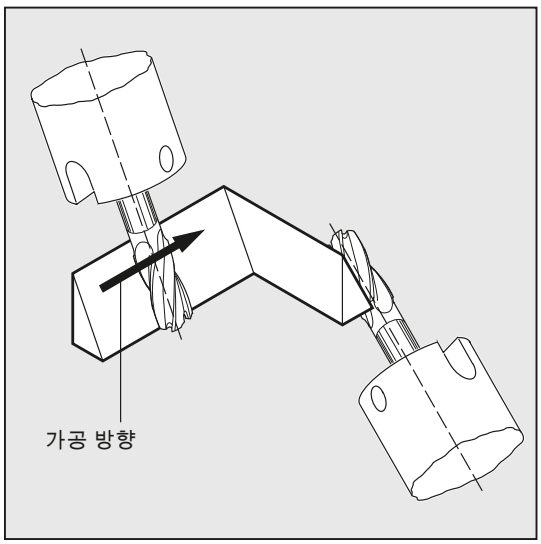
7.5.5 3D 공구 옵셋 내부 / 외부 코너 및 교차 절차 (G450/G451)

작동

내부 코너 / 외부 코너

내부 및 외부 코너는 별도로 취급됩니다 . 내부 코너 및 외부 코너의 정의는 공구 방향에 따라 달라집니다 .

예를 들어 , 코너에서 방향이 바뀌면 , 가공을 하는 동안 , 코너 유형이 변경될 수 있습니다 . 이렇게 변경될 때마다 , 오류 메시지와 함께 가공 작업이 중단됩니다 .



구문

```
G450
G451
```

중요도

G450	전이 원 (공구는 원형 경로에서 둥근 공작물 코너를 이동)
G451	등거리 경로의 교차 (공구는 공작물 코너에서 물러남)

그 밖의 정보

3D 보정에 대한 교차 절차

이제 3D 원주 밀링을 사용하여 , G 코드 G450/G451 이 평가됩니다 . 즉 , 옵셋 곡선의 교차점 복귀할 수 있습니다 . 원은 최대 SW 4 까지 항상 외부 코너에 삽입됩니다 . 교차 절차는 일반적으로 CAD에 의해 생성되는 3D 프로그램을 사용할 때 특히 편리합니다. 이 CAD는 짧은 직선 블록으로 구성되어 있는 경우가 많습니다 (부드러운 곡선에 가까워짐). 여기에서 인접한 블록 사이의 이동은 거의 관계가 없습니다 .

지금까지 , 형상의 외부에 공구 반경 보정을 사용하여 , 원은 보통 외부 코너를 우회하도록 삽입되었습니다 . 이 블록은 접선 전이를 할 때 길이가 모자라는 경우가 많아 , 결과적으로 뜻하지 않는 속도 저하를 가져옵니다 .

이렇게 속도가 저하되면 , 2 ½ D 반경 보정의 경우처럼 , 관련된 2 개의 곡선이 확장됩니다 . 두 개의 확장된 곡선의 교차점이 서로 접근합니다 .

교차는 2 개의 관련된 블록의 옵셋 곡선의 확장 및 공구 방향에 대한 평면 수직의 코너에 있는 블록 2 개의 교차 정의에 의해 결정됩니다 . 이런 교차가 없는 경우 , 코너는 이전 , 즉 , 원이 삽입된 것과 같이 처리됩니다 .

교차 절차에 관한 자세한 정보는
참조 문서 :
기능 메뉴얼 , 특수 기능 ; 3D 공구 반경 보정 (W5) 을 참조하십시오 .

7.5.6 3D 공구 옵셋 한계 표면을 사용한 3D 원주 밀링

CAD 프로그램의 조건에 대한 3D 원주 밀링의 개조

CAD 시스템으로 생성된 NC 프로그램은 일반적으로 짧은 선형 블록의 큰 번호를 사용하는 표준 공구의 중앙 경로에 가까워집니다 . 수 많은 가공 형상의 블록이 이런 방식으로 생성되도록 하기 위해서는 원본 형상을 가능한 한 정밀하게 매핑해야 합니다. 가공 프로그램에 특정한 변경을 가하기 위해서는 필요한 과정입니다 .

보정을 최적화하는데 필요한 중요한 정보는 더 이상 부품 프로그램에서 사용할 수 없으므로 적절한 방법으로 교체해야 합니다 . 여기에는 중요한 전이를 보정할 수 있는 일반적인 방법이 나와 있습니다. 이 방법은 부품 프로그램에서 직접적인 방법을 사용하거나 실제 형상을 결정할 때 (예 : 공구 절입 사용) 사용할 수 있는 방법입니다 .

애플리케이션

표준 공구를 대신한 일반적인 사용 방법에 추가하여 , 실제 공구는 중심점 경로와 3D 공구 보정을 사용하는 실린더 공구를 설명합니다 . 이런 경우 , 프로그램된 경로는 가공 표면에 있는 형상을 참조합니다 . 관련 한계 표면은 공구로부터 독립적입니다 . 기존의 공구 반경 보정과 동일하게 , 전체 반경은 한계 표면의 수직 옵셋을 계산하는 데 사용됩니다 .

7.5.7 3D 공구 옵셋 한계 표면 고려 (CUT3DCC, CUT3DCCD)

작동

실제 공구를 사용하는 3D 원주 밀링

공구 방향의 지속적인 변경 또는 형상을 사용하는 3D 원주 밀링에서 , 공구 중심점 경로는 정의된 표준 공구에 대해 자주 프로그램됩니다 . 실제 상황에서 적절한 표준 공구는 사용하지 못하는 경우가 많기 때문에 표준 공구에서 너무 많이 벗어나지 않은 공구를 사용할 수도 있습니다 .

CUT3DCCD 은 프로그램된 표준 공구가 정의한 실제 차등 공구에 대한 한계 표면을 고려합니다 . NC 프로그램은 표준 공구의 중심점 경로를 정의합니다 .

실린더 공구를 사용하는 CUT3DCC 는 프로그램된 표준 공구가 도달했을 한계 표면을 고려합니다 . 가공 표면의 형상을 정의하는 NC 프로그램

구문

```
CUT3DCCD
CUT3DCC
```

중요도

CUT3DCCD	공구 중심점 경로에서 다른 공구를 사용하는 한계 표면의 원주 밀링에 대한 3D 공구 옵셋 활성화 : 한계 표면으로 절입 .
CUT3DCC	한계 표면에 3D 반경 보정을 사용하는 원주 밀링에 대한 3D 공구 옵셋 활성화 : 형상의 가공 표면 .

주

G41, G42 를 사용한 공구 반경 보정

CUT3DCCD 또는 CUT3DCC 이 활성화되어 있는 상태에서 G41, G42 를 사용한 공구 반경 보정이 프로그램된 경우 , " 방향 변환 " 옵션을 활성화시켜야만 합니다 .

코너 라운딩을 사용하는 표준 공구

표준 공구를 사용한 코너 라운딩은 공구 파라미터 $TC_DP7 에 의해 정의됩니다 . 공구 파라미터 $TC_DP16 은 표준 공구와 비교했을 때 실제 공구의 코너 라운딩의 편차를 설명합니다 .

예제

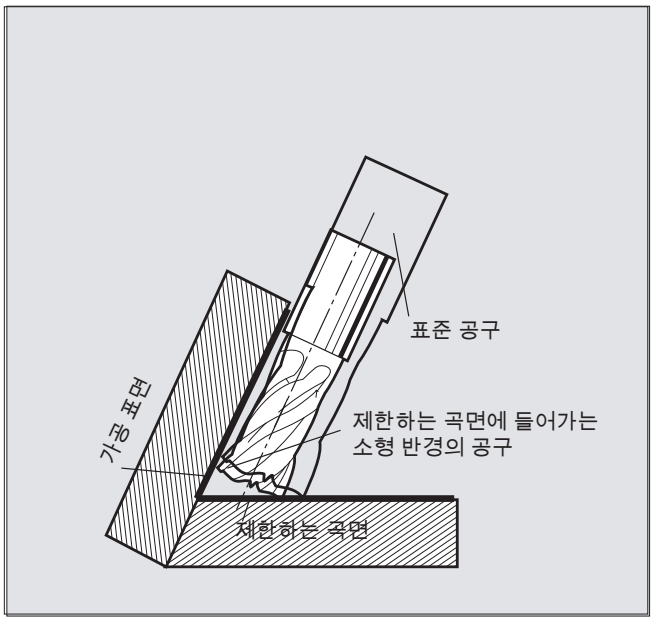
표준 공구와 비교했을 때 반경이 감소한 환상면 밀러의 공구 수치 .

공구 종류	R = 생크 반경	r = 코너 반경
코너 라운딩을 사용하는 표준 공구	R = $TC_DP6	r = $TC_DP7
코너 라운딩을 사용하는 실제 공구 : 공구 유형 121 및 131 토러스 밀링 공구 (샤프트 밀링 공구)	R' = $TC_DP6 + $TC_DP15 + OFFN	r' = $TC_DP7 + $TC_DP16
이번 예제에서는 $TC_DP15 + OFFN 및 $TC_DP16 이 모두 음수입니다 . 공구 유형 ($TC_DP1) 이 평가됩니다 .		
원통형 샹크(실린더 또는 엔드 밀)를 사용하는 커터 종류 , 환상면 밀러 (유형 121 및 131) 및 제한된 경우 , 원통형 다이 밀 (유형 110) 만이 허용됩니다 .	이렇게 허용된 커터 종류에 있어서, 코너 반경 r은 샹크 반경 R과 동일합니다. 그 이외의 모든 허용된 공구 유형은 실린더 커터 및 코너 라운딩으로 지정된 수치는 평가되지 않은 것으로 해석됩니다 .	
111 번 및 155, 156, 157 번을 제외한 1-399 번의 모든 공구 유형이 허용됩니다 .		

그 밖의 정보

한계 표면 CUT3DCCD 까지 절입되는 공구 중심점

적절한 표준 공구의 반경보다 작은 공구를 사용하는 경우 , 가공은 밀링 공구를 사용하여 실행됩니다. 이 밀링 공구는 포켓의 바닥(베이스)에 닿을 때까지 수직 방향으로 절입됩니다. 이 공구는 가공 표면 및 한계 표면에 의해 형성된 코너에서 최대한 많은 재료를 제거합니다 . 이 과정에는 원주 및 페이스 밀링을 혼합한 가공 형식이 포함됩니다 . 반경이 낮은 공구와 비슷하게 반경이 증가된 공구에 대한 절입은 반대 방향으로 이루어집니다 .



G 코드 그룹 22의 모든 공구 옵셋과 반대로, CUT3DCCD에 지정된 공구 파라미터 $TC_DP6는 공구 반경 및 그 결과로 나오는 보정에 영향을 주지 않습니다 .

보정 옵셋은 다음 사항의 합계입니다 .

- 공구 반경의 마모값 (공구 파라미터 $TC_DP15)
- 한계 표면에 대한 수직 옵셋을 계산하도록 프로그램된 공구 옵셋 OFFN

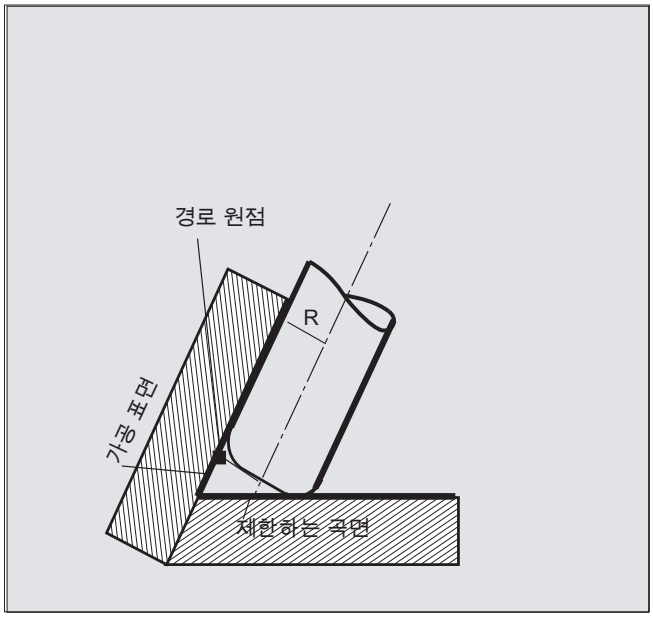
생성된 부품 프로그램에서는 가공될 표면이 경로의 오른쪽인지 왼쪽인지 지정하지 않습니다 . 그렇기 때문에 반경이 양수 값이고 원래 공구의 마모값이 음수 값이라고 가정하겠습니다. 음수 마모 값은 항상 직경이 작은 공구를 가리킵니다 .

원통형 공구 사용

원통형 공구를 사용하는 경우 , 가공 표면 및 한계 표면이 정확한 각을 이룰 때 (90 도 이하) 만 절입이 필요합니다 . 토러스 밀링 공구 (라운딩 된 코너를 사용하는 실린더) 가 사용된 경우 , 공구를 예각 및 둔각으로 절입할 때 수직 방향이 되어야 합니다 .

CUT3DCC, 가공 표면의 형상을 사용하는 3D 반경 보정

토러스 밀링 공구를 사용하여 CUT3DCC 가 활성화된 경우 , 프로그램된 경로는 수치가 동일한 가상의 실린더 밀링 공구를 참조합니다 . 결과 경로 원점은 다음의 토러스 밀링 공구에 대한 다이어그램에 나와 있습니다 .



가공 표면과 한계 표면 사이의 각도는 동일한 블록 내라 할지라도 예각에서 둔각 또는 둔각에서 예각으로 변경될 수 있습니다 .

이 공구는 실질적으로 표준 공구보다 크건 작건 상관 없이 사용됩니다 . 하지만 결과 코너 반경은 음수가 되어야 하며 , 결과 공구 반경 기호는 반드시 유지해야 합니다 .

CUT3DCC 에서 , NC 부품 프로그램은 가공 표면의 형상을 참조합니다 . 기존의 공구 반경 보정에 관한한 , 총 공구 반경은 다음의 합계를 구성하는 데 사용됩니다 .

- 공구 반경 (공구 파라미터 $TC_DP6)
- 마모값 (공구 파라미터 $TC_DP15)
- 한계 표면에 대한 수직 옵셋을 계산하도록 프로그램된 공구 옵셋 OFFN

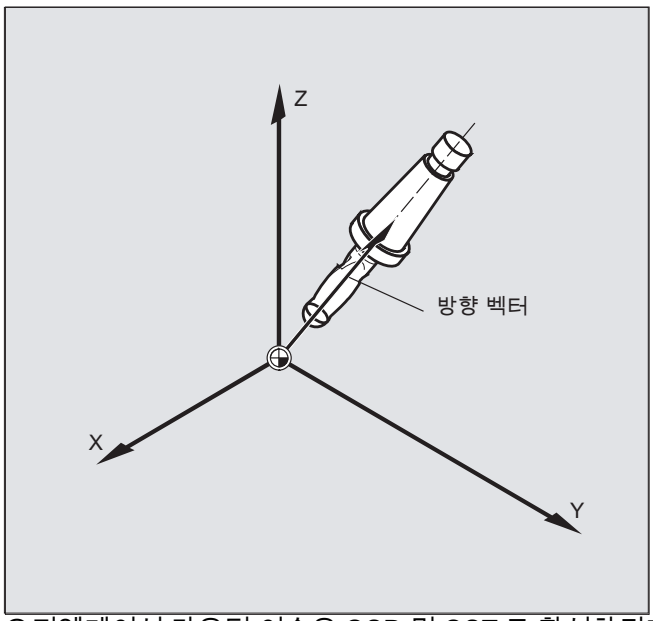
한계 표면의 위치는 다음에 나와 있는 두 개의 값의 차이에 의해 결정됩니다 .

- 표준 공구의 치수
- 공구 반경 (공구 파라미터 $TC_DP6)

7.6 공구 오리엔테이션 (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST)

기능

공구 오리엔테이션이란 공간 내에서 공구의 기하학적 정렬을 말합니다 . 5 축 기계에서는 프로그램 명령을 사용하여 공구 오리엔테이션을 설정할 수 있습니다 .



오리엔테이션 라운딩 이송은 OSD 및 OST 로 활성화되며 , 이 둘은 공구 오리엔테이션에 대한 보간 유형에 따라 다르게 형성됩니다 .

벡터 보간이 활성화되어 있는 경우 , 스무딩된 오리엔테이션 특성 역시 벡터 보간을 사용하여 보간할 수 있습니다. 반면, 로터리 축 보간이 활성화되어 있는 경우, 그 오리엔테이션은 로터리 축 움직임을 직접 사용하여 스무딩 됩니다 .

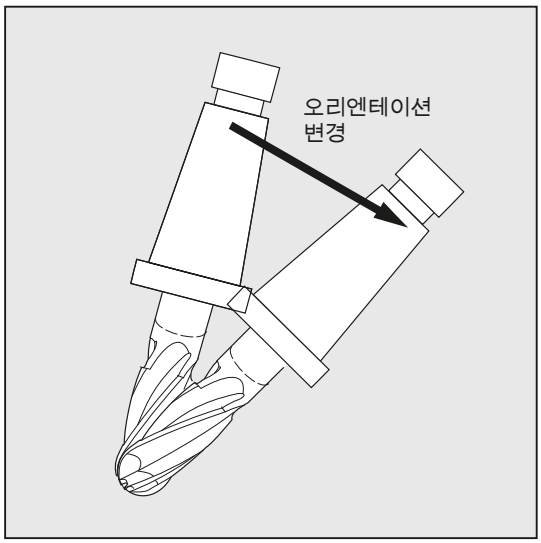
프로그래밍

오리엔테이션 변경 프로그래밍 :

공구 오리엔테이션을 변경하도록 프로그램할 수 있는 방법

- 로터리 축 A, B, C 를 직접 프로그래밍 (로터리 축 보간)
- 오일러 (Euler) 각도 또는 RPY 각도
- 방향 벡터 (A3, B3 또는 C3 지정을 통한 벡터 보간)
- LEAD/TILT (페이스 밀링)

기준 좌표계는 기계 좌표계 (ORIMKS) 또는 현재 공작물 좌표계 (ORIWKS) 중 하나가 될 수 있습니다 .



공구 오리엔테이션 프로그래밍 :

명령	의미
ORIC:	오리엔테이션 및 경로 이송이 평행
ORID:	오리엔테이션 및 경로 이송이 일관됨
OSOF:	오리엔테이션 스무딩 없음
OSC:	일관된 오리엔테이션
OSS:	블록 시작에서만 오리엔테이션 스무딩
OSSE:	블록의 시작 및 끝에서 오리엔테이션 스무딩
ORIS:	밀리미터 (mm) 당 각도 단위로 활성화된 오리엔테이션 스무딩을 사용한 오리엔테이션 변경 속도 (OSS 및 OSSE 에 적용)
OSD:	셋팅 데이터로 스무딩 거리를 지정하여 오리엔테이션 스무딩 : SD42674 $SC_ORI_SMOOTH_DIST
OST:	셋팅 데이터로 벡터 보간에 대한 각도 공차를 지정하여 오리엔테이션 스무딩 : SD42676 $SC_ORI_SMOOTH_TOL 로터리 축 보간을 사용하여 지정되어 있는 공차가 오리엔테이션 축의 최대 편차가 되도록 하십시오 .

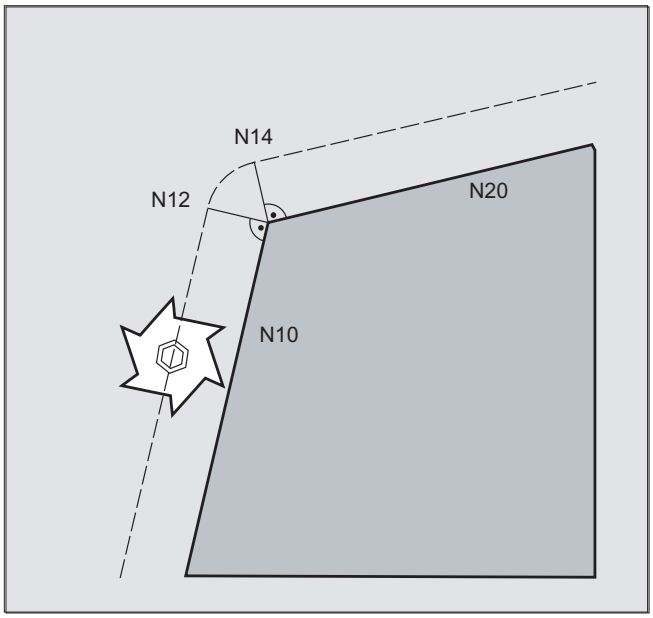
주

공구 오리엔테이션을 스무딩하기 위한 모든 명령 (OSOF, OSC, OSS, OSSE, OSD 및 OST) 은 G 코드 그룹 34 에 요약되어 있습니다 . 이들은 모두 모달입니다 . 따라서 이들 명령은 한 번에 하나씩만 적용할 수 있습니다 .

예제

예제 1: ORIC

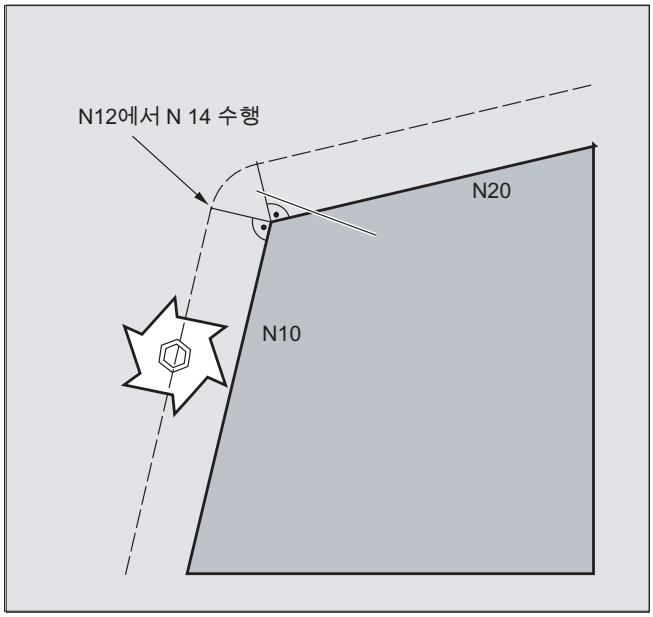
ORIC이 활성화되어 있고, N10 및 N20 이송 블록 사이에 프로그램 된 방향(예:A2= B2= C2=) 이 변경된 블록이 둘 이상이 있는 경우 , 삽입된 원 블록이 각도의 절대 변경에 따라 중간 블록으로 배분됩니다 .



프로그램 코드	설명
ORIC	
N8 A2=… B2=… C2=…	
N10 X… Y… Z…	
N12 C2=… B2=… N14 C2=… B2=…	; 외부 코너에 삽입된 원호 블록은 오리엔테이션 변경에 따라 N12와 N14 사이에 배분됩니다 . 원호 이송 및 오리엔테이션 변경은 평행하게 실행됩니다 .
N20 X =…Y=… Z=… G1 F200	

예제 2: ORID

ORID 가 활성화되면 , 2 개의 이송 블록 사이에 있는 모든 블록은 첫 번째 이송 블록의 끝에서 실행됩니다 . 일정 오리엔테이션을 사용하는 원호 블록은 두 번째 이송 블록 직전에 실행됩니다 .



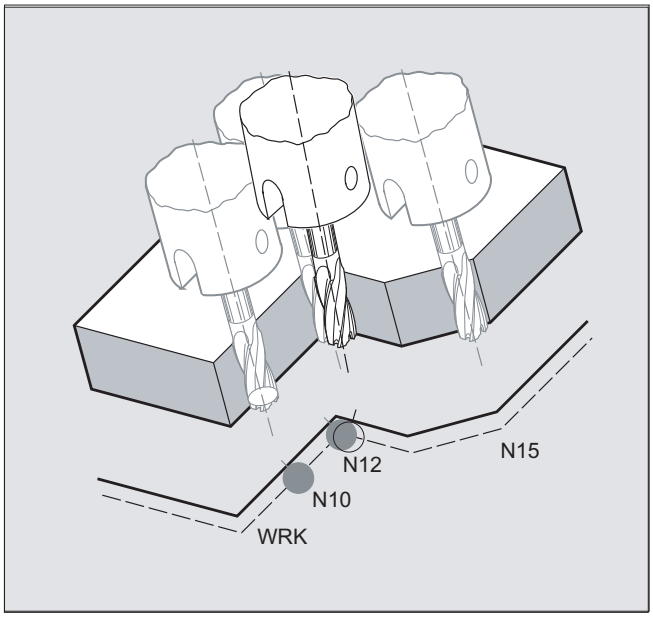
프로그램 코드	설명
ORID	
N8 A2=… B2=… C2=…	
N10 X… Y… Z…	
N12 A2=… B2=… C2=…	; N12 및 N14 블록은 N10 블록의 끝에서 실행됩니다 . 그 이후 원호 블록은 실제 오리엔테이션으로 실행됩니다 .
N14 M20	; 도움말 기능 등
N20 X… Y… Z…	

주

외부 형상에서 오리엔테이션 변경에 사용되는 방법은 외부 코너의 첫 번째 이송 블록에서 활성화된 프로그램 명령을 사용하여 결정됩니다 .

오리엔테이션 변경 없음 : 블록 경계에서 오리엔테이션이 변경되지 않은 경우 , 형상의 교차 섹션은 양쪽 형상과 모두 접촉하는 원이 됩니다 .

예제 3: 내부 코너에서 오리엔테이션 변경



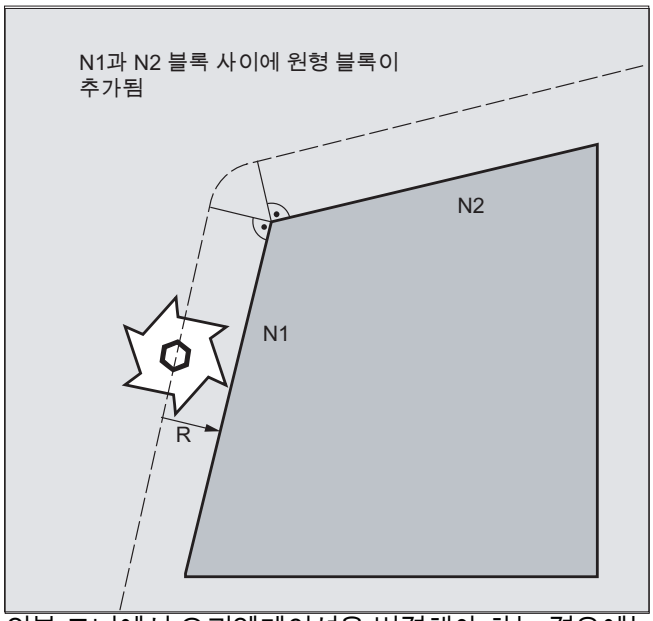
프로그램 코드
ORIC
N10 X …Y… Z… G1 F500
N12 X …Y… Z… A2=… B2=… C2=…
N15 X …Y… Z… A2=… B2=… C2=…

추가 정보

외부 코너에서의 동작

커터의 반경을 사용하는 원호 블록은 항상 외부 코너에 삽입됩니다 .

ORIC 및 ORID 프로그램 명령은 N1 및 N2 블록 사이에 프로그램된 오리엔테이션을 변경할 것인지 결정하는 데 사용되며 , 삽입된 원호 블록이 처리되기 전에 실행되거나 또는 동시에 실행됩니다 .



외부 코너에서 오리엔테이션을 변경해야 하는 경우에는 경로 이동과 함께 별도로 실행되거나 또는 보간과 동시에 실행되도록 할 수 있습니다 .

ORID 가 프로그램된 경우 , 삽입된 블록은 경로 이동 없이 먼저 실행됩니다 . 코너를 생성하는 원호 블록은 두 개의 이송 블록 중 두 번째 블록 직전에 삽입됩니다 .

몇 개의 오리엔테이션 블록이 외부 코너에 삽입되고 , ORIC 가 선택된 경우 , 원호 이송은 오리엔테이션 변경의 절대값에 따라 개별적으로 삽입된 블록 사이로 배분됩니다 .

OSD 또는 OST 를 사용한 오리엔테이션 스무딩

G642를 사용하여 블렌딩할 때, 기하 축 및 오리엔테이션 축에 대한 최대 편차는 되도록 적어야 합니다 . 공차 중 작은 쪽은 스무딩 모션 및 / 또는 각도 공차의 유형을 결정합니다 . 따라서 높은 형상 편차를 수용하지 않고도 오리엔테이션 특성을 강하게 스무딩합니다 .

심한 형상 편차 없이 지정된 스무딩 거리 및 각도 공차대로 오리엔테이션 특성에서 아주 조금만 벗어나게 스무딩하도록 OSD 및 OST 를 활성화할 수 있습니다 .

주

G642를 사용한 형상 (및 오리엔테이션 특성) 의 라운딩 가공과는 달리, OSD 및/또는 OST를 사용하는 오리엔테이션을 라운딩할 때 , 별도의 블록이 형성되지 않는 대신 라운딩 움직임이 프로그램된 원본 블록에 직접 추가됩니다 .

OSD 및 / 또는 OST 를 사용하는 블록 트렌지션은 공구 오리엔테이션에 대한 보간의 유형이 변경 (벡터 → 로터리 축 , 로터리 축 → 벡터) 된 경우에는 라운딩이 될 수 없습니다 . 이러한 블록 트렌지션은 필요한 경우 , 표준 라운딩 기능 G641, G642 및 G643 을 사용하여 라운딩될 수 있습니다 .

7.7 D 번호 , 절삭날 번호의 자유 할당

7.7.1 D 번호 , 절삭날 번호의 자유 할당 (CE 어드레스)

D 번호

D 번호는 형상 번호로도 사용될 수 있습니다 . 어드레스 CE 를 통해 절삭날 번호를 어드레스 지정할 수도 있습니다 . 절삭날 번호를 설명하기 위해 시스템 변수인 $TC_DPCE 를 사용할 수 있습니다 .

기본값 : 보정 번호 == 공구 날 번호

머신 데이타는 D 번호의 최대 번호 (절삭날 번호) 및 공구 당 절삭날의 최대 갯수 (→ 기계류 구문 OEM)를 정의하는 데 사용됩니다. 다음 명령은 최대 절삭 번호(MD18105)가 공구 당 절삭 번호 (MD18106) 보다 클 때만 사용됩니다 . 기계 제조업체의 사양을 참조하십시오 .

주

D 번호 할당과 관련하여 추가적으로 D 번호는 " 평평한 " 또는 " 절대 " D 번호로도 할당될 수 있습니다 . 이 때 T 번호 (" 평평한 D 번호 구조 " 기능 내) 는 참조하지 않습니다 .

참조 문서

기능 메뉴얼 기본 기능 ; 공구 옵셋 (W1)

7.7.2 D 번호의 자유 할당 : D 번호 확인 (CHKDNO)

작동

CKKDNO 명령을 사용하면 기존 D 번호가 독자적으로 할당된 것인지 확인할 수 있습니다 . 모든 공구의 D 번호는 TO 유닛 내에서 정의되며 , 한 번만 나옵니다 . 예비 공구는 허용하지 않습니다 .

구문

```
state=CHKDNO(Tno1,Tno2,Dno)
```

중요도

state	= TRUE:	D 번호는 확인된 영역에만 독자적으로 할당됩니다 .
	FALSE :	D 번호 충돌이 발생했거나 파라미터가 유효하지 않습니다 . Tno1, Tno2 및 Dno 는 충돌을 야기한 파라미터를 반환합니다 . 이 데이터는 부품 프로그램에서 평가됩니다 .
CHKDNO(Tno1,Tno2)	지정된 부품의 D 번호가 모두 확인되었습니다 .	
CHKDNO(Tno1)	Tno1 의 D 번호가 다른 공구와 비교하여 모두 확인되었습니다 .	
CHKDNO	공구의 D 번호가 다른 공구와 비교하여 모두 확인되었습니다 .	

7.7.3 D 번호의 자유 할당 : D 번호 이름 변경 (GETDNO, SETDNO)

작동

독자적인 D 번호를 할당하셔야만 합니다 . 2 개의 서로 다른 공구의 절삭날에 동일한 D 번호를 할당해서는 안 됩니다 .

GETDNO

이 명령은 공구 번호 t 를 사용하는 공구의 특정 절삭날 (ce) 의 D 번호를 반환합니다 . 입력된 파라미터에 대한 D 번호가 없는 경우에는 d=0 으로 설정됩니다 . D 번호가 유효하지 않는 경우에는 32000 이상의 값이 반환됩니다 .

SETDNO

이 명령은 공구 t 의 절삭날 ce 에 대한 D 번호의 값인 d 를 할당합니다 . 이 문구의 결과는 상태 (TRUE 또는 FALSE) 를 통해 반환됩니다 . 지정된 파라미터에 대한 데이터 블록이 없는 경우에는 FALSE 값이 반환됩니다 . 구문 오류가 발생하면 알람이 발생합니다 . D 번호를 직접 0 으로 설정해서는 안 됩니다 .

구문

d = GETDNO (t,ce)

state = SETDNO (t,ce,d)

중요도

d	공구 날의 D 번호
t	공구의 T 번호
ce	공구의 절삭날 번호 (CE 번호)
state	명령의 실행 여부를 나타냅니다 (TRUE 또는 FALSE).

D 번호 이름 변경의 예제

프로그래밍	비고
$TC_DP2[1.2]=120	;
$TC_DP3[1,2] = 5.5	;
$TC_DPCE[1,2] = 3	; 절삭날 번호 CE
...	;
N10 def int DNoOld, DNoNew = 17	;
N20 DNoOld = GETDNO(1,3)	;
N30 SETDNO(1,3,DNoNew)	;

새로운 D 값인 17 은 이제 절삭날 CE=3 에 할당됩니다 . 이제 절삭날에 대한 데이터는 D 번호 17 을 통해 어드레스됩니다 . 이 둘은 모두 시스템 변수를 통하거나 NC 어드레스를 사용하는 프로그래밍을 통합니다 .

7.7.4 D 번호의 자유 할당 : 지정된 D 번호의 T 번호를 결정 (GETACTTD)

작동

GETACTTD 명령을 사용하여 절대 D 번호와 관련된 T 번호를 결정합니다 . 이 때는 독자성에 대해서는 확인하지 않습니다 . TO 유닛 내에 있는 몇 개의 D 번호가 동일한 경우 , 검색에서 발견된 첫 번째 공구의 T 번호가 반환됩니다 . 이 명령은 " 평평한 " D 번호를 사용할 때는 적절하지 않습니다 . 이 경우 (데이터베이스 내에 T 번호가 없음), 값 "1" 은 항상 반환되기 때문입니다 .

구문

```
status=GETACTTD(Tnr,Dnr)
```

중요도

Dno	검색하게 될 T 번호에 대한 D 번호	
Tno	발견된 T 번호	
status	값 :	의미 :
	0	T 번호가 발견되었습니다 . Tno 에는 T 번호의 값이 포함되어 있습니다 .
	-1	지정된 D 번호에 대한 T 번호가 없습니다 . Tno=0
	-2	D 번호가 절대치가 아닙니다 . Tno 가 첫 번째 발견된 공구의 값을 받았습니다 . 이 값은 Dno 값을 사용하는 D 번호를 포함하고 있습니다 .
	-5	다른 이유로는 이 기능을 실행할 수 없습니다 .

7.7.5 D 번호의 자유 할당 : 무효화된 D 번호 (DZERO)

작동

DZERO 명령은 공구 정비를 지원하는 데 사용됩니다. 이 명령이 붙어 있는 보정 데이터 세트는 더 이상 CHKDNO 명령에 의해 검증되지 않습니다 . 이런 데이터 세트는 SETDNO 를 사용하여 D 번호를 설정하면 다시 액세스할 수 있게 됩니다 .

구문

```
DZERO
```

중요도

DZERO	TO 유닛의 D 번호를 모두 유효하지 않은 것으로 표시합니다 .

7.8 공구 홀더 구조

필요 조건

공구 홀더는 공간 내에서 다음과 같은 경우에만 사용 가능한 모든 방향에서 공구의 방향을 잡을 수 있습니다 .

- 2 개의 로타리 축인 V_1 및 V_2 가 있는 경우
- 로타리 축이 서로 직각인 경우
- 공구의 세로 축이 두 번째 로타리 축인 V_2 에 대해 수직인 경우

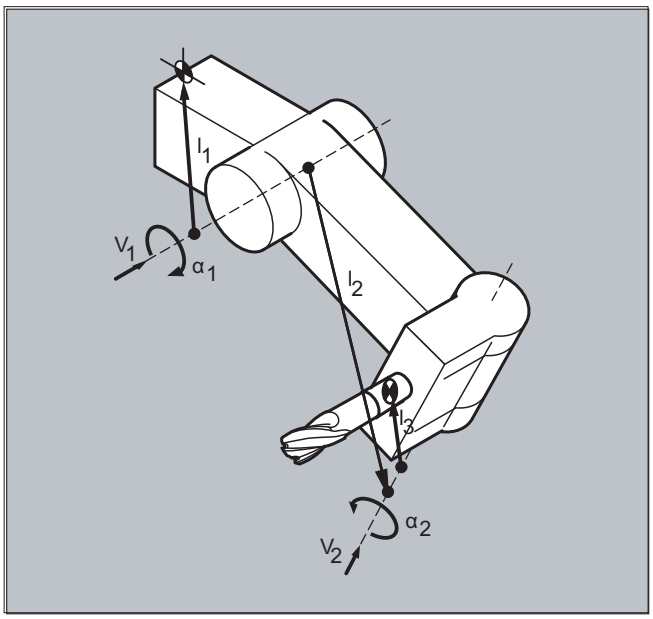
또한 , 모든 사용 가능한 방향을 설정할 수 있는 경우 기계에 다음 요구 사항이 적용됩니다 .

- 공구의 세로 축이 첫 번째 로타리 축인 V_1 에 대해 수직이 되어야 합니다

작동

최대 2 개의 로타리 축인 v_1 또는 v_2 를 사용하는 공구 홀더 구조는 17 개의 시스템 변수인 $TC_CARR1[m]부터 $TC_CARR17[m]을 사용하여 정의됩니다. 공구 홀더의 설명은 다음으로 구성되어 있습니다 .

- 공구 홀더의 첫 번째 로타리 축으로부터 벡터 거리 l_1, 첫 번째 로타리 축으로부터 두 번째 로타리 축까지의 벡터 거리 l_2, 두 번째 로타리 축으로부터 공구의 원점까지의 벡터 거리 l_3.
- 로타리 축 V_1, V_2 의 방향 벡터 .
- 2개 축의 회전 각도 α_1, α_2 회전 각도는 로타리 축 벡터는 보기 방향, 즉 회전의 시계 방향인 양의 방향으로 계산됩니다 .



결정된 구조를 사용하는 기계 (공구 및 부품 모두 회전 가능) 에서 , 시스템 변수는 엔트리를 사용하여 확장되었습니다 .

- $TC_CARR18[m] 부터 $TC_CARR23[m].

파라미터

방향 조정이 가능한 공구 홀더에 대한 시스템 변수의 기능			
지정	**x 요소**	**y 요소**	**y 요소**
l_1 옵셋 벡터	$TC_CARR1[m]	$TC_CARR2[m]	$TC_CARR3[m]
l_2 옵셋 벡터	$TC_CARR4[m]	$TC_CARR5[m]	$TC_CARR6[m]
v_1 로타리 축	$TC_CARR7[m]	$TC_CARR8[m]	$TC_CARR9[m]
v_2 로타리 축	$TC_CARR10[m]	$TC_CARR11[m]	$TC_CARR12[m]
α_1 회전 각도 α_2 회전 각도	$TC_CARR13[m] $TC_CARR14[m]		
l_3 옵셋 벡터	$TC_CARR15[m]	$TC_CARR16[m]	$TC_CARR17[m]

방향 조정이 가능한 공구 홀더에 대한 시스템 변수의 확장			
이름 :	**x 요소**	**y 요소**	**z 요소**
l_4 옵셋 벡터	$TC_CARR18[m]	$TC_CARR19[m]	$TC_CARR20[m]
축 식별자 로타리 축 v_1 로타리 축 v_2	로타리 축의 축 식별자 v_1 및 v_2(0 으로 시작) $TC_CARR21[m] $TC_CARR22[m]		
운동학적 유형 **공**구 **부**품 **혼**합 모드	$TC_CARR23[m]		
	운동학적 유형 T ->	운동학적 유형 P ->	운동학적 유형 M
	공구만 회전 가능 (기본).	부품만 회전 가능	부품 및 공구 회전 가능
옵셋 로타리 축 v_1 로타리 축 v_2	초기 설정 $TC_CARR24[m] $TC_CARR25[m] 을 취하는 로타리 축 v_1 및 v_2 의 각도		
각도 옵셋 로타리 축 v_1 로타리 축 v_2	로타리 축 v_1 및 v_2 $TC_CARR26[m] $TC_CARR27[m] 에 대한 각도에서 Hirth 투스 시스템의 옵셋		
각도 증분 v_1 로타리 축 v_2 로타리 축	로타리 축 v_1 및 v_2 $TC_CARR28[m] $TC_CARR29[m] 에 대한 각도에서 Hirth 투스 시스템의 옵셋		
최소 위치 로타리 축 v_1 로타리 축 v_2	로타리 축 v_1 and v_2 $TC_CARR30[m] $TC_CARR31[m] 의 최소 위치에 대한 소프트웨어 제한		
최대 위치 로타리 축 v_1 로타리 축 v_2	로타리 축 v_1 및 v_2 $TC_CARR32[m] $TC_CARR33[m] 의 최대 위치에 대한 소프트웨어 제한		
공구 홀더 이름	공구 홀더에는 번호 대신 이름을 붙일 수 있습니다 . $TC_CARR34[m]		

방향 조정이 가능한 공구 홀더에 대한 시스템 변수의 확장			
사용자 : 축 이름 1 축 이름 2 식별자 **위치**	사용자 측정 싸이클 $TC_CARR35[m] $TC_CARR36[m] $TC_CARR37[m] 에서 의도적으로 사용		
	$TC_CARR38[m]	$TC_CARR39[m]	$TC_CARR40[m]
미세조정 옵셋	파라미터는 **기본 파라미터에 있는 값**에 추가될 수 있습니다 .		
l_1 옵셋 벡터	$TC_CARR41[m]	$TC_CARR42[m]	$TC_CARR43[m]
l_2 옵셋 벡터	$TC_CARR44[m]	$TC_CARR45[m]	$TC_CARR46[m]
l_3 옵셋 벡터	$TC_CARR55[m]	$TC_CARR56[m]	$TC_CARR57[m]
l_4 옵셋 벡터	$TC_CARR58[m]	$TC_CARR59[m]	$TC_CARR60[m]
v_1 로타리 축	$TC_CARR64[m]		
v_2 로타리 축	$TC_CARR65[m]		

주

파라미터에 대한 설명

"m" 은 프로그램 될 공구 홀더의 개수를 지정합니다 .

$TC_CARR47 부터 $TC_CARR54 및 $TC_CARR61 부터 $TC_CARR63 는 정의되지 않았으며 , 읽기 또는 쓰기 액세스가 시도되는 경우에는 알람이 생성됩니다 .

축에 있는 거리 벡터의 시작 / 끝점은 자유롭게 선택할 수 있습니다 . 2 축을 중심으로 하는 회전 각도 α_1, α_2 는 공구 홀더의 초기 상태인 0° 에서 정의됩니다 . 이런 방법으로 공구 홀더의 운동학은 사용 가능한 어떤 번호에 대해서도 프로그램이 될 수 있습니다 .

로타리 축이 없거나 , 한 개만 있는 공구 홀더는 한 개 또는 양쪽의 로타리 축의 방향 벡터를 0 으로 설정하여 설명할 수 있습니다 .
로타리 축이 없는 공구 홀더에서, 거리 벡터는 가공 평면(G17부터 G19)의 변경에 의해 영향을 받을 수 없는 요소를 갖고 있는 추가적인 공구 보정으로 작동합니다 .

파라미터 확장자

로타리 축의 파라미터

시스템 변수는 엔트리인 $TC_CARR24[m] 부터 $TC_CARR33[m] 까지 및 다음에 나와 있는 사항에 의해 확장되었습니다 .

로타리 축 v_1, v_2 의 **옵셋**	방향이 정해진 공구 홀더의 초기 설정에 대한 로타리 축 v_1 또는 v_2 의 위치가 변경됩니다 .
로타리 축 v_1, v_2의 **각도 옵셋 / 각도 증분**	로타리 축 v_1 및 v_2의 Hirth 투스 시스템의 옵셋 또는 각도 증분 프로그램 또는 계산된 각도는 n이 정수일 때 phi = s + n * 계산에서 나온 결론인 다음 값까지 라운딩이 됩니다 .
로타리 축 v_1, v_2의 **최소 위치 및 최대 위치**	로타리 축 v1 및 v2의 로타리 축 제한 각도(소프트웨어 제한)의 최소 및 최대 위치

사용자용 파라미터

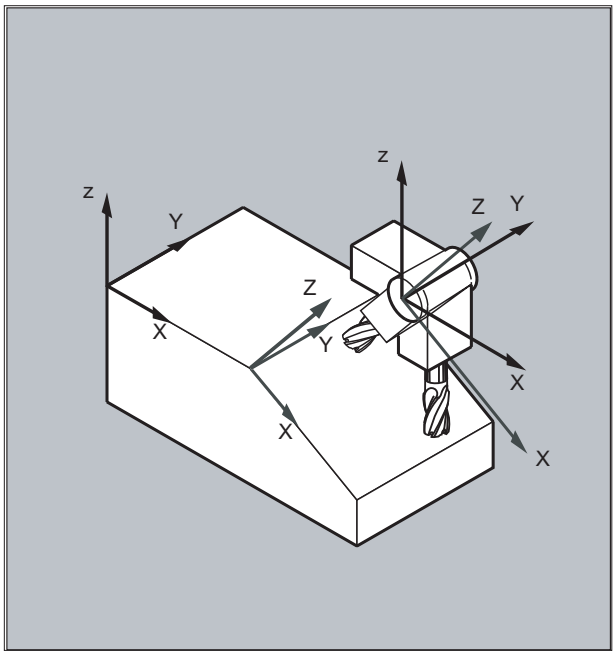
$TC_CARR34 to $TC_CARR40에는 파라미터가 포함되어 있습니다. 이 파라미터는 사용자가 자유롭게 사용할 수 있으며 , 최대 SW 6.4 까지 표준화되어 있습니다 . NCK 에서는 더 이상 평가되지 않으며 중요하지 않습니다 .

미세조정 옵셋 파라미터

$TC_CARR41 부터 $TC_CARR65 에는 기본 파라미터에 있는 값에 추가할 수 있는 미세조정 파라미터가 포함되어 있습니다. 기본 파라미터에 할당된 미세조정 옵셋 값은 파라미터 번호에 값 40 이 추가되었을 때 얻어집니다 .

예제

다음 예제에서 사용되는 공구 홀더는 Y 축을 중심으로 한 회전에 의해 모두 설명될 수 있습니다 .



프로그램 코드	비고
N10 $TC_CARR8[1]=1	; 공구 홀더 1 의 첫 번째 로타리 축의 Y 요소에 대한 설명
N20 $TC_DP1[1,1] = 120	; 샤프트 밀러에 대한 설명
N30 $TC_DP3[1,1]=20	; 길이 20mm 의 샤프트 밀러에 대한 설명
N40 $TC_DP6[1,1]=5	; 반경 5 mm 의 샤프트 밀러에 대한 설명
N50 ROT Y37	; Y 축을 중심으로 37° 를 회전하는 프레임 정의
N60 X0 Y0 Z0 F10000	; 시작 위치로 접근
N70 G42 CUT2DF TCOFR TCARR=1 T1 D1 X10	; 회전된 프레임에서 반경 보정 , 공구 길이를 설정한 뒤 공구 홀더 1, 공구 1 을 선택
N80 X40	; 37° 회전 하에서 가공 실행
N90 Y40	
N100 X0	
N110 Y0	
N120 M30	

그 밖의 정보

결정된 구조

결정된 구조를 사용하는 머신 (회전 가능한 공작물뿐 아니라 공구도 포함) 에 있어서 , 시스템 변수는 다음을 포함하여 $TC_CARR18[m] 에서 최대 $TC_CARR23[m] 까지 항목에 의해 확장됩니다 .

다음으로 구성된 회전 가능한 공구 테이블

- 두 번째 로타리 축 V_2 와 회전 가능한 세 번째 로타리 축의 I_4 인 공구 테이블의 원점 사이의 벡터 공차

다음으로 구성된 로타리 축

- 로타리 축 V_1 및 V_2 의 참조에 관한 2 개 채널 식별자로 , 필요한 경우 전향적이 될 수 있는 공구 홀더의 방향을 결정하기 위해 그 위치로 액세스할 수 있음

T, P, M 값 중 하나를 사용하는 운동학적 유형

- 운동학적 유형 T: 공구만 회전 가능
- 운동학적 유형 P: 부품만 회전 가능
- 운동학적 유형 M: 공구 및 부품 회전 가능

공구 홀더 데이터 공차

공구 홀더 데이터 세트의 모든 데이터는 $TC_CARR1[0]=0 를 사용하여 삭제할 수 있습니다 .

운동학적 유형 $TC_CARR23[T]=T 는 3 개의 사용 가능한 대문자 또는 소문자 (T, P, M) 중 하나로 구성되며 , 이런 이유로 삭제되어서는 안 됩니다 .

공구 홀더 데이터 변경

설명된 값은 각각 부품 프로그램에 새로운 값을 할당하여 수정될 수 있습니다 . T, P, M 이외의 다른 문자를 사용하면 전향적이 될 수 있는 공구 홀더 활성화 시도를 할 때 , 알람을 발생시킵니다 .

공구 홀더 데이터 읽기

설명된 값은 각각의 값을 부품 프로그램에 있는 변수에 할당하여 읽혀질 수 있습니다 .

미세조정 옵셋

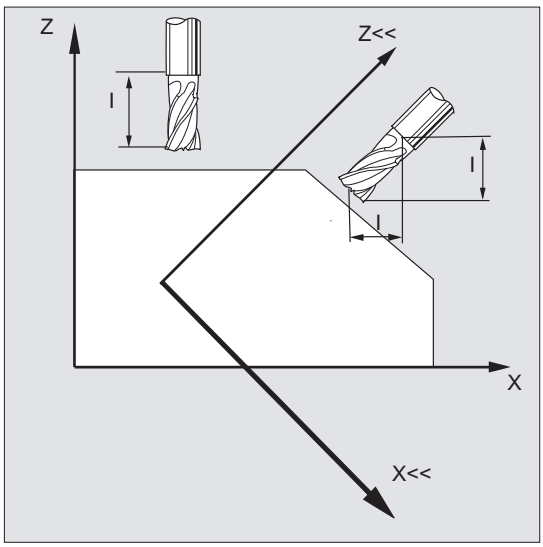
허용되지 않는 미세조정 옵셋값은 전향적이 될 수 있는 공구 홀더가 활성화된 경우에만 감지됩니다 . 이 공구 홀더에는 이런 값이 포함되어 있으며 , 동시에 셋팅 데이터는 SD42974 $SC_TOCARR_FINE_CORRECTION = TRUE 가 됩니다 .

최대 허용 가능한 미세조정 옵셋은 머신 데이터 내의 허용 가능한 값으로 제한됩니다 .

7.9 방향 조정 가능한 공구 홀더에 대한 공구 길이 보정 (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ)

기능

공구의 공간적인 오리엔테이션이 변경된 경우 , 이 공구의 길이 보정 역시 변경됩니다 .



수동 셋팅 또는 고정된 공간 오리엔테이션을 사용하는 공구 홀더를 변경하는 등의 방법으로 리셋을 하면 공구 길이 보정 역시 다시 결정되어야만 합니다. 이런 사항은 TCOABS 및 TCOFR 경로 명령을 사용하여 수행됩니다 .

전향적이 될 수 있는 활성화된 프레임의 공구 홀더에 대해, TCOFRZ, TCOFRY 및 TCOFRX를 사용하는 공구를 선택하면 공구가 향해야 하는 방향을 결정할 수 있게 됩니다 .

구문

```
TCARR=[<m>]
TCOABS
TCOFR
TCOFRZ
TCOFRY
TCOFRX
```

의미

TCARR=[<m>]:	"m" 숫자를 사용하여 공구 홀더 요청
TCOABS:	현재 공구 홀더의 오리엔테이션에서 공구 길이 요소 결정
TCOFR:	활성화된 프레임의 오리엔테이션에서 공구 길이 요소 결정
TCOFRZ:	Z 방향을 가리키는 공구를 사용하는 활성화된 프레임으로부터 전향적 공구 홀더
TCOFRY:	Y 방향을 가리키는 공구를 사용하는 활성화된 프레임으로부터 전향적 공구 홀더
TCOFRX:	X 방향을 가리키는 공구를 사용하는 활성화된 프레임으로부터 전향적 공구 홀더

그 밖의 정보

공구 홀더의 방향에서 공구 길이 보정 결정 (TCOABS)

TCOABS 는 시스템 변수 $TC_CARR13 및 $TC_CARR14 에 저장되어 있는 공구 홀더의 현재 방향 각도에서 공구 길이 보정을 계산합니다 .

시스템 변수를 사용한 공구 홀더 운동학에 대한 설명은 " 공구 홀더 구조 [쪽 437]" 을 참조하십시오 .

프레임이 변경되었을 때 공구 길이 보정을 새로 계산하도록 하기 위해서는 공구를 다시 선택해야 합니다 .

활성화된 프레임으로부터 공구 방향

방향 능력이 있는 공구 홀더는 설정되어 있기 때문에 공구는 다음과 같은 방향을 가리키게 됩니다 .

- Z 방향에서 TCOFR 또는 TCOFRZ
- Y 방향에서 TCOFRY
- X 방향에서 TCOFRX

TCOFR 에서 TCOABS 으로 바뀌게 되면 공구 길이 보정이 다시 계산됩니다 .

공구 홀더 요청 (TCARR)

TCARR 을 사용하면 형상 데이터 (보정 메모리) 와 함께 공구 홀더 번호 m 이 요청됩니다 .

m=0 을 사용하면 활성화된 공구 홀더는 선택 취소됩니다 .

공구 홀더의 형상 데이터는 공구를 호출한 뒤에만 활성화됩니다 . 선택된 공구는 공구 홀더 변경이 수행된 이후에도 활성화된 상태로 남습니다 .

현재 공구 홀더에 대한 형상 데이터 역시 해당하는 시스템 변수를 통해 부품 프로그램에서 정의된 것입니다 .

프레임 변경을 위해 공구 길이 보정 재계산 (TCOABS)

프레임이 변경되었을 때 공구 길이 보정을 새로 계산하도록 하기 위해서는 공구를 다시 선택해야 합니다 .

주

공구 방향은 활성화된 프레임에 맞게 수동으로 조정되어야만 합니다 .

공구 길이 보정이 계산된 경우 , 공구 홀더의 회전 각도는 중간 단계에서 계산됩니다 . 2 개의 로타리 축을 사용하는 공구 홀더를 사용하면 보통 2개 세트의 회전 각도가 생기게 됩니다. 이는 공구 방향을 활성화된 프레임에 맞게 조정하는 데 사용됩니다 . 그러므로 , 회전 각도 값은 시스템 변수에 저장되는데 , 이 때 기계적으로 설정된 회전 각도와 비슷하게 맞아야 합니다 .

주

공구 방향

프레임 방향을 사용하여 계산한 회전 각도를 기계에서 설정 가능한지를 확인하도록 제어하는 것은 불가능합니다 .

도달할 수 없는 프레임 방향을 사용하여 공구 방향을 계산하여 이를 공구 홀더의 로타리 축의 방향을 정하는 데 사용하게 되면 알람이 출력됩니다 .

공구 정밀도 보정 및 움직일 수 있는 공구 홀더의 공구 길이 보정에 대한 기능을 합치는 것은 허용되지 않았습니다 . 이 두 가지 기능을 동시에 호출하게 되면 오류 메시지가 발생됩니다 .

TOFRAME 기능은 프레임이 선택한 공구 홀더의 방향에 따라 정의되도록 허용해 줍니다 . 자세한 내용은 " 프레임 " 장을 참조하십시오 .

방향 변환이 활성화된 경우 (3, 4, 또는 5 축 동시보간), 알람을 발생시키지 않으면서 제로 위치에서 벗어난 방향을 사용하는 공구 홀더를 선택할 수 있습니다 .

표준 및 측정 싸이클에서 파라미터 전송

표준 및 측정 싸이클의 파라미터 전송에 대해서는 다음에 지정되어 있는 값 범위를 적용하십시오 .

각도값에 대해서는 다음에 따라 그 값 범위가 달라집니다 .

- 1 차 기하축을 중심으로 회전 : - 180 도 + 180 도
- 2 차 기하축을 중심으로 회전 : - 90 도 + 90 도
- 3 차 기하축을 중심으로 회전 : - 180 도 + 180 도

프레임 장의 " 프로그램 가능한 회전 (ROT, AROT, RPL)" 을 참조하십시오 .

주

각도값을 표준 또는 측정 싸이클로 전송할 때는 다음을 주의 깊게 관찰하여야 합니다 .

NC 의 계산 결과보다 작은 값은 반올림하여 0 이 되어야 합니다 !

각도 위치에 대한 NC 의 계산 결과는 머신 데이터에 정의되어 있습니다 .

MD10210 $MN_INT_INCR_PER_DEG

7.10 온라인 공구 길이 보정 (TOFFON, TOFFOF)

기능

시스템 변수 $AA_TOFF[<n>] 를 사용하면 실시간으로 3 개의 공구 방향에 따른 유효 공구 길이를 3 차원으로 오버레이할 수 있습니다 .

3 개의 기하 축 이름이 색인 <n> 으로 사용됩니다 . 이렇게 하면 동시에 기하축 활성화에 의한 보정의 활성화된 방향의 번호도 정의됩니다 .

모든 보상은 동시에 활성화될 수 있습니다 .

온라인 공구 길이 보정 기능은 다음에 사용될 수 있습니다 .

- 방향 오리엔테이션 TRAORI
- 전향적 공구 홀더 TCARR

주

온라인 공구 길이 보정은 **옵션**이며 , 사전에 활성화되어 있어야만 합니다 . 이 기능은 오리엔테이션 변환 또는 전향적 공구 홀더의 활성화와 연결되어 실행되어야만 합니다 .

구문

```
TRAORI
TOFFON(< 보정 방향 >[,< 옵셋 값 >])
WHEN TRUE DO $AA_TOFF[< 보정 방향 >]                    ; 동기 동작인 경우
...
TOFFOF(< 보정 방향 >)
```

동기화 동작에서 온라인 공구 길이 보정 프로그래밍에 관한 자세한 정보는 ' 온라인 공구 길이 보정 ($AA_TOFF) [쪽 594]' 을 참조하십시오 .

의미

TOFFON: 온라인 공구 길이 보정 **활성화**

< 보정 방향 >: 온라인 공구 길이 보정이 실행되는 공구 방향 (X, Y, Z)

< 옵셋 값 >: 이를 활성화할 때 , 옵셋 값은 보정에 상응하는 방향으로 지정될 수 있으며 , 이는 곧바로 복구됩니다 .

TOFFOF: 온라인 공구 길이 보정 **리셋**

지정된 보정 방향의 보정 값이 리셋되고 전저리 정지가 시작됩니다 .

예제

예제 1: 공구 길이 보정 선택

프로그램 코드	설명
MD21190 $MC_TOFF_MODE = 1 MD21194 $MC_TOFF_VELO[0] =1000 MD21196 $MC_TOFF_VELO[1] =1000 MD21194 $MC_TOFF_VELO[2] =1000 MD21196 $MC_TOFF_ACCEL[0] =1 MD21196 $MC_TOFF_ACCEL[1] =1 MD21196 $MC_TOFF_ACCEL[2] =1	; 절대값으로 접근
N5 DEF REAL XOFFSET	
N10 TRAORI(1)	; 변환 활성화
N20 TOFFON(Z)	; Z 공구 방향에 대한 온라인 공구 길이 보정 활성화
N30 WHEN TRUE DO $AA_TOFF[Z]=10 G4 F5	; Z 공구 방향에 대해 10 의 TLC 가 보간됩니다 .
...	
N100 XOFFSET=$AA_TOFF_VAL[X]	; X 방향으로 실제 보정 할당
N120 TOFFON(X,-XOFFSET) G4 F5	; X 공구 방향에 대해 TLC 가 0 으로 다시 감소합니다 .

예제 2: 공구 길이 보정 선택 취소

프로그램 코드	설명
N10 TRAORI(1)	; 변환 활성화
N20 TOFFON(X)	; X 공구 방향에 대한 온라인 공구 길이 보정 활성화
N30 WHEN TRUE DO $AA_TOFF[X] = 10 G4 F5	; X 공구 방향에 대해 10 의 TLC 가 보간됩니다 .
...	
N80 TOFFOF(X)	; X 공구 방향의 위치 옵셋이 삭제됩니다 ...$AA_TOFF[X]=0 축이 이동하지 않습니다 . 실제 오리엔테이션에 상응하는 워크의 실제 위치에 위치 옵셋이 추가됩니다 .

추가 정보

블록 준비

사전 처리에서 블록 준비를 하는 중 , 현재 기본 실행에서 공구 길이 옵셋 활성화 역시 고려 사항에 들어갑니다 . 최대 허용 축 속도를 최대한 활용할 수 있도록 하려면 공구 옵셋이 설정되는 동안 STOPRE 전처리 정지를 사용하여 블록 준비를 중지시켜야 합니다 .

공구 옵셋은 IPO 버퍼가 런인과 기본 실행 사이에 적응될 수 있는 것보다는 프로그램 시작 이후 공구 길이 옵셋이 변경되지 않았거나 공구 길이 옵셋이 변경된 이후 더 많은 블록이 처리되는 런인 시간에 항상 알려집니다 .

Variable $AA_TOFF_PREP_DIFF

현재 보간에 적용된 보정과 블록 준비 중에 적용된 보정 간의 치수 차이는 변수 $AA_TOFF_PREP_DIFF[<n>] 에 폴링할 수 있습니다 .

머신 데이터 및 셋팅 데이터 조절

다음 시스템 데이터는 온라인 공구 길이 옵셋에 사용할 수 있습니다 .

- MD20610 $MC_ADD_MOVE_ACCEL_RESERVE (오버레이 동작에 대한 가속 마진)
- MD21190 $MC_TOFF_MODE

  시스템 변수 $AA_TOFF[<n>] 의 내용이 절대치로 옮겨지거나 통합됩니다 .
- MD21194 $MC_TOFF_VELO (온라인 공구 길이 보정의 속도)
- MD21196 $MC_TOFF_ACCEL (온라인 공구 길이 보정의 가속)
- 사전 설정된 한계값의 셋팅 데이터 :
  SD42970 $SC_TOFF_LIMIT (공구 길이 보정 값의 상한값)

참고 자료 :
기능 매뉴얼 , 특수 기능 , F2: 다중 축 변환

7.11 회전 가능한 공구에 대한 절삭 데이터 수정 (CUTMOD)

기능

" 공구에 대한 절삭 데이터 수정 " 기능을 사용하면 , 공구를 회전시킬 때 가공되는 공작물과 관련하여 얻어지는 변경된 기하학적 관계 (주요 선삭 공구뿐 아니라 드릴링 공구 및 밀링 공구) 는 공구 보정과 관련하여 고려될 수 있습니다 .

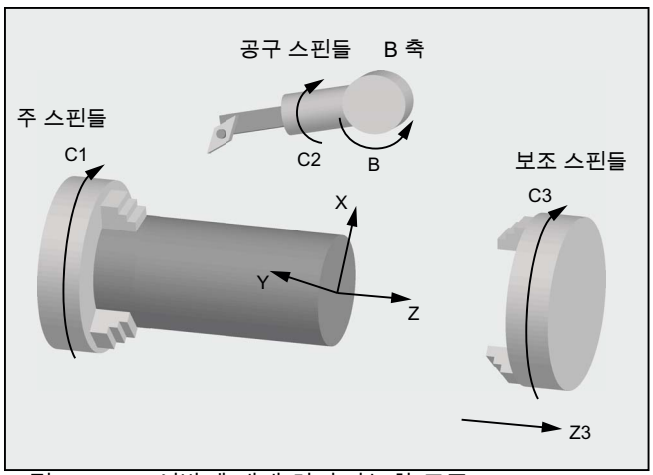


그림 7-1 선반에 대해 회전 가능한 공구

실제 공구 회전은 항상 전향적이 될 수 있는 현재 활성화된 공구 홀더에 의해 결정됩니다("방향 조정 가능한 공구 홀더에 대한 공구 길이 보정 (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) [쪽 443]" 참조).

이 기능은 _movePathCircular() 명령을 사용하여 활성화됩니다 .

구문

```
CUTMOD=<value>
```

의미

CUTMOD " 회전 가능한 공구에 대한 절삭 데이터 수정 " 기능을 켜는 명령

<value> CUTMOD 명령에 할당 가능한 값 :

0 함수가 중지됩니다 .
시스템 변수 $P_AD... 에서 공급된 값은 상응하는 공구 파라미터와 동일합니다 .

> 0 이 함수는 지정된 번호를 사용하여 공구 홀더가 전향적이 될 수 있는 경우에 활성화됩니다 . 즉 , 활성화는 전향적이 될 수 있는 특정 공구 홀더와 연결되어 있습니다 .
시스템 변수 $P_AD... 에서 공급되는 값은 활성화된 회전에 의해 정해지는 해당 공구 파라미터에 의해 수정될 수 있습니다 .
일시적으로 전향적이 될 수 있는 지정된 공구 홀더의 비활성화는 함수를 비활성화시킵니다 . 영구적으로 전향적이 될 수 있는 다른 공구 홀더의 활성화는 역시 이 함수를 비활성화시킵니다 . 우선 첫 번째의 경우, 함수는 전향적이 될 수 있는 동일한 공구 홀더를 다시 선택했을 때 다시 활성화됩니다 . 두 번째의 경우 , 그 이후에라도 새로 선택해야만 합니다. 전향적이 될 수 있는 공구 홀더는 지정된 번호를 사용하여 다시 활성화됩니다 .
이 함수는 리셋의 영향을 받지 않습니다 .

-1 전향적이 될 수 있는 공구 홀더가 활성화되어 있는 경우 , 함수는 항상 활성화됩니다 .
공구 홀더를 변경하거나 이를 선택 취소하고 난 후 새로 선택하면 CUTMOD 를 다시 설정하지 않아도 됩니다 .

-2 이 함수는 전향적이 될 수 있는 공구 홀더가 활성화되어 있고, 그 번호가 전향적이 될 수 있는 현재 활성화된 공구 홀더와 같은 경우에는 항상 활성화됩니다 .
전향적이 될 수 있는 공구 홀더가 활성화되지 않은 경우, CUTMOD=0와 동일한 의미를 갖게 됩니다. 전향적이 될 수 있는 공구 홀더가 활성화된 경우 , 실제 공구 홀더 번호를 직접 지정했을 때와 동일한 의미를 갖게 됩니다 .

< -2 -2 보다 작은 값은 무시됩니다 . 즉 , 이 경우는 CUTMOD 가 프로그래밍되지 않은 것처럼 취급됩니다 .
주 :
이 값의 범위가 사용 가능한 이후의 확장을 위해 남겨진 것처럼 사용되어서는 안 됩니다 .

주

SD42984 $SC_CUTDIRMOD

CUTMOD 명령을 사용해 활성화시킬 수 있는 함수는 셋팅 데이터인 SD42984 $SC_CUTDIRMOD 를 사용하여 활성화될 수 있는 함수를 대체합니다 . 하지만 이 함수는 변경되지 않은 채로 사용할 수 있게 남아 있습니다 . 그렇다고 해서 두 개의 함수가 병렬로 사용되는 것이 말이 안되는 것처럼 이 함수 역시 CUTMOD 가 0 일 경우에만 활성화될 수 있습니다 .

예제

다음 예제는 공구 노즈 위치가 3 이고 공구를 B 축으로 회전시키는 전향적이 될 수 있는 공구 홀더를 사용하는 공구를 나타냅니다 .

기계에서 블록의 끝 위치를 지정하는 비고의 숫자값은 X, Y, Z 순서에서 동일 (MCS) 합니다 .

프로그램 코드	설명			
N10 $TC_DP1[1,1]=500				
N20 $TC_DP2[1,1]=3	;	공구 노즈 위치		
N30 $TC_DP3[1,1]=12				
N40 $TC_DP4[1,1]=1				
N50 $TC_DP6[1,1]=6				
N60 $TC_DP10[1,1]=110	;	홀더 각		
N70 $TC_DP11[1,1]=3	;	절삭 방향		
N80 $TC_DP24[1,1]=25	;	공차 각도		
N90 $TC_CARR7[2]=0 $TC_CARR8[2]=1 $TC_CARR9[2]=0	;	B 축		
N100 $TC_CARR10[2]=0 $TC_CARR11[2]=0 $TC_CARR12[2]=1	;	C 축		
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				

설명 :

N180 블록에서 , 우선 공구는 CUTMOD=0 에 대해 선택되고 , 비회전 공구 홀더는 전향적이 될 수 있습니다 . 전향적이 될 수 있는 공구 홀더의 모든 옵셋 벡터는 0 이며 , $TC_DP3[1,1] 및 $TC_DP4[1,1] 에서 지정된 공구 길이에 해당하는 위치는 접근합니다 .

B 축을 중심으로 30° 회전하는 전향적이 될 수 있는 공구 홀더는 N200 블록에서 활성화됩니다. 공구 노즈 위치가 CUTMOD=0 로 인해 수정되지 않기 때문에 예전의 공구 노즈 원점은 이전과 마찬가지로 중요합니다. N210 블록에서 위치는 예전 공구 노즈 원점을 0으로 유지하는 위치로 접근하기 때문입니다 (즉 , 벡터 (1, 12) 가 Z/X 평면에서 30° 회전).

N260 블록에서 블록에 반대되는 N200 CUTMOD=2 가 적용됩니다 . 전향적이 될 수 있는 공구 홀더를 회전시킨 결과 , 수정된 공구 노즈 위치는 8 이 됩니다 . 이 결과는 다른 축 위치에서도 마찬가지입니다 .

공구 반경 보정(TRC)는 N220 및/또는 N270 블록에서 활성화됩니다. 양쪽 프로그램 섹션에서 서로 다른 공구 노즈 위치는 TRC 가 활성화되어 있는 블록의 끝 위치에 영향을 주지 않습니다. 그러므로 해당 위치는 동일합니다 . 서로 다른 공구 노즈 위치는 N260 및 / 또는 N300 블록을 다시 선택 취소했을 때만 적용됩니다 .

추가 정보

수정된 절삭 데이터의 효율성

수정된 공구 노즈 위치와 수정된 공구 노즈 원점은 공구가 이미 활성화된 경우에도 프로그래밍을 하면 즉시 적용됩니다 . 이를 위해 공구를 다시 선택할 필요는 없습니다 .

활성화된 가공 평면의 영향

수정된 공구 노즈 위치 , 절삭 방향 및 홀더나 공차각을 결정하기 위해서는 활성화된 평면 (G17 - G19) 에서 절삭날의 평가가 중요합니다 .

하지만 셋팅 데이터 SD42940 $SC_TOOL_LENGTH_CONST (평면을 선택했을 때 공구 길이 콤포넌트를 변경), 유효한 값이 0 이 아닌 경우 (플러스 또는 마이너스 17, 18 또는 19), 그 콘텐트는 관련 수량이 평가된 평면을 정의합니다 .

시스템 변수

다음과 같은 시스템 변수를 사용할 수 있습니다 .

시스템 변수	중요도
$P_CUTMOD_ANG / $AC_CUTMOD_ANG	활성화된 가공 평면에 (라운딩이 안 된) 각도를 공급하십시오 . 이 각도는 CUTMOD 및/또는 $SC_CUTDIRMOD를 사용하여 활성화된 함수에 대한 절삭 데이터(공구 노즈 위치, 절삭 방향, 공차 각 및 홀더각)의 수정을 위한 기초로 사용되었습니다 . $P_CUTMOD_ANG 는 전처리 과정에서 실제 상태를 기준으로 하며 , $AC_CUTMOD_ANG 는 실제 기본 실행 블록을 기준으로 합니다 .
$P_CUTMOD / $AC_CUTMOD	CUTMOD 명령을 사용하여 마지막으로 프로그램되어 현재 유효한 값을 읽으십시오 (공구 홀더의 번호는 절삭 데이터 수정에 사용하기 위해 활성화되어야 합니다). 마지막으로 프로그램된 CUTMOD 값 = -2 (전향화될 수 있는 현재 활성화된 공구 홀더와 함께 활성화) 인 경우 , -2 값은 $P_CUTMOD 에서 반환되지 않지만 , 프로그래밍을 할 때 전향화될 수 있는 활성화된 공구 홀더의 번호는 반환됩니다 . $P_CUTMOD 는 전처리 과정에서 실제 상태를 기준으로 하며 , $AC_CUTMOD 는 실제 기본 실행 블록을 기준으로 합니다 .
$P_CUT_INV / $AC_CUT_INV	공구가 회전하여 회전의 스핀들 방향이 전도되어야 하는 경우에는 TRUE 값을 공급하십시오 . 이를 위해서는 읽기 작업에서 참조해야 하는 블록이 다음의 4 가지 조건을 충족해야 합니다 . 1. 선삭 공구 또는 연삭 공구가 활성화된 경우 (공구 유형 400 에서 599 및 / 또는 SD42950 $SC_TOOL_LENGTH_TYPE = 2). 2. 절삭 영향은 CUTMOD 언어 명령을 사용하여 활성화되어야 합니다 . 3. 전향화될 수 있는 공구 홀더가 활성화됩니다 . 이 공구 홀더는 CUTMOD 의 숫자 값을 사용하도록 지정되어 있습니다 . 4. 오리엔테이션 가능한 공구 홀더는 공구를 가공 평면에 있는 축 (보통 C 축) 을 중심으로 회전시키기 때문에 그 결과 공구 절삭날의 수직면은 처음 위치 대비 90° 이상의 각도 (보통 180°) 로 회전됩니다 . 지정된 4 개의 조건 중 하나라도 충족되지 않으면 변수의 콘텐트는 FALSE 가 됩니다 . 공구 노즈 위치가 정의되지 않은 공구의 변수 값은 항상 FALSE 입니다 . $P_CUT_INV 는 전처리 과정에서 실제 상태를 기준으로 하며 , $AC_CUT_INV 는 실제 기본 실행 블록을 기준으로 합니다 .

전체 메인 실행 변수 ($AC_CUTMOD_ANG, $AC_CUTMOD 및 $AC_CUT_INV) 는 동기 동작에서 사용할 수 있습니다 . 사전 처리 작업에서 읽기 액세스 작업은 전처리 정지를 생성합니다 .

수정된 절삭 데이터 :

공구 회전이 활성화된 경우 , 수정된 데이터는 다음 시스템 변수에서 사용 가능합니다 .

시스템 변수	중요도
$P_AD[2]	공구 노즈 위치
$P_AD[10]	홀더 각
$P_AD[11]	절삭 방향
$P_AD[24]	공차 각도

주

CUTMOD 명령을 사용하여 " 공구에 대한 절삭 데이터 수정 " 함수가 활성화되고 , 회전을 유발하는 전향적이 될 수 있는 공구 홀더 데이터가 활성화된 경우 , 데이터는 항상 공구 파라미터 ($TC_DP2[..., ...] 등) 에 상응하여 수정됩니다 .

참고 자료

" 공구에 대한 절삭 데이터 수정 " 기능에 관한 자세한 정보는 다음을 참조하십시오 .

기능 매뉴얼 기본 기능 ; 공구 옵셋 (W1)

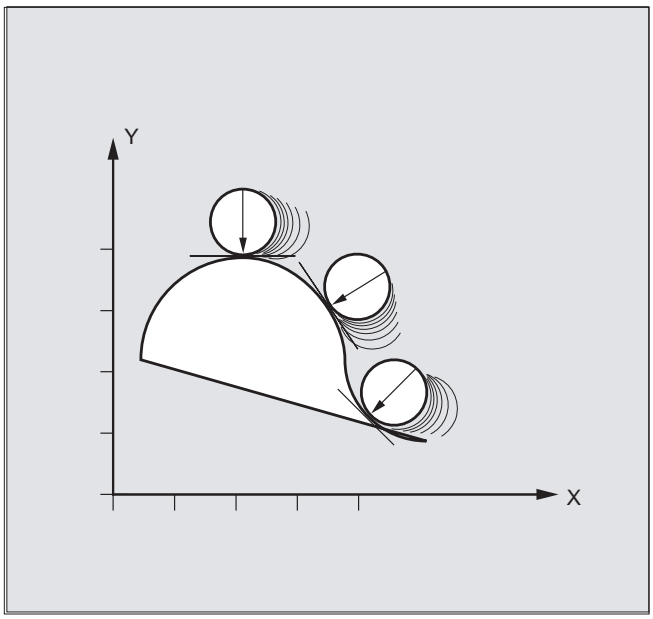
경로 이송 동작

8

8.1 접선 제어 (TANG, TANGON, TANGOF, TLIFT, TANGDEL)

기능

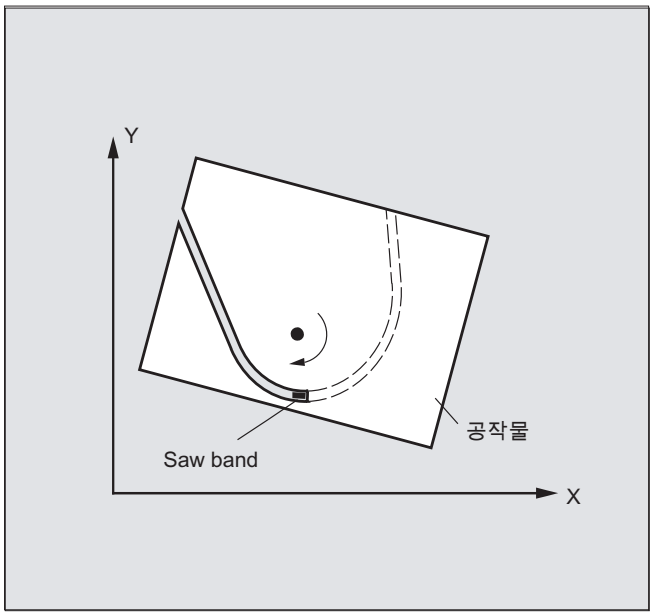
추종 축은 접선을 따라 리딩 축의 경로를 따릅니다 . 그러면 공구의 정렬이 형상과 평행을 이루게 됩니다 . TANGON 명령에 프로그램되어 있는 각을 사용하여 , 공구를 접선과 관련하여 위치되도록 할 수 있습니다 .



적용

접선 제어는 예를 들어 다음과 같은 작업에서 사용할 수 있습니다 .

- 니블링을 하는 동안 회전 가능한 공구의 접선 포지셔닝
- 띠톱에 대한 공작물 정렬 추적 (아래 다이어그램 참조)
- 연삭 휠에서 드레싱 공구의 포지셔닝
- 유리 또는 종이 가공에 대한 연삭 휠의 포지셔닝
- 5- 축 용접을 위한 와이어 접선 방향 공급 .



구문

접선 추적 정의 :
TANG(<Faxis>,<Laxis1>,<Laxis2>,< 커플링 팩터 >,<KS>,<Opt>)

접선 제어 활성화 :
TANGON(<Faxis>,< 각도 >,<Dist>,< 각도 공차 >)

접선 제어 비활성화 :
TANGOF (<Faxis>)

" 형상 코너에서 중간 블록 삽입 " 기능 활성화 :
TLIFT(<Faxis>)

TLIFT 연산은 TANG(...) 으로 축을 지정한 후에 선택합니다 .

" 형상 코너에서 중간 블록 삽입 " 기능 비활성화 :

후속 TLIFT(<Faxis>) 없이 TANG(...) 연산을 반복하십시오 .

접선 추적 정의 삭제 :
TANGDEL(<Faxis>)

동일한 추종 축을 사용하는 새로운 접선 추적을 TANG 호출 준비에 정의하면 기존의 사용자 정의 접선 추적은 삭제해야 합니다 . 접선 추적 삭제는 TANGOF(<Faxis>) 를 사용하는 커플링이 비활성화될 때만 가능합니다 .

의미

TANG:	접선 추적 정의를 위한 준비 작업 :
TANGON:	지정된 추종 축에 대한 접선 제어 활성화
TANGOF:	지정된 추종 축에 대한 접선 제어 비활성화
TLIFT:	" 형상 코너에서 중간 블록 삽입 " 기능 활성화
TANGDEL:	접선 추적 정의 삭제
<Faxis>:	추종 축 : 로터리 축을 따르는 추가적인 접선 .
<Laxis1>,<Laxis2>:	리딩 축 : 추종 축의 접선을 결정하는 경로 축입니다 .
< 커플링 팩터 >:	커플링 팩터 : 접선의 각도 변경과 추적 대상 축 간의 관계 디폴트 : 1 **참고 :** 1 의 커플링 팩터는 직접 프로그래밍하지 않아도 됩니다 .
<CS>:	좌표계 식별 문자 "B": 기본 좌표계 (디폴트) **참고 :** <CS> = "B" 는 직접 프로그래밍할 필요가 없습니다 . "W": 공작물 좌표계 (사용 안함)
<Opt>:	최적화 "S": 표준 (디폴트) **참고 :** <Opt> = "S" 는 직접 프로그래밍할 필요가 없습니다 . "P": 접선축 및 형상의 특성 오버타임의 자동 조정 **참고 :** <Opt> = "P" 로 설정하면 추종 축의 다이나믹 속성이 리딩 축의 속도 제한에 적용됩니다 . 이 설정은 특히 좌표계 변환을 사용할 때 권장되는 설정입니다 .
< 각도 >:	추종 축의 옵셋 각도
<Dist>:	추종 축의 스무딩 경로 (<Opt> = "P" 에 필요)
< 각도 공차 >:	추종 축의 각도 공차 (옵션 , <Opt> = "P" 인 경우에만 평가) **참고 :** 파라미터 <Dist> 및 <각도 공차>는 추적 대상 축과 리딩 축의 접선 간의 차이를 구체적으로 제한합니다 .

예제

예제 1: 접선 추적 정의 및 활성화

프로그램 코드	설명
N10 TANG(C,X,Y,1,"B","P")	; 접선 추적 정의: 로터리 축 C는 기하 축 X와 Y를 추적해야 합니다.
N20 TANGON(C,90)	; C 축은 추종 축입니다 . 경로 축이 움직일 때마다 경로 접선과 90° 각도로 위치가 회전됩니다 .
...	

주

간단한 프로그래밍

TANG(C,X,Y,1,"B","P") 를 간편하게 TANG(C,X,Y,,,"P") 으로 프로그래밍할 수 있습니다 .

예제 2: 평면 변경

프로그램 코드	설명
N10 TANG(A,X,Y,1)	; 1. 접선 추적 정의
N20 TANGON(A)	; 커플링 활성화
N30 X10 Y20	; 반경
...	
N80 TANGOF(A)	; 첫 번째 커플링 비활성화
N90 TANGDEL(A)	; 첫 번째 정의 삭제
...	
TANG(A,X,Z)	; 2. 접선 추적 정의
TANGON(A)	; 새로운 커플링 활성화
...	
N200 M30	

예제 3: 기하 축 전환 및 TANGDEL

생성되는 알람은 없습니다 .

프로그램 코드	설명
N10 GEOAX(2,Y1)	; Y1 은 기하 축 2 입니다 .
N20 TANG(A,X,Y)	; 1. 접선 추적 정의
N30 TANGON(A,90)	; Y1 을 사용하여 추적 활성화
N40 G2 F8000 X0 Y0 I0 J50	
N50 TANGOF(A)	; Y1 을 사용하여 추적 비활성화
N60 TANGDEL(A)	; 첫 번째 정의 삭제
N70 GEOAX(2, Y2)	; Y2 는 새로운 기하 축 2 입니다 .
N80 TANG(A,X,Y)	; 2. 접선 추적 정의
N90 TANGON(A,90)	; Y2 를 사용하여 추적 활성화
...	

예제 4: 자동 최적화를 사용하여 접선 추적

Y1 은 기하 축 2 입니다 .

프로그램 코드	설명
...	
N80 G0 C0	
N100 F=50000	
N110 G1 X1000 Y500	
N120 TRAORI	
N130 G642	; 허용된 최대 경로 편차 스무딩 및 유지 :
N171 TRANS X50 Y50	
N180 TANG(C,X,Y,1,,"P")	; 경로 속도의 자동 최적화를 사용하여 접선 추적 정의
N190 TANGON(C,0,5.0,2.0)	; 자동 최적화를 사용하여 접선 추적 활성화 스무딩 거리 5 mm, 각도 공차 2°
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	

추가 정보

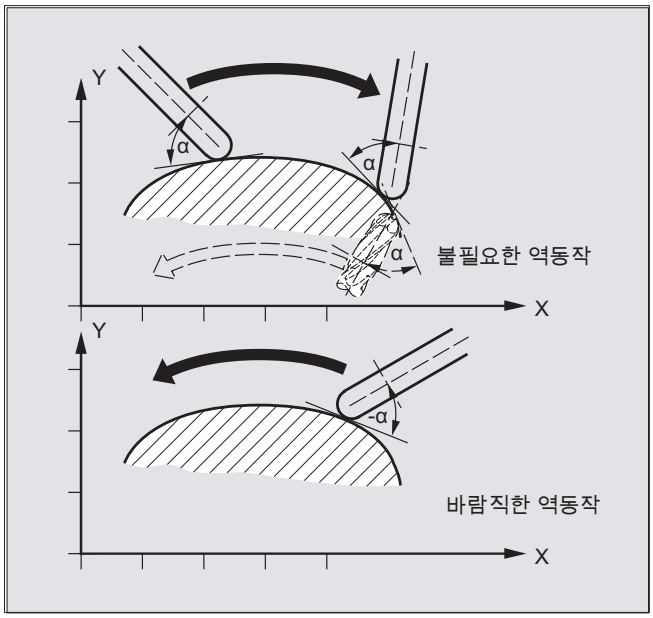
추종 축 및 리딩 축 정의

TANG 은 추종 축 및 리딩 축을 정의하는 데 사용됩니다 .

커플링 팩터는 변하는 접선의 각도와 추종 축 간의 관계를 지정합니다. 그 값은 보통 1(기본)이 됩니다 .

작업 영역 제한을 사용한 제한 각도

앞 뒤로 진동하는 경로 이동에서 , 접선은 추종 축의 변화에 따라 축의 경로 및 오리엔테이션의 전환점에서 180° 로 도약합니다 . 이러한 동작은 일반적으로 부적절합니다 . 복귀는 접근과 동일한 음수 옵셋 각도로 이송되어야 합니다 .



이렇게 하려면 추종 축 (G25, G26) 의 작업 영역을 제한해야 합니다 . 작업 영역 제한은 경로 반전 (WALIMON) 시 활성 상태이어야 합니다 . 옵셋 각도가 작업 영역 제한 밖에 있는 경우 , 음수 옵셋 각도를 사용하여 허용된 작업 영역으로 돌아오도록 시도해야 합니다 .

형상 코너에서 중간 블록 삽입 (TLIFT)

형상의 한 코너에서 접선이 변하여 추종 축의 지령치 위치가 갑자기 바뀝니다 . 축은 보통 가능한 최대 속도로 이 단계 변화를 보정하려고 합니다 . 하지만 이것은 코너 다음의 형상 상의 특정 거리에 대해 필요한 접선 설정으로부터 편차를 초래합니다. 기술적 이유로 이러한 편차를 허용할 수 없는 경우 TLIFT 명령을 사용하여 시스템을 코너에서 강제 정지시키고 자동으로 생성된 중간 블록에서 새 접선 방향으로 추종 축을 회전시킬 수 있습니다 .

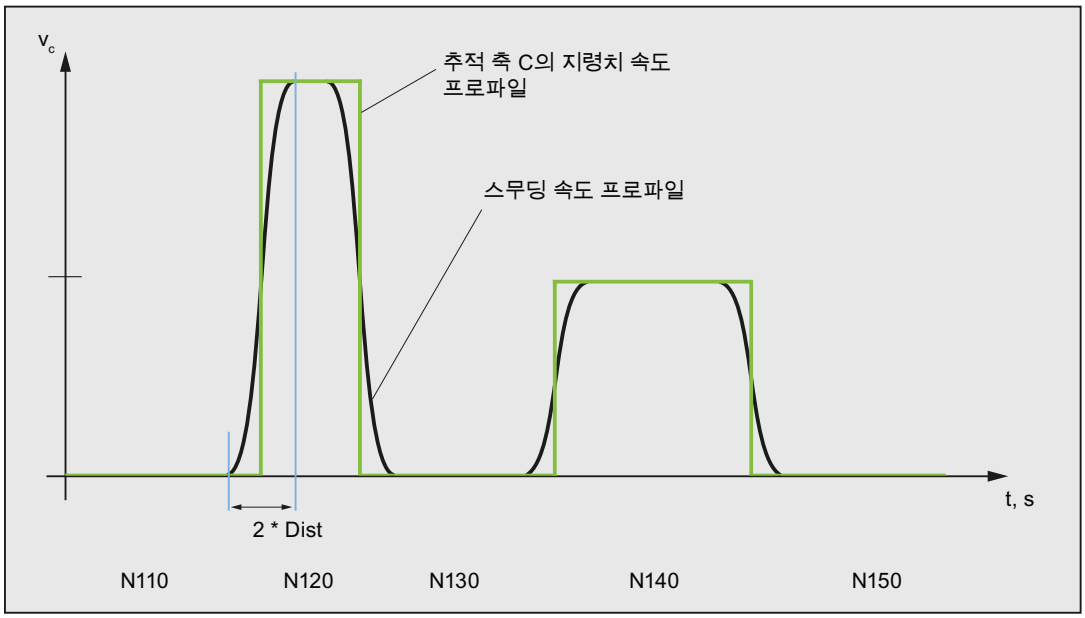
추종 축이 경로 축으로 한 번 사용된 경우 , 경로 축이 회전을 위해 사용됩니다 . 추종 축의 최대 축 속도는 TFGREF[<axis>]=0.001 함수를 사용하여 계산할 수 있습니다 .

추적 축이 이전에 경로 축으로 이송되지 않았었다면 , 이제 포지셔닝 축으로 이송됩니다 . 그 속도는 머신 데이터의 포지셔닝 속도에 따릅니다 .

축은 가능한 최대 속도로 회전됩니다 .

최적화 방법

자동 최적화 (<Opt>="P") 를 선택하고 파라미터 스무딩 거리 (<Dist>) 및 각도 공차 (< 각도 공차>) 를 추종 축에 지정하면 접선 추적 시 리딩 축 형상의 각 단계에 따라 슬레이브 축의 속도 단계가 스무딩됩니다. 이 경우 편차를 최소화하기 위해 추종 축을 선행 제어 (다이어그램 참조) 를 사용하여 제어합니다 .



각도 변경 정의

중간 블록이 자동으로 삽입되는 지점인 각도 변경 한계는 다음 머신 데이터를 사용하여 정의합니다 .

MD37400 $MA_EPS_TLIFT_TANG_STEP (코너 인식을 위한 접선 각도)

변환에 대한 효과

추적 제어가 적용된 로터리 축의 위치는 변환을 위해 입력 값으로 작동할 수 있습니다 .

추종 축의 양의 포지셔닝

리딩 축을 따르는 축이 양의 위치에 있다면 , 이 위치는 프로그램된 옵셋 각도에 추가됩니다 .

모든 거리 데이터 (경로 축 및 포지셔닝 축 모션) 가 허용됩니다 .

커플링 상태

커플링 상태는 NC 가공 프로그램에서 시스템 변수 $AA_COUP_ACT[< 축 >] 을 사용하여 확인할 수 있습니다 .

값	의미
0	커플링 없음
1,2,3	접선 추적 활성

8.2 이송 속도 응답 (FNORM, FLIN, FCUB, FPO)

기능

피드 특성을 유연하게 정의하도록 허용하기 위해
DIN 66205 에 따른 피드 프로그래밍이 직선 및 큐빅 특성에 의해 확장되었습니다 .

큐빅 특성은 직접 프로그램 하거나 보간 스플라인으로 프로그램 할 수 있습니다 . 추가된 특성을 사용하여 가공할 공작물의 굴곡에 따라 연속적으로 완만한 속도 특성을 프로그램할 수 있습니다 .

추가된 특성을 사용하여 가공할 공작물의 굴곡에 따라 연속적으로 완만한 속도 특성을 프로그램할 수 있습니다 .

구문

F… FNORM
F… FLIN
F… FCUB
F=FPO(…,…,…)

의미

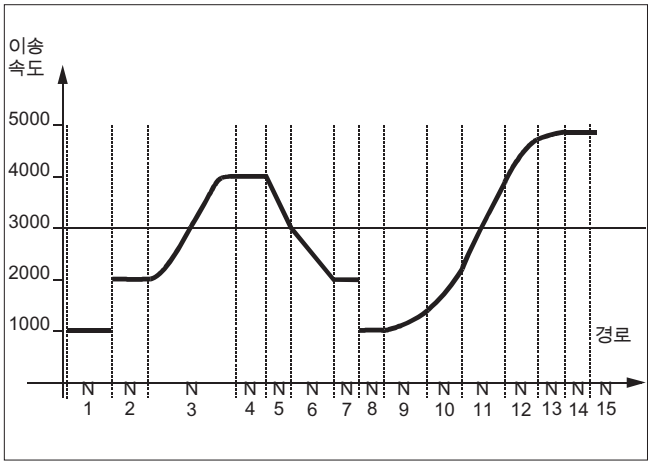
FNORM	기본 셋팅 피드 값은 블록의 이송 경로의 기능에 의해 정의되며 , 그 이후 모달 값으로 유효화됩니다 .
FLIN	**직선 경로 속도 특성 :** 피드 값은 블록 시작부터 블록 끝의 현재 값에서 이송 경로를 통해 직선으로 접근하며 그 이후에는 모달 값으로 유효화됩니다 . G93 및 G94 를 사용하여 반응을 결합할 수 있습니다 .
FCUB	**큐빅** 경로 속도 특성 : 블록 방향으로 프로그램 된 F 값 (블록의 끝과 연관됨) 은 스플라인으로 연결됩니다 . 스플라인은 이전 및 이후 정의된 이송 속도를 사용하여 직선적으로 시작 및 끝나며 , G93 및 G94 를 사용하여 적용됩니다 . 블록에서 F 어드레스가 누락된 경우 , 프로그램 될 마지막 F 값이 사용됩니다 .
F=FPO…	**폴리노미널** 경로 속도 특성 : F 어드레스는 현재 값에서 블록 끝까지 폴리노미널을 통해 피드 특성을 정의합니다 . 종료값은 그 이후 모달 값으로 유효화됩니다 .

곡선화 된 경로 섹션에서 피드 최적화

피드 폴리노미널 F=FPO 및 피드 스플라인 FCUB 은 주속 일정 제어 CFC 에서 이송되므로 , 생성될 저크 없는 (jerk free) 지령치 피드 특성을 허용합니다 . 그러면 연속 가속 지령치 피드 특성이 실행할 수 있습니다 .

예제 : 다양한 피드 특성

이 예제는 다양한 피드 특성의 프로그래밍 및 그래픽 설명을 보여줍니다 .

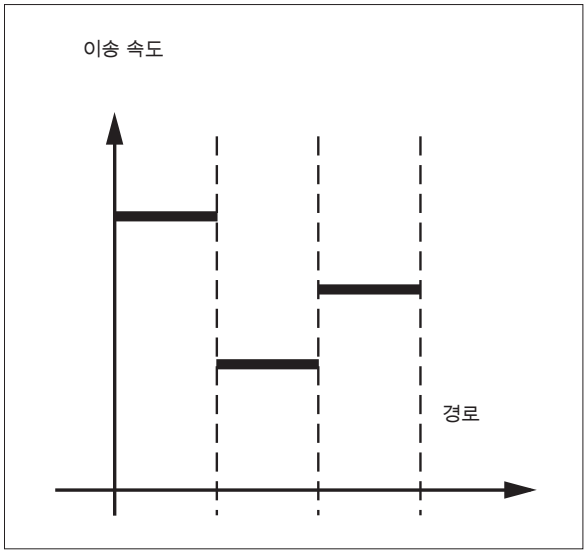


프로그램 코드	설명
N1 F1000 FNORM G1 X8 G91 G64	; 일정속도 특성 , 증분 치수 데이터
N2 F2000 X7	; 지령치 속도 단계 변경
N3 F=FPO(4000, 6000, -4000)	; 블록 끝에서 피드 4000 을 사용한 폴리노미널을 통한 피드 특성
N4 X6	; 폴리노미널 이송 속도 4000 은 모달 값으로 사용됨
N5 F3000 FLIN X5	; 직선 이송 속도 특성
N6 F2000 X8	; 직선 이송 속도 특성
N7 X5	직선식 이송 속도는 모달 값으로 유효화됨
N8 F1000 FNORM X5	; 가속 단계 변경을 사용한 일정속도 특성
N9 F1400 FCUB X8	; 블록에서 프로그램 된 다음의 F 값은 모두 스플라인과 연결되어 있습니다 .
N10 F2200 X6	
N11 F3900 X7	
N12 F4600 X7	
N13 F4900 X5	; 스위치 아웃 스플라인 특성
N14 FNORM X5	
N15 X20	

FNORM

피드 어드레스 F 는 경로 피드는 DIN 66025 에 따른 상수 값으로 정의합니다 .

이 주제에 관해 자세한 사항은 " 기본 " 프로그래밍 메뉴얼을 참조하십시오 .

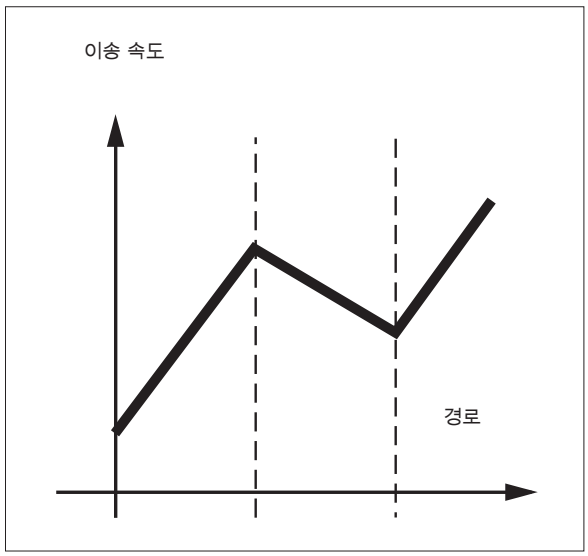


FLIN

피드 특성은 블록이 끝날 때까지 현재 피드 값에서 프로그램된 F 값까지 선형으로 접근합니다 .

예 :

N30 F1400 FLIN X50



FCUB

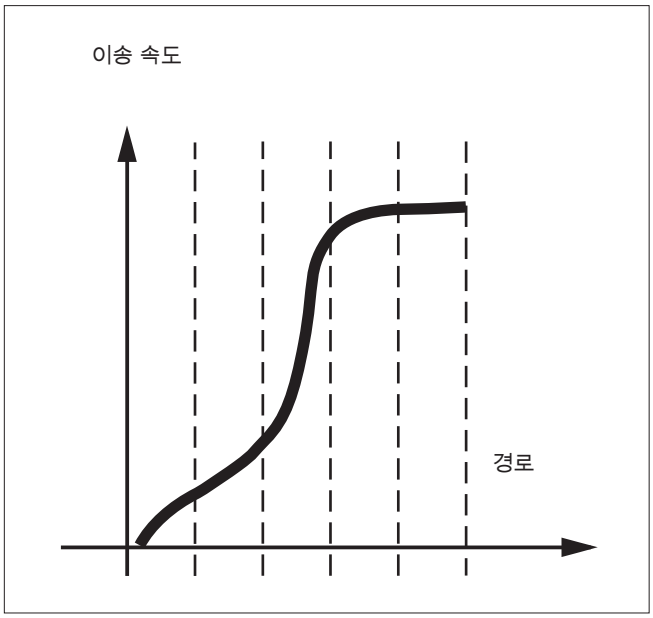
피드는 블록이 끝날 때까지 현재 피드 값에서 프로그램된 F 값까지 큐빅 특성에 따라 접근합니다 . 제어는 활성 FCUB 를 보유하고 있는 비 모달식으로 프로그램된 모든 피드 값에 연결하기 위해 스플린을 사용합니다 . 여기에서 피드 값은 스플린 보간을 계산하기 위한 보간점으로 작동합니다 .

예 :

N50 F1400 FCUB X50

N60 F2000 X47

N70 F3800 X52



F=FPO(…,…,…)

피드 특성은 폴리노미널을 통해 직접 프로그램됩니다 . 폴리노미널 계수는 폴리노미널 보간과 동일한 방법으로 지정됩니다 .

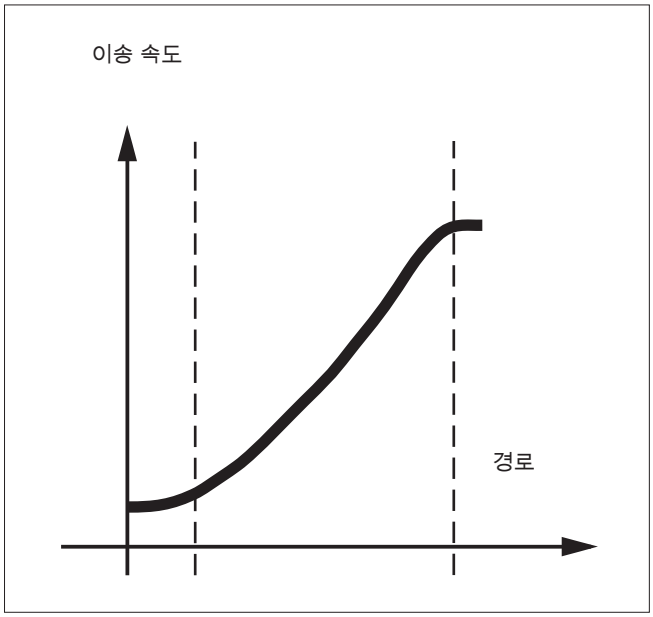
예 :

F=FPO(endfeed, quadf, cubf)

endfeed, quadf 및 cubf 는 사전에 정의된 변수입니다 .

endfeed:	블록 끝의 피드
quadf:	사분원 폴리노미널 계수
cubf:	규빅 폴리노미널 계수

FCUB 가 활성화된 상태에서 , 스플린은 블록 시작과 끝에서 FPO 을 통해 정의된 특성에 직선적으로 연결됩니다 .



제한사항

경로 이송 특성 프로그램밍을 위한 함수는 프로그램된 피드 특성과 상관 없이 적용됩니다 .

프로그램된 피드 특성은 항상 G90 또는 G91 에 상관 없이 절대적입니다 .

피드 응답 FLIN 및 FCUB 활성화는

G93 및 G94 를 사용하여 이루어집니다 .

FLIN 및 FCUB **비활성화는**

G95, G96/G961 및 G97/G971 을 사용하여 이루어집니다 .

활성 콤프레서 COMPON

활성 콤프레서 COMPON 을 사용하여 몇 개의 블록이 결합하여 스플린 분할을 형성할 때 다음이 적용됩니다 .

FNORM:

그룹 내 마지막 블록의 F 워드는 스플린 분할에 적용됩니다 .

FLIN:

그룹 내 마지막 블록의 F 워드는 스플린 분할에 적용됩니다 .
프로그램된 F 값은 분할 마지막까지 적용되며 , 그 이후에는 선형적으로 접근합니다 .

FCUB:

생성된 피드 분할은 머신 데이터 $MC_COMPESS_VELO_TOL 에서 설정된 값을 넘지 않는 양으로 프로그램 된 끝점에서 벗어납니다 .

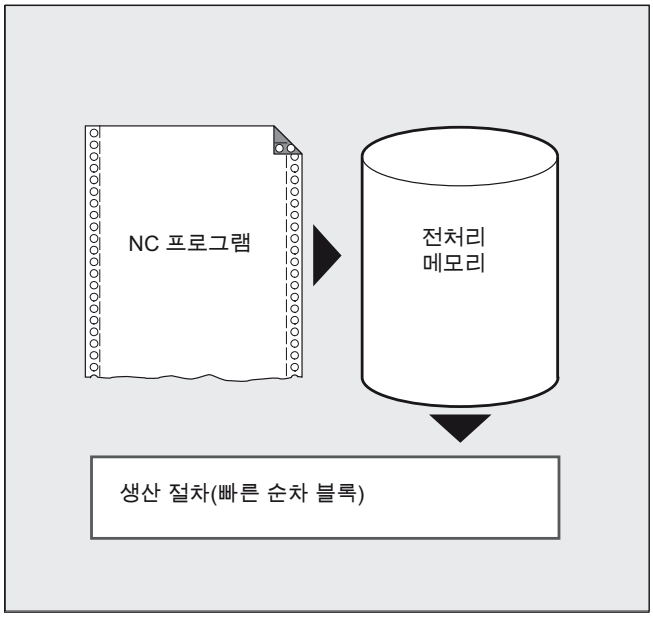
F=FPO(...,...,...)

이 블록은 압축되지 않았습니다 .

8.3 전처리 메모리를 사용한 프로그램 순서 (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE)

기능

제어 시스템은 확장 정도에 따라 특정 수량의 소위 전처리 메모리를 보유하게 됩니다 . 전처리 메모리에는 프로그램 실행 전에 준비된 블록이 저장되며 그 이후 , 가공을 진행하는 동안 고속 순차 블록으로 출력됩니다 . 이런 순서는 짧은 경로가 고속으로 이송될 수 있도록 합니다 . 잔여 제어 시간이 충분하도록 제공되었기 때문에 전처리 메모리는 항상 채워져 있습니다 .



가공 단계 지명

전처리 메모리에 버퍼링될 가공 단계의 시작과 끝은 가공 프로그램에서 각각 STOPFIFO 및 STARTFIFO 를 사용하여 확인할 수 있습니다 . 전처리되어 버퍼링된 블록에 대한 가공은 STARTFIFO 명령 이후 또는 전처리 메모리가 완전히 채워진 경우에 시작됩니다 .

자동 전처리 메모리 제어

자동 전처리 메모리 제어는 FIFOCTRL 명령을 사용하여 호출합니다 . FIFOCTRL 은 초기에는 STOPFIFO 와 거의 동일한 역할을 합니다 . 프로그래밍이 어떻게 되어 있든 , 전처리 메모리가 채워지기 전에는 가공이 시작되지 않습니다 . 그러나 전처리 메모리를 비울 때에 대한 반응은 다릅니다 . FIFOCTRL 의 경우에는 전처리 메모리가 2/3 까지 채워지면 완전히 비워져서 감속되어 정지 상태가 되는 것을 방지하기 위해 경로 속도가 점차 줄어듭니다 .

전처리 정지

블록에 STOPRE 명령을 프로그래밍하면 블록 전처리 및 버퍼링이 중단됩니다 . 다음 블록은 전처리되어 저장된 모든 블록이 완전히 실행되기 전에는 실행되지 않습니다 . 선행 블록은 정위치 정지로 중단됩니다 (예 : G9 사용).

구문

도표 8-1 가공 단계 확인 :

```
STOPFIFO
...
STARTFIFO
```

도표 8-2 자동 전처리 메모리 제어 :

```
...
FIFOCTRL
...
```

도표 8-3 전처리 정지 :

```
...
STOPRE
...
```

주

STOPFIFO, STARTFIFO, FIFOCTRL 및 STOPRE 명령은 개별 블록에 프로그래밍해야 합니다 .

의미

STOPFIFO: STOPFIFO는 전처리 메모리에 버퍼링될 가공 단계의 시작을 나타냅니다. STOPFIFO는 가공을 중단하고 다음의 조건이 될 때까지 전처리 메모리를 채웁니다 .

- STARTFIFO 또는 STOPRE 가 탐지됨

  또는
- 전처리 메모리가 채워짐

  또는
- 프로그램 끝에 도달함

STARTFIFO: STARTFIFO는 가공 단계의 신속한 처리를 시작합니다. 이와 함께 전처리 메모리가 채워집니다 .

FIFOCTRL: 자동 전처리 메모리 제어 활성화

STOPRE: 전처리 정지

주

가공 단계에 버퍼링되지 않은 동작 (기준 검색, 측정 기능 등) 을 요구하는 명령이 가공 단계에 포함된 경우에는 전처리 메모리가 채워지지 않거나 채우기가 중단됩니다 .

주

상태 데이터 ($SA...) 에 대한 액세스가 발생하는 경우 제어 시스템은 내부 전처리 정지를 생성합니다 .

⚠ 주의

공구 옵셋 또는 스플라인 보간이 활성 상태인 경우에는 인접한 블록 순서가 중단되기 때문에 STOPRE 명령을 프로그래밍해서는 안 됩니다 .

예제 : 전처리 정지

프로그램 코드	설명
...	
N30 MEAW=1 G1 F1000 X100 Y100 Z50	; 첫 번째 측정 입력 및 직선 보간에서 프로브를 통한 측정 블록
N40 STOPRE	; 전처리 정지
...	

8.4 조건에 따라 중단 가능한 프로그램 섹션 (DELAYFSTON, DELAYFSTOF)

작동

조건에 따라 중단 가능한 프로그램 섹션은 중단 지연 섹션을 호출합니다 . 특정 프로그램 섹션 내에서 **중단**이 발생해서는 안 되며 **피드**가 변경되어서는 안 됩니다 . 기본적으로 나사 가공과 같이 짧은 프로그램 섹션에서는 거의 모든 정지 이벤트가 발생하지 않도록 보호해야 합니다 . 정지는 프로그램 섹션이 완료될 때까지 적용되지 않습니다 .

구문

DELAYFSTON
DELAYFSTOF

이 명령은 가공 프로그램 라인에서 별도로 프로그램됩니다 .

이 두 명령은 가공 프로그램에서만 허용되지만 동기식으로는 작동하지 않습니다 .

중요도

DELAYFSTON	시작 범위를 정의하십시오 . 이 범위 내에서 " 소프트 " 중단은 지연 정지 구간의 끝까지 지연됩니다 .
DELAYFSTOF	중단 지연 섹션의 끝을 지정합니다 .

주

머신 데이터 MD11550 $MN_STOP_MODE_MASK Bit 0 = 0 (기본값) 에 대해서 지연 정지 구간은 G331/G332 이 활성화되어 있거나 경로 이송 또는 G4 이 프로그램되어 있는 경우에는 암시적으로 정의됩니다 .

예 : 정지 이벤트

지연 정지 구간에서 **이송 속도 및 이송 억제** 변경은 무시됩니다 . 지연 정지 구간 이후에만 적용됩니다 .

정지 이벤트는 다음과 같이 나뉩니다 .

" 소프트 " 정지 이벤트	응답 : 지연됨
" 하드 " 정지 이벤트	응답 : 즉시

짧은 정지라도 유발하는 정지 이벤트의 번호 선택

이벤트 이름	응답	중단 파라미터
RESET	즉시	IS: DB21,… DBX7.7 and DB11, … DBX20.7
PROG_END	알람 16954	NC 프로그램 : M30
INTERRUPT	지연됨	IS: FC-9 및 ASUP DB10, ... DBB1
SINGLEBLOCKSTOP	지연됨	지연 정지 구간의 단일 블록 모드가 활성화됨 : NC 가 지연 정지 구간 외부의 첫 번째 블록 끝에서 정지됩니다 . 단일 블록이 지연 정지 구간 이전에 이미 선택됨 : NST: " 블록 한계에서 NC 정지 " DB21, ... DBX7.2
STOPPROG	지연됨	IS: DB21,… DBX7.3 and DB11, … DBX20.5
PROG_STOP	알람 16954	NC 프로그램 : M0 및 M1
WAITM	알람 16954	NC 프로그램 : WAITM
WAITE	알람 16954	NC 프로그램 : WAITE
STOP_ALARM	즉시	알람 : 알람 구성 STOPBYALARM
RETREAT_MOVE_THREAD	알람 16954	NC 프로그램 : LFON 을 사용한 알람 16954 (G33 사용 불가능한 상황에서 정지 및 고속 리프트)
WAITMC	알람 16954	NC 프로그램 : WAITMC
NEWCONF_PREP_STOP	알람 16954	NC 프로그램 : NEWCONF
SYSTEM_SHUTDOWN	즉시	840Di 을 사용하여 시스템 중지
ESR	지연됨	확장된 정지 및 후진
EXT_ZERO_POINT	지연됨	외부 제로 옵셋
STOPRUN	알람 16955	OPI: PI "_N_FINDST" STOPRUN

응답에 대한 설명

즉시 (" 하드 " 정지 이벤트)	지연 정지 구간이라 하더라도 즉시 정지시킵니다 .
지연됨 (" 소프트 " 정지 이벤트)	지연 정지 구간 이후까지 (잠시라도) 정지시키지 않습니다 .
알람 16954	지연 정지 구간에서 허용되지 않는 프로그램 명령으로 인해 프로그램이 중단됩니다 .
알람 16955	프로그램이 계속됩니다. 지연 정지 구간에서 허용되지 않는 동작이 발생했습니다 .
알람 16957	DELAYFSTON 및 DELAYFSTOF 으로 둘러싸인 프로그램 섹션 (지연 정지 구간) 은 활성화될 수 없습니다 . 섹션 내에서는 모든 정지가 적용되며 , 지연되지 않습니다 .

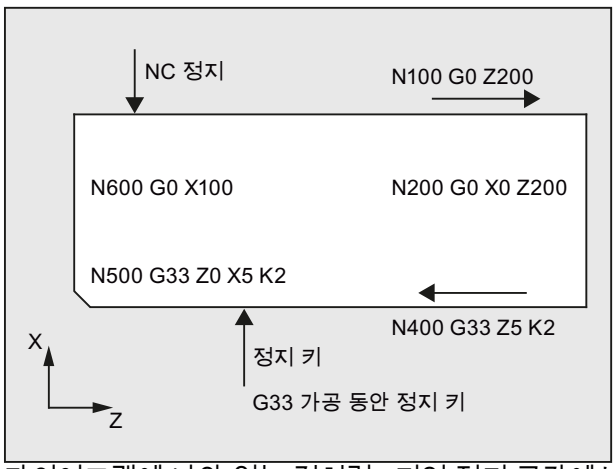
정지 이벤트에 대한 다른 응답의 목록은 다음을 참조하십시오 .
참조 문서 :
기능 메뉴얼 기본 구성 ; 모드 그룹 , 채널 , 프로그램 작업 , (K1), " 정지 이벤트에 대한 영향력 "

예 : 2 개의 프로그램 레벨에 지연 정지 구간 중첩시키기

프로그램 코드	비고
N10010 DELAYFSTON()	; N10xxx 프로그램 레벨 1 을 사용하는 블록
N10020 R1 = R1 + 1	
N10030 G4 F1	; 지연 정지 구간 시작
...	
N10040 subprogram2	
...	
...	; 서브프로그램 2 의 해석
N20010 DELAYFSTON()	; 효과 없음 , 반복된 시작 , 2 번째 레벨
...	
N20020 DELAYFSTOF()	; 효과 없음 , 다른 레벨에서 종료
N20030 RET	
N10050 DELAYFSTOF()	; 동일 레벨에서 지연 정지 구간 종료
...	
N10060 R2 = R2 + 2	
N10070 G4 F1	; 지연 정지 구간 종료 . 이제 정지는 즉시 작동함

예 : 프로그램 발췌

다음 프로그램 블록은 루프에서 반복됩니다 .



다이어그램에 나와 있는 것처럼 , 지연 정지 구간에서 "Stop" 을 누르면 NC 는 지연 정지 구간 외부 , 즉 블록 N100 내에서 감속을 시작합니다 . 그러면 NC 는 N100 시작에서 정지됩니다 .

프로그램 코드	비고
...	
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	

SERUPRO 유형의 블록의 세부 사항은 G331/G332 와 관련하여 검색 및 피드를 합니다 . 보정 척 없는 태핑용 피드는

참조 문서 :

기능 메뉴얼 기본 기능 ; 모드 그룹 , 채널 , 프로그램 작업 (K1)
기능 메뉴얼 기본 기능 ; 이송 속도 (V1) 를 참조하십시오 .

지연 정지 구간의 장점

프로그램 섹션은 속도 저하 없이 처리됩니다 .

RESET 으로 정지한 뒤 사용자가 프로그램을 중단한 경우 , 중단된 프로그램 블록은 보호된 섹션 이후가 됩니다 . 이 프로그램 블록은 다음 블록 탐색의 탐색 대상으로 적절합니다 .

다음 기본 실행 축은 지연 정지 구간을 진행하고 있는 동안에는 정지되지 않습니다 .

- 명령축
- POSA 와 함께 이동하는 위치 지정 축

가공 프로그램 명령 G4 는 일시 정지 (예 : WAITM) 를 유발하는 다른 가공 프로그램이 허용되지 않는 지연 정지 구간 내에서 허용됩니다 .

예제 : 이송 속도 개입

지연 정지 구간 이전에 오버라이드가 6% 로 줄어든 경우 , 오버라이드는 지연 정지 구간 내에서 활성화됩니다 .

지연 정지 구간에서 오버라이드가 100% 에서 6% 사이인 경우 , 100% 에서 지연 정지 구간이 완료되고 , 그 이후에서는 6% 까지 프로그램이 지속됩니다 .

지연 정지 구간에서 피드 사용 금지는 적용되지 않습니다 . 프로그램은 지연 정지 구간 이후까지 정지되지 않습니다 .

오버래핑 / 중첩 :

NC 명령 및 MD 11550: STOP_MODE_MASK 머신 데이터에서 나온 2 개의 지연 정지 구간이 오버랩된 경우 , 가장 큰 사용 가능한 지연 정지 구간이 생성됩니다 .

다음 특징들은 중첩과 서브루틴 끝을 사용하는 NC 명령 DELAYFSTON 과 DELAYFSTOF 사이의 상호 작용을 규제합니다 .

1. DELAYFSTOF 은 DELAYFSTON 가 호출되는 서브루틴의 끝에서 암시적으로 활성화됩니다 .
2. DELAYFSTON 지연 정지 구간은 영향력이 없습니다 .
3. 지연 정지 구간에서 서브루틴 1 이 서브루틴 2 를 호출할 경우 , 서브루틴 2 의 전체가 지연 정지 구간이 됩니다 . DELAYFSTOF 은 특히 서브루틴 2 에서 영향력이 없습니다 .

주

REPOSA 는 서브루틴 명령의 끝이며 DELAYFSTON 은 항상 선택 취소됩니다 .

"하드" 정지 이벤트와 "지연 정지 구간"이 동시에 발생하는 경우, "지연 정지 구간" 전체가 선택 휘소됩니다 ! 그러므로 , 이 프로그램 섹션 내에서 다른 정지가 발생하면 , 바로 정지됩니다 . 새로운 지연 정지 구간을 시작하기 위해서는 새 프로그램 설정 (새 DELAYFSTON) 을 만들어야만 합니다 .

지연 정지 구간 이전에 중지 키를 눌렀고 , NCK 를 중단시키기 위해 지연 정지 구간으로 이동해야만 하는 경우, NCK는 지연 정지 구간에서 정지되고 지연 정지 구간은 선택 취소된 상태로 유지됩니다 !

오버라이드 0% 인 상태에서 입력된 지연 정지 구간는 수용되지 **않습니다** !

이는 모든 " 소프트 " 정지 이벤트에 적용됩니다 .

STOPALL은 지연 정지 구간에서 감속에 사용될 수 있습니다. 하지만 STOPALL은 이전에 지연되었던 다른 모든 정지 이벤트를 즉시 활성화합니다 .

시스템 변수

지연 정지 구간은 $P_DELAYFST를 사용하는 가공 프로그램에서 감지될 수 있습니다. 시스템 변수의 비트 0 이 1 로 설정되어 있는 경우 , 가공 프로그램은 이제 지연 정지 구간에서 처리됩니다 .

지연 정지 구간은 $AC_DELAYFST 와 동기화 활동에서 감지될 수 있습니다 . 시스템 변수의 비트 0 이 1 로 설정되어 있는 경우 , 가공 프로그램은 이제 지연 정지 구간에서 처리됩니다 .

호환성

머신 데이터 MD 11550: STOP_MODE_MASK Bit 0 = 0 의 기본값은 G 코드 그룹 G331/G332 동안 및 경로 이동 또는 G4 가 프로그램되었을 때 암시적 지연 정지 구간을 트리거합니다 .

비트 0 = 1 은 G 코드 그룹 G331/G332 동안 및 경로 이동 또는 G4 가 프로그램되었을 때 (SW6 까지 동작) 정지를 허용합니다 . DELAYFSTON/DELAYFSTOF 명령은 지연 정지 구간을 정의하는 데 사용되어야만 합니다 .

8.5 SERUPRO (IPTRLOCK, IPTRUNLOCK) 에 대한 방지 프로그램 위치

작동

가공을 할 때 일부 복잡한 기계적 상황에서 블록 검색 SERUPRO 를 중단해야 합니다 .

프로그램 가능한 중단 지점을 사용하여 "중단 지점 탐색"을 사용한 추적 불가능한 지점 이전에 개입할 수 있습니다 .

NCK 를 아직 재입력할 수 없는 가공 프로그램 섹션에서 추적 불가능한 섹션을 정의하는 것도 가능합니다 . 프로그램이 중단되면 , NCK 는 처리가 가능했던 마지막 블록을 저장하고 그 이후 HMI 조작 인터페이스를 통해 탐색됩니다 .

구문

IPTRLOCK
IPTRUNLOCK

명령은 가공 프로그램 라인에 위치하고 있으며 프로그램 가능한 중단 지점을 허용합니다 .

중요도

IPTRLOCK	추적 불가능한 프로그램 섹션의 시작
IPTRUNLOCK	추적 불가능한 프로그램 섹션의 끝

이 두 명령은 가공 프로그램에서만 허용되지만 동기식으로는 작동하지 **않습니다** .

예제

추적 불가능한 프로그램 섹션은 암시적인 IPTRUNLOCK을 사용하여 두 개의 프로그램 레벨에 중첩되어 있습니다 . 서브 프로그램 1 에 있는 암시적인 IPTRUNLOCK 는 추적 불가능한 섹션에서 끝납니다 .

프로그램 코드	비고
N10010 IPTRLOCK()	
N10020 R1 = R1 + 1	
N10030 G4 F1	; 탐색을 방해하는 프로그램 섹션의 홀드 블록을 시작합니다 .
...	
N10040 subprogram2	
...	; 서브프로그램 2 의 해석
N20010 IPTRLOCK ()	; 효과 없음 , 반복된 시작
...	
N20020 IPTRUNLOCK ()	; 효과 없음 , 다른 레벨에서 종료
N20030 RET	
...	
N10060 R2 = R2 + 2	
N10070 RET	; 탐색을 방해하는 프로그램 섹션의 끝
N100 G4 F2	; 메인 프로그램이 계속됩니다 .

중단 지점은 이후 100 에서 다시 중단을 생산합니다 .

추적 불가능한 섹션의 획득 및 찾기

탐색이 불가능한 프로그램 섹션은 언어 명령 IPTRLOCK 및 IPTRUNLOCK을 사용하여 식별될 수 있습니다 .

IPTRLOCK 명령은 기본 실행 (SBL1) 에서 실행 가능한 단일 블록에서 중단 지점을 멈추니다 . 이 블록은 홀드 블록 아래로 참조하게 됩니다 . 프로그램이 IPTRLOCK 이후에 중단된 경우 , 홀드 블록은 HMI 사용자 인터페이스에서 찾아볼 수 있습니다 .

현재 블록에서 계속

중단 지점은 다음 프로그램 섹션에 대한 중단 지점으로 IPTRUNLOCK 을 사용한 현재 블록에 놓여 있습니다 .

일단 탐색 대상이 발견되면 , 홀드 블록을 사용하여 새로운 탐색 대상을 반복할 수 있습니다 .

사용자에 의해 편집된 중단 지점은 HMI 를 통해 다시 제거되어야만 합니다 .

충첩 규칙 :

다음 특징들은 중첩과 서브루틴 끝을 사용하는 NC 명령 IPTRLOCK 과 IPTRUNLOCK 사이의 상호 작용을 규제합니다 .

1. IPTRLOCK 은 IPTRUNLOCK 가 호출되는 서브루틴의 끝에서 암시적으로 활성화됩니다 .
2. 추적 불가능한 섹션의 IPTRLOCK 는 영향력이 없습니다 .
3. 만일 추적 불가능한 섹션에서 서브루틴 1 이 서브루틴 2 를 호출한 경우 , 2 개의 서브루틴 모두 추적 불가능한 상태로 남습니다 . IPTRUNLOCK 은 특히 서브루틴 2 에서 영향력이 없습니다 .

자세한 정보를 보시려면
/FB1/ 기능 메뉴얼 , 기본 기능 ; 모드 그룹 , 채널 , 프로그램 작업 모드 (K1) 를 참조하십시오 .

시스템 변수

추적 불가능한 섹션은 $P_IPTRLOCK 를 사용하는 가공 프로그램에서 감지될 수 있습니다 .

자동 중단 지점

자동 중단 지점은 이전에 정의된 결합 유형을 자동적으로 추적 불가능한 것으로 정의합니다 . 다음에 대한 머신 데이터

- EGON 을 사용한 전자 기어박스
- LEADON 을 사용한 축 지도 값 결합

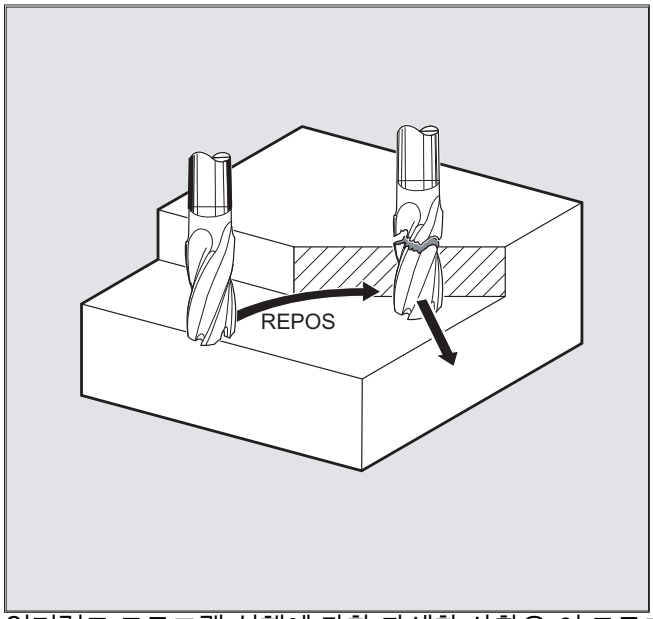
이들은 자동 중단 지점을 활성화시키는 데 사용됩니다 . 프로그램된 중단 지점 및 중단 지점이 자동 중단 지점 오버랩으로 활성화된 경우 , 가장 큰 추적 불가능한 섹션이 생성됩니다 .

8.6 형상에 대해 위치 변경 (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN)

기능

가공 중 공구가 부러졌거나 또는 수치를 확인하고 싶어 , 프로그램 실행을 중단시키고 공구를 후진시킨 경우 , 프로그램에 의해 제어되고 있는 상황에서 이전 중단된 위치로 축을 복귀시킬 수 있습니다 .

REPOS 명령은 서브프로그램 리턴 점프와 동일한 방법으로 작동합니다 (예 : M17 을 통함). 인터럽트 루틴의 명령 이후에 프로그램된 블록은 실행되지 않습니다 .



인터럽트 프로그램 실행에 관한 자세한 사항은 이 프로그래밍 매뉴얼의 " 플렉시블 NC 프로그래밍 " 섹션 , " 인터럽트 루틴 " 단락을 참조하십시오 .

구문

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=…

의미

접근 경로

REPOSA	전체 축이 라인을 따라 이동
REPOSL	라인을 따라 이동
REPOSQ DISR=…	반경 DISR 을 통해 4 분원을 따라 이동
REPOSQA DISR=…	전체 축이 반경 DISR 을 통해 4 분원을 따라 이동
REPOSH DISR=…	너비 DISR 을 사용하여 반원을 따라 이동
REPOSHA DISR=…	너비 DISR 을 사용하여 반원을 따라 이동

재접근 지점

RMI	인터럽트 지점 접근
RMI DISPR=…	인터럽트 지점 앞에서 mm/inch 로 거리 DISPR 에 있는 엔트리 포인트
RMB	블록 시작점 접근
RME	블록 끝점 접근
RME DISPR=…	끝점 앞의 거리 DISPR 에서 블록 종점 접근
RMN	가장 가까운 경로점 접근
A0 B0 C0	앞으로 접근하게 될 곳에 있는 축

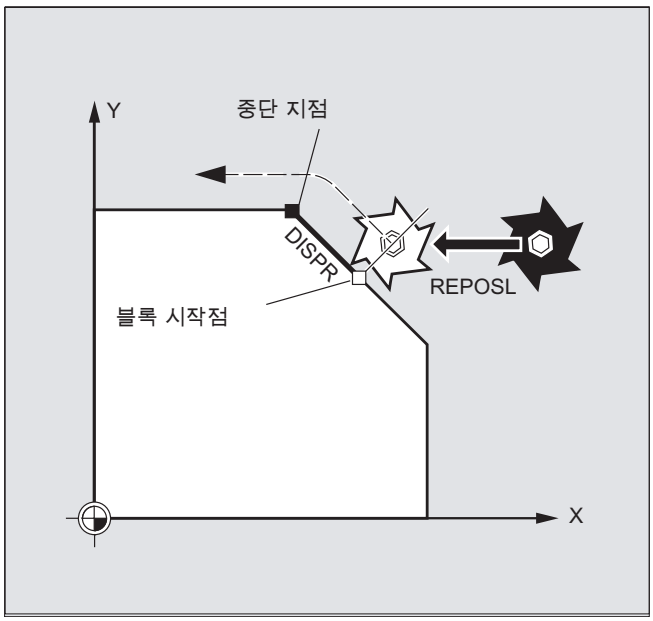
예제 : 직선 , REPOSA, REPOSL 을 따라 접근

공구는 직선을 따라 복귀 지점으로 접근합니다 .

모든 축은 REPOSA 명령에 따라 자동적으로 이동합니다 . REPOSL 을 사용하여 이송될 축을 지정할 수 있습니다 .

예제 :
REPOSL RMI DISPR=6 F400

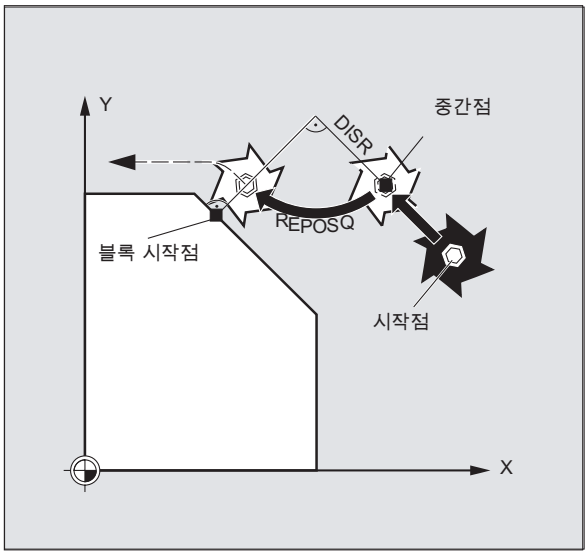
또는
REPOSA RMI DISPR=6 F400



예제 : 4 분원 , REPOSQ, REPOSQA 에서 접근

공구는 DISR=… 의 반경을 사용하여 4 분원을 따라 복귀 지점으로 접근합니다 . 제어 시스템은 시작점과 복귀 지점 사이의 중간점을 자동으로 계산합니다 .

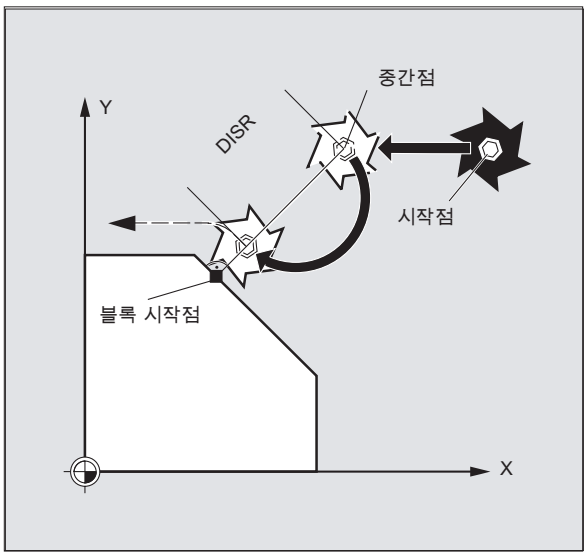
예제 :
REPOSQ RMI DISR=10 F400



예제 : 반원 , REPOSH, REPOSHA 으로 공구 접근

공구는 DISR=… 의 수치를 사용하여 반원을 따라 복귀 지점으로 접근합니다 . 제어 시스템은 시작점과 복귀 지점 사이의 필요한 중간점을 자동으로 계산합니다 .

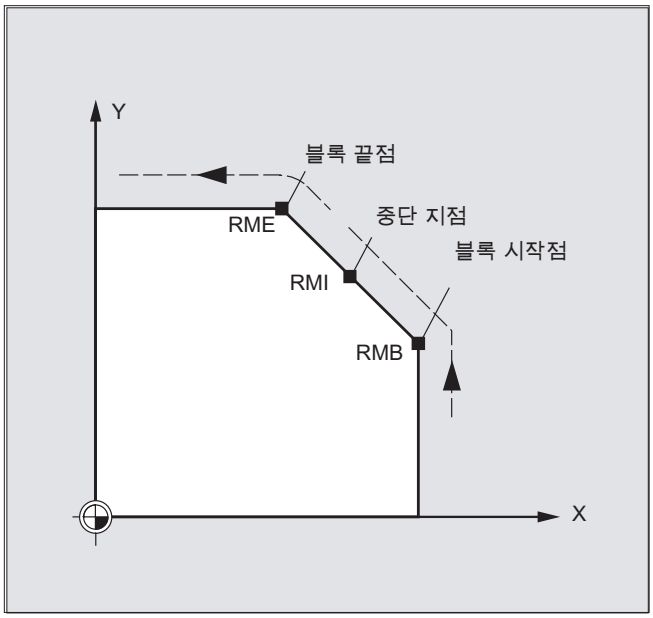
예제 :
REPOSH RMI DISR=20 F400



복귀 지점 지정 (RMN 을 사용하여 접근하는 SERUPRO 에 대한 접근이 아님)

프로그램 작동을 중단시킨 NC 블록을 기준으로 하여 3 개의 서로 다른 복귀 지점 중 하나를 선택할 수 있습니다 .

- RMI, 중단 지점
- RMB, 블록 시작점 또는 마지막 끝점
- RME, 블록 끝점



RMI DISPR=… 또는 RME DISPR=… 은 중단점 또는 블록 끝점 앞에 놓이는 복귀 지점을 선택할 수 있도록 해 줍니다 .

DISPR=… 은 끝점 **앞의** 중간점과 복귀 지점 사이의 형상 거리를 mmi/inch 로 나타낼 수 있도록 해 줍니다 . 값이 높더라도 이 점은 블록 시작점보다 멀리 나갈 수 없습니다 .

DISPR=… 명령이 프로그램되지 않은 경우 , DISPR=0 가 적용되고 , 이와 함께 중단점 (RMI 사용) 또는 블록 끝점 (RME 사용) 가 함께 적용됩니다 .

DISPR 부호

DISPR 부호가 평가됩니다 . 양수 부호인 경우 , 동작은 앞과 같습니다 .

음수 부호인 경우 , 접근은 중단점 뒤가 되거나 RMB 를 사용하는 경우에는 블록 시작점 뒤가 됩니다 .

중단점과 접근점 사이의 거리는 DISPR 의 값에 의해 달라집니다 . 값이 높더라도 이 점은 최대한 블록 끝점에 위치할 수 있습니다 .

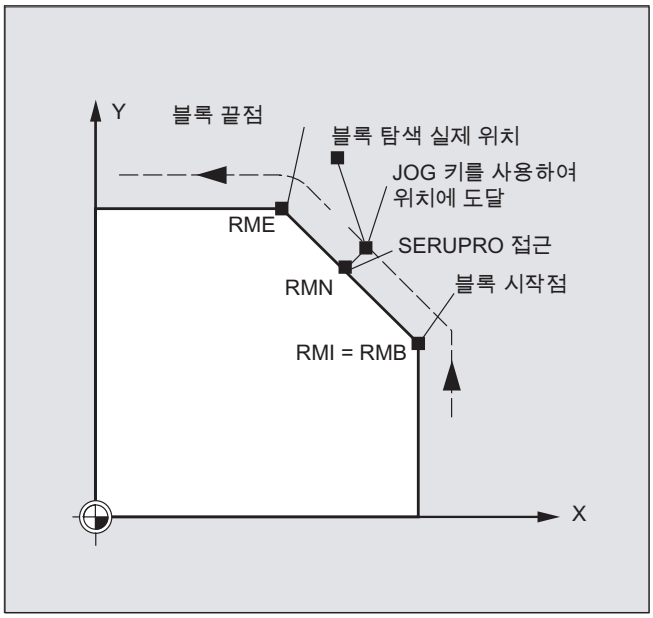
이용 예 :

센서는 클램프로 접근하는 것을 알아챕니다 . ASUP 은 클램프를 우회하도록 시작됩니다 .

그 이후 , 음의 DISPR 은 클램프 뒤의 한 점 위로 위치 변경되며 프로그램은 계속됩니다 .

RMN 을 사용한 SERUPRO 접근

어느 위치에서건 가공 중 중단이 강제된 경우 , 중단 지점으로부터 가장 짧은 경로는 SERUPRO 접근과 RMN을 사용하여 접근하기 때문에 그 이후에는 이동할 거리만 처리됩니다. 사용자는 중단 블록에서 SERUPRO 공정을 시작하고 JOG 키를 사용하여 대상 블록의 문제가 있는 요소 앞으로 이동합니다 .



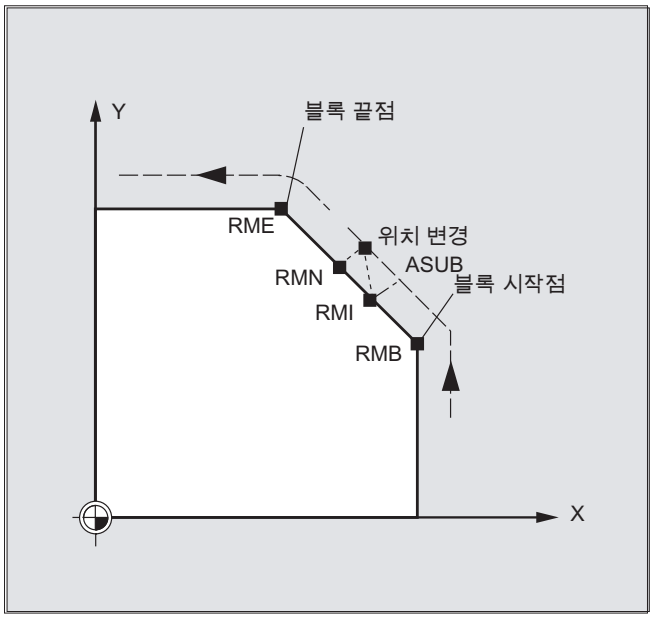
주

SERUPRO

SERUPRO , RMI 및 RMB 에 대해 동일합니다 . RMN 은 SERUPRO 로 제한되어 있지 않지만 일반적으로 사용 가능합니다 .

가장 가까운 경로점인 RMN 에서 접근

REPOSA 가 해석된 경우 , RMN 을 사용한 위치 변경 블록은 중단 이후 전면적으로 다시 시작되지 않고 단지 이동할 거리만 처리됩니다 . 중단된 블록의 가장 가까운 경로점에 접근합니다 .



유효한 REPOS 모드의 상태

중단된 블록의 유효한 REPOS 모드는 동기화 활동 및 다양한 $AC_ REPOS_PATH_MODE 을 사용하여 리드됩니다 .

0: 접근은 정의되지 않음

1 RMB: 시작으로 접근

2 RMI: 중단 지점으로 접근

3 RME: 블록 끝점으로 접근

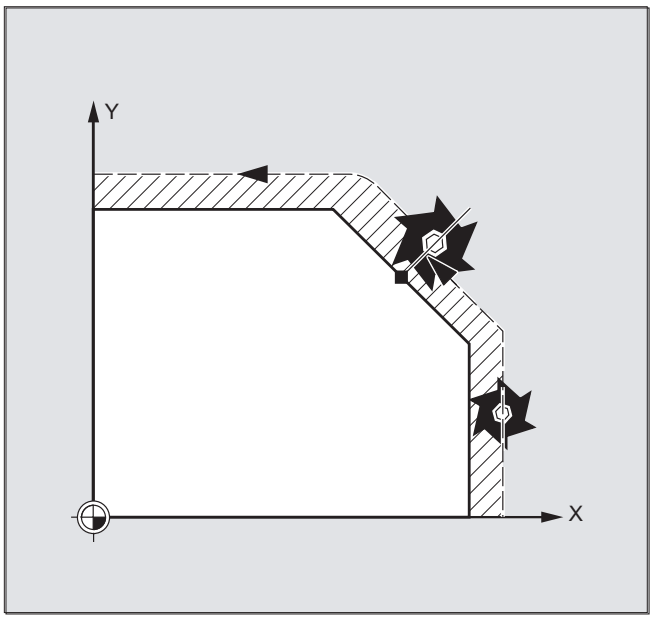
4 RMN: 중단된 블록의 다음 경로점으로 접근

새로운 공구를 사용하여 접근

공구 파손으로 인해 프로그램 작동을 중단시킨 경우 , 다음이 적용됩니다 .

새로운 D 번호가 프로그램된 경우 , 가공 프로그램은 복귀 지점에서 수정된 공구 옵셋 값을 사용하여 계속됩니다 .

공구 옵셋 값이 수정된 경우, 중단점으로 다시 접근할 수 없습니다. 이런 경우, 새로운 형상에서 중단점에 가장 가까운 지점으로 접근합니다 (DISPR 에 의해 수정됐을 가능성이 많음).



접근 형상

공구가 사용하는 움직임은 형상 위에서 위치가 변경되며 프로그램될 수 있습니다. 이송해야 할 축의 주소를 0 으로 입력하십시오 .

REPOSA, REPOSQA 및 REPOSHA 명령은 모든 축에서 자동으로 위치 변경됩니다 . 개별 축에 대해 이름을 지정해야 합니다 .

REPOSL, REPOSQ 및 REPOSH 명령이 프로그램된 경우 , 모든 기하 축은 자동으로 이송됩니다. 즉, 명령에서 이름을 지정할 필요가 없습니다. 다른 축은 모두 명령에서 지정해야 합니다 .

다음은 REPOSH 및 REPOSQ 원형 이동에 적용됩니다 .

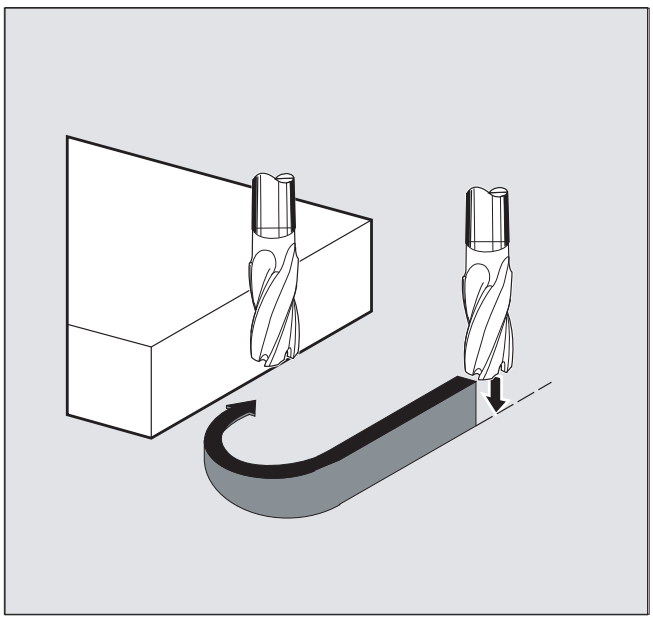
원형은 특정한 작업 평면인 G17 에서 G19 에서 적용됩니다 .

접근 브록에서 3차 형상축을 지정(절입 방향)한 경우, 복귀 지점은 공구 위치 및 절입 방향으로 프로그램된 위치가 동일하지 않을 때 나선을 따라 접근합니다 .

다음과 같은 경우 , 제어는 자동적으로

선형 접근인 REPOSL 로 전환됩니다 .

- DISR 에 대한 값을 지정하지 않았습니다 .
- 정의된 접근 방향을 사용할 수 없습니다 (이송 정보가 없는 블록에서 프로그램 중단).
- 접근 방향으로 사용하여 현재 작업 평면에 대해 수직이 됩니다 .



8.7 모션 제어에 영향

8.7.1 백분율 저크 수정 (JERKLIM)

기능

NC 명령 JERKLIM 을 사용하여 경로 모션 중 축의 최대 저크 (머신 데이터로 설정) 를 중요 프로그램 섹션에서 감소 또는 증가시킬 수 있습니다 .

사전 조건

가속 모드 SOFT 는 활성화되어 있어야만 합니다 .

유효성

이 기능은 다음 경우에 유효합니다 .

- AUTOMATIC 조작 모드인 경우
- 경로 축만 대상인 경우

구문

JERKLIM[<axis>]=<value>

의미

JERKLIM:	저크 수정에 대한 명령
< 축 >:	기계 축의 저크 제한값을 조정해야 합니다 .
< 값 >:	경로 모션에 설정된 최대 축 저크 (MD32431 $MA_MAX_AX_JERK) 를 참조한 백분율 수정 값 값 범위 : 1 ... 200 100 값은 저크에 영향을 주지 않습니다 .

주

가공 프로그램 종료 및 채널 리셋 시 JERKLIM 의 동작은 머신 데이터 MD32320 $MA_DYN_LIMIT_RESET_MASK 를 비트 0 으로 설정합니다 .

- 비트 0 = 0:
  JERKLIM 에 프로그래밍된 값은 채널 리셋 /M30 과 함께 100% 로 리셋됩니다 .
- 비트 0 = 1:
  JERKLIM 에 프로그래밍된 값은 채널 리셋 /M30 이후에도 유지됩니다 .

예제

프로그램 코드	설명
...	
N60 JERKLIM[X]=75	; X 방향에서 축 슬라이드는 해당 축에 대해 저크 허용치의 75% 까지만 가속 / 감속될 수 있습니다 .
...	

8.7.2 백분율 속도 수정 (VELOLIM)

기능

NC 명령 VELOLIM 을 사용하면 축 작업 시 축 / 스핀들의 최대 속도 또는 스핀들 작업 시 기어 유닛 단계별 최대 스핀들 속도 (머신 데이터를 사용하여 설정 : 속도 제어 모드인 경우 M3, M4, M5, 포지셔닝 모드인 경우 SPOS, SPOSA, M19) 를 기계에 작용하는 스트레스 감소 또는 가공 품질 향상 등을 위해 중요 프로그램 섹션에서 감소시킬 수 있습니다 .

유효성

이 기능은 다음 경우에 유효합니다 .

- AUTOMATIC 조작 모드인 경우
- 경로 축 및 포지셔닝 축이 대상인 경우
- 스핀들 / 축 작업에서 스핀들이 대상인 경우

구문

VELOLIM[< 축 / 스핀들 >]=< 값 >

의미

VELOLIM: 속도 수정에 대한 명령

< 축 / 스핀들 >: 속도 한계값을 조정해야 하는 기계 축 또는 스핀들

스핀들 VELOLIM

가공 프로그램을 프로그래밍할 때 머신 데이터 (MD30455 $MA_MISC_FUNCTION_MASK, 비트 6) 를 사용하여 스핀들 또는 축 (비트 6 = 1) 사용 여부에 관계 없이 VELOLIM 을 실행할지 여부 또는 조작 모드별로 다르게 프로그래밍 (비트 6 = 0) 을 허용할지 여부를 설정할 수 있습니다 . 조작 모드에 따라 다르게 프로그래밍하는 경우 프로그래밍을 할 때 이름을 사용하여 대상을 선택합니다 .

- 스핀들 조작 모드인 경우 스핀들 이름 S<n>
- 축 조작 모드인 경우 축 이름 (예 : "C")

< 값 >: 백분율 수정 값

수정 값은 다음과 같습니다 .

- 축 조작 모드에서 축 / 스핀들에 대해 (MD30455 비트 6 = 0 인 경우):

  설정된 최대 축 속도 (MD32000 $MA_MAX_AX_VELO) 로 수정

- 스핀들 또는 축 조작 모드에서 스핀들에 대해 (MD30455 비트 6 = 1 인 경우):

  활성 기어 유닛 단계의 최대 속도 (MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT[<n>]) 로 수정

값 범위 : 1 ... 100

값 100 은 속도에 영향을 미치지 않습니다 .

주

가공 프로그램 종료 및 채널 리셋 시 동작

가공 프로그램 종료 및 채널 리셋 시 VELOLIM 의 동작은 머신 데이터 MD32320 $MA_DYN_LIMIT_RESET_MASK 를 비트 0 으로 설정합니다 .

- 비트 0 = 0:

  VELOLIM 프로그래밍된 값은 채널 리셋 /M30 과 함께 100% 로 리셋됩니다 .

- 비트 0 = 1:

  VELOLIM 에 프로그래밍된 값은 채널 리셋 /M30 이후에도 유지됩니다 .

주

동기 동작 중 스핀들 VELOLIM

동기 동작에 VELOLIM 을 프로그래밍할 때는 스핀들 조작과 축 조작을 구분하지 않습니다 . 프로그래밍에 사용된 이름과 관계 없이 스핀들 조작 모드의 속도와 축 조작 모드의 속도가 동일한 수준으로 제한됩니다 .

진단

스핀들 조작 시 VELOLIM 진단

스핀들 조작 시 시스템 변수 $AC_SMAXVELO 및 $AC_SMAXVELO_INFO 를 읽어와서 VELOLIM (100 % 미만) 을 사용하는 활성 속도 제한을 구분할 수 있습니다 .

속도 제한의 경우 $AC_SMAXVELO 는 VELOLIM 이 생성한 속도 한계를 제공합니다 . 이 때 변수 $AC_SMAXVELO_INFO 는 속도 제한 변수 VELOLIM 을 위한 식별자로 값 16 을 반환합니다 .

예제

예제 1: 기계 축 속도 제한

프로그램 코드	설명
...	
N70 VELOLIM[X]=80	; X 방향의 축 슬라이드는 축에 허용된 속도의 최대 80% 로 이송할 수 있습니다 .
...	

예제 2: 스핀들 속도 제한

프로그램 코드	설명
N05 VELOLIM[S1]=90	; 스핀들 1 의 최대 속도를 1000 rpm 의 90% 로 제한
...	
N50 VELOLIM[C]=45	; 속도를 1000 rpm 의 45% 로 제한 , C 는 S1 의 축 이름 .
...	

스핀들 1 의 데이터 설정 (AX5):

MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT[1,AX5]=1000	; 기어 유닛 1 단계의 최대 속도 = 1000 rpm
MD30455 $MA_MISC_FUNCTION_MASK[AX5] = 64	; 비트 6 = 1: VELOLIM 프로그래밍은 프로그래밍된 식별자와 관계 없이 스핀들 조작과 축 조작에 모두 적용됩니다 .

8.7.3 JERKLIM 및 VELOLIM 에 대한 프로그램의 예

다음 프로그램은 백분율 저크 및 속도 제한에 대해 적정한 예를 나타냅니다 .

프로그램 코드	비고
N1000 G0 X0 Y0 F10000 SOFT G64	
N1100 G1 X20 RNDM=5 ACC[X]=20	
ACC[Y]=30	
N1200 G1 Y20 VELOLIM[X]=5	; X 방향에서 축 슬라이드는 축에 대해 속도 허용치의 5% 까지만 이송해야 합니다 .
JERKLIM[Y]=200	; X 방향에서 축 슬라이드는 축에 대해 저크 허용치의 200% 까지만 가속 / 감속될 수 있습니다 .
N1300 G1 X0 JERKLIM[X]=2	; X 방향에서 축 슬라이드는 축에 대해 저크 허용치의 최대 2% 까지만 가속 / 감속될 수 있습니다 .
N1400 G1 Y0	
M30	

8.8 프로그램 가능한 형상 / 오리엔테이션 공차 (CTOL, OTOL, ATOL)

기능

CTOL, OTOL 및 ATOL 명령을 사용하면 컴프레서 기능 (COMPON, COMPCURV, COMPCAD) 에 대해 정의된 가공 공차 , 스무딩 유형 G642, G643, G645, OST 그리고 NC 프로그램에서 머신 데이터 및 셋팅 데이터를 사용하는 오리엔테이션 ORISON 을 적용할 수 있습니다 .

프로그래밍된 값은 음의 값 지정을 통해 다시 프로그래밍되거나 삭제될 때까지 유효합니다 . 또한 프로그램 끝에서, 채널 리셋, 모드 그룹 리셋, NCK 리셋 (웜 리스타트) 그리고 전원을 켤 때 (콜드 리스타트) 도 프로그래밍 값이 삭제됩니다 . 이 값을 삭제하면 머신 데이터 및 셋팅 데이터의 값이 복원됩니다 .

구문

CTOL=< 값 >
OTOL=< 값 >
ATOL[< 축 >]=< 값 >

의미

CTOL **형상 공차**를 프로그래밍하는 명령어
CTOL 를 사용하는 기능 :
- 모든 컴프레서 기능
- G641 및 G644 를 제외한 모든 라운딩 유형

< 값 >: 형상 공차 값은 길이로 지정됩니다 .
유형 : REAL
단위 : inch/mm (현재 치수 설정에 따라)

OTOL **오리엔테이션 공차**를 프로그래밍하는 명령어
OTOL 을 사용하는 기능 :
- 모든 컴프레서 기능
- ORISON 오리엔테이션 스무딩
- G641, G644, OSD 를 제외한 모든 라운딩 유형

< 값 >: 오리엔테이션 공차 값은 각도로 지정됩니다 .
유형 : REAL
단위 : 각도

ATOL **축 공차**를 프로그래밍하는 명령어

ATOL 를 사용하는 기능 :

- 모든 컴프레서 기능
- ORISON 오리엔테이션 스무딩
- G641, G644, OSD 를 제외한 모든 라운딩 유형

< 축 >: 축 공차를 프로그래밍할 축의 이름

< 값 >: 축 공차 값은 축 유형 (직선 축 또는 로터리 축) 에 따라 길이 또는 각도로 지정됩니다 .

유형 : REAL

단위 : 직선 축 : inch/mm (현재 치수 설정에 따라)

로터리 축 : 각도

주

CTOL 및 OTOL 은 ATOL 보다 우선합니다 .

일반 조건

스케일링 프레임

스케일링 프레임은 축 위치와 같은 방식으로 프로그래밍된 공차에 영향을 줍니다 . 즉 , 상대적 공차는 계속 동일합니다 .

예제

프로그램 코드	설명
COMPCAD G645 G1 F10000	; COMPCAD 컴프레서 기능을 활성화합니다 .
X... Y... Z...	; 머신 데이터 및 셋팅 데이터가 여기에 적용됩니다 .
X... Y... Z...	
X... Y... Z...	
CTOL=0.02	; 형상 공차 0.02mm 가 여기부터 적용됩니다 .
X... Y... Z...	
X... Y... Z...	
X... Y... Z...	
ASCALE X0.25 Y0.25 Z0.25	; 형상 공차 0.005mm 가 여기부터 적용됩니다 .
X... Y... Z...	
X... Y... Z...	
X... Y... Z...	
CTOL=–1	; 머신 데이터 및 셋팅 데이터가 여기부터 다시 적용됩니다 .
X... Y... Z...	
X... Y... Z...	
X... Y... Z...	

추가 정보

공차 값 읽기

보다 향상된 적용 또는 진단을 위해 컴프레서 기능 (COMPON, COMPCURV, COMPCAD), 스무딩 유형 G642, G643, G645, OST 그리고 오리엔테이션 스무딩 ORISON 의 현재 유효 공차를 그 발생 방식에 관계없이 시스템 변수를 통해서 읽을 수 있습니다 .

- 가공 프로그램에서 시스템 변수를 통한 동기 동작 또는 전처리 정지 사용 :

$AC_CTOL	현재의 주 실행 기록이 전처리되었을 때 형상 공차가 적용됩니다 . 형상 공차가 적용되지 않은 경우 $AC_CTOL 은 기하 축의 공차 제곱 합계의 근을 반환합니다 .
$AC_OTOL	현재의 주 실행 기록이 전처리되었을 때 오리엔테이션 공차가 적용됩니다 . 오리엔테이션 공차가 적용되지 않은 경우 $AC_OTOL 은 오리엔테이션 변환이 활성 상태일 때 오리엔테이션 축의 공차 제곱 합계의 근을 반환합니다 . 그 밖의 경우에는 '-1' 값을 반환합니다 .
$AA_ATOL[< 축 >]	현재의 주 실행 기록이 전처리되었을 때 축 공차가 적용됩니다 . 활성 상태인 형상 공차가 없는 경우 $AA_ATOL[<기하 축 >] 은 형상 공차를 기하 축 수의 근으로 나눈 값을 반환합니다 . 오리엔테이션 공차 및 오리엔테이션 변환이 활성 상태인 경우 $AA_ATOL[< 오리엔테이션 축 >] 은 오리엔테이션 공차를 오리엔테이션 축 수의 근으로 나눈 값을 반환합니다 .

주

공차 값을 프로그래밍하지 않으면 $A 변수는 하나의 값만 선언할 수 있으므로 각 기능의 공차에 있을 수 있는 차이를 구별할 수 있을 정도로 달라지지 않습니다 .

이러한 상황은 머신 데이터 및 셋팅 데이터가 컴프레서 기능 , 스무딩 및 오리엔테이션 스무딩에 대해 서로 다른 공차를 설정한 경우에 발생할 수 있습니다 . 그러면 변수들은 현재 활성 기능과 관련하여 주로 쓰는 가장 큰 값을 반환합니다 .

예를 들어 컴프레서 기능이 오리엔테이션 공차 0.1° 로 활성 상태이고 ORISON 오리엔테이션 스무딩이 1° 로 설정된 경우 $AC_OTOL 변수는 값 '1' 을 반환합니다 . 오리엔테이션 스무딩이 비활성화된 경우에는 '0.1' 값만 읽혀집니다 .

- 가공 프로그램에서 시스템 변수를 통한 전처리 정지 없음 :

$P_CTOL	프로그래밍된 형상 공차
$P_OTOL	프로그래밍된 오리엔테이션 공차
$AA_ATOL	프로그래밍된 축 공차

주

공차 값을 프로그래밍하지 않은 경우 $P 변수는 '-1' 값을 반환합니다 .

8.9 G0 모션 공차 (STOLF)

G0 공차 팩터

G0 모션 (급 이송, 절입 모션) 은 공작물 가공과 달리 공차 수준을 더 높게 설정할 수 있습니다. 이렇게 하면 G0 모션의 실행 시간이 단축된다는 장점이 있습니다 .

G0 모션의 경우 G0 공차 팩터 (MD20560 $MC_G0_TOLERANCE_FACTOR) 를 구성하여 공차를 설정합니다 .

G0 공차 팩터는 다음의 경우에만 적용됩니다 .

- 다음 함수 중 하나가 활성인 경우
  - 컴프레서 기능 : COMPON, COMPCURV 및 COMPCAD
  - 스무딩 함수 : G642 및 G645
  - 오리엔테이션 스무딩 : OST
  - 오리엔테이션 스무딩 : ORISON
  - 경로 관련 오리엔테이션 스무딩 : ORIPATH
- 연속 G0 블록이 여러 개 (≥ 2) 인 경우

  G0 블록이 1 개인 경우 G0 공차 팩터가 적용되지 않습니다 . 비 G0 모션에서 G0 모션으로 , 또는 그 반대로 **트렌지션할 때** 항상 ' **더 낮은 공차** ', 즉 공작물 가공 공차가 적용되기 때문입니다 .

기능

설정된 G0 공차 팩터 (MD20560) 를 가공 프로그램에 STOLF 를 프로그래밍하여 일시적으로 덮어쓸 수 있습니다 . 이 경우 MD20560 의 값 자체는 변경되지 않습니다 . 리셋 또는 가공 프로그램 종료 후 설정된 공차 팩터가 다시 적용됩니다 .

구문

STOLF=< 공차 팩터 >

의미

STOLF:	G0 공차 팩터를 프로그래밍하는 명령
< 공차 팩터 >:	G0 공차 팩터 공차 팩터는 1 보다 크거나 1 보다 적을 수 있습니다 . 하지만 G0 모션에는 일반적으로 더 높은 공차가 설정됩니다 . STOLF=1.0 인 경우 (1 은 설정된 표준 값에 해당) 비 G0 모션에 적용되는 공차와 동일한 공차가 G0 모션에 적용됩니다 .

시스템 변수

가공 프로그램 또는 실제 IPO 블록에 적용된 G0 공차 팩터는 시스템 변수를 사용하여 읽을 수 있습니다 .

- 시스템 변수를 통한 가공 프로그램에서의 동기 동작 또는 전처리 정지 :

  $AC_STOLF 활성 G0 공차 팩터
  실제 메인 실행 블록을 처리할 때 적용된 G0 공차 팩터

- 시스템 변수를 통한 가공 프로그램의 전처리 정지 안 함 :

  $P_STOLF 프로그램된 G0 공차 팩터

활성 가공 프로그램에 STOLF 의 값을 프로그래밍하지 않으면 이 두 시스템 변수는 MD20560 $MC_G0_TOLERANCE_FACTOR 를 사용하여 설정한 값을 제공합니다 .

블록에 사용 중인 급 이송 (G0) 이 없으면 이 시스템 변수는 항상 값 1 을 제공합니다 .

예제

프로그램 코드	설명
COMPCAD G645 G1 F10000	; 컴프레서 함수 COMPCAD
X... Y... Z...	; 머신 데이터 및 셋팅 데이터가 여기에 적용됩니다 .
X... Y... Z...	
X... Y... Z...	
G0 X... Y... Z...	
G0 X... Y... Z...	; 머신 데이터 $MC_G0_TOLERANCE_FACTOR (예 : 3) 이 여기에 적용됩니다 . 즉 , $MC_G0_TOLERANCE_FACTOR*$MA_COMPRESS_POS_TOL 의 스무딩 공차가 적용됩니다 .
CTOL=0.02	
STOLF=4	
G1 X... Y... Z...	; 여기부터 형상 공차 0.02 mm 가 적용됩니다 .
X... Y... Z...	
X... Y... Z...	
G0 X... Y... Z...	
X... Y... Z...	; 여기부터 G0 공차 팩터로 4 가 적용됩니다 . 즉 , 형상 공차 0.08 mm 가 적용됩니다 .

축 커플링

9

9.1 결합 작동 (TRAILON, TRAILOF)

기능

정의 된 리딩 축이 움직일 때 , 커플 축 (= 추종 축) 은 커플링 팩터가 허용하는 리딩 축에 의해 설정된 거리를 이송하는 데 할당됩니다 .

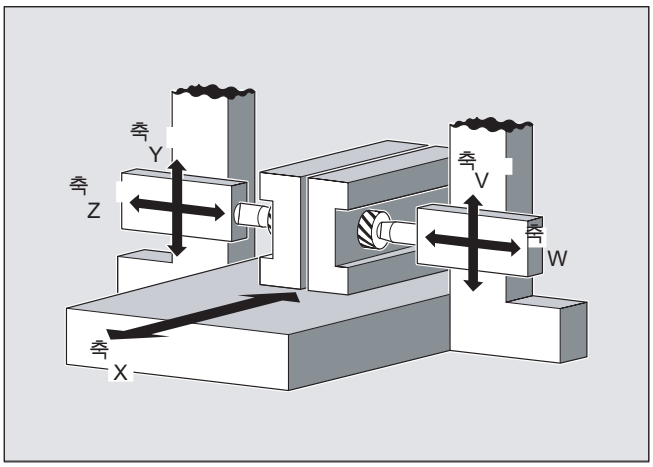
커플 축은 리딩 축과 추종 축으로 구성됩니다 .

적용

- 가상의 축을 사용하여 축을 이송합니다. 리딩 축은 가상의 축으로 커플 축이 실제 축입니다. 이런 방법으로 , 커플링 팩터의 기능에 의해 실제 축이 이송됩니다 .
- 2 개의 커플 동작 그룹을 통한 양면 가공 :

  1. 리딩 축 Y, 커플 축 V

  2. 리딩 축 Z, 커플 축 W



구문

TRAILON(< 추종 축 >,< 리딩 축 >,< 커플링 팩터 >)
TRAILOF(< 추종 축 >,< 리딩 축 >,< 리딩 축 2>)
TRAILOF(< 추종 축 >)

의미

TRAILON	커플 축 그룹핑 활성화 및 정의를 위한 명령 활성 : 모달
< 추종 축 >	파라미터 1: 트레일링 축의 축 이름 **참고 :** 커플 축은 다른 커플 축의 리딩 축 역할도 할 수 있습니다 . 이런 방법으로 서로 다른 커플 축 그룹화의 범위를 만들 수 있습니다 .
< 리딩 축 >	파라미터 2: 트레일링 축의 축 이름
< 커플링 팩터 >	파라미터 3: 커플링 팩터 커플링 팩터는 커플 축과 리딩 축의 경로 사이에 알맞은 관계를 지정합니다 . < 커플링 팩터 > = 커플 축의 경로 / 리딩 축의 경로 유형 : REAL 표준 : 1 마이너스 값을 입력하면 마스터 및 커플 축이 반대 방향으로 이송됩니다 . 커플링 팩터가 프로그램되지 않은 경우 , 커플링 팩터 1 이 자동적으로 적용됩니다 .
TRAILOF	커플 축 그룹핑 해제 명령 적용 : 모달 2 개의 파라미터를 사용한 TRAILOF 는 지정된 리딩 축에 대한 커플링만 해제합니다 . TRAILOF(< 추종 축 >,< 리딩 축 >) 커플 축 하나에 리딩 축이 2 개인 경우에는 3 개의 파라미터가 적용된 TRAILOF 를 호출하여 두 커플링을 모두 해재할 수 있습니다 . TRAILOF(< 추종 축 >,< 리딩 축 >,< 리딩 축 2>) 리딩 축을 지정하지 않고 TRAILOF 를 프로그래밍해도 동일한 결과가 나옵니다 . TRAILOF(< 추종 축 >)

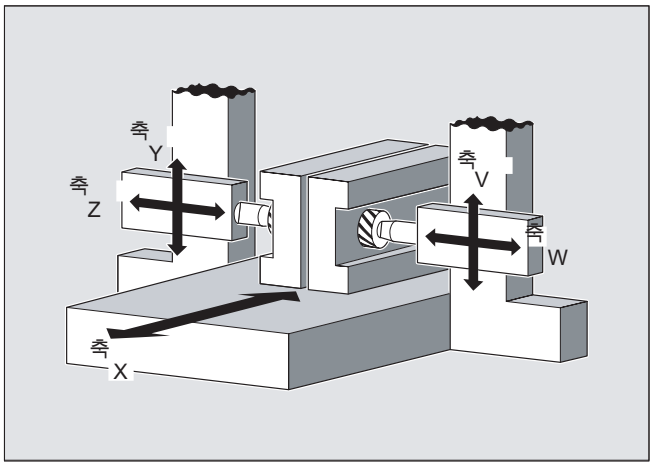
주

커플 축 동작은 항상 베이직 좌표계 (BCS) 에서 실행됩니다 .

동시에 활성화될 수 있는 커플 축 그룹핑의 갯수는 기계 상에서 조합될 수 있는 최대 갯수의 조합으로 제한됩니다 .

예제

다이어그램에 나타나 있는 축 구성을 사용하여 공작물을 양면에서 가공합니다 . 이를 위해 , 2 개의 커플 축 조합을 만듭니다 .



프로그램 코드	설명
…	
N100 TRAILON(V,Y)	; 첫 번째 커플 축 그룹핑 활성화
N110 TRAILON(W,Z,–1)	; 두 번째 커플 축 그룹핑 활성화 , 음의 커플링 팩터: 커플 축이 리딩 축과 반대 방향으로 이송함
N120 G0 Z10	; 축 반대 방향으로 Z 축 및 W 축 절입
N130 G0 Y20	; 축과 동일한 방향으로 Y 축 및 V 축 절입
…	
N200 G1 Y22 V25 F200	; 커플 축 V 의 종속적 및 독립적 이동 오버레이
…	
TRAILOF(V,Y)	; 첫 번째 커플 축 그룹핑 해제
TRAILOF(W,Z)	; 두 번째 커플 축 그룹핑 해제

추가 정보

축 종류

커플 축 그룹핑은 원하는 리니어 축 및 로터리 축의 조합으로 구성됩니다 . 시뮬레이션 축 또한 리딩 축으로 지정할 수 있습니다 .

커플 축

동시에 최대 2 개의 리딩 축을 트레일링 축으로 지정할 수 있습니다 . 서로 다른 커플 축의 조합에 할당할 수 있습니다 .

커플 축은 사용 가능한 모든 모션 명령 (G0, G1, G2, G3 등) 을 사용하여 프로그래밍할 수 있습니다 .. 커플 축은 정의된 경로를 독립적으로 이송하는 것뿐 아니라 커플링 팩터를 기반으로 하여 리딩 축에서 나온 것도 이송합니다 .

다이나믹 한계

다이나믹 한계는 커플 축 그룹핑의 활성화 유형에 따라 달라집니다 .

- 가공 프로그램에서 활성화

  가공 프로그램에서 활성화가 이루어지고 모든 리딩 축은 활성화된 채널에서 프로그램 축의 역할을 하는 경우 커플 축의 과부하를 방지하기 위해 리딩 축의 이송 과정에서 모든 커플 축의 다이나믹 반응이 고려됩니다 .

  가공 프로그램에서 활성화가 이루어지고 리딩 축은 활성화된 채널에서 프로그램 축의 역할을 하지 않는 경우 ($AA_TYP ≠ 1) 리딩 축의 이송 과정에서 커플 축의 다이나믹 반응이 고려되지 않습니다 . 이로 인해 커플링에 필요한 정도 이하의 다이나믹 반응에 따라 커플 축에 과부하가 발생할 수 있습니다 .

- 동기 동작의 활성화

  동기 동작에서 활성화가 이루어지면 리딩 축의 이송 과정에서 커플 축의 다이나믹 반응이 고려되지 않습니다 . 이로 인해 커플링에 필요한 정도 이하의 다이나믹 반응에 따라 커플 축에 과부하가 발생할 수 있습니다 .

⚠ 주의

다음과 같은 경우에서 커플 축 그룹핑이 활성화 되면 :

- 동기 동작에서
- 커플 축의 채널에서 프로그램 축 역할을 하지 않는 리딩 축을 가진 가공 프로그램에서

리딩 축 이송으로 인해 커플 축에 과부하가 발생하지 않도록 적절한 조치를 취하는 것은 사용자 / 장비 제조업체의 책임입니다 .

커플링 상태

축의 커플링 상태는 가공 프로그램에서 시스템 변수로 확인할 수 있습니다 .

$AA_COUP_ACT[< 축 >]

값	의미
0	커플링 동작 안 함
8	커플링 동작

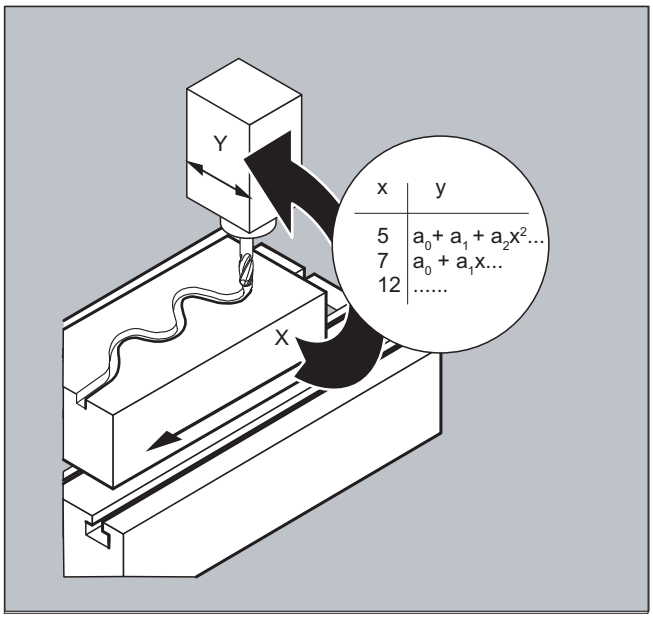
9.2 커브 테이블 (CTAB)

기능

커브 테이블은 두 축 (리딩 축과 추종 축) 의 위치 및 속도 관계를 프로그래밍하는 데 사용할 수 있습니다 . 커브 테이블은 가공 프로그램에서 정의됩니다 .

적용

커브 테이블은 기계적 캠을 대신합니다 . 커브 데이블은 리딩 및 추종 값 간의 기능적인 관계를 생성하여 축 마스터 값 커플링의 기본을 구성합니다 . 적절한 프로그램을 사용하여 , 시스템은 리딩 축 및 추종 축의 상대적 위치로부터 캠까지에 해당하는 폴리노미널을 계산합니다 .

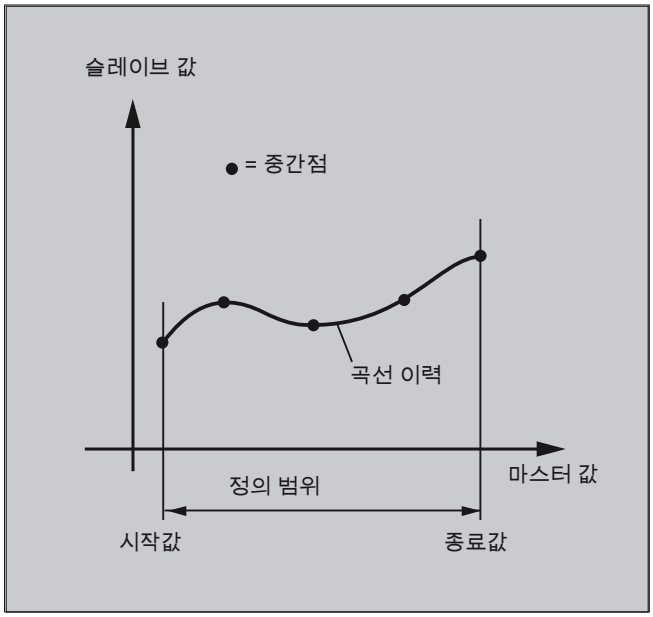


9.2.1 커브 테이블 정의 (CTABDEF, CATBEND)

기능

커브 테이블이란 시작 부분에 CTABDEF 을 그리고 끝 부분에 CTABEND 을 포함한 가공 프로그램 또는 가공 프로그램의 섹션을 의미합니다 .

이러한 가공 프로그램의 섹션에는 모션 조작을 통해 리딩 축의 개별 위치에 고유의 추종 축이 지정되며 , 이 추종 축의 위치는 폴리노미널 형태의 커브 정의를 최대 5 차까지 계산할 때 중간점으로 사용됩니다 .



조건

커브 테이블 정의에 충분한 메모리 공간을 마련하려면 이에 따라 MD를 설정해야 합니다 (장비 제조업체 사양 참조).

구문

CTABDEF(< 추종 축 >,< 리딩 축 >,<n>,< 주기성 >[,< 메모리 위치 >])
...
CTABEND

의미

CTABDEF( )	커브 테이블 정의 시작
CTABEND	커브 테이블 정의 끝
< 추종 축 >	커브 테이블을 사용하여 모션을 계산할 축
< 리딩 축 >	추종 축 모션 계산을 위한 마스터 값을 제공하는 축
<n>	커브 테이블 번호 (**ID**) 커브 테이블 번호는 고유한 것이며 메모리 위치와 무관합니다 . 그러므로 스태틱 및 다이나믹 NC 메모리에 동일한 번호의 테이블이 존재할 수 없습니다 .
< 주기성 >	테이블 주기성 0 테이블은 주기적이지 않습니다 . (로터리 축의 경우에도 테이블은 단 한 번만 처리됩니다 .) 1 테이블은 리딩 축에 주기적으로 적용됩니다 . 2 테이블은 리딩 축 및 추종 축에 주기적으로 적용됩니다 .
< 메모리 위치 >	메모리 위치 지정 (옵션) 'SRAM' **스태틱** NC 메모리에 커브 테이블이 생성됩니다 . 'DRAM' **다이나믹** NC 메로리에 커브 테이블이 생성됩니다 . **참고 :** 이 파라미터에 대한 값을 프로그래밍하지 않으면 MD20905 $MC_CTAB_DEFAULT_MEMORY_TYPE 을 통해 설정된 기본 메모리 위치가 사용됩니다 .

주

덮어쓰기

특정 커브 테이블 번호 (<n>) 가 다른 테이블 정의에 사용되면 즉시 그 커브 테이블을 덮어쓰게 됩니다 . 다만 커브 테이블이 축 커플링에서 활성 상태이거나 CTABLOCK 으로 잠금 설정된 경우는 예외입니다 . **커브 테이블을 덮어쓴 경우에는 경고가 출력되지 않습니다 .**

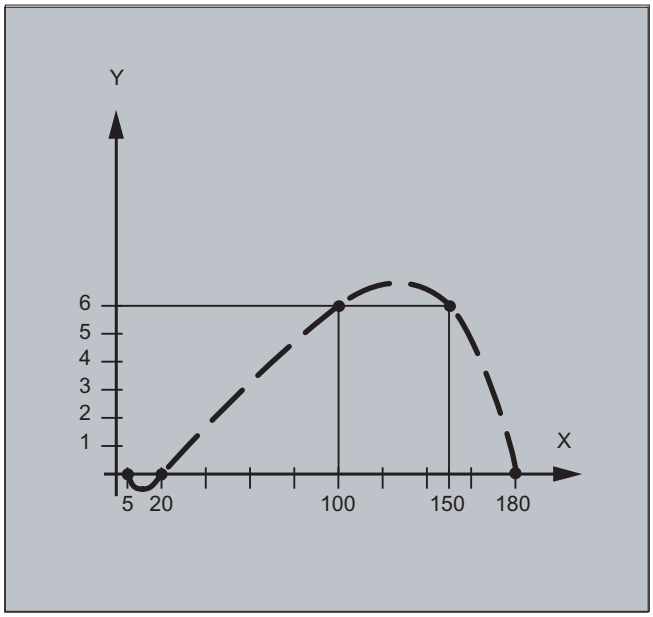
예제

예제 1: 커브 테이블 정의 프로그램 섹션

프로그램 섹션은 커브 테이블 정의 시 변경되지 않은 상태로 사용됩니다 . 전처리 정지를 위한 STOPRE 에 대한 명령은 그대로 유지되며 프로그램 섹션이 테이블 정의에 사용되지 않고 CTABDEF 및 CTABEND 가 제거됨과 동시에 다시 활성화됩니다 .

프로그램 코드	설명
…	
CTABDEF(Y,X,1,1)	; 커브 테이블 정의 .
…	
($P_CTABDEF) 이 아닌 경우	
STOPRE	
ENDIF	
…	
CTABEND	

예제 2: 비 주기적 커브 테이블 정의



프로그램 코드	설명
N100 CTABDEF(Y,X,3,0)	; 번호가 3 인 비주기적 커브 테이블 정의 시작 .
N110 X0 Y0	; 첫 번째 모션 동작 . 시작 값 및 첫 번째 중간점을 정의합니다 . 마스터 값 : 0, 추종 값 : 0
N120 X20 Y0	; 두 번째 중간점 마스터 값 : 0…20, 추종 값 : 시작값 …0
N130 X100 Y6	; 세 번째 중간점 : 마스터 값 : 20…100, 추종 값 : 0 ~ 6
N140 X150 Y6	; 네 번째 중간점 : 마스터 값 : 100…150, 추종 값 : 6 ~ 6
N150 X180 Y0	; 다섯 번째 중간점 : 마스터 값 : 150…180, 추종 값 : 6 ~ 0
N200 CTABEND	; 정의 끝 커브 테이블은 내부 표현식에서 최대 5 차까지의 폴리노미널로 생성됩니다 . 중간점이 지정된 커브 정의 계산은 모달로 선택된 보간 유형 (원 , 직선 , 스플라인 보간) 에 따라 달라집니다 . 정의를 시작하기 전 가공 프로그램 상태가 복원됩니다 .

예제 3: 주기적 커브 테이블 정의

번호가 2 인 주기적 커브 테이블의 정의 , 0 부터 360 까지 마스터 값 범위 , 0 부터 45 및 다시 0 으로 돌아가는 추종 축 이송 :

프로그램 코드	설명
N10 DEF REAL DEPPOS	
N20 DEF REAL GRADIENT	
N30 CTABDEF(Y,X,2,1)	; 정의 시작
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	; 정의 종료
;Y 를 X 에 커플링 하여 커브 테스트 :	
N120 G1 F1000 X0	
N130 LEADON(Y,X,2)	
N140 X360	
N150 X0	
N160 LEADOF(Y,X)	
N170 DEPPOS=CTAB(75.0,2,GRADIENT)	; 마스터 값 75.0 에 대한 테이블 기능 읽기
N180 G0 X75 Y=DEPPOS	; 리딩 및 추종 축의 위치를 다시 결정합니다 .
; 커플링 동작 후 , 추종 축을 동기화하지 않아도 됩니다 .	
N190 LEADON(Y,X,2)	
N200 G1 X110 F1000	
N210 LEADOF(Y,X)	
N220 M30	

추가 정보

커브 테이블의 시작값 및 종료값

커브 테이블의 정의의 범위 첫 부분에 사용되는 시작값은 커브 테이블 정의에서 지정된 최초의 관련 축 좌표 (최초의 이송 구문) 입니다 . 커브 테이블의 정의 범위에 대한 종료값은 최종 이송 명령에 따라 결정됩니다 .

사용 가능한 언어 범위

커브 테이블 정의에서 , 전체 NC 언어가 사용됩니다 .

주

커브 테이블 정의에서는 다음의 입력이 허용되지 않습니다 .

- 전처리 정지
- 리딩 축 동작에서의 점프 (예 : 변환 변경 시)
- 추종 축 전용 이송 구문
- 리딩 축 반전 (즉 리딩 축의 위치는 항상 고유해야 합니다 .)
- 다른 프로그램 레벨에서의 CTABDEF 및 CTABEND 동작

모달 동작의 효과

테이블 정의가 완료되면 커브 테이블 정의에서 작성된 모든 모달 구문이 적용되지 않습니다 . 따라서 테이블 정의 가공 프로그램이 만들어지기 이전 및 이후 테이블 정의 상태가 동일합니다 .

R 파라미터에 대한 지정

테이블 정의에서 R 파라미터에 대한 지정은 CTABEND 이후에 리셋됩니다 .

예제 :

프로그램 코드	설명
...	
R10=5 R11=20	; R10=5
...	
CTABDEF	
G1 X=10 Y=20 F1000	
R10=R11+5	; R10=25
X=R10	
CTABEND	
...	; R10=5

ASPLINE, BSPLINE, CSPLINE 활성화

ASPLINE, BSPLINE 또는 CSPLINE 이 커브 테이블 정의 CTABDEF ... CTABEND 에서 활성화되면 이 스플라인 활성화 전에 적어도 하나의 시작점을 프로그래밍해야 합니다. 그렇지 않으면 스플라인이 커브 테이블 정의 전에 현재 축 위치에 의존하므로 CTABDEF 직후의 활성화는 피해야 합니다 .

예제 :

프로그램 코드

```
...
CTABDEF(Y,X,1,0)
X0 Y0
ASPLINE
X=5 Y=10
X10 Y40
...
CTABEND
```

커브 테이블 반복 사용

커브 테이블을 사용하여 계산된 리딩 축과 추종 축 사이의 기능적인 관계는 테이블이 스태틱 NC 메모리 (SRAM) 에 저장된 경우 가공 프로그램 종료 및 전원 차단 후 선택된 테이블 번호에 준하여 유지됩니다 .

다이나믹 메모리 (DRAM) 에 생성된 테이블은 전원 ON 시 삭제되므로 재생성해야 할 수도 있습니다 .

생성된 커브 테이블은 리딩 축 및 추종 축으로 구성된 모든 축 조합에 적용될 수 있으며 커브 테이블을 생성하는 데 사용된 축과는 무관합니다 .

커브 테이블 덮어쓰기

특정 커브 테이블 번호가 다른 테이블 정의에 사용되면 즉시 그 커브 테이블을 덮어쓰게 됩니다 .

예외 : 커브 테이블이 축 커플링에서 활성화되거나 CTABLOCK 으로 잠금 설정되었을 때 .

주

커브 테이블을 덮어쓴 경우에는 경고가 출력되지 않습니다 .

커브 테이블 정의 활성 상태의 확인

언제든지 시스템 변수 $P_CTABDEF를 가공 프로그램에서 사용하여 커브 테이블 정의가 활성 상태인지 여부를 확인할 수 있습니다 .

커브 테이블 정의 취소

커브 테이블 정의와 관련된 동작이 제외되면 가공 프로그램 섹션을 실제 가공 프로그램으로 다시 사용할 수 있습니다 .

' 외부 소스에서 실행 ' 을 사용하여 커브 테이블 로딩

외부 소스로부터 커브 테이블을 실행하는 경우 전체 커브 테이블 정의를 리로드 버퍼에 동시에 저장하는 것을 지원하는 크기의 리로드 버퍼 (DRAM) 를
MD18360 $MN_MM_EXT_PROG_BUFFER_SIZE 에서 선택해야 합니다 . 그렇지 않은 경우에는 알람이 발생되어 가공 프로그램 처리가 취소됩니다 .

추종 축에서 점프

리딩 축에서 어떤 동작이 누락된 경우 머신 데이터
MD20900 $MC_CTAB_ENABLE_NO_LEADMOTION
의 설정에 따라 추종 축에서의 점프가 묵인될 수도 있습니다 .

9.2.2 커브 테이블의 존재 검사 (CTABEXISTS)

기능

CTABEXISTS 명령은 특정 커브 테이블 번호가 NC 메모리에 있는지 여부를 검사하는 데 사용할 수 있습니다 .

구문

CTABEXISTS(<n>)

의미

CTABEXISTS	스태틱 또는 다이나믹 NC 메모리에 커브 테이블 번호 <n> 이 있는지 검사합니다 . 0 테이블이 없음 1 테이블이 있음
<n>	커브 테이블 번호 (ID)

9.2.3 커브 테이블 삭제 (CTABDEL)

기능

CTABDEL 을 사용하여 커브 테이블을 삭제할 수 있습니다 .

주

축 커플링에 사용 중인 커브 테이블은 삭제할 수 없습니다 .

구문

CTABDEL(<n>)
CTABDEL(<n>,<m>)
CTABDEL(<n>,<m>,< 메모리 위치 >)
CTABDEL ()
CTABDEL(,,< 메모리 위치 >)

의미

CTABDEL	커브 테이블 삭제 명령
<n>	삭제할 커브 테이블 번호 (**ID**) 커브 테이블 범위 CTABDEL(<n>,<m>) 을 삭제할 때 <n> 은 그 범위에서 첫 번째 커브 테이블 번호를 지정하는 데 사용됩니다 .
<m>	커브 테이블 범위 CTABDEL(<n>,<m>) 를 삭제할 때 <m> 은 그 범위에서 마지막 커브 테이블 번호를 지정하는 데 사용됩니다 . <m> 은 <n> 보다 커야 합니다 .
< 메모리 위치 >	메모리 위치 지정 (옵션) 메모리 위치를 **지정하지 않고** 삭제하는 경우에는 스태틱 및 다이나믹 NC 메모리에서 지정된 커브 테이블이 삭제됩니다 . 메모리 위치를 **지정하고** 삭제하는 경우에는 지정된 커브 테이블 중에서 지정된 메모리 위치에 있는 테이블만 삭제됩니다 . 나머지는 그대로 유지됩니다 . 'SRAM' **스태틱** NC 메모리에서 삭제 'DRAM' **다이나믹** NC 메모리에서 삭제

삭제할 커브 테이블을 지정하지 않고 CTABDEL 을 프로그래밍하면 **전체** 커브 테이블 또는 지정된 메모리의 모든 커브 테이블이 삭제됩니다 .

CTABDEL ()	스태틱 및 다이나믹 NC 메모리의 모든 커브 테이블 삭제
CTABDEL(,,"SRAM")	스태틱 NC 메모리의 모든 커브 테이블 삭제
CTABDEL(,,"DRAM")	다이나믹 NC 메모리의 모든 커브 테이블 삭제

주

CTABDEL(<n>,<m>) 또는 CTABDEL() 을 사용하여 여러 개를 삭제하는 경우 삭제할 커브 테이블 중 적어도 하나가 커플링에서 활성 상태이면 삭제 명령이 실행되지 않습니다 . 즉 , 지정된 커브 테이블 중 그 어느 것도 삭제되지 **않습니다** .

9.2.4 삭제 및 덮어쓰기 방지를 위한 커브 테이블 잠금 (CTABLOCK, CTABUNLOCK)

기능

의도하지 않은 커브 테이블 삭제 및 덮어쓰기를 방지하기 위해 잠금을 설정할 수 있습니다 . 잠금을 설정한 후에는 언제든지 취소할 수 있습니다 .

구문

잠금 :
CTABLOCK(<n>)
CTABLOCK(<n>,<m>)
CTABLOCK(<n>,<m>,< 메모리 위치 >)
CTABLOCK()
CTABLOCK(,,< 메모리 위치 >)

잠금 해제 :
CTABUNLOCK(<n>)
CTABUNLOCK(<n>,<m>)
CTABUNLOCK(<n>,<m>,< 메모리 위치 >)
CTABUNLOCK()
CTABUNLOCK(,,< 메모리 위치 >)

의미

CTABLOCK	삭제 / 덮어쓰기 방지를 위한 잠금 **설정** 명령
CTABUNLOCK	삭제 / 덮어쓰기 방지를 위한 잠금 **취소** 명령 CTABUNLOCK 은 CTABLOCK 을 통해 잠긴 커브 테이블을 해제합니다 . 활성 커플링과 관련된 커브 테이블은 잠금 상태가 유지되며 삭제가 불가능합니다. 활성 커플링으로 인한 잠금이 커플링 비활성화로 인해 해제되면 CTABLOCK 를 통한 잠금은 즉시 해제됩니다 . 그러므로 이 테이블을 삭제할 수 있습니다 . CTABUNLOCK 를 다시 호출할 필요는 없습니다 .
<n>	잠금 / 해제할 커브 테이블 번호 (**ID**) 커브 테이블 범위 CTABLOCK(<n>,<m>)/CTABUNLOCK(<n>,<m>) 를 잠금 / 해제할 때 <n> 은 그 범위에서 첫 번째 커브 테이블 번호를 지정하는 데 사용됩니다 .

<m>	커브 테이블 범위 CTABLOCK(<n>,<m>)/CTABUNLOCK(<n>,<m>) 를 잠금 / 해제할 때 <m> 은 그 범위에서 마지막 커브 테이블 번호를 지정하는 데 사용됩니다 . <m> 은 <n> 보다 커야 합니다 .
< 메모리 위치 >	메모리 위치 지정 (옵션) 메모리 위치를 **지정하지 않고** 잠금 / 해제하는 경우에는 스태틱 및 다이나믹 NC 메모리에서 지정된 커브 테이블이 잠금 / 해제됩니다 . 메모리 위치를 **지정하고** 잠금/해제하는 경우에는 지정된 커브 테이블 중에서 지정된 메모리 위치에 있는 테이블만 잠금 / 해제됩니다 . 나머지는 잠금 / 해제되지 않습니다 . 'SRAM' **스태틱** NC 메모리에서 잠금 / 해제 'DRAM' **다이나믹** NC 메모리에서 잠금 / 해제

잠금 / 해제할 커브 테이블을 지정하지 않고 CTABLOCK/CTABUNLOCK 를 프로그래밍하면 **전체** 커브 테이블 또는 지정된 메모리의 모든 커브 테이블이 잠금 / 해제됩니다 .

CTABLOCK()	스태틱 및 다이나믹 NC 메모리의 모든 커브 테이블 잠금
CTABLOCK(,,"SRAM")	스태틱 NC 메모리의 모든 커브 테이블 잠금
CTABLOCK(,,"DRAM")	다이나믹 NC 메모리의 모든 커브 테이블 잠금
CTABUNLOCK()	스태틱 및 다이나믹 NC 메모리의 모든 커브 테이블 잠금 해제
CTABUNLOCK(,,"SRAM")	스태틱 NC 메모리의 모든 커브 테이블 잠금 해제
CTABUNLOCK(,,"DRAM")	다이나믹 NC 메모리의 모든 커브 테이블 잠금 해제

9.2.5 커브 테이블 : 테이블 속성 결정 (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD)

기능

이 명령은 커브 테이블의 중요한 속성 (테이블 번호 , 잠금 상태 , 메모리 위치 , 주기성) 을 조사하는 데 사용할 수 있습니다 .

구문

CTABID(<p>)
CTABID(<p>,< 메모리 위치 >)
CTABISLOCK(<n>)
CTABMEMTYP(<n>)
TABPERIOD(<n>)

의미

CTABID	지정된 메모리에서 <p> 번째 커브 테이블로 입력한 **테이블 번호**를 보고합니다 . 예제 : CTABID(1,"SRAM") 는 스태틱 NC 메모리의 첫 번째 커브 테이블 번호를 보고합니다 . 이 상황에서 첫 번째 커브 테이블이란 테이블 번호가 가장 높은 커브 테이블을 의미합니다 . **참고 :** 가령 CTABDEL 을 사용한 커브 테이블 삭제로 인해 CTABID 의 연속적인 호출 사이에서 메모리의 커브 테이블 순서가 달라지면 , CTABID(<p>,...) 는 동일한 번호 <p> 를 가진 다른 커브 테이블을 보고할 수도 있습니다 .
CTABISLOCK	커브 테이블 번호 <n> 의 **잠금 상태**를 보고합니다 . 0 테이블이 잠기지 않음 1 테이블이 CTABLOCK 에 의해 잠겨 있음 2 테이블이 활성 커플링에 의해 잠겨 있음 3 테이블이 CTABLOCK 및 활성 커플링에 의해 잠겨 있음 -1 테이블이 없음
CTABMEMTYP	커브 테이블 번호 <n> 의 **메모리 위치**를 보고합니다 . 0 스태틱 NC 메모리의 테이블 1 다이나믹 NC 메모리의 테이블 -1 테이블이 없음
CTABPERIOD	커브 테이블 번호 <n> 의 **주기성**을 보고합니다 . 0 커브 테이블이 주기적이지 않음 1 테이블이 리딩 축에서 주기적임 2 테이블이 리딩 축 및 추종 축에서 주기적임 -1 테이블이 없음
<p>	메모리에서의 입력 번호
<n>	커브 테이블 번호 (**ID**)
< 메모리 위치 >	메모리 위치 지정 (옵션) 'SRAM' **스태틱** NC 메모리 'DRAM' **다이나믹** NC 메모리 **참고 :** 이 파라미터에 대한 값을 프로그래밍하지 않으면 MD20905 $MC_CTAB_DEFAULT_MEMORY_TYPE 을 통해 설정된 기본 메모리 위치가 사용됩니다 .

9.2.6 커브 테이블 값 읽기 (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX)

기능

다음의 커브 테이블을 가공 프로그램에서 읽을 수 있습니다 .

- 커브 테이블의 시작과 끝에 있는 추종 축 및 리딩 축 값
- 커브 분할의 시작과 끝에 있는 추종 축 값
- 리딩 축 값에 대한 추종 축 값
- 추종 축 값에 대한 리딩 축 값
- 추종 축의 최소값 및 최대값
  - 커브 테이블의 전체 정의 범위 이내

    또는
  - 정의된 커브 테이블 간격 이내

구문

CTABTSV(<n>,< 경사도 >[,< 추종 축 >])
CTABTEV(<n>,< 경사도 >[,< 추종 축 >])
CTABTSP(<n>,< 경사도 >[,< 리딩 축 >])
CTABTEP(<n>,< 경사도 >[,< 리딩 축 >])
CTABSSV(< 마스터 값 >,<n>,< 경사도 >[,< 추종 축 >])
CTABSEV(< 마스터 값 >,<n>,< 경사도 >[,< 추종 축 >])
CTAB(< 마스터 값 >,<n>,< 경사도 >[,< 추종 축 >],< 리딩 축 >])
CTABINV(< 추종 값 >,< 근사치 >,<n>,< 경사도 >[,< 추종 축 >,< 리딩 축 >]
CTABTMIN(<n>[,< 추종 축 >])
CTABTMAX(<n>[,< 추종 축 >])
CTABTMIN(<n>,<a>,<b>[,< 추종 축 >,< 리딩 축 >])
CTABTMAX(<n>,<a>,<b>[,< 추종 축 >,< 리딩 축 >])

의미

CTABTSV:	커브 테이블 번호 <n> 의 **시작**에서 추종 축 값 읽기
CTABTEV:	커브 테이블 번호 <n> 의 **끝**에서 추종 축 값 읽기
CTABTSP:	커브 테이블 번호 <n> 의 **시작**에서 리딩 축 값 읽기
CTABTEP:	커브 테이블 번호 <n> 의 **끝**에서 리딩 축 값 읽기
CTABSSV:	지정된 리딩 축 값 (< 마스터 값 >) 에 속하는 커브 분할의 **시작**에서 추종 축 값 읽기
CTABSEV:	지정된 리딩 축 값 (< 마스터 값 >) 에 속하는 커브 분할의 **끝**에서 추종 축 값 읽기
CTAB:	지정된 리딩 축 값 (< 마스터 값 >) 에 대한 추종 축 값 읽기
CTABINV:	지정된 추종 축 값 (< 추종 값 >) 에 대한 리딩 축 값 읽기

CTABTMIN:	추종 축 **최소값** 정의 : • 커브 테이블의 전체 정의 범위 이내 또는 • 정의된 간격 <a> 에서 <b> 사이
CTABTMAX:	추종 축 **최대값** 정의 : • 커브 테이블의 전체 정의 범위 이내 또는 • 정의된 간격 <a> 에서 <b> 사이
<n>:	커브 테이블 번호 (**ID**)
< 경사도 >:	< 경사도 > 파라미터는 계산된 위치에서 커브 테이블 기능의 **경사도**를 반환합니다 .
< 추종 축 >:	커브 테이블을 사용하여 모션을 계산할 축 (옵션)
< 리딩 축 >:	추종 축 모션 계산을 위한 마스터 값을 제공하는 축 (옵션)
< 추종 값 >:	CTABINV 에 대한 리딩 축의 관련 값을 읽는 추종 축의 값
< 리딩 값 >:	리딩 축 값 : • 연관된 추종 축 값을 CTAB 로 읽을 시 또는 • CTABSSV/CTABSEV 를 통한 커브 분할을 선택할 시
< 근사치 >:	CTABINV 를 사용하여 추종 축 값에 지정하는 리딩 축 값이 반드시 고유한 값일 필요는 없습니다 . 따라서 CTABINV 은 예상되는 리딩 축 값에 대한 근사치를 파라미터로 요구합니다 .
<a>:	CTABTMIN/CTABTMAX 를 통한 마스터 값 간격의 **하한값**
<b>:	CTABTMIN/CTABTMAX 를 통한 마스터 값 간격의 **상한값**
	참고 : <a> 에서 <b> 사이의 마스터 값 간격은 반드시 커브 테이블의 정의 범위 이내여야 합니다 .

예제

예제 1:

커브 테이블의 전체 정의 범위에 포함되는 추종 축의 최소값 및 최대값과 함께 커브 테이블의 시작과 끝에서 추종 축 및 리딩 축의 값을 정의합니다 .

프로그램 코드	설명
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)	; 테이블 정의 시작
N110 X0 Y10	; 첫 번째 테이블 분할 시작 위치
N120 X30 Y40	; 첫 번째 테이블 분할 종료 위치 = 두 번째 테이블 분할 시작 위치
N130 X60 Y5	; 두 번째 테이블 분할 종료 위치 = 기타
N140 X70 Y30	
N150 X80 Y20	
N160 CTABEND	; 테이블 정의 종료
...	
N200 STARTPOS=CTABTSV(1,GRADIENT)	; 커브 테이블 시작 부분의 추종 축 값 = 10
N210 ENDPOS=CTABTEV(1,GRADIENT)	; 커브 테이블 종료 부분의 추종 축 값 = 20
N220 STARTPARA=CTABTSP(1,GRADIENT)	; 커브 테이블 시작 부분의 리딩 축 값 = 0
N230 ENDPARA=CTABTEP(1,GRADIENT)	; 커브 테이블 종료 부분의 리딩 축 값 = 80
N240 MINVAL=CTABTMIN(1)	; Y=5 인 추종 축의 최소값
N250 MAXVAL=CTABTMAX(1)	; Y=40 인 추종 축의 최대값

예제 2:

리딩 축 값 X=30 과 관련된 커브 분할의 시작 및 종료 부분에서 추종 축 값 결정

프로그램 코드	설명
N10 DEF REAL STARTPOS	
N20 DEF REAL ENDPOS	
N30 DEF REAL GRADIENT	
...	
N100 CTABDEF(Y,X,1,0)	; 테이블 정의 시작
N110 X0 Y0	; 첫 번째 테이블 분할 시작 위치
N120 X20 Y10	; 첫 번째 테이블 분할 종료 위치 = 두 번째 테이블 분할 시작 위치
N130 X40 Y40	두 번째 테이블 분할 종료 위치 = 기타
N140 X60 Y10	
N150 X80 Y0	
N160 CTABEND	; 테이블 정의 종료
...	
N200 STARTPOS=CTABSSV(30.0,1,GRADIENT)	; 두 번째 분할의 시작 위치 Y = 10
N210 ENDPOS=CTABSEV(30.0,1,GRADIENT)	; 두 번째 분할의 종료 위치 Y = 40

추가 정보

동기 동작에서 사용

커브 테이블 값 읽기를 위한 모든 명령을 동기 동작에서도 사용할 수 있습니다 (' 모션 동기 동작 ' 장 참조).

CTABINV, CTABTMIN 및 CTABTMAX 명령을 사용할 때는 다음 사항을 명확하게 해야 합니다 .

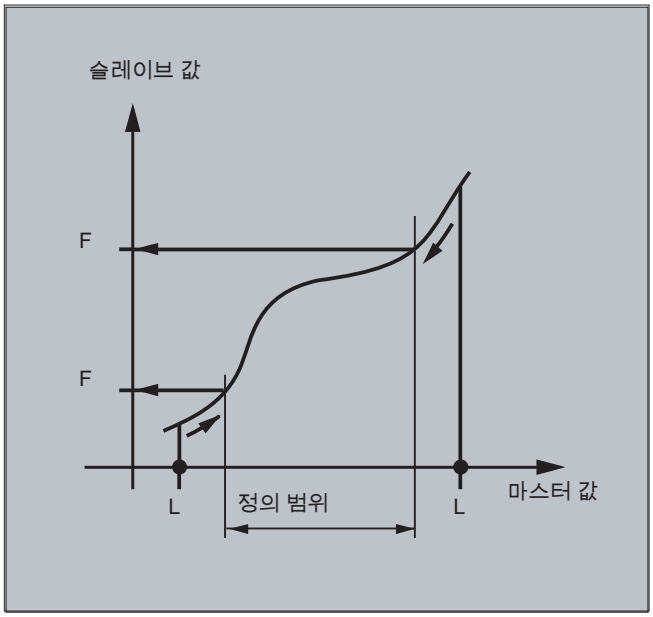
- 실행 시 사용할 NC 전력이 충분한지

  또는

- 필요하다면 관련 테이블을 분할할 수 있도록 커브 테이블의 분할 수가 호출 전에 조회되는지

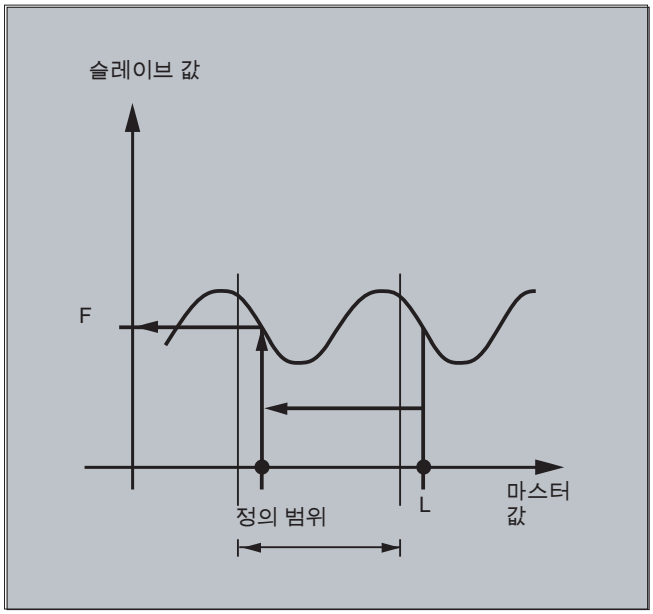
비 주기적 커브 테이블을 포함한 CTAB

지정된 < 마스터 값 > 이 정의 범위를 벗어나는 경우 상한값 또는 하한값은 추종 값으로 출력됩니다 .



주기적 커브 테이블을 포함한 CTAB

지정된 < 마스터 값 > 이 정의 범위를 벗어나는 경우 정의 범위의 모듈로를 기준으로 마스터 값이 산정되고 그에 상응하는 추종 값이 출력됩니다 .



CTABINV 에 대한 근사값

따라서 CTABINV 명령은 예상 마스터 값의 근사값을 요구합니다 . CTABINV 는 근사치에 가장 가까운 마스터 값을 반환합니다 . 예를 들어 이전의 보간 사이클에서 도출한 마스터 값이 근사치로 결정될 수 있습니다 .

커브 테이블 기능의 경사

경사 (< 경사도 >) 출력을 통해 해당 위치에서 리딩 축 또는 추종 축의 속도를 계산할 수 있습니다 .

리딩 축 또는 추종 축 지정

리딩 축과 추종 축이 다른 길이 단위로 설정된 경우 리딩 축 및 / 또는 추종 축의 옵션 지정이 중요합니다 .

CTABSSV, CTABSEV

아래와 같은 경우 CTABSSV 및 CTABSEV 명령은 프로그래밍된 분할을 조회하는 데 적합하지 **않습니다** .

- 원호 또는 나선형이 프로그래밍된 경우
- CHF/RND 를 이용한 모따기나 라운딩이 활성 상태인 경우
- G643 을 이용한 스무딩이 활성 상태인 경우
- COMPON/COMPCURV/COMPCAD 를 이용한 NC 블록 압축이 활성 상태인 경우

9.2.7 커브 테이블 : 리소스 사용 확인 (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL)

기능

프로그래머는 이 명령을 사용하여 커브 테이블 , 테이블 분할 및 폴리노미널을 위한 리소스 사용에 대한 최신 정보를 구할 수 있습니다 .

구문

CTABNO
CTABNOMEM(< 메모리 위치 >)
CTABFNO(< 메모리 위치 >)
CTABSEGID(<n>,< 메모리 위치 >)
CTABSEG(< 메모리 위치 >,< 분할 유형 >)
CTABFSEG(< 메모리 위치 >,< 분할 유형 >)
CTABMSEG(< 메모리 위치 >,< 분할 유형 >)
CTABPOLID(<n>)
CTABPOL(< 메모리 위치 >)
CTABFPOL(< 메모리 위치 >)
CTABMPOL(< 메모리 위치 >)

의미

CTABNO	**정의된** 커브 테이블의 전체 개수 확인 (스태틱 및 다이나믹 NC 메모리에서)
CTABNOMEM	지정된 < 메모리 위치 > 에서 **정의된** 커브 테이블의 개수 확인
CTABFNO	지정된 < 메모리 위치 > 에서 **남아 있을 수 있는** 커브 테이블의 개수 확인
CTABSEGID	커브 테이블 번호 <n>에서 사용하는 지정된 <분할 유형>의 커브 분할 개수 확인
CTABSEG	지정된 <메모리 위치>에서 **사용되는** 지정된 <분할 유형>의 커브 분할 개수 확인
CTABFSEG	지정된 < 메모리 위치 > 에 **남아 있을 수 있는** 지정된 < 분할 유형 > 의 커브 분할 개수 확인
CTABMSEG	지정된 < 메모리 위치 > 에서 지정된 < 분할 유형 > 의 **가능한 최대** 커브 분할 개수 확인
CTABPOLID	커브 테이블 번호 <n> 에서 사용하는 커브 폴리노미널의 개수 확인
CTABPOL	지정된 < 메모리 위치 > 에서 **사용된** 커브 폴리노미널의 개수 확인
CTABFPOL	지정된 < 메모리 위치 > 에서 **남아 있을 수 있는** 커브 폴리노미널의 개수 확인
CTABMPOL	지정된 < 메모리 위치 > 에 남아 있을 수 있는 **최대** 커브 폴리노미널의 개수 확인
<n>	커브 테이블 번호 (**ID**)

< 메모리 위치 > 메모리 위치 지정 (옵션)

'SRAM' **스태틱** NC 메모리

'DRAM' **다이나믹** NC 메모리

참고 :
이 파라미터에 대한 값을 프로그래밍하지 않으면 MD20905 $MC_CTAB_DEFAULT_MEMORY_TYPE 을 통해 설정된 기본 메모리 위치가 사용됩니다 .

< 분할 유형 > 분할 유형 지정 (옵션)

'L' 직선 분할

'P' 폴리노미널 분할

참고 :
이 파라미터에 대한 값을 프로그래밍하지 않으면 직선 분할 및 폴리노미널 분할의 합계가 출력됩니다 .

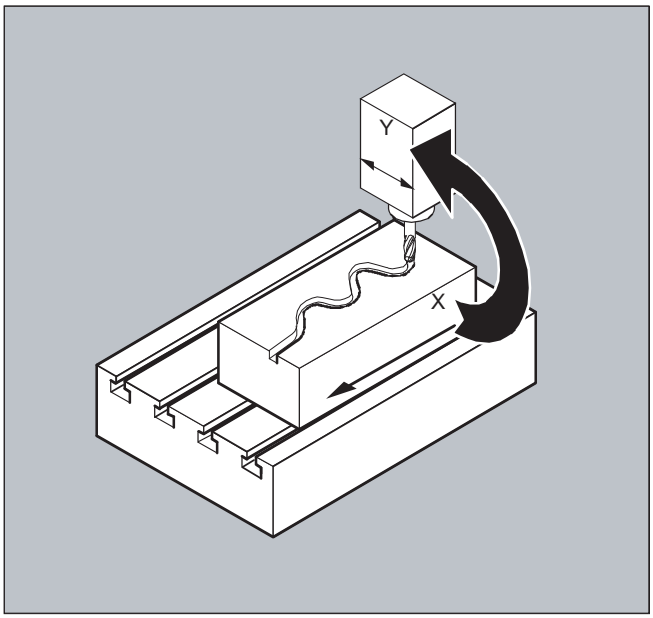
9.3 축 관련 리딩 값 결합 (LEADON, LEADOF)

주

이 기능은 SINUMERIK 828D 에 사용할 수 없습니다 !

기능

축 관련 커플링을 사용한 경우, 커플링을 사용한 후, 리딩 축과 추종 축이 동기되어 이동합니다. 필요한 경우, 커브 테이블이나 그로 인해 얻은 폴리노미널을 사용한 시뮬레이션으로 리딩 축의 위치에 추종 축의 위치를 할당할 수 있습니다 .



리딩 축은 커브 테이블에 입력값을 제공하는 축입니다 . **추종 축**은 커브 테이블에 의해 계산된 위치를 차지하는 축입니다 .

실제값과 지령치 커플링

아래 항목을 마스터 값 (추종 축의 위치 산정용 출력 값) 으로 사용할 수 있습니다 .

- 리딩 축 위치의 실제값 : 실제값 커플링
- 리딩 축 위치의 지령치 : 지령치 값 커플링

기본 좌표계에는 항상 마스터 값 커플링이 적용됩니다 .

커브 테이블 작성 방법에 대해서는 ' 커브 테이블 ' 단원을 참조하십시오 .
마스터 값 커플링에 대해서는 /FB/, M3, 커플 축 동작 및 마스터 값 커플링을 참조하십시오 .

구문

LEADON(FAxis,LAxis,n)
LEADOF(FAxis,LAxis)

또는 리딩 축을 지정하지 않은 채 비활성화 :
LEADOF(FAxis)

동기 동작의 이동 중이나 가공 프로그램에서 마스터 값 커플링을 활성화 또는 비활성화할 수 있습니다 . ' 동기 동작 ' 을 참조하십시오 .

의미

LEADON	마스터 값 커플링 활성화
LEADOF	마스터 값 커플링 해제
Faxis	추종 축
Laxis	리딩 축
n	커브 테이블 번호
$SA_LEAD_TYPE	지령치 커플링 및 실제치 커플링 간의 전환

마스터 값 커플링 해제 , LEADOF

마스터 값 커플링을 해제하면 추종 축이 다시 일반 지령 축으로 전환됩니다 !

축 관련 마스터 값 커플링 및 다양한 동작 상태 , RESET

머신 데이터의 설정 상태에 따라 RESET 을 누르면 마스터 값 커플링을 해제하기도 합니다 .

동기 동작에서 마스터 값 커플링 예제

프레스 공장에서 , 이송 축 및 보조 축으로 구성된 이송 시스템 축과 리딩 축 (Stanchion Shaft) 간의 일반적인 기계적 커플링을 전기적 커플링으로 대체할 수 있습니다 .

기계적 이송 시스템을 전기적 이송 시스템으로 대체하는 방법은 아래와 같습니다 . 커플링 및 커플링 해제 과정은 **스태틱 동기 동작**으로 구현됩니다 .

리딩 축 LV(Stanchion Shaft) 에서 이송 축과 보조 축은 커브 테이블을 통해 정의된 추종 축처럼 제어됩니다 .

추종 축

X 피드 또는 마스터 값 축
YL 폐쇄 또는 이송 축
ZL 리프트 축
U 롤러 이송 , 보조 축
V 가이드 헤드 , 보조 축
W 윤활유 공급 , 보조 축

동작

발생하는 동작에는 다음과 같은 동기 동작이 포함됩니다 .

- 커플링 활성화 , LEADON(추종 축 , 리딩 축 , 커브 테이블 번호)
- 커플링 해제 , LEADOF(추종 축 , 리딩 축)
- 실제값 설정 , PRESETON(축 , 값)
- 마커 설정 , $AC_MARKER[i]= 값
- 커플링 형식 : 실제 / 가상 마스터 값
- 축 위치 접근 , POS[축]= 값

조건

부울 (Boolean) 연산자 'AND' 를 사용하여 고속 디지털 입력 , 실시간 변수 $AC_MARKER, 그리고 위치 비교를 조건으로 연결합니다 .

주

아래 예제에서는 단지 프로그램의 가독성 향상을 위해 라인 변경 , 들여쓰기 및 **고딕체**가 사용됐습니다 . 따라서 특정 라인 번호 이후에 오는 모든 내용들이 실제로는 한 라인을 구성합니다 .

설명

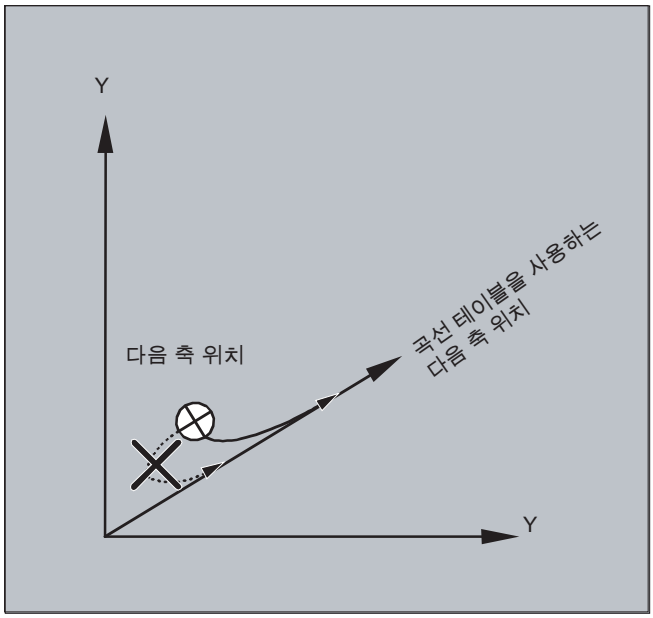
프로그램 코드	설명
	; 모든 스태틱 동기 동작 정의
	; **** 리셋 마커
N2 $AC_MARKER[0]=0 $AC_MARKER[1]=0 $AC_MARKER[2]=0 $AC_MARKER[3]=0 $AC_MARKER[4]=0 $AC_MARKER[5]=0 $AC_MARKER[6]=0 $AC_MARKER[7]=0	
	; **** E1 0=>1 전송 ON
N10 IDS=1 EVERY ($A_IN[1]==1) AND ($A_IN[16]==1) AND ($AC_MARKER[0]==0) DO LEADON(X,LW,1) LEADON(YL,LW,2) LEADON(ZL,LW,3) $AC_MARKER[0]=1	
	; **** E1 0=>1 롤러 이송 커플링 ON
N20 IDS=11 EVERY ($A_IN[1]==1) AND ($A_IN[5]==0) AND ($AC_MARKER[5]==0) DO LEADON(U,LW,4) PRESETON(U,0) $AC_MARKER[5]=1	
	; **** E1 0->1 정렬 헤드 커플링 ON
N21 IDS=12 EVERY ($A_IN[1]==1) AND ($A_IN[5]==0) AND ($AC_MARKER[6]==0) DO LEADON(V,LW,4) PRESETON(V,0) $AC_MARKER[6]=1	
	; **** E1 0->1 윤활 커플링 ON
N22 IDS=13 EVERY ($A_IN[1]==1) AND ($A_IN[5]==0) AND ($AC_MARKER[7]==0) DO LEADON(W,LW,4) PRESETON(W,0) $AC_MARKER[7]=1	
	; **** E2 0=>1 커플링 OFF
N30 IDS=3 EVERY ($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	

설명

마스터 값 결합에는 리딩 축과 추종 축의 동기화가 수반돼야 합니다 . 마스터 값 결합이 활성 상태일 때 추종 축이 곡선 테이블에서 산정한 곡선 정의의 공차 범위를 벗어나지 않은 경우에 한하여 동기화가 가능합니다 .

추종 축의 위치에 대한 공차 범위는 기계 데이터 MD 37200: COUPLE_POS_POL_COARSE A_LEAD_TYPE 을 통해 정의됩니다 ..

마스터 값 결합이 활성화된 상태에서 추종 축이 아직 올바른 위치에 있지 않은 경우 , 추종 축에 대해 산정된 위치 지령치 값이 실제 추종 축의 위치에 근접한 즉시 , 동기화 실행이 자동으로 초기화됩니다 . 동기화 절차가 진행되는 동안 , 추종 축은 추종 축의 지령치 속도 (마스터 스핀들에서 산정 및 CTAB 곡선 테이블 사용) 에 의해 정의된 방향으로 이송됩니다 .

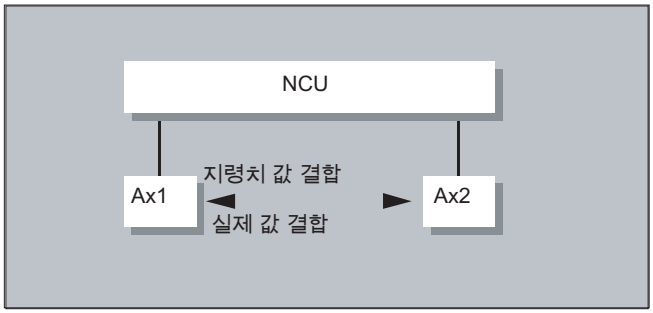


동기화 불가

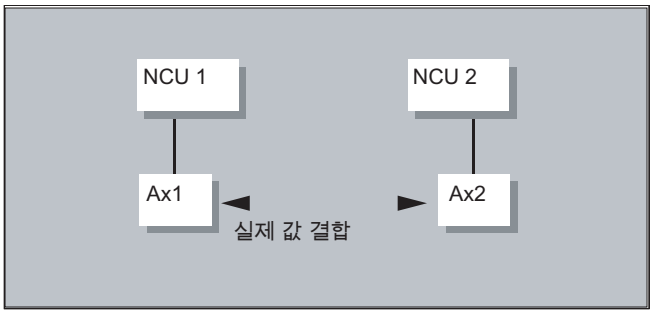
마스터 값 결합이 활성 상태일 때 산정된 추종 축 위치가 현재 추종 축 위치를 벗어난 경우 , 동기화를 실행할 수 없습니다 .

실제값과 지령치 결합

지령치 결합은 실제값 결합에 비해 리딩 축과 추종 축 동기화에 더 효과적이므로 지령치 결합이 기본값으로 설정됩니다 .



리딩 축과 추종 축이 동일한 NCU 에 의해 보간되는 경우에만 지령치 결합이 가능합니다 . 외부 리딩 축을 사용하는 경우 , 추종 축은 실제값을 통해서만 리딩 축과 결합될 수 있습니다 .



전환은 설정 데이타 $SA_LEAD_TYPE 을 통해 프로그래밍할 수 있습니다 .

실제값과 지령치 결합간의 전환은 반드시 추종 축이 정지했을 때 이뤄져야 합니다 . 축이 정지하고 전환을 마친 후에 한하여 동기화가 가능합니다 .

용도

기계가 심하게 진동하는 동안에 실제값을 판독하면 오류가 발생합니다. 프레스 이송에 마스터 값 결합을 사용하는 경우 , 상황에 따라 진동이 가장 극심한 작업 단계에 실제값 결합에서 지령치 결합으로 전환할 필요가 있습니다 .

지령치 결함을 이용한 **마스터 값 시뮬레이션**

기계 데이터를 사용해서 서보에서 리딩 축 보간자의 연결을 해제할 수 있습니다 . 이런 방법으로 실제로는 리딩 축의 위치를 변경하지 않고서도 지령치 결합용 지령치를 생성할 수 있습니다 .

동기화 동작 등에 사용할 목적으로 다음 변수에서 지령치 연결로 생성된 마스터 값을 판독할 수 있습니다 .

- $AA_LEAD_P	마스터 값 위치
- $AA_LEAD_V	마스터 값 속도

마스터 값 생성

다른 자체 프로그램으로 마스터 값을 생성할 수 있는 옵션이 있습니다 . 이런 방법으로 생성된 마스터 값은 변수에 기록하거나 변수에서 읽어올 수 있습니다 .

- $AA_LEAD_SP	마스터 값 위치
- $AA_LEAD_SV	마스터 값 속도

이 변수들을 사용하려면 먼저 설정 데이터 $SA_LEAD_TYPE = 2를 반드시 설정해야 합니다.

결합 상태

다음과 같은 시스템 변수를 사용하여 NC 프로그램의 결합 상태를 조회할 수 있습니다 .

$AA_COUP_ACT[[axis]]
0: 활성화된 결합 없음
16: 마스터 값 결합 활성화됨

동기화 동작의 상태 관리

전환 및 결합 이벤트는 다음과 같은 실시간 변수로 관리됩니다 .

$AC_MARKER[i] = n
관리 도구 :
i 플래그 번호
n 상태값

9.4 전자 기어 (EG)

기능

' 전자 기어 ' 기능을 사용하면 선형 이송 블록에 따라 **추종 축** 동작을 최대 5 개의 **리딩 축** 기능처럼 관리할 수 있습니다 . 각 리딩 축과 추종 축의 관계는 결합 팩터로 정의됩니다 .

추종 축 동작부는 모든 리딩 축 동작부에 각각 해당 결합 팩터를 곱한 다음 가산해서 산정됩니다 전자 기어 축 그룹화가 활성화되면 지정된 위치와 관련한 추종 축을 동기화할 수 있습니다 . 기어 그룹은 다음 작업이 가능합니다 .

- 정의 ,
- 활성화 ,
- 비활성화 ,
- 삭제 .

추종 축 동작은 다음 요소에서 선택적으로 도출할 수 있습니다 .

- 리딩 축의 지령치 및
- 리딩 축의 실제값

각 리딩 축과 추종 축간의 비형선적 관계 역시 **곡선 테이블**을 사용하여 확장 용도로 구현할 수도 있습니다 (' 경로 이송 특성 ' 단원 참조). 전자 기어는 계단식 (Cascade) 으로 구성할 수 있습니다 . 다시 말해서 , 전자 기어의 추종 축이 전자 기어 추가용 리딩 축 역할을 할 수 있습니다 .

9.4.1 전자 기어 정의 (EGDEF)

기능

전자 기어의 축 그룹은 추종 축 및 관련 결합 형식이 존재하는 5 개 미만의 리딩 축을 지정함으로써 정의됩니다 .

요구 사항

전자 기어 축 그룹 정의 전제 조건

추종 축에 대한 축 결합을 정의하는 것은 허용되지 않습니다 (또는 EGDEL 을 사용하여 기존의 축 결합을 먼저 삭제해야 합니다).

구문

EGDEF(추종 축 , 리딩 축 1, 결합 형식 1, 리딩 축 2, 결합 형식 2,...)

의미

EGDEF	전자 기어 정의
추종 축	리딩 축의 영향을 받는 축
리딩 축 1 ,..., 리딩 축 5	추종 축에 영향을 미치는 축
결합 형식 1 ,..., 결합 형식 5	결합 형식 모든 리딩 축에 대한 결합 형식이 반드시 동일할 필요는 없으며 각각의 마스터 축을 기준으로 별도로 프로그래밍되어야 합니다 . **값 :** **의미 :** 0 추종 축이 해당 리딩 축의 **실제값**의 영향을 받습니다 . 1 추종 축이 해당 리딩 축의 **지령치**의 영향을 받습니다 .

주

전자 기어 축 그룹화가 정의되면 결합 팩터가 0 으로 사전 설정됩니다 .

주

EGDEF를 사용하면 사전 처리 중지 기능이 실행됩니다. 하나 이상의 리딩 축이 **곡선 테이블**을 통해 추종 축에 영향을 미치는 경우 , EGDEF 를 이용한 기어박스 정의를 수정하지 않은 채 사용해야 합니다 .

예제

프로그램 코드	비고
EGDEF(C,B,1,Z,1,Y,1)	; 기어박스 축 그룹 정의 리딩 축 B, Z, Y 가 지령치를 통해 추종 축 C 에 영향을 미칩니다 .

9.4.2 전자 기어박스 활성화 (EGON, EGONSYN, EGONSYNE)

기능

전자 기어 축 그룹을 활성화하는 방법으로는 세 가지가 있습니다 .

구문

변용 1:

다음 구문으로 동기화하지 않은 채 전자 기어 축 그룹을 선택적으로 활성화합니다 .
EGON(FA,"block change mode",LA1,Z1,N1,LA2,Z2,N2,...,LA5,Z5,N5)

변용 2:

다음 구문으로 동기화한 채 전자 기어 축 그룹을 선택적으로 활성화합니다 .
EGONSYN(FA,"block change mode",SynPosFA,[,LAi,SynPosLAi,Zi,Ni])

변용 3:

다음 구문으로 접근 모드를 지정하고 동기화한 채 전자 기어 축 그룹을 선택적으로 활성화합니다 .
EGONSYNE(FA,"block change mode",SynPosFA,approach mode[,LAi,SynPosLAi,Zi,Ni])

의미

버전 1:

FA	추종 축	
블록 변경 모드	다음과 같은 모드를 사용할 수 있습니다 .	
	"NOC"	즉시 블록 변경이 이뤄집니다 .
	"FINE"	블록 변경이 ' 미세조정 동기화 ' 로 수행됩니다 .
	"COARSE"	블록 변경이 ' 일반 동기화 ' 로 수행됩니다 .
	"IPOSTOP"	지령치 기반 동기화를 위해 블록 변경이 수행됩니다 .
LA1, ... LA5	리딩 축	
Z1, ... Z5	결합 팩터 i 의 카운터	
N1, ... N5	결합 팩터 i 의 분모	
	결합 팩터 i = 카운터 i/ 분모 i	

EGDEF 명령으로 미리 지정한 리딩 축에 한하여 활성화 라인에 프로그래밍할 수 있습니다 .
최소 하나의 리딩 축이 프로그래밍돼야 합니다 .

버전 2:

FA	추종 축	
블록 변경 모드	다음과 같은 모드를 사용할 수 있습니다 .	
	"NOC"	즉시 블록 변경이 이뤄집니다 .
	"FINE"	블록 변경이 ' 미세조정 동기화 ' 로 수행됩니다 .
	"COARSE"	블록 변경이 ' 일반 동기화 ' 로 수행됩니다 .
	"IPOSTOP"	지령치 기반 동기화를 위해 블록 변경이 수행됩니다 .
[,LAi,SynPosLAi,Zi,Ni]	(대괄호는 입력하지 마십시오)	
	최소 1 개 , 최대 5 개 연속 :	
LA1, ... LA5	리딩 축	
SynPosLAi	i 번째 리딩 축의 동기화 위치	
Z1, ... Z5	결합 팩터 i 의 분자	
N1, ... N5	결합 팩터 i 의 분모	
	결합 팩터 i = 분자 i/ 분모 i	

EGDEF 명령으로 미리 지정한 리딩 축에 한하여 활성화 라인에 프로그래밍할 수 있습니다 . 프로그래밍된 추종 축 (SynPosFA) 및 리딩 축 (SynPosLA) 의 ' 동기화 위치 ' 를 통해 위치가 정의되며 , 이 위치를 기준으로 축 그룹이 *동기화*된 것으로 해석됩니다 . 그룹화가 가동됐을 때 전자 기어가 동기화된 상태가 아닌 경우 , 추종 축이 지정된 동기화 위치로 이송합니다 .

버전 3:

파라미터는 버전 2 와 동일합니다 .

접근 모드	다음과 같은 모드를 사용할 수 있습니다 .	
	"NTGT"	다음 간극의 시간 최적화 돌입
	"NTGP"	다음 간극의 경로 최적화 돌입
	"ACN"	- 방향으로 로터리 축 이동
	"ACP"	+ 방향으로 로터리 축 이동
	"DCT"	프로그래밍된 동기화 위치에 맞게 시간 최적화
	"DCP"	프로그래밍된 동기화 위치에 맞게 거리 최적화

변용 3은 모듈로 리딩 축과 결합한 모듈로 추종 축에만 영향을 미칩니다. 시간 최적화에는 추종 축의 속도 한계가 고려됩니다 .

추가 정보

활성화 버전 설명

버전 1:

그룹화가 실행된 순간 , 리딩 축 및 추종 축의 위치가 '동기화된 위치' 로 저장됩니다 . '동기화된 위치 ' 는 시스템 변수 $AA_EG_SYN 로 판독할 수 있습니다 .

버전 2:

커플링 그룹에 모듈로 축이 포함된 경우 , 그 위치 값들은 절대값을 토대로 감소됩니다 . 이로 인해 다음 동기 위치에 근접할 수 있습니다 (소위, *상대 동기화*: 예를 들어, 다음 간극).'추종 축 오버라이드 실행' 인터페이스 신호 DB(30 + 축 번호), DBX 26 비트 4 가 발생한 경우에 한하여 동기화된 위치에 접근합니다 . 이 신호가 발생하지 않은 경우 , 이 신호가 설정될 때까지 EGONSYN 블록에서 프로그램이 중단되고 자체 정제 알람 16771 이 출력됩니다 .

버전 3:

간극 (각도) 은 360 * Zi/Ni 로 계산됩니다 . 호출 순간에 추종 축이 정지한 경우 , 시간 최적화에 맞춰서 경로 최적화 반환 신호가 응답합니다 .

추종 축이 이미 작동 중인 경우 , 추종 축의 현재 속도와 무관하게 NTGP 가 다음 간극에서 동기화를 시작합니다 . 추종 축이 이미 작동 중인 경우 , 추종 축의 현재 속도에 맞춰 NTGT 가 다음 간극에서 동기화를 시작합니다 . 필요한 경우 , 축의 속도 역시 감소됩니다 .

커브 테이블

리딩 축 중 하나에 **커브 테이블**이 사용된 경우 :

Ni	직선 커플링과 관련한 커플링 팩터의 분모는 반드시 0 으로 설정돼야 합니다 (직선 커플링에서 분모 0 은 규칙 위반으로 간주됩니다). 분모 0 은 시스템에 다음 정보를 전달합니다 .
Zi	Zi 는 사용할 커브 테이블의 번호입니다 . 전원을 켰을 때 지정된 번호의 커브 테이블이 이미 정의된 상태여야 합니다 .
LAi	지정된 리딩 축이 커플링 팩터를 통해 결합 (직선 커플링) 용으로 지정된 축에 응답합니다 .

커브 테이블 사용과 전자 기어 계단식 구성 및 동기화에 대한 자세한 내용은 다음 내용을 참조하십시오 .
참조 :
기능 매뉴얼의 특수 기능 , 결합 축 및 ESR(M3), " 결합 작동 및 리딩 값 커플링 ".

전원 , ON, 리셋 , 모드 변경 , 블록 탐색 시 , 전자 기어의 반응

- 전원을 켠 후에도 모든 커플링은 활성화되지 **않습니다** .
- 리셋이나 작동 모드 전환으로 인해 활성 커플링 상태가 달라지지 않습니다 .
- 블록 탐색 도중 , 전환 명령 , 전자 기어 삭제 및 정의가 실행되거나 호출되지 않고 무시됩니다 .

전자 기어의 시스템 변수

가공 프로그램은 전자 기어의 시스템 변수를 통해 전자 기어의 현재 축 커플링 상태를 파악하고 필요한 경우 이에 반응합니다 .

전자 기어 박스의 시스템 변수는 다음과 같이 설계됩니다 .

$AA_EG_ ...

또는

$VA_EG_ ...

참조 :
시스템 변수 설명서

9.4.3 전자 기어박스 활성화 (EGOFS, EGOFC)

기능

전자 기어 축 그룹을 비활성화하는 방법으로는 세 가지가 있습니다 .

프로그래밍

버전 1:

구문	설명
EGOFS(추종 축)	전자 기어가 비활성화됩니다 . 추종 축이 제동으로 인해 정지합니다 . 이 구문을 호출하면 사전 처리 중지 기능이 실행됩니다 .

버전 2:

구문	설명
EGOFS(추종 축 , 리딩 축 1,…, 리딩 축 5)	이 명령 파라미터 설정을 사용하면 각각의 리딩 축이 추종 축의 동작에 미치는 영향을 **선택적으로** 제거할 수 있습니다 .

최소 하나의 리딩 축이 지정돼야 합니다 . 지정된 리딩 축이 추종 축에 미치는 영향이 선택적으로 차단됩니다 . 이 구문을 호출하면 사전 처리 중지 기능이 실행됩니다 . 구문에 활성 상태의 리딩 축이 포함된 경우 , 추종 축이 계속 리딩 축의 영향을 받습니다 . 이 방법으로 모든 리딩 축의 영향이 배제된 경우 , 추종 축이 제동으로 인해 정지합니다 .

버전 3:

구문	설명
EGOFC(추종 스핀들 1)	전자 기어가 비활성화됩니다 . 비활성화 시점에 적용된 속도로 추종 스핀들이 계속 이동합니다. 이 구문을 호출하면 사전 처리 중지 기능이 실행됩니다 .

주

이 버전은 스핀들에만 사용할 수 있습니다 .

9.4.4 전자 기어 정의 삭제 (EGDEL)

기능

전자 기어의 정의를 삭제하려면 먼저 전자 기어의 축 그룹을 비활성화해야 합니다 .

프로그래밍

구문	의미
EGDEL(추종 축)	축 그룹의 결합 정의가 삭제됩니다 . 동시 활성화가 가능한 최대 축 그룹 갯수에 도달할 때까지 EGDEF 를 사용해서 축 그룹을 추가로 정의할 수 있습니다 . 이 구문을 호출하면 사전 처리 중지 기능이 실행됩니다 .

9.4.5 회전 이송 속도 (G95)/ 전자 기어 (FPR)

기능

FPR 명령을 사용하여 전자 기어의 추종 축을 회전 이송 속도를 결정하는 축으로 지정할 수 있습니다 . 이 명령에 대해서는 아래 내용을 참조하십시오 .

- 이송 속도는 전자 기어 추종 축의 지령치 속도에 의해 결정됩니다 .
- 지령치 속도는 리딩 스핀들 및 (경로 축에 해당하지 않는) 모듈로 축의 속도와 관련 결합 팩터로 산정됩니다 .
- 선형 또는 비모듈로 리딩 축의 속도 부분과 추종 축의 오버레이 이동은 고려되지 않습니다 .

9.5 동기 스핀들

기능

동기식 작동에는 **동기식 스핀들 쌍**으로 불리우는 추종 스핀들 (FS) 과 리딩 스핀들 (LS) 이 포함됩니다 . 정의된 기능적 상관성에 따라 결합이 활성화 (동기식 작동) 되면 추종 스핀들이 리딩 스핀들의 동작을 모방합니다 .

전용 채널 기계 데이터를 통해 고정된 구성에 각 기계의 동기식 스핀들 쌍을 할당하거나 CNC 가공 프로그램을 통해 특정 용도에 맞게 동기식 스핀들 쌍을 정의할 수 있습니다 . 각 NC 채널에서 최대 2 개의 동기식 스핀들 쌍이 동시에 작동할 수 있습니다 .

다음의 결합 동작에 대해서는 해당 가공 프로그램을 참조하십시오 .

- 정의 또는 변경
- 활성화
- 비활성화
- 가공 프로그램에서 삭제

.

또한 소프트웨어의 상태에 따라 ...

- 동기화 조건을 기다릴 수 있습니다
- 블록 변경 방법을 변경할 수 있습니다
- 지령치 결합 또는 실제값 결합 형식을 선택하거나 리딩 스핀들과 추종 스핀들간의 각도 옵셋을 지정할 수 있습니다
- 결합이 활성화되면 추종 축의 사전 프로그램이 전송됩니다
- 확인되거나 기존에 알려진 동기화 편차가 교정됩니다

.

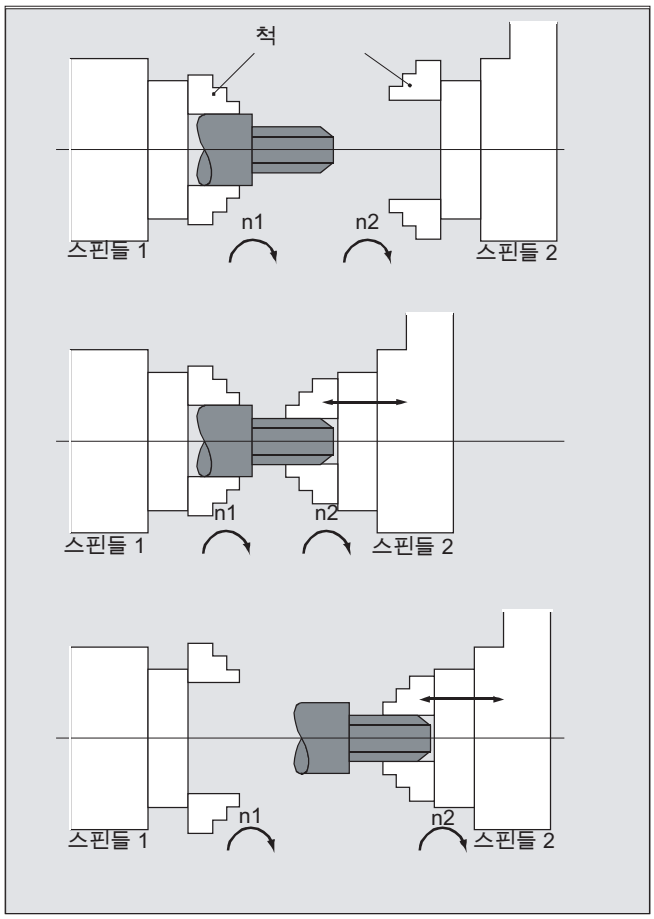
9.5.1 동기식 스핀들 : 프로그래밍 (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC)

기능

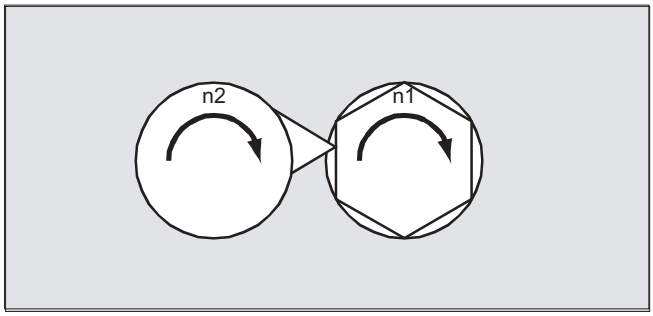
공작물 즉시 전송 등에서 , 동기 스핀들 기능을 사용하면 두 스핀들 (추종 스핀들 FS 및 리딩 스핀들 LS) 의 동시 이송을 빠르게 처리할 수 있습니다 .

이 기능은 다음의 모드를 지원합니다 .

- 속도 동기 ($n_{FS} = n_{LS}$)
- 위치 동기 ($\phi_{FS} = \phi_{LS}$)
- 각도 옵셋으로 위치 동기 ($\phi_{FS} = \phi_{LS} + \Delta\phi$)



리딩 스핀들과 추종 스핀들 사이의 변환 비율을 1 이외의 값으로 지정하면 멀티 에지 가공도 가능합니다 .



구문

COUPDEF(<FS>,<LS>,<TFS>,<TLS>,< 블록 변경 >,< 커플링 유형 >)
COUPON(<FS>,<LS>,<POSFS>)
COUPONC(<FS>,<LS>)
COUPOF(<FS>,<LS>,<POSFS>,<POSLS>)
COUPOFS(<FS>,<LS>)
COUPOFS(<FS>,<LS>,<POSFS>)
COUPRES(<FS>,<LS>)
COUPDEL(<FS>,<LS>)
WAITC(<FS>,< 블록 변경 >,<LS>,< 블록 변경 >)

주

약어 표시

COUPOF, COUPOFS, COUPRES 및 COUPDEL 문은 리딩 스핀들을 지정하지 않은 약어 표시를 지원합니다 .

의미

COUPDEF:	사용자 기반의 커플링 정의 / 변경
COUPON:	커플링 활성화 . 추종 스핀들이 실제 속도를 기준으로 리딩 스핀들과 동기화합니다 .
COUPONC:	이전에 프로그래밍된 M3 S... 또는 M4 S... 로 동작 시 커플링 . 추종 스핀들의 속도차가 즉시 전송됩니다 .
COUPOF:	커플링 해제 • 즉각적인 블록 변경 시 : COUPOF(<S2>,<S1>) • <POSFS> 또는 <POSLS> 비활성화 위치가 교차한 후에만 블록 변경 : COUPOF(<S2>,<S1>,<POSFS>) COUPOF(<S2>,<S1>,<POSFS>,<POSLS>)
COUPOFS:	추종 스핀들 정지를 통한 커플링 해제 . 즉각적인 블록 변경으로 가급적 빨리 블록 변경 : COUPOFS(<S2>,<S1>) 스위치 OFF 위치를 통과한 이후에만 블록 변경 : COUPOFS(<S2>,<S1>,<POSFS>)

COUPRES:	설정된 MD 및 SD 로 커플링 파라미터 리셋 .
COUPDEL:	사용자 정의 커플링 삭제 .
WAITC:	동기화 실행 조건 대기 (블록 변경이 이뤄지는 동안 NOC 가 IPO 로 증가)
<FS>:	추종 스핀들 지정

옵션 파라미터 :

<LS>:	메인 스핀들 지정 스핀들 번호로 지정 (예 : S2, S1)
<TFS>, <TLS>:	FS 및 LS 간의 변환 비율 . <TFS> = 분자 , <TLS> = 분모 디폴트 : <TFS> / <TLS> = 1.0 ; 분모 지정은 옵션
< 블록 변경 >:	블록 변경 특성 블록 변경 특성은 다음과 같습니다 . "NOC" 즉시 "FINE" ' 미세 동기 ' 도달 시 "COARSE" ' 일반 동기 ' 도달 시 "IPOSTOP" IPOSTOP 도달 시 . 즉 , 지령치에 의한 동기 (표준) 이후 블록 변경 특성은 모달로 효력이 발생합니다 .
< 커플링 유형 >:	커플링 형식 : FS 와 LS 간의 커플링 "DV" 지령치 커플링 (기본 설정) "AV" 실제값 커플링 "VV" 속도 커플링 커플링 형식은 모달입니다 .
<POSFS>:	리딩 스핀들과 추종 스핀들간의 각도 옵셋 범위 : 0°… 359,999°
<POSFS>, <POSLS>:	추종 스핀들과 리딩 스핀들의 스위치 OFF 위치 "POS_{FS}, POS_{LS} 를 통과하고 나면 블록 변경이 실행됩니다 " 범위 : 0°… 359,999°

예제

예제 1: 리딩 스핀들과 추종 스핀들을 사용한 가공

프로그래밍	설명
	; 리딩 스핀들 = 마스터 스핀들 = 스핀들 1
	; 추종 스핀들 = 스핀들 2
N05 M3 S3000 M2=4 S2=500	; 리딩 스핀들 3000 rpm 으로 회전 , 추종 스핀들 500 rpm 으로 회전
N10 COUPDEF(S2,S1,1,1,"NOC","Dv")	; 커플링 정의 (설정 가능)
...	
N70 SPCON	; 리딩 스핀들을 폐 - 루프 위치 제어 상태로 전환 (지령치 커플링)
N75 SPCON(2)	; 추종 스핀들을 폐 - 루프 위치 제어 상태로 전환
N80 COUPON(S2,S1,45)	; 옵셋 위치를 45° 로 즉시 커플링
...	
N200 FA[S2]=100	; 포지셔닝 속도 = 100 도 / 분
N205 SPOS[2]=IC(-90)	; 음의 방향으로 90° 오버레이해서 이송
N210 WAITC(S2,"Fine")	; ' 미세조정 ' 동기 대기
N212 G1 X... Y... F...	; 가공
...	
N215 SPOS[2]=IC(180)	; 양의 방향으로 180° 오버레이해서 이송
N220 G4 S50	; 드웰 시간 = 마스터 스핀들 50 회 회전
N225 FA[S2]=0	; 설정된 속도 활성화 (MD)
N230 SPOS[2]=IC(-7200)	; 20 회 회전 . 음의 방향으로 설정된 속도로 회전 .
...	
N350 COUPOF(S2,S1)	; 즉시 커플링 해제 , S=S2=3000
N355 SPOSA[2]=0	; 0° 에서 FS 정지
N360 G0 X0 Y0	
N365 WAITS(2)	; 스핀들 2 대기
N370 M5	; FS 정지
N375 M30	

예제 2: 속도차 프로그래밍

프로그래밍	설명
	; 리딩 스핀들 = 마스터 스핀들 = 스핀들 1
	; 추종 스핀들 = 스핀들 2
N01 M3 S500	; 500 rpm 으로 리딩 스핀들 회전
N02 M2=3 S2=300	; 300 rpm 으로 추종 스핀들 회전
...	
N10 G4 F1	; 마스터 스핀들의 드웰 시간
N15 COUPDEF (S2,S1,-1)	; 커플링 팩터 -1:1
N20 COUPON(S2,S1)	; 커플링 활성화 . 스핀들의 속도는 리딩 스핀들의 속도와 커플링 팩터에 따라 달라집니다 .
...	
N26 M2=3 S2=100	; 속도차 프로그래밍

예제 3: 속도차에 대한 이동 명령 전송 예제

1. COUPON 으로 추종 스핀들의 이전 프로그래밍 도중 커플링 활성화

프로그래밍	설명
	; 리딩 스핀들 = 마스터 스핀들 = 스핀들 1
	; 추종 스핀들 = 스핀들 2
N05 M3 S100 M2=3 S2=200	; 리딩 스핀들 100 rpm 으로 회전 , 추종 스핀들 200 rpm 으로 회전 .
N10 G4 F5	; 마스터 스핀들의 드웰 시간 = 5 초
N15 COUPDEF(S2,S1,1)	; FS 에서 LS 로의 변환 비율 = 1.0 (디폴트)
N20 COUPON(S2,S1)	; 리딩 스핀들에 즉시 커플링
N10 G4 F5	; 100 rpm 으로 추종 스핀들 회전

2. COUPONC 으로 추종 스핀들의 이전 프로그래밍 도중 커플링 활성화

프로그래밍	설명
	; 리딩 스핀들 = 마스터 스핀들 = 스핀들 1
	; 추종 스핀들 = 스핀들 2
N05 M3 S100 M2=3 S2=200	; 리딩 스핀들 100 rpm 으로 회전 , 추종 스핀들 200 rpm 으로 회전 .
N10 G4 F5	; 마스터 스핀들의 드웰 시간 = 5 초
N15 COUPDEF(S2,S1,1)	; FS 에서 LS 로의 변환 비율 = 1.0 (디폴트)
N20 COUPONC(S2,S1)	; 리딩 스핀들에 즉시 커플링 및 S2 에 이전에 사용한 속도를 전송
N10 G4 F5	; 300 rpm(100 rpm + 200 rpm) 으로 S2 회전

3. COUPON 으로 추종 스핀들 정지 커플링 활성화

프로그래밍	설명
	; 리딩 스핀들 = 마스터 스핀들 = 스핀들 1
	; 추종 스핀들 = 스핀들 2
N05 SPOS=10 SPOS[2]=20	; 포지셔닝 모드의 추종 스핀들 S2
N15 COUPDEF(S2,S1,1)	; FS 에서 LS 로의 변환 비율 = 1.0 (디폴트)
N20 COUPON(S2,S1)	; 리딩 스핀들에 즉시 커플링
N10 G4 F1	; 커플링 종료 , S2 는 20° 에서 정지 .

4. COUPONC 으로 추종 스핀들 정지 커플링 활성화

주

포지셔닝 모드 또는 축 모드

커플링 전에 추종 스핀들이 포지셔닝 모드 또는 축 모드에 있는 경우 추종 스핀들은 COUPON(<FS>,<LS>) 과 COUPONC(<FS>,<LS>) 의 경우와 동일하게 동작합니다 .

유의사항
리딩 스핀들 및 축 동작 커플링을 정의하기 전에 리딩 스핀들이 동작에 연관된 경우 , 머신 데이터의 속도 한계값 MD32000 $MA_MAX_AX_VELO (최대 축 속도) 은 커플링이 활성화된 후에도 적용됩니다 . 이런 동작을 방지하려면 커플링을 정의하기 전에 축을 스핀들 모드 (M3 S... 또는 M4 S...) 로 전환해야 합니다 .

추가 정보

스핀들 동기 페어 정의

설정된 커플링

커플링 설정에서 , 리딩 스핀들과 추종 스핀들이 머신 데이터에 정의됩니다 . 설정된 스핀들은 가공 프로그램에서 변경할 수 없습니다 . 쓰기 보호가 설정되어 있지 않은 경우 , COUPDEF 를 사용하여 가공 프로그램에서 커플링을 설정할 수 있습니다 .

사용자 정의 커플링 :

COUPDEF 를 사용하면 가공 프로그램에서 커플링을 재정의 또는 변경할 수 있습니다 . 커플링이 이미 활성 상태인 경우에는 먼저 COUPDEL 을 사용하여 삭제한 후 새 커플링을 정의해야 합니다 .

커플링 정의 : COUPDEF

다음에 의해 커플링이 정의됩니다 :

COUPDEF(<FS>,<LS>,<TFS>,<TLS>, 블록 변경 동작 , 커플링 유형)

추종 스핀들 (FS) 및 리딩 스핀들 (LS)

커플링은 추종 스핀들 (FS) 과 리딩 스핀들 (LS) 의 교유한 축 이름을 사용하여 정의됩니다 . 축 이름은 모든 COUPDEF 동작에서 프로그래밍해야 합니다 . 다른 파라미터들은 모달이므로 , 변경할 필요가 있을 경우에만 프로그램 해야 합니다 .

예제 :

COUPDEF(S2,S1)

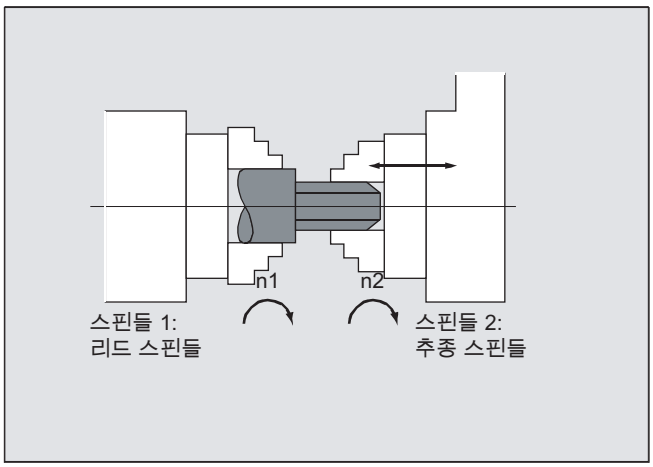
변환 비율 , TFS / TLS

변환 비율은 추종 스핀들 (분자) 및 리딩 스핀들 (분모) 간의 속도 비율입니다 . 분자는 반드시 프로그래밍해야 합니다 . 프로그래밍된 값이 없는 경우 분모는 1.0 으로 설정됩니다 .

예제 :

추종 스핀들 S2 및 리딩 스핀들 S1, 변환 비율 = 1 / 4 = 0.25.

COUPDEF(S2,S1,1.0, 4.0)



주

또한 커플링이 활성 상태이고 스핀들이 회전하는 경우 즉시 변환 비율을 변경할 수 있습니다 .

블록 변경 동작 NOC, FINE, COARSE, IPOSTOP

블록 변경 동작을 프로그래밍할 때 다음 약어 표시를 사용할 수 있습니다 .

- "NO": 즉시 (표준)
- "FI": ' 미세 동기 ' 도달 시
- "CO": ' 일반 동기 ' 도달 시
- "IP": IPOSTOP 도달 시 . 즉 , 지령치에 의한 동기 후

커플링 형식 DV, AV

⚠ 주의
커플링 형식은 커플링이 해제된 변경할 수 있습니다 !

동기 모드 활성화 (COUPON, POSFS)

- LS 와 FS 간의 각도를 기준으로 커플링 활성화 :
  - COUPON(S2,S1)
  - COUPON(S2,S1,<POSFS>)
  - COUPON(S2)
- 각도 옵셋 <POSFS> 로 커플링 활성화

  입체 공작물에 대한 위치 동기 커플링 .
  <POSFS> 는 양의 방향 회전에서 리딩 스핀들의 0° 위치를 참조합니다 .
  <POSFS> 값 범위 : 0°… 359,999°

  - COUPON(S2,S1,30)

  이 방법을 이용하면 커플링이 이미 활성 상태인 경우에도 각도 옵셋을 변경할 수 있습니다 .

추종 스핀들의 포지셔닝

동기 스핀들 커플링이 활성 상태인 경우 , 마스터 스핀들에 의해 시작된 동작에 구애 받지 않고 ±180° 의 범위 내에서 추종 스핀들의 위치도 지정할 수 있습니다 .

포지셔닝 SPOS

SPOS=… 로 추종 스핀들을 보간할 수 있습니다 .

예제 :

SPOS[2] = IC(-90)

SPOS 에 대한 추가 정보는 다음을 참조하십시오 .
참고 자료 :
프로그래밍 매뉴얼 , 기본편

속도 편차 (M3 S... 또는 M4 S...)

속도 편차는 두 속도 소스의 신호 중첩으로 인해 발생하며 추종 스핀들을 기준으로 다시 프로그래밍됩니다 (예 : 동기 스핀들 커플링이 활성 상태인 동안 속도 제어 모드에서 S<n>=... 또는 M<n>=3, M<n>=4). 그 과정에서 커플링 팩터와 적합한 접두어가 추가된 추종 스핀들을 사용하여 메인 스핀들로부터 이 속도 콤포넌트가 도출됩니다 .

주

회전 방향이 M3 또는 M4 인 경우 속도 S... 도 프로그래밍해야 합니다 . 그렇지 않을 경우 프로그래밍이 되지 않았음을 알리는 알람이 발생합니다 .

속도 편차에 대한 자세한 정보는 다음을 참조하세요 .
참고 자료 :
기능 매뉴얼 , 확장 기능 : 동기 스핀들 (S3)

COUPONC 에서 속도 편차

속도 편차에 대한 이동 작업 전송

COUPONC 를 사용하여 동기 스핀들 커플링을 활성화하면 현재 속도가 추종 스핀들 (M3 S... 또는 M4S...) 에 오버레이됩니다 .

주

오버레이 사용

동기 스핀들 커플링 COUPONC 를 사용하는 스핀들 속도 (M3 S... 또는 M4 S...) 오버레이는 오버레이를 사전에 활성화한 경우에만 적용됩니다 .

리딩 스핀들의 다이나믹 제한

리딩 스핀들의 다이나믹 반응은 추종 스핀들에 오버레이가 적용될 때 다이나믹 한계 값을 초과하지 않는 수준으로 제한해야 합니다 .

속도 , 가속도 : FA, ACC, OVRA, VELOLIMA

추종 스핀들의 축 속도 및 가속도는 다음을 사용하여 프로그래밍할 수 있습니다 .

- FA[SPI(S<n>)] 또는 FA[S<n>] (축 속도)
- ACC[SPI(S<n>)] 또는 ACC[S<n>] (축 가속)
- OVRA[SPI(S<n>)] 및 OVRA[S<n>] (축 오버라이드)
- VELOLIMA[SPI(S<n>)] 및 VELOLIMA[S<n>] (각각 축 속도 증가 및 감소)

여기서 <n> = 1, 2, 3, ... (추종 스핀들의 스핀들 번호)

참고 자료 :
프로그래밍 매뉴얼 , 기본 사항

주

가속 콤포넌트 JERKLIMA[S<n>]

축 속도 증가 또는 감소 프로그래밍 기능은 현재 스핀들에서는 사용할 수 없습니다 .

축 다이나믹 응답 구성에 대한 추가 정보는 다음을 참조하십시오 .
참고 자료 :
기능 매뉴얼 , 확장 기능 : 로터리 축 (R2)

프로그램 가능한 블록 변경 동작 WAITC

커플링 파라미터 또는 포지셔닝 동작 변경 후와 같은 상황에서 WAITC 를 사용하면 다양한 동기 조건 (일반 , 미세 , IPOSTOP) 을 적용하여 블록 변경 동작을 정의할 수 있습니다 . 동기 조건을 지정하지 않으면 COUPDEF 정의에서 지정한 블록 변경 동작이 적용됩니다 .

예제 :

COUPDEF 에 따른 동기 조건 충족 대기

WAITC( )

추종 스핀들 S2 에 대한 동기 조건 FINE (미세 동기) 및 추종 스핀들 S4 에 대한 동기 조건 COARSE (일반 동기) 충족 대기

WAITC(S2,"FINE",S4,"COARSE")

커플링 해제 COUPOF

COUPOF 는 커플링 해제 동작을 정의하는 데 사용할 수 있습니다 .

- 블록 변경 후 즉시 커플링 해제 :
  - COUPOF(S2,S1) (리딩 스핀들 지정)
  - COUPOF(S2) (리딩 스핀들을 지정하지 않음)
- 스위치 off 위치에 도달된 후 커플링 해제 . 스위치 OFF 위치에 도달한 후 블록 변경이 이루어집니다 .
  - COUPOF(S2,S1,150) (FS 스위치 OFF 위치 : 150°)
  - COUPOF(S2,S1,150, 30) (FS 스위치 OFF 위치 : 150°, LS: 30°)

추종 스핀들 정지를 통한 커플링 해제 COUPOFS

COUPOFS 는 추종 스핀들 정지를 통한 커플링 해제 동작을 정의하는 데 사용할 수 있습니다 .

- 추종 스핀들 정지 및 즉각적인 블록 변경을 통한 커플링 해제 :
  - COUPOFS(S2,S1) (리딩 스핀들 지정)
  - COUPOFS(S2) (리딩 스핀들을 지정하지 않음)
- 추종 스핀들 정지를 통해 스위치 off 위치를 통과한 후 커플링 해제 . 스위치 OFF 위치에 도달한 후 블록 변경이 이루어집니다 .
  - COUPOFS(S2,S1,150) (FS 스위치 OFF 위치 : 150°)

커플링 삭제 COUPDEL

COUPDEL 은 커플링을 삭제합니다 .

- COUPDEL(S2,S1) (리딩 스핀들 지정)
- COUPDEL(S2) (리딩 스핀들 미지정)

커플링 파라미터 리셋 COUPRES

COUPRES 는 머신 데이터 및 셋팅 데이터에서 설정된 커플링 값을 활성화합니다 .

- COUPRES(S2,S1) (리딩 스핀들 지정)
- COUPRES(S2) (리딩 스핀들 미지정)

시스템 변수

추종 스핀들의 현재 커플링 상태

추종 스핀들의 현재 커플링 상태는 다음 시스템 변수를 사용하여 읽을 수 있습니다 .

$AA_COUP_ACT[<FS>]

값	의미
0	커플링 없음
4	동기 스핀들 커플링 동작
참고 기타 시스템 변수 값은 축 동작을 참조합니다 . **참고 자료** 리스트 매뉴얼 , 시스템 변수	

현재 각도 옵셋

리딩 스핀들과 연관된 추종 스핀들의 현재 각도 옵셋은 다음 시스템 변수를 사용하여 읽을 수 있습니다 .

- $AA_COUP_OFFS[<FS>] (지령치에 각도 옵셋)
- $VA_COUP_OFFS[<FS>] (실제 값에 각도 옵셋)

주

시스템이 디스에이블 되었다가 , 커플링 동작 및 추종 모드로 인해 다시 인에이블 된 경우 , 시스템 재 인에이블 시의 위치 옵셋은 프로그램된 값과 차이가 납니다 . 이 경우에는 새 위치 옵셋을 읽고 , 필요한 경우 가공 프로그램에서 수정할 수 있습니다 .

9.6 마스터 / 슬레이브 그룹 (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS)

기능

SW 6.4 이하 버전의 마스터 / 슬레이브 결합 기능은 관련 축이 정지한 동안에만 마스터 축에 슬레이브 축 결합을 허용합니다 .

하지만 SW 6.5 확장 버전에서는 **회전** , 속도 제어 스핀들 , 그리고 동적 구성의 결합 및 결합 해제가 가능합니다 .

구문

MASLON(Slv1,Slv2,...,)	
MASLOF(Slv1,Slv2,...,)	
MASLDEF(Slv1,Slv2,..., 마스터 축)	동적 구성 확장
MASLDEL(Slv1,Slv2,...,)	동적 구성 확장을 위한 확장
MASLOFS(Slv1, Slv2, ...,)	슬레이브 스핀들 확장

주

MASLOF/MASLOFS 의 경우 , 실질적인 사전 처리 중지가 필요하지 않습니다 . 사전 처리 중지가 누락된 이유로 다음 프로그램이 진행될 때까지 슬레이브 축의 시스템 변수 $P 에 의해 값 업데이트가 이뤄지지 않습니다 .

의미

일반

MASLON	임시 결합 닫기 (활성화).
MASLOF	활성 상태의 결합 열기 . 반드시 스핀들 확장에 유의해야 합니다 .

동적 구성 확장

MASLDEF	기계 데이터를 사용하여 사용자 정의 결합 또는 가공 프로그램에서 작성 / 변경
MASLOFS	MASLOF 에 상응하는 결합 비활성화 및 자동으로 슬레이브 스핀들 정지
MASLDEL	마스터 / 슬레이브 축 그룹 결합 해제 및 그룹 정의 삭제
Slv1, Slv2, ...	마스터 축에 의해 유도되는 슬레이브 축
Master axis	마스터 / 슬레이브 그룹에 정의된 슬레이브 축을 제어하는 축

예

예제 1: 마스터 / 슬레이브 결합 동적 구성

가공 프로그램에서 마스터 / 슬레이브 결합 동적 구성 :

축 컨테이너 회전 후에 관련 축이 마스터 축이 되어야 합니다 .

프로그램 코드	비고
MASLDEF(AUX,S3)	; S3 마스터 축을 보조 축으로 지정
MASLON(AUX)	; 보조 축 결합 활성화
M3=3 S3=4000	; 시계 방향 회전
MASLDEL(AUX)	; 구성 삭제 및 결합 비활성화
AXCTSWE(CT1)	; 컨테이너 회전

예

예제 2: 슬레이브 축의 실제값 결합

PRESETON 을 사용하여 마스터 축의 동일한 값에 슬레이브 축의 실제값 결합 :

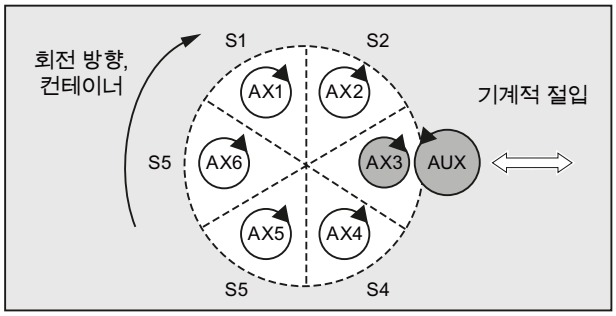
영구 정의된 마스터 / 슬레이브 결합의 경우 , PRESETON 을 사용하여 슬레이브 축의 실제값을 변경할 수 있습니다 .

프로그램 코드	비고
N37262 $MA_MS_COUPLING_ALWAYS_ACTIVE[AX2]=0	; 일시적으로 영구 결합 비활성화
N37263 NEWCONF	
N37264 STOPRE	
MASLOF(Y1)	; 임시 결합 비활성화
N5 PRESETON(Y1,0,Z1,0,B1,0,C1,0,U1,0)	; 전원을 켰을 때 슬레이브 축이 활성화되면 기준이 되지 않은 슬레이브 축의 실제값 설정
N37262 $MA_MS_COUPLING_ALWAYS_ACTIVE[AX2]=1	; 영국 결합 활성화
N37263 NEWCONF	

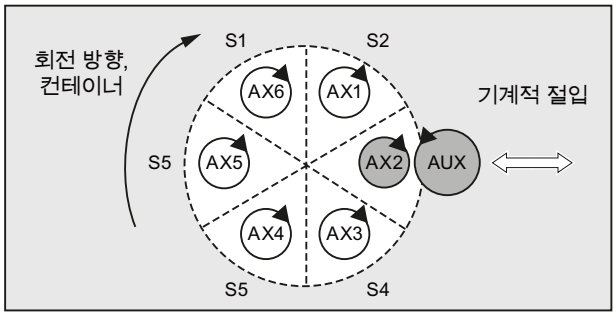
예제 3: 결합 순서 , 위치 3/ 컨테이너 CT1

컨테이너 회전 후 다른 스핀들과의 결합을 활성화하려면 이전 결합을 해제하고 구성을 삭제한 후 새 결합을 구성해야 합니다 .

초기 상황 :



한 개의 슬롯만큼 회전 후 :



참고 문서 :

기능 설명서의 확장 기능 , 다양한 조작 패널 전면 및 NCU(B3), " 축 컨테이너 " 단원에 수록된 " 축 컨테이너 "

그 밖의 정보

일반

MASLOF	이 명령은 속도 제어 모드에서 스핀들용으로 직접 실행됩니다 . 이 시점에 회전 중이던 슬레이브 스핀들의 새 속도가 프로그래밍될 때까지 기존 속도를 유지합니다 .

동적 구성 확장

MASLDEF	가공 프로그램에서 마스터/슬레이브 그룹 정의. 그에 앞서, 기계 데이터를 이용한 정의가 별도로 이뤄집니다 .
MASLDEL	명령에 의해 마스터 축에 슬레이브 축 할당이 취소됨과 동시에 MASLOF과 같은 결합이 비활성화됩니다 . 기계 데이터에서 정의한 마스터 / 슬레이브 정의는 그대로 유지됩니다 .
MASLOFS	MASLOFS 는 결합이 비활성화됐을 때 슬레이브 스핀들을 자동으로 정지시키는 데 사용할 수 있습니다 . 위치 지정 모드에서 축과 스핀들의 경우 , 정지 상태 (속도 0) 에 한하여 결합이 활성화 또는 비활성화됩니다 .

주

슬레이브 축의 경우 , PRESETON 을 사용하여 실제값을 마스터 축과 동일한 값으로 동기화할 수 있습니다. 이를 위해서는 전원을 켰을 때 기준이 되지 않는 슬레이브 축의 실제값을 마스터/축의 값에 상응하게 설정할 수 있도록 영구 / 슬레이브 결합을 일시적으로 비활성화해야 합니다 . 그리고 나면 결합이 영구적으로 재구축됩니다 .

MD 설정 MD37262 $MA_MS_COUPLING_ALWAYS_ACTIVE = 1 을 사용하면 영구 마스터 / 슬레이브 결합이 활성화되는데 , 이 설정은 임시 결합 언어 명령에 전혀 영향을 미치지 않습니다 .

스핀들 결합 특성 !

개 - 루프 제어 모드에서 스핀들의 경우 , MASLON, MASLOF, MASLOFS 및 MASLDEL 은 기계 데이터 MD37263 $MA_MS_SPIND_COUPLING_MODE 를 사용하여 명확하게 정의됩니다 .

기본 설정 상태 (MD37263 = 0) 에서는 관련 축이 정지 상태인 경우에 한하여 슬레이브 축이 결합 및 결합 해제됩니다 . MASLOFS 는 MASLOF 에 대응합니다 .

MD37263 = 1 의 상태에서는 결합 명령이 즉시 실행되므로 , 결과적으로 즉시 가동이 시작됩니다 . MASLON 에서는 결합이 즉시 활성화되는 반면 , MASLOFS 나 MASLOF 에서는 즉시 비활성화됩니다 . 이 시점에 회전 중이던 슬레이브 스핀들이 MASLOFS 에서는 자동으로 정지하고 , MASLOF 에서는 새 속도가 프로그래밍될 때까지 기존 속도를 유지합니다 .

동기 동작

10

10.1 기본원리

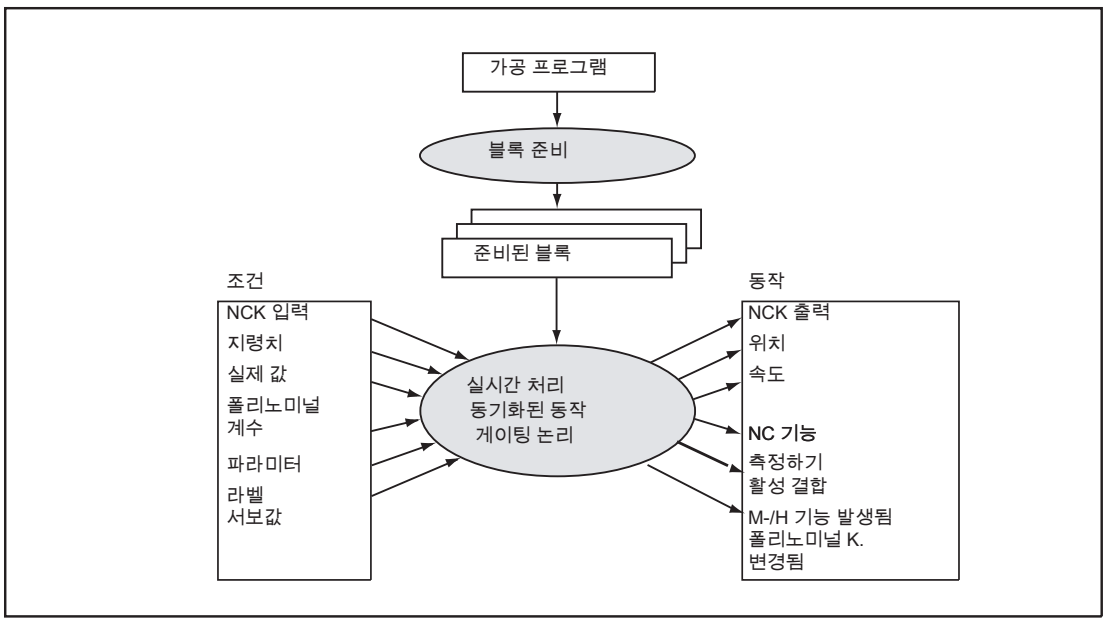
기능

동기화 동작을 이용하면 해당 작업을 실행하여 가공 블록과 동기화할 수 있습니다 .

또한 해당 작업이 실행될 시간을 조건으로 정의할 수 있습니다 . 조건은 보간 싸이클로 모니터링됩니다. 따라서 작업이 실시간 이벤트에 응답하며 실행 역시 블록 경계의 제한을 받지 않습니다 .

동기화 동작에는 서비스 수명과 프로그래밍된 주 실행 변수의 검사 빈도에 대한 정보는 물론이고 작업이 시작되는 빈도에 대한 정보도 포함됩니다 . 이런 방법으로 단발적으로 또는 보간 싸이클에 맞춰 주기적으로 작업을 실행할 수 있습니다 .

가능한 용도

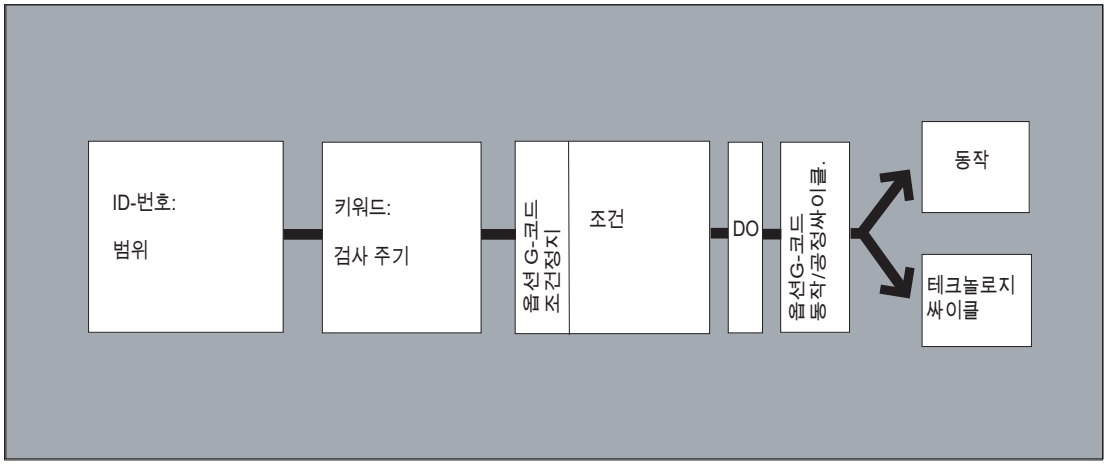


- 가동에 중요한 영향을 미치는 애플리케이션 (예 : 공구 변경) 최적화
- 외부 이벤트에 신속하게 응답
- AC 컨트롤 프로그래밍
- 안전 기능 설정
- ....

프로그래밍

동기화 동작은 별도의 블록에 자체적으로 프로그래밍되며 실행 가능한 다음 블럭을 기준으로 기계의 기능을 실행합니다 (예 : G0, G1, G2, G3 으로 이송).

동기화 동작은 아래와 같이 각기 다른 작업을 담당하는 최대 5개의 명령어 요소로 구성됩니다.



구문 :
DO < 동작 1> < 동작 2> …
< 키워드 > < 조건 > DO < 동작 1> < 동작 2> …
ID=<n> < 키워드 > < 조건 > DO < 동작 1> < 동작 2> …
IDS=<n> < 키워드 > < 조건 > DO < 동작 1> < 동작 2> …

의미 :

DO	프로그래밍된 동작 (들) 을 초기화하는 명령어 < 조건 > 을 충족하는 경우 (프로그래밍된 경우) 에만 유효합니다 . → " 동작 " 참조
< 동작 1> < 동작 2> …	시작될 동작 예제 : • 변수 할당 • 공정 싸이클 시작
< 키워드 >	키워드 (WHEN, WHENEVER, FROM 또는 EVERY) 를 사용하여 동기화 동작의 < 조건 > 에 대한 주기적 검사를 정의합니다 . → " 주기적으로 조건 검사 " 참조
< 조건 >	주 실행 변수의 게이팅 논리 IPO 클록 싸이클로 조건을 검사합니다 .
ID=<n> or IDS=<n>	식별 번호 가공 순서에서 위치와 적용 범위는 식별 번호를 사용하여 정의됩니다 . → " 적용 범위 및 가공 순서 " 참조

동기화 동작 / 공정 사이클 조정

다음 명령은 동기화 동작 / 공정 사이클을 조정하는 데 사용할 수 있습니다 .

명령	의미
CANCEL(<n>)	동기화 동작 취소 → " 동기화 동작 취소 " 참조
LOCK(<n>)	동기화 동작 잠금
UNLOCK(<n>)	동기화 동작 잠금 해제
RESET	공정 사이클 리셋 LOCK, UNLOCK 및 RESET 에 대해서는 " 해제 , 인에이블 , 리셋 " 을 참조하십시오 .

예제

프로그램 코드	비고
WHEN $AA_IW[Q1]>5 DO M172 H510	; 축 Q1 의 실제값이 5mm 를 초과한 경우 , PLC 인터페이스에 보조 함수 M172 와 H510 이 출력됩니다 .

10.1.1 적용 범위 및 가공 순서 (ID, IDS)

기능

적용 범위

동기화 동작 적용 범위는 식별 ID 나 IDS 를 사용하여 정의됩니다 .

모달 ID 없음 :	자동 모드에서 비모달식 동기화 동작 진행
ID:	프로그램 종료까지 자동 모드에서 ID=n 모달식 동기화 동작 진행
IDS:	프로그램 종료 이후를 포함하여 모든 작동 모드에 모달식으로 적용되는 정적 동기화 동작 진행

용도

- JOG 모드에서 AC 루프
- SI(Safety Integrated) 를 준한 논리 연산
- 기능 모니터링 , 모든 모드에서 기계 상태에 응답

실행 순서

정적으로 적용되는 모달식 동기화 동작은 해당 ID 또는 IDS 번호 (ID=<n> 또는 IDS=<n>) 순서대로 보간 클록 사이클로 처리됩니다 .

(ID 번호가 없는) 비모달식 동기화 동작은 모달식 동기화 동작 실행 후에 프로그래밍된 순대대로 실행됩니다 .

주

모달식 동기화 동작은 기계 데이터 설정에 의해 수정 또는 삭제가 금지됩니다 (→ 기계건축 OEM!)

프로그래밍

구문	의미
모달 ID 없음	**자동 모드**에서만 동기화 동작이 적용됩니다 . 실행 가능한 다음 블럭 (동작 구문이나 기타 기계 작업이 있는 블록) 에만 적용되며 **비모달식**입니다 . 예 : WHEN $A_IN[3]==TRUE DO $A_OUTA[4]=10
ID=<n> ...	동기화 동작이 다음 블록에 **모달식으로** 적용되며 , CANCEL(<n>) 으로 동기화 동작을 비활성화하거나 동일한 ID 로 새로운 동기화 동작을 프로그래밍하여 기존 동기화 동작을 덮어쓸 수 있습니다 . M30 블록에 적용된 동기화 동작이 프로그램 종료를 지연시킵니다 . **자동 모드**에서만 동기화 동작이 적용됩니다 . <n> 의 값 범위 : 1 ... 255 예 : ID=2 EVERY $A_IN[1]==1 DO POS[X]=0
IDS=<n>	정적 동기화 동작이 **모든 모드에서 모달식으로** 적용됩니다 . 심지어 프로그램 종료 이후에서 활성 상태를 유지하며 ASUB 를 사용하여 전원을 켠 직후에 활성화할 수 있습니다 . 그러므로 선택된 작동 모드에 구애 받지 않고 NC 에서 실행되어야 할 동작을 활성화할 수 있습니다 . <n> 의 값 범위 : 1 ... 255 예 : IDS=1 EVERY $A_IN[1]==1 DO POS[X]=100

10.1.2 주기적으로 조건 검사 (WHEN, WHENEVER, FROM, EVERY)

기능

하나의 키워드를 사용하여 동기화 동작 조건을 주기적으로 검사할 수 있습니다 . 키워드가 하나도 프로그래밍되지 않을 경우 , 동기화 동작이 IPO 싸이클마다 한 차례씩 실행됩니다 .

키워드

키워드 없음	어떤 조건에서도 동기화 동작이 실행되지 않고 , 간극 싸이클에 맞춰 주기적으로 관련 동작이 실행됩니다 .
WHEN	조건을 한 차례 충족할 때까지 보간 싸이클마다 조건을 검사합니다 . 결과적으로 , 관련 동작이 한 번만 실행됩니다 .
WHENEVER	간극 싸이클마다 주기적으로 조건을 검사합니다 . 조건을 충족하는 동안에는 간극 싸이클마다 관련 동작이 실행됩니다 .
FROM	조건을 한 차례 충족할 때까지 보간 싸이클마다 조건을 검사합니다 . 그리고 나서 더 이상 조건을 충족하지 않더라도 동기화 동작이 활성 상태인 동안에는 관련 동작이 실행됩니다 .
EVERY	보간 싸이클마다 조건을 검사합니다. 조건을 충족할 때 한 차례만 관련 동작이 실행됩니다 . 에지 트리거 (Edge Triggering): ' 참 ' 상태에서 ' 거짓 ' 상태로 조건이 바뀌면 관련 동작이 다시 실행됩니다 .

주 실행 변수

사용되는 변수는 보간 싸이클로 검사됩니다 . 동기화 동작의 주 실행 변수를 사용해도 사전 처리 중지 기능은 실행되지 않습니다 .

분석 :

주 실행 변수가 가공 프로그램에 나타난 경우 (예 : 실제값 , 디지털 입력 또는 출력의 위치 등), 이전 블록이 실행되고 주 실행 변수의 값을 확보할 때까지 사전 처리가 중지됩니다 .

예

예제 1: 키워드 없음

프로그램 코드	비고
DO $A_OUTA[1]=$AA_IN[X]	; 아날로그 출력으로 실제값 출력

예제 2: WHENEVER

프로그램 코드	비고
WHENEVER $AA_IM[X] > 10.5*SIN(45) DO …	; 사전 처리가 진행되는 동안 계산된 표현식과 비교
WHENEVER $AA_IM[X] > $AA_IM[X1] DO …	; 다른 주 실행 변수와 비교
WHENEVER ($A_IN[1]==1) OR ($A_IN[3]==0) DO ...	; 게이트에 의해 상관 관계에 있는 두 개의 연산 비교

예제 3: EVERY

프로그램 코드	비고
ID=1 EVERY $AA_IM[B]>75 DO POS[U]=IC(10) FA[U]=900	; 축 B 의 실제값이 기계 좌표에서 75 의 값을 초과한 경우 항상 비교 - 축 이송과 더불어 U 축이 10 의 단위로 앞으로 이동해야 합니다 .

그 밖의 정보

조건

조건은 부울 (Boolean) 연산자를 이용한 모든 방식의 논리식으로 구성됩니다 . 부울 연산식은 항상 대괄호 안에 명기되어야 합니다 .

조건은 보간 싸이클마다 검사됩니다 .

조건 앞에는 G 코드를 명기할 수 있습니다 . G 코드를 명기하면 현재 가공 프로그램의 상태에 관계 없이 조건 검사 및 동작 / 공정 싸이클이 정의된 설정에 맞게 실행됩니다 . 실행 조건을 충족했을 경우 지정된 초기 상태에서 언제라도 동기화 동작이 실행되어야 하므로 동기화 동작을 프로그램 환경과 별로도 구현할 필요가 있습니다 .

용도

G 코드 (G70, G71, G700, G710) 를 통한 조건 검사 및 관련 동작에 대한 측정 체계 정의

특정 동작에 G 코드가 별도로 지정되지 않은 경우 , 조건에 지정된 G 코드가 조건 검사 및 동작에 적용됩니다 .

각 조건 구간에는 G 코드 그룹 중 오직 **하나의 G 코드**만 프로그래밍할 수 있습니다 .

가능한 조건

- 주 실행 변수 비교 (아날로그 / 디지털 입력 / 출력 등)
- 비교 결과를 부울 게이트로 표현
- 실시간 표현식 계산
- 블록 시작 부분과의 거리 / 시간
- 블록 종료 부분과의 거리
- 측정된 값 , 측정 결과
- 서보값
- 가속도 , 축 상태

10.1.3 동작 (DO)

기능

동기화 동작에서는 하나 이상의 동작을 프로그래밍할 수 있습니다 . 하나의 블록에 프로그래밍된 동작들은 동일한 보간 싸이클에서 모두 효력을 발휘합니다 .

구문

DO < 동작 1> < 동작 2> …

의미

DO	조건을 충족했을 때 하나의 동작이나 공정 싸이클을 초기화합니다 .
< 동작 >	조건(예: 변수 할당, 축 결합 활성화, NCK 출력 설정, M 출력, S 및 H 함수, 프로그래밍된 G 코드 지정 등) 이 충족한 경우 , 동작이 시작됩니다 .

동작 / 공정 싸이클에 맞게 동기화 동작에 **G 코드**를 프로그래밍할 수 있습니다 . G 코드는 블록 및 공정 싸이클의 모든 동작에 대한 조건과 다르게 지정할 수 있습니다 . 동작 구간에 공정 싸이클이 포함된 경우 , 공정 싸이클이 완료된 이후에도 현재 G 코드는 다음 G 코드가 나타날 때까지 모든 동작에 대해 모달식으로 효력을 유지합니다 .

각 동작 구간마다 G 코드 그룹 중 오직 **하나의 G 코드** (G70, G71, G700, G710) 만 프로그래밍할 수 있습니다 .

예 : 두 개의 동작이 있는 동기화 동작

프로그램 코드	비고
WHEN $AA_IM[Y]>=35.7 DO M135 $AC_PARAM=50	; 조건을 충족한 경우 PLC 에 M135 가 출력되고 오버라이드는 50% 로 설정됩니다 .

10.2 조건 및 동작 관련 연산자

비교 (==, <>, <, >, <=, >=)	조건에서 변수 또는 부분식을 비교할 수 있습니다 . 모든 결과는 항상 데이터 유형 BOOL로 표현됩니다. 일반적인 비교 연산자는 모두 사용할 수 있습니다 .
부울 연산자 (NOT, AND, OR, XOR)	변수 , 상수 또는 비교 연산자가 일반 부울 연산자로 서로 연결될 수 있습니다 .
비트 단위 (Bit-by-bit) 연산자 (B_NOT, B_AND, B_OR, B_XOR)	사용할 수 있는 비트 연산자로는 B_NOT, B_AND, B_OR, B_XOR 가 있습니다 .
기본 산술 연산 (+, -, *, /, DIV, MOD)	주 실행 변수를 서로 연결하거나 기본 산술 형식으로 상수에 연결할 수 있습니다 .
수학 함수 (SIN, COS, TAN, ASIN, ACOS, ABS, TRUNC, ROUND, LN, EXP, ATAN2, POT, SQRT, CTAB, CTABINV).	데이터 유형 REAL 의 변수에는 수학 함수를 사용할 수 없습니다 .
지수 연동	주 실행 표현식을 사용하여 지수로 연동할 수 있습니다 .

예제

- **함께 사용되는 기본 산술 연산**

곱셈과 나눗셈이 덧셈과 뺄셈보다 먼저 수행되며 표현식을 대괄호로 묶을 수 있습니다. 데이터 유형 REAL 에는 연산자 DIV 와 MOD 를 사용할 수 있습니다 .

프로그래밍	비고
DO $AC_PARAM[3] = $A_INA[1]-$AA_IM[Z1]	; ; 두 주 실행 변수의
	; 뺄셈
WHENEVER $AA_IM[x2] < $AA_IM[x1]-1.9 DO $A_OUT[5] = 1	; ; 변수에서 상수 뺄셈
DO $AC_PARAM[3] = $INA[1]-4*SIN(45.7 $P_EP[Y])*R4	; 사전 처리 과정에서 계산된 상수수식

- **수학 함수**

프로그래밍	비고
DO $AC_PARAM[3] = COS($AC_PARAM[1])	;
	;

- **실시간 표현식**

프로그래밍	비고
ID=1 WHENEVER ($AA_IM[Y]>30) AND ($AA_IM[Y]<40) DO $AA_OVR[S1]=80	; 위치 범위 선택
ID=67 DO $A_OUT[1]=$A_IN[2] XOR $AN_MARKER[1]	; 두 개의 부울 신호 검사
ID=89 DO $A_OUT[4]=$A_IN[1] OR ($AA_IM[Y]>10)	; 비교 결과 출력

- **색인 관련 주 실행 변수**

프로그래밍	비고
WHEN…DO $AC_PARAM[$AC_MARKER[1]] = 3	;
규칙 위배	;
$AC_PARAM[1] = $P_EP[$AC_MARKER]	;

10.3 동기 동작에 사용되는 주 실행 변수

10.3.1 시스템 변수

기능

시스템 변수를 활용하여 NC 데이터를 판독하거나 기록할 수 있습니다 . 사전 처리 변수와 주 실행 시스템 변수는 명확하게 구분됩니다 . 사전 처리 변수는 항상 사전 처리 시점에 실행됩니다. 반면에, 주 실행 변수는 항상 현재의 주 실행 상태를 기준으로 그 값을 계산합니다.

명칭

일반적으로 , 시스템 변수의 명칭에는 앞에 특수문자 "$" 가 붙습니다 .

사전 처리 변수 :

$M...	기계 데이터
$S...	설정 데이터 , 보호 영역
$T...	공구 관리 데이터
$P...	프로그램된 값 , 사전 처리 데이터
$C...	ISO 의 싸이클 변수에는 주기가 포함됩니다 .
$O...	옵션 데이터
R ...	R 파라미터

주 실행 변수 :

$$A...	실제 주 실행 데이터
$$V...	서보 데이터
$R...	R 파라미터

두 번째 문자는 다음 변수를 사용할 수 있는 옵션을 의미합니다 .

N...	NCK 일반값 (일반적으로 유효한 값)
C...	채널 관련 값
A...	축 관련 값

일반적으로 , 두 번째 문자는 주 실행 변수에만 사용됩니다 . 예를 들어 , '$P_' 와 같은 사전 처리 변수는 두 번째 문자가 없이 실행되는 게 일반적입니다 .

밑줄과 그 뒤에 오는 변수 명칭(일반적으로, 영어 명칭이나 약어)에는 접두어가 앞에 옵니다('$ ' 다음에는 한 , 두자의 문자가 옵니다).

데이터 유형

주 실행 변수는 다음과 같은 데이터 유형으로 구성됩니다 .

INT	접두어 표시가 있는 전체 값에 대한 정수
REAL	합리적 계산에 적합한 실수
BOOL	부울변수의 참과 거짓
CHAR	ASCII 문자
STRING	영숫자 문자로 구성된 문자열
AXIS	축 주소 및 스핀들

사전 처리 변수는 다음과 같은 데이터 유형으로 구성됩니다 .

FRAME	좌표 변환

변수 배열

시스템 변수는 1 차원 ~ 3 차원 배열로 설정할 수 있습니다 .

지원되는 데이터 유형 : BOOL, CHAR, INT, REAL, STRING, AXIS

지수의 데이터 유형 중에는 필요한 경우 정렬이 가능한 INT 와 AXIS 가 있습니다 .

STRING 변수는 2 차원으로만 설정할 수 있습니다 .

배열 정의 예제 :

```
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]
```

주

시스템 변수로 OPI 변수가 있는 경우 , 아무 제약 없이 3 차원 시스템 변수를 표시할 수 있습니다 .

10.3.2 함축적 유형 변환

기능

값 할당 및 파라미터 전송 과정에서 다른 데이터 유형의 변수가 할당되거나 전송됩니다 .

함축적 유형 변환 기능은 내부적으로 값의 유형을 변환합니다 .

변환 가능한 유형

변경 후	REAL	INT	BOOL	CHAR	STRING	AXIS	FRAME
변경 전							
REAL	Yes	yes*	Yes[1])	–	–	–	–
INT	Yes	Yes	Yes[1])	–	–	–	–
BOOL	Yes	Yes	Yes	–	–	–	–

설명

* REAL 에서 INT 로 유형을 변환할 때 , 분수값이 0.5 보다 크거나 같으면 반올림되고 작으면 버리게 됩니다 (비교 : ROUND 함수)
값이 초과된 경우 알람이 출력됩니다 .

1) 값 <> 0 은 참에 , 값 == 0 은 거짓에 해당됩니다 .

결과

REAL 또는 INTEGER 에서 BOOI 로 유형 변환

결과 : BOOL = TRUE — REAL 또는 INTEGER 값이 0 과 **같지 않은** 경우

결과 : BOOL = FALSE — REAL 또는 INTEGER 값이 0 과 **같은** 경우

BOOI 에서 REAL 또는 INTEGER 로 유형 변환

결과 : REAL TRUE — BOOL 값 = TRUE (1) 인 경우

결과 : INTEGER = TRUE — BOOL 값 = TRUE (1) 인 경우

BOOI 에서 REAL 또는 INTEGER 로 유형 변환

결과 : REAL FALSE) — BOOL 값 = FALSE (0) 인 경우

결과 : INTEGER = FALSE — BOOL 값 = FALSE (0) 인 경우

함축적 유형 변환 예제

```
INTEGER 에서 BOOL 로 유형 변환
$AC_MARKER[1]=561
ID=1 WHEN $A_IN[1] == TRUE DO $A_OUT[0]=$AC_MARKER[1]

REAL 에서 BOOL 로 유형 변환
R401 = 100.542
WHEN $A_IN[0] == TRUE DO $A_OUT[2]=$R401

BOOL 에서 INTEGER 로 유형 변환
ID=1 WHEN $A_IN[2] == TRUE DO $AC_MARKER[4] = $A_OUT[1]]

BOOL 에서 REAL 로 유형 변환
R401 = 100.542
WHEN $A_IN[3] == TRUE DO $R10 = $A_OUT[3]
```

10.3.3 GUD 변수

동기 동작이 가능한 GUD 변수

특정 시스템 변수 외에 미리 정의된 전역 동기 동작 사용자 변수 (동기 동작 GUD) 도 동기 동작에서 사용할 수 있습니다 . 사용자가 사용할 수 있는 동기 동작 GUD 항목 개수는 특정 데이터 유형에 따라 설정되며 다음 머신 데이터를 사용하여 액세스할 수 있습니다 .

- MD18660 $MM_NUM_SYNACT_GUD_**REAL**[<x>] = < 번호 >
- MD18661 $MM_NUM_SYNACT_GUD_**INT**[<x>] = < 번호 >
- MD18662 $MM_NUM_SYNACT_GUD_**BOOL**[<x>] = < 번호 >
- MD18663 $MM_NUM_SYNACT_GUD_**AXIS**[<x>] = < 번호 >
- MD18664 $MM_NUM_SYNACT_GUD_**CHAR**[<x>] = < 번호 >
- MD18665 $MM_NUM_SYNACT_GUD_**STRING**[<x>] = < 번호 >

인덱스 <x>는 각 데이터 유형 (REAL, INT 등) 에 대한 동기 동작 GUD의 개수를 지정하기 위해 데이터 블록 (사용 권한) 및 값 < 번호 > 을 지정하는 데 사용됩니다 . 그러면 다음과 같은 명명 규칙에 의한 1 차원적 배열 변수가 각 데이터 유형의 관련 데이터 블록에서 생성됩니다 .
SYG_< 데이터 유형 >< 사용 권한 >[< 인덱스 >]:

인덱스 <x>		데이터 유형 (MD18660 - MD18665)					
	블록	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]
여기서 i = 0 - (< 번호 > - 1) 블록 : _N_DEF_DIR/_N_ ... _DEF(예 : SGUD ⇒ _N_DEF_DIR/_N_**SGUD**_DEF 의 경우)							

속성

동기 동작 GUD 는 다음과 같은 속성을 가지고 있습니다 .

- 동기 동작 GUD 는 동기 동작 및 가공 프로그램 / 사이클에서 읽고 쓸 수 있습니다 .
- 동기 동작 GUD 는 OPI 를 통해 액세스할 수 있습니다 .
- 동기 동작 GUD 는 HMI 상의 ' 파라미터 ' 영역에서 표시됩니다 .
- 동기 동작 GUD 는 HMI 상의 위저드 , 변수 보기 및 변수 로그에서 사용할 수 있습니다 .
- STRING 유형 동기 동작 GUD 의 배열 크기는 고정 값 32 (31 글자 + \0) 로 설정됩니다 .
- 전역 사용자 데이터 (GUD) 에 대한 정의 파일을 직접 생성하지 않았더라도 머신 데이터를 사용하여 정의된 동기 동작 GUD 를 HMI 의 해당 GUD 블록에서 읽을 수 있습니다 .

유의사항
만약 동기 동작 GUD 가 동일한 이름으로 설정되지 않은 경우 (MD18660 - MD18665), 사용자 변수 (GUD, LUD, PUD) 들은 동기 동작 GUD 에 지정된 이름과 동일한 이름으로 지정되어야 합니다 (DEF ... SYG_xy). 이와 같은 사용자 정의 GUD 항목들은 동기 동작에 사용될 수 **없습니다** .

사용 권한

GUD 정의 파일에 정의된 사용 권한은 계속 유효하며 이 GUD 정의 파일에서 정의된 GUD 변수만 참조합니다 .

삭제 동작

특정 GUD 정의 파일의 내용이 재활성화된 경우 , 액티브 파일 시스템에서 이전 GUD 데이터 블록이 먼저 삭제됩니다. 설정된 동기 동작 GUD도 이 시점에 리셋됩니다. 이 프로세스는 HMI 상에서 " 서비스 " > " 정의 및 활성화된 사용자 데이터 (GUD)" 를 통해서도 가능합니다 .

10.3.4 기본 설정 축 식별자 (NO_AXIS)

기능

특정 값에 의해 초기화되지 않은 AXIS 유형의 변수나 파라미터에 기본 설정된 축 식별자를 사용할 수 있습니다 . 그러면 정의되지 않은 축 변수가 이 디폴트 값으로 초기화됩니다 .

"NO_AXIS" 변수를 조회함으로써 초기화되지 않은 유효 축 명칭을 동기화 동작에서 인지합니다 . 그리고 머신 데이터에 의해 초기화되지 않은 축 이름이 설정된 기본 축 이름으로 할당됩니다 .

장비 제조업체

머신 데이터를 사용하여 최소 하나의 유효한 기존의 축 식별자를 반드시 정의하고 미리 할당해야 합니다 . 그러나 유효한 기존의 모든 식별자는 미리 할당할 수 있습니다 . 장비 제조업체의 설명서를 참조하십시오 .

주

정의 과정에서 기본 설정 축 명칭 관련 머신 데이터에 저장된 값에 새로 생성된 변수가 자동으로 주어집니다 .

머신 데이터를 통해 적용할 수 있는 정의에 대한 자세한 내용은 다음 설명서를 참조하십시오 .
참조 :
/FBSY/ 기능 매뉴얼의 동기화 동작

구문

```
PROC UP(AXIS PAR1=NO_AXIS, AXIS PAR2=NO_AXIS)
IF PAR1 <>NO_AXIS...
```

의미

PROC	서브프로그램 정의
UP	인지용 서브프로그램 명칭
PARn	파라미터 n
NO_AXIS	기본 설정 축 식별자로 공식 파라미터 초기화

예 : 메인 프로그램에서 축 변수 정의

프로그램 코드

```
DEF AXIS AXVAR
UP( , AXVAR)
```

10.3.5 동기화 동작 마커 ($AC_MARKER[n])

기능

배열 변수 $AC_MARKER[n] 을 동기화 작동에서 판독하거나 기록할 수 있습니다 . 이 변수는 활성 또는 비활성 파일 시스템의 메모리에 저장할 수 있습니다 .

동기화 동작 변수 : 데이터 유형 INT

$AC_MARKER[n]	채널 관련 마커 / 카운터 , INTEGER 데이터 유형
$MC_MM_NUM_AC_MARKER	동기화 동작용 채널 관련 마커 번호를 설정하는 데 사용되는 기계 데이터
n	변수 0-n 의 배열 색인

마커 변수 판독 및 기록 예제 :

프로그램 코드

```
WHEN ... DO $AC_MARKER[0] = 2
WHEN ... DO $AC_MARKER[0] = 3
WHENEVER $AC_MARKER[0] == 3 DO $AC_OVR=50
```

10.3.6 동기화 동작 파라미터 ($AC_PARAM[n])

기능

동기화 동작 파라미터 $AC_PARAM[n] 은 계산용으로 , 동기화 동작의 중간 메모리처럼 사용됩니다 . 이 변수는 활성 또는 비활성 파일 시스템의 메모리에 저장할 수 있습니다 .

동기화 동작 변수 : 데이터 유형 :REAL

이 매개변수들은 동일한 이름에 속한 각 채널에 한 번씩 존재합니다 .

$AC_PARAM[n]	동기화 동작 관련 산술 변수 (REAL)
$MC_MM_NUM_AC_PARAM	동기화 동작과 관련한 파라미터 번호(최대 20000)를 설정하는데 사용되는 기계 데이터
n	파라미터 0n 의 배열 색인

동기화 동작 파라미터 예제 ($AC_PARAM[n])

프로그램 코드

```
$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 산술 파라미터 ($R[n])

기능

이 정적 배열 정수는 가공 프로그램과 동기화 동작에 계산 용도로 사용됩니다 .

구문

가공 프로그램에서 프로그래밍 :
REAL R[n]
REAL Rn

동기 동작에서 프로그래밍 :
REAL $R[n]
REAL $Rn

산술 파라미터

산술 파라미터 사용이 허용되는 경우 :

- 프로그램 종료 , NC 리셋 , 전원 가동 이후에도 유지되길 원하는 값 저장
- R 파라미터 디스플레이에 저장된 값 표시

예

프로그램 코드	비고
WHEN $AA_IM[X]>=40.5 DO $R10=$AA_MM[Y]	; 동기화 동작에 R10 사용
G01 X500 Y70 F1000	
STOPRE	; 전처리 중지
IF R10>20	; 산술 변수 검사

프로그램 코드
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 NC 기계 데이터 및 NC 설정 데이터 판독 및 기록

기능

동기화 동작의 기계 데이터와 설정 데이터를 판독 및 기록할 수도 있습니다 . 기계 데이터의 배열 요소를 판독 및 기록할 경우 , 프로그래밍 과정에서 색인은 배제됩니다 . 가공 프로그램에서 이런 현상이 발생하는 경우 , **첫 번째** 배열 요소를 판독할 때와 기록할 때 배열의 모든 요소가 값으로 기술됩니다 .

이런 경우 , 동기화 동작에서는 **첫 번째** 요소만 판독 및 기록할 수 있습니다 .

정의

다음 문자가 있는 MD, SD

$: 동기화 동작 해석 시간에 값 판독

$$: 주 실행 시간에 값 판독

전처리 시간에 MD 및 SD 값 판독

MD 및 SD 값은 $ 문자를 사용하여 동기화 동작 내에서 주소가 지정되며 전처리 시간에 검사됩니다 .

```
ID=2 WHENEVER $AA_IM[z]<$SA_OSCILL_REVERSE_POS2[Z]-6 DO $AA_OVR[X]=0
```

여기서 반전 범위 2 는 작동 중 정적 상태를 유지한다고 가정됐으며 오실레이션 용도로 주소가 지정됩니다 .

주 실행 시간에 MD 및 SD 값 판독

MD 및 SD 값은 $ 문자를 사용하여 동기화 동작 내에서 주소가 지정되며 주 실행 시간에 검사됩니다 .

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

여기서는 가공 도중 명령에 의해 반전 위치가 수정될 수 있다고 가정됐습니다 .

주 실행 시간에 MD 및 SD 판독

기록을 위한 접근을 허용하도록 접근 권한 수준이 설정되어 있어야 합니다 . 모든 MD 및 SD 의 활성 상태 목록은 다음 문헌을 참조하십시오 . **참조 :** /LIS/, 목록 (1 권).

기록할 MD 와 SD 는 반드시 $$ 가 앞에 붙은 상태로 주소가 지정되어야 합니다 .

예제

프로그램 코드	비고
ID=1 WHEN $AA_IW[X]>10 DO $$SN_SW_CAM_PLUS_POS_TAB_1[0]=20 $$SN_SW_CAM_MINUS_POS_TAB_1[0]=30	; 소프트웨어 캠의 전환 위치 변경 참고 : 해당 위치에 도달하기 전에 2~3 보간 싸이클로 전환 위치가 반드시 변경되어야 합니다 .

10.3.9 타이머 변수 ($AC_Timer[n])

기능

시스템 변수 $AC_TIMER[n] 을 사용하면 정의된 지연 기간이 지난 이후에 특정 동작이 시작되도록 지정할 수 있습니다 .

타이머 변수 : 데이터 유형 :REAL

$AC_TIMER[n]	데이터 유형 REAL 의 채널 관련 타이머
s	초 단위
n	타이머 변수 색인

타이머 설정

다음과 같은 값 할당에 의해 타이머 변수의 증분이 시작됩니다 .
$AC_TIMER[n]= 값

n:	타이버 변수 번호
값 :	시작값 (일반적으로 '0')

타이머 정지

다음과 같이 음수 (-) 값을 할당하여 타이머 변수의 증분을 중지할 수 있습니다 .
$AC_TIMER[n]=-1

타이머 판독

타이머 변수의 실행 또는 중지 여부에 관계 없이 , 현재 타이머 값을 판독할 수 있습니다 . -1 을 할당하여 타이머 변수가 중지한 후에도 현재 시간 값이 저장되어 판독이 가능합니다 .

예제

디지털 입력을 감지한 후 아날로그 출력 500ms 로 실제값 출력 :

프로그램 코드	비고
WHEN $A_IN[1] == 1 DO $AC_TIMER[1]=0	; 타이머 리셋 후 시작
WHEN $AC_TIMER[1]>=0.5 DO $A_OUTA[3]=$AA_IM[X] $AC_TIMER[1]=-1	

10.3.10 FIFO 변수 ($AC_FIFO1[n] ... $AC_FIFO10[n])

기능

관련 데이터 배열을 저장하는 데 10 개의 FIFO 변수 (순환 버퍼 저장) 를 사용할 수 있습니다 .
데이터 형식 : REAL

용도 :

- 주기적 측정
- 전송 실행

판독 또는 기록 상태에서 각 요소에 접근할 수 있습니다 .

FIFO 변수

사용 가능한 FIFO 변수 번호는 기계 데이터 MD28260 $MC_NUM_AC_FIFO 에 정의됩니다 .

FIFO 변수에 기록할 수 있는 값의 번호는 기계 데이터 MD28264 $MC_LEN_AC_FIFO 를 사용하여 정의됩니다 . 모든 FIFO 변수는 길이가 동일합니다 .

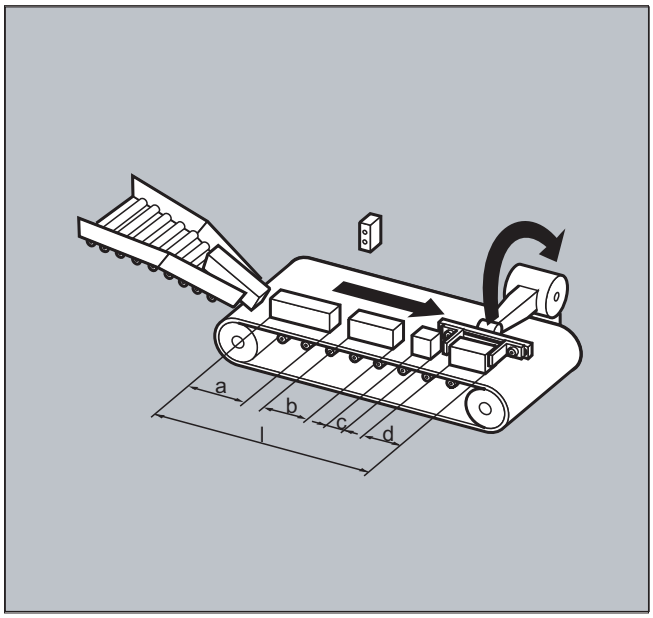
FIFO 요소의 총계는 MD28266 $MC_MODE_AC_FIFO 에 비트 0 이 설정된 경우에 한하여 산출됩니다 .

색인 **0 ~ 5** 는 다음과 같은 특별한 의미를 갖습니다 .

색인	의미	
0	기록 도중 :	새 값이 FIFO 에 저장됩니다 .
	판독 도중 :	가장 오래된 요소를 판독하여 FIFO 에서 제거합니다 .
1	가장 오래된 요소에 접근	
2	가장 최근에 저장된 요소에 접근	
3	FIFO 요소의 총계	
4	FIFO 에서 이용 가능한 요소의 개수 FIFO 의 각 요소에 판독 및 기록 접근을 할당할 수 있습니다 . 요소의 번호 (예 : 첫 번째 FIFO 변수의 번호) 를 리셋하면 FIFO 변수가 리셋됩니다 . $AC_FIFO1[4]=0	
5	FIFO 시작 부분과 관련된 현재의 기록 색인	
6 ~ n 최대	n 번째 FIFO 요소에 접근	

예 : 순환 메모리

생산 작업 과정에서 컨베이어 벨트가 각기 다른 길이의 제품들 (a, b, c, d) 을 이송하는 데 사용됩니다 . 그러므로 컨베이어 벨트는 이 공정과 관련된 개별 제품의 길이에 따라 불규칙한 번호의 제품을 이송하게 됩니다 . 이송 속도가 일정한 상태에서 도착 시간이 각기 다른 제품에 맞게 제품을 벨트에서 제거하는 기능이 적용돼야 합니다 .



프로그램 코드	비고
DEF REAL INTV=2.5	; 배치되는 제품간의 일정 거리
DEF REAL TOTAL=270	; 길이 측정 및 제거 위치간의 거리
EVERY $A_IN[1]==1 DO $AC_FIFO1[4]=0	; 공정이 시작되면 FIFO 이 리셋됩니다 .
EVERY $A_IN[2]==1 DO $AC_TIMER[0]=0	; 제품이 차광막 (Light Barrier) 에 도달하면 시간 측정이 시작됩니다 .
EVERY $A_IN[2]==0 DO $AC_FIFO1[0]=$AC_TIMER[0]*$AA_VACTM[B]	; 제품이 차광막을 떠나면 측정된 시간과 이송 속도를 감안하여 제품 길이를 산정한 후 FIFO 에 저장합니다 .
EVERY $AC_FIFO1[3]+$AC_FIFO1[4]*ZWI>=GESAMT DO POS[Y]=-30 $R1=$AC_FIFO1[0]	; 모든 제품 길이의 총계와 중간 거리가 제품이 배치된 위치와 제거된 지점간의 거리보다 크거나 같은 경우 , 제거 지점의 컨베이어 벨트에서 해당 제품을 제거하고 FIFO 에서 관련 제품의 길이를 판독합니다 .

10.3.11 보간자의 블록 유형에 관한 정보 ($AC_BLOCKTYPE, $AC_BLOCKTYPEINFO, $AC_SPLITBLOCK)

기능

동기와 작동에서 주 실행 시간에 현재 실행 중인 블록에 대한 정보를 확보하는 데 사용할 수 있는 시스템 변수는 다음과 같습니다 .

- $AC_BLOCKTYPE
- $AC_BLOCKTYPEINFO
- $AC_SPLITBLOCK

블록 유형 및 블록 유형 정보 변수

$AC_BLOCKTYPE		$AC_BLOCKTYPEINFO				
값 :		값 :				
0	0 과 같지 않음	T	H	Z	E	의미 :
초기 블록	중간 블록					중간 블록 관련 실행 :
	1	1	0	0	0	내부에서 생성된 블록 , 추가 정보 없음
	2	2	0	0	1	모따기 / 라운딩 : 직선
	2	2	0	0	2	모따기 / 라운딩 : 원
	3	3	0	0	1	WAB: 직선을 따라 접근
	3	3	0	0	2	WAB: 4 분원을 따라 접근
	3	3	0	0	3	WAB: 반원을 따라 접근
						공구 보정 :
	4	4	0	0	1	STOPRE 이후의 접근 블록
	4	4	0	0	2	교차점을 찾지 못한 경우의 연결 블록
	4	4	0	0	3	내부 코너에 점선 원 (TRACYL 전용)
	4	4	0	0	4	외부 코너에 우회 원 (또는 원뿔 절삭)
	4	4	0	0	5	옵셋 억제용 접근 블록
	4	4	0	0	6	TRC 반복 활성화 관련 접근 블록
	4	4	0	0	7	과도한 굴곡으로 인한 블록 분할
	4	4	0	0	8	3D 페이스 밀링 관련 보정 블록 (공구 벡터 || 범위 벡터)
						코너 라운딩 :
	5	5	0	0	1	G641
	5	5	0	0	2	G642
	5	5	0	0	3	G643

$AC_BLOCKTYPE		$AC_BLOCKTYPEINFO				
값 :		값 :				
0	0 과 같지 않음	T	H	Z	E	의미 :
초기 블록	중간 블록					중간 블록 관련 실행 :
	5	5	0	0	4	G644
						TLIFT 블록
	6	6	0	0	1	도피 동작 없이 접선 축의 직선 이송
	6	6	0	0	2	도피 동작 없이 접선 축의 비직선 이송 (폴리노미얼)
	6	6	0	0	3	리프트 동작 , 접선 축 이동 및 리프트 동작 시작 동시 진행
	6	6	0	0	4	리프트 동작 , 특정 리프트 위치에 도달할 때까지 접선 축 시작 안 함 .
						경로 분할 :
	7	7	0	0	1	펀칭 또는 니블링이 비활성 상태인 프로그래밍 경로 분할
	7	7	0	0	2	펀칭 또는 니블링이 활성 상태인 프로그래밍 경로 분할
	7	7	0	0	3	자동으로 내부에서 생성된 경로 분할
						컴파일 싸이클 :
	8	ID 애플리케이션				블록을 생성한 컴파일 싸이클 애플리케이션의 ID

T: 천 단위 숫자
H: 백 단위 숫자
Z: 십 단위 숫자
E: 단 단위 숫자

주

중간 블록이 존재하는 경우 $AC_BLOCKTYPEINFO 에는 항상 천 단위 (T) 로 블록 유형 값이 포함됩니다 . 0 이 아닌 $AC_BLOCKTYPE 에서는 천 단위 숫자가 허용되지 않습니다 .

$AC_SPLITBLOCK	
값 :	의미 :
0	변경되지 않은 프로그래밍 블록 (컴프레서에 의해 생성된 블록 역시 프로그래밍 블록으로 취급)
1	내부에서 생성된 블록이나 축소된 초기 블록이 존재
3	내부에서 생성된 블록이나 축소된 초기 블록으로 구성된 체인 중에서 마지막 블록 사용 가능

예 : 블렌딩 블록 계수

프로그램 코드	비고
$AC_MARKER[0]=0	
$AC_MARKER[1]=0	
$AC_MARKER[2]=0	
...	
	; 블렌딩 블록이 계수되는 동기화 동작 정의
	; 모든 블렌딩 블록은 $AC_MARKER[0] 에서 계수됩니다 .
ID=1 WHENEVER ($AC_TIMEC==0) AND ($AC_BLOCKTYPE==5) DO $AC_MARKER[0]=$AC_MARKER[0]+1	
...	
	; G641 를 사용하여 생성된 블렌딩 블록은 $AC_MARKER[1] 에서 계수됩니다 .
ID=2 WHENEVER ($AC_TIMEC==0) AND ($AC_BLOCKTYPEINFO==5001) DO $AC_MARKER[1]=$AC_MARKER[1]+1	
	; G642 를 사용하여 생성된 블렌딩 블록은 $AC_MARKER[2] 에서 계수됩니다 .
ID=3 WHENEVER ($AC_TIMEC==0) AND ($AC_BLOCKTYPEINFO==5002) DO $AC_MARKER[2]=$AC_MARKER[2]+1	
...	

10.4 동기 동작의 동작들

10.4.1 동기화 동작에서 가능한 동작

동기화 동작에서 동작들은 값 할당 , 함수 또는 파라미터 호출 , 키워드 또는 공정 싸이클로 구성됩니다 . 또한 연산자를 사용하여 복합적인 실행도 가능합니다 .

가능한 용도 :

- IPO 싸이클에서 복잡한 표현식 계산
- 축 이동 및 스핀들 제어
- 온라인으로 동기화 동작의 설정 데이터 변경 및 산정 가능 (예 : PLC 또는 NC I/O 에서 소프트웨어 캠의 위치 및 시간 출력)
- PLC 에 보조 기능 출력
- 별도의 안전 기능 마련
- 이중 동작 , 온라인 공구 옵셋 및 공차 센서 설정
- 모든 작동 모드에서 동작 실행
- PLC 에서 동기화 동작 제어
- 공정 싸이클 실행
- 디지털 및 아날로그 신호 출력
- 보간 싸이클에 동기화 동작의 성과 기록 및 로드 보고서 용도로 위치 컨트롤러의 계산 시간 기록
- 사용자 인터페이스의 진단 기능

동기화 동작	설명
DO $V…= DO $A...=	변수 할당 (서보값) 변수 할당 (주 실행 변수)
DO $AC…[n]= DO $AC_MARKER[n]= DO $AC_PARAM[n]=	특수한 주 실행 변수 동기화 동작 마커 판독 또는 기록 동기화 동작 파라미터 판독 또는 기록
DO $R[n]=	산술 변수 판독 또는 기록
DO $MD...= DO $$SD...=	보간 순간에 즉시 MD 값 판독 주 실행 시간에 SD 값 기록
DO $AC_TIMER[n]= 시작값	타이머
DO $AC_FIFO1[n] …FIFO10[n]=	FIFO 변수
DO $AC_BLOCKTYPE= DO $AC_BLOCKTYPEINFO= DO $AC_SPLITBLOCK=	현재 블록 해석 (주 실행 변수)
DO M-, S 및 H(예 : M07)	M, S 및 H 보조 기능 출력
DO RDISABLE	리드인 비활성 설정
DO STOPREOF	전처리 중지 취소
DO DELDTG	전처리 중지 없이 이동할 거리 고속 삭제

동기화 동작	설명
FTCDEF(폴리노미널 , LL, UL , 계수) DO SYNFCT(폴리노미널 , 출력 , 입력)	폴리노미널 정의 동기화 기능 활성화 : 적응식 제어
DO FTOC	온라인 공구 옵셋
DO G70/G71/G700/G710	위치 지정 작업에 필요한 치수(인치 또는 미터) 단위 정의
DO POS[축]= / DO MOV[축]= DO SPOS[스핀들]=	명령 축 시작 / 배치 / 정지 스핀들 시작 / 배치 / 정지
DO MOV[축]= 값	명령 축 무한 이동 시작 / 배치
DO POS[축]= FA [축]=	축 관련 이송 FA
ID=1 ... DO POS[축]= FA [축]= ID=2 ... DO POS[축]= $AA_IM[축] FA [축]=	동기화 동작에서 위치 지정
DO PRESETON(축 , 값)	실제값 설정 (동기화 동작에서 사전 설정)
ID=1 EVERY $A_IN[1]=1 DO M3 S… ID=2 EVERY $A_IN[2]=1 DO SPOS=	스핀들 시작 / 배치 / 정지
DO TRAILON(FA,LA, 결합 팩터) DO LEADON(FA,LA,NRCTAB,OVW)	결합 축 동작 활성화 리딩 값 결합 활성화
DO MEAWA(축)= DO MEAC(축)=	축 관련 측정 활성화 연속 측정 활성화
DO [배열 n, m]=SET(값 , 값 , ...) DO [배열 n, m]=REP(값 , 값 , ...)	값 목록으로 배열 변수 초기화 동일한 값으로 배열 변수 초기화
DO SETM(마커 번호) DO CLEARM(마커 번호)	대기 마커 설정 대기 마커 삭제
DO SETAL(알람 번호)	싸이클 알람 설정 (별도의 안전 기능)
DO FXS[축]= DO FXST[축]= DO FXSW[축]= DO FOCON[축]= DO FOCOF[축]=	고정 정지점으로 이동 선택 클램핑 토크 변경 모니터링 범위 변경 토크 / 힘이 제한된 상태의 이동 활성화 (모달) FOC 토크 / 힘이 제한된 상태의 이동 비활성화 (특정 블록에 동기화 동작 적용)
ID=2 EVERY $AC_BLOCKTYPE==0 DO $R1=$AC_TANEB	현재 블록 끝 부분의 경로 접선과 프로그래밍된 다음 블록 시작 부분의 경로 접선간의 각도
DO $AA_OVR= DO $AC_OVR= DO $AA_PLC_OVR DO $AC_PLC_OVR DO $AA_TOTAL_OVR DO $AC_TOTAL_OVR	축 오버라이드 경로 오버라이드 PLC 에서 지정된 축 오버라이드 PLC 에서 지정된 경로 오버라이드 적용 후의 축 오버라이드 적용 후의 결과 오버라이드

동기화 동작	설명
$AN_IPO_ACT_LOAD= $AN_IPO_MAX_LOAD= $AN_IPO_MIN_LOAD= $AN_IPO_LOAD_PERCENT= $AN_SYNC_ACT_LOAD= $AN_SYNC_MAX_LOAD= $AN_SYNC_TO_IPO=	실제 IPO 계산 시간 IPO 최장 계산 시간 IPO 최단 계산 시간 IPO 클록 싸이클에 비례한 실제 IPO 계산 시간 전체 채널에서 동기화 동작의 실제 계산 시간 전체 채널에서 동기화 동작의 최장 계산 시간 전체 동기화 동작의 백분률 구성
DO TECCYCLE	공정 싸이클 실행
DO LOCK(n, n, ...) DO UNLOCK(n, n, ...) DO RESET(n, n, ...)	금지 허용 공정 싸이클 리셋
CANCEL(n, n, ...)	가공 프로그램에서 지정 ID 의 모달 동기화 동작 삭제

10.4.2 보조 기능 출력

기능

출력 타이밍

보조 기능들은 해당 동작의 출력 시간에서 동기 동작을 통해 직접 출력됩니다 . 보조 기능에 대해 머신 데이터에서 정의된 출력 타이밍은 동작하지 않습니다 .

조건이 충족된 경우 , 출력 타이밍이 주어집니다 .

예제 :

특정 축 위치에서 절삭유 ON
WHEN $AA_IM[X]>=15 DO M07 POS[X]=20 FA[X]=250

넌모달 동기 동작 (모달 ID 없음) 에 허용되는 키워드

넌모달 동기 동작 (모달 ID 없음) 의 보조 기능은 WHEN 또는 EVERY 키워드를 사용해야만 프로그래밍할 수 있습니다 .

주

다음과 같은 보조 기능들은 동기 동작에 사용할 수 없습니다 :

- M0, M1, M2, M17, M30: 프로그램 중지/종료 (M2, M17, M30은 테크놀로지 사이클에 사용 가능)
- M6 또는 머신 데이터를 통한 공구 교환 명령

예제

프로그램 코드	설명
WHEN $AA_IW[Q1]>5 DO M172 H510	; 축 Q1의 실제값이 5mm를 초과한 경우, PLC에서 도움말 기능 M172 와 H510 을 출력합니다 .

10.4.3 리드인 비활성 설정 (RDISABLE)

기능

RDISABLE 을 사용하면 조건을 충족했을 때 메인 프로그램에서 별도의 블록 처리가 보류됩니다 . 하지만 프로그래밍된 동기 동작은 그대로 실행되고 다음 블록도 준비됩니다 .

리드인 디스에이블의 활성 상태 유지 여부와 관계 없이 RDISABLE 이 포함된 블록의 끝에서 정위치 정지가 실행됩니다 . 또한 시스템이 연속 경로 모드 (G64, G641 ... G645) 인 경우에도 정위치 정지가 실행됩니다 .

적용

RDISABLE 은 예를 들어 보간 싸이클에서 프로그램을 외부 입력 기능으로 시작하는 데 사용할 수 있습니다 .

예제

프로그램 코드	코멘트
WHENEVER $A_INA[2]<7000 DO RDISABLE	; 값 1000이 1 V에 해당하는 것으로 가정할 때 입력 2의 전압이 7 V 이하로 떨어지면 프로그램 처리가 중단됩니다 .
...	
N10 G01 X10	; RDISABLE 은 처리 중 조건이 충족되면 N10 끝에서 실행됩니다 .
N20 Y20	
...	

일반 조건

RDISABLE 이 축 교환에 미치는 영향

RDISABLE 이 축 교환이 수행되는 블록에서 실행되면 축 교환에 의해 시작된 REPOSA 블록에서도 RDISABLE 이 실행됩니다 .

샘플 프로그램 :

```
프로그램 코드
N100 G0 G60 X300 Y300
N105 WHEN TRUE DO POS[X]=20 FA[X]=20000
N110 WHENEVER $AA_IM[X]<>20 DO RDISABLE
N115 G0 Y20
N120 Y-20
N125 M30
```

동기 동작의 결과로 경로에서 X 축을 가져오고 REORG(REPOSA) 이 실행됩니다 . RDISABLE 기능이 REPOSA 블록 상에서 실행됩니다 . X 축은 지정 위치로 이동한 후 N115 의 Y20 으로 이동합니다 .

RELEASE(X) 또는 WAITP(X) 를 N101 에 프로그래밍하면 X 축이 예를 들어 지령 축이 되어 이송하도록 활성화시켜 REORG 를 방지할 수 있습니다 .

프로그램 코드
N100 G0 G60 X300 Y300
N101 RELEASE(X)
N105 WHEN TRUE DO POS[X]=20 FA[X]=20000
...

10.4.4 전처리 중지 취소 (STOPREOF)

기능

명확하게 프로그래밍된 전처리 정지 기능 STOPRE나 동작 중인 동기 동작에 의해 암시적으로 활성화되는 전처리 정지의 경우 , STOPREOF 의 경우 , 조건이 충족된 경우 다음 가공 블록 이후에 STOPREOF 가 전처리 정지를 취소합니다 .

주

STOPREOF 은 반드시 키워드 WHEN 이 동반된 상태에서 넌모달 (ID 번호 없음) 로 프로그래밍돼야 합니다 .

예제

블록 끝에서 고속 프로그램 분기

프로그램 코드		설명
WHEN $AC_DTEB<5 DO STOPREOF	;	블록 끝까지의 거리가 5mm 미만인 경우 , 전처리 중지가 취소됩니다 .
G01 X100	;	직선 보간 실행 후 전처리 정지 제거 .
IF $A_INA[7]>500 GOTOF MARKE1=X100	;	입력 7 에서 5V 의 전압을 초과한 경우 레이블 1 로 점프

10.4.5 이동할 거리 삭제 (DELDTG)

기능

조건에 따라 지정된 축과 경로에 대해 이동할 거리 삭제 기능이 실행될 수 있습니다 .

다음과 같이 사용할 수 있습니다 :

- 이동할 거리 고속 예약 삭제
- 이동할 거리 즉석 삭제

DELDTG의 이동할 거리 예약 삭제를 이용하면 실행 중인 이벤트에 빠르게 응답할 수 있으므로 다음과 같이 시간이 중시되는 경우에 적합합니다 .

- 이동할 거리 삭제 시간과 다음 블록 시작 시간 사이의 공백이 대단히 짧아야 하는 경우
- 이동할 거리 삭제 조건의 충족 가능성이 대단히 높은 경우

주

DELDTG 뒤에 대괄호로 지정되는 축은 오직 **하나의** 포지셔닝 축에 대해서만 유효합니다 .

구문

경로를 대상으로 한 이동할 거리 삭제 :
DO DELDTG

축을 대상으로 한 이동할 거리 삭제 :
DO DELDTG(축 1) DELDTG(축 2) ...

경로를 대상으로 한 이동할 거리 고속 삭제의 예제

프로그램 코드		설명
WHEN $A_IN[1]==1 DO DELDTG		
N100 G01 X100 Y100 F1000	;	입력이 설정될 경우 동작 중지
N110 G01 X…		
IF $AA_DELT>50…		

축을 대상으로 한 이동할 거리 고속 삭제의 예제

프로그램 코드		설명
포지셔닝 축 이동 삭제		
ID=1 WHEN $A_IN[1]==1 DO MOV[V]=3 FA[V]=700	;	축 시작
WHEN $A_IN[2]==1 DO DELDTG(V)	;	이동할 거리 삭제 , MOV=0 을 통해 축 정지
입력 전압에 따라 이동할 거리 삭제 :		
WHEN $A_INA[5]>8000 DO DELDTG(X1)	;	입력 5 의 전압이 8V 를 초과한 경우 , 즉시 축 X1 에 대한 이동할 거리 삭제 경로 동작 계속
POS[X1]=100 FA[X1]=10 G1 Z100 F1000		

그 밖의 정보

이동할 거리 예약 삭제가 실행되는 이송 블록이 끝나면 전처리 중지가 암시적으로 활성화됩니다 .

이동할 거리 고속 삭제가 실행되는 블록이 끝나면 연속 경로 모드나 위치 지정 축 이동이 중단됩니다 .

주

이동할 거리 예약 삭제 :

- 공구 반경 수정 활성 상태에서는 사용할 수 없습니다 .
- 동작은 반드시 비모달식 동기화 동작 (ID 번호 없음) 에서 프로그래밍돼야 합니다 .

10.4.6 폴리노미널 정의 (FCTDEF)

기능

FCTDEF를 사용하여 $y=a_0+a_1x+a_2x^2+a_3x^3$의 형식으로 3차 폴리노미널을 정의할 수 있습니다. 이 폴리노미널은 온라인 공구 옵셋 (FTOC) 및 평가 기능 (SYNFCT) 에서 사용됩니다 .

구문

FCTDEF (폴리노미널 번호 ,LLIMIT,ULIMIT,a_0,a_1,a_2,a_3)

의미

Polynomial_No.	3 차 폴리노미널의 번호
LLIMIT	하한 값
ULIMIT	상한 값
a0, a1, a2, a3	폴리노미널 계수

이 값은 시스템 변수를 통해서도 사용할 수 있습니다 .

$AC_FCTLL[n]	하한 값
$AC_FCTUL[n]	상한 값
$AC_FCT0[n]	a_0
$AC_FCT1[n]	a_1
$AC_FCT2[n]	a_2
$AC_FCT3[n]	a_3

주

시스템 변수 작성

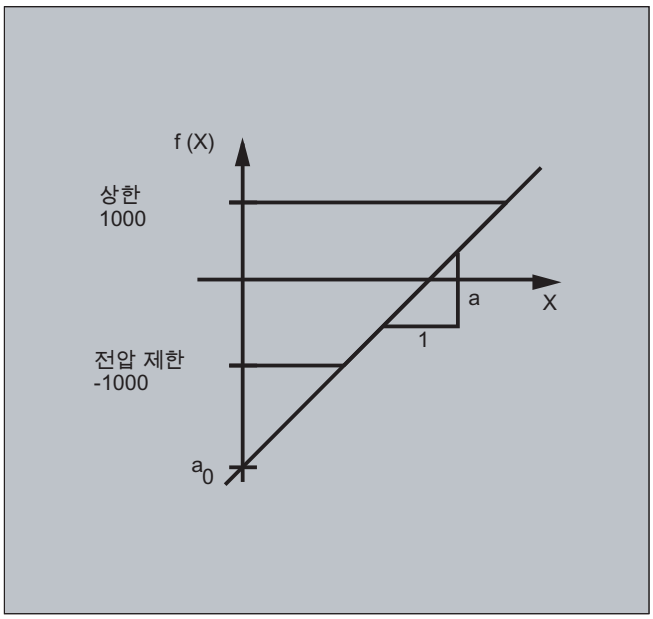
- 가공 프로그램이나 동기 동작에서 시스템 변수를 작성할 수 있습니다 . 가공 프로그램에서 시스템 변수를 작성하는 경우 , 작성되는 값이 블록과 동기될 수 있도록 하기 위해 STOPRE 를 프로그래밍해야 합니다 .
- $AC_FCTLL[n], $AC_FCTUL[n], $AC_FCT0[n] ~ $AC_FCTn[n] 시스템 변수는 동기 동작에서 변경할 수 있습니다 .

동기 동작에서 시스템 변수를 기록하는 경우 , 폴리노미널 계수 및 기능값 한계가 즉시 적용됩니다 .

직선 단면의 폴리노미널 예제

상한값 1000, 하한값 -1000, 세로 단면 a_0=$AA_IM[X] 및 직선 경사도 1 인 폴리노미널 :

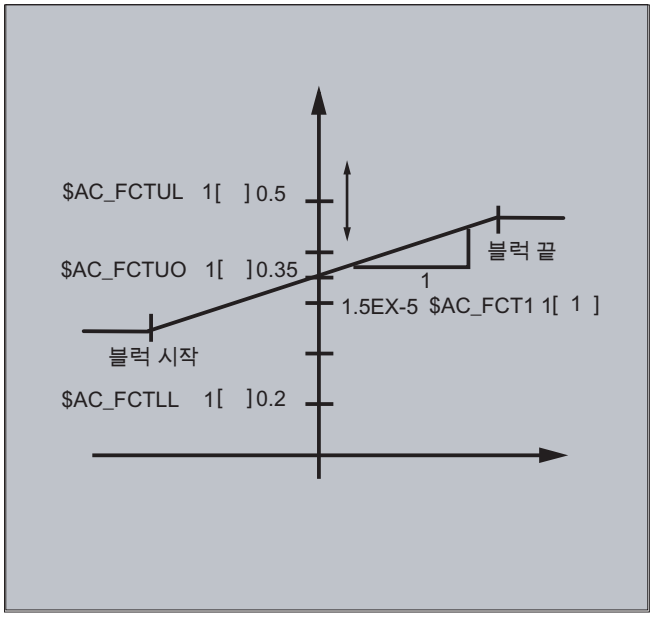
FCTDEF(1, -1000,1000,$AA_IM[X],1)



레이저 출력 제어 예제

폴리노미널 정의를 활용할 수 있는 사례 중 하나로 레이저 출력 제어를 손꼽을 수 있습니다 .

레이저 출력 제어의 의미 :
예를 들어 , 아날로그 출력이 경로 속도에 영향을 미칩니다 .



프로그램 코드	설명
$AC_FCTLL[1]=0.2	; 폴리노미널 계수 정의
$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	; 상한 변경
ID=2 DO SYNFCT(1,$A_OUTA[1],$AC_VACTW)	; $AC_VACTW 에 저장된 경로 속도에 따라 아날로그 출력 1 을 통해 레이저 출력 제어

주

위에 정의된 폴리노미널은 SYNFCT 와 함께 사용됩니다 .

10.4.7 동기화 기능 (SYNFCT)

기능

SYNFCT 는 입력 변수를 사용하여 산출된 폴리노미널 3 등급의 출력값을 계산합니다 . 결과는 출력 변수에 나타나며 최대 및 최소 한계값이 존재합니다 .

산정 기능 사용 분야는 다음과 같습니다 .

- AC 제어 (적응식 제어)
- 레이저 출력 제어
- 위치 피드포워드 (Feed-forward)

구문

SYNFCT(폴리노미널 번호 , 주 실행 변수 출력 , 주 실행 변수 입력)

의미

출력 변수의 경우 ...

- 증가 효과의 변수 선택 가능
- 배가 효과의 변수 선택 가능
- 위치 옵셋 역할을 하는 변수 선택 가능
- 가공 과정에 직접적인 영향을 미치는 변수 선택 가능

.

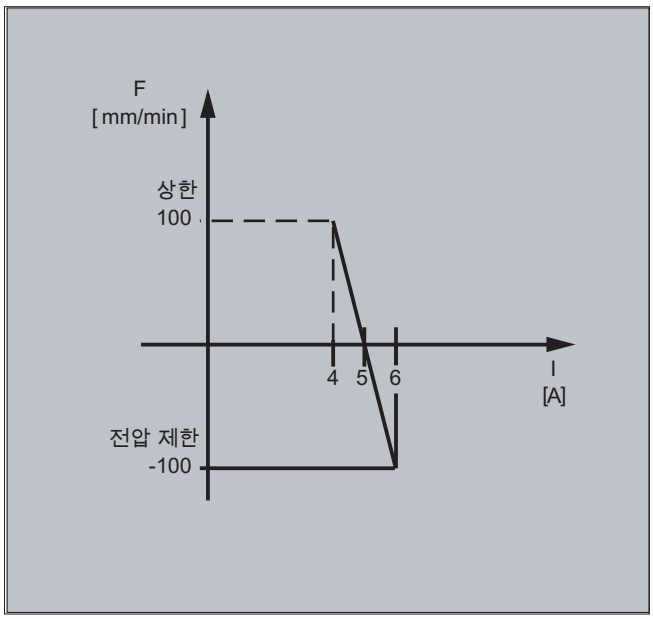
DO SYNFCT	산정 기능 활성화
폴리노미널 번호	FCTDEF 로 정의된 폴리노미널 사용 (소단락 ' 폴리노미널 정의 ' 참조)
주 실행 변수 출력	주 실행 변수 기록
주요 실행 변수 입력	주 실행 변수 판독

적응식 제어 예제 (증가)

프로그래밍된 이송 속도에 미치는 증가 효과

프로그래밍된 이송 속도는 X 축(절입 축)의 전류를 사용하여 증가식으로 제어할 수 있습니다.

이송 송도는 오직 +/- 100 mm/ 분의 단위로만 변경해야 하면 전류는 5A 를 기준으로 +/-1A 씩 변경됩니다 .



1. 폴리노미널 정의

계수 산정

$y = f(x) = a_0 + a_1x + a_2x^2 + a_3x^3$

a_1 = -100mm/1 min A

a_0 = -(-100)*5 =500

$a_2 = a_3 = 0$ (정사각 및 정육면체 요소 없음)

상한값 = 100

하한값 = -100

결과 :

FCTDEF(1,-100,100,500,-100,0,0)

2. AC 제어 활성화

ID=1 DO SYNFCT(1,$AC_VC,$AA_LOAD[x])

$AA_LOAD[x] 를 통해 전류 축 부하 (최대 드라이브 전류의 백분률) 를 판독하고 위에 정의된 폴리노미널로 경로 이송 속도 오버라이드를 계산합니다 .

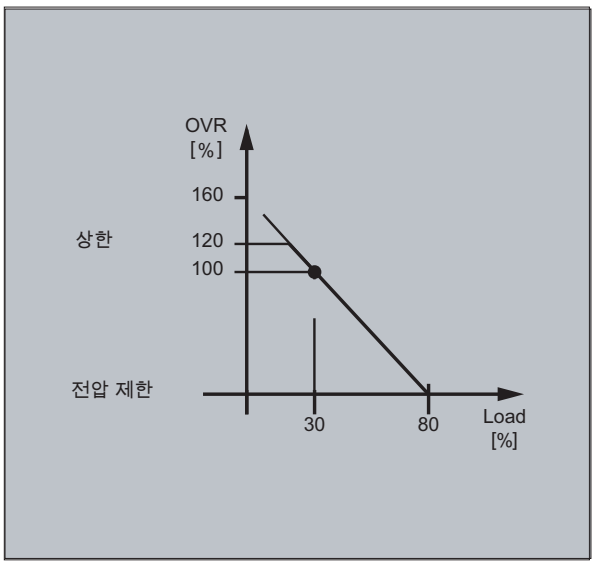
적응식 제어 예제 (배가)

프로그래밍된 이송 속도에 배가 방식 적용

목적은 프로그래밍된 이송 속도에 배가 방식을 적용하는 것입니다 . 이송 속도는 드라이브의 부하에 따라 특정 속도를 초과하지 않아야 합니다 .

- 80% 의 드라이브 부하에서 이송 속도 정지 : 오버라이드 = 0
- 30% 의 드라이브 부하에서 , 프로그래밍된 이송 속도로 이송 가능 : 오버라이드 = 100%.

이송 속도는 20% 까지 초과 가능 :
최대 오버라이드 = 120%.



1 차 폴리노미널 정의

계수 산정

$y = f(x) = a_0 + a_1x + a_2x^2 + a_3x^3$

$a_1 = -100\%/(80-30)\% = -2$

$a_0 = 100 + (2*30) = 160$

$a_2 = a_3 = 0$ (정사각 및 정육면체 요소 없음)

상한값 = 120

하한값 = 0

결과 :

FCTDEF(2,0,120,160,-2,0,0)

2. AC 제어 활성화

ID=1 DO SYNFCT(2,$AC_OVR,$AA_LOAD[x])

$AA_LOAD[x] 를 통해 전류 축 부하 (최대 드라이브 전류의 백분률) 를 판독하고 위에 정의된 폴리노미널로 이송 속도 오버라이드를 계산합니다 .

10.4.8 폐 - 루프 공차 센서의 제한 교정 ($AA_OFF_MODE)

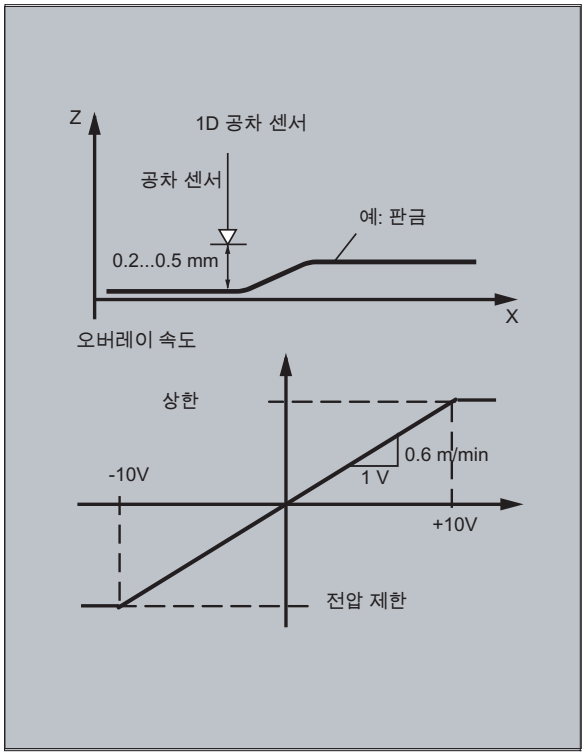
주

이 기능은 SINUMERIK 828D 에 사용할 수 없습니다 !

기능

다음과 같은 한계 범위 검사로 간격 값의 통합 계산을 실시할 수 있습니다 .

$AA_OFF_MODE = 1



유의사항
상위 제어 루프의 루프 게인은 보간 싸이클의 설정에 따라 달라집니다 . 해결책 : IPO 클록 싸이클에 대한 MD 를 확인하여 이를 계산에 반영합니다 .

주

12 ms Ipo 싸이클의 경우 MD32020 JOG_VELO 를 사용하여 상위 보간기의 속도를 제한합니다 .

속도 계산 공식 :

$$\frac{0.120\text{mm}}{0.6\text{ms}} / \text{mV} = 0.6 \frac{\text{m}}{\text{min}} / \text{V}$$

예제

서브프로그램 "AON": 간격 제어 ON

프로그램 코드	설명
PROC AON	
$AA_OFF_LIMIT[Z]=1	; 한계값 지정
FCTDEF(1, -10, +10, 0, 0.6, 0.12)	; 폴리노미얼 정의
ID=1 DO SYNFCT(1,$AA_OFF[Z],$A_INA[3])	; 간격 제어 동작
ID=2 WHENEVER $AA_OFF_LIMIT[Z]<>0 DO $AA_OVR[X] = 0	; 한계 범위를 초과한 경우 X 축 제한
RET	
ENDPROC	

서브프로그램 "AOFF": 간격 제어 OFF

프로그램 코드	설명
PROC AOFF	
CANCEL(1)	; 동기 동작 삭제 , 간격 제어
CANCEL(2)	; 한계 범위 검사 취소
RET	
ENDPROC	

메인 프로그램 "MAIN"

프로그램 코드	설명
AON	; 간격 제어 ON
...	
G1 X100 F1000	
AOFF	; 간격 제어 OFF
M30	

그 밖의 정보

기본 좌표계에서 위치 옵셋

시스템 변수 $AA_OFF[축] 을 사용하여 해당 채널의 모든 축에 동작을 중첩시킬 수 있습니다 . 이것이 기본 좌표계에서 위치 옵셋 역할을 합니다 .

이런 방법으로 프로그래밍된 위치 옵셋은 프로그램에 의한 축의이동 여부에 관계 없이 , 관련 축에서 즉시 오버레이됩니다 .

주 실행 변수 출력 제한 :

절대값을 제한함으로써 설정 데이터 SD43350 $SA_AA_OFF_LIMIT에 저장된 값으로 수정(주 실행 벼수 출력) 할 수 있습니다 .

공차 중첩 형식은 기계 데이터 MD36750 $MA_AA_OFF_MODE 를 이용하여 정의됩니다 .

값	의미
0	비례 산정
1	통합 산정

시스템 변수 $AA_OFF_LIMIT[축] 을 사용하면 수정값이 한계 범위 내에 있는지에 관계 없이 방향 별로 값을 산정할 수 있습니다 . 동기화 동작 시점이나 한계값에 도달했을 때 시스템 변수 사용을 중단할 수 있으므로 축 정지나 알람 설정과 같은 작업에 활용할 수 있습니다 .

0: 범위를 벗어난 옵셋 값

1 + 방향에 도달한 옵셋 값 한계

-1: - 방향에 도달한 옵셋 값 한계

10.4.9 온라인 공구 옵셋 (FTOC)

기능

FTOC 를 사용하면 , 예를 들어 축의 실제값이 될 수 있는 기준값에 따라 FCTDEF 로 프로그래밍된 폴리노미널 다음의 기하 축에 대한 오버레이 이동이 가능합니다 .

FTOC에 대한 기능 정의 FCTDEF(...) 의 계수 a_0이 평가됩니다. 상한값 및 하한값은 a_0에 따라 달라집니다 .

FTOC 를 사용하면 모달 온라인 공구 옵셋 또는 간격 제어를 동기 동작으로 프로그래밍할 수 있습니다 .

이 기능은 동일 채널이나 다른 채널 (가공 채널 및 드레싱 채널) 에서 이뤄지는 공작물 가공 및 연삭 휠의 드레싱에 사용됩니다 .

연삭 휠 드레싱의 일반 조건 및 사양은 PUTFTOCF 를 통한 온라인 공구 옵셋에 적용될 때와 동일한 방식으로 FTOC 에 적용됩니다 (' 온라인 공구 옵셋 (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) [쪽 408]' 참조).

구문

```
FCTDEF(< 기능 >,< 하한 >,< 상한 >,<a0>,<a1>,<a2>,<a3>)
FTOC(< 기능 >,< 기준 값 >,< 공구 파라미터 >,< 채널 >,< 스핀들 >)
...
```

의미

FCTDEF: FCTDEF 는 FTOC 에 대한 폴리노미널 기능을 정의하는 데 사용됩니다 .

파라미터 :

< 기능 >:	폴리노미널 기능 번호	
	유형 :	INT
	범위 :	1 ~ 3
< 하한 >:	하한값	
	유형 :	REAL
< 상한 >:	상한값	
	유형 :	REAL
<a0> - <a3>:	폴리노미널 기능 계수	
	유형 :	REAL

DO FTOC: ' 온라인 공구 옵셋의 연속 모달 작성 ' 기능 실행

파라미터 :

< 기능 >: 폴리노미널 기능 번호

유형 : INT

범위 : 1 ~ 3

참고 :
FCTDEF 에 대한 설정과 일치해야 합니다 .

< 기준 값 >: FCTDEF 로 정의된 폴리노미널 기능을 사용하여 기능 값을 계산할 주 실행 변수 .

유형 : VAR REAL

< 공구 파라미터 >: 옵셋 값이 추가될 마모 파라미터 (길이 1, 2 또는 3) 번호

유형 : INT

< 채널 >: 온라인 공구 옵셋이 적용될 채널 번호

유형 : INT

참고 :
활성 채널에서 옵셋이 적용되지 않도록 하려는 경우에만 채널 번호를 지정해야 합니다 .

< 스핀들 >: 온라인 공구 옵셋이 적용될 스핀들 번호

유형 : INT

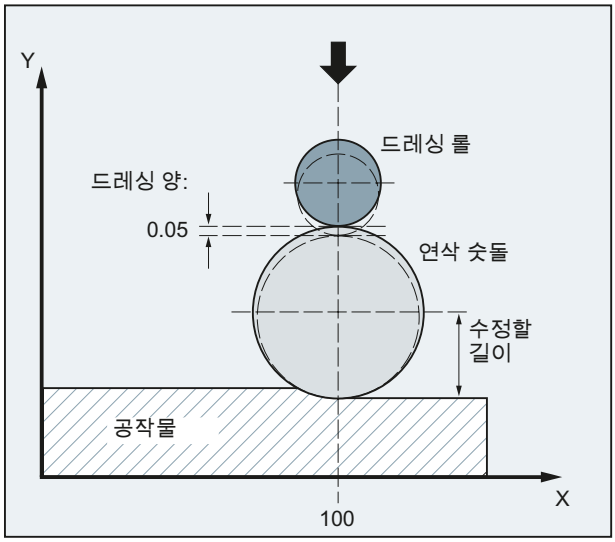
참고 :
현재 사용중인 활성 공구가 아닌 비활성 연삭 휠에 옵셋을 적용하려는 경우에만 스핀들 번호를 지정해야 합니다 .

주

대상 채널에서 FTOCON 이 반드시 활성화되어야 합니다 .

예제

활성 연삭 휠의 길이를 보정합니다 .



프로그램 코드	설명
FCTDEF(1,-1000,1000,-$AA_IW[V],1)	; 기능 정의
ID=1 DO FTOC(1,$AA_IW[V],3,1)	; 온라인 공구 옵셋 선택 : V 축의 실제값은 폴리노미널 1 의 입력값입니다 . 채널 1 에서 보정값으로 활성 상태인 연삭 디스크의 길이 3 에 결과가 추가됩니다 .
WAITM(1,1,2)	; 가공 채널과 동기
G1 V-0.05 F0.01 G91	; 드레싱 이송 동작
G1 V-0.05 F0.02	
...	
CANCEL(1)	; 온라인 옵셋을 선택 취소하십시오 .
...	

10.4.10 온라인 공구 길이 보정 ($AA_TOFF)

기능

시스템 변수 $AA_TOFF[ ] 를 사용하면 실시간으로 3 개의 공구 방향에 따른 유효 공구 길이를 3 차원적으로 오버레이할 수 있습니다 .

3 개의 기하 축 식별자가 색인으로 사용됩니다 . 이렇게 하면 동시에 활성화된 기하 축에 의해 활성화된 보정 방향의 번호가 정의됩니다 .

모든 보정은 동시에 활성화될 수 있습니다 .

구문

```
N... TRAORI
N... TOFFON(X,< 옵셋 값 >)
N... WHEN TRUE DO $AA_TOFF[X]
N... TOFFON(Y,< 옵셋 값 >)
N... WHEN TRUE DO $AA_TOFF[Y]
N... TOFFON(Z,< 옵셋 값 >)
N... WHEN TRUE DO $AA_TOFF[Z]
```

의미

TOFFON:	온라인 공구 길이 보정 **활성화**	
	X, Y, Z:	온라인 공구 길이 보정이 활성화되어야 하는 공구 방향
	< 옵셋 값 >:	보정을 활성화할 때 해당 보정 방향에 대한 옵셋 값을 지정할 수 있고 이 값은 즉시 복구됩니다 .
TOFFOF:	온라인 공구 길이 보정 **리셋** 지정된 보정 방향의 보정 값이 리셋되고 전저리 정지가 시작됩니다 .	
$AA_TOFF[X]=< 값 >:	X 방향 오버레이	
$AA_TOFF[Y]=< 값 >:	Y 방향 오버레이	
$AA_TOFF[Z]=< 값 >:	Z 방향 오버레이	

예

예제 1: 공구 길이 보정 선택

프로그램 코드	설명
N10 TRAORI(1)	; 변환 활성화
N20 TOFFON(Z)	; Z 공구 방향에 대한 온라인 공구 길이 보정 활성화
N30 WHEN TRUE DO $AA_TOFF[Z]=10 G4 F5	; Z 공구 방향에 대해 10 의 TLC 가 보간됩니다 .
N40 TOFFON(X)	; X 공구 방향에 대한 온라인 공구 길이 보정 활성화
N50 ID=1 DO $AA_TOFF[X]=$AA_IW[X2] G4 F5	; X 공구 방향에 대해 축 X2 의 위치에 따른 보정이 실시됩니다 .
...	
	; X 방향의 실제 보정 할당 X 공구 방향에 대해 TLC 가 0 으로 환원됩니다 .
N100 XOFFSET=$AA_TOFF_VAL[X] N120 TOFFON(X,-XOFFSET) G4 F5	

예제 2: 공구 길이 보정 선택 취소

프로그램 코드	설명
N10 TRAORI(1)	; 변환 활성화
N20 TOFFON(X)	; X 공구 방향에 대한 온라인 공구 길이 보정 활성화
N30 WHEN TRUE DO $AA_TOFF[X] = 10 G4 F5	; X 공구 방향에 대해 10 의 TLC 가 보간됩니다 .
...	
N80 TOFFOF(X)	; X 공구 방향의 위치 옵셋이 삭제됩니다 ...$AA_TOFF[X]=0 축이 이동하지 않습니다 . 실제 오리엔테이션에 상응하는 워크의 실제 위치에 위치 옵셋이 추가됩니다 .

10.4.11 이동 위치 지정

기능

가공 프로그램과 관련하여 완전히 동기화되지 않은 상태라도 동기화 동작에서 축의 위치를 지정할 수 있습니다 . 이벤트에 절대적으로 의존하는 주기적 배열이나 작업의 경우 , 동기화 동작에서 위치 지정 축을 프로그래밍하는 게 바람직합니다 . 동기화 동작에서 프로그래밍된 축을 **명령 축**이라고 합니다 .

프로그래밍

참조 :
/PG/ 프로그래밍 안내서 기본사항의 " 경로 세부사항 " 단원
/FBSY/ 기능 설명의 동기화 동작 " 명령 축 시작 "

파라미터

동기화 동작의 위치 지정 작업용 측정 시스템은 G 코드 G70/G71/G700/G710 으로 지정됩니다 .

동기화 동작에서 G 코드를 프로그래밍함으로써 가공 프로그램 내용에 관계 없이 동기화 동작에 대한 INCH/METRIC 검사를 정의할 수 있습니다 .

10.4.12 축 위치 지정 (POS)

기능

가공 프로그램에서 프로그래밍하는 것과는 달리, 포지셔닝 축의 이동은 가공 프로그램 실행에 아무런 영향을 미치지 않습니다 .

구문

POS[axis]= 값

의미

DO POS	명령 축 시작 / 포지셔닝
축	이송될 축의 이름
값	(이송 모드에 따라) 이송될 값

예제

예제 1:

프로그램 코드	설명
ID=1 EVERY $AA_IM[B]>75 DO POS[U]=100	; 이송 모드에 따라 U 축이 100 (inch/mm) 까지 차츰 증가하거나 제로에서 100 (inch/mm) 까지 이동합니다 . ; 메인 실행 변수에서 계산된 거리까지 U 축 이동
ID=1 EVERY $AA_IM[B]>75 DO POS[U]=$AA_MW[V]-$AA_IM[W]+13.5	

예제 2:

프로그램 환경이 포지셔닝 축의 위치 이동에 영향을 미칩니다
(동기 동작의 요소에 G 코드 없음).

프로그램 코드	설명
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	

동기 동작의 요소에서 G71은 프로그램 환경에 관계 없이 포지셔닝 축의 위치 이동을 명확하게 정의합니다 .

프로그램 코드	설명
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 는 항상 10mm 로 지정)
N170 G71 Z10 F10	; Z=10mm X=10mm
N180 M30	

블록 시작 시점에 축 이동이 시작되지 않은 경우 , 동기 동작에서부터 요청된 시작 시점까지 해당 축의 오버라이드가 0 으로 보류됩니다 .

프로그램 코드	설명
WHENEVER $A_IN[1]==0 DO $AA_OVR[W]=0 G01 X10 Y25 F750 POS[W]=1500 FA=1000	
	; 디지털 입력 1 = 0 인 경우 , 포지셔닝 축이 이송되지 않습니다 .

10.4.13 지정된 참조 범위에서 위치 지정 (POSRANGE)

기능

POSRANGE( ) 기능은 현재 보간된 지령치 위치가 지정된 기준 위치의 범위에 존재하는지 파악하는 데 사용할 수 있습니다 . 위치는 지정 가능한 좌표계를 참조하여 지정할 수 있습니다 .

모듈 축의 실제 축 위치를 산정할 때는 모듈 옵셋이 고려됩니다 .

주

이 기능은 동기 동작에서만 호출할 수 있습니다 . 가공 프로그램에서 호출할 경우 알람 14091 %1 블록 %2가 발생되고 기능이 실행되지 않으며, 인덱스 5와 함께 인덱스: %3이 호출됩니다.

구문

BOOL POSRANGE(Axis, Refpos, Winlimit,[Coord])

의미

BOOL POSRANGE	명령 축의 현재 위치가 지정된 기준 위치의 범위에 있습니다 .
AXIS < 축 >	기계 축 , 채널 축 또는 기하 축의 축 이름
REAL Refpos	Coord 좌표계의 기준 위치
REAL Winlimit	위치 범위의 한계 결과
INT Coord	MCS 가 동작 (옵션). 다음 내용들이 허용됩니다 : 0 = MCS (기계 좌표계) 1 = BCS (베이직 좌표계) 2 = SZS (공작물 제로 시스템) 3 = WCS (공작물 좌표계)

기능값

지정 좌표계의 위치 정보에 따른 현재 지령치

함수값 : TRUE	Refpos(Coord) - abs(Winlimit) ≤ Actpos(Coord) ≤ Refpos(Coord) + abs(Winlimit) 인 경우
함수값 : FALSE	그 밖의 경우

10.4.14 축 시작 / 정지 (MOV)

기능

MOV[축]=값을 사용하면 종료 위치를 지정하지 않고서도 명령 축을 시작할 수 있습니다. 다른 동작 또는 포지셔닝 명령에 의해 다른 이동이 설정되거나 , 정지 명령으로 축이 정지할 때까지 프로그래밍된 방향으로 축이 이동합니다 .

구문

MOV[축] = 값

의미

DO MOV	지령 축 이동 시작
축	시작할 축의 이름
값	이동 / 정지 명령이 시작됩니다 . 신호에 따라 이동 방향이 결정됩니다 . 값의 형식은 INTEGER 입니다 .
값 > 0 (일반적으로 +1)	+ 방향
값 < 0 (일반적으로 -1)	- 방향
값 == 0	축 정지

주

MOV[Axis]=0 으로 인덱스 축이 정지된 경우 , 다음 인덱스 위치에서 축이 정지합니다 .

예제

프로그램 코드	설명
... DO MOV[U]=0	; U 축 정지

10.4.15 축 대체 (RELEASE, GET)

기능

공구 변경을 위해 GET(축) 함수를 사용하여 상응하는 명령 축을 동기화 동작의 특정 동작으로 요청할 수 있습니다 . 그 채널에 배정된 축 유형과 그 시간에 직접 연결된 보간을 $AA_AXCHANGE_TYPE 시스템 변수를 사용하여 조회할 수 있습니다 . 실제 상태와 그 축에 대한 현재 보간 권한을 가진 채널에 따라 다른 공정도 가능합니다 .

공구 변경이 완료되면 RELEASE(축) 를 사용하여 동기화 동작의 한 동작으로 채널에 맞게 이 명령 축을 해제할 수 있습니다 .

장비 제조업체

기계 데이터를 통해 관련 축이 반드시 채널에 배정돼야 합니다 . 기계 제조업체의 사양을 참조하십시오 .

구문

GET(axis[,axis{,...}]) 축 가져오기

RELEASE(axis[,axis{,...}]) 축 해제

의미

DO RELEASE	축을 중립 축으로 해제
DO GET	축 대체를 위해 축 가져오기
축	시작할 축의 이름

예 : 축 대체용 프로그램 순서 (2 개 채널)

첫 번째 채널과 두 번째 채널에서 Z 축이 선언됐습니다 .

첫 번째 채널의 프로그램 순서

프로그램 코드	비고
WHEN TRUE DO RELEASE(Z)	; Z 축이 중립 축으로 전환
WHENEVER($AA_TYP[Z]==1) DO RDISABLE	; Z 축이 프로그램 축인 경우 리드인 비활성
N110 G4 F0.1	
WHEN TRUE DO GET(Z)	; Z 축이 다시 NC 프로그램 축으로 지정
WHENEVER($AA_TYP[Z]<>1) DO RDISABLE	; Z 축이 프로그램 축으로 지정될 때까지 리드인 비활성
N120 G4 F0.1	
WHEN TRUE DO RELEASE(Z)	; Z 축이 중립 축으로 전환
WHENEVER($AA_TYP[Z]==1) DO RDISABLE	; Z 축이 프로그램 축인 경우 리드인 비활성
N130 G4 F0.1	;
N140 START(2)	; 두 번째 채널 시작

두 번재 채널의 프로그램 순서

프로그램 코드	비고
WHEN TRUE DO GET(Z)	; Z 축이 두 번째 축으로 이동
WHENEVER($AA_TYP[Z]==0) DO RDISABLE	; Z 축이 다른 채널에 있는 동안 리드인 비활성 ;
N210 G4 F0.1	
WHEN TRUE DO GET(Z)	; Z 축이 NC 프로그램 축으로 지정
WHENEVER($AA_TYP[Z]<>1) DO RDISABLE	; Z 축이 프로그램 축으로 지정될 때까지 리드인 비활성
N220 G4 F0.1	
WHEN TRUE DO RELEASE(Z)	; 두 번째 채널의 Z 축으로 중립 축으로 전환
WHENEVER($AA_TYP[Z]==1) DO RDISABLE	; Z 축이 프로그램 축인 경우 리드인 비활성
N230 G4 F0.1	
N250 WAITM(10, 1, 2)	; 채널 1 과 동기화

첫 번재 채널의 프로그램 순서 계속

프로그램 코드	비고
N150 WAIM(10, 1, 2)	; 채널 2 와 동기화
WHEN TRUE DO GET(Z)	; Z 축이 이 축으로 이동
WHENEVER($AA_TYP[Z]==0) DO RDISABLE	; Z 축이 다른 채널에 있는 동안 리드인 비활성
N160 G4 F0.1	
N199 WAITE(2)	
N999 M30	; 채널 2 에서 프로그램 종료 대기

예 : 공정 싸이클에서 축 대체

첫 번째 채널과 두 번째 채널에서 U 축($MA_AUTO_GET_TYPE=2)이 선언되고 채널 2에 현재 보간 권한이 주어집니다 . 다음 공정 싸이클은 채널 2 에서 시작됩니다 .

프로그램 코드	비고
GET(U)	; U 축을 채널로 가져오기
POS[U]=100	; U 축이 위치 100 으로 이동할 수 있습니다 .

U 축이 채널 2 로 이동할 때까지는 명령 축 이동 라인 POS[U] 가 실행되지 않습니다 .

순서

축 대체를 위해 알아야 할 축 유형과 관련하여 , GET (축) 동작이 활성화되는 시점에 요청된 축을 시스템 변수 ($AA_AXCHANGE_TYP[<axis>]) 를 통해 판독할 수 있습니다 .

- 0: 축이 NC 프로그램에 배정됩니다 .
- 1: 축이 PLC 또는 활성 상태의 축에 명령 축 또는 오실레이션 축으로 배정됩니다 .
- 2: 다른 채널에 보간 권한이 주어집니다 .
- 3: 축이 중립 축으로 전환됩니다 .
- 4: PLC 에 의해 중립 축이 제어됩니다 .
- 5: 다른 채널에 보간 권한이 주어지고 NC 프로그램을 위해 축을 요청합니다 .
- 6: 다른 채널에 보간 권한이 주어지고 축이 중립 축으로 요청됩니다 .
- 7: PLC 에 사용되거나 명령 축 또는 오실레이션 축으로 배정된 축이 활성화되고 PLC 프로그램을 위해 축을 요청합니다 .
- 8: PLC 에 사용되거나 명령 축 또는 오실레이션 축으로 배정된 축이 활성화되고 축이 중립 축으로 요청됩니다 .

한계 조건

기계 데이터를 통해 관련 축이 반드시 채널에 배정돼야 합니다 .

PLC 에 의해 독자적으로 제어되는 축은 NC 프로그램에 배정할 수 없습니다 .

참조 :
/FB2/ 기능 설명서의 확장 기능 , 위치 지정 축 (P2)

다른 채널에서 축을 요청하려는 경우 GET 사용

GET 동작이 활성화됐을 때 **다른 채널에 축에 대한 기록 권한** (보간 권한 부여) 이 부여된 경우 ($AA_AXCHANGE_TYP[< 축 >] == 2), 축 대체를 통해 그 채널로부터 축을 가져와서($AA_AXCHANGE_TYP[<축>]==6) 요청 중인 채널에 가급적 빨리 그 축을 배정할 수 있습니다 .

그러면 해당 축이 중립 축으로 전환됩니다 ($AA_AXCHANGE_TYP[< 축 >]==3).

요청 중인 채널의 재편성 작업은 이뤄지지 않습니다 .

재편성을 통해 NC 프로그램 축으로 배정

GET 동작이 활성화됐을 때($AA_AXCHANGE_TYP[< 축 >]==6), 축을 중립 축으로 전환하려는 시도가 이미 진행 중인 경우 , NC 프로그램을 위해 축이 요청되고 ($AA_AXCHANGE_TYP[< 축 >]==5) 가급적 빨리 그 축이 NC 프로그램에 배정됩니다 ($AA_AXCHANGE_TYP[< 축 >]==0).

요청된 채널에 이미 배정된 축

재편성을 통해 NC 프로그램 축으로 배정

활성화 시점에 요청된 축이 요청 중인 채널에 이미 배정된 경우 , 현재 상태가 (PLC 에 의해 제어되지 않는) 중립 축의 상태로 지정되고 $AA_AXCHANGE_TYP[< 축 >]==3), 그 축이 NC 프로그램에 배정됩니다 .

PLC 에 의해 제어되는 중립 축 상태의 축

축이 PLC 에 의해 제어되는 중립 축 상태인 경우 ($AA_AXCHANGE_TYP[< 축 >]==4), 그 축이 중립 축으로 요청됩니다 ($AA_AXCHANGE_TYP[< 축 >] == 8). 이로 인해 MD 10722: AXCHANGE_MASK(비트 0 == 0) 의 비트 0 의 값에 따라 채널 간의 자동 축 교체를 위해 축이 잠금 설정됩니다 . 그리고 ($AA_AXCHANGE_STAT[< 축 >] == 1) 에 응답합니다 .

축이 중립 명령 축 / 오실레이션 축으로 활성화되거나 PLC 에 배정됩니다 .

축이 중립 명령 축 / 오실레이션 축으로 활성화되거나 이동을 위해 PLC 에 배정된 경우 , PLC 축은 동시 위치 지정 축이 되고 ($AA_AXCHANGE_TYP[< 축 >]==1), 그 축이 중립 축으로 요청됩니다 ($AA_AXCHANGE_TYP[< 축 >] == 8). 이로 인해 MD 10722: AXCHANGE_MASK(비트 0 == 0) 의 비트 0 의 값에 따라 채널 간의 자동 축 교체를 위해 축이 잠금 설정됩니다 . 그리고 ($AA_AXCHANGE_STAT[< 축 >] == 1) 에 응답합니다 .

새로운 GET 코드는 NC 프로그램에 사용될 축을 요청하게 됩니다 ($AA_AXCHANGE_TYP[< 축 >] changes to == 7).

NC 프로그램에 이미 배정된 축

축이 이미 NC 프로그램에 배정됐거나 ($AA_AXCHANGE_TYP[< 축 >]==0) 배정이 요청된 경우 (예 : NC 프로그램에 의해 축 대체 실행)($AA_AXCHANGE_TYP[< 축 >]==5 또는 $AA_AXCHANGE_TYP[< 축 >] == 7), 상태는 원래대로 유지됩니다 .

10.4.16 축 관련 이송 (FA)

기능

명령 축의 축 관련 이동은 모달로 작동합니다 .

구문

FA[< 축 >]=< 값 >

예제

프로그램 코드	비고
ID=1 EVERY $AA_IM[B]>75 DO POS[U]=100 FA[U]=990	; 이송 속도 고정값 입력
	; 주 실행 변수에서 이송 속도 값 생성 :
ID=1 EVERY $AA_IM[B]>75 DO POS[U]=100 FA[U]=$AA_VACTM[W]+100	

10.4.17 소프트웨어 제한 스위치

기능

G25/G26 으로 프로그래밍된 작업 작업 범위 한계가 설정 데이터 $SA_WORKAREA_PLUS_ENABLE 에 따라 명령 축에 고려됩니다 .

G 코드 WALIMON/WALIMOF 를 사용하여 작업 범위 한계를 활성화 또는 비활성화해도 명령 축에는 아무런 영향을 미치지 않습니다 .

10.4.18 축 좌표

기능

일반적으로 동기화 동작에서 위치 지정 축으로 축이 이동하거나 가공 프로그램에서 축이 이동합니다 .

그러나 동일한 축이 가공 프로그램에서 교대로 이동하거나 동기화 동작에서 경로 축 또는 위치 지정 축으로 이동하는 경우 , 두 축 이동 사이의 이송점이 조정됩니다 .

가공 프로그램에서 명령 축이 이동하는 경우 , 전처리가 재편성돼야 합니다 . 그리고 나면 이로 인해 전처리 중지와 유사하게 가공 프로그램 처리가 중단됩니다 .

가공 프로그램 및 동기화 동작에서 X 축의 교대 이동 예제

프로그램 코드	비고	
N10 G01 **X100** Y200 F1000	;	가공 프로그램에 프로그래밍된 X 축
…		
N20 ID=1 WHEN $A_IN[1]==1 DO	;	디지털 입력이 존재할 경우 동기화 동작에서 위치 지정 시작
POS[X]=150 FA[X]=200		
…		
CANCEL(1)	;	동기화 동작 선택
…		
N100 G01 **X240** Y200 F1000	;	X 가 경로 축이 됩니다 . 디지털 입력이 1 이고 동기화 동작에서 X 의 위치가 지정된 경우 , 축 이동으로 인한 지연 시간이 이동 전에 발생합니다 .

동일 축에 대한 이송 명령 변경 예제 :

프로그램 코드	비고	
ID=1 EVERY $A_IN[1]>=1 DO POS[V]=100 FA[V]=560	;	디지털 입력이 1 보다 크거나 같은 경우 동기화 동작에서 위치 지정 시작
ID=2 EVERY $A_IN[2]>=1 DO POS[V]=$AA_IM[V] FA[V]=790	;	축 추적이 진행되고 두 번째 입력이 설정됩니다. 다시 말해서 , 2 개의 동기화 동작이 동시에 활성화되기 때문에 V 축에 대한 종료 위치 및 피드 속도를 이동하는 동안 즉시 추적하게 됩니다 .

10.4.19 실제값 설정 (PRESETON)

기능

PRESETON (축, 값) 이 실행될 경우, 현재 축 위치는 변경되지 않지만 현재 축 위치에 새 값이 할당됩니다 .

다음의 경우 동기 동작에서 PRESETON 을 프로그래밍할 수 있습니다 .

- 가공 프로그램에서 시작된 모듈로 로터리 축
- 동기 동작에서 시작된 모든 명령 축

구문

DO PRESETON(축 , 값)

의미

DO PRESETON	동기 동작의 실제값 설정
축	컨트롤 제로 (Control Zero) 가 변경되어야 할 축
값	컨트롤 제로 (Control Zero) 변경에 기준이 되는 값

축 제한사항

변환과 관련된 축에 대해서는 PRESETON 을 프로그래밍할 수 없습니다 .

하나의 동일 축은 오직 다른 시간에만 가공 프로그램 및 동기 동작에서 이동할 수 있습니다 . 이런 이유로, 동일한 축이 첫 번째로 동기 동작에 프로그래밍된 경우, 가공 프로그램에서 축을 프로그래밍하는 데 지연이 발생할 수 있습니다 .

동일 축이 교대로 사용되는 경우, 두 축 이동 사이의 이송점이 조정됩니다. 이를 위해서는 가공 프로그램 실행이 중단돼야 합니다 .

예제

축의 컨트롤 제로 이동

프로그램 코드	설명
WHEN $AA_IM[a] >= 89.5 DO PRESETON(a4,10.5)	; + 방향으로 축의 컨트롤 제로 위치를 10.5 만큼 (inch 또는 mm) 이동

10.4.20 축 컨테이너 회전에 대한 인에이블 취소 (AXCTSWEC)

기능

AXCTSWEC 명령을 사용하면 이미 내려진 축 컨테이너 회전 인에이블 신호를 다시 취소할 수 있습니다 . 이 명령은 재구성을 통해 전처리 정지를 실행시킵니다 (STOPRE).

효율성

채널에서 축 컨테이너 회전 인에이블 신호를 다시 취소할 수 있도록 하려면 다음 조건을 만족해야 합니다 .

- 채널에서 축 컨테이너 회전이 이미 인에이블된 상태여야 합니다 .
  - AXCTSWE(< 축 컨테이너 >)
  - $AC_AXCTSWA[< 축 컨테이너 >] == 1
- 축 컨테이너 회전이 시작된 상태가 아니어야 합니다 .
  - $AN_AXCTSWA[< 축 컨테이너 >] == 0

인에이블 신호를 성공적으로 취소하기 위한 피드백 신호로 채널별 시스템 변수를 리셋해야 합니다 .

$AC_AXCTSWA[< 축 컨테이너 >] == 0

참고 자료 :
변수 체계에 대해서는 " 축 컨테이너 (AXCTSWE, AXCTSWED, AXCTSWEC) [쪽 675]" 를 참조하십시오 .

구문

DO AXCTSWEC(< 축 컨테이너 >)

중요도

DO AXCTSWEC:	채널의 축 컨테이너 회전에 대한 인에이블 취소	
< 축 컨테이너 >:	축 컨테이너의 식별자 가능한 데이터 :	
	CT< 컨테이너 번호 >:	축 컨테이너 번호는 CT 문자 조합에 추가됩니다 . 예 : CT3
	< 컨테이너 이름 >:	MD12750 $MN_AXCT_NAME_TAB 를 사용하여 설정한 축 컨테이너의 개별적인 이름 예 : A_CONT3
	< 축 이름 >:	관련 채널에서 알려진 컨테이너 축의 축 이름

예

샘플 프로그램 :

프로그램 코드	코멘트
N100 Id=1 DO CTSWEC	; 아래 공정 싸이클 참조
; init	
NEXT:	
N200 G0 X30 Z1	
N210 G95 F.5	
N220 M3 S1000	
N230 G0 X25	
N240 G1 Z-10	
N250 G0 X30	
N260 M5	
N270 AXCTSWE(S1)	; 채널에 지정된 컨테이너 스핀들 S1 을 통해 축 컨테이너 회전을 인에이블합니다 .
N280 GOTO NEXT	

공정 싸이클 CTSWEC:

프로그램 코드	코멘트
CTSWEC PROC(_ex_CT="CT1" STRING _ex_CTsl_BITmask=1H LONG _ex_CT_SL_Number=1 _ex_WAIT_number_of_IPOs=1000) DISPLOF ICYCOF	
DEFINE _ex_CT[3]	; 컨테이너 이름
DEFINE _ex_CTsl_BITmask	; 슬롯 비트
DEFINE _ex_CT_SL_Number LONG =1	
DEFINE _ex_WAIT_number_of_IPOs INTEGER =1000	
DEFINE _ex_number_of_IPOs $AC_MARKER[0]	
N110 IF ($A_STOP_COND[0] OR $A_STOP_COND[1] OR $A_STOP_COND[2] OR $A_STOP_COND[3] OR $A_STOP_COND[4] OR $A_STOP_COND[5] OR $A_STOP_COND[6] OR $A_STOP_COND[7] OR $A_STOP_COND[8] OR $A_STOP_COND[9] OR $A_STOP_COND[10]) is true))	
N120 _ex_number_of_IPOs=_ex_number_of_IPOs+1	; 1000 IPO 싸이클 이상 보류 조건을 사용할 수 있는 경우
N130 IF _ex_number_of_IPOs >= _ex_WAIT_number_of_IPOs AND $AN_AXCTSWEC[_ex_CT] == _ex_CTsl_BITmask	; 그리고 해당 슬롯이 아직 인에이블되지 않은 경우 :
N140 AXCTSWEC	; 인에이블을 취소합니다 .
; … ELSE	; 보류 조건이 해제되는지 여부를 기다립니다 .
N150 ENDIF	
N160 ELSE	
N170 _ex_number_of_IPOs=0	
N180 ENDIF	
N190 RET	

일반 조건

AXCTSWEC 를 호출하기 전에 컨테이너 축 사용

AXCTSWE 에서는 프로그램 처리가 중단되지 않으므로 , 동기 동작 DO AXCTSWEC 를 프로그래밍할 때는 다음 사항을 준수해야 합니다 .

예 :

프로그램 코드	코멘트
N10 AXCTSWE(CT3)	; 축 컨테이너 회전 인에이블 .
N20 AX_A10	; AX_A = 컨테이너 축 . ; 시스템은 축 컨테이너 회전이 완료될 때까지 대기합니다 . $AN_AXCTSWA[CT3] == 0
WHEN <condition> DO AXCTSWEC(AX_A)	; 인에이블 취소 . **영향 없음 !**
N30 G4 F1	

블록 N20을 통한 축 컨테이너의 회전 인에이블을 포함하는 블록 N10 다음에 축 컨테이너의 축 (AX_A) 이 사용되고 이로 인해 시스템은 축 컨테이너 회전이 끝날 때까지 기다리게 되므로 , 동기 동작은 메인 실행에서 프로그램 블록 N30 과 함께 실행되며 아무런 영향을 미치지 않습니다 .

해결책 :

프로그램 코드	코멘트
N11 AXCTSWE(CT3)	; 축 컨테이너 회전 인에이블 .
WHEN <condition> DO AXCTSWEC(AX_A)	; 인에이블 취소 .
N21 ...	; 실행 가능한 NC 블록 .
N31 AX_A10	; 시스템은 축 컨테이너 회전이 완료될 때까지 대기합니다 . $AN_AXCTSWA[CT3] == 0

유의사항
실행 가능한 블록 N21 이 없으면 동기 동작은 메인 실행에서 다음으로 실행 가능한 프로그램 블록 N31 을 포함한 축 컨테이너 회전이 끝난 후에 구현되므로 위 예제와 마찬가지로 아무 영향을 미치지 않습니다 .

10.4.21 스핀들 이동

기능

가공 프로그램과 관련하여 완전히 동기화되지 않은 상태라도 동기화 동작에서 스핀들의 위치를 지정할 수 있습니다 . 이벤트에 절대적으로 의존하는 주기적 배열이나 작업의 경우 , 이런 프로그래밍 형식이 바람직합니다 .

동시에 활성화된 동기화 동작을 통해 명령어들의 충돌 현상이 발생할 경우 , 가장 최근의 스핀들 명령이 우선적으로 적용됩니다 .

스핀들 시작 / 정지 / 위치 지정 예제

프로그램 코드	비고
ID=1 EVERY $A_IN[1]==1 DO M3 S1000	; 회전 방향 및 속도 설정
ID=2 EVERY $A_IN[2]==1 DO SPOS=270	; 스핀들 위치 지정

회전 방향 및 속도 설정 / 스핀들 위치 지정 예제

프로그램 코드	비고
ID=1 EVERY $A_IN[1]==1 DO M3 S300	; 회전 방향 및 속도 설정
ID=2 EVERY $A_IN[2]==1 DO M4 S500	; 새 회전 방향 및 새 속도 지정
ID=3 EVERY $A_IN[3]==1 DO S1000	; 새 속도 지정
ID=4 EVERY ($A_IN[4]==1) AND ($A_IN[1]==0) DO SPOS=0	; 스핀들 위치 지정

10.4.22 결합 동작 (TRAILON, TRAILOF)

기능

동기화 동작에서 결합 동작이 활성화되면 리딩 축이 이동 상태가 될 수 있습니다 . 이 경우 , 설정된 속도까지 추종 축이 가속됩니다 . 속도 동기화 시점의 리딩 축 위치가 결합 축 동작의 시작 위치가 됩니다 . 결합 축 동작의 기능은 ' 경로 이송 특성 ' 단락에 설명되어 있습니다 .

구문

결합 동작 활성화

DO TRAILON(추종 축 , 리딩 축 , 결합 팩터)

결합 축 동작 비활성화

DO TRAILOF(추종 축 , 리딩 축 , 리딩 축 2)

의미

비동기식 결합 동작 활성화	
... DO TRAILON(FA, LA, Kf)	여기서 : FA: 추종 축 LA: 리딩 축 Kf: 결합 팩터
비동기식 결합 동작 비활성화	
... DO TRAILOF(FA, LA, LA2)	여기서 : FA: 추종 축 LA: 리딩 축 , LA2(선택사항): 리딩 축 2(선택사항)
... DO TRAILOF(FA)	추종 축과의 모든 결합이 해제됩니다 .

예제

프로그램 코드	비고
$A_IN[1]==0 DO TRAILON(Y,V,1)	; 디지털 입력이 1 인 경우 , 첫 번째 결합 동작 그룹 활성화
$A_IN[2]==0 DO TRAILON(Z,W,-1)	; 두 번째 결합 축 그룹 활성화
G0 Z10	; 반대 축 방향의 Z 축 및 W 축 절입
G0 Y20	; 동일한 축 방향의 Z 축 및 W 축 절입
...	
G1 Y22 V25	; 결합 동작 축인 V 의 의존적 이동 및 독립적 이동 중첩
...	
TRAILOF (Y,V)	; 첫 번째 결합 축 그룹 비활성화
TRAILOF (Z,W)	; 두 번째 결합 축 그룹 비활성화

TRAILOF 를 이용한 충돌 방지 예제

TRAILOF 함수 호출을 통해 결합 축을 채널 축으로 사용할 수 있도록 결합축이 다시 해제됩니다. 이로 인해 채널이 관련 축을 요청하기 전에 TRAILOF 이 먼저 실행됩니다. 그러나 이 예제는 그 경우에 해당하지 않습니다 .

…

N50 WHEN TRUE DO TRAILOF(Y,X)

N60 Y100

…

이 경우 , 비모달식 동기화 동작이 TRAILOF 로 TRAILOF 와 동기화되지 않기 때문에 축이 조기에 해제되지 않습니다 . 동기화 동작의 ' 구조 기본 정보 ' 단락을 참조하십시오 .
충돌 현상을 피하려면 다음 절차를

따라야 합니다 .

…

N50 WHEN TRUE DO TRAILOF(Y,X)

N55 WAITP(Y)

N60 Y100

10.4.23 리딩 값 결합 (LEADON, LEADOF)

주

이 기능은 SINUMERIK 828D 에 사용할 수 없습니다 !

기능

동기 동작에서 제약 없이 축 리딩 값 커플링을 프로그래밍할 수 있습니다 . 동기 동작에서는 미리 재 동기할 필요 없이 기존 커플링의 커브 테이블을 변경할 수 있습니다 .

구문

마스터 값 커플링 활성화

DO LEADON (추종 축 , 리딩 축 , 커브 테이블 번호 , OVW)

리딩 값 커플링 해제

DO LEADOF(추종 축 , 리딩 축 , 리딩 축 2)

의미

축 리딩 값 커플링 활성화	
...DO LEADON(FA, LA, NR, OVW)	여기서 : FA: 추종 축 LA: 리딩 축 NR: 저장된 커브 테이블의 번호 OVW: 변경된 커브 테이블로 기존 커플링 덮어쓰기 허용
축 리딩 값 커플링 해제	
...DO LEADOF(FA, LA)	여기서 : FA: 추종 축 LA: 리딩 축 (선택사항)
... DO TRAILOF(FA)	리딩 축을 지정하지 않은 축소 형식

동기 동작으로 접근 활성화 (RELEASE)

RELEASE 함수를 호출함으로써 동기 동작에서 커플링될 축이 릴리즈 됩니다 .

예제 :

RELEASE (XKAN)

ID=1 every SR1==1 to LEADON(CACH,XKAN,1)

OVW=0 (기본값))

재동기를 거치지 않고서는 기존 커플링에 대해 새 커브 테이블을 지정할 수 없습니다 . 커브 테이블을 변경하려면 기존 커플링을 사전에 비활성화하고 , 변경된 커브 테이블 번호로 재활성화해야 합니다 . 그러면 커플링 재동기가 이뤄집니다 .

OVW=1 를 이용하여 기존 커플링에 대한 커브 테이블 변경

OVW=1 을 사용하여 기존 커플링에 새 커브 테이블을 지정할 수 있습니다. 재동기는 수행되지 않습니다 . 추종 축은 최대한 빨리 새 커브 테이블에 지정된 위치로 이동됩니다 .

즉각 분할 예제

절삭 공구의 동작 범위를 지속적으로 통과하는 사출 성형 원료는 반드시 동일한 길이의 비율로 절단되어야 합니다 .

X 축 : 사출 성형 원료가 이동하는 축 WCS
X1 축 : 사출 성형 원료의 기계 축 , MCS
Y 축 : 절삭 공구가 사출 성형 원료를 ' 추적 ' 하는 축

절삭 공구 제어 및 절입은 PLC 을 통해 제어된다는 가정을 전제로 합니다 . 사출 성형 원료와 절삭 공구가 동기됐는지 확인하기 위해 PLC 인터페이스의 신호가 검사를 실시할 수 있습니다 .

동작
커플링 활성화 , LEADON
커플링 해제 , LEADOF
실제값 설정 , PRESETON

프로그램 코드	설명
N100 R3=1500	; 절단할 부분의 길이
N200 R2=100000 R13=R2/300	
N300 R4=100000	
N400 R6=30	; 시작 위치 Y 축
N500 R1=1	; 컨베이어 축의 시작 조건
N600 LEADOF(Y,X)	; 모든 기존 커플링 삭제
N700 CTABDEF(Y,X,1,0)	; 테이블 정의
N800 X=30 Y=30	; 값 쌍
N900 X=R13 Y=R13	
N1000 X=2*R13 Y=30	
N1100 CTABEND	; 테이블 정의 종료
N1200 PRESETON(X1,0)	; 시작 시 PRESET
N1300 Y=R6 G0	; 시작 위치 Y 축 (직선)
N1400 ID=1 WHENEVER $AA_IW[X]>$R3 DO PESETON(X1,0)	; 길이 R3 이후 PRESET, 연결 해제 후 새로 시작
N1500 RELEASE(Y)	
N1800 ID=6 EVERY $AA_IM[X]<10 DO LEADON(Y,X,1)	; X가 10보다 작은 경우, 테이블 1 을 통해 Y 에 X 커플링
N1900 ID=10 EVERY $AA_IM[X]>$R3-30 DO EADOF(Y,X)	; 이송된 분할 거리 앞의 30 미만 , 커플링 해제
N2000 WAITP(X)	
N2100 ID=7 WHEN $R1==1 DO MOV[X]=1 FA[X]=$R4	; 지속적으로 이동할 사출 성형 원료의 축 설정
N2200 M30	

10.4.24 측정 (MEAWA, MEAC)

기능

가공 프로그램의 이송 블록 사용과 비교하여 측정 기능을 원하는 때에 활성화 및 비활성화할 수 있습니다 .

측정과 관련한 자세한 내용은 특수 동작 명령어 ' 확장된 측정 기능 ' 을 참조하십시오 .

구문

이동할 거리를 삭제하지 않고 축 관련 측정

MEAWA[축]=(mode, trigger_event_1, ..._4)

이동할 거리를 삭제하지 않고 연속 측정

MEAC[축]=(mode, measurement_memory, trigger_event_1, ..._4)

의미

프로그램 코드	비고		
DO MEAWA	;	축 관련 측정 활성화	
DO MEAC	;	연속 측정 활성화	
축	;	측정 받을 축 이름	
Mode	;	**10 의 단위** 디케이드 (Decade) 설정 0: 활성 상태의 측정 시스템 (모드를 기준으로 한) 측정 시스템 번호 1: 1. 측정 시스템 2: 2. 측정 시스템 3: 2 개의 측정 시스템	**1 의 단위** 디케이드 설정 0: 측정 작업 취소 최대 4 개의 트리거 이벤트 활성화 가능 1: 동시 2: 연속 3: 그러나 2 의 경우에는 시작 단계에서 트리거 이벤트 1 에 대한 모니터링이 이뤄지지 않음
Trigger_event_1 to trigger_event_4	;	: 상승 에지 , 프로브 -1: 하강 에지 , 프로브 1(선택사항) 2: 상승 에지 , 프로브 2(선택사항) -2: 하강 에지 , 프로브 2(선택사항)2	
측정 메모리	;	FIFO 순환 저장 번호	

10.4.25 배열 변수 초기화 (SET, REP)

기능

동기화 동작에서 배열 변수를 특정 값으로 초기화 또는 기술할 수 있습니다 .

주

동기화 동작에 기술될 수 있는 변수에 한하여 초기화가 가능합니다 . 따라서 기계 데이터는 초기화할 수 없습니다 . 축 변수는 NO_AXIS 값으로 지정할 수 없습니다 .

구문

DO ARRAY[n,m]=SET(< 값 1>,< 값 2>,...)
DO ARRAY[n,m]=SET(< 값 1>,< 값 2>,...)

의미

ARRAY[n,m]	프로그래밍된 배열 색인 세트 프로그래밍된 배열 색인 초기화 시작 2D 배열의 경우 두 번째 색인이 먼저 증가합니다 . 축 색인으로는 이런 작업이 이뤄지지 않습니다 .
SET(< 배열 1>,< 배열 2>,...)	값 목록으로 초기화 SET 파라미터를 사용하여 배열이 프로그래밍된 배열 색인부터 앞쪽으로 기술됩니다 . 값으로 할당된 다수의 배열 요소가 프로그래밍됩니다 . 프로그래밍된 상태로 남아 있는 배열 요소보다 존재하는 값이 더 많은 경우 알람이 발동됩니다 .
REP(< 값 >)	동일한 값으로 초기화 배열은 프로그래밍된 배열 색인부터 배열끝까지 기술되고 파라미터 REP의 <값>을 사용하여 반복됩니다.

예제

프로그램 코드	비고
WHEN TRUE DO SYG_IS[0]=REP(0)	; 결과 :
WHEN TRUE DO SYG_IS[1]=SET(3,4,5)	; SYG_IS[0]=0 SYG_IS[1]=3 SYG_IS[2]=4 SYG_IS[3]=5 SYG_IS[4]=0

10.4.26 대기 마커 설정 / 삭제 (SETM, CLEARM)

기능

예를 들어 , 채널들을 서로 조정할 목적으로 동기화 동작에서 대기 마커를 설정 또는 삭제할 수 있습니다 .

구문

DO SETM(< 마커 번호 >)
DO CLEARM(< 마커 번호 >)

의미

SETM	채널을 대상으로 대기 마커를 설정하는 명령 가공 프로그램 및 동기화 동작의 해당 동작 요소에서 SETM 명령을 프로그래밍할 수 있습니다 . 그러면 명령이 적용되는 채널을 대상으로 마커 (< 마커 번호 >) 가 설정됩니다 .
CLEARM	채널을 대상으로 대기 마커를 삭제하는 명령 가공 프로그램 및 동기화 동작의 해당 동작 요소에서 CLEARM 명령을 프로그래밍할 수 있습니다 . 그러면 명령이 적용되는 채널을 대상으로 마커 (< 마커 번호 >) 가 설정됩니다 .
< 마커 번호 >	대기 마커

10.4.27 고장 응답 (SETAL)

기능

동기 동작을 이용하여 고장 응답을 프로그래밍할 수 있습니다 . 그러면 상태 변수가 산정되고 상응하는 동작이 시작됩니다 .

에러 조건에 대해 이뤄질 수 있는 응답 유형은 다음과 같습니다 .

- 축 정지 (오버라이드 =0)
- 알람 설정

  SETAL 을 사용하여 동기 동작에서 사이클 알람을 설정할 수 있습니다 .
- 출력 설정
- 동기 동작에서 이뤄질 수 있는 모든 동작

구문

사이클 알람 설정 :
DO SETAL(< 알람 번호 >)

의미

SETAL	사이클 알람 설정 명령	
< 알람 번호 >	알람 번호	
	사용자의 사이클 알람 범위	65000 ... 69999

예제

프로그램 코드	설명
ID=67 WHENEVER ($AA_IM[X1]-$AA_IM[X2])<4.567 DO $AA_OVR[X2]=0	; 축 X1과 X2 사이의 안전 여유가 너무 작은 경우 축 X2 정지
ID=67 WHENEVER ($AA_IM[X1]-$AA_IM[X2])<4.567 DO SETAL(65000)	; 축 X1과 X2 사이의 안전 여유가 너무 작은 경우 알람 65000 설정

10.4.28 고정 정지점으로 이동 (FXS, FXST, FXSW, FOCON, FOCOF)

기능

' 고정 정지점 이동 ' 기능과 관련한 명령은 가공 프로그램 명령인 FXS, FXST 및 FXSW 를 사용하여 동기 동작 / 고정 사이클에 프로그래밍됩니다 .

이동 없이도 활성화가 이뤄질 수 있으며 토크가 즉시 제한됩니다 . 지령치를 통해 축이 이동한 순간부터 제한 정지 모니터가 활성화됩니다 .

제한된 토크 / 힘으로 이동 (FOC)

이 기능을 이용하면 동기 동작을 통해 언제라도 토크/힘을 변경할 수 있습니다. 또한 이 기능은 모달 또는 블록별로 활성화가 가능합니다 .

구문

FXS[< 축 >]
FXST[< 축 >]
FXSW[< 축 >]
FOCON[< 축 >]
FOCOF[< 축 >]

의미

FXS	디지털 드라이브 (FDD, MSD, HLA) 를 갖춘 시스템에서만 선택 가능
FXST	클램핑 토크 FXST 변경
FXSW	모니터링 범위 FXSW 변경
FOCON	모달 적용 토크 / 힘 제한 활성화
FOCOF	토크 / 힘 제한 해제
< 축 >	축 이름 다음을 사용할 수 있습니다 : • 기하 축 이름 • 채널 축 이름 • 기계 축 이름

주

선택은 한번만 수행할 수 있습니다 .

예제

예제 1: 동기 동작을 이용하여 고정 정지점으로 이동 (FXS) 기능 실행

프로그램 코드	설명
Y 축 :	; 스태틱 동기 동작
활성화 :	
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	
	; $R1=1 을 설정하면 FXS 가 활성화되고 유효 토크가 10% 로 감소하며 고정 정지점으로 이동을 시작합니다 .
N11 IDS=2 WHENEVER ($AA_FXS[Y]==4) DO FXST[Y]=30	; 고정 정지점을 감지한 순간 ($AA_FXS[Y]==4), 토크가 30% 로 감소됩니다 .
N12 IDS=3 WHENEVER ($AA_FXS[Y]==1) DO FXST[Y]=$R0	; 고정 정지점에 도착한 이후부터 R0 에 따라 토크가 제어됩니다 .
N13 IDS=4 WHENEVER (($R3==1) AND $AA_FXS[Y]==1)) DO FXS[Y]=0 FA[Y]=1000 POS[Y]=0	
	; R3 에 따라 선택 취소 및 후진
N20 FXS[Y]=0 G0 G90 X0 Y0	; 정상적인 프로그램 실행 :
N30 RELEASE(Y)	; 동기 동작의 특정 동작을 위해 Y 축이 해제됩니다 .
N40 G1 F1000 X100	; 다른 축의 동작
N50 ...	
N60 GET(Y)	; 경로 그룹에 다시 Y 축 포함

예제 2: 토크 / 힘 제한 활성화 (FOC)

프로그램 코드	설명
N10 FOCON[X]	; 모달 제한 활성화
N20 X100 Y200 FXST[X]=15	; 감소된 토크 (15%) 로 X 이동
N30 FXST[X]=75 X20	; 토크가 75% 로 변경되고 변경된 토크로 X 가 이동합니다 .
N40 FOCOF[X]	; 토크 제한이 비활성화됩니다 .

그 밖의 정보

다중 선택

(FXS[축] = 1)) 를 활성화하고 나서, 잘못된 프로그래밍으로 인해 이 함수가 두 번 이상 호출될 경우 다음과 같은 알람이 출력됩니다 .

알람 20092 " 고정 정지점으로 이동 기능이 아직 활성화 상태입니다 "

$AA_FXS[ ] 또는 전용 비트 메모리 (여기서는 R1) 를 조건에서 산정하는 프로그램이 ' 가공 프로그램 프래그먼트 ' 활성화를 취소합니다 .

프로그램 코드

```
N10 R1=0
N20 IDS=1 WHENEVER ($R1==0 AND
$AA_IW[AX3] > 7) DO R1=1 FXST[AX1]=12
```

블록 관련 동기화 동작

블록 관련 동기화 동작을 프로그래밍함으로써 접근 동작이 진행되는 동안 고정 정지점으로 이동 기능을 연결할 수 있습니다 .

예 :

프로그램 코드	비고
N10 G0 G90 X0 Y0	
N20 WHEN $AA_IW[X] > 17 DO FXS[X]=1	; X 가 17mm 이상의 위치에 도달한 경우 FXS 가 활성화됩니다 .
N30 G1 F200 X100 Y110	

정적 동기화 동작 및 블록 관련 동기화 동작

정적 동기화 동작 및 블록 관련 동기화 동작에서는 동일한 명령어인 FXS, FXST 및 FXSW 를 일반 공정 프로그램 실행에서처럼 사용할 수 있습니다 . 할당 값은 계산에서 산출할 수 있습니다 .

10.4.29 동기화 동작에서 경로 접선 산정

기능

동기화 동작에서 판독할 수 있는 시스템 변수 $AC_TANEB(**블록끝**의 **접선각도**) 는 현재 블록 끝의 경로 접선과 프로그래밍된 다음 블록 시작 부분의 경로 접선 사이의 각도를 계산합니다 .

파라미터

접선 각도는 항상 0.0 ~ 180.0 도 범위의 양수로 출력됩니다 . 주 실행 시간에 다음 블록이 존재하지 않은 경우 -180.0 도가 출력됩니다 .

시스템 변수 $AC_TANEB 는 시스템에 의해 생성된 블록 (중간 블록) 을 판독하지 못합니다 . 시스템 변수 $AC_BLOCKTYPE 는 블록의 프로그래밍 (기본 블록) 여부를 파악하는 데 사용됩니다 .

예제

```
ID=2 EVERY $AC_BLOCKTYPE==0 DO $SR1 = $AC_TANEB
```

10.4.30 현재 오버라이드 산정

기능

현재 오버라이드

동기화 동작에서 (NC 구성요소) 는 다음과 같은 시스템 변수로 판독 및 기록할 수 있습니다 .

$AA_OVR 축 오버라이드

$AC_OVR 경로 오버라이드

.

다음과 같은 시스템 변수를 판독할 수 있도록 PLC 에 의해 정의된 오버라이드가 동기화 동작에서 지원됩니다 .

$AA_PLC_OVR 축 오버라이드

$AC_PLC_OVR 경로 오버라이드

결과 오버라이드

다음과 같은 시스템 변수를 판독할 수 있도록 동기화 동작에 지원됩니다 .

$AA_TOTAL_OVR 축 오버라이드

$AC_TOTAL_OVR 경로 오버라이드

결과 오버라이드를 다음과 같이 계산할 수 있습니다 .

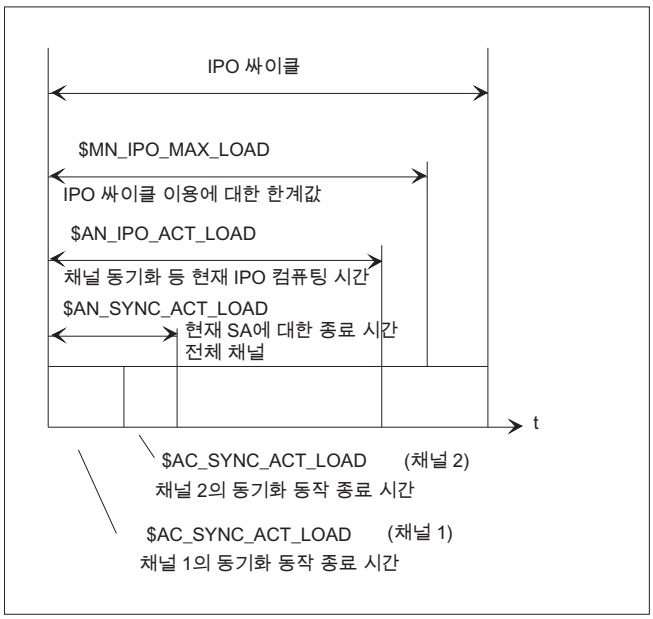
$AA_OVR * $AA_PLC_OVR 또는

$AC_OVR * $AC_PLC_OVR

10.4.31 동기화 동작 사용 점검

기능

보간 싸이클에서 NC에 의해 동기화 동작 해석 및 이동 계산이 이뤄져야 합니다. 아래에 명시된 시스템 변수는 보간 싸이클에 대해 동기화 동작이 취하는 현재 시간 공유 및 위치 제어기의 계산 시간에 관한 정보를 동기화 동작에 제공합니다 .



의미

기계 데이터 $MN_IPO_MAX_LOAD가 0보다 큰 경우 변수는 유효 값만을 취합니다. 그밖에는 HMI 에 의해 발생한 중단 시간을 더 이상 고려하지 않은 순수 계산 시간이 SINUMERIK 파워라인 및 솔루션 라인 시스템 관련 변수에 의해 항상 지정됩니다 . 순수 계산 시간 결과 :

- 동기화 동작 시간 ,
- 위치 제어 시간 및
- HMI 에 의해 발생한 중단 시간을 반영하지 않은 상태에서 남은 IPO 계산 시간

시스템 변수에는 항상 **이전** IPO 싸이클의 값이 포함됩니다 .

$AN_IPO_ACT_LOAD	현재 IPO 계산 시간 (모든 채널의 동기화 동작 포함)
$AN_IPO_MAX_LOAD	IPO 계산 최장시간 (모든 채널의 동기화 동작 포함)
$AN_IPO_MIN_LOAD	IPO 계산 최단시간 (모든 채널의 동기화 동작 포함)
$AN_IPO_LOAD_PERCENT	IPO 싸이클의 백분률 (%) 로 표현된 현재 IPO 계산 시간
$AN_SYNC_ACT_LOAD	전체 채널에서 동기화 동작의 현재 계산 시간
$AN_SYNC_MAX_LOAD	전체 채널에서 동기화 동작의 최장 계산 시간
$AN_SYNC_TO_IPO	(전체 채널에서) 완료된 IPO 계산 시간에 대한 동기화 동작의 점유율 (%)
$AC_SYNC_ACT_LOAD	채널에서 동기화 동작의 현재 계산 시간
$AC_SYNC_MAX_LOAD	채널에서 동기화 동작의 최장 계산 시간
$AC_SYNC_AVERAGE_LOAD	채널에서 동기화 동작의 평균 계산 시간
$AN_SERVO_ACT_LOAD	위치 제어기의 현재 계산 시간
$AN_SERVO_MAX_LOAD	위치 제어기의 최장 계산 시간
$AN_SERVO_MIN_LOAD	위치 제어기의 최단 계산 시간

과부하 통지 관련 변수

기계 데이터 $MN_IPO_MAX_LOAD는 시스템 변수 $AN_IPO_LOAD_LIMIT가 참(TRUE)으로 설정될
순수 IPO 계산 시간 (IPO 싸이클의 백분률로 표시) 을
설정하는 데 사용됩니다 . 현재 부하가 이 한계 이하로 떨어진 경우 , 변수가 다시
거짓 (FALSE) 로 설정됩니다 . 기계 데이터가 0 인 경우 , 전체 진단 기능이 비활성화됩니다 .

$AN_IPO_LOAD_LIMIT 의 점검 기능을 이용하면 기준 초과를 막는 데 필요한 전략을 정의할 수 있습니다 .

10.5 공정 싸이클

기능

동기 동작의 특정 동작으로서 프로그램을 호출할 수 있습니다 . 이를 위해서는 동기 동작에서 동작으로 허용되는 기능들로만 구성돼야 합니다 . 이런 방식으로 구성된 프로그램을 고정 사이클이라고 합니다 .

고정 사이클은 시스템에서 서브프로그램으로 저장됩니다 .

여러 고정 사이클이나 동작을 하나의 채널에서 병렬로 처리할 수 있습니다 .

프로그래밍

고정 사이클을 프로그래밍할 때는 다음 규칙이 적용됩니다 .

- 프로그램 종료는 M02/M17/M30/RET 로 지정됩니다 .
- ICYCOF에 지정된 모든 동작은 동일한 프로그램 레벨에 사용된 사이클을 대기할 필요 없이 하나의 사이클로 처리할 수 있습니다 .
- 하나의 동기 동작당 최대 8 개의 고정 사이클을 차례대로 조회할 수 있습니다 .
- 또한 블록별로 활성화되는 동기 동작에서도 고정 사이클이 지원됩니다 .
- IF 확인 구조 및 GOTO, GOTOF 그리고 GOTOB 점프 명령을 프로그래밍할 수 있습니다 .
- DEF 및 DEFINE 명령을 사용하면 다음 규칙이 블록에 적용됩니다
  - DEF 및 DEFINE 명령을 판독하여 고정 사이클에 전송합니다 .
  - 이로 인해 구문이 잘못되거나 불완전한 경우 알람 메시지가 발동됩니다 .
  - 자체적으로 설정되지 않은 알람 메시지가 없는 경우라도 명령을 판독할 수 있습니다 .
  - 가공 프로그램 사이클처럼 이 명령들이 값 할당에 반영됩니다 .

파라미터 전송

고정 사이클에 파라미터를 전송할 수 있습니다 . 규칙에 준한 ' 값에 따른 호출 ' 파라미터로 전송되는 데이터 형식과 고정 사이클이 호출됐을 때 효력이 발생하는 기본 설정이 고려됩니다. 이것에는 다음이 포함됩니다 :

- 전송 파라미터가 프로그래밍되지 않은 경우의 프로그래밍된 기본값
- 초기값의 표준 파라미터
- 초기화되지 않은 실제 파라미터가 기본값 상태로 전송됩니다 .

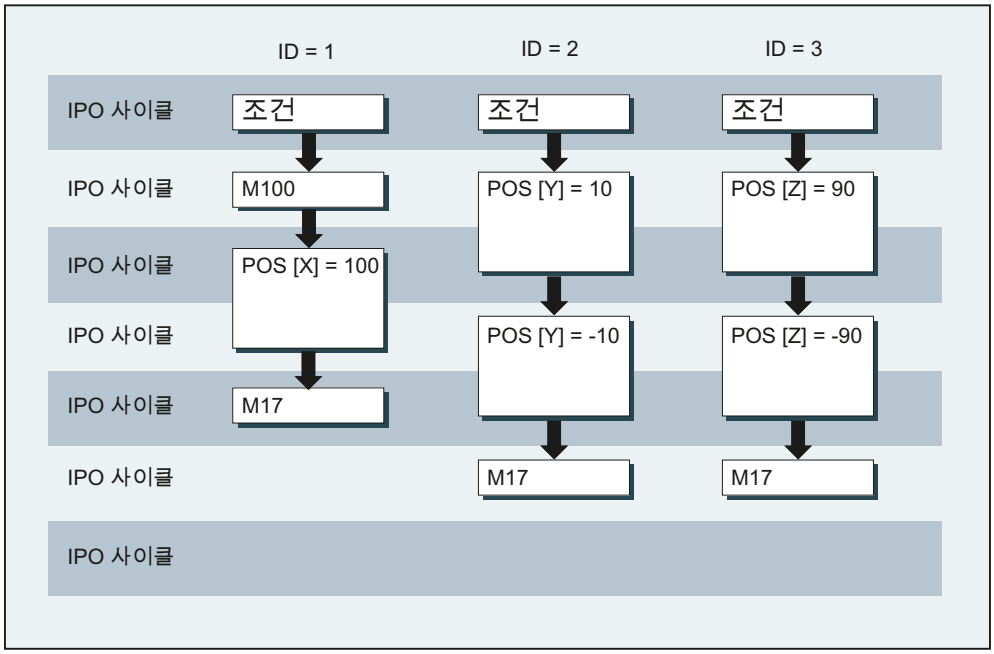
순서

조건을 충족한 즉시 고정 사이클이 시작됩니다 . 고정 사이클의 각 라인은 별도의 IPO 사이클에서 처리됩니다 . 포지셔닝 축을 실행하려면 복수의 IPO 사이클이 필요합니다 . 다른 기능들은 하나의 사이클에서 실행됩니다 . 블록은 고정 사이클에서 순차적으로 실행됩니다 .

동일한 보간 사이클에서 상호 배타적인 동작들이 호출된 경우 , ID 번호가 더 높은 동작이 동기 동작에서 호출되어 활성화됩니다 .

예제

예제 1: 디지털 입력을 설정하여 축 프로그램 시작



메인 프로그램 :

프로그램 코드	설명
ID=1 EVERY $A_IN[1]==1 DO AXIS_X	; 입력 1 이 1 인 경우 축 프로그램 AXIS_X 시작
ID=2 EVERY $A_IN[2]==1 DO AXIS_Y	; 입력 2 가 1 인 경우 축 프로그램 AXIS_Y 시작
ID=3 EVERY $A_IN[3]==1 DO AXIS_Z	; 입력 3 이 1 인 경우 축 프로그램 AXIS_Z 시작
M30	

축 프로그램 AXIS_X:

프로그램 코드

```
M100
POS[X]=100 FA[X]=300
M17
```

축 프로그램 AXIS_Y:

프로그램 코드

```
POS[Y]=10 FA[Y]=200
POS[Y]=-10
M17
```

축 프로그램 AXIS_Z:

프로그램 코드

```
POS[Z]=90 FA[Z]=250
POS[Z]=-90
M17
```

예제 2: 고정 사이클의 다양한 프로그램 순서

프로그램 코드

```
PROC CYCLE
N10 DEF REAL VALUE=12.3
N15 DEFINE ABC AS G01
```

변수 설정 및 / 또는 매크로 설정 없이 두 블록을 모두 판독합니다 .

프로그램 코드

```
PROC CYCLE
N10 DEF REAL
N15 DEFINE ABC G01
```

구문이 올바르게 작성되지 않아 두 블록에서 NC 알람이 발생됩니다 .

프로그램 코드

```
PROC CYCLE
N10 DEF AXIS AXIS1=XX2
```

축 XX2를 인식하지 못한 경우, 알람 12080이 발생됩니다. 그렇지 않을 경우 해당 블록은 알람 및 변수 셋업 없이 판독됩니다 .

프로그램 코드

```
PROC CYCLE
N10 DEF AXIS AXIS1
N15 G01 X100 F1000
N20 DEF REAL VALUE1
```

첫 번째 프로그램 명령 다음에는 정의 동작이 허용되지 않으므로 블록 N20 에서는 항상 알람 14500 이 발생됩니다 .

10.5.1 문맥 변수 ($P_TECCYCLE)

기능

$P_TECCYCLE 변수를 사용하여 프로그램을 동기 동작 프로그램과 전처리 프로그램으로 분할할 수 있습니다. 그리고 나면 (구문의 준하여) 올바르게 작성된 블록 또는 프로그램 순서를 처리하거나 고정 프로그램 사이클로서 블록 또는 프로그램 순서를 처리할 수도 있습니다 .

문맥 변수 해석

$P_TECCYCLE 시스템 변수를 사용하면 고정 사이클에서 프로그램 부분의 특정 문맥 해석을 제어할 수 있습니다 .

```
IF $P_TECCYCLE==TRUE
...                                ;  동기 동작에서 고정 사이클에 대한 프로그램 순서
ELSE
...                                ;  고정 프로그램 사이클에 대한 프로그램 순서
ENDIF
```

주

블록에 잘못되거나 허용되지 않은 프로그램 구문이나 알 수 없는 할당 값이 있는 경우 , 고정 프로그램 사이클에서 알람이 발생됩니다 .

예제

고정 사이클에서 $P_TECCYCLE 의 질의가 있는 프로그램 순서 :

프로그램 코드	설명
PROC CYCLE	
N10 DEF REAL VALUE1	; 고정 사이클에서 판독
N15 G01 X100 F1000	
N20 IF $P_TECCYCLE==TRUE	
...	; 고정 사이클에 대한 프로그램 순서 (변수 VALUE1 이 없음)
N30 ELSE	
...	; 가공 프로그램 사이클에 대한 프로그램 순서 (변수 VALUE1 사용)
N40 ENDIF	

10.5.2 값에 따른 호출 파라미터

기능

값에 따른 호출 파라미터를 사용하여 고정 사이클을 정의할 수 있습니다 . INT, REAL, CHAR, STRING, AXIS 및 BOOL 과 같은 단순한 데이터 형식을 파라미터로 사용할 수 있습니다 .

주

값에 따른 호출로 전송되는 형식 파라미터는 배열이 될 수 없습니다 .

실제 파라미터가 기본 파라미터로 이루어질 수도 있습니다 (' 기본 파라미터 초기화 [쪽 630]' 참조).

구문

ID=1 WHEN $AA_IW[X]>50 DO TEC(IVAL,RVAL,,SVAL,AVAL)

초기화되지 않은 실제 파라미터에 기본값이 전송됩니다 .
ID=1 WHEN $AA_IW[X]>50 DO TEC(IVAL,RVAL,,SYG_SS[0],AVAL)

10.5.3 기본 파라미터 초기화

기능

PROC 명령에서 기본 파라미터가 초기값으로 지원될 수도 있습니다 .

구문

공정 싸이클에서 기본 파라미터 할당

PROC TEC (INT IVAL=1, REAL RVAL=1.0, CHAR CVAL='A', STRING[10] SVAL="ABC", AXIS AVAL=X, BOOL BVAL=TRUE)

현재 파라미터가 기본 파라미터 구성된 경우 , PROC 명령에서 초기값이 전송됩니다 . 이 기능은 가공 프로그램과 동기화 동작에 모두 적용됩니다 .

예제

프로그램 코드	비고
TEC (IVAL, RVAL, SVAL, AVAL)	; CVAL 과 BVAL 에 대해 초기값 적용

10.5.4 공정 싸이클을 제어 처리 (ICYCOF, ICYCON)

기능

ICYCOF 및 ICYCON 언어 명령을 사용하여 고정 사이클의 시간 처리를 제어할 수 있습니다 .

ICYCOF 사이클을 사용하면 하나의 보간 사이클에서 특정 고정 사이클의 모든 블록이 처리됩니다 . ICYCOF 를 사용하면 복수의 사이클을 필요로 하는 모든 동작이 병렬로 처리됩니다 .

적용

ICYCON 를 사용하면 고정 사이클의 처리 시간에 맞게 명령 축 이동을 지연시킬 수 있습니다 . 지연이 필요하지 않은 경우, ICYCOF를 사용하여 대기 시간 없이 하나의 보간 사이클에서 모든 동작을 처리할 수 있습니다 .

구문

고정 사이클의 주기적 처리 시간에 다음과 같은 명령이 적용됩니다 .

ICYCON 고정 사이클에서 개별 블록은 ICYCON 다음의 별도 IPO 사이클로 처리됩니다 .

ICYCOF 고정 사이클에서 개별 블록은 ICYCOF 다음 하나의 IPO 사이클로 처리됩니다 .

주

해당 프로그램 레벨에서는 오직 두 개의 ICYCON 및 ICYCOF 명령만 적용됩니다 . 이 두 명령은 가공 프로그램에서 응답하지 않고 간과되기 쉽습니다 .

ICYCOF 처리 모드 예제

프로그램 코드	설명
IPO 사이클	; 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 계단식 공정 싸이클

기능

최대 8 개의 공정 싸이클을 연속 처리할 수 있습니다 . 그리고 나서 하나의 동기화 동작에서 다수의 공정 싸이클을 프로그래밍할 수 있습니다 .

구문

ID=1 WHEN $AA_IW[X]>50 DO TEC1($R1) TEC2 TEC3(X)

실행 순서

위에 설명한 프로그램에 준하여 왼쪽 작업에서 오른쪽의 작업 순서 (계단식) 로 공정 싸이클이 처리됩니다 . ICYCON 모드에서 싸이클이 처리되는 경우 , 모든 후속 처리 동작이 지연되고 알람이 모든 후속 동작을 취소합니다 .

10.5.6 비모달식 동기화 동작의 공정 싸이클

기능

비모달식 동기화 동작에서도 공정 싸이클을 사용할 수 있습니다 .

공정 싸이클의 처리 시간이 관련 블록의 처리 시간보다 긴 경우 , 블록이 변경될 때 공정 싸이클이 취소됩니다 .

주

공정 싸이클은 블록 변경을 막지 못합니다 .

10.5.7 점검 구조 (IF)

기능

동기화 동작에서 IF 점검 구조를 사용하여 공정 싸이클의 처리 순서의 분기를 프로그래밍할 수 있습니다 .

구문

IF < 조건 >

$R1=1

[ELSE] (선택사항)

$R1=0

ENDIF

10.5.8 점프 명령 (GOTO, GOTOF, GOTOB)

기능

공정 싸이클에서 점프 명령(GOTO, GOTOF, GOTOB)을 사용할 수 있습니다. 알람이 발동되는 것을 막으려면 반드시 서브프로그램에 지정된 레이블이 존재해야 합니다 .

주

라벨 및 블록 번호는 오직 상수만 사용될 수 있습니다 .

구문

무조건 점프

GOTO 라벨 , 블록 번호

GOTOF 라벨 , 블록 번호

GOTOB 라벨 , 블록 번호

점프 명령 및 점프 대상

GOTO	정방향 점프 후 역방향 점프
GOTOF	정방향 점프
GOTOB	역방향 점프
라벨 :	분기 라벨
블록 번호	이 블록의 점프 대상
N100	블록 번호 - 하위 블록
:100	블록 번호 - 메인 블록

10.5.9 해제 , 인에이블 , 리셋 (LOCK, UNLOCK, RESET)

기능

또 다른 모달식 동기화 동작을 사용하여 공정 싸이클의 순서를 해제 , 인에이블 또는 리셋할 수 있습니다 .

구문

LOCK(<n1>,<n2>,...)
UNLOCK(<n1>,<n2>,...)
RESET(<n1>,<n2>,...)

의미

LOCK	동기화 동작 해제 명령 활성 상태의 동기화 동작이 중단됩니다 .
UNLOCK	동기화 동작 인에이블 명령
RESET	공정 싸이클 리셋 명령
<n1>,<n2>,...	해제 , 인에이블 또는 리셋하려는 동기화 동작 또는 공정 싸이클의 식별 번호

동기화 동작 연동

ID 번호 **<n> = 1 ... 64** 에 해당하는 모달식 동기화 동작을 PLC 에서 연동할 수 있습니다 . NCK 에서 관련 조건을 더 이상 검사하지 않고 관련 기능 실행이 중단됩니다 .

PLC 인터페이스에서 하나의 신호로 모든 동기화 동작을 무차별적으로 중단할 수 있습니다 .

주

프로그래밍된 동기화 동작이 표준으로 활성화되어 기계 데이터 설정에 의한 덮어쓰기 / 해제 방지를 위해 보호 받습니다 .

최종 사용자는 장비 제조업체에서 정의한 동기화 동작을 수정할 수 없습니다 .

예

예제 1: 동기화 동작 해제 (LOCK)

프로그램 코드

```
N100 ID=1 WHENEVER $A_IN[1]==1 DO M130
...
N200 ID=2 WHENEVER $A_IN[2]==1 DO LOCK(1)
```

예제 2: 동기화 동작 인에이블 (LOCK)

프로그램 코드

```
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)
```

예제 3: 공정 싸이클 중단 (RESET)

프로그램 코드

```
N100 ID=1 WHENEVER $A_IN[1]==1 DO M130
...
N200 ID=2 WHENEVER $A_IN[2]==1 DO RESET(1)
```

10.6 동기화 동작 삭제 (CANCEL)

기능

CANCEL 명령은 가공 프로그램에서 모달 또는 스태틱 동기 동작을 취소 (삭제) 하는 데 사용할 수 있습니다 .

활성화된 포지셔닝 축이 이동하는 동안 동기 동작이 취소된 경우 , 포지셔닝 축 이동이 중단됩니다 . 포지셔닝 축 이동이 중단되는 것을 원치 않는 경우에는 CANCEL 명령 실행에 앞서 축 이동 거리 삭제로 축 모션을 중단할 수 있습니다 .

구문

CANCEL(<n1>,<n2>,...)

의미

CANCEL:	프로그래밍된 동기 동작 삭제 명령
<n1>,<n2> 등 :	삭제할 동기 동작의 식별 번호 **참고 :** 식별 번호를 지정하지 않으면 **모든** 모달 / 스태틱 동기 동작이 삭제됩니다 .

예제

예제 1: 동기 동작 취소

프로그램 코드	설명
N100 ID=2 WHENEVER $A_IN[1]==1 DO M130	
...	
N200 CANCEL(2)	; 모달 동기 동작 번호 2 삭제

예제 2: 동기 동작을 취소하기 전에 이동할 거리 삭제

프로그램 코드	설명
N100 ID=17 EVERY $A_IN[3]==1 DO POS[X]=15 FA[X]=1500	; 포지셔닝 축 이동 시작
...	
N190 WHEN ... DO DELDTG(X)	; 포지셔닝 축 이동 종료
N200 CANCEL(17)	; 모달 동기 동작 번호 17 삭제

10.7 특정 동작 상태에서의 제어 동작

POWER ON

POWER ON 중에는 동기 동작이 활성화되지 않습니다 . PLC 에 의해 시작된 비동기 서브프로그램으로 (ASUB) 스태틱 동기 동작을 활성화할 수 있습니다 .

모드 변경

키워드 IDS 에 의해 활성화된 동기 동작은 동작 모드 변경 후에도 활성 상태를 유지합니다 . 기타 모든 동기 동작은 동작 모드 변경 (예: 축 포지셔닝) 이후 비활성화되고 위치 변경 후 자동 모드로 복구되면 다시 활성화됩니다 .

RESET

NC 리셋은 모든 넌모달 및 모달 동기 동작을 종료합니다 . 스태틱 동기 동작은 활성 상태를 유지합니다. 스태틱 동기 동작은 새로운 동작을 시작할 수 있습니다. 리셋할 때 지령 축 이동이 활성화되면 이 동작이 취소됩니다 . 이전에 실행된 WHEN 형식의 동기 동작은 리셋 후 다시 처리되지 않습니다 .

RESET 이후의 응답		
동기 동작 / 고정 사이클	**모달 / 넌모달**	**스태틱 (IDS)**
	활성 상태의 동작이 취소되고 동기 동작도 삭제됩니다 .	활성 상태의 동작이 취소되고 고정 사이클이 리셋됩니다 .
축 / 스핀들 포지셔닝	모션이 취소됩니다 .	모션이 취소됩니다 .
속도 제어 상태의 스핀들	$MA_SPIND_ACTIVE_AFTER_RESET==1: 스핀들 활성 상태 유지 $MA_SPIND_ACTIVE_AFTER_RESET==0: 스핀들이 정지됩니다 .	
마스터 값 커플링	$MC_RESET_MODE_MASK, bit13 == 1: 마스터 값 커플링 활성 상태 유지 $MC_RESET_MODE_MASK, bit13 == 0: 마스터 값 커플링 해제	
측정 작업	동기 동작에서 시작된 측정 과정이 취소됩니다 .	스태틱 동기 동작에서 시작된 측정 과정이 취소됩니다 .

NC 정지

스태틱 동기 동작이 NC 정지를 위해 활성 상태를 유지합니다 . 스태틱 동기 동작에서 시작된 이동은 취소되지 않습니다 . 활성 상태의 블록에 속하는 **로컬 프로그램**의 동기 동작은 활성 상태를 유지하고 그로부터 시작된 이동은 중단됩니다 .

프로그램 끝

프로그램 끝과 동기 동작은 서로 영향을 미치지 않습니다 . 현재 동기 동작은 프로그램 종료 후에도 완료됩니다 . M30 블록에 활성화된 동기 동작은 활성 상태를 유지합니다 . 이 상태가 바람직하지 않은 경우에는 CANCEL 로 프로그램을 종료하기 전에 동기 동작을 취소해야 합니다 .

프로그램 종료 이후의 응답		
동기 동작 / 고정 사이클	**모달 / 넌모달 동작 이 취소됨**	**스태틱 동작 (IDS) 은 활성 상태 유지**
축 / 스핀들 포지셔닝	축 / 스핀들이 정지할 때까지 M30 이 지연됩니다 .	모션 계속
속도 제어 상태의 스핀들	프로그램 끝 : $MA_SPIND_ACTIVE_AFTER_RESET==1: 스핀들 활성 상태 유지 $MA_SPIND_ACTIVE_AFTER_RESET==0: 스핀들 정지 동작 모드 변경 시에도 스핀들은 활성 상태 유지	스핀들 활성 상태 유지
마스터 값 커플링	$MC_RESET_MODE_MASK, bit13 == 1: 마스터 값 커플링 활성 상태 유지 $MC_RESET_MODE_MASK, bit13 == 0: 마스터 값 커플링 분리	스태틱 동기 동작에서 시작된 커플링은 활성 상태 유지
측정 과정	동기 동작에서 시작된 측정 과정이 취소됩니다 .	스태틱 동기 동작에서 시작된 측정 과정은 활성 상태를 유지합니다 .

블록 탐색

동기 동작이 블록 탐색 과정에서 수집되어 NC Start 과정에서 검사됩니다. 그리고 나면 필요한 경우 관련 동작이 시작됩니다 . 블록 탐색 중에는 스태틱 동기 동작이 활성 상태를 유지합니다 . 블록 탐색 과정에서 FCTDEF 로 프로그래밍된 폴리노미널 계수가 발견된 경우에는 즉시 활성화됩니다 .

비동기 서브프로그램 ASUB 를 사용하여 프로그램 중단

ASUB 시작 :

모달 및 스태틱 동기 동작은 활성 상태를 유지하고 비동기 서브프로그램에서도 실행됩니다 .

ASUB 종료 :

비동기 서브프로그램이 REPOS 로 재개되지 않는 경우 비동기 서브프로그램에서 수정된 모달 및 스태틱 동기 동작은 메인 프로그램에서 활성 상태를 유지합니다 .

위치 변경 (REPOS)

위치 변경 (REPOS) 이후 , 중단된 블록에서 적용되던 동기 동작이 다시 활성화됩니다 . 잔여 블록을 처리할 때 비동기 서브프로그램에서 변경된 모달 동기 동작은 REPOS 이후에 더 이상 유효하지 않습니다 .

FCTDEF 로 프로그래밍된 폴리노미널 계수는 비동기 서브프로그램과 REPOS 의 영향을 받지 않습니다 . 폴리노미널 계수의 프로그래밍 위치와 관계 없이 REPOS 실행 후에도 비동기 서브프로그램과 메인 프로그램에서 언제라도 사용할 수 있습니다 .

알람에 대한 반응

동기 동작에 의해 시작된 축 및 스핀들 모션은 모션 정지와 관련된 알람에 대한 반응으로 감속됩니다 . 기타 모든 동작 (예 : 출력 설정) 은 계속 실행됩니다 .

동기 동작 자체에서 알람이 발생된 경우 , 처리는 중단되고 이 동기 동작의 후속 동작은 실행되지 않습니다 . 동기 동작이 모달인 경우 , 다음 보간 사이클에서 더 이상 처리되지 않습니다 . 즉 , 알람은 단 한 번만 출력됩니다 . 기타 모든 동기 동작의 처리는 정상적으로 계속됩니다 .

해석기를 정지시키는 알람은 미리 코드화된 블록이 처리된 후에만 적용됩니다 .

고정 사이클이 모션 정지와 함께 알람을 생성하면 관련 사이클의 처리가 중단됩니다 .

오실레이션 11

11.1 비동기 오실레이션 (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB)

기능

오실레이션 축은 오실레이션이 비활성화될 때까지 정의된 피드로 두 개의 반전 지점 1 과 2 사이에서 앞 / 뒤로 이동합니다 .

오실레이션 도중 다른 축들은 원하는 대로 보간할 수 있습니다 . 포지셔닝 축 혹은 경로 이송을 통해 연속 절입이 이루어질 수 있지만 , 오실레이션 이송 및 절입 이송 간에는 **아무런 관계가 없습니다** .

비동기 오실레이션 속성

- 비동기 오실레이션은 블록 제한을 초과하는 특정 축에서 작동합니다 .
- 가공 프로그램을 통해 오실레이션 이동이 블록 방향으로 활성화됩니다 .
- 몇 개 축을 결합하여 보간하거나 오실레이션 경로를 이중화할 수는 없습니다 .

프로그래밍

다음 명령은 가공 프로그램에서 비동기 오실레이션을 활성화 및 제어하는 데 사용할 수 있습니다 .

프로그램 값은 기본 실행 중에 블록 동기를 통해 해당 셋팅 데이터에 입력되며 다시 변경할 때까지 활성화 상태로 유지됩니다 .

구문

OSP1[< 축 >]=< 값 > OSP2[< 축 >]=< 값 >
OST1[< 축 >]=< 값 > OST2[< 축 >]=< 값 >
FA[< 축 >]=< 값 >
OSCTRL[< 축 >]=(< 설정 옵션 >,< 리셋 옵션 >)
OSNSC[< 축 >]=< 값 >
OSE[< 축 >]=< 값 >
OSB[< 축 >]=< 값 >
OS[< 축 >] = 1
OS[< 축 >] = 0

의미

< 축 > 오실레이션 축 이름

OS 오실레이션 활성화 / 비활성화

값 : 1 오실레이션 on

0 오실레이션 off

OSP1 반전 지점 1 위치 정의

OSP2 반전 지점 2 위치 정의

참고 :
증분 이동이 활성화되면 위치는 NC 프로그램에서 마지막으로 프로그래밍된 해당 반전 위치로 증분에 따라 계산됩니다 .

OST1 [s] 의 반전 위치 1 에서 정지 시간 정의

OST2 [s] 의 반전 지점 2 에서 정지 시간 정의

< 값 >:

-2 정위치 정지를 기다리지 않고 보간 계속

-1 일반 정위치 정지 대기

0 미세 정위치 정지 대기

>0 미세 정위치 정지를 대기하고 나서 지정된 정지 시간 동안 대기

참고 :
정지 시간의 단위는 G4 로 프로그래밍한 정지 시간의 단위와 동일합니다 .

FA 이송 속도 정의

이송 속도는 포지셔닝 축에 정의된 속도입니다 . 이송 속도가 정의되지 않은 경우에는 머신 데이터에 저장된 값이 적용됩니다 .

OSCTRL 설정 옵션 및 리셋 옵션 지정

옵션 값 0 - 3 은 비활성화 상태에서 반전 지점의 동작을 암호화합니다 . 0 - 3 사이의 변용 중 하나를 선택할 수 있습니다 . 나머지 설정은 선택한 변용에 따라 원하는 대로 조합할 수 있습니다 . + 기호를 사용하여 여러 옵션을 추가할 수 있습니다 .

< 값 >:

- 0 오실레이션 비활성화 상태에서 다음 반전 지점에서 정지 (표준)

  참고 :
  1 과 2 값이 리셋된 경우에만 사용 가능합니다 .
- 1 오실레이션이 비활성화된 경우 반전 지점 1 에서 정지합니다 .
- 2 오실레이션이 비활성화된 경우 반전 지점 2 에서 정지합니다 .
- 3 오실레이션이 비활성화된 경우 스파크아웃 스트로크가 프로그램되지 않은 경우 반전 지점에 접근하지 않습니다 .
- 4 스파크아웃 이후에 끝점에 접근합니다 .
- 8 이동할 거리를 삭제하여 오실레이션을 취소한 경우에는 스파크아웃 스트로크를 실행하고 필요하다면 끝점까지 접근해야 합니다 .
- 16 이동할 거리를 삭제하여 오실레이션을 취소한 경우에는 차단할 때와 같이 해당 반전 지점에 접근해야 합니다 .
- 32 새 피드는 다음 반전 지점 이후에만 활성화됩니다 .
- 64 FA 가 0 입니다 . FA = 0: 경로 오버레이가 활성화됩니다 .
  FA 가 0 이 아닙니다 . FA <> 0: 속도 오버레이가 활성화됩니다 .
- 128 로터리 축 DC (최단 경로) 의 경우
- 256 스파크아웃 스트로크는 이중 스트로크입니다 (표준). 1=단일 스트로크
- 512 1 차 접근 시작 위치

OSNSC 스파크아웃 스트로크 개수 정의

OSE 오실레이션 비활성화 후 접근할 종료 위치 정의 (공작물 좌표계에서)

참고 :
OSE 를 프로그래밍할 때 옵션 4 가 OSCTRL 에서 암시적으로 적용됩니다 .

OSB 오실레이션 활성화 전 접근할 시작 위치 정의 (공작물 좌표계에서)

시작 위치에는 반전 지점 1 이전에 접근합니다 . 시작 위치가 반전 지점 1 과 일치하는 경우에는 다음으로 반전 지점 2 에 접근하게 됩니다 . 시작 위치에 도달한 후에는 이 위치가 반전 지점 1 과 일치하더라도 정지 시간이 적용되지 않습니다 . 그 대신 축이 미세 정위치 정지 신호를 기다립니다 . 설정된 정위치 정지 조건이 모두 충족되어야 합니다 .

참고 :
시작 위치에 대한 접근을 초기화하도록 셋팅 데이터 SD43770 $SA_OSCILL_CTRL_MASK 의 비트 9 를 설정해야 합니다 .

예제

예제 1: 두 반전 지점 간의 오실레이션을 위한 오실레이션 축

오실레이션 축 Z 은 위치 10 에서 100 사이를 오실레이션 할 것입니다 . 반전 지점 1 은 미세 정위치 정지를 통해서, 반전 지점 2는 일반 정위치 정지를 통해서 접근합니다. 오실레이션 축에 대한 이송 속도는 250 이 되어야 합니다 . 3 개의 스파크아웃 스트로크를 가공 동작 끝에서 실행하고 오실레이션은 종료 위치 200 에 접근해야 합니다 . 절입 축의 이송 속도는 1 이어야 하며 X 방향의 절입 끝은 위치 15 에 도달해야 합니다 .

프로그램 코드	설명
WAITP(X,Y,Z)	; 시작 위치
G0 X100 Y100 Z100	; 포지셔닝 축으로 전환
WAITP(X,Z)	
OSP1[Z]=10 OSP2[Z]=100	; 반전 지점 1, 반전 지점 2
OSE[Z]=200	; 끝 위치
OST1[Z]=0 OST2[Z]=–1	; U1 에서의 정지 시간 : 미세 정위치 정지 ; U2 에서의 정지 시간 : 일반 정위치 정지
FA[Z]=250 FA[X]=1	; 오실레이션 축의 이송 속도 , 절입 축의 이송 속도
OSCTRL[Z]=(4,0)	; 설정 옵션
OSNSC[Z]=3	; 3 개의 스파크아웃 스트로크
OS[Z]=1	; 오실레이션 시작
WHEN $A_IN[3]==TRUE DO DELDTG(X)	; 이동할 거리 삭제
POS[X]=15	; 시작 위치 X 축
POS[X]=50	종료 위치 X 축
OS[Z]=0	; 오실레이션 중지
M30	

주

OSP1[Z]=... - OSNCS[Z]=... 명령 순서도 블록에서 프로그래밍할 수 있습니다 .

예제 2: 반전 지점을 온라인으로 수정하여 오실레이션

비동기 오실레이션에 필요한 셋팅 데이터를 가공 프로그램에서 설정할 수 있습니다 .

셋팅 데이터가 프로그램에 직접 기술된 경우, 전처리 과정에서 변경사항이 적용됩니다. 전처리 정지 (STOPRE) 에 의해 동기화 반응이 이루어질 수 있습니다 .

프로그램 코드	설명
$SA_OSCILL_REVERSE_POS1[Z]=-10	
$SA_OSCILL_REVERSE_POS2[Z]=10	
G0 X0 Z0	
WAITP(Z)	
ID=1 WHENEVER $AA_IM[Z] < $$AA_OSCILL_REVERSE_POS1[Z] DO $AA_OVR[X]=0 ID=2 WHENEVER $AA_IM[Z] < $$AA_OSCILL_REVERSE_POS2[Z] DO $AA_OVR[X]=0	; 오실레이션 축의 실제 값이 반전 지점을 초과하면 절입 축이 정지됩니다 .
OS[Z]=1 FA[X]=1000 POS[X]=40	; 오실레이션 활성화
OS[Z]=0	; 오실레이션 비활성화
M30	

추가 정보

오실레이션 축

다음은 오실레이션 축에 적용됩니다 .

- 모든 축을 오실레이션 축으로 사용할 수 있습니다 .
- 몇 개 오실레이션 축을 동시에 활성화할 수 있습니다 (최대 : 포지셔닝 축 개수).
- 직선 보간 G1은 오실레이션 축에 대해 항상 활성 상태이며, 프로그램에서 현재 사용하는 G 명령과는 관계가 없습니다 .

오실레이션 축은 다음과 같이 작동합니다 .

- 다이나믹 변환을 위한 입력 축의 역할을 합니다 .
- 갠트리와 커플 모션 축의 가이드 축 역할을 합니다 .
- 다음 조건에서 이송될 수 있습니다 .
  - 저크 제한 없음 (BRISK)

    또는
  - 저크 제한 있음 (SOFT)

    또는
  - 굴곡부를 포지셔닝 축으로 하는 가속 커브 사용

오실레이션 반점 지점

오실레이션 위치를 정의할 때 현재 옵셋을 적용해야 합니다 .

- 절대 지정

  OSP1[Z]=< 값 >

  반점 지점 위치 = 옵셋 + 프로그램된 값의 합계

- 상대 지정

  OSP1[Z]=IC(< 값 >)

  반전 지점 위치 = 반전 지점 1 + 프로그램된 값

예제 :

프로그램 코드

```
N10 OSP1[Z]=100 OSP2[Z]=110
...
...
N40 OSP1[Z]=IC(3)
```

WAITP

기하 축을 사용하여 오실레이션을 수행하는 경우에는 WAITP 를 사용하여 해당 축에 대해 오실레이션을 활성화해야 합니다 .

오실레이션이 완료되면 WAITP 는 오실레이션 축을 포지셔닝 축으로 다시 입력하는 데 사용되므로 정상적인 사용이 가능해집니다 .

모션 동기 동작 및 정지 시간을 통한 오실레이션

설정된 정지 시간이 만료되고 나서 (축을 기준으로 한 새로운 이동 거리로 지정된) 오실레이션이 진행되는 동안 내부 블록이 변경됩니다 . 블록이 변경될 때 비활성화 기능이 검사됩니다 . 비활성화 기능은 모션 순서에 대한 제어 설정 (OSCTRL) 에 따라 정의됩니다 . *이러한 다이나믹 응답은 피드 오버라이드의 영향을 받을 수 있습니다 .*

스파크아웃 스트로크가 시작되거나 끝점에 접근하기 전에 오실레이션 스트로크가 실행됩니다 . *비활성화 응답이 변경된 것처럼 보이더라도 , 이 예제는 그 경우에 해당하지 않습니다 .*

11.2 동기 동작에 의한 오실레이션 제어 (OSCILL)

기능

이 오실레이션 모드를 사용할 경우 , 반전 지점이나 정의된 반전 범위 내에서만 절입 동작이 실행됩니다 .

요구사항에 따라 오실레이션 모션이 다음과 같이 진행될 수 있습니다 .

- 계속 또는
- 절입 실행이 끝날 때까지 중단

구문

1. **오실레이션 관련 파라미터 정의**
2. **동기화 동작 정의**
3. **축 할당 및 절입 정의**

의미

OSP1[< 오실레이션 축 >]=	반전 지점 위치 1
OSP2[< 오실레이션 축 >]=	반전 지점 위치 2
OST1[< 오실레이션 축 >]=	반전 지점 1 에서의 정지 시간 (초)
OST2[< 오실레이션 축 >]=	반전 지점 2 에서의 정지 시간 (초)
FA[< 오실레이션 축 >]=	오실레이션 축 피드
OSCTRL[< 오실레이션 축 >]=	설정 또는 리셋 옵션
OSNSC[< 오실레이션 축 >]=	스파크아웃 스트로크 개수
OSE[< 오실레이션 축 >]=	종점
WAITP(< 오실레이션 축 >)	오실레이션 축 허용

축 할당 및 절입
OSCILL[< 오실레이션 축 >]=(< 절입 축 1>,< 절입 축 2>,< 절입 축 3>)
POSP[< 절입 축 >]=(< 종점 >,< 부분 길이 >,< 모드 >)

OSCILL:	오실레이션 축에 대해 절입 축 할당
POSP:	완전 절입 및 부분 절입 정의 (' 파일 및 프로그램 관리 ' 단원 참조)
끝 위치 :	모든 부분 절입 이후의 절입 축 관련 종점이 이송됩니다 .
부분 길이 :	반전 지점 / 반전 범위의 부분 절입 길이
모드 :	완전 절입을 부분 절입으로 분할 = 크기가 동일한 2 개의 잔여 단계 (디폴트); = 크기가 동일한 모든 부분 절입

동기화 동작

WHEN… … DO	... 일 때 ... 실행
WHENEVER … DO	... 할 때마다 ... 실행

예제

반전 지점 1 에서는 절입이 절대 이뤄지지 않아야 합니다 . 반전 지점 2 의 경우 , 반전 지점 2 이전의 ii2 거리에서 절입이 시작돼야 하고 , 오실레이션 축은 부분 절입 끝의 반전 지점에서 대기하지 않아야 합니다 . Z 축이 오실레이션 축이고 X 축이 절입 축입니다 .

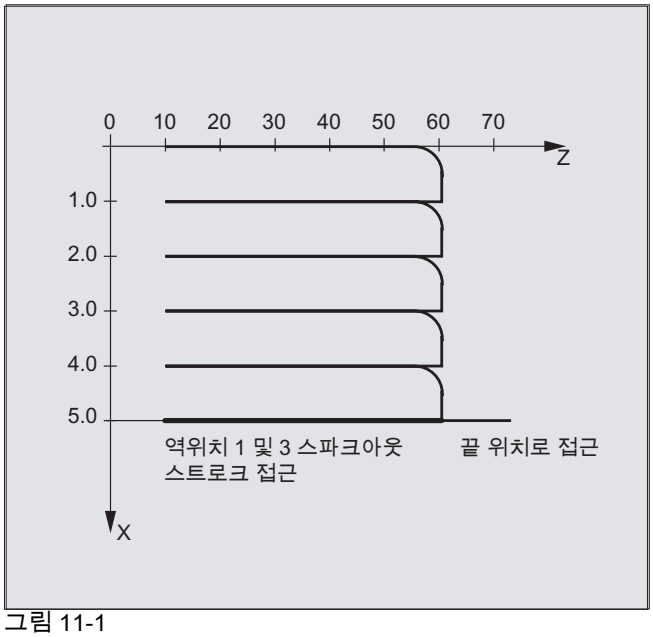


그림 11-1

1. 오실레이션 관련 파라미터

프로그램 코드		설명
DEF INT ii2	;	반전 범위 2 에 대한 변수 정의
OSP1[Z]=10 OSP2[Z]=60	;	반전 지점 1 및 2 정의
OST1[Z]=0 OST2[Z]=0	;	반전 지점 1: 미세 정위치 정지 반전 지점 2: 미세 정위치 정지
FA[Z]=150 FA[X]=0.5	;	Z 축 피드 속도 및 절입 축 X 피드 속도 오실레이션
OSCTRL[Z]=(2+8+16,1)	;	DTG 스파크아웃 접근 종점 삭제 후 및 DTG 접근 반전 지점 삭제 후 반전 지점 2 에서 오실레이팅 모션 비활성화
OSNC[Z]=3	;	스파크아웃 스트로크
OSE[Z]=70	;	종점 = 70
ii2=2	;	반전 지점 범위 설정
WAITP(Z)	;	Z 축에 대해 오실레이션 허용

2. 동기화 동작

프로그램 코드		설명
WHENEVER $AA_IM[Z]<$SA_OSCILL_REVERSE_POS2[Z] DO -> $AA_OVR[X]=0 $AC_MARKER[0]=0	;	MCS에서 오실레이션 축 Z의 실제 위치가 반전 범위 2 의 시작 지점보다 작은 경우 , 절입 축 X 의 축 오버라이드가 항상 0% 로 설정되고 색인 0 에 속한 비트 메모리가 0 의 값으로 설정됩니다 .
WHENEVER $AA_IM[Z]>=$SA_OSCILL_REVERSE_POS2[Z] DO $AA_OVR[Z]=0	;	MCS에서 오실레이션 축 Z의 실제 위치가 반전 범위 2 보다 큰 경우 , 오실레이션 축 Z 의 축 오버라이드가 항상 0% 로 설정됩니다
WHENEVER $AA_DTEPW[X] == 0 DO $AC_MARKER[0]=1	;	부분 절입의 남은 이동 거리가 0이면 항상 비트 메모리의 인덱스 0 을 1 로 설정합니다 .
WHENEVER $AC_MARKER[0]==1 DO $AA_OVR[X]=0 $AA_OVR[Z]=100	;	비트 메모리의 인덱스 0 이 1 인 경우 절입 축 X 의 축 오버라이드를 0% 로 설정하십시오 . 이렇게 하면 정상보다 이른 절입 , 즉 오실레이션 축 Z 가 여전히 좌측 반전 영역 2 에 있는 상태지만 절입 축 X 가 새로운 절입을 준비하는 상태가 되는 것이 방지됩니다. Z축을 이동시키기 위해 오실레이션 축 Z 의 축 오버라이드를 0% (두 번째 동기 동작 실행) 에서 다시 100% 로 설정하십시오 .

-> 반드시 싱글 블록으로 프로그래밍돼야 합니다 .

3. 오실레이션 시작

프로그램 코드		설명
OSCILL[Z]=(X) POSP[X]=(5,1,1)	;	축 시작 오실레이션 축 Z 가 X 축에 절입 축으로 할당됩니다 . X 축이 종점 5까지 1 단계로 이동해야 합니다.
M30	;	프로그램 끝

설명

1. **오실레이션 파라미터 정의**
   오실레이션 관련 파라미터는 절입 축 및 오실레이션 축이 포함된 이동 블록과 절입 정의에 앞서 정의되어야 합니다 (' 비동기식 오실레이션 ' 참조).
2. **동기화 동작 정의**
   다음과 같은 동기화 조건을 정의할 수 있습니다 .
   절입 억제
   .
   반전 지점에서 절입이 이뤄지는 동안 오실레이션 동작 정지 .
   부분 절입 완료 시 오실레이션 동작 재시작 .
   다음 부분 절입 시작 정의 .
3. **오실레이션 축 및 절입 축** 외에도 **부분 절입 및 완전 절입** 정의 .

오실레이션 파라미터 정의

오실레이션 축 및 절입 축 할당 : OSCILL

OSCILL[오실레이션 축] = (절입 축 1, 절입 축 2, 절입 축 3)

OSCILL 명령으로 축 할당 및 오실레이션 동작시작이 정의됩니다 .

오실레이션 축에 최대 3 개의 절입 축을 할당할 수 있습니다 .

주

오실레이션을 시작하기에 앞서 축의 특성에 대한 동기화 조건을 정의해야 합니다 .

절입 정의 : POSP

POSP[절입 축] = (끝점 , 부분 길이 , 모드)

POSP 명령으로 다음과 같은 작업을 정의해서 제어할 수 있습니다 .

- (끝점을 기준으로 한) 완전 절입
- 반전 지점이나 반전 범위의 부분 절입 길이
- (끝점을 기준으로 해서) 끝점에 도달했을 때 부분 절입 응답

Mode = 0	최종 2 개의 부분 절입에 대한 대상 지점까지의 이동할 거리는 2 개의 동등한 단계로 분할됩니다 (기본 설정).
Mode = 1	크기가 동일한 모든 부분 절입 완전 절입을 기점으로 해서 계산됩니다 .

동기화 동작 정의

아래 열거된 동기화 동작이 일반적인 오실레이션에 사용됩니다 .

아래 주어진 개별 작업에 대한 해결 사례를 사용자 전용 오실레이션 동작용 모듈로 활용할 수 있습니다 .

주

동기화 조건은 개인적인 경우에 따라 다르게 프로그래밍할 수 있습니다 .

키워드

WHEN … DO …	... 인 경우 , ...
WHENEVER … DO	항상 ... 인 경우 , ...

기능

아래 자세히 설명된 정보를 시용하여 다음과 같은 기능을 구현할 수 있습니다 .

:

1. 반전 지점에서 절입
2. 반전 범위에서 절입
3. 두 반전 지점에서 절입
4. 반전 지점에서 오실레이션 동작 중지
5. 오실레이션 동작 재시작
6. 너무 이른 부분 절입 시작 금지

여기에 제시된 모든 동기화 동작 사례는 다음 조건을 전제로 합니다 .

- 반전 지점 1 < 반전 지점 2
- Z = 오실레이션 축
- X = 절입 축

주

자세한 내용은 ' 동기화 동작 ' 단락을 참조하십시오 .

오실레이션 축 , 절입 축 , 부분 절입 및 완전 절입 정의

반전 지점 범위에서 절입

반드시 반전 지점에 도달하기 전에 반전 범위 내에서 절입 동작이 시작돼야 합니다 .

이 동기화 동작으로 인해 오실레이션 축이 반전 범위 내에 도달할 때까지 절입 동작이 금지됩니다 .

위의 가정의 준하여 다음과 같은 명령이 사용됩니다 .

반전 지점 범위 1: WHENEVER $AA_IM[Z]>$SA_OSCILL_RESERVE_POS1[Z]+ii1 DO $AA_OVR[X] = 0	MCS 에서 오실레이션 축의 실제 위치가 반전 범위 1 보다 큰 경우 , 절입 축의 축 오버라이드가 0% 로 설정됩니다 .
반전 지점 범위 2: WHENEVER $AA_IM[Z]<$SA_OSCILL_RESERVE_POS2[Z]+ii2 DO $AA_OVR[X] = 0	MCS 에서 오실레이션 축의 실제 위치가 반전 범위 2 시작 위치보다 작은 경우 , 절입 축의 축 오버라이드가 0% 로 설정됩니다 .

반전 지점에서 절입

오실레이션 축이 반전 지점에 도달하기 전에는 절입 축이 이동하지 않습니다 .

(위에 언급된) 전제 하에 다음과 같은 명령을 사용할 수 있습니다 .

반전 범위 1: WHENEVER $AA_IM[Z]<>$SA_OSCILL_RESERVE_POS1[Z] DO $AA_OVR[X] = 0 → → $AA_OVR[Z] = 100	MCS 에서 오실레이션 축 Z 의 실제 위치가 위치 반전 지점 1보다 크거나 작을 때마다 절입 축 X의 축 오버라이드가 0%로 설정되고 오실레이션 축 Z 의 축 오버라이드가 100% 로 설정됩니다 .
반전 범위 2: 반점 지점 2 의 경우 : WHENEVER $AA_IM[Z]<>$SA_OSCILL_RESERVE_POS2[Z] DO $AA_OVR[X] = 0 → → $AA_OVR[Z] = 100	MCS 에서 오실레이션 축 Z 의 실제 위치가 위치 반전 지점 2보다 크거나 작을 때마다 절입 축 X의 축 오버라이드가 0%로 설정되고 오실레이션 축 Z 의 축 오버라이드가 100% 로 설정됩니다 .

반전 지점에서 오실레이션 동작 중지

반전 지점에서 오실레이션 축이 정지하고 그와 동시에 절입 동작이 시작됩니다 . 절입 동작이 완료되면 오실레이션 동작이 계속됩니다 .

그와 동시에 , 아직 활성 상태인 이전 동기화 동작에 의해 오실레이션 축이 정지한 경우 이 동기화 동작을 이용하여 절입 동작을 시작할 수 있습니다 .

(위에 언급된) 전제 하에 다음과 같은 명령을 사용할 수 있습니다 .

반전 범위 1: WHENEVER $SA_IM[Z]==$SA_OSCILL_RESERVE_POS1[Z] DO $AA_OVR[X] = 0 → → $AA_OVR[Z] = 100	MCS 에서 오실레이션 축의 실제 위치가 위치 반전 지점과 동일할 때마다 오실레이션 축의 축 오버라이드가 0% 로 설정되고 절입 축의 축 오버라이드가 100% 로 설정됩니다 .
반전 범위 2: WHENEVER $SA_IM[Z]==$SA_OSCILL_RESERVE_POS2[Z] DO $AA_OVR[X] = 0 → → $AA_OVR[Z] = 100	MCS 에서 오실레이션 축 Z 의 실제 위치가 위치 반전 지점 2 와 동일할 때마다 오실레이션 축 X 의 축 오버라이드가 0% 로 설정되고 절입 축의 축 오버라이드가 100% 로 설정됩니다 .

반전 지점 온라인 검사

비교 우측에 $$ 로 코드화된 주 실행 변수가 있는 경우 , IPO 싸이클에서 두 변수를 검사하여 서로 지속적으로 비교합니다 .

주

자세한 내용은 ' 동기화 동작 ' 단락을 참조하십시오 .

오실레이션 동작 재시작

이 동기화 동작의 목적은 부분 절입 동작이 완료된 순간에 오실레이션 축이 계속 이동하는 것입니다 .

(위에 언급된) 전제 하에 다음과 같은 명령을 사용할 수 있습니다 .

WHENEVER $AA_DTEPW[X]==0 DO $AA_OVR[Z]= 100	WCS 에서 절입 축 X 의 부분 절입과 관련한 남은 거리가 0 인 경우 , 오실레이션 축의 축 오버라이드가 항상 100% 로 설정됩니다 .

다음 부분 절입

절입이 완료됐을 때 다음 부분 절입의 너무 이른 시작이 반드시 제한돼야 합니다 .

채널 관련 마커 ($AC_MARKER[Index]) 가 이 용도로 사용됩니다 . 채널 관련 마커는 부분 절입 (부분 절입의 이동할 거리 ≡ 0) 끝에 실행되고 축이 반점 범위를 벗어났을 때 삭제됩니다 . 그리고 나서 동기화 동작에 의해 다음 절입 동작이 제한됩니다 .

주어진 전제를 토대로 반전 지점 1 에 대해 다음과 같은 명령이 적용됩니다 .

1. 마커 설정 : WHENEVER $AA_DTEPW[X] == 0 DO $AC_MARKER[1]=1	WCS 에서 절입 축 X 의 부분 절입과 관련한 남은 거리가 0 일 때마다 색인 0 에 속한 비트 메모리가 항상 1 의 값으로 설정됩니다 .
2. 마커 삭제 WHENEVER $AA_IM[Z]<> $SA_OSCILL_RESERVE_POS1[Z] DO $AC_MARKER[1] = 0	MCS 에서 오실레이션 축 Z 의 실제 위치가 위치 반전 지점 1 의 위치보다 크거나 작을 때마다 비트 메모리 1 이 0 으로 설정됩니다 .
3. 절입 금지 WHENEVER $AC_MARKER[1]==1 DO $AA_OVR[X]=0	비트 메모리 1 이 동일할 때마다 절입 축 X 의 축 오버라이드가 0% 로 설정됩니다 .

펀칭 및 니블링 12

12.1 활성 , 비활성

12.1.1 펀칭 및 니블링 활성화 / 비활성화 (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC)

기능

펀칭 및 니블링 활성화 / 비활성화

PON 및 SON은 펀칭 및 니블링 기능을 활성화하는 데 사용됩니다. SPOF는 모든 펀칭 및 니블링 관련 기능을 종료합니다 . 모달 명령인 PON 및 SON 은 상호 배타적입니다 . 예를 들어 , PON 가 활성화되면 SON 이 비활성화됩니다 .

리더를 사용한 펀칭 및 니블링

SONS 및 PONS 기능은 펀칭 또는 니블링 기능을 활성화하는 데도 사용됩니다 .

보간 레벨에서 스트로크를 제어하는 SON/PON 과는 반대로 , 이 기능을 사용하면 스트로크 초기화에 대한 신호 관련 제어가 서보 레벨에서 이루어집니다 . 이것은 스트로크 주파수가 더 커지게 하므로 펀칭 기능이 향상됩니다 .

리더에서 신호가 평가되는 동안에는 니블링 또는 펀칭 축에서 위치를 변경할 수 있도록 하는 모든 기능 (예 : 핸드휠 이송 , PLC 를 통한 프레임 변경 , 측정 기능) 이 금지됩니다 .

지연 상태의 펀칭

PDELAYON 은 펀칭 스트로크의 출력 지연을 유도합니다 . 모달로 적용되는 명령에는 준비 기능이 있으며, 따라서 일반적으로 PON 이전에 위치합니다. 정상적인 펀칭은 PDELAYOF 이후 일반 펀칭이 재개됩니다 .

주

지연 시간은 셋팅 데이터 SD42400 $SC_PUNCH_DWELLTIME 에서 설정됩니다 .

이송 종속 가속

PUNCHACC는 구멍 간격에 따라 서로 다른 가속도를 정의하는 가속 특성을 지정하는 데 사용할 수 있습니다 .

2 차 펀칭 인터페이스

2 차 펀칭 인터페이스 (보조 펀칭 유닛 또는 비교 가능한 미디어) 를 사용해야 하는 기계를 제어기의 2 차 고속 디지털 입출력 페어 (I/O 페어) 로 전환할 수도 있습니다. 전체 펀칭/니블링 기능을 양쪽 인터페이스에서 사용할 수 있습니다 . SPIF1 및 SPIF2 명령은 1 차 펀칭 인터페이스와 2 차 펀칭 인터페이스를 전환하는 데 사용됩니다 .

주

조건: 머신 데이터에서 펀칭 기능에 대한 2차 I/O 페어를 정의해야 합니다 (장비 제조업체 사양 참조).

구문

```
PON G... X... Y... Z...
SON G... X... Y... Z...
SONS G... X... Y... Z...
PONS G... X... Y... Z...
PDELAYON
PDELAYOF
PUNCHACC(<Smin>,<Amin>,<Smax>,<Amax>)
SPIF1/SPIF2
SPOF
```

의미

PON	펀칭 ON.
SON	니블링 ON.
PONS	리더를 통한 펀칭 ON.
SONS	리더를 통한 니블링 ON.
SPOF	펀칭 / 니블링을 해제
PDELAYON	지연을 포함한 펀칭 ON.
PDELAYOF	지연을 포함한 펀칭 OFF
PUNCHACC	이송에 따른 가속을 활성화합니다 . 파라미터 : <Smin> 최소 홀 간격 <Amin> 초기 가속 <Amin> 이 <Amax> 보다 클 수도 있습니다 . <Smax> 최대 홀 간격 <Amax> 최종 가속 <Amax> 가 <Amin> 보다 클 수도 있습니다 .
SPIF1	**1 차** 펀칭 인터페이스를 활성화합니다 . 스트로크는 1 차 고속 I/O 페어를 사용하여 제어됩니다 .
SPIF2	**2 차** 펀칭 인터페이스를 활성화합니다 . 스트로크는 2 차 고속 I/O 페어를 사용하여 제어됩니다 . **참고 :** 1 차 펀칭 인터페이스는 항상 리셋 또는 제어 시스템의 전원이 켜진 후 활성화됩니다 . 하나의 펀칭 인터페이스만 사용될 경우에는 프로그래밍할 필요가 없습니다 .

예제

예제 1: 니블링 활성화

프로그램 코드	설명
...	
N70 X50 SPOF	; 펀칭 시작 없는 위치
N80 X100 SON	; 니블링 활성화 , 모션 이전 (X=50) 및 프로그래밍된 이동 완료 (X=100) 시 스트로크 시작
...	

예제 2: 지연 펀칭

프로그램 코드	설명
...	
N170 PDELAYON X100 SPOF	; 펀칭 시작 없는 위치 , 지연 펀칭 시작 활성화
N180 X800 PON	; 펀칭을 활성화합니다 . 펀칭 스트로크는 종료 위치에 도달할 때 지연과 함께 출력됩니다 .
N190 PDELAYOF X700	; 지연 펀칭 해제 , 프로그래밍된 이동 완료 시 정상 펀칭 시작
...	

예제 3: 2 개의 펀칭 인터페이스를 사용한 펀칭

프로그램 코드	설명
...	
N170 SPIF1 X100 PON	; 블록 종료 시 스트로크는 첫 번째 고속 출력에서 시작됩니다. '스트로크 적용 ' 신호가 첫 번째 입력에서 모니터링됩니다 .
N180 X800 SPIF2	; 두 번째 스트로크는 두 번째 고속 출력에서 시작됩니다. '스트로크 적용' 신호가 두 번째 입력에서 모니터링됩니다 .
N190 SPIF1 X700	; 첫 번째 인터페이스를 통해 모든 추가 스트로크가 제어됩니다 .
...	

추가 정보

리더가 있는 펀칭 및 니블링 (PONS/SONS)

리더가 있는 펀칭 및 니블링은 둘 이상의 채널에서 동시에 활성화될 수 없습니다 . PONS 또는 SONS 는 한 번에 하나의 채널에서만 활성화될 수 있습니다 .

이송 종속 가속 (PUNCHACC)

예제 :
PUNCHACC(2,50,10,100)

2mm 이하의 심공 간격 :

축은 최대 가속의 50% 에 해당하는 속도에서 가속화됩니다 .

2 mm ~ 10 mm 의 심공 간격 :

가속이 간격에 비례하여 100% 까지 증가됩니다 .

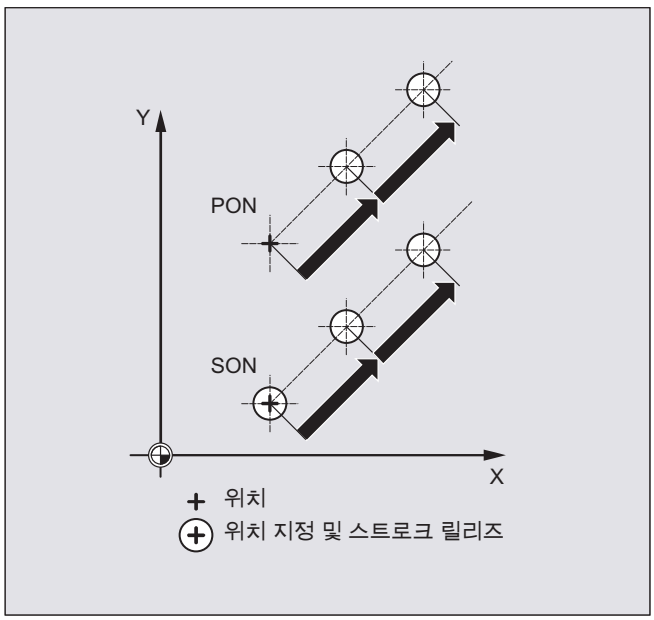
10 mm 이상의 심공 간격 :

이송 가속도가 최대 가속도의 100% 입니다 .

최초 스트로크 시작

기능 활성화 이후에 최초 스트로크가 시작되는 순간은 니블링 또는 펀칭 선택 여부에 따라 달라집니다 .

- PON/PONS:
  - 심지어 활성화 이후 첫 번째 블록의 스트로크를 포함하여 모든 블록이 블록 끝에서 실행됩니다 .
- SON/SONS:
  - 니블링 기능 활성화 이후 첫 번째 스트로크가 블록 시작 시점에 실행됩니다 .
  - 이후의 모든 스트로크는 블록 끝에서 시작됩니다 .



즉석 펀칭 니블링

블록에 펀칭 축이나 니블링 축 (활성 평편의 축) 에 대한 이송 정보가 포함된 경우에 한하여 스트로크가 시작됩니다 .

그러나 동일한 위치에서 스트로크를 시작하려면 펀칭 / 니블링 축 중 하나를 이송 경로 0 으로 프로그래밍하면 됩니다 .

회전 가능한 공구를 이용한 가공

주

접선에 프로그래밍된 경로까지 회전 가능한 공구의 위치를 지정하려면 접선 제어 기능을 사용하십시오 .

M 명령 사용

이전 버전과 마찬가지로 매크로를 통해 언어 명령 대신 특수 M 코드를 사용할 수 있습니다 (호환성). 이전 버전에서 사용된 M 코드 및 이에 해당하는 언어 명령은 다음과 같습니다 .

```
M20, M23   ≙   SPOF
M22        ≙   SON
M25        ≙   PON
M26        ≙   PDELAYON
```

매크로 파일 예제 :

프로그램 코드	설명
DEFINE M25 AS PON	; 펀칭 ON
DEFINE M125 AS PONS	; 리더가 있는 펀칭 활성화
DEFINE M22 AS SON	; 니블링 ON
DEFINE M122 AS SONS	; 리더가 있는 니블링 활성화
DEFINE M26 AS PDELAYON	; 지연 펀칭 ON
DEFINE M20 AS SPOF	; 펀칭 / 니블링 OFF
DEFINE M23 AS SPOF	; 펀칭 , 니블링 OFF

프로그래밍 예제 :

프로그램 코드	설명
...	
N100 X100 M20	; 펀칭 시작 없는 위치
N110 X120 M22	; 니블링 활성화 , 모션 전후에 스트로크 시작
N120 X150 Y150 M25	; 펀칭 활성화 , 모션 종료 시 스트로크 시작
...	

12.2 자동 경로 분할

기능

경로 구간으로 분할

펀칭 또는 니블링이 활성화된 상태에서 SPP 및 SPN 함수를 사용하여 경로 축에 대해 프로그래밍된 전체 이송 범위를 동일한 길이의 여러 경로 구간으로 분할할 수 있습니다 . 내부적으로 각 경로 구간은 하나의 블록에 해당합니다 .

스트로크 횟수

펀칭이 활성화된 경우 , 첫 번째 경로 구간의 끝점에서 첫 번째 스트로크가 실행됩니다 . 반면에 니블링이 활성화된 경우 , 첫 번째 경로 구간의 시작 점에서 스트로크가 이뤄집니다 . 따라서 전체 이송 구간에서 다음과 같은 횟수가 산정됩니다 .

펀칭 : 스크로크 횟수 = 경로 구간 개수

니블링 : 스크로크 횟수 = 경로 구간 개수 + 1

보조 기능

생성된 블록 중 첫 번째 블록에서 보조 기능이 실행됩니다 .

구문

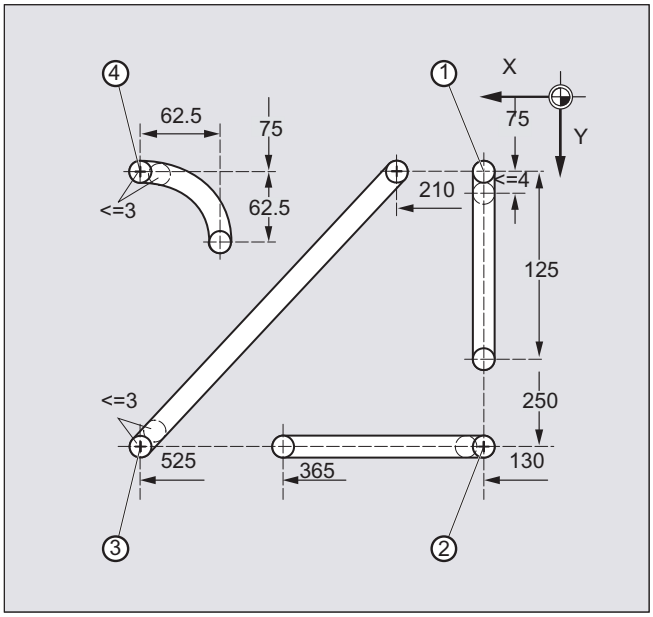
```
SPP=
SPN=
```

의미

SPP	경로 구간의 크기 (스크로크간 최대 거리), 모달
SPN	블록당 경로 구간 개수 , 모달식 활성화

예제 1

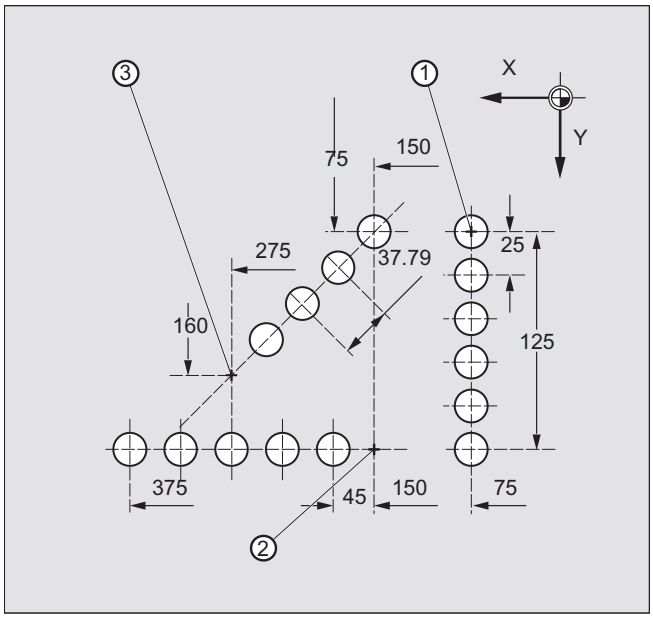
프로그래밍된 니블링 구간이 자동으로 경로 구간으로 분할돼야 합니다 .



프로그램 코드	비고
N100 G90 X130 Y75 F60 SPOF	; 시작 지점 1 에 배치
N110 G91 Y125 SPP=4 SON	; 니블링 활성화 , 자동 경로 분할과 관련한 최대 경로 구간 길이 : 4 mm (3.81 lb)
N120 G90 Y250 SPOF	; 니블링 비활성화 , 시작 지점 2 에 배치
N130 X365 SON	; 니블링 활성화 , 자동 경로 분할과 관련한 최대 경로 구간 길이 : 4 mm (3.81 lb)
N140 X525 SPOF	; 니블링 비활성화 , 시작 지점 3 에 배치
N150 X210 Y75 SPP=3 SON	; 니블링 활성화 , 자동 경로 분할과 관련한 최대 경로 구간 길이 : 3 mm (3.81 lb)
N140 X525 SPOF	; 니블링 비활성화 , 시작 지점 4 에 배치
N170 G02 X-62.5 Y62.5 I J62.5 SPP=3 SON	; 니블링 활성화 , 자동 경로 분할과 관련한 최대 경로 구간 길이 : 3 mm (3.81 lb)
N180 G00 G90 Y300 SPOF	; 니블링 비활성화

예제 2

개별적인 일련의 심공을 대상으로 자동 경로 분할이 이뤄져야 합니다 . 분할을 위해 최대 경로 구간 길이 (SPP 값) 가 지정됩니다 .



프로그램 코드	비고	
N100 G90 X75 Y75 F60 PON	;	시작 지점 1 에 배치 개별 심공 펀칭
N110 G91 Y125 SPP=25	;	자동 경로 분할과 관련한 최대 경로 구간 길이 : 25 mm (3.81 lb)
N120 G90 X150 SPOF	;	펀칭 비활성화 , 시작 지점 2 에 배치
N130 X375 SPP=45 PON	;	펀칭 활성화 , 자동 경로 분할과 관련한 최대 경로 구간 길이 : 45 mm (3.81 lb)
N140 X275 Y160 SPOF	;	펀칭 비활성화 , 시작 지점 3 에 배치
N150 X150 Y75 SPP=40 PON	;	펀칭 활성화 , 프로그래밍된 경로 구간 길이 (40 mm) 대신 , 산정된 경로 구간 길이 (37.79 mm) 가 사용됩니다 .
N160 G00 Y300 SPOF	;	펀칭 비활성화 , 위치 지정

12.2.1 경로 축을 대상으로 경로 분할

SPP 경로 구간 길이

SPP 는 전체 이송 거리가 분할된 스트로크간 최대 거리와 최대 경로 구간 길이를 지정하는 데 사용됩니다 . 이 명령은 SPOF 또는 SPP=0 으로 활성화할 수 있습니다 .

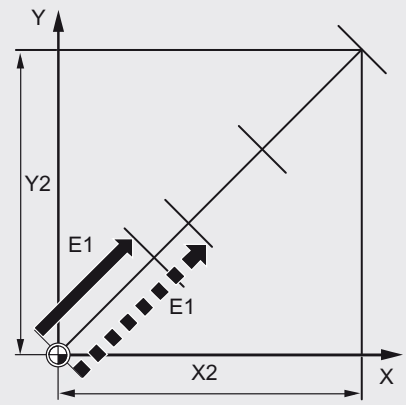
예 :

```
N10 SON X0 Y0
N20 SPP=2 X10
```

10 mm 의 전체 이송 길이가 2 mm 의 경로 구간 5 개로 분할됩니다 (SPP=2).

주

SPP 에 의해 활성화된 경로 구간은 항상 등거리 (모든 구간의 길이가 동일) 가 됩니다 . 다시 말해서 , 프로그래밍된 경로 구간 크기 (SPP 설정) 는 전체 이송 거리의 등거리와 SPP 값이 정수인 경우에만 유효합니다. 정수가 아닌 경우, 정수 비율을 산출하기 위해 경로 구간 크기가 내부적으로 감소됩니다 .



X2/Y2 프로그램된 위치 이송 경로(니블링 또는 펀칭 블록)
E1 프로그램된 경로 분할 길이
E1' 자동적으로 라운딩 된 경로 분할 길이

예 :

```
N10 G1 G91 SON X10 Y10
N20 SPP=3.5 X15 Y15
```

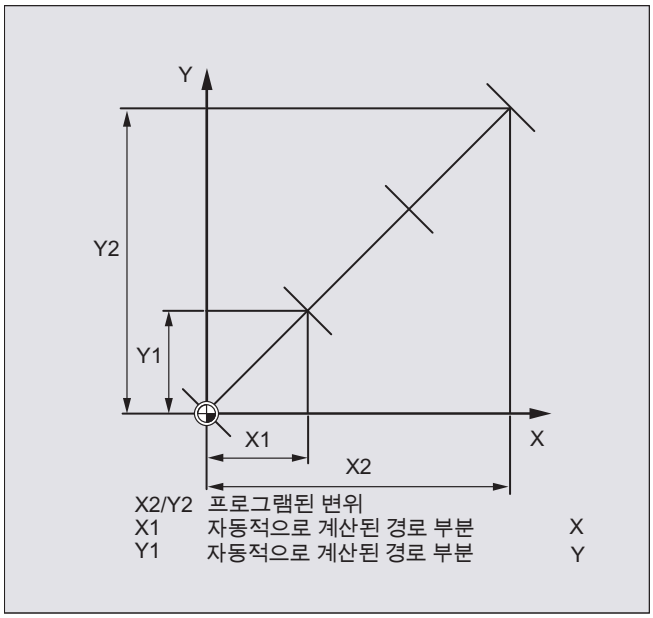
전체 이송 거리가 15 mm 이고 경로 구간 길이가 3.5 mm 인 경우 , 분할 비율은 정수값이 아닙니다 (4.28) 이 경우 , 산출 가능한 다음 정수 비율로 SPP 값이 감소됩니다 . 이 예제에서 경로 구간 길이는 3 mm 가 됩니다 .

SPN 경로 구간 개수

SPN 를 사용하여 전체 이송 거리에서 산출될 경로 구간 개수를 정의할 수 있습니다 . 구간 길이는 자동으로 산정됩니다 . SPN 는 비모달식이기 때문에 펀칭이나 니블링이 PON 또는 SON 으로 미리 활성화돼야 합니다 .

동일한 블록의 SPP 및 SPN

경로 구간 길이(SPP)와 경로 구간 개수(SPN)를 동일한 블록에 프로그래밍할 경우, 그 블록에는 SPN 이 적용되고 나머지 모든 후속 블록에는 SPP 가 적용됩니다 . SPN 보다 SPP 가 먼저 활성화된 경우 , SPN 해당 블록 뒤에 다시 SPP 가 적용됩니다 .



주

펀칭 / 니블 기능을 사용할 수 있는 경우 , 이 공정에 구애 받지 않고 SPN 또는 SPP 함수를 이용해서 자동 경로 분할 기능을 프로그래밍할 수 있습니다 .

12.2.2 단일 축을 대상으로 경로 분할

경로 축 왜에 단일 축이 펀칭 / 니블링 축으로 정의된 경우 , 단일 축을 대상으로 자동 경로 분할 기능을 활성화할 수 있습니다 .

SPP 에 대한 단일 축의 응답

프로그래밍된 경로 구간 길이(SPP)는 기본적으로 경로 축을 의미합니다. 이런 이유로 단일 축 동작 및 SPP 값은 존재하지만 프로그래밍된 경로 축은 포함되지 않는 블록에서는 SPP 값이 무시됩니다 .

단일 축과 경로 축이 함께 특정 블록에 프로그래밍된 경우 , 적합한 기계 데이터의 설정에 따라 단일 축이 응답합니다 .

1. 표준 설정

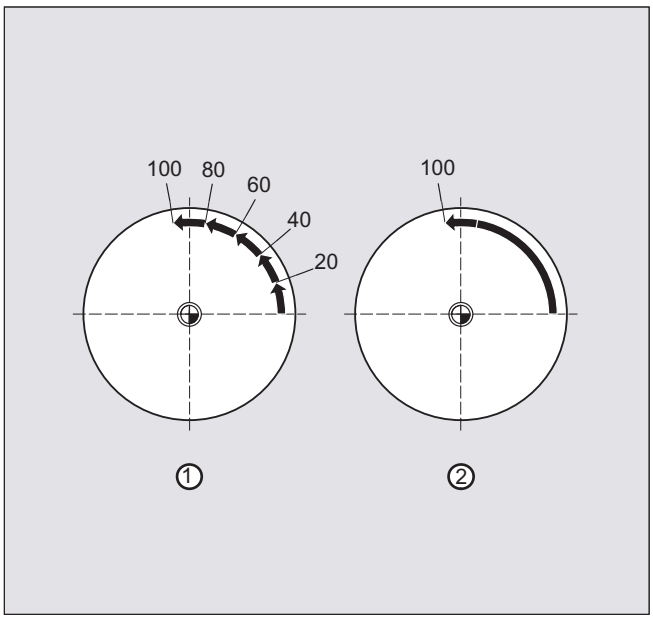
   단일 축에 의해 이송되는 경로는 SPP 에 의해 생성된 중간 블록 사이에 균등하게 분할됩니다 .

예 :

```
N10 G1 SON X10 A0
N20 SPP=3 X25 A100
```

스크로크간의 거리가 3 mm 로 프로그래밍됐기 때문에 15 mm 의 X 축 (경로 축) 의 전체 이송 거리를 대상으로 5 개의 블록이 생성됩니다 .

따라서 A 축은 모든 블록에서 20° 로 회전합니다 .



1. 경로 분할이 이뤄지지 않은 단일 축

   생성된 블록 중 첫 번째 블록의 전체 거리를 단일 축이 이송합니다 .

2. 경로 분할 유 / 무

   단일 축의 응답은 다음과 같은 경로 축의 보간에 따라 다릅니다 .

- 원형 보간 : 경로 분할
- 선형 보간 : 경로 분할 없음

SPN 에 응답

경로 축이 동일한 블록에 프로그래밍되지 않은 경우라도 프로그래밍된 경로 구간 개수가 적용될 수 있습니다 .

요구 사항 : 단일 축이 펀칭 / 니블링 축으로 정의됩니다 .

연삭 13

13.1 가공 프로그램에서 연삭 관련 도구 모니터링 (TMON, TMOF)

기능

TMON 명령을 사용하면 가공 프로그램에서 절삭 공구 (유형 400 - 499) 에 대한 기하 및 속도 모니터링을 활성화할 수 있습니다 . TMOF 명령을 사용하여 가공 프로그램에서 비활성화할 때까지 모니터링 활성 상태가 유지됩니다 .

주

장비 제조업체의 설명서를 참조하십시오 .

요구 사항

절삭 관련 공구 파라미터 $TC_TPG1 to $TC_TPG9 를 반드시 설정해야 합니다 .

구문

```
TMON(<T-No.>)
TMOF(<T-No.>)
```

설명

TMON	절삭 관련 공구 모니터링 **활성화**
TMOF	절삭 관련 공구 모니터링 **비활성화**
<T-No.>	T 번호 지정 **참고 :** T 번호가 있는 공구가 비활성화된 경우에만 필요
TMOF(0)	모든 공구 모니터링 비활성화

그 밖의 정보

절삭 관련 공구 파라미터

파라미터	의미	데이터 형식
$TC_TPG1	스핀들 번호	INT
$TC_TPG2	연결 규칙 파라미터는 왼쪽 및 오른쪽 절삭 휠 측면에 대해 자동으로 동일하게 유지됩니다 .	INT
$TC_TPG3	최소 휠 반경	REAL
$TC_TPG4	최소 휠 너비	REAL
$TC_TPG5	현재 휠 너비	REAL
$TC_TPG6	최대 속도	REAL
$TC_TPG7	최대 주변 속도	REAL
$TC_TPG8	경사 휠 각도	REAL
$TC_TPG9	반경 계산을 위한 파라미터 번호	INT

참조 :
기능 설명서의 기본 기능 , 공구 옵셋 (W1)

공구를 선택하여 공구 모니터링 활성화

기계 데이터 설정에 따라 공구 선택이 활성화됐을 때 절삭 공구 (유형 400 - 499) 에 대한 모니터링을 자동으로 활성화할 수 있습니다 .

각 스핀들에 대해 한 번에 오직 **하나**의 모니터링 작업만 활성화할 수 있습니다 .

기하 모니터링

현재 휠 반경과 현재 휠 너비가 모니터링됩니다 .

스핀들 오버라이드를 감안하여 속도 제한과 대조하면서 설정 속도를 주기적으로 모니터링합니다 .

속도 제한은 최대 휠 원주 속도와 현재 휠 반경에서 산정된 속도를 최대 속도와 비교해서 산출된 더 작은 값입니다 .

T 또는 D 번호 없이 작업

표준 **T** 번호와 표준 **D** 번호를 기계 데이터마다 설정할 수 있습니다 .

기계 데이터는 다시 프로그래밍할 필요가 없으며 전원을 켜거나 리셋 후에 자동 적용됩니다 .

예 : 모든 가공이 동일한 절삭 휠로 수행됩니다 .

기계 데이터를 이용해서 리셋을 위해 활성 상태의 공구가 유지되도록 설정할 수 있습니다 ("D 번호 , 절삭날 번호의 자유 할당 (CE 어드레스) [쪽 433]" 참조).

추가 기능 14

14.1 축 기능 (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL)

기능

AXNAME 는 축 이름을 인식하지 못한 경우 보편적으로 적용되는 싸이클을 생성하는 데 사용됩니다 .

AX 는 기화 축 및 동기화 축을 간접 프로그래밍하는 데 사용됩니다 . 축 식별자는 AXIS 형식의 변수에 저장되거나 AXNAME 또는 SPI 와 같은 명령에서 제공됩니다 .

SPI 는 스핀들을 대상으로 축 기능이 프로그래밍된 경우 (예 : 동기화 스핀들) 에 사용됩니다 .

AXTOSPI 는 축 식별자를 스핀들 색인으로 변환 (SPI 으로 함수 역변환) 하는 데 사용됩니다 .

AXSTRING는 축 식별자(데이터 형식 AXIS)를 문자열로 변환(AXNAME으로 함수 역변환)하는 데 사용됩니다 .

ISAXIS 는 특정 기하 축이 반드시 존재해서 오류 메시지로 인한 후속 $P_AXNX 호출 취소 방지를 목적으로 일반 싸이클에 사용됩니다 .

MODAXVAL 은 모듈로 로타리 축의 모듈로 위치를 파악할 목적으로 사용됩니다 .

구문

AXNAME(" 문자열 ")
AX[AXNAME(" 문자열 ")]
SPI(n)

AXTOSPI(A) 또는 AXTOSPI(B) 또는 AXTOSPI(C)
AXSTRING(SPI(n))
ISAXIS(< 기하 축 번호 >)
< 모듈로 위치 >=MODAXVAL(< 축 >,< 축 위치 >)

의미

AXNAME	입력 문자열을 축 식별자로 변환합니다 . 입력 문자열에 반드시 유효한 축 이름이 포함돼야 합니다 .
AX	변수 축 식별자
SPI	스핀들 번호를 축 식별자로 변환합니다. 전송 파라미터에 반드시 유효한 스핀들 번화가 포함돼야 합니다 .
n	스핀들 번호
AXTOSPI	축 식별자를 정수 스핀들 색인으로 변환합니다 . AXTOSPI 은 SPI 로의 변환 기능에 해당합니다 .
X, Y, Z	변수 또는 상수로 표현된 AXIS 형식의 축 식별자
AXSTRING	관련 스핀들 번호와 함께 문자열이 출력됩니다 .
ISAXIS	지정된 기화 축의 존재 여부를 확인합니다 .
MODAXVAL	모듈로 로타리 축을 대상으로 모듈로 위치를 파악합니다 . 이는 파라미터로 표시된 모듈로 범위에 명시되는 모듈로 리셋에 해당합니다 (기본 설정은 0 ~ 360 도입니다 . 모듈로 범위 시작 및 크기는 MD30340 MODULO_RANGE_START 와 MD30330 $MA_MODULO_RANGE 를 사용해서 변경할 수 있습니다).

주

SPI 확장

축 함수 SPI(n) 은 프레임 요소를 판독하거나 기록하는 용도로도 사용할 수 있습니다 . 예를 들어 , 구문 $P_PFRAME[SPI(1),TR]=2.22 로 프레임을 기록할 수 있습니다 .

주소 AX[SPI(1)]=< 축 위치 > 를 사용하여 축 위치를 추가로 프로그래밍함으로써 축을 이송할 수 있습니다 . 단 , 스핀들이 위치 지정 모드나 축 모드여야 한다는 전제 조건이 있습니다 .

예

예제 1: AXNAME, AX, ISAXIS

프로그램 코드	비고
OVRA[AXNAME(" 이송 축 ")]=10	; 이송 축 오버라이드
AX[AXNAME(" 이송 축 ")]=50.2	; 이송 축의 끝점
OVRA[SPI(1)]=70	; 스핀들 1 오버라이드
AX[SPI(1)]=180	; 스핀들 1 의 끈점
IF ISAXIS(1) == FALSE GOTOF CONTINUE	; 가로 축 이용 가능 여부
AX[$P_AXN1]=100	; 가로 축 이동
CONTINUE:	

예제 2: AXSTRING

AXSTRING[SPI(n)] 으로 프로그래밍한 경우 , 스핀들에 할당된 축의 축 색인이 더 이상 스핀들 번호로서 출력되지 않는 대신 문자열 "Sn" 이 출력됩니다 .

프로그램 코드	비고
AXSTRING[SPI(2)]	; 문자열 "S2" 가 출력됩니다 .

예제 3: MODAXVAL

모듈로 로타리 축 A 의 모듈로 위치를 파악해야 합니다 .

축 위치 372.55 가 계산에 사용될 시작값입니다 .

파라미터로 표현된 모듈로 범위는 0 ~ 360 도입니다 .

MD30340 MODULO_RANGE_START = 0

MD30330 $MA_MODULO_RANGE = 360

프로그램 코드	비고
R10=MODAXVAL(A,372.55)	; 계산된 모듈로 위치 R10 = 12.55.

예제 4: MODAXVAL

프로그래밍된 축 식별자가 모듈로 로타리 축을 참조하지 않는 경우 , 변환돼야 할 값 (< 축 위치 >) 이 변경되지 않고 반환됩니다 .

프로그램 코드	비고
R11=MODAXVAL(X,372.55)	; X 는 선형 축입니다 . R11 = 372.55.

14.2 대체 가능한 기하 축 (GEOAX)

기능

' 대체 가능한 기하 축 ' 기능을 사용하면 기계 데이터를 통해 구성된 기하 축 그룹화를 가공 프로그램에서 수정할 수 있습니다. 어떤 기하 축이든 특수 동기화 축으로 정의된 채널 축으로 대체할 수 있습니다 .

구문

GEOAX(<n>,< 채널 축 >,<n>,< 채널 축 >,<n>,< 채널 축 >)
GEOAX()

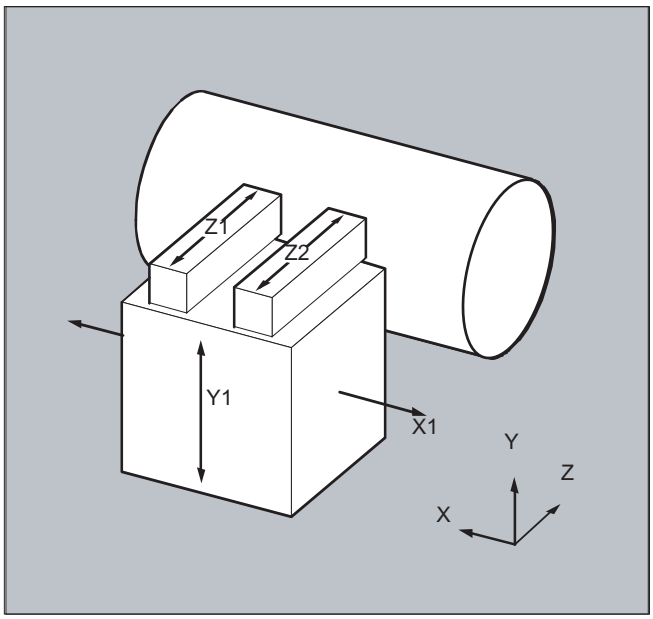
의미

GEOAX(...)	기하 축 대체 명령 **참고 :** GEOAX() 에 파라미터가 없는 경우 , 기하 축의 기본 구성이 호출됩니다 .
<n>	이 파라미터는 차후에 지정되는 채널 축에 할당될 기하 축 번호를 지정하는 데 사용됩니다 . 값의 범위 : 1, 2, 또는 3 **참고 :** <n>=0 인 경우 , 차후에 지정되는 채널 축이 대체되지 않고 기하 축 그룹에서 완전히 제거될 수 있습니다 .
< 채널 축 >	이 파라미터는 기하 축 그룹에 포함될 채널 축의 이름을 지정하는 데 사용됩니다 .

예

예제 1: 기하 축으로 대체하도록 2 개의 축 전환

채널 축 X1, Y1, Z1, Z2 을 이용하여 공구 슬라이드를 이송할 수 있습니다 .



전원 가동 후 먼저 Z1 이 기하 축의 이름 'Z' 에 준하여 3 차 기하 축으로 적용되고 X1 및 Y1 과 함께 기하 축 그룹을 형성하도록 기하 축들이 구성됩니다 .

기하축 Z1 과 Z2 이 이제 가공 프로그램에서 기하 축 Z 으로 대체되어 사용됩니다 .

프로그램 코드	비고
...	
N100 GEOAX(3,Z2)	; 채널 축 Z2 가 3 차 기하 축 (Z) 역할을 합니다 .
N110 G1 ...	
N120 GEOAX(3,Z1)	; 채널 축 Z1 이 3 차 기하 축 (Z) 역할을 합니다 .
...	

예제 2: 6 개의 채널을 기하 축으로 대체

한 대의 기계에는 XX, YY, ZZ, U, V, W 라는 이름의 채널 축 6 개가 있습니다 .

기계 대이터를 통한 기하 축 구성에 관한 기본 설정은 다음과 같습니다 .

채널 축 XX = 1 차 기하 축 (X 축)

채널 축 YY = 2 차 기하 축 (X 축)

채널 축 ZZ = 3 차 기하 축 (Z 축)

프로그램 코드	비고
N10 GEOAX()	; 기하 축의 기본 구성이 적용됩니다 .
N20 G0 X0 Y0 Z0 U0 V0 W0	; 모든 축이 위치 0 으로 신속하게 이송합니다 .
N30 GEOAX(1,U,2,V,3,W)	; 채널 축 U 가 1 차 (X), V 가 2 차 (Y), W 가 3 차 (Z) 기하 축이 됩니다 .
N40 GEOAX(1,XX,3,ZZ)	; 채널 축 XX가 1차, ZZ가 3차(Z) 기하 축이 됩니다. 채널 축 V는 2차(Y) 기하축을 유지합니다 .
N50 G17 G2 X20 I10 F1000	; X/Y 평면의 완전 원 . 채널 축 XX 및 V 가 이송합니다 .
N60 GEOAX(2,W)	; 채널 축 W 가 2 차 (Y) 기하축이 됩니다 .
N80 G17 G2 X20 I10 F1000	; X/Y 평면의 완전 원 . 채널 축 XX 및 W 가 이송합니다 .
N90 GEOAX()	; 초기 상태로 리셋됩니다 .
N100 GEOAX(1,U,2,V,3,W)	; 채널 축 U 가 1 차 (X), V 가 2 차 (Y), W 가 3 차 (Z) 기하 축이 됩니다 .
N110 G1 X10 Y10 Z10 XX=25	; 채널 축 U, V, W가 각각 위치 10으로 이송합니다. 보충 축 역할을 하는 XX 가 위치 25 로 이송합니다 .
N120 GEOAX(0,V)	; V 가 기하 축 그룹에서 제거됩니다 . U 와 W 는 1 차 (X) 및 3 차 (Z) 기하 축을 유지합니다 . 2 차 (Y) 기하 축은 미할당 상태를 유지합니다 .
N130 GEOAX(1,U,2,V,3,W)	; 채널 축 U는 1차(X) 기하 축을 유지하고 V가 2차(Y), W는 3차(Z) 기하 축을 유지합니다 .
N140 GEOAX(3,V)	; V 는 W 를 덮어써서 3 차 (Z) 기하 축이 되고 기하 축 그룹에서 제외됩니다 . 2 차 (Y) 기하 축은 이전과 마찬가지로 미할당 상태를 유지합니다 .

주

축 구성

아래 기계 데이터는 기하 축 , 특수 축 , 채널 축 및 기계 축 외에도 개별 축 유형의 이름을 할당하는 데 사용됩니다 .

MD20050 $MC_AXCONF_GEOAX_ASIGN_TAB(채널 축에 기하 축

할당)

MD20060 $MC_AXCONF_GEOAX_NAME_TAB(채널에서 기하 축 이름)

MD20070 $MC_AXCONF_MACHAX_USED(채널에서 유효한 기계 축 번호)

MD20080 $MC_AXCONF_CHANAX_NAME_TAB(채널에서 채널 축 이름)

MD10000 $MN_AXCONF_MACHAX_NAME_TAB(기계 축 이름)

MD35000 $MA_SPIND_ASSIGN_TO_MACHAX(기계 축에 스핀들 할당)

참조 :
기능 설명서의 기본 기능 , 축 , 좌표계 , 프레임 (K2)

제한사항

- 다음 상태에서는 기하 축을 대체할 수 없습니다 .
  - 변환 활성 상태
  - 스플라인 보간 활성 상태
  - 공구 반경 보정 활성 상태
  - 미세 공구 보정 활성 상태
- 기하 축과 채널 축의 이름이 동일한 경우 , 해당 기하 축을 변경할 수 없습니다 .
- 대체에 관련된 모든 기하 축은 예를 들어, A 형식 위치 지정 축이나 추종 축처럼 블록 한계를 벗어난 동작에 포함될 수 없습니다 .
- 시스템 전원을 가동했을 때 (결과적으로 , 새로 정의된 기하 축이 존재하지 않습니다), 이미 존재하는 기하 축의 경우 GEOAX 명령을 사용해서만 대체할 수 있씁니다 .
- 형상 테이블 (CONTPRON, CONTDCON) 을 실행하는 동안 GEOAX 으로 축을 대체하려고 시도한 경우 알람이 출력됩니다 .

보충 조건

대체 후 축 상태

채널 축 이름을 통한 대체 작업 후에 기하 축 그룹에서 대체 기능에 의해 대체된 축을 보충 축으로 프로그래밍할 수 있습니다 .

프레임 , 보호 영역 , 작업 영역 한계

기하 축 대체 후에는 모든 프레임 , 보호 영역 , 작업 영역 한계가 삭제됩니다 .

극 좌표계

GEOAX 을 사용하여 기하 축을 대체하면 아날로그가 G17-G19 를 이용한 레벨 변경으로 설정되고 모달 극 좌표가 0 의 값으로 설정됩니다 .

DRF, ZO

가능한 핸드휠 옵셋 (DRF) 또는 외부 워크 옵셋 (ZO) 은 대체 후에도 적용 상태를 유지합니다 .

기하 축의 기본 구성

GEOAX() 명령은 기하 축 그룹의 기본 구성을 호출합니다 .

전원 가동 이후 또는 '원점 복귀' 모드로 전환했을 때 시스템이 기본 구성으로 자동 복구됩니다.

공구 길이 보정

활성 상태의 공구 길이 보정 역시 대체 후에도 적용 상태를 유지합니다 . 그러나 새로 추가되거나 위치가 대체된 기하 축의 경우 , 아직 이동하지 않은 것으로 인식됩니다 . 이 기하 축에 대한 첫 번째 이동 명령의 경우 , 산출되는 이송 거리는 공구 길이 보정 총계와 프로그래밍된 이송 거리로 구성됩니다 .

대체 작업 이후 축 그룹에서 기존 위치를 유지하는 기하 축은 공구 길이 보정과 관련해서도 원래 상태를 유지합니다 .

변환 활성 상태와 관련한 기하 축 구성

(기계 데이터를 사용하여 정의된) 변환 활성 상태에 적용될 수 있는 기하 축 구성은 ' 대체 가능한 기하 축 ' 기능을 사용해서 변경할 수 없습니다 .

변환과 관련하여 기하 축 구성을 수정할 필요가 있는 경우 , 추가 변환을 이용하는 것이 유일한 방법입니다 .

A geometry axis configuration changed using GEOAX 를 사용하여 변경된 기하 축 구성은 변환을 활성화함을써 삭제할 수 있습니다 .

변환과 관련한 기계 데이터 설정이 기하 축 대체와 관련한 기계 데이터 설정과 일치하지 않은 경우 , 변환 관련 설정이 우선 적용됩니다 .

예 :

하나의 변환이 활성 상태입니다 . 기계 데이터에 따라 변환이 유지돼야 리셋이 가능합니다 . 하지만 그와 동시에 기하 축의 기본 구성이 마련돼야만 합니다 . 이런 경우에는 변환을 사용하여 정의된 기하 축 구성이 유지됩니다 .

14.3 축 컨테이너 (AXCTSWE, AXCTSWED, AXCTSWEC)

기능

로터리 인덱싱 기계 및 멀티스핀들 기계에서 공작물을 고정하는 축은 기존 가공 장치에서 다른 가공 장치로 이동합니다. 가공 장치는 다양한 채널에 종속되기 때문에 스테이션/위치가 변경된 경우 상응하는 채널에 공작물을 고정하는 축이 동적으로 재할당돼야 합니다 . 축 컨테이너는 이런 용도로 사용됩니다 .

한번에 오직 하나의 공작물 클램핑 축 / 스핀들만 로컬 가공에 사용됩니다 . 축 컨테이너는 모든 클램핑 축 / 스핀들에 가능한 연결을 겸비하지만 가공 장치가 사용할 수 있도록 그 중 하나만을 활성화합니다 .

셋팅 데이터를 통해 입력되는 증분 데이터 (슬롯 번호) 를 통해 축 컨테이너의 엔트리를 바꿈으로써 , 축 컨테이너에 정의된 이용 가능한 축을 변경할 수 있습니다 .

AXCTSWE 또는 AXCTSWED 를 사용하여 축 컨테이너 회전 인에이블을 가공 프로그램 또는 동기 동작에 프로그래밍할 수 있습니다 . 이 축 컨테이너에 포함된 축의 모든 채널에서 인에이블 신호를 사용할 수 있게 되면 즉시 회전이 실행됩니다 .

AXCTSWEC 명령을 프로그래밍하여 가공 프로그램 또는 동기 동작에서 축 컨테이너 회전 인에이블을 취소할 수 있습니다 (동기 동작 프로그래밍에 대해서는 " 축 컨테이너 회전에 대한 인에이블 취소 (AXCTSWEC) [쪽 608]" 참조).

구문

AXCTSWE(< 축 컨테이너 >)
AXCTSWED(< 축 컨테이너 >)
AXCTSWEC(< 축 컨테이너 >)

중요도

AXCTSWE: 축 컨테이너 회전 요청

프로그램 처리는 AXCTSWE 로 중단되지 않습니다 . 제어 시스템에서 컨테이너 축과 관련한 모든 채널의 허용 신호를 모두 사용할 수 있는 경우, SD41700 $SN_AXCT_SWWIDTH[<컨테이너 번호 >] 에 저장된 컨테이너 관련 증분 데이터에 따라 컨테이너가 회전합니다 .

AXCTSWED: 활성 채널의 배타적 동작에 따른 축 컨테이너 회전 요청

참고

가공 프로그램 또는 동기 동작 스타트업을 단순화하기 위한 명령 버전 .

참고

연관된 축 컨테이너에 포함된 축의 추가 채널과 관련된 동작은 다음 머신 데이터를 사용하여 지정할 수 있습니다 .

MD12760 $MN_ AXCT_FUNCTION_MASK, Bit 0

AXCTSWEC: 축 컨테이너 회전 인에이블 취소

참고

인에이블된 축 컨테이너 회전은 아직 회전이 시작되지 않은 경우에만 취소할 수 있습니다 .

$AN_AXCTSWA[< 축 컨테이너 >] == 0

시스템 변수에 대해서는 " 축 컨테이너 (AXCTSWE, AXCTSWED, AXCTSWEC) [쪽 675]" 참조

< 축 컨테이너 >: 축 컨테이너의 식별자

가능한 데이터 :

CT< 컨테이너 번호 >:	축 컨테이너 번호는 CT 문자 조합에 추가됩니다 . 예 : CT3
< 컨테이너 이름 >:	MD12750 $MN_AXCT_NAME_TAB 를 사용하여 설정한 축 컨테이너의 개별적인 이름 예 : A_CONT3
< 축 이름 >:	관련 채널에서 알려진 컨테이너 축의 축 이름

일반 조건

AXCTSWEC 를 호출하기 전에 컨테이너 축 사용

AXCTSWE 에서는 프로그램 처리가 중단되지 않으므로 , 동기 동작 DO AXCTSWEC 를 프로그래밍할 때는 다음 사항을 준수해야 합니다 .

예 :

프로그램 코드	코멘트
N10 AXCTSWE(CT3)	; 축 컨테이너 회전 인에이블 .
N20 AX_A10	; AX_A = 컨테이너 축 . ; 시스템은 축 컨테이너 회전이 완료될 때까지 대기합니다 . $AN_AXCTSWA[CT3] == 0
WHEN <condition> DO AXCTSWEC(AX_A)	; 인에이블 취소 . **영향 없음 !**
N30 G4 F1	

블록 N20을 통한 축 컨테이너의 회전 인에이블을 포함하는 블록 N10 다음에 축 컨테이너의 축 (AX_A) 이 사용되고 이로 인해 시스템은 축 컨테이너 회전이 끝날 때까지 기다리게 되므로 , 동기 동작은 메인 실행에서 프로그램 블록 N30 과 함께 실행되며 아무런 영향을 미치지 않습니다 .

해결책 :

프로그램 코드	코멘트
N11 AXCTSWE(CT3)	; 축 컨테이너 회전 인에이블 .
WHEN <condition> DO AXCTSWEC(AX_A)	; 인에이블 취소
N21 ...	; 실행 가능한 NC 블록 .
N31 AX_A10	; 시스템은 축 컨테이너 회전이 완료될 때까지 대기합니다 . $AN_AXCTSWA[CT3] == 0

유의사항
실행 가능한 블록 N21 이 없으면 동기 동작은 메인 실행에서 다음으로 실행 가능한 프로그램 블록 N31 을 포함한 축 컨테이너 회전이 끝난 후에 구현되므로 위 예제와 마찬가지로 아무 영향을 미치지 않습니다 .

도 참조

축 컨테이너 (AXCTSWE, AXCTSWED, AXCTSWEC) 축 컨테이너 (AXCTSWE, AXCTSWED, AXCTSWEC) [쪽 675]

추가 정보

축 컨테이너

축 컨테이너를 통해 다음을 할당할 수 있습니다 .

- 로컬 축 및 / 또는
- 링크 축

링크 축이 있는 축 컨테이너는 제어 시스템을 통해 조정되는 NCU 교차 장치 (NCU 기본 장치) 입니다 . 또한 로컬 축 관리 전용으로 사용되는 축 컨테이너를 갖출 수도 있습니다 .

참조 :
축 컨테이너 구성에 대한 자세한 내용은
기능 매뉴얼 , 확장 기능 , B3: NCU 화면 조작반 , 분산형 시스템 참조

허용 기준

AXCTSWE( )

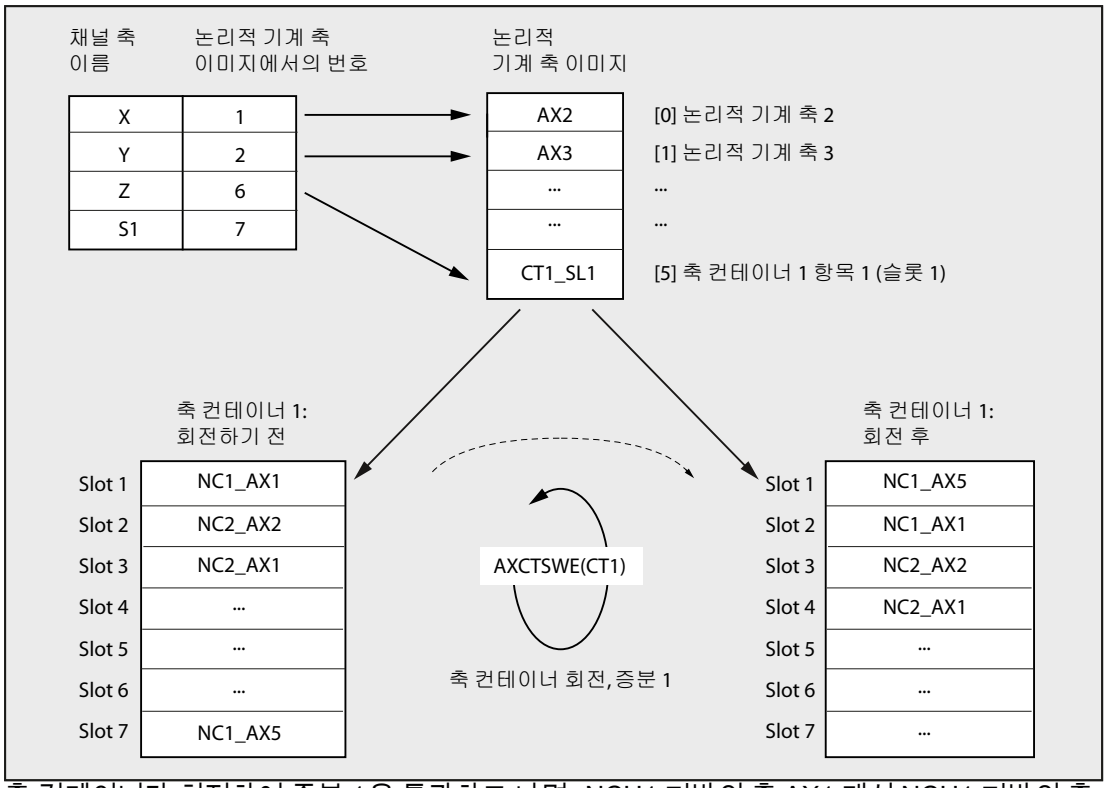
채널별로 축 컨테이너 회전을 인에이블하려면 AXCTSWE 명령을 사용합니다 . 컨테이너에 축을 가진 모든 채널이 인에이블 신호를 내리는 경우 회전이 실행됩니다 .

셋팅 데이터에 저장된 증분으로 축 컨테이너 회전이 실행됩니다 .

SD41700 $SN_AXCT_SWWIDTH[< 축 컨테이너 >]

다음 예제 참조

예 :



축 컨테이너가 회전하여 증분 1을 통과하고 나면, NCU1 기반의 축 AX1 대신 NCU1 기반의 축 AX5 에 채널 축 Z 가 할당됩니다 .

AXCTSWED( )

AXCTSWED 명령은 가공 프로그램 또는 동기 동작 스타트업을 단순화하는 데 사용됩니다 . AXCTSWED 명령이 실행되는 즉시 축 컨테이너 회전이 이루어집니다 . 이 축 컨테이너에 축을 가진 다른 채널들이 인에이블 신호를 반드시 내려야 하는 것은 아닙니다 . 인에이블된 축 컨테이너 회전이 실행되도록 하려면 특정 채널이 리셋 상태에 있어야 합니다 . 이와 관련하여 어떤 채널을 고려해야 하는지 선택하려면 다음 머신 데이터를 사용합니다 .

MD12760 $MN_ AXCT_FUNCTION_MASK		
비트	값	중요도
0	0	다이렉트 축 컨테이너 회전 (AXCTSWED) 을 위해서는 다른 모든 채널이 리셋 상태여야 합니다 .
	1	다이렉트 축 컨테이너 회전 (AXCTSWED) 을 위해서는 축 컨테이너의 축에 대한 보간 권한이 있는 채널만 리셋 상태여야 합니다 .

셋팅 데이터에 저장된 증분으로 축 컨테이너 회전이 실행됩니다 .

SD41700 $SN_AXCT_SWWIDTH[< 축 컨테이너 >]

효율성

축 컨테이너가 회전하고 나면 채널이 논리 기계 축 이미지를 통해 회전 축 컨테이너를 참조하는 모든 NCU 가 새 축 할당에 포함됩니다 .

진단

축 컨테이너의 실제 상태는 다음 시스템 변수를 사용하여 가공 프로그램 및 동기 동작에서 읽을 수 있습니다 .

시스템 변수	종류	설명	
$AC_AXCTSWA[< 축 컨테이너 >]	BOOL	축 컨테이너의 채널별 상태	
		1	지정된 축 컨테이너의 채널이 축 컨테이너 회전을 인에이블했습니다 . 회전이 아직 실행되지 않았습니다 .
		0	축 컨테이너 회전이 실행되었습니다 .
$AN_AXCTSWA[< 축 컨테이너 >]	BOOL	축 컨테이너의 NCU 별 상태	
		1	NCU의 모든 채널이 축 컨테이너 회전을 인에이블했습니다. 축 컨테이너가 현재 회전 중입니다 .
		0	NCU 의 모든 채널이 축 컨테이너 회전을 인에이블한 것은 아닙니다. 현재 아무런 축 컨테이너도 회전하고 있지 않습니다.
$AN_AXCTSWE[< 축 컨테이너 >]	INT	축 컨테이너 회전의 슬롯별 상태 시스템 변수는 축 컨테이너 슬롯의 상태를 **비트별로** 제공합니다. 각 비트는 슬롯에 해당합니다 .	
		1	슬롯에 회전이 인에이블되었습니다 .
		0	슬롯에 회전이 인에이블되지 **않았습니다** .
$AN_AXCTAS[< 축 컨테이너 >]	INT	축 컨테이너가 연결된 위치 (슬롯) 의 수 . 전원을 켠 후의 초기화 값 : 0	
		값 범위 :	0 ... 축 컨테이너에 할당된 최대 슬롯 수 - 1

축 컨테이너 회전에 대한 인에이블 취소 (AXCTSWEC)

필요한 경우에는 축 컨테이너 회전을 위한 실제 채널 인에이블을 다음과 같은 방법으로 취소할 수 있습니다 .

- 가공 프로그램에서 AXCTSWEC 프로그래밍
- 동기 동작 사용 (" 축 컨테이너 회전에 대한 인에이블 취소 (AXCTSWEC) [쪽 608]" 참조)

기능을 활성화하려면 다음 조건이 충족되어야 합니다 .

- 실제 채널이 축 컨테이너 회전 인에이블 신호를 이미 전송한 상태여야 합니다 (이 채널의 경우에는 시스템 변수 $AC_AXCTSWA[<n>] = 1).
- 축 컨테이너 회전이 아직 시작되지 않은 상태여야 합니다 (예 : 시스템 변수 $AN_AXCTSWA[<n>] = 0).

AXCTSWEC 명령을 실행한 후 시스템 변수 $AC_AXCTSWA[<n>] 는 이 특정 채널에서 "0" 값을 전송합니다 .

암묵적 GET/GETD 를 이용한 축 컨테이너 회전

축 컨테이너 회전이 인에이블되면 GET 또는 GETD 로 모든 컨테이너 축이 해당 채널에 할당됩니다 . 축 컨테이너가 회전하기 전에는 축이 해제되지 않습니다 .

주

이런 특성은 기계 데이터를 사용하여 설정할 수 있습니다 . 장비 제조업체의 설명서를 참조하십시오 .

주

암묵적 GET/GETD 가 사용된 축 컨테이너 회전은 " 메인 실행 축 " 상태의 축 (예 : PLC 축) 에 사용할 수 **없습니다** . 축 컨테이너 회전을 위해 이 축이 메인 실행 상태에서 제외돼야 하기 때문입니다 .

14.4 유효 축 위치 대기 (WAITENC)

기능

NC 프로그램은 언어 명령 WAITENC를 사용하여 MD34800 $MA_WAIT_ENC_VALID = 1으로 설정된 축에 동기 축 위치 또는 복구된 축 위치를 사용할 수 있을 때까지 대기합니다 .

대기 상태에서 ASUB 를 시작하거나 조작 모드를 JOG 로 바꾸면 대기 상태가 중단됩니다 . 프로그램이 계속되면 대기 상태가 재개됩니다 .

주

사용자 인터페이스에는 대기 상태가 보류 상태 " 측정 시스템 대기 " 로 표시됩니다 .

구문

WAITENC 는 모든 NC 프로그램의 프로그램 섹션에서 프로그래밍이 가능합니다 .

단 , 다음과 같이 전용 블록에서 프로그래밍해야 합니다 .

```
...
WAITENC
...
```

예제

WAITENC 는 다음 적용 예제와 같이 이벤트 제어 사용자 프로그램 (예 : .../_N_CMA_DIR/_N_PROG_EVENT_SPF) 에서 사용됩니다 .

적용 예제 : 오리엔테이션 변환 중 전원 공급이 중단된 이후 공구 후퇴

공구 오리엔테이션을 적용하여 가공하는 도중 정전으로 가공이 중단되었습니다 .
다시 전원이 들어오면 이벤트 제어 사용자 프로그램 .../_N_CMA_DIR/_N_PROG_EVENT_SPF 가 호출됩니다 .

이벤트 제어 사용자 프로그램에서 시스템은 WAITENC 를 사용하여 동기 축 위치 또는 복구된 축 위치를 기다립니다 . 이 위치가 알게 되면 프레임을 계산하고 이에 따라 워크의 공구 방향을 정렬할 수 있습니다 .

프로그램 코드	설명
...	
IF $P_PROG_EVENT == 4	; 기동
IF $P_TRAFO <> 0	; 변환이 선택되었습니다 .
WAITENC	; 오리엔테이션 축의 유효 축 위치를 확인할 때까지 대기합니다 .
TOROTZ	; 워크의 Z 축을 공구 축을 향해 회전시킵니다 .
ENDIF	
M17	
ENDIF	
...	

JOG 모드에서 공구를 공구 축 방향으로 후퇴 이동시켜 공구를 후퇴시킬 수 있습니다 .

14.5 NC 언어 (STRINGIS) 의 범위 확인

기능

STRINGIS(...) 함수를 사용하면 지정된 문자열을 실제 언어 범위에서 NC 프로그래밍 언어 요소로 사용할 수 있는지 여부를 확인할 수 있습니다 .

정의

INT STRINGIS(STRING < 이름 >)

구문

STRINGIS(< 이름 >)

의미

STRINGIS:	리턴 값이 있는 함수
< 이름 >:	검사할 NC 프로그래밍 언어 요소의 이름
리턴 값 :	리턴 값 형식은 yxx (10 진수)

NC 프로그래밍 언어 요소

다음과 같은 NC 프로그래밍 언어 요소를 검사할 수 있습니다 .

- 기존 G 코드 그룹의 G 코드 (예 : G0, INVCW, POLY, ROT, KONT, SOFT, CUT2D, CDON, RMB, SPATH)
- DIN 또는 NC 어드레스 (예 : ADIS, RNDM, SPN, SR, MEAS)
- 함수 (예 : TANG(...) 또는 GETMDACT)
- 절차 (예 : SBLOF)
- 키워드 (예 : ACN, DEFINE 또는 SETMS)
- 시스템 데이터 (예 : 머신 데이터 $M..., 셋팅 데이터 $S..., 옵션 데이터 $O...)
- 시스템 변수 $A... , $V... , $P...
- 산술 파라미터 R...
- 활성화된 싸이클의 싸이클 이름
- GUD 및 LUD 변수
- 매크로 이름
- 라벨명

리턴 값

리턴 값 중 처음 세 자리만 유효합니다. 리턴 값 형식은 yxx입니다. 여기서 y는 기본 정보, xx는 상세 정보입니다 .

리턴 값	의미	
000	이 시스템에서 알지 못하는 ' 이름 ' 문자열입니다 .[1]	
100	'이름' 문자열이 NC 프로그래밍 언어 요소이지만 현재 프로그래밍할 수 없습니다 (옵션/기능 비활성 상태).	
2xx	' 이름 ' 문자열이 NC 프로그래밍 언어 요소이며 프로그래밍 가능합니다 (옵션 / 기능 활성 상태). 상세 정보 xx 는 요소 유형에 대한 추가 정보를 포함하고 있습니다 .	
	xx	의미
	01	DIN 어드레스 또는 NC 어드레스 [2]
	02	G 코드 (예 : G04, INVCW)
	03	리턴 값이 있는 함수
	04	리턴 값이 없는 함수
	05	키워드 (예 : DEFINE)
	06	머신 데이터 ($M...), 셋팅 데이터 ($S...) 또는 옵션 데이터 ($O...)
	07	시스템 파라미터 (예 : 시스템 변수 ($...)) 또는 산술 파라미터 (R...)
	08	싸이클 (싸이클은 NCK 에 로드되어야 하고 싸이클 프로그램은 활성화되어야 합니다 .) [3]
	09	GUD 변수 (GUD 변수는 GUD 정의 파일에 정의해야 하고 활성화해야 합니다 .)
	10	매크로 이름 (매크로는 매크로 정의 파일에 정의해야 하고 활성화시켜야 합니다 .) [4]
	11	실제 가공 프로그램의 LUD 변수
	12	ISO G 코드 (ISO 언어 모드가 활성화되어야 합니다 .)
400	' 이름 ' 문자열은 xx == 01 또는 xx == 10 로 식별되지 않았던 NC 어드레스이며 G 또는 R 코드가 아닙니다 .	
y00	구체적인 지정 불가능	

1) 제어 시스템에 따라 특정 상황에서는 Siemens NC 언어 명령의 서브세트만 인식됩니다 (예: SINUMERIK 802D sl). 이러한 시스템의 경우 원칙적으로 Siemens NC 언어 명령인 문자열에는 값 0 이 반환됩니다 . MD10711 $MN_NC_LANGUAGE_CONFIGURATION 을 사용하면 이와 같은 동작을 변경할 수 있습니다 . MD10711 = 1 인 경우 Siemens NC 언어 명령에 항상 100 이라는 값이 반환됩니다 .

2) NC 어드레스는 문자 A, B, C, E, I, J, K, Q, U, V, W, X, Y, Z 입니다 . 또한 , 이 NC 어드레스는 어드레스 확장자와 함께 프로그래밍할 수 있습니다 . 어드레스 확장자는 STRINGIS 로 검사할 때 지정할 수 있습니다 . 예제 : 201 == STRINGIS("A1").
문자 D, F, H, L, M, N, O, P, S, T 는 NC 어드레스이거나 사용자가 정의한 보조 기능입니다 . 이 NC 어드레스나 보조 기능에는 값 400 이라는 값이 반환됩니다 . 예제 : 400 == STRINGIS("D"). 이 NC 어드레스는 STRINGIS 로 검사할 때 어드레스 확장자와 함께 지정할 수 없습니다 .
예 : 000 == STRINGIS("M02"), 400 == STRINGIS("M").

3) 싸이클 파라미터의 이름은 STRINGIS 로 검사할 수 없습니다 .
4) G, H, M, L 처럼 매크로로 정의된 어드레스는 매크로로 식별됩니다 .

예제

다음 예제에서는 문자열로 지정된 NC 언어 요소를 별도 언급이 없는 한 원칙적으로 제어 시스템에 프로그래밍할 수 있다고 가정합니다 .

1. 문자열 "T" 는 보조 기능으로 정의됩니다 .

   400 == STRINGIS("T")

   000 == STRINGIS ("T3")

2. 문자열 "X" 는 축으로 정의됩니다 .

   201 == STRINGIS("X")

   201 == STRINGIS("X1")

3. 문자열 "A2" 는 확장자가 있는 어드레스로 정의됩니다 .

   201 == STRINGIS("A")

   201 == STRINGIS("A2")

4. 문자열 "INVCW" 는 G 코드로 정의됩니다 .

   202 == STRINGIS("INVCW")

5. 문자열 "$MC_GCODE_RESET_VALUES" 는 머신 데이터로 정의됩니다 .

   206 == STRINGIS("$MC_GCODE_RESET_VALUES")

6. 문자열 "GETMDACT" 는 NC 언어 함수입니다 .

   203 == STRINGIS("GETMDACT ")

7. 문자열 "DEFINE" 은 키워드입니다 .

   205 == STRINGIS("DEFINE")

8. 문자열 "$TC_DP3" 은 시스템 파라미터 (공구 길이 콤포넌트) 입니다 .

   207 == STRINGIS("$TC_DP3")

9. 문자열 "$TC_TP4" 는 시스템 파라미터 (공구 크기) 입니다 .

   207 == STRINGIS("$TC_TP4")

10. 문자열 "$TC_MPP4" 는 시스템 파라미터 (메거진 포켓 상태) 입니다 .

    - 공구 메거진 관리가 활성 상태입니다 . 207 == STRINGIS("$TC_MPP4") ;
    - 공구 메거진 관리가 비활성 상태입니다 . 000 == STRINGIS("$TC_MPP4")

    아래 공구 메거진 관리 단원도 참조하십시오 .

11. 문자열 "MACHINERY_NAME" 은 GUD 변수로 정의됩니다 .

    209 == STRINGIS("MACHINERY_NAME")

12. 문자열 "LONGMACRO" 는 매크로로 정의됩니다 .

    210 == STRINGIS("LONGMACRO")

13. 문자열 "MYVAR" 은 LUD 변수로 정의됩니다 .

    211 == STRINGIS("MYVAR")

14. 문자열 "XYZ" 는 NCK, GUD 변수 , 매크로 또는 싸이클 이름에서 인식할 수 없는 명령입니다 .

    000 == STRINGIS("XYZ")

공구 메거진 관리

공구 메거진 관리 기능이 비활성 상태이면 STRINGIS 가 머신 데이터와 관계 없이 공구 메거진 관리의 시스템 파라미터를 제공합니다 .

- MD10711 $MN_NC_LANGUAGE_CONFIGURATION

값은 항상 000 입니다 .

ISO 모드

"ISO 모드 " 기능이 활성화 상태인 경우

- MD18800 $MN_MM_EXTERN_LANGUAGE (활성화 , 외부 NC 언어)
- MD10880 $MN_ MM_EXTERN_CNC_SYSTEM (조정할 제어 시스템)

STRINGIS 는 지정된 문자열을 먼저 SINUMERIK G 코드인지 검사합니다 . 문자열이 SINUMERIK G 코드가 아니면 이어서 ISO G 코드인지 검사합니다 .

프로그래밍된 전환 (G290 (SINUMERIK 모드), G291 (ISO 모드)) 은 STRINGIS 에 영향을 미치지 않습니다 .

예제

STRINGIS(...) 함수에 해당하는 머신 데이터는 다음과 같은 값을 갖습니다 .

- MD10711 $MN_NC_LANGUAGE_CONFIGURATION = 2 (옵션이 설정된 NC 언어 명령만 인식되는 것으로 간주)
- MD19410 $ON_TRAFO_TYPE_MASK = 'H0' (옵션 : 변환)
- MD10700 $MN_PREPROCESSING_LEVEL='H43' (싸이클 전처리가 활성 상태)

다음 프로그램 예제는 에러 메시지 없이 실행됩니다 .

프로그램 코드	설명
N1 R1=STRINGIS("TRACYL")	; R1 == 0, TRACYL 이 변환 옵션을 ; 찾을 수 없어 ' 알 수 없음 ' 으로 ; 인식됩니다 .
N2 IF STRINGIS("TRACYL") == 204	;
N3 TRACYL(1,2,3)	; N3 을 스킵합니다 .
N4 ELSE	
N5 G00	; 대신 N5 가 실행됩니다 .
N6 ENDIF	
N7 M30	

14.6 함수 호출 ISVAR 과 판독 기계 데이터 배열 색인

기능

ISVAR 명령어는 NC 언어에 정의된 함수입니다 .

- BOOL 함수 값
- 문자형 전송 파라미터

전송 파라미터가 NC(기계 데이터 , 셋팅 데이터 , 시스템 변수 , GUD 와 같은 일반 변수) 에서 인식 가능한 변수값을 포함하고있으면 , ISVAR 명령어는 TRUE 값을 반환합니다 .

구문

ISVAR(<variable identifier>)
ISVAR(<identifier>,[<value>,<value>])

의미

<Variable identifier>	문자형 전송 파라미터는 1차원, 또는 2차원으로 표시 할 수 없습니다 .
<identifier>	기계 데이터 , 셋팅 데이터 , 시스템 변수나 일반 변수처럼 배열 인덱스 있이 / 없이 인식 가능한 변수의 식별자 **확장** 일반 그리고 채널 관련 기계 데이터에서는 인덱스에 특정값이 없어도 배열의 첫 번째 요소를 읽습니다 .
<value>	BOOL 함수 값

검사

다음 검사는 전송 파라미터와 일치하여 만들었습니다 .

- 식별자가 존재하는지 검사
- 1 차원 또는 2 차원 배열인지 검사
- 허가된 배열 인덱스인지 검사

위의 모든 검사가 긍정이면 , 참 (TRUE) 값을 반환합니다 . 검사가 부정값을 가지고있거나 구문 오류 (syntax error) 가 발생하였다면 , 거짓 (FALSE) 값을 반환합니다 . 축 변수들은 축 이름을 인덱스로 표시하나 검사되지 않습니다 .

확장 : 인덱스 없이 기계 데이터와 셋팅 데이터 배열 읽기

일반 그리고 채널 관련 기게 데이터를 위한 인덱스가 없다면 , 알람 12400 " 채널 % 1 블럭 % 2 배열 % 3 요소가 존재하지 않습니다 " 이 **더 이상** 발생하지 않습니다 .

적어도 축 관련 기계 데이터를 위한 **축 인덱스**는 반드시 프로그래밍 해야합니다 . 그렇지 않으면 알람 12400 이 발생합니다 .

예 : 함수 호출 ISVAR

프로그램 코드	설명
DEF INT VAR1	
DEF BOOL IS_VAR=FALSE	; 전송 파라미터는 일반 변수 입니다 .
N10 IS_VAR=ISVAR("VAR1")	; 이 경우 , IS_VAR 은 참 (TRUE) 입니다 .
DEF REAL VARARRAY[10,10]	
DEF BOOL IS_VAR=FALSE	; 다양한 구문 버전 (syntax version)
N20 IS_VAR=ISVAR("VARARRAY[,]")	; IS_VAR 은 2 차원 배열로 참 (TRUE) 입니다 .
N30 IS_VAR=ISVAR("VARARRAY")	; IS_VAR 은 참 (TRUE), 변수 존재
N40 IS_VAR=ISVAR("VARARRAY[8,11]")	; IS_VAR 은 거짓 (FALSE), 배열 인덱스 허가되지 않음
N50 IS_VAR=ISVAR("VARARRAY[8,8")	; IS_VAR 은 거짓 (FALSE), "]" 누락 구문 오류 (syntax error)
N60 IS_VAR=ISVAR("VARARRAY[,8]")	; IS_VAR 은 참 (TRUE), 배열 인덱스 허가되지 않음
N70 IS_VAR=ISVAR("VARARRAY[8,]")	; IS_VAR 은 참 (TRUE) 입니다 .
DEF BOOL IS_VAR=FALSE	; 전송 파라미터는 기계 데이터입니다 .
N100 IS_VAR=ISVAR("$MC_GCODE_RESET_VALUES[1]"	; IS_VAR 은 참 (TRUE) 입니다 .
DEF BOOL IS_VAR=FALSE	; 전송 파라미터는 시스템 변수 입니다 .
N10 IS_VAR=ISVAR("$P_EP")	; 이 경우 , IS_VAR 은 참 (TRUE) 입니다 .
N10 IS_VAR=ISVAR("$P_EP[X]")	; 이 경우 , IS_VAR 은 참 (TRUE) 입니다 .

예 : 인덱스 있이 / 없이 기계 데이터 배열 읽기

첫 번재 요소를 읽습니다 .

R1=$MC_EXTERN_GCODE_RESET_VALUES

이전 처럼 , 다음

R1=$MC_EXTERN_GCODE_RESET_VALUES[0]

또는 첫 번째 요소를 읽습니다

R1=$MA_POSTCTRL_GAIN[X1]

이전에 상응하는

R1=$MA_POSTCTRL_GAIN[0, X1]

동기화 동작내의 첫 번째 요소를 읽습니다

WHEN TRUE DO $R1 = $MC_EXTERN_GCODE_RESET_VALUES

이전에 상응하는

WHEN TRUE DO $R1 = $MC_EXTERN_GCODE_RESET_VALUES[0]

그리고 사전에 알람 12400 과 함께 **읽지 않았습니다** .

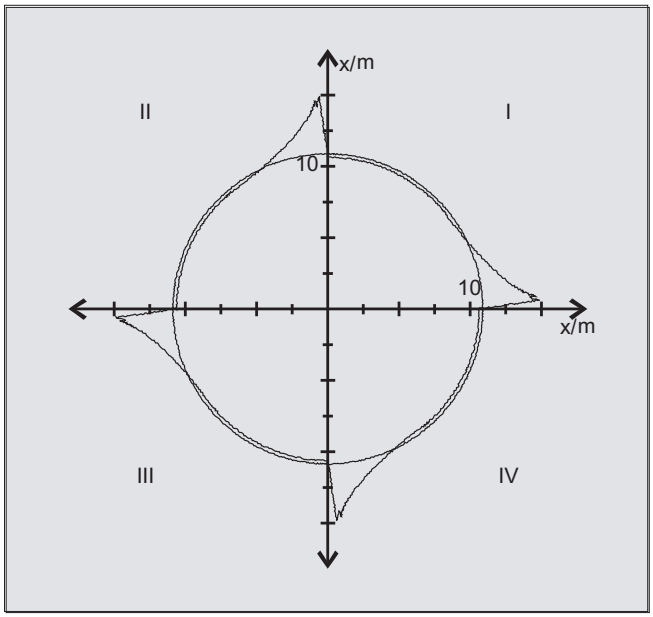
알람 12400 이 발생합니다 .

R1=$MA_POSTCTRL_GAIN

14.7 보정 특성 배우기 (QECLRNON, QECLRNOF)

기능

상한 돌기 보정 (QEC) 은 기계 비충실도 (예, 마찰, 백래시) 나 비틀림 (torsion) 으로 인해 이송 방향이 바뀔 때 발생하는 형상 오류를 감소시킵니다. 신경 회로망을 기초로 하여, 보정 특성을 자동으로 결정하기 위해 학습 단계 동안 제어로 최적의 보정 데이터를 조정 할 수 있습니다. 최대 4 축까지 동시에 학습이 적용됩니다.



구문

QECLRNON

QECLRNOF

인지 작업 활성화 : QECLRNON

축을 지정하는 QECLRNON 명령어를 사용하여 NC 프로그램상에 인지 작업이 활성화됩니다.

QECLRNON (X1, Y1, Z1, Q)

명령어가 활성화 되어 있는 경우에만 특성이 변경됩니다.

학습 비활성화 : QECLRNOF

필수 축의 학습 이동이 완료된 후, 학습 과정은 QECLRNOF 를 사용하는 모든 축에 동시에 비활성화됩니다.

의미

QECLRNON (axis 1,…4)	" 상한 돌기 보정 학습 " 기능 활성화
QECLRNO	" 상한 돌기 보정 학습 " 기능 해제
QECLRN.SPF	학습 싸이클
QECDAT.MPF	시스템 변수를 할당하고 학습 싸이클을 설정하는 NC 프로그램 예
QECTEST.MPF	순환성 테스트를 위한 NC 프로그램 예

설명

인지 과정에 필요한 축의 이송 이동은 NC 프로그램을 사용하여 실행됩니다 . 인지 이동은 인지 사이클 형식의 프로그램에 저장됩니다 .

첫 번째 학습

표준 PLC 프로그램 디스크를 포함된 샘플 NC 프로그램은 이동을 학습하는데 사용되고 제어가 시작할 동안 인지 초기 단계에서 QEC 시스템 변수를 할당합니다 .

재인지

인지된 특성들은 후속 인지로 최적화됩니다. 사용자 메모리에 저장된 데이터는 최적화를 위한 기초로 사용됩니다 . 최적화는 사용자의 요구에 맞게 샘플 NC 프로그램을 적응하는 것으로 수행됩니다 .

인지 사이클 파라미터 (예 , QECLRN.SPF) 는 " 재인지 " 로 변경되어야 합니다 .

- " 인지 모드 " = 1 로 설정
- 필요시 " 인지 실행 횟수 " 를 낮추세요
- 필요시 " 모듈러 인지 " 를 활성화하고 영역 제한을 규정하세요 .

14.8 가공 프로그램 (MMC) 에서 창 양방향 호출

기능

가공 프로그램의 HMI 상에 사용자 정의 대화 창 (대화 표시) 을 나타내기 위해 MMC 명령어를 사용할 수 있습니다 .

HMI 시스템 소프트웨어가 변경되지 않은 상태로 남아있는 동안 , 대화 창 출현이 순수 텍스트 구성 (주기 디렉토리안의 COM 파일) 에 정의되었습니다 .

사용자 정의 대화창은 다른 채널에서 동시에 호출 할 수 없습니다 .

구문

MMC(CYCLES,PICTURE_ON,T_SK.COM,BILD,MGUD.DEF,BILD_3.AWB,TEST_1,A1","S")

중요도

MMC	HMI 의 가공 프로그램에서 대화 창 양방향 호출 .
CYCLES	구성된 사용자 대화 박스가 작동 영역에 실행됩니다 .
PICTURE_ON or PICTURE_OFF	명령 : 선택을 표시 또는 비선택을 표시 .
T_SK.COM	Com 파일 : 대화 표시 파일명 (사용자 주기). 이곳에서 대화 표시 디자인을 정의합니다 . 대화 화면은 사용자 변수 그리고 / 또는 설명 텍스트를 표시하는데 사용됩니다 .
DISPLAY	대화명 표시 : 개별적 표시들이 대화명을 통해 선택됩니다 .
MGUD.DEF	변수의 읽기 / 쓰기 동안 어드레스 지정되는 사용자 데이터 정의 파일 .
PICTURE_3.AWB	그래픽 파일
TEST_1	시간 또는 인지 변수 표시 .
A1	텍스트 변수 ...",
"S"	인지 모드 : 동기식 , "OK" 소프트 키를 통한 인지 .

참조 문서

MMC 명령어 프로그래밍의 자세한 정보는 테스트 설명서를 참조하세요 (프로그래밍 예제 포함).

14.9 프로그램 실행 시간 / 가공 카운터

14.9.1 프로그램 실행 시간 / 가공 카운터 (전체 보기)

프로그램 실행 시간의 정보와 공구 카운터는 기계 공구 작동을 지원하기 위해 제공됩니다 .

본 정보는 NC 그리고 / 또는 PLC 프로그램에서 시스템 변수로 처리 될 수 있습니다 . 본 정보를 작업자 인터페이스에 표시 할 수 있습니다 .

14.9.2 프로그램 실행 시간

기능

" 프로그램 실행 시간 " 기능은 NC 및 채널 관련 시스템 변수를 통해 가공 프로그램 및 동기 동작을 읽을 수 있는 공정을 체크하는 내부 NC 타이머를 제공합니다 .

실행 시간 측정 ($AC_PROG_NET_TIME_TRIGGER) 용 트리거는 쓰기 가능하고 측정 프로그램 섹션에서 선택적으로 사용 되는 기능 시스템 변수입니다 . NC 프로그램에 트리거 쓰기에 의해 , 시간 측정이 활성화 및 비활성화 할 수 있습니다 .

시스템 변수	의미	활동
NC 관련		
$AN_SETUP_TIME	디폴트 값 (' 콜드 리스타트 ') 으로 마지막 제어기를 작동한 후 경과한 시간 (분). 디폴트 값으로 제어기를 가동할 때마다 자동으로 '0' 으로 리셋됩니다 .	• 항상 활성화
$AN_POWERON_TIME	제어기를 정상적으로 마지막 가동 ('웜 리스타트') 한 후 경과한 시간 (분). 정상적으로 제어기를 가동할 때마다 자동으로 '0' 으로 리셋됩니다 .	
채널 관련		
$AC_OPERATING_TIME	자동 모드에서 NC 프로그램의 총 런타임 (초). 제어기를 가동할 때마다 자동으로 '0' 으로 리셋됩니다 .	• MD27860 을 통해 활성화됨 • 자동 모드에서만
$AC_CYCLE_TIME	선택한 NC 프로그램의 런타임 (초). 새 NC 프로그램을 시작할 때마다 자동으로 '0' 으로 리셋됩니다 .	
$AC_CUTTING_TIME	가공 시간 (초) 경로 축 (적어도 하나가 활성 상태일 때) 의 런타임은 급 이송이 활성 상태가 아닐 때 프로그램의 NC 시작 및 종료 /NC 리셋 사이에 있는 모든 NC 프로그램에서 측정됩니다 . 드웰 시간이 동작 중일 때 측정이 중단됩니다 . 디폴트 값으로 제어기를 가동할 때마다 자동으로 '0' 으로 리셋됩니다 .	

시스템 변수	의미		활동
$AC_ACT_PROG_NET_TIME	현재 NC 프로그램의 실제 순수 런타임 (초). 새 NC 프로그램이 시작되면 자동으로 '0' 으로 리셋됩니다 .		• 항상 활성화 • 자동 모드에서만
$AC_OLD_PROG_NET_TIME	M30 으로 정확히 종료된 프로그램의 순수 실행시간 (초당)		
$AC_OLD_PROG_NET_TIME_COUNT	$AC_OLD_PROG_NET_TIME 로 변경 POWER ON 후 , $AC_OLD_PROG_NET_TIME_COUNT 의 값은 "0" 입니다 . 제어가 $AC_OLD_PROG_NET_TIME 에 새로운 기록을 남기면 $AC_OLD_PROG_NET_TIME_COUNT 값은 항상 증가합니다 .		
$AC_PROG_NET_TIME_TRIGGER	실행 시간 측정을 위한 트리거		• 자동 모드에서만
	0	중립 상태 트리거가 비활성 상태입니다 .	
	1	종료 측정을 종료하고 $AC_ACT_PROG_NET_TIME의 값을 $AC_OLD_PROG_NET_TIME 로 복사하세요 . $AC_ACT_PROG_NET_TIME 값이 "0" 으로 설정되고 계속 실행합니다 .	
	2	시작 측정을 시작합니다 . 그렇게 함으로서 $AC_ACT_PROG_NET_TIME 값을 "0" 으로 설정합니다 . $AC_OLD_PROG_NET_TIME 값은 변경되지 않았습니다 .	
	3	정지 측정을 중지 합니다 . 재개 될 때까지 $AC_OLD_PROG_NET_TIME 값을 변경하지 않고 $AC_ACT_PROG_NET_TIME 값을 유지합니다 .	
	4	재개 측정이 재개됩니다 , 예 , 이전에 중지된 측정이 계속 진행됩니다 . $AC_ACT_PROG_NET_TIME 값은 계속 실행합니다 . $AC_OLD_PROG_NET_TIME 값은 변경되지 않았습니다 .	
POWER ON 시 , 전체 시스템 변수는 0 으로 리셋됩니다 .			
참고 자료 : 나열된 시스템 변수에 대한 자세한 설명은 다음을 참조하십시오 . 기능 매뉴얼 , 기본 기능 : 모드 그룹 , 채널 , 프로그램 작동 , 리셋 응답 (K1), 프로그램 실행 시간 장			

주

장비 제조업체

머신 데이터 MD27860 $MC_PROCESSTIMER_MODE 는 활성화 가능한 타이머를 동작시키기 위해 사용됩니다 .

특정 기능 (예 : GOTOS, 오버라이드 = 0%, 활성 시운전 이송 , 프로그램 테스트 , ASUB, PROG_EVENT 등) 의 활성 시간 측정 동작은 머신 데이터 MD27850 $MC_PROG_NET_TIMER_MODE 및 MD27860 $MC_PROCESSTIMER_MODE를 사용하여 설정합니다 .

참고 자료 :
기능 매뉴얼 , 기본 기능 ; BAG, 채널 , 프로그램 작동 , 리셋 응답 (K1), 장 : 프로그램 실행 시간

주

공작물의 잔여 시간

동일한 공작물들을 차례로 생산하는 경우의 타이머 값 :

- 마지막으로 생산된 공작물의 가공 시간 ($AC_OLD_PROG_NET_TIME 참조)

및

- 현재 가공 시간 ($AC_ACT_PROG_NET_TIME 참조)

공작물의 나머지 잔여 시간을 확인하는 데 사용할 수 있습니다 .

잔여 시간은 현재 가공 시간과 함께 작업자 인터페이스에 표시됩니다 .

유의사항
STOPRE 사용 시스템 변수 $AC_OLD_PROG_NET_TIME 및 $AC_OLD_PROG_NET_TIME_COUNT 는 암시적인 전처리 정지를 발생하지 않습니다 . 이전 실행 프로그램에서 시스템 변수들의 값을 가져 온다면 가공 프로그램에서 사용 할 수 있습니다 . 그러나 , 실행 시간 측정 ($AC_PROG_NET_TIME_TRIGGER) 을 위한 트리거가 빈번하게 쓰여지고 결과로 $AC_OLD_PROG_NET_TIME 값이 빈번하게 변경된다면 , 명시적 STOPRE 는 가공 프로그램에서 사용해야 합니다 .

일반 조건

- **블록 탐색**

  블록 탐색을 통해서는 프로그램 런타임이 확인되지 않습니다 .

- **REPOS**

  REPOS 프로세스 시간이 현재 가공 시간 ($AC_ACT_PROG_NET_TIME) 에 추가됩니다 .

예제

예제 1: "mySubProgrammA" 의 기간 측정

프로그램 코드

```
...
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
```

프로그램이 라인 N80 을 처리한 후 , "mySubProgrammA" 의 순수 실행 시간은 $AC_OLD_PROG_NET_TIME 에 위치합니다 .

$AC_OLD_PROG_NET_TIME 값 :

- M30 을 초과하여 유지합니다 .
- 루프가 실행될 때마다 업데이트됩니다 .

예제 2: "mySubProgrammA" 와 "mySubProgrammC" 의 지속 시간 측정

프로그램 코드

```
...
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 공작물 카운터

기능

' 소재 계수기 ' 기능을 사용하면 특히 제어기 내부에서 사용할 수 있는 다양한 계수기로 공작물을 계수할 수 있습니다 .

계수기는 0 - 999,999,999 값 범위에서 읽기 및 쓰기 액세스가 가능한 채널 관련 시스템 변수로 존재합니다 .

시스템 변수	의미
$AC_REQUIRED_PARTS	생산할 공작물 개수 (공작물 지령치 수) 이 계수기에서 실제 공작물 수 ($AC_ACTUAL_PARTS) 가 '0' 으로 리셋되는 공작물 개수를 정의할 수 있습니다 .
$AC_TOTAL_PARTS	가공된 공작물의 총 개수 (실제 총 공작물 수) 이 계수기는 시작 시점 이후 생산된 모든 공작물의 합계를 지정합니다 . 이 값은 기본값으로 제어기를 가동할 때마다 자동으로 '0' 으로 리셋됩니다 .
$AC_ACTUAL_PARTS	가공된 공작물 개수 (실제 총 공작물 수) 시작 시점 이후 생산된 모든 공작물의 총합이 계산됩니다 . $AC_REQUIRED_PARTS > 0 인 조건에서 필요한 공작물 수 ($AC_REQUIRED_PARTS) 에 도달하면 계수기는 자동으로 '0' 으로 리셋됩니다 .
$AC_SPECIAL_PARTS	사용자가 선택한 공작물 개수 이 계수기에서는 사용자별 공작물 개수를 설정할 수 있습니다. 공작물의 지령치 수에 도달하면 ($AC_REQUIRED_PARTS) 알람이 출력되도록 정의할 수 있습니다 . 사용자가 카운터를 직접 리셋해야 합니다 .

주

제어기가 기본값으로 시작되고 활성화와 무관하게 읽고 쓸 수 있는 경우에는 모든 계수기가 '0' 으로 설정됩니다 .

주

채널별 머신 데이터를 사용하면 계수기 활성화 , 계수기 리셋 시간 및 계수 알고리즘을 제어할 수 있습니다 .

주

사용자 정의 M 명령을 사용한 소재 계수

다양한 소재 계수기의 계수 펄스가 프로그램 끝이 아닌 사용자 정의된 M 명령 (M2/M30) 을 통해 실행되도록 머신 데이터를 설정할 수 있습니다 .

참고 자료

' 소재 계수기 ' 기능에 대한 자세한 내용은 다음을 참조하십시오 .

- 기능 매뉴얼 , 기본 기능 ; BAG, 채널 , 프로그램 작동 , 리셋 응답 (K1), 장 : 공작물 카운터

14.10 외부 장치 / 파일로 출력 (EXTOPEN, WRITE, EXTCLOSE)

기능

이 기능을 사용하면 가공 프로그램의 데이터를 외부 장치 / 외부 파일에 기록할 수 있습니다 . 예를 들어 , 생산 데이터를 기록하거나 제어 시스템에서 다른 장비를 제어할 수 있습니다 .

외부 장치 / 파일로의 출력은 3 단계로 이루어집니다 .

1. 외부 장치 / 파일 열기

   EXTOPEN 명령을 사용하여 기록할 채널에 해당하는 외부 장치 / 파일을 엽니다 .

2. 데이터 기록

   NC 언어 (" 문자열 연산 [쪽 75]") 의 문자열 기능 (예 : SPRINT) 을 사용하여 출력 데이터를 처리할 수 있습니다 . 기록에는 WRITE 명령이 사용됩니다 .

3. 외부 장치 / 파일 닫기

   채널에 할당된 외부 장치/파일은 프로그램 종료에 도달하거나 (M30) 채널 리셋이 필요할 때 EXTCLOSE 명령을 사용하여 다시 해제됩니다 .

주

가공 프로그램 / 채널에 외부 장치 / 파일을 하나 이상 할당할 수도 있습니다 .

사용 가능도

이 기능은 다음과 같이 사용 가능합니다 .

- 가공 프로그램에서만 (동기 동작에서는 **사용 불가**)
- 사용 가능한 (구성된) 모든 출력 장치에 대해 NCK 의 모든 가공 채널에서 동시에

각 출력 장치에서 장치를 열 때 단 하나의 채널만이 독점적으로 장치를 사용하도록 할지 여러 채널이 장치에 출력할 수 있도록 할지 (" 공유 " 모드) 지정할 수 있습니다 .

구문

```
DEF INT < 에러 >
DEF STRING[<n>] < 출력 >
…
EXTOPEN(< 에러 >,"<ExtG>",< 가공 모드 >,< 사용 모드 >,
< 기록 모드 >)
…
< 출력 >=" 출력 데이터 "
WRITE(< 에러 >,"<ExtG>",< 출력 >)
…
EXTCLOSE(< 에러 >,"<ExtG>")
```

중요도

EXTOPEN: 외부 장치 / 파일 열기 명령

< 에러 >: **파라미터 1:** 에러 값을 반환하기 위한 변수
에러 값을 사용하면 작업이 성공적이었는지 여부를 프로그램에서 평가한 다음 이에 따라 처리를 계속할 수 있습니다 .

유형 : INT

값 :

0	오류 없음
1	외부 장치를 열 수 없습니다 .
2	외부 장치가 구성되지 않았습니다 .
3	외부 장치의 경로가 올바르지 않습니다 .
4	외부 장치를 사용할 수 있는 권한이 없습니다
5	사용 모드 : 외부 장치가 이미 " 독점 " 사용되고 있습니다 .
6	사용 모드 : 외부 장치가 이미 " 공유 " 되고 있습니다 .
7	파일 길이가 LOCAL_DRIVE_MAX_FILESIZE 보다 깁니다 .
8	최대 외부 장치 수를 초과했습니다 .
9	LOCAL_DRIVE 옵션이 설정되지 않았습니다 .
11	V.24 인터페이스가 이미 Easy-Message 에 할당되었습니다 (828D 만 해당).
12	기록 모드 : 데이터가 extdev.ini 와 충돌합니다 .
16	잘못된 외부 경로가 프로그래밍되었습니다 .
22	외부 장치가 장착되지 않았습니다 .

<ExtG>: **파라미터 2:** 열고자 하는 외부 장치 / 파일의 심볼 이름

유형 : STRING

심볼 이름은 다음으로 구성됩니다 .

1. 논리적 디바이스 이름
2. 타당한 경우 이름 뒤에 파일 경로가 옵니다 ("/" 를 사용하여 추가).

다음 **논리적 디바이스 이름**이 정의되었습니다 .

"LOCAL_DRIVE":	로컬 콤팩트 플래시 카드 (사전 정의)
"CYC_DRIVE":	SIEMENS 싸이클에 사용하기 위한 예비용 드라이브 이름 (사전 정의)
"/dev/ext/1",... "/dev/ext/9":	사용 가능한 네트워크 드라이브 **참고 :** extdev.ini 파일에서 구성해야 합니다 !
"/dev/cyc/1", "/dev/cyc/2":	SIEMENS 싸이클에 사용하기 위한 예비용 드라이브 이름 **참고 :** extdev.ini 파일에서 구성해야 합니다 !
"/dev/v24":	V.24 인터페이스 **참고 :** extdev.ini 파일에서 구성해야 합니다 !

파일 경로 :

- "LOCAL_DRIVE" 및 "CYC_DRIVE" 에 대한 파일 경로를 지정해야 합니다 (예 :

  "LOCAL_DRIVE/my_dir/my_file.txt").

- 논리적 디바이스 이름 "/dev/ext/1...9" 및 "/dev/cyc/1...2" 를 다음과 같이 구성할 수 있습니다 .
  - 파일을 참조하기 위해 . 이 경우에는 논리적 디바이스 이름만 지정할 수 있습니다 (예 :

    "/dev/ext/4").
  - 또는 디렉토리에 대해 . 파일 경로를 지정해야 합니다 (예 :

    "/dev/ext/5/my_dir/my_file.txt").

- 파일 경로를 "/dev/v24" 에 추가하는 것은 허용되지 않습니다 .

참고 :
논리적 디바이스 이름 "/dev/ext/1...9", "/dev/v24" 및 "/dev/cyc/1...2" 에는 대소문자를 구분하지 않습니다 . 파일에 경로를 지정하는 경우에는 대소문자를 구분합니다 . "LOCAL_DRIVE" 및 "CYC_DRIVE" 에는 대문자만 허용됩니다 .

< 처리 모드 >: **파라미터 3:** 이 장치 / 파일에 대한 WRITE 명령의 처리 모드

유형 : STRING

값 :

"SYN": 동시 쓰기
쓰기 작업이 완료될 때까지 프로그램 실행이 중단됩니다 .
동시 쓰기 작업이 성공적으로 완료되었는지 여부는 WRITE 명령의 에러 변수를 평가하여 확인할 수 있습니다 .

"ASYN": 비동시 쓰기
프로그램 실행이 WRITE 명령에 의해 중단되지 않습니다 .
참고 :
이 모드에서는 WRITE 명령의 에러 변수가 아무 정보도 제공하지 않으며 항상 값이 0 (에러 없음) 입니다 . 이 모드에서는 WRITE 명령이 성공했는지 여부를 확인할 수 없습니다 .

< 사용 모드 >: **파라미터 4:** 이 장치 / 파일의 사용 모드

유형 : STRING

값 :

"SHARED": 장치 / 파일이 " 공유 " 모드에서 요청됩니다 . 다른 채널도 장치를 사용할 수 있습니다 . 즉 , 다른 채널도 이 모드에서 개방할 수 있습니다 .

"EXCL": 장치 / 파일이 채널에서 배타적으로 사용되며 , 다른 채널은 장치를 사용할 수 없습니다 .

< 쓰기 모드 >: **파라미터 5:** 이 장치 / 파일에 대한 WRITE 명령의 쓰기 모드 (옵션)

유형 : STRING

값 : "APP": 첨부
파일은 항상 그 내용을 유지합니다 . 따라서 쓰기 호출은 끝 부분에 첨부됩니다 .

"OVR": 덮어쓰기
파일 내용이 삭제되고 이어지는 쓰기 호출을 사용하여 다시 생성됩니다 .

참고 :
이 파라미터를 사용하면 extdev.ini 파일에 설정된 쓰기 모드를 덮어쓸 수 **없습니다** . 충돌이 있는 경우에는 EXTOPEN 호출이 승인되면서 에러가 발생합니다 .

WRITE: 출력 데이터 쓰기 명령
자세한 내용은 " 파일 쓰기 (WRITE) [쪽 138]" 를 참조하십시오 .

EXTCLOSE : 열려 있는 외부 장치 / 파일 닫기 명령

< 에러 >: **파라미터 1:** 에러 값을 반환하기 위한 변수

유형 : INT

값 : 0 오류 없음
16 잘못된 외부 경로가 프로그래밍되었습니다 .
21 외부 장치를 닫을 때 에러 발생

<ExtG>: **파라미터 2:** 닫고자 하는 외부 장치 / 파일의 심볼 이름
자세한 내용은 EXTOPEN 을 참조하십시오 !

참고 :
식별자는 EXTOPEN 명령에서 지정한 식별자와 동일해야 합니다 .

예

프로그램 코드

```
N10     DEF INT RESULT
N20     DEF BOOL EXTDEVICE
N30     DEF STRING[80] OUTPUT
N40     DEF INT PHASE
N50     EXTOPEN(RESULT,"LOCAL_DRIVE/my_file.txt","SYN","SHARED")
N60     IF RESULT > 0
N70         MSG("error for EXTOPEN:" << RESULT)
N80     ELSE
N90         EXTDEVICE=TRUE
N100    ENDIF
…
N200    PHASE=4
N210    IF EXTDEVICE
N220        OUTPUT=SPRINT("end of phase: %D",PHASE)
N230        WRITE(RESULT,"LOCAL_DRIVE/my_file.txt",OUTPUT)
N240    ENDIF
…
```

추가 정보

연속 경로 모드에 미치는 영향

EXTOPEN, WRITE 및 EXTCLOSE 명령은 전처리 정지를 실행하므로 연속 경로 모드를 중단시킵니다 .

블록 탐색 동작

WRITE 를 사용하여 " 계산을 포함한 블록 탐색 " 을 수행하는 동안에는 출력은 이루어지지 않습니다 . 그러나 , 탐색 대상에 도달한 후 NC 시작과 함께 EXTOPEN 및 EXTCLOSE 명령이 수집되고 활성화됩니다 . 따라서 다음 WRITE 명령은 일반 프로그램 처리와 동일한 환경에서 실행됩니다 .

" 프로그램 테스트 " 모드 (SERUPRO) 에서 계산을 포함하여 탐색하는 경우 일반 프로그램 처리와 동일하게 EXTOPEN, WRITE 및 EXTCLOSE 명령이 실행됩니다 .

리셋 동작

가공 프로그램 종료 및 채널 리셋과 함께 이 채널에서 열려 있던 외부 장치 / 파일이 모두 종료됩니다 .

사용 가능한 외부 장치

다음을 외부 장치 / 파일로 사용할 수 있습니다 .

- 로컬 콤팩트 플래시 카드의 파일

  로컬 콤팩트 플래시 카드는 심볼 이름 LOCAL_DRIVE 를 사용하여 HMI 에서 참조되는 메모리를 의미합니다 . SINUMERIK 840D sl 의 경우에는 로컬 드라이브 , SINUMERIK 828D 에서는 사용자 콤팩트 플래시 카드입니다 .

  주

  SINUMERIK 840D sl 의 경우에는 LOCAL_DRIVE 장치로 출력하려면 "NCU 의 CF 카드에 있는 추가 xxx MB HMI 사용자 메모리" 옵션이 필요합니다. SINUMERIK 828D의 경우에는 사용자 콤팩트 플래시 카드를 사용할 수 있어야 하며 여기에는 옵션이 필요하지 않습니다 .

- 네트워크 드라이브의 파일
- V.24 인터페이스

  주

  SINUMERIK 840D sl 의 경우 V.24 인터페이스에서 출력하려면 NCU 옵션 모듈 RS232 인터페이스가 필요합니다 . SINUMERIK 828D 의 경우에는 통합 V.24 인터페이스에서 출력이 실행됩니다 (전제 조건 : MD51233 $MNS_ENABLE_GSM_MODEM = 0).

설정

사용할 외부 장치는 /oem/sinumerik/nck/extdev.ini 또는 /user/sinumerik/nck/extdev.ini 파일에서 설정됩니다 . 두 파일 모두 사용 가능한 경우에는 사용자 영역에 있는 항목이 우선 순위를 가집니다 . SYSTEMDATEN/CF 카드의 COMMISSIONING 작동 영역에서 파일을 업데이트할 수 있습니다 .

주

LOCAL_DRIVE 및 CYC_DRIVE 를 사용할 때는 extdev.ini 파일에서 설정할 필요가 없습니다 . 해당 옵션이 설정되거나 사용자 콤팩트 플래시 카드를 사용할 수 있으면 두 장치는 항상 사용 가능합니다 .

사용할 외부 장치는 extdev.ini 파일의 [ExternalDevices] 섹션에 정의/열거되어 있습니다. 직렬 장치 (/dev/v24) 및 최대 9 개의 파일 또는 디렉토리 (/dev/ext/1…9) 를 장치로 지정할 수 있습니다 . 장치를 지정할 때는 Linux 표기법이 사용됩니다 . ";" 로 시작되는 라인은 설명이며 무시됩니다 .

/dev/v24를 제외한 장치들을 "/"로 끝나는 디렉토리 경로 또는 "/"로 끝나지 않고 파일 이름으로 끝나는 파일 경로 (즉 , 전체 경로) 로 선언할 수 있습니다 . 가공 프로그램에서 사용하는 경우에는 디렉토리 경로를 포함한 파일 이름 (경로) 도 장치에 지정해야 합니다 .

"Server", "Path" 및 옵션인 "Write mode" 의 경우 /dev/v24 를 제외한 장치는 쉼표로 분리되는 데이터 3 개를 사용하여 정의됩니다 .

파일 또는 디렉토리 (디렉토리에 포함된 모든 파일에 적용) 에 대해 파일을 연 후 해당 파일을 덮어쓸 것인지 ("O" = Overwrite) 또는 출력을 파일에 첨부할 것인지 ("A" = Append) 여부를 지정할 수 있습니다 . 기본값은 "A" 입니다 . 존재하지 않는 파일 / 디렉토리는 열 때 새로 생성됩니다 .

장치 V.24 인터페이스의 경우 전송 속도 , 데이터 비트 , 정지 비트 , 패리티 , 프로토콜 및 종료 설정만 이 순서로 지정됩니다 .

LOCAL_DRIVE에서 생성/저장된 파일의 경우에는 LOCAL_DRIVE_MAX_FILESIZE 데이터를 사용하여 최대 파일 크기를 바이트 단위로 설정할 수 있습니다 . 그러면 이 설정이 모든 파일에 대한 표준 설정이 됩니다. 첨부 모드에서 EXTOPEN 명령을 실행하면 파일 크기가 확인됩니다. 옵션으로 LOCAL_DRIVE_FILE_MODE 데이터를 사용하여 쓰기 모드 ("O" = Overwrite, "A" = Append) 를 정의할 수 있습니다 . 기본값은 "A" 입니다 .

주

extdev.ini 설정 파일에 대한 복사 템플릿은 /siemens/sinumerik/nck 디렉토리에 있습니다 .

주

extdev.ini 파일에 대한 변경 사항은 NCK 를 재시작 / 부팅한 후에 적용됩니다 .

주

USB 장치

SINUMERIK 828D 의 경우 "usb" (파티션 데이터는 포함하지 않음 !) 를 전면에 삽입한 USB 장치에 대한 대상으로 정의할 수도 있습니다 . USB 장치는 심볼 장치 이름 "/dev/ext/x" 를 사용하여 가공 프로그램에서 직접 주소를 지정할 수 있습니다 .

SINUMERIK 840D sl 의 경우에는 TCU 의 정적으로 연결된 USB 인터페이스만 USB 장치로 구성할 수 있습니다 . 위의 경우 , SERVER:/PATH 유형을 "Server" 에 대한 지정으로 사용하여 구성됩니다 . 여기서 SERVER 는 TCU 이름이며 , /PATH 는 USB 인터페이스의 경로입니다 . TCU의 USB 인터페이스 주소는 "dev0-0", "dev0-1", "dev1-0"으로 지정됩니다. 경로 데이터는 항상 "/Partition" 으로 시작합니다 . 이 때 , 2 자리수 파티션 번호 또는 파티션 이름을 사용하여 파티션을 지정할 수 있으며 , 필요한 경우 다음과 같이 필요한 대상까지 포함하는 파일 경로로 지정할 수 있습니다 .

/dev/ext/8 = "TCU4:/dev0-0, /01/, A"

/dev/ext/8 = "TCU4:/dev0-0, /01/mydir.dir/"

/dev/ext/8 = "TCU4:/dev0-0, /myfirstpartition/Mydir.dir/myfile.txt, O"

예 :

```
[ExternalDevices]
; 코멘트 라인
; V24 의 예
; /dev/v24 = "9600, 8, 1, none, rts [, etx]"
; 네트워크 드라이브 예제
; /dev/ext/1 = "//[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE/, /, A"
; /dev/ext/2 = "//[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE, /myfile.txt, O"
; /dev/ext/3 = "//[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE, /mydir/, A"
; /dev/ext/4 = "SERVER:/dev0-0, /01/, A"
; …
; SINUMERIK 828 만 해당 (USB)
; /dev/ext/9 = "usb, / [ , O]"
; 기본값 : 파티션 번호 = 1

; SIEMENS 만 해당
; /dev/cyc/1= "//[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE, /mydir/, A"
; /dev/cyc/2= "//[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE/mydir, /, A"

LOCAL_DRIVE_MAX_FILESIZE = 50000
LOCAL_DRIVE_FILE_MODE = "O"
```

EXTOPEN 파라미터 < 쓰기 모드 > 의 효율성

쓰기 모드를 지정하면 extdev.ini 파일에서 구성할 때 및 EXTOPEN 호출 시 인증 충돌이 발생할 수 있습니다 . 이 충돌은 EXTOPEN 에서 승인되고 다음 에러가 발생될 수 있습니다 .

extdev.ini 의 값	EXTOPEN 파라미터의 값		
	"OVR"	"APP"	-
"O"	O	에러	O
"A"	에러	A	A
-	O	A	A
	설명 : O: " 덮어쓰기 " 모드가 활성입니다 . A: " 첨부 " 모드가 활성입니다 . 에러 : EXTOPEN 호출이 승인되고 오류가 발생합니다 .		

LOCAL_DRIVE: 파일 속성

LOCAL-DRIVE 에서 EXTOPEN 으로 생성된 파일에는 다음과 같은 파일 속성이 할당됩니다 .

- 소유자 : "user" 읽기 / 쓰기 권한 세트
- 그룹 : "operator" 읽기 / 쓰기 권한 세트

열린 외부 장치의 최대 개수

모든 NC 채널에서 출력 장치를 최대 10개까지 동시에 열 수 있습니다. 또한 Siemens 싸이클에 사용할 수 있는 두 가지 항목이 있습니다 .

현재 이들 장치에서 최대 5 개 작업을 동시에 활성화할 수 있습니다 .

14.11 알람 (SETAL)

기능

알람을 NC 프로그램 내에 설치 할 수 있습니다 . 알람은 조작자 인터페이스의 별도 필드에 표시됩니다 . 알람은 알람 카테고리에 의해 컨트롤러의 응답에 협력하며 움직입니다 .

참조 :
알람 응답에 관한 자세한 정보는 테스트 설명서를 참조하세요 .

구문

SETAL(< 알람 번호 >[,< 문자열 >])

중요도

SETAL:	알람 프로그래밍 키워드 . SETAL 은 개별 NC 블록으로 프로그래밍되어야 합니다 .
< 알람 이름 >:	정수형 (INT) 변수 알람 번호 포함 . 알람 번호의 유효 범위는 60000 에서 69999 이며 , 이 중 60000 에서 64999 는 SIEMENS 주기를 위해 보관되며 사용자는 65000 에서 69999 까지 사용 할 수 있습니다 .
< 문자열 >:	사용자 싸이클 알람을 프로그램할 때 파라미터를 최대 4 개까지 포함하는 문자열을 지정할 수 있습니다 . 변수 사용자 텍스트는 이런 파라미터에 정의할 수 있습니다 . 사용할 수 있는 사전 정의 파라미터는 다음과 같습니다 .

파라미터	중요도
%1	채널 번호
%2	블록 번호 , 레이블
%3	싸이클 알람용 텍스트 색인
%4	추가 알람 파라미터

주

알람 텍스트는 조작자 인터페이스에서 구성해야 합니다 .

주

알람을 사용자 인터페이스에 사용되는 언어로 출력하려면 현재 HMI 에 설정된 언어에 대한 정보가 필요합니다 . 이 정보는 가공 프로그램 및 동기 동작에서 시스템 변수 $AN_LANGUAGE_ON_HMI를 사용하여 질의할 수 있습니다 ("현재 HMI에서 설정된 언어 [쪽 882]" 참조).

예

프로그램 코드	코멘트
...	
N100 SETAL (65000)	; 알람 번호 65000 을 설정
...	

14.12 드라이브 통합 확장 정지 및 후퇴 (ESR)

14.12.1 드라이브 통합 정지 (ESRS) 설정

기능

드라이브 통합 ESR 기능의 " 정지 " 를 위한 드라이브 파라미터는 ESRS(...) 기능을 사용하여 설정합니다 .

구문

ESRS(<access_1>,<stopping time_1>[,...,<axis_n>,<stopping time_n>])

중요도

ESRS(...):	ESR 기능 " 정지 " 를 위한 드라이브 파라미터에 쓰기 기능 . 기능 : • 블록에 단독으로 있어야 합니다 . • 전처리 정지를 실행합니다 . • 동기 동작에 사용할 수 없습니다 .
<axis_1>, ..., <axis_n>:	드라이브 통합 정지를 설정할 축 . 이 축의 경우에는 드라이브 파라미터 p0888 (설정) 이 드라이브에 기록됩니다 . p0888 = 1 유형 : AXIS 값 범위 : 채널 축 이름
<stopping time_1>, ..., <stopping time_n>:	에러가 발생한 후 드라이브가 실제 속도 지령치로 이송되는 시간 . 지정된 축의 경우에는 드라이브 파라미터 p0892 (타이머) 가 드라이브에 기록됩니다 . p0892 = < 정지 시간 > 단위 : s 유형 : REAL 값 범위 : 0.00 - 20.00

기능 호출에 최대 5 개 축을 프로그래밍할 수 있습니다 (n = 5).

참고 자료

기능 매뉴얼 , 특수 기능 , 확장 정지 및 후퇴 (R3), 드라이브 통합 ESR

14.12.2 드라이브 통합 후퇴 (ESRS) 설정

기능

드라이브 통합 ESR 기능의 " 후퇴 " 를 위한 드라이브 파라미터는 ESRR(...) 기능을 사용하여 설정합니다 .

구문

ESRR(<axis_1>,<retraction distance_1>,<retraction velocity_1>[,...,<axis_n>,<retraction distance_n>,<retraction velocity_n>])

중요도

ESRR(...):	ESR 기능 " 후퇴 " 를 위한 드라이브 파라미터에 쓰기 기능 . 기능 : • 블록에 단독으로 있어야 합니다 . • 전처리 정지를 실행합니다 . • 동기 동작에 사용할 수 없습니다 .
<axis_1>, ..., <axis_n>:	드라이브 통합 후퇴를 설정할 축 . 이 축의 경우에는 드라이브 파라미터 p0888 (설정) 이 드라이브에 기록됩니다 . p0888 = 2 유형 : AXIS 값 범위 : 채널 축 이름
<retraction distance_1>, ..., <retraction distance_n>:	드라이브의 경우 후퇴 거리는 후퇴 속도로 변환됩니다 . 지정된 축의 경우에는 드라이브 파라미터 p0893 (속도) 에 값이 기록됩니다 . p0893 = (<retraction distance _n> converted into retraction speed) 단위 : mm/min, inch/min, degrees/min (축의 단위에 따라) 유형 : REAL 값 범위 : MIN - MAX
<retraction velocity_1>, ..., <retraction velocity_n>:	드라이브의 경우 후퇴 속도는 시간으로 변환됩니다 . 지정된 축의 경우에는 드라이브 파라미터 p0892 (타이머) [s] 에 값이 기록됩니다 . p0892 = <retraction distance_n> / <retraction velocity _n> 단위 : mm/min, inch/min, degrees/min (축의 단위에 따라) 유형 : REAL 값 범위 : 0.00 - MAX

기능 호출에 최대 5 개 축을 프로그래밍할 수 있습니다 (n = 5).

참고 자료

기능 매뉴얼 , 특수 기능 , 확장 정지 및 후퇴 (R3), 드라이브 통합 ESR

사용자 선삭 프로그램 15

15.1 스톡 제거 기능 지원

기능

사전 설정된 스톡 제거 프로그램이 제공됩니다 . 또한 다음과 같은 기능을 이용해 특정한 스톡 제거 프로그램을 생성할 수 있습니다 .

- 형상 테이블 생성 (CONTPRON)
- 코드화 형상 테이블의 생성 (CONTDCON)
- 형상 준비 종료 (EXECUTE)
- 형상 요소 2 개 사이의 교차점을 결정 (INTERSEC)

  (CONTPRON 을 이용해 생성한 테이블에서만)
- 테이블의 형상 요소를 블록별로 실행 (EXECTAB)

  (CONTPRON 을 이용해 생성한 테이블에서만)
- 원 데이터 계산 (CALCDAT)

주

이런 기능은 스톡 제거 이외의 다른 목적에도 사용할 수 있습니다 .

필요 조건

CONTPRON 또는 CONTDCON 기능을 호출하기 전에 완료해야 하는 작업은 다음과 같습니다 .

- 충돌 방지 가공을 허용하는 시작점에 접근해야 합니다 .
- 절삭 반경 보정은 `G40` 으로 종료해야 합니다 .

15.2 형상 테이블 생성 (CONTPRON)

기능

형상 준비는 CONTPRON 명령을 이용해 활성화시킵니다 . 후속으로 호출되는 NC 블록은 실행되지 않지만 개별 이동으로 나뉘고 형상 테이블에 저장됩니다 . 각 형상 요소는 형상 테이블의 2 차원 배열에서 한 행에 해당합니다 . 릴리프 절삭 번호가 반환됩니다 .

구문

형상 준비 활성화 :

```
CONTPRON(<contour table>,<machining type>,<relief cuts>,
<machining direction>)
```

형상 준비 종료 및 정상 실행 모드로 복귀 :

```
EXECUTE(<ERROR>)
```

" 형상 준비 종료 (EXECUTE) " 를 참조합니다 .

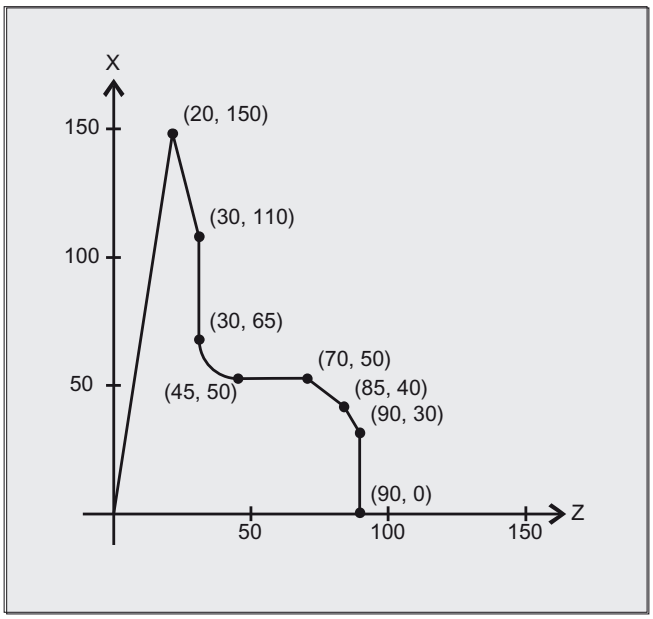
중요도

CONTPRON	형상 테이블을 생성하기 위해 형상 준비를 활성화시키는 명령		
<contour table>	형상 테이블의 이름		
<machining type>	가공 형식의 파라미터		
	형식 :	CHAR	
	값 :	"G"	세로 방향 선삭 : 내부 가공
		"L"	세로 방향 선삭 : 외부 가공
		"N"	단면 선삭 가공 내부 가공
		"P"	단면 선삭 가공 외부 가공
<relief cuts>	생성할 릴리프 절삭 요소의 결과 변수		
	형식 :	INT	
<machining direction>	가공 방향의 파라미터		
	형식 :	INT	
	값 :	0	형상 준비 , 정방향 (기본값)
		1	양방향으로 형상 준비

예제 1

형상 테이블 생성 시 :

- 이름 : "KTAB"
- 최대 30 개의 형상 요소 (원 , 직선)
- 생성할 릴리프 절삭 요소의 변수 1 개
- 고장 메시지의 변수 1 개



NC 프로그램 :

프로그램 코드	비고
N10 DEF REAL KTAB[30.11]	; 이름이 KTAB 이며 최대 30 개의 형상 요소를 포함한 형상 테이블 , 파라미터 값 11(테이블 열의 수) 은 고정
N20 DEF INT ANZHINT	; 이름이 ANZHINT 인 릴리프 절삭 요소 번호의 변수
N30 DEF INT ERROR	; 고장 피드백 신호의 변수 (0= 정상 , 1= 고장)
N40 G18	
N50 CONTPRON(KTAB,"G",ANZHINT)	; 형상 준비 활성화
N60 G1 X150 Z20 N70 X110 Z30 N80 X50 RND=15 N90 Z70 N100 X40 Z85 N110 X30 Z90 N120 X0	; N60 에서 N120: 형상 설명
N130 EXECUTE(ERROR)	; 형상 테이블에 입력 종료 , 정상 프로그램 실행으로 전환
N140 …	; 테이블 처리 계속

형상 테이블 KTAB:

색인 행	열									
(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

열 내용에 대한 설명 :

(0) 다음 형상 요소의 포인터 (해당 열에서 행 번호로)

(1) 이전 형상 요소의 포인터

(2) 이동을 위한 형상 모드 코딩

가능한 값은 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) 형상 요소의 시작점
현재 면의 (3) = 가로축 , (4) = 세로축

(5), (6) 형상 요소의 시작점
현재 면의 (5) = 가로축 , (6) = 세로축

(7) 최대 / 최소 표시기 : 형상의 국소 최대와 최대 값을 표시

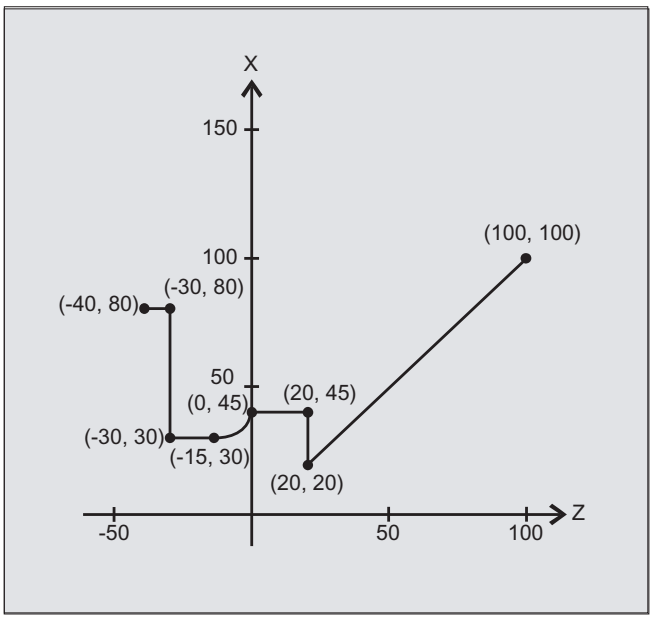
(8) 형상 요소와 가로축 (세로 방향 가공) 이나 세로축 (전면 절삭) 사이의 최대값 .
각도는 프로그램한 가공 형식에 따라 달라짐 .

(9), (10) 원 블록인 경우에 형상 요소의 중심점 좌표
(9) = 가로축 , (10) = 세로축

예제 2

형상 테이블 생성 시 :

- 이름 : KTAB
- 최대 92 개의 형상 요소 (원 , 직선)
- 모드 : 세로 선삭 , 외부 가공
- 준비 , 정방향과 역방향



NC 프로그램 :

프로그램 코드	비고
N10 DEF REAL KTAB[92.11]	; 이름이 KTAB 이며 최대 92 개의 형상 요소를 포함한 형상 테이블 , 파라미터 값 11 은 고정
N20 DEF CHAR BT="L"	; CONTPRON 모드 : 세로 선삭 , 외부 가공
N30 DEF INT HE=0	; 릴리프 절삭 요소의 개수 =0
N40 DEF INT MODE=1	; 준비 , 정방향과 역방향
N50 DEF INT ERR=0	; 고장 피드백 신호
...	
N100 G18 X100 Z100 F1000	
N105 CONTPRON(KTAB,BT,HE,MODE)	; 형상 준비 활성화
N110 G1 G90 Z20 X20 N120 X45 N130 Z0 N140 G2 Z-15 X30 K=AC(-15) I=AC(45) N150 G1 Z-30 N160 X80 N170 Z-40	
N180 EXECUTE(ERR)	; 형상 테이블에 입력 종료 , 정상 프로그램 실행으로 전환
...	

형상 테이블 KTAB:

형상 준비를 마친 후 형상은 양방향으로 이용할 수 있습니다 .

색인	열										
라인	(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

0, 1, 6, 8, 83, 85, 91 행에서 열의 내용과 비고 설명

예제 1 에 제시한 열의 내용 설명을 적용합니다 .

테이블 행은 항상 0:

1) 선행 : n 행은 형상 끝을 포함 (정방향)

2) 후속 : n 행은 형상 테이블 끝을 포함 (정방향)

형상 요소 정방향에서 :

3) 선행 : 형상 시작 (정방향)

4) 후속 : 형상 끝 (정방향)

항상 행 형상 테이블 끝 (정방향) +1 에서 :

5) 선행 : 릴리프 절삭 수 (정방향)

6) 후속 : 릴리프 절삭 수 (역방향)

형상 요소 역방향에서 :

7) 후속 : 형상 끝 (역방향)

8) 선행 : 형상 시작 (역방향)

항상 테이블의 마지막 행에서 :

9) 선행 : n 행은 형상 테이블 시작을 포함 (역방향)

10) 후속 : n 행은 형상 시작을 포함 (역방향)

그 밖의 정보

사용할 수 있는 이송 명령 (좌표계)

형상 프로그램에 사용할 수 있는 G 명령어 :

- G 그룹 1: G0, G1, G2, G3

또한 다음 작업이 가능합니다 .

- 라운딩과 모따기
- CIP 및 CT 를 이용한 원 프로그래밍

스플라인 , 폴리노미널 , 나사 기능에서는 오류가 발생합니다 .

프레임 활성화를 통한 좌표계 변경은 CONTPRON 와 EXECUTE 사이에서 허용되지 않습니다 . 또한 G70 와 G71 또는 G700 와 G710 사이에서도 좌표계 변경이 불가능합니다 .

형상 테이블을 준비하는 동안 기하축을 GEOAX 로 변경하면 알람이 발생합니다 .

릴리프 절삭 요소

개별 릴리프 절삭 요소의 형상 설명은 서브프로그램이나 개별 블록에서 수행할 수 있습니다 .

프로그램한 형상 방향과 독립적인 스톡 제거

CONTPRON 을 이용한 형상 준비는 형상 테이블을 호출해 프로그램한 방향에 상관없이 이용할 수 있도록 확장되었습니다 .

15.3 코드화 형상 테이블의 생성 (CONTDCON)

작동

CONTDCON 으로 형상 준비를 시작하면 다음과 같은 호출한 NC 블록들은 메모리 사용을 최적화하기 위해 6 열 형상 테이블에 코드화 형태로 저장됩니다 . 각 형상 요소는 형상 테이블에서 한 행에 해당합니다 . 아래에 정의한 코딩 규칙에 익숙해지면 테이블 행에서 싸이클용 DIN 코드 프로그램을 조합할 수 있습니다 . 출력 지점의 데이터는 테이블 행에 0 번으로 저장됩니다 .

구문

형상 준비 활성화 :
CONTDCON(<contour table>,<machining direction>)

형상 준비 종료 및 정상 실행 모드로 복귀 :
EXECUTE(<ERROR>)

" 형상 준비 종료 (EXECUTE) " 를 참조합니다 .

의미

CONTDCON	코드화 형상 테이블을 생성하기 위해 형상 준비를 활성화시키는 명령		
<contour table>	형상 테이블의 이름		
<machining direction>	가공 방향의 파라미터		
	형식 :	INT	
	값 :	0	형상 블록의 순서에 따라 형상 준비 (기본값)
		1	허용되지 않음

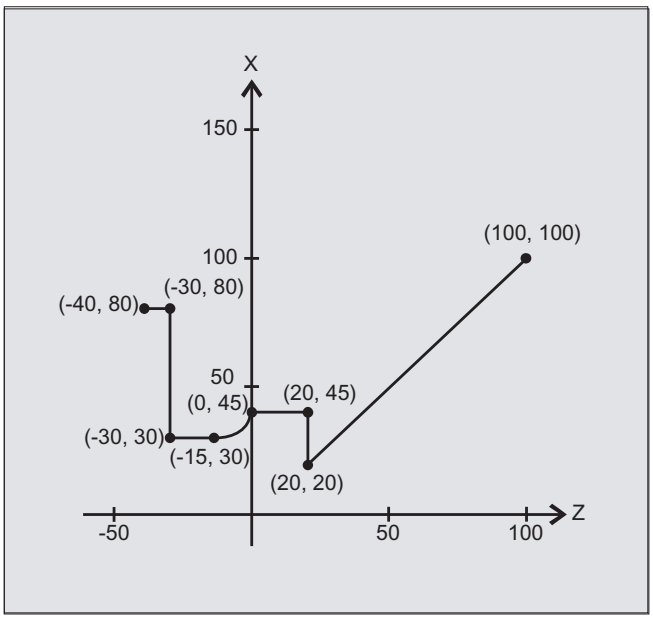
주

테이블에 포함될 프로그램 섹션에서 CONTDCON 에 사용할 수 있는 G 코드는 CONTPRON 보다 많습니다 . 또한 피드 속도 및 피드 속도 유형은 각 형상 섹션에 저장됩니다 .

예

형상 테이블 생성 시 :

- 이름 : "KTAB"
- 형상 요소 (원 , 직선)
- 모드 : 선삭
- 가공 방향 : Upwards



NC 프로그램 :

프로그램 코드	비고
N10 DEF REAL KTAB[9,6]	; 이름이 KTAB 이고 테이블 셀이 9 개인 형상 테이블 . 8 개의 형상 세트를 사용 가능 . 파라미터 값 6(테이블의 열 개수) 은 고정된 크기 .
N20 DEF INT MODE = 0	; 가공 방향 변수 . 표준값 0: 프로그램한 형상 방향에서만 .
N30 DEF INT ERROR = 0	; 고장 피드백 신호의 변수
...	
N100 G18 G64 G90 G94 G710	
N10 G1 X100 F1000	
N105 CONTDCON (KTAB, MODE)	; 형상 준비를 호출 (MODE 는 생략 가능)
N110 G1 Z20 X20 F200 N120 G9 X45 F300 N130 Z0 F400	; 형상 설명
N140 G2 Z-15 X30 K=AC(-15) I=AC(45)F100	
N10 G64 Z-30 F600 N160 X80 F700 N170 Z-40 F800	

프로그램 코드	비고
N180 EXECUTE(ERROR)	; 형상 테이블에 입력 종료 , 정상 프로그램 실행으로 전환
...	

형상 테이블 KTAB:

	열 색인					
	0	1	2	3	4	5
행 색인	형상 모드	끝점 가로축	끝점 세로축	중심점 가로축	중심점 세로축	피드 속도
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

열 내용에 대한 설명 :

시작점에서 0 행 코딩 :

0 열 : 10^0 (단위 숫자): G0 = 0
10^1 (10 자리 숫자): G70 = 0, G71 = 1, G700 = 2, G710 = 3
1 열 : 가로축의 시작점
2 열 : 세로축의 시작점
3-4 열 : 0
5 열 : 테이블에서 마지막 형상 단편의 행 색인

1-n 행 : **형상 단편**의 엔트리
0 열 : 10^0 (단위 숫자): G0 = 0, G1 = 1, G2 = 2, G3 = 3
10^1 (10 자리 숫자): G70 = 0, G71 = 1, G700 = 2, G710 = 3
10^2 (100 자리 숫자): G90 = 0, G91 = 1
10^3 (1000 자리 숫자): G93 = 0, G94 = 1, G95 = 2, G96 = 3
10^4 (10000 자리 숫자): G60 = 0, G44 = 1, G641 = 2, G642 = 3
10^5 (100000 자리 숫자): G9 = 1
1 열 : 끝점 가로축
2 열 : 끝점 세로축

3 열 : 원 보간의 중심점 가로축
4 열 : 원 보간의 중심점 세로축
5 열 : 피드 속도

그 밖의 정보

사용할 수 있는 이송 명령 (좌표계)

형상 프로그램에 사용할 수 있는 G 그룹과 G 명령어 :

G 그룹 1:	G0: G1, G2, G3
G 그룹 10:	G60, G64, G641, G642
G 그룹 11:	G9
G 그룹 13:	G70, G71, G700, G710
G 그룹 14:	G90, G91
G 그룹 15:	G93, G94, G95, G96, G961

또한 다음 작업이 가능합니다 .

- 라운딩과 모따기
- CIP 및 CT 를 이용한 원 프로그래밍

스플라인 , 폴리노미널 , 나사 기능에서는 오류가 발생합니다 .

프레임 활성화를 통한 좌표계 변경은 CONTDCON 와 EXECUTE 사이에서 허용되지 않습니다 . 또한 G70 와 G71 또는 G700 와 G710 사이에서도 좌표계 변경이 불가능합니다 .

형상 테이블을 준비하는 동안 기하축을 GEOAX 로 변경하면 알람이 발생합니다 .

가공 방향

CONTDCON 을 이용해 생성한 형상 테이블은 프로그램한 형상 방향에서 스톡 제거에 사용합니다 .

15.4 형상 요소 2 개 사이의 교차점을 결정 (INTERSEC)

작동

INTERSEC 는 CONTPRON 을 이용해 생성한 형상 테이블에서 법선 형상 요소 2 개의 교차점을 결정합니다 .

구문

```
<Status>=INTERSEC(<contour table_1>[<contour element_1>],
<contour table_2>[<contour element_2>],<intersection
point>,<machining type>)
```

의미

INTERSEC	CONTPRON 을 이용해 생성한 형상 테이블에서 형상 요소 2 개 사이의 교차점을 결정하는 키워드		
<status>	교차점 상태의 변수		
	형식 :	BOOL	
	값 :	TRUE	교차점 발견
		FALSE	교차 없음
<contour table_1>	첫 번째 형상 테이블의 이름		
<contour element_1>	첫 번째 형상 테이블에서 형상 요소의 이름		
<contour table_2>	두 번째 형상 테이블의 이름		
<contour element_2>	두 번째 형상 테이블에서 형상 요소의 이름		
<point of intersection>	활성 면에서 교차 좌표 (G17 / G18 / G19)		
	형식 :	REAL	
<machining type>	가공 형식의 파라미터		
	형식 :	INT	
	값 :	0	파라미터 2 (표준값) 로 활성 면에서 교차점 계산
		1	활성 면과 상관없이 교차점 계산

주

변수는 명령을 사용하기 전에 정의해야 합니다 .

형상을 전송할 때는 CONTPRON 으로 정의한 값을 준수해야 합니다 .

파라미터	의미
2	이동을 위한 형상 모드 코딩
3	형상 시작점 가로축
4	형상 시작점 세로축
5	형상 끝점 가로축
6	형상 끝점 세로축
9	가로축의 중심점 좌표 (원 형상에서만)
10	세로축의 중심점 좌표 (원 형상에서만)

예

TABNAME1 테이블에서 형상 요소 3 과 TABNAME2 테이블에서 형상 요소 7 의 교차를 계산합니다. 활성 면에서 교차 좌표는 변수 ISCOORD(첫 번째 요소는 가로축, 두 번째 요소는 세로축) 으로 저장됩니다 . 교차가 없으면 프로그램은 NOCUT(교차 없음) 으로 이동합니다 .

프로그램 코드	비고
DEF REAL TABNAME1[12,11]	; 형상 테이블 1
DEF REAL TABNAME2[10,11]	; 형상 테이블 2
DEF REAL ISCOORD [2]	; 교차점 좌표의 변수
DEF BOOL ISPOINT	; 교차점 상태의 변수
DEF INT MODE	; 가공 형식의 변수
…	
MODE=1	; 활성 면과 상관없이 계산
N10 ISPOINT=INTERSEC(TABNAME1[3],TABNAME2[7],ISCOORD,MODE)	; 형상 요소의 교차점 호출
N20 IF ISPOINT==FALSE GOTOF NOCUT	; NOCUT 으로 이동
…	

15.5 테이블의 형상 요소를 블록별로 실행 (EXECTAB)

작동

EXECTAB 명령을 사용하면 CONTPRON 명령으로 생성한 테이블의 형상 요소를 블록별로 실행할 수 있습니다 .

구문

```
EXECTAB(<contour table>[<contour element>])
```

의미

EXECTAB	형상 요소를 실행하는 명령
<contour table>	형상 테이블의 이름
<contour element>	형상 요소의 이름

예

KTAB 테이블의 0 에서 2 까지형상 요소가 블록별로 실행됩니다 .

프로그램 코드	비고
N10 EXECTAB(KTAB[0])	; KTAB 테이블의 형상 요소 0 을 이송
N20 EXECTAB(KTAB[1])	; KTAB 테이블의 형상 요소 1 을 이송
N30 EXECTAB(KTAB[2])	; KTAB 테이블의 형상 요소 2 를 이송

15.6 원 데이터 계산 (CALCDAT)

작동

CALCDAT 명령을 사용하면 원의 3 개나 4 개 지점에서 원의 반경과 중심점 좌표를 계산할 수 있습니다 . 지정하는 지점들은 서로 달라야 합니다 . 4 개 지점이 원에 직접 놓이지 않으면 원 중심점과 반경을 평균값으로 계산합니다 .

구문

```
<Status>=CALCDAT(<circle points>[<number>,<type>],<number>,<result>)
```

의미

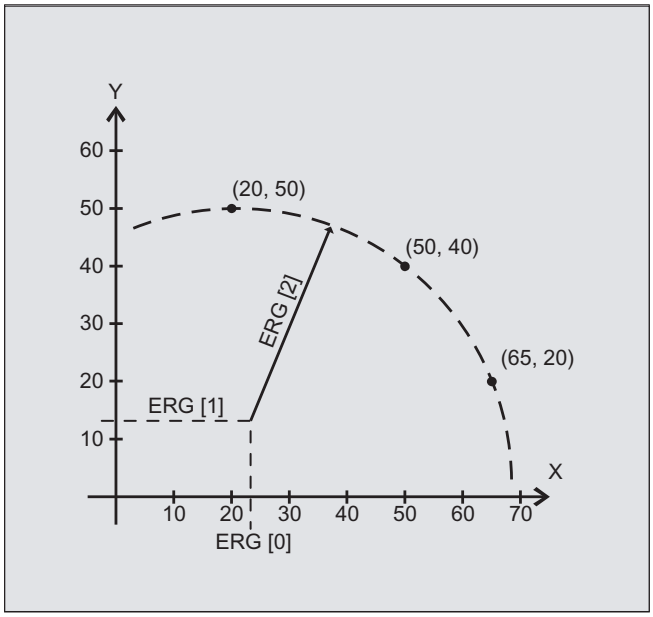
CALCDAT	3 개나 4 개 지점에서 원의 반경과 중심점 좌표를 계산하는 명령
<status>	원 계산 상태의 변수 형식 : BOOL 값 : TRUE 지정한 지점이 원에 놓임 FALSE 지정한 지점이 원을 벗어남
<circle points>[]	파라미터를 이용해 원 지점을 지정하는 변수 <number> 원 지점의 개수 (3 이나 4) <type> 2 점 좌표일 경우 좌표 데이터를 2 로 입력
<number>	계산에 사용한 지점 개수의 파라미터 (3 이나 4)
<result>[3]	결과의 변수 : 원의 중심점 좌표와 반경 0 원 중심점 좌표 : 가로축 값 1 원 중심점 좌표 : 세로축 값 2 반경

주

변수는 명령을 사용하기 전에 정의해야 합니다 .

예

3 개 지점을 사용할 때는 3 개 지점이 원에 놓이는지 확인해야 합니다 .



프로그램 코드	비고
N10 DEF REAL PT[3,2]=(20,50,50,40,65,20)	; 원에서 지점을 지정하는 변수
N20 DEF REAL RES[3]	; 결과의 변수
N30 DEF BOOL STATUS	; 상태의 변수
N40 STATUS=CALCDAT(PKT,3,ERG)	; 결정한 원 데이터를 호출
N50 IF STATUS == FALSE GOTOF ERROR	; 오류로 이동

15.7 형상 준비 종료 (EXECUTE)

작동

EXECUTE 명령은 형상 준비를 종료하고 정상 실행 모드로 복귀하는 데 사용합니다 .

구문

```
EXECUTE(<ERROR>)
```

의미

EXECUTE	형상 준비를 종료하는 명령
<FAULT>	고장 피드백 신호의 변수
	형식 : INT
	형상 준비 시 나타나는 변수의 값 :
	0 오류
	1 에러 없음

예

프로그램 코드

```
...
N30 CONTPRON(...)
N40 G1 X... Z...
...
N100 EXECUTE(...)
...
```

외부에서 싸이클 프로그래밍 16

16.1 공정 싸이클

16.1.1 소개

내용

이 장에서는 외부 NC 프로그램을 만들기 위해 버전 2.6 부터 사용되는 공정 싸이클에 대해 설명합니다 .

구조

문서는 다음과 같이 구성됩니다 .

- **프로그래밍**

  전송 파라미터의 싸이클 이름 및 호출 순서

- **파라미터**

  개별 파라미터를 설명하는 테이블

파라미터 설명

테이블에는 내부에서 사용되는 파라미터의 이름과 그 의미에 대한 설명 그리고 값 범위가 포함됩니다 . 파라미터 간의 관계도 설명합니다 . 마스크의 파라미터 참조를 위한 열은 컨트롤러에 대해 외부에서 생성된 싸이클 호출을 재컴파일할 때 프로그래밍된 값을 다시 찾는데 사용됩니다 .

" 인터페이스 전용 " 파라미터

특정 파라미터는 테이블에 " 인터페이스 전용 " 이라고 표시됩니다 . 이 파라미터는 싸이클 작동과 무관합니다. 이 파라미터는 싸이클 호출을 완벽하게 재컴파일하는 데에만 필요합니다. 파라미터를 프로그래밍하지 않아도 싸이클을 재컴파일할 수는 있습니다 . 그런 경우 필드는 색상별로 식별되며 마스크에서 완성해야 합니다 .

" 예비용 " 파라미터

" 예비용 " 으로 표시되는 파라미터는 값 0 또는 쉼표로 프로그래밍해야 다음 호출 파라미터 지정이 내부 싸이클 파라미터와 일치하게 됩니다 . 예외 : 값 "" 또는 쉼표를 사용한 문자열 파라미터 .

호환성

버전 2.6 이상의 공정 싸이클은 GIV 1.5 까지의 SINUMERIK 840D sl 용 싸이클 패키지 (버전 7.5까지의 싸이클) 가 더욱 발전된 형태입니다. 그 이전 소프트웨어 버전에 대한 싸이클 호출을 포함한 NC 프로그램도 계속 사용할 수 있습니다 .

대부분의 싸이클은 새 전송 파라미터에 의해 확장되거나 , 기존 파라미터 범위가 확장되어 새 기능을 프로그래밍할 수 있게 되었습니다 (예 : 자주 사용되는 가공 유형별 _VARI 파라미터).

이 문서에서 "호환성"이라는 말은 이전에 프로그래밍된 적이 없는 입력 값을 나타냅니다. 이에 따라 값이 제공되면 싸이클은 버전 7.5 까지와 동일한 기능으로 실행됩니다 .

위치 패턴에서 싸이클 반복

위치 패턴에서 드릴링 및 밀링 싸이클을 반복할 수 있습니다 (모달 호출). 그런 경우에는 싸이클 전과 동일한 라인에 MCALL 을 기록해야 합니다 (예 : MCALL CYCLE83(...)).

주

특정한 전송 파라미터 (예 : _VARI, _GMODE, _DMODE, _AMODE) 를 간접적으로 프로그래밍한 경우에는 재컴파일할 때 입력 마스크가 개방되지만 지정된 선택 필드에 대한 고유 할당이 없으므로 저장할 수는 없습니다 .

16.1.2 드릴링 , 센터링 - CYCLE81

프로그래밍

```
CYCLE81(REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL _DTB,
INT _GMODE, INT _DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	RP	RTP	후퇴 평면 (abs)
2	Z0	RFP	원점 (abs)
3	SC	_SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
4	Z1/ ∅	_DP	드릴링 깊이 (abs)/ 센터링 직경 (abs), _GMODE 참조
5	Z1	-DPR	드릴링 깊이 (inc)
6	DT	_DTB	최종 드릴링 깊이에서 드웰 시간 , _AMODE 참조
7		_GMODE	형상 모드 (프로그래밍된 형상 데이터 평가)
			UNITS: 예비용
			TENS: 깊이 / 직경을 기준으로 한 센터링
			0 = 호환성 , 깊이 1 = 직경
8		_DMODE	표시 모드
			UNITS: 가공 평면 G17/G18/G19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
9		_AMODE	대체 모드
			UNITS: 드릴링 깊이 Z1 (abs/inc)
			0 = 호환성 , DP/DPR 프로그래밍에서 1 = 증분 2 = 절대
			TENS: 최종 드릴링 깊이에서 드웰 시간 DT (초 / 회전)
			0 = 호환성 , DTB 부호에서 (> 0 초 또는 < 0 회전) 1 = 초 단위 2 = 회전 단위

16.1.3 드릴링 , 카운터보링 - CYCLE82

프로그래밍

```
CYCLE82(REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL DTB,
INT _GMODE, INT _DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	RP	RTP	후퇴 평면 (abs)
2	Z0	RFP	원점 (abs)
3	SC	SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
4	Z1	DP	드릴링 깊이 (abs), _AMODE 참조
5	Z1	DPR	드릴링 깊이 (inc), _AMODE 참조
6	DT	DTB	최종 드릴링 깊이에서 드웰 시간 , _AMODE 참조
7		_GMODE	형상 모드 (프로그래밍된 형상 데이터 평가)
			UNITS: 예비용
			TENS: 팁 / 직경을 기준으로 한 드릴링 깊이
			0 = 호환성 , 팁 1 = 직경
8		_DMODE	표시 모드
			UNITS: 가공 평면 G17/G18/G19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
9		_AMODE	대체 모드
			UNITS: 드릴링 깊이 Z1 (abs/inc)
			0 = 호환성 , DP/DPR 프로그래밍에서 1 = 증분 2 = 절대
			TENS: 최종 드릴링 깊이에서 드웰 시간 DT (초 / 회전)
			0 = 호환성 , DT 부호에서 (> 0 초 또는 < 0 회전) 1 = 초 단위 2 = 회전 단위

16.1.4 리밍 - CYCLE85

프로그래밍

```
CYCLE85(REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL DTB,
REAL FFR, REAL RFF, INT _GMODE, INT _DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	RP	RTP	후퇴 평면 (abs)
2	Z0	RFP	원점 (abs)
3	SC	SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
4	Z1	DP	드릴링 깊이 (abs), _AMODE 참조
5	Z1	DPR	드릴링 깊이 (inc), _AMODE 참조
6	DT	DTB	최종 드릴링 깊이에서 드웰 시간 , _AMODE 참조
7	F	FFR	이송 속도
8	FR	RFF	후퇴 동안 피드 속도
9		_GMODE	예비용
10		_DMODE	표시 모드
			UNITS: 가공 평면 G17/G18/G19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
11		_AMODE	대체 모드 (드릴링)
			UNITS: 드릴링 깊이 Z1 (abs/inc)
			0 = 호환성 , DP/DPR 프로그래밍에서 1 = 증분 2 = 절대
			TENS: 최종 드릴링 깊이에서 드웰 시간 DT (초 / 회전)
			0 = 호환성 , DT 부호에서 (> 0 초 또는 < 0 회전) 1 = 초 단위 2 = 회전 단위

16.1.5 심공 드릴링 - CYCLE83

프로그래밍

```
CYCLE83(REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL FDEP,
REAL FDPR, REAL _DAM, REAL DTB, REAL DTS, REAL FRF, INT VARI, INT
_AXN, REAL _MDEP, REAL _VRT, REAL _DTD, REAL _DIS1, INT _GMODE, INT
_DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	RP	RTP	후퇴 평면 (abs)
2	Z0	RFP	원점 (abs)
3	SC	SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
4	Z1	DP	최종 드릴링 깊이 (abs), _AMODE 참조
5	Z1	DPR	최종 드릴링 깊이 (inc), _AMODE 참조
6	D	FDEP	1. 드릴링 깊이 (abs), _AMODE 참조
7	D	FDPR	1. 드릴링 깊이 (inc), _AMODE 참조
8	DF	_DAM	각 추가 절입에 해당하는 거리 / 백분율 (하강 거리 / 백분율), _AMODE 참조
9	DTB	DTB	드릴링 깊이에서 드웰 시간 , _AMODE 참조
10	DTS	DTS	시작점에서 드웰 시간 (칩 제거에만 해당), _AMODE 참조
11	FD1	FRF	첫 번째 절입의 이송 속도에 대한 백분율 , _AMODE 참조
12		VARI	가공 형식
			UNITS: 칩 분쇄 / 칩 제거
			0 = 칩 분쇄 1 = 칩 제거
13		_AXN	공구 축 : 0 = 3. 기하 축 1 = 1. 기하 축 2 = 2. 기하 축 > 2 = 3. 기하 축
14	V1	_MDEP	최소 절입 (하강 백분률에만 해당)
15	V2	_VRT	각 가공 단계 후의 후퇴 거리 (칩 분쇄에만 해당) > 0 = 가변 후퇴 거리 0 = 표준 값 1 mm
16	DT	_DTD	최종 드릴링 깊이에서 드웰 시간 , _AMODE 참조
17	V3	_DIS1	제한 거리 (칩 제거에만 해당), _AMODE 참조
18		_GMODE	형상 모드 (프로그래밍된 형상 데이터 평가)
			UNITS: 예비용
			TENS: 팁 / 직경을 기준으로 한 드릴링 깊이
			0 = 팁 1 = 직경

번호	Param Mask	Param intern	설명
19		_DMODE	표시 모드
			UNITS: 가공 평면 G17/G18/G19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
20		_AMODE	대체 모드
			UNITS: 드릴링 깊이 = 최종 드릴링 깊이 Z1 (abs/inc)
			0 = 호환성 , DP/DPR 프로그래밍에서 1 = 증분 2 = 절대
			TENS: 최종 드릴링 깊이에서 드웰 시간 DTB (초 / 회전)
			0 = 호환성 , DTB 부호에서 (> 0 초 또는 < 0 회전) 1 = 초 단위 2 = 회전 단위
			HUNDREDS: DTS 의 시작점에서 드웰 시간 (초 / 회전)
			0 = 호환성 , DTS 부호에서 (> 0 초 또는 < 0 회전) 1 = 초 단위 2 = 회전 단위
			THOUSANDS: 최종 드릴링 깊이에서 드웰 시간 DT (초 / 회전)
			0 = 호환성 , DTD 부호에서 (> 0 초 또는 < 0 회전) 1 = 초 단위 2 = 회전 단위
			TEN THOUSANDS: 1. 드릴링 깊이 D (abs/inc)
			0 = 호환성 , FDEP/FDPR 프로그래밍에서 1 = 증분 2 = 절대
			HUNDRED THOUSANDS: 각 추가 절입에 해당하는 거리 / 백분율 DAM (하강)
			0 = 호환성 , DAM 부호에서 (> 0 초 또는 < 0 계수 0.001~1.0) 1 = 거리 2 = 백분율 (0.001 부터 100 % 까지)
			MILLION: 제한 거리 V3 자동 / 수동
			0 = 호환성 , _DIS1 부호에서 (= 0 자동 또는 > 0 수동) 1 = 자동 (싸이클에서 계산) 2 = 수동 (프로그래밍된 값)
			TEN MILLION: 첫 번째 절입 FRF 에 대한 계수 / 백분율로서의 이송 속도 계수
			0 = 호환성 , 계수로서 (0.001 ~ 1.0, FRF = 0 은 100% 를 의미함) 1 = 백분율 (0.001 부터 999.999 % 까지)

16.1.6 보링 - CYCLE86

프로그래밍

```
CYCLE86(REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL DTB,
INT SDIR, REAL RPA, REAL RPO, REAL RPAP, REAL POSS, INT _GMODE, INT
_DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	RP	RTP	후퇴 평면 (abs)
2	Z0	RFP	원점 (abs)
3	SC	SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
4	Z1	DP	드릴링 깊이 (abs), _AMODE 참조
5	Z1	DPR	드릴링 깊이 (inc), _AMODE 참조
6	DT	DTB	최종 드릴링 깊이에서 드웰 시간 , _AMODE 참조
7	DIR	SDIR	스핀들 회전 방향 3 = M3 4 = M4
8	DX	RPA	X 방향의 도피 거리
9	DY	RPO	Y 방향의 도피 거리
10	DZ	RPAP	Z 방향의 도피 거리
11	SPOS	POSS	도피를 위한 스핀들 위치 (방향이 지정된 스핀들 정지의 경우 , 단위 : 도)
12		_GMODE	형상 모드
			UNITS: 도피 모드
			0 = 도피 , 호환성 1 = 도피 안 함
13		_DMODE	표시 모드
			UNITS: 가공 평면 G17/G18/G19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
14		_AMODE	대체 모드
			UNITS: 드릴링 깊이 Z1 (abs/inc)
			0 = 호환성 , DP/DPR 프로그래밍에서 1 = 증분 2 = 절대
			TENS: 최종 드릴링 깊이에서 드웰 시간 DT (초 / 회전)
			0 = 호환성 , DT 부호에서 (> 0 초 또는 < 0 회전) 1 = 초 단위 2 = 회전 단위

16.1.7 동기 태핑 - CYCLE84

프로그래밍

```
CYCLE84(REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL DTB,
INT SDAC, REAL MPIT, REAL PIT, REAL POSS, REAL SST, REAL SST1, INT
_AXN, INT _PITA, INT _TECHNO, INT _VARI, REAL _DAM, REAL _VRT,
STRING[15] _PITM, STRING[5] _PTAB, STRING[20] _PTABA, INT _GMODE, INT
_DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명	
1	RP	RTP	후퇴 평면 (abs)	
2	Z0	RFP	원점 (abs)	
3	SC	SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)	
4	Z1	DP	드릴링 깊이 = 최종 드릴링 깊이 (abs), _AMODE 참조	
5	Z1	DPR	드릴링 깊이 = 최종 드릴링 깊이 (inc), _AMODE 참조	
6	DT	DTB	드릴링 깊이에서 머무는 초 단위의 드웰 시간	
7	SDE	SDAC	싸이클 종료 후 회전 방향	
8		MPIT	"ISO 미터법 " 전용의 나사 크기 (피치는 실행 도중 내부에서 계산됨)	
9	P	PIT	피치 값 , 단위는 _PITA 참조	
10	αS[1)]	POSS	방향이 지정된 스핀들 정지를 위한 스핀들 위치	
11	S	SST	태핑 스핀들 속도	
12	SR	SST1	후퇴 스핀들 속도	
13		_AXN	드릴링 축 : 0 = 3. 기하 축 1 = 1. 기하 축 2 = 2. 기하 축 ≥ 3 = 3. 기하 축	
14		_PITA	나사 피치 단위 0 = 피치 (mm) 1 = 피치 (mm) 2 = 피치 (TPI) 3 = 피치 (inch) 4 = MODULE	(PIT 및 MPIT 평가) - MPIT/PIT 평가 - PIT 평가 - PIT 평가 (인치당 나사 수) - PIT 평가 - PIT 평가

번호	Param Mask	Param intern	설명
15		_TECHNO	테크놀로지 [1)]
			UNITS: 정위치 정지 반응
			0 = 싸이클 호출 전과 같이 정위치 정지 반응 활성 상태 1 = 정위치 정지 G601 2 = 정위치 정지 G602 3 = 정위치 정지 G603
			TENS: 포워드 제어
			0 = 싸이클 호출 전과 같이 포워드 제어 사용 / 사용 안 함 1 = 포워드 제어 FFWON 사용 2 = 포워드 제어 FFWON 사용 안 함
			HUNDRED: 가속
			0 = 싸이클 호출 전과 같이 SOFT/BRISK/DRIVE 활성 상태 1 = 저크 제한 SOFT 사용 2 = 저크 제한 SOFT 사용 안 함 3 = 감속 DRIVE
			THOUSANDS: MCALL 스핀들 모드
			0 = MCALL 에서 스핀들 작업 재활성화 1 = MCALL 에서 위치 제어 유지
16		_VARI	가공 유형 :
			UNITS:
			0 = 1 절삭 1 = 칩 분쇄 (심공 태핑) 2 = 칩 제거 (심공 태핑)
			THOUSANDS: 입력 마스크의 경우에는 ISO/SIEMENS 모드가 적합하지 않음
			1 = ISO 호환성에 따른 호출 0 = SIEMENS 상황에 따른 호출
17	D	_DAM	최대 깊이 절입 (칩 제거 / 칩 분쇄만 해당)
18	V2	_VRT	각 가공 단계 후의 후퇴 거리 (칩 분쇄에만 해당), _AMODE 참조
19		_PITM	피치 입력을 위한 마커로서의 문자열 [2)]
20		_PTAB	나사 테이블의 문자열 ("", "ISO", "BSW", "BSP", "UNC")[2)]
21		_PTABA	나사 테이블에서 선택하기 위한 문자열 (예 : "M 10", "M 12", ...)[2)]
22		_GMODE	형상 모드 (프로그래밍된 형상 데이터 평가)
			UNITS: 예비용
			TENS: 예비용

번호	Param Mask	Param intern	설명
23		_DMODE	표시 모드
			UNITS: 가공 평면 G17/G18/G19
			0 = 호환성 , 싸이클 호출 이전에 유효한 평면은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
			TENS: 예비용
			HUNDREDS: 예비용
			THOUSANDS: 호환 모드 (또는 재컴파일 입력 마스크만), MD 52216 bit0 = 1 인 경우 [1]
			0 = 테크놀로지 파라미터가 표시됨 (호환성): TECHNO 파라미터 유효 1 = 테크놀로지 파라미터가 표시되지 않음 : " 싸이클 호출 이전과 같이 " 테크놀로지 활성 상태
24		_AMODE	대체 모드
			UNITS: 드릴링 깊이 = 최종 드릴링 깊이 Z1 (abs/inc)
			0 = 호환성 , DP/DPR 프로그래밍에서 1 = 증분 2 = 절대
			TENS: 예비용
			HUNDREDS: 예비용
			THOUSANDS: 나사 회전 방향 우측 / 좌측
			0 = 호환성 , PIT/MPTI 부호에서 1 = 우측 2 = 좌측
			TEN THOUSANDS: 예비용
			HUNDRED THOUSANDS: 예비용
			MILLION: 개별 가공 단계 이후 후퇴 거리 V2 수동 / 자동
			0 = 호환성 , _VRT 프로그래밍에서 (> 0 가변 값 또는 ≤ 0 표준 값 1 mm/0.0394 inch) 1 = 자동 (표준 값 1mm/0.0394 inch) 2 = 수동 (V2 에서와 마찬가지로 프로그래밍)

[1] 테크놀로지 필드는 설정 날짜 SD52216 $MCS_FUNCTION_MASK_DRILL 에 따라 숨겨질 수 있습니다 .

[2] 파라미터 19, 20, 21 은 입력 마스크 나사 테이블에서 나사 선택에만 사용됩니다 . 싸이클 런타임에는 싸이클 정의를 통해 나사 테이블에 액세스할 수 없습니다

16.1.8 태핑 척을 사용한 태핑 - CYCLE840

프로그래밍

```
CYCLE840(REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL DTB,
INT SDR, INT SDAC, INT ENC, REAL MPIT, REAL PIT, INT _AXN, INT _PITA,
INT _TECHNO, STRING[15] _PITM, STRING[5] _PTAB, STRING[20] _PTABA,
INT _GMODE, INT _DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	RP	RTP	후퇴 평면 (abs)
2	Z0	RFP	원점 (abs)
3	SC	SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
4	Z1	DP	드릴링 깊이 (abs), _AMODE 참조
5	Z1	DPR	드릴링 깊이 (inc), _AMODE 참조
6	DT	DTB	드릴링 깊이에서 드웰 시간 (초) / 후퇴 후 안전 거리 , ENC 참조
7		SDR	후진 시 회전 방향
8	SDE	SDAC	싸이클 종료 후 회전 방향
9		ENC	스핀들 장착된 엔코더를 사용한 태핑 (G33)/ 스핀들 장착된 엔코더를 사용하지 않은 태핑 (G63) 0 = 스핀들 장착된 엔코더 사용 - MPIT/PIT 와의 피치 - DT 사용 안 함 20 = 스핀들 장착된 엔코더 사용 - MPIT/PIT 와의 피치 - 안전 거리로 후퇴 후의 DT 사용 - MPIT/PIT 와의 피치 - 드릴링 깊이에서 DT 사용 11 = 스핀들 장착된 엔코더 사용 안 함 - 프로그래밍된 이송 속도와의 피치 - 드릴링 깊이에서 DT 사용 (이송 속도 = 속도 · 피치) 1 = 스핀들 장착된 엔코더 사용 안 함
10		MPIT	"ISO 미터법 " 전용의 나사 크기 (피치는 실행 도중 내부에서 계산됨) 값 범위 : 3 ~ 48 (M3 ~ M48 의 경우), PIT 를 대신함
11		PIT	피치 값 , 단위는 _PITA) 참조 값 범위 : > 0, MPIT 를 대신함
12		_AXN	드릴링 축 : 0 = 3. 기하 축 1 = 1. 기하 축 2 = 2. 기하 축 ≥ 3 = 3. 기하 축

번호	Param Mask	Param intern	설명
13		_PITA	피치 단위 (PIT 및 MPIT 평가)
			0 = 피치 (mm) - MPIT/PIT 평가 1 = 피치 (mm) - PIT 평가 2 = 피치 (TPI) - PIT 평가 (인치당 나사 수) 3 = 피치 (inch) - PIT 평가 4 = MODULE - PIT 평가
14		_TECHNO	테크놀로지 [1]
			UNITS: 정위치 정지 반응
			0 = 싸이클 호출 전과 같이 정위치 정지 활성 상태 1 = 정위치 정지 G601 2 = 정위치 정지 G602 3 = 정위치 정지 G603
			TENS: 포워드 제어
			0 = 싸이클 호출 전과 같이 포워드 제어 사용 / 사용 안 함 1 = 포워드 제어 FFWON 사용 2 = 포워드 제어 FFWON 사용 안 함
15		_PITM	피치 입력을 위한 마커로서의 문자열 [2]
16		_PTAB	나사 테이블의 문자열 ("", "ISO", "BSW", "BSP", "UNC")[2]
17		_PTABA	나사 테이블에서 선택하기 위한 문자열 (예 : "M 10", "M 12", ...)[2]
18		_GMODE	예비용
19		_DMODE	표시 모드
			UNITS: 가공 평면 G17/G18/G19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
			TENS: 예비용
			HUNDREDS: 예비용
			THOUSANDS: 호환 모드 (또는 재컴파일 입력 마스크만), MD 52216 bit0 = 1 인 경우 [1]
			0 = 테크놀로지 파라미터가 표시됨 (호환성): TECHNO 파라미터 유효 1 = 테크놀로지 파라미터가 표시되지 않음 : " 싸이클 호출 이전과 같이 " 테크놀로지 활성 상태
20		_AMODE	대체 모드
			UNITS: 드릴링 깊이 Z1 (abs/inc)
			0 = 호환성 , DP/DPR 프로그래밍에서 1 = 증분 2 = 절대

[1] 테크놀로지 필드는 설정 날짜 SD52216 MCS_FUNCTION_MASK_DRILL 에 따라 숨겨질 수 있습니다 .

[2] 파라미터 15, 16, 17 은 입력 마스크 나사 테이블에서 나사 선택에만 사용됩니다 . 싸이클 런타임에는 싸이클 정의를 통해 나사 테이블에 액세스할 수 없습니다

16.1.9 나사 밀링 - CYCLE78

프로그래밍

```
CYCLE78(REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _ADPR, REAL
_FDPR, REAL _LDPR, REAL _DIAM, REAL _PIT, INT _PITA, REAL _DAM, REAL
_MDEP, INT _VARI, INT _CDIR, REAL _GE, REAL _FFD, REAL _FRDP, REAL
_FFR, REAL _FFP2, INT _FFA, STRING[15] _PITM, STRING[20] _PTAB,
STRING[20] _PTABA, INT _GMODE, INT _DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	RP	_RTP	후퇴 평면 (abs)
2	Z0	_RFP	공구 축 원점 (abs)
3	SC	_SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
4	Z1	_DP	최종 드릴링 깊이 (abs/inc), _AMODE 참조
5		_ADPR	드릴링 속도 (inc) 를 줄인 사전 드릴링 깊이 . VARI TEN THOUSAND 에서 유효함
6	D	_FDPR	최대 깊이 절입 (inc) D ≥ Z1 ⇒ 최종 드릴링 깊이까지 1 회 절입 D < Z1 ⇒ 여러 번 절입 및 칩 제거를 포함한 심공 드릴링 싸이클
7	ZR	_LDPR	FR 피드로 관통 보링 (inc) 하는 경우의 나머지 드릴링 깊이
8	∅	-_DIAM	나사의 호칭경
9	P	_PIT	숫자 값 피치
10		_PITA	나사 피치 P 평가 1 = 피치 (mm/rev) 2 = 피치 (thread/inch) 3 = 피치 (inch/rev) 4 = 피치 (MODULE)
11	DF	_DAM	각 추가 절입에 해당하는 거리 / 백분율 (하강), _AMODE 참조
12	V1	_MDEP	최소 절입 (inc), 하강에서만 활성 상태
13		_VARI	가공 형식
			UNITS: 예비용
			TENS:
			0 = 나사 밀링 전에 칩 제거 안 함 (최종 드릴링 깊이에서만 활성 상태) 1 = 나사 밀링 전에 칩 제거 (최종 드릴링 깊이에서만 활성 상태)
			HUNDREDS:
			0 = 오른 나사 1 = 왼 나사
			THOUSANDS:
			0 = 드릴링 속도 FR 에서 남은 드릴링 깊이 없음 1 = 드릴링 속도 FR 에서 남은 드릴링 깊이 있음
			TEN THOUSANDS:
			0 = 감소된 속도로 사전 드릴링 안 함 1 = 감소된 속도로 사전 드릴링 사전 드릴링 속도 = 0.3 F1, F1< 0.15 mm/rev 인 경우 사전 드릴링 속도 = 0.1 mm/rev, F1 ≥ 0.15 mm/rev 인 경우

번호	Param Mask	Param intern	설명
14		_CDIR	밀링 방향 0 = 상향 1 = 일반 4 = 일반 + 상향 (황삭 + 정삭 조합)
15	Z2	_GE	나사 밀링 전의 후퇴 거리 (inc)
16	F1	_FFD	드릴링 속도 (mm/min, in/min 또는 mm/rev)
17	FR	_FRDP	남은 드릴링 깊이에 대한 드릴링 속도 (mm/min 또는 mm/rev)
18	F2	-FFR	나사 밀링 속도 (mm/min 또는 mm/tooth)
19	FS	_FFP2	CDIR=4 에 대한 정삭 속도 (mm/min 또는 mm/tooth)
20		_FFA	이송 속도 평가
			UNITS: 드릴링 피드 F1
			TENS: 남은 드릴링 깊이 FR 에 대한 드릴링 속도
			HUNDREDS: 나사 밀링 F2 에 대한 이송 속도
			THOUSANDS: 정삭 속도 FS
21		_PITM	피치 입력을 위한 마커로 사용되는 문자열 (인터페이스만 해당)[1)]
22		_PTAB	나사 테이블의 문자열 ("", "ISO", "BSW", "BSP", "UNC") (인터페이스만 해당)[1)]
23		_PTABA	나사 테이블에서 선택하기 위한 문자열 (예 : "M 10", "M 12", ...) (인터페이스만 해당)[1)]
24		_GMODE	형상 모드 , 예비용
25		_DMODE	표시 모드
			UNITS: 가공 평면 G17/18/19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
26		_AMODE	대체 모드
			UNITS: 드릴링 깊이 = 최종 드릴링 깊이 Z1 (abs/inc)
			0 = 절대 1 = 증분
			TENS: 각 추가 절입에 해당하는 거리 / 백분율 DF (하강)
			0 = 거리 1 = 백분율 (0.001 부터 100 % 까지)

주

1) 파라미터 21, 22, 23 은 입력 마스크 나사 테이블에서 나사 선택에만 사용됩니다 . 싸이클 런타임에는 싸이클 정의를 통해 나사 테이블에 액세스할 수 없습니다

16.1.10 자유 프로그래밍 위치 - CYCLE802

프로그래밍

```
CYCLE802(INT _XA, INT _YA, REAL _X0, REAL _Y0, REAL _X1, REAL _Y1,
REAL _X2, REAL _Y2, REAL _X3, REAL _Y3, REAL _X4, REAL _Y4, REAL _X5,
REAL _Y5, REAL _X6, REAL _Y6, REAL _X7, REAL _Y7, REAL _X8, REAL _Y8,
INT _VARI, INT _UMODE, INT _DMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1		_XA	모든 X 위치에 대한 대체 값 (9 자리 십진수) 자릿수 : 876543210 (숫자 위치는 드릴링 위치 Xn 에 해당) 위치 값 : 1 = 절대 (처음 프로그래밍된 위치가 항상 절대치임) 2 = 증분
2		_YA	모든 Y 위치에 대한 대체 값 (9 자리 십진수) 자릿수 : 876543210 (숫자 위치는 드릴링 위치 Yn 에 해당) 위치 값 : 1 = 위치 입력 (abs) 2 = 위치 입력 (inc)
3	X0	_X0	1. 위치 X
4	Y0	_Y0	1. 위치 Y
5	X1	_X1	2. 위치 X
6	Y1	_Y1	2. 위치 Y
7	X2	_X2	3. 위치 X
8	Y2	_Y2	3. 위치 Y
9	X3	_X3	4. 위치 X
10	Y3	_Y3	4. 위치 Y
11	X4	_X4	5. 위치 X
12	Y4	_Y4	5. 위치 Y
13	X5	_X5	6. 위치 X
14	Y5	_Y5	6. 위치 Y
15	X6	_X6	7. 위치 X
16	Y6	_Y6	7. 위치 Y
17	X7	_X7	8. 위치 X
18	Y7	_Y7	8. 위치 Y
19	X8	_X8	9. 위치 X
20	Y8	_Y8	9. 위치 Y
21		_VARI	예비용

번호	Param Mask	Param intern	설명
22		_UMODE	예비용
23		_DMODE	표시 모드
			UNITS: 가공 평면 G17/18/19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)

주

파라미터 X1/Y1 ~ X8/Y8 에 필요하지 않은 위치는 무시할 수 있습니다 .

그러나 9 개 위치 모두에 대해서 _XA 및 _YA 의 대체 값을 제공해야 합니다 .

16.1.11 여러 구멍으로 이루어진 행 – HOLES1

프로그래밍

```
HOLES1(REAL SPCA, REAL SPCO, REAL STA1, REAL FDIS, REAL DBH, INT NUM,
INT _VARI, INT _UMODE, STRING[200] _HIDE, INT _NSP, INT _DMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	X0	SPCA	첫 번째 축을 따라 여러 개의 홀로 이루어진 행의 원점 (abs)
2	Y0	SPCO	두 번째 축을 따라 여러 개의 홀로 이루어진 행의 원점 (abs)
3	α0	STA1	기본 회전 각도 (첫 번째 축에 대한 각도)
4	L0	FDIS	첫 번째 홀에서 원점까지의 거리
5	L	DBH	홀 사이 간격 지정
6	N	NUM	홀 개수
7		_VARI	예비용
8		_UMODE	예비용
9		_HIDE	숨겨진 위치 • 최대 198 자 • 연속 위치 번호 지정 (예 : "1,3", 위치 1 과 3 은 실행되지 않음)
10		_NSP	예비용
11		_DMODE	표시 모드
			UNITS: 가공 평면 G17/18/19
			0 = 호환성 , 싸이클 호출 이전에 유효한 평면은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)

16.1.12 그리드 또는 프레임 - CYCLE801

프로그래밍

```
CYCLE801(REAL _SPCA, REAL _SPCO, REAL _STA, REAL _DIS1, REAL _DIS2,
INT _NUM1, INT _NUM2, INT _VARI, INT _UMODE, REAL _ANG1, REAL _ANG2,
STRING[200] _HIDE, INT _NSP, INT _DMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	X0	_SPCA	첫 번째 축을 따른 위치 패턴 (그리드 / 프레임) 의 원점 (abs)
2	Y0	_SPCO	두 번째 축을 따른 위치 패턴 (그리드 / 프레임) 의 원점 (abs)
3	α0	_STA	기본 회전 각도 (첫 번째 축에 대한 각도) < 0 = CW 방향 회전 0 = CCW 방향 회전
4	L1	_DIS1	열에 대한 거리 (첫 번째 축과의 거리 , 부호 없이 입력)
5	L2	_DIS2	행에 대한 거리 (두 번째 축과의 거리 , 부호 없이 입력)
6	N1	_NUM1	열의 수
7	N2	_NUM2	행 수
8		_VARI	가공 형식
			UNITS: 위치 패턴
			0 = 그리드 1 = 프레임
			TENS: 예비용
			HUNDREDS: 예비용
9		_UMODE	예비용
10	αX	_ANG1	첫 번째 축과의 정렬 각도 (첫 번째 축과 사면으로 정렬된 라인) < 0 = CW 방향 측정 (0 ~ -90 도) > 0 = CCW 방향 측정 (0 ~ 90 도)
11	αY	_ANG2	두 번째 축과의 정렬 각도 (두 번째 축과 사면으로 정렬된 라인) < 0 = CW 방향 측정 (0 ~ -90 도) > 0 = CCW 방향 측정 (0 ~ 90 도)
12		_HIDE	숨겨진 위치 • 최대 198 자 • 연속 위치 번호 지정 (예 : "1,3", 위치 1 과 3 은 실행되지 않음)
13		_NSP	예비용
14		_DMODE	표시 모드
			UNITS: 가공 평면 G17/18/19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)

16.1.13 여러 개의 구멍으로 형성된 원 - HOLES2

프로그래밍

```
HOLES2(REAL CPA, REAL CPO, REAL RAD, REAL STA1, REAL INDA, INT NUM,
INT _VARI, INT _UMODE, STRING[200] _HIDE, INT _NSP, INT _DMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	X0	CPA	첫 번째 축을 따라 여래 개의 홀로 이루어진 원호의 중심점 (abs)
2	Y0	CPO	두 번째 축을 따라 여러 개의 홀로 이루어진 원호의 중심점 (abs)
3	R	RAD	원호 패턴의 반경
4	α0	STA1	시작각
5	α1	INDA	전진 각도 (피치 원에만 해당) < 0 = CW 방향 > 0 = CCW 방향
6	N	NUM	위치 개수 .
7		_VARI	가공 형식
			UNITS: 예비용
			TENS: 포지셔닝 유형
			0 = 접근 위치 - 직선 1 = 접근 위치 - 원호 경로
			HUNDREDS: : 예비용
			THOUSANDS: 원호 패턴
			0 = 호환 모드 , INDA = 0 인 경우 동심원 , INDA <> 0 인 경우 피치 원 1 = 동심원 2 = 피치 원
8		_UMODE	예비용
9		_HIDE	숨겨진 위치 • 최대 198 자 • 연속 위치 번호 지정 (예 : "1,3", 위치 1 과 3 은 실행되지 않음)
10		_NSP	예비용
11		_DMODE	표시 모드
			UNITS: 가공 평면 G17/18/19
			0 = 호환성 , 싸이클 호출 이전에 유효한 평면은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)

16.1.14 페이스 밀링 - CYCLE61

프로그래밍

```
CYCLE61(REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _PA, REAL
_PO, REAL _LENG, REAL _WID, REAL _MID, REAL _MIDA, REAL _FALD, REAL
_FFP1, INT _VARI, INT _LIM, INT _DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	RP	_RTP	후퇴 평면 (abs)
2	Z0	_RFP	공구 축 원점 , 소재 높이 (abs)
3	SC	_SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
4	Z1	_DP	완성 부품의 높이 (abs/inc), _AMODE 참조
5	X0	_PA	첫 번째 축의 코너 지점 1 (abs)
6	Y0	_PO	두 번째 축의 코너 지점 2 (abs)
7	X1	_LENG	첫 번째 축의 코너 지점 2 (abs/inc), _AMODE 참조
8	Y1	_WID	두 번째 축의 코너 지점 2 (abs/inc), _AMODE 참조
9	DZ	_MID	최대 깊이 절입
10	DXY	_MIDA	최대 평면 절입 (단위에 대해서는 _AMODE 참조)
11	UZ	_FALD	정삭 여유 , 깊이
12	F	_FFP1	가공 속도
13		_VARI	가공 형식
			UNITS: 가공
			1 = 황삭 2 = 정삭
			TENS: 가공 방향
			1 = 첫 번째 축과 평행 , 한 방향으로 2 = 두 번째 축과 평행 , 한 방향으로 3 = 첫 번째 축과 평행 , 여러 방향으로 4 = 두 번째 축과 평행 , 여러 방향으로

번호	Param Mask	Param intern	설명
14		_LIM	한계
			UNITS: 첫 번째 축 마이너스 방향 제한
			0 = 아니오 1 = 예
			TENS: 첫 번째 축 플러스 방향 제한
			0 = 아니오 1 = 예
			HUNDREDS: 두 번째 축 마이너스 방향 제한
			0 = 아니오 1 = 예
			THOUSANDS: 두 번째 축 플러스 방향 제한
			0 = 아니오 1 = 예
15		_DMODE	표시 모드
			UNITS: 가공 평면 G17/18/19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
16		_AMODE	대체 모드
			UNITS: 최종 깊이 (_DP)
			0 = 절대 1 = 증분
			TENS: 평면 절입 단위 (_MIDA)
			0 = mm 1 = 공구 직경의 %
			HUNDREDS: 예비용
			THOUSANDS: 표면 길이
			0 = 증분 1 = 절대
			TEN THOUSANDS: 표면 너비
			0 = 증분 1 = 절대

16.1.15 사각 포켓 밀링 - POCKET3

프로그래밍

```
POCKET3(REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _LENG, REAL
_WID, REAL _CRAD, REAL _PA, REAL _PO, REAL _STA, REAL _MID, REAL
_FAL, REAL _FALD, REAL _FFP1, REAL _FFD, INT _CDIR, INT _VARI, REAL
_MIDA, REAL _AP1, REAL _AP2, REAL _AD, REAL _RAD1, REAL _DP1, INT
_UMODE, REAL _FS, REAL _ZFS, INT _GMODE, INT _DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	RP	_RTP	후퇴 평면 (abs)
2	Z0	_RFP	공구 축 원점 (abs)
3	SC	_SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
4	Z1	_DP	포켓 깊이 (abs/inc), _AMODE 참조
5	L	_LENG	포켓 길이 (inc, 부호를 포함하여 입력)
6	W	_WID	포켓 너비 (inc, 부호를 포함하여 입력)
7	RP	_CRAD	포켓의 코너 반경
8	X0	_PA	원점 , 첫 번째 축 (abs)
9	YO	_PO	원점 , 두 번째 축 (abs)
10	α0	_STA	회전 각도 , 세로 축 (L) 과 첫 번째 축 사이의 각도
11	DZ	_MID	최대 깊이 절입
12	UXY	_FAL	정삭 여유 , 평면
13	UZ	_FALD	정삭 여유 , 깊이
14	F	_FFP1	평면에서 이송 속도
15	FZ	_FFD	깊이 절입 속도
16		_CDIR	밀링 방향 : 0 = 상향 1 = 일반
17		_VARI	가공 형식
			UNITS:
			1 = 황삭 2 = 정삭 4 = 에지 정삭 5 = 모따기
			TENS:
			0 = 사전 드릴링 , G0 으로 절입 1 = 수직 , G1 로 절입 2 = 헬리컬 3 = 포켓 세로 축을 따라 오실레이션
			HUNDREDS: 예비용

번호	Param Mask	Param intern	설명
18	DXY	_MIDA	최대 평면 절입 , 단위에 대해서는 _AMODE 참조
19	L1	_AP1	사전 가공 길이 (inc)
20	W1	_AP2	사전 가공 너비 (inc)
21	AZ	_AD	사전 가공 깊이 (inc)
22	ER	_RAD1	헬리컬 절입에서 헬리컬 경로의 반경
	EW		오실레이션을 위한 최대 절입 각도
23	EP	_DP1	헬리컬 절입에서 헬리컬 피치
24		_UMODE	예비용
25	FS	_FS	모따기 너비 (inc)
26	ZFS	_ZFS	모따기에서 삽입 깊이 (공구 팁) (abs/inc), _AMODE 참조
27		_GMODE	형상 모드
			UNITS: 예비용
			TENS: 예비용
			HUNDREDS: 가공 선택 / 시작점 계산만
			0 = 호환 모드 1 = 일반 가공
			THOUSANDS: 중심 / 코너를 지나는 치수 지정
			0 = 호환 모드 1 = 중심을 지나는 치수 지정 2 = 코너 지점의 치수 지정 , 포켓 위치 +LENG/+WID 3 = 코너 지점의 치수 지정 , 포켓 위치 -LENG/+WID 4 = 코너 지점의 치수 지정 , 포켓 위치 +LENG/-WID 5 = 코너 지점의 치수 지정 , 포켓 위치 -LENG/-WID
			TEN THOUSANDS: 전체 가공 / 재가공
			0 = 호환 모드 (이전과 같이 _AP1, _AP2 및 _AD 처리) 1 = 전체 가공 2 = 재가공
28		_DMODE	표시 모드
			UNITS: 가공 평면 G17/G18/G19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
			TENS: 이송 속도 유형 : G 그룹 (G94/G95), 표면 및 깊이 이송 속도의 경우
			0 = 호환 모드 1 = 싸이클 호출 이전과 같은 G 코드 . 표면 및 깊이 이송 속도의 경우 G94/G95 는 동일

번호	Param Mask	Param intern	설명
29		_AMODE	대체 모드
			UNITS: 포켓 깊이 (Z1)
			0 = 절대 (호환 모드) 1 = 증분
			TENS: 평면 절입 단위 (DXY)
			0 = mm 1 = 공구 직경의 %
			HUNDREDS: 모따기를 위한 삽입 깊이 (ZFS)
			0 = 절대 1 = 증분

16.1.16 원호 포켓 밀링 – POCKET4

프로그래밍

```
POCKET4(REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _CDIAM, REAL
_PA, REAL _PO, REAL _MID, REAL _FAL, REAL _FALD, REAL _FFP1, REAL
_FFD, INT _CDIR, INT _VARI, REAL _MIDA, REAL _AP1, REAL _AD, REAL
_RAD1, REAL _DP1, INT _UMODE, REAL _FS, REAL _ZFS, INT _GMODE, INT
_DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	RP	_RTP	후퇴 평면 (abs)
2	Z0	_RFP	공구 축 원점 (abs)
3	SC	_SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
4	Z1	_DP	포켓 깊이 (abs/inc), _AMODE 참조
5	∅	_DIAM	포켓 직경 또는 반경 , _DMODE 참조
6	X0	_PA	원점 , 첫 번째 축 (abs)
7	Y0	_PO	원점 , 두 번째 축 (abs)
8	DZ	_MID	최대 깊이 설정 , _VARI = 평면 참조 최대 헬리컬 설정 , _VARI = 헬리컬 참조
9	UXY	_FAL	정삭 여유 , 평면
10	UZ	_FALD	정삭 여유 , 깊이
11	F	_FFP1	표면 가공 이송 속도
12	FZ	_FFD	깊이 절입 속도
13		_CDIR	밀링 방향 0 = 상향 1 = 일반
14		_VARI	가공 형식
			UNITS:
			1 = 황삭 2 = 정삭 4 = 에지 정삭 5 = 모따기
			TENS: 절입 유형 (황삭 및 정삭)
			0 = 사전 드릴링 , G0 으로 절입 (포켓은 사전 가공) 1 = 수직 , G1 으로 절입 2 = 헬리컬
			HUNDRED: 예비용
			THOUSANDS:
			0 = 평면 1 = 헬리컬

번호	Param Mask	Param intern	설명
15	DXY	_MIDA	최대 평면 절입 , _AMODE, 0 = 0.8 · 공구 직경 참조
16	∅	_AP1	사전 가공 직경 / 반경 (inc)
17	AZ	_AD	사전 가공 깊이 (inc)
18	ER	_RAD1	헬리컬 절입에서 헬리컬 경로의 반경
19	EP	_DP1	헬리컬 경로에서 절입 시 헬리컬 피치
20		_UMODE	예비용
21	FS	_FS	모따기 너비 (inc)
22	ZFS	_ZFS	모따기에서 삽입 깊이 (공구 팁) (abs/inc), _AMODE 참조
23		_GMODE	형상 모드
			UNITS: 예비용
			TENS: 예비용
			HUNDREDS: 가공 / 시작점 계산
			0 = 호환 모드 1 = 일반 가공
			THOUSANDS: 예비용
			TEN THOUSANDS: 전체 가공 / 재가공
			0 = 호환 모드 (이전과 같이 _AP1 및 _AD 처리) 1 = 전체 가공 2 = 재가공
24		_DMODE	표시 모드
			UNITS: 가공 평면 G17/18/19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
			TENS: 이송 속도 유형 : G 그룹 (G94/G95), 표면 및 깊이 이송 속도의 경우
			0 = 호환 모드 1 = 싸이클 호출 이전과 같은 G 코드 . 표면 및 깊이 이송 속도의 경우 G94/G95 는 동일
			HUNDREDS:
			0 = 호환 모드 (반경으로 _CDIAM/_AP1 입력) 1 = 직경으로 _CDIAM/_AP1 입력
25		_AMODE	대체 모드
			UNITS: 포켓 깊이 (Z1)
			0 = 절대 (호환 모드) 1 = 증분
			TENS: 절입 너비 단위 (DXY)
			0 = mm 1 = 공구 직경의 %
			HUNDREDS: 모따기를 위한 삽입 깊이 (ZFS)
			0 = 절대 1 = 증분

16.1.17 사각 스피곳 밀링 - CYCLE76

프로그래밍

```
CYCLE76(REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _DPR, REAL
_LENG, REAL _WID, REAL _CRAD, REAL _PA, REAL _PO, REAL _STA, REAL
_MID, REAL _FAL, REAL _FALD, REAL _FFP1, REAL _FFD, INT _CDIR, INT
_VARI, REAL _AP1, REAL _AP2, REAL _FS, REAL _ZFS, INT _GMODE, INT
_DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	RP	_RTP	후퇴 평면 (abs)
2	Z0	_RFP	공구 축 원점 (abs)
3	SC	_SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
4	Z1	_DP	스피곳 깊이 (abs)
5		_DPR	Z0 를 기준으로 한 스피곳 깊이 (inc) (부호 없이 입력)
6	L	_LENG	스피곳 길이 , _GMODE 참조 (부호 없이 입력)
7	W	_WID	스피곳 너비 , _GMODE 참조 (부호 없이 입력)
8	R	_CRAD	스피곳 코너 반경 (부호 없이 입력)
9	X0	_PA	평명의 첫 번째 축에서 스피곳의 원점 (abs)
10	Y0	_PO	평명의 두 번째 축에서 스피곳의 원점 (abs)
11	α0	_STA	회전 각도 , 세로 축 (L) 과 평면 첫 번째 축 사이의 각도
12	DZ	_MID	최대 깊이 절입 (inc, 부호 없이 입력)
13	UXY	_FAL	정삭 여유 , 평면 (inc), 에지 형상의 공차
14	UZ	_FALD	정삭 여유 깊이 (inc), 베이스에서 공차 (부호 없이 입력)
15	FX	_FFP1	형상에서의 이송 속도
16	FZ	_FFD	깊이 절입 속도
17		_CDIR	밀링 방향 (부호 없이 입력)
			UNITS:
			0 = 상향 1 = 일반
18		_VARI	가공
			UNITS:
			1 = 황삭 2 = 정삭 5 = 모따기
19	L1	_AP1	소재 스피곳 길이
20	W1	_AP2	소재 스피곳 너비
21	FS	_FS	모따기 너비 (inc)
22	ZFS	_ZFS	모따기에서 삽입 깊이 (공구 팁) (abs, inc), _AMODE 참조

번호	Param Mask	Param intern	설명
23		_GMODE	프로그래밍된 형상 데이터 평가 모드
			UNITS: 예비용
			TENS: 예비용
			HUNDREDS: 가공 선택 또는 시작점 계산만
			0 = 호환 모드 1 = 일반 가공
			THOUSANDS: 중심 또는 코너 기준으로 스피곳 치수 지정
			0 = 호환 모드 1 = 중심을 지나는 치수 지정 2 = 코너 지점의 치수 지정 , 스피곳 +L +W 3 = 코너 지점의 치수 지정 , 스피곳 -L +W 4 = 코너 지점의 치수 지정 , 스피곳 +L -W 5 = 코너 지점의 치수 지정 , 스피곳 -L -W
			TEN THOUSANDS: 전체 가공 또는 재가공
			0 = 호환 모드 1 = 전체 가공 2 = 재가공
24		_DMODE	표시 모드
			UNITS: 가공 평면 G17/G18/G19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
25		_AMODE	대체 모드
			UNITS: 최종 깊이 Z1 (abs/inc)
			0 = 호환성 1 = Z1 (inc) 2 = Z1 (abs)
			TENS: 예비용
			HUNDREDS: 모따기를 위한 삽입 깊이 ZFS
			0 = ZFS (abs) 1 = ZFS (inc)

16.1.18 원주 밀링 - CYCLE77

프로그래밍

```
CYCLE77(REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _DPR, REAL
_CDIAM, REAL _PA, REAL _PO, REAL _MID, REAL _FAL, REAL _FALD, REAL
_FFP1, REAL _FFD, INT _CDIR, INT _VARI, REAL _AP1, REAL _FS, REAL
_ZFS, INT _GMODE, INT _DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	RP	_RTP	후퇴 평면 (abs)
2	Z0	_RFP	공구 축 원점 (abs)
3	SC	_SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
4	Z1	_DP	스피곳 깊이 (abs)
5		_DPR	Z0 를 기준으로 한 스피곳 깊이 (inc) (부호 없이 입력)
6	∅	_CDIAM	스피곳 직경 (부호 없이 입력)
7	X0	_PA	평명의 첫 번째 축에서 스피곳의 원점 (abs)
8	Y0	_PO	평명의 두 번째 축에서 스피곳의 원점 (abs)
9	DZ	_MID	최대 깊이 절입 (inc, 부호 없이 입력)
10	UXY	_FAL	정삭 여유 , 평면 (inc), 에지 형상의 공차
11	UZ	_FALD	정삭 여유 깊이 (inc), 베이스에서 공차 (부호 없이 입력)
12	FX	_FFP1	형상에서의 이송 속도
13	FZ	_FFD	깊이 절입 속도
14		_CDIR	밀링 방향 (부호 없이 입력)
			UNITS:
			0 = 상향 1 = 일반
15		_VARI	가공
			UNITS:
			1 = 최종 가공 공차까지 황삭 2 = 정삭 (여유 X/Y/Z=0) 5 = 모따기
16	∅ 1	_AP1	소재 스피곳 직경
17	FS	_FS	모따기 너비 (inc)
18	ZFS	_ZFS	모따기에서 삽입 깊이 (공구 팁) (abs/inc), _AMODE 참조

번호	Param Mask	Param intern	설명
19		_GMODE	프로그래밍된 형상 데이터 평가 모드
			UNITS: 예비용
			TENS: 예비용
			HUNDREDS: 가공 선택 / 시작점 계산만
			0 = 호환 모드 1 = 일반 가공
			THOUSANDS: 예비용
			TEN THOUSANDS: 전체 가공 / 재가공
			0 = 호환 모드 (이전과 같이 _AP1 처리) 1 = 전체 가공 2 = 재가공
20		_DMODE	표시 모드
			UNITS: 가공 평면 G17/G18/G19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
21		_AMODE	대체 모드
			UNITS: 최종 깊이 Z1 (abs/inc)
			0 = 호환성 1 = Z1 (inc) 2 = Z1 (abs)
			TENS: 예비용
			HUNDREDS: 모따기를 위한 삽입 깊이 ZFS
			0 = ZFS (abs) 1 = ZFS (inc)

16.1.19 다각형 - CYCLE79

프로그래밍

```
CYCLE79(REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, INT _NUM, REAL
_SWL, REAL _PA, REAL _PO, REAL _STA, REAL _RC, REAL _AP1, REAL _MIDA,
REAL _MID, REAL _FAL, REAL _FALD, REAL _FFP1, INT _CDIR, INT _VARI,
REAL _FS, REAL _ZFS, INT _GMODE, INT _DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	RP	_RTP	후퇴 평면 (abs)
2	Z0	_RFP	공구 축 원점 (abs)
3	SC	_SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
4	Z1	_DP	다각형 깊이 (abs/inc), _AMODE 참조
5	N	_NUM	면 개수 (1...n)
6	SW/L	_SWL	평판을 가로지르는 너비 및 에지 길이 (_VARI 에 따라) (평판을 가로지르는 너비의 경우 "SW", 에지 길이의 경우 "L") 평판을 가로지르는 너비 (에지 수가 짝수일 경우 및 단일 에지의 경우에만)
7	X0	_PA	스피곳 원점 , 첫 번째 축 (abs)
8	Y0	_PO	스피곳 원점 , 두 번째 축 (abs)
9	α0	_STA	회전 각도 , 첫 번째 축 기준으로 에지 중심 (X 축)
10	R1/FS1	_RC	_NUM > 2 일 때의 코너 라운딩 (반경 / 모따기 , _AMODE 참조) (inc, 부호 없이 입력) (반경의 경우 "R1", 모따기의 경우 "FS1")
11	∅	_AP1	가공되지 않은 스피곳 직경
12	DXY	_MIDA	최대 절입 너비 (단위에 대해서는 _AMODE 참조)
13	DZ	_MID	최대 깊이 절입
14	UXY	_FAL	정삭 여유 , 평면
15	UZ	_FALD	정삭 여유 , 깊이
16	F	_FFP1	가공 속도
17		_CDIR	밀링 방향
			0 = 상향 1 = 일반
18		_VARI	가공 형식
			UNITS: 가공
			1 = 황삭 2 = 정삭 3 = 에지 정삭 5 = 모따기
			TENS: 평판을 가로지르는 너비 및 에지 길이
			0 = 평판을 가로지르는 너비 1 = 에지 길이

번호	Param Mask	Param intern	설명
19	FS	_FS	모따기 너비 (inc)
20	ZFS	_ZFS	모따기에서 삽입 깊이 (공구 팁) (abs/inc), _AMODE 참조
21		_GMODE	형상 모드
			UNITS: 예비용
			TENS: 예비용
			HUNDREDS: 가공 / 시작점 계산
			1 = 일반 가공
22		_DMODE	표시 모드
			UNITS: 가공 평면 G17/G18/G19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
23		_AMODE	대체 모드
			UNITS: 최종 깊이 (_DP)
			0 = 절대 1 = 증분
			TENS: 평면 절입 단위 (_MIDA)
			0 = mm 1 = 공구 직경의 %
			HUNDREDS: 모따기를 위한 삽입 깊이 (_ZFS)
			0 = 절대 1 = 증분
			THOUSANDS: 코너 라운딩 (_RC)
			0 = 반경 1 = 모따기

16.1.20 일반 슬롯 - SLOT1

프로그래밍

```
SLOT1 (REAL RTP, REAL RFP, REAL SDIS, REAL _DP, REAL _DPR, INT NUM,
REAL LENG, REAL WID, REAL _CPA, REAL _CPO, REAL RAD, REAL STA1, REAL
INDA, REAL FFD, REAL FFP1, REAL _MID, INT CDIR, REAL _FAL, INT VARI,
REAL _MIDF, REAL FFP2, REAL SSF, REAL _FALD, REAL _STA2, REAL _DP1,
INT _UMODE, REAL _FS, REAL _ZFS, INT _GMODE, INT _DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	RP	RTP	후퇴 평면 (abs)
2	Z0	RFP	공구 축 원점 (abs)
3	SC	SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
4	Z1	_DP	슬롯 깊이 (abs)
5		_DPR	Z0 를 기준으로 한 슬롯 깊이 (inc) (부호 없이 입력)
6		NUM	슬롯 개수 = 1
7	L	LENG	슬롯 길이
8	W	WID	슬롯 너비
9	X0	_CPA	평면의 첫 번째 축 원점
10	Y0	_CPO	평면의 두 번째 축 원점
11		_RAD	예비용
12	α	STA1	회전각
13		INDA	예비용
14	FZ	FFD	깊이 절입 속도
15	F	FFP1	이송 속도
16	DZ	_MID	최대 깊이 절입
17		CDIR	밀링 방향 0 = 상향 1 = 일반
18	UXY	_FAL	평면 또는 슬롯 에지의 정삭 여유

번호	Param Mask	Param intern	설명
19		VARI	가공 형식
			UNITS:
			0 = 예비용 1 = 황삭 2 = 정삭 4 = 에지 정삭 (에지만 가공) 5 = 모따기
			TENS: 접근
			0 = 사전 드릴링 , G0 으로 절입 (슬롯은 사전 가공) 1 = 수직 , G1 로 절입 2 = 헬리컬 3 = 왕복
			HUNDREDS: 예비용
20	DZF	MIDF	예비용
21	FF	FFP2	예비용
22	SF	SSF	예비용
23	UZ	_FALD	정삭 여유 , 깊이
24	ER	_STA2	헬리컬 절입에서 헬리컬 경로의 반경
	EW		오실레이션을 위한 최대 절입 각도
25	EP	_DP1	나선에 대한 회전 당 삽입 깊이
26		_UMODE	예비용
27	FS	_FS	모따기를 위한 모따기 너비 (inc)
28	ZFS	_ZFS	모따기에서 삽입 깊이 (공구 팁) (abs/inc), _AMODE 참조
29		_GMODE	형상 모드
			UNITS: 예비용
			TENS: 예비용
			HUNDREDS: 가공 선택 또는 시작점 계산만
			1 = 일반 가공
			THOUSANDS: 원점 치수 지정 , 슬롯 길이
			0 = 중간 1 = 내부 좌측 +L 2 = 내부 우측 -L 3 = 좌측 에지 +L 4 = 우측 에지 -L

번호	Param Mask	Param intern	설명
30		_DMODE	표시 모드
			UNITS: 가공 평면 G17/18/19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
			TENS: 예비용
			HUNDREDS: 예비용
			THOUSANDS: 소프트웨어 버전 식별
			1 = 기능 확장 SLOT1
31		_AMODE	대체 모드
			UNITS: 최종 깊이 Z1 (abs/inc)
			0 = 호환성 1 = Z1 (inc) 2 = Z1 (abs)
			TENS: 예비용
			HUNDREDS: 모따기를 위한 삽입 깊이 ZFS
			0 = ZFS (abs) 1 = ZFS (inc)

주

이전 소프트웨어 버전에는 없던 새로운 기능을 포함한 싸이클이 제공됩니다 . 따라서 입력 마스크의 특정 파라미터 (NUM, RAD, INDA) 는 더 이상 표시되지 않습니다 . "MCALL" 을 사용하고 원하는 위치 패턴 (예 : HOLES2) 을 호출하여 한 위치에서 여러 슬롯을 프로그래밍할 수 있습니다 .

16.1.21 원호 슬롯 - SLOT2

프로그래밍

```
SLOT2(REAL RTP, REAL RFP, REAL SDIS, REAL _DP, REAL _DPR, INT NUM,
REAL AFSL, REAL WID, REAL _CPA, REAL _CPO, REAL RAD, REAL STA1, REAL
INDA, REAL FFD, REAL FFP1, REAL _MID, INT CDIR, REAL _FAL, INT VARI,
REAL _MIDF, REAL FFP2, REAL SSF, REAL _FFCP, INT _UMODE, REAL _FS,
REAL _ZFS, INT _GMODE, INT _DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	RP	RTP	후퇴 평면 (abs)
2	Z0	RFP	공구 축 원점 (abs)
3	SC	SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
4	Z1	_DP	슬롯 깊이 (abs)
5		_DPR	Z0 를 기준으로 한 슬롯 깊이 (inc) (부호 없이 입력)
6	N	NUM	슬롯 개수
7	α1	AFSL	슬롯의 틈 각도
8	W	WID	슬롯 너비
9	X0	_CPA	원점 = 원호 중심점 , 평면의 첫 번째 축
10	Y0	_CPO	원점 = 원호 중심점 , 평면의 두 번째 축
11	R	RAD	원의 반경
12	α0	STA1	시작각
13	α2	INDA	증분 각도
14	FZ	FFD	깊이 절입 속도
15	F	FFP1	이송 속도
16	DZ	_MID	최대 깊이 절입
17		CDIR	밀링 방향 0 = 상향 1 = 일반
18	UXY	_FAL	평면 또는 슬롯 에지의 정삭 여유
19		VARI	가공 형식
			UNITS:
			0 = 전체 가공 1 = 황삭 2 = 정삭 3 = 에지 정삭 5 = 모따기
			TENS:
			0 = G0 라인으로 중간 포지셔닝 1 = 원호 경로에 중간 포지셔닝
			HUNDREDS: 예비용
			THOUSANDS:
			0 = 호환 모드 , INDA = 0 인 경우 동심원 , INDA <> 0 인 경우 피치 원 1 = 동심원 2 = 피치 원

번호	Param Mask	Param intern	설명
20	DZF	_MIDF	예비용
21		FFP2	예비용
22		SSF	예비용
23	FF	_FFCP	예비용
24		_UMODE	예비용
25	FS	_FS	모따기 너비 (inc)
26	ZFS	_ZFS	모따기에서 삽입 깊이 (공구 팁) (abs/inc), _AMODE 참조
27		_GMODE	형상 모드
			UNITS: 예비용
			TENS: 예비용
			HUNDREDS: 가공 선택 또는 시작점 계산만
			0 = 호환 모드 1 = 일반 가공
28		_DMODE	표시 모드
			UNITS: 가공 평면 G17/18/19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
			TENS: 예비용
			HUNDREDS: 예비용
			THOUSANDS: 소프트웨어 버전 식별
			1 = 소프트웨어 버전 2.5 부터의 SLOT2 기능
29		_AMODE	대체 모드
			UNITS: 최종 깊이 Z1 (abs/inc)
			0 = 호환성 1 = Z1 (inc) 2 = Z1 (abs)
			TENS: 예비용
			HUNDREDS: 모따기를 위한 삽입 깊이 ZFS
			0 = ZFS (abs) 1 = ZFS (inc)

16.1.22 밀링 개방형 슬롯 - CYCLE899

프로그래밍

```
CYCLE899(REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _LENG, REAL
_WID, REAL _PA, REAL _PO, REAL _STA, REAL _MID, REAL _MIDA, REAL
_FAL, REAL _FALD, REAL _FFP1, INT _CDIR, INT _VARI, INT _GMODE, INT
_DMODE, INT _AMODE, INT _UMODE, REAL _FS, REAL _ZFS)
```

파라미터

번호	Param Mask	Param intern	설명
1	RP	_RTP	후퇴 평면 (abs)
2	Z0	_RFP	공구 축 원점 (abs)
3	SC	_SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
4	Z1	_DP	슬롯 깊이 (abs/inc), _AMODE 참조
5	L	_LENG	슬롯 길이 (inc)
6	W	_WID	슬롯 너비 (inc)
7	X0	_PA	원점 / 시작점 첫 번째 축 (abs)
8	Y0	_PO	원점 / 시작점 두 번째 축 (abs)
9	α0	_STA	첫 번째 축을 중심으로 한 회전 각도
10	DZ	_MID	최대 절입 깊이 (inc) – 나선 밀링에만 해당
11	DXY	_MIDA	최대 평면 절입 , _AMODE 참조
12	UXY	_FAL	정삭 여유 , 평면
13	UZ	_FALD	정삭 여유 , 깊이
14	F	_FFP1	이송 속도
15		_CDIR	밀링 방향
			UNITS:
			0 = 상향
			1 = 일반
			4 = 교차

번호	Param Mask	Param intern	설명
16		_VARI	가공
			UNITS:
			1 = 황삭 2 = 정삭 3 = 베이스 정삭 4 = 에지 정삭 5 = 중삭 6 = 모따기
			TENS: 예비용
			HUNDREDS: 예비용
			THOUSANDS:
			1 = 나선 밀링 2 = 플런지 절삭
17		_GMODE	형상 데이터 확인
			UNITS: 예비용
			TENS: 예비용
			HUNDREDS: 가공 선택 / 시작점 계산만
			1 = 일반 가공
			THOUSANDS: 중심 / 에지를 지나는 치수 지정
			0 = 중심을 지나는 치수 지정 1 = 에지를 사용한 " 좌측 " 치수 지정 (첫 번째 축 "-" 방향) 2 = 에지를 사용한 " 우측 " 치수 지정 (첫 번째 축 "+" 방향)
18		_DMODE	표시 모드
			UNITS: 가공 평면 G17/G18/G19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
19		_AMODE	대체 모드
			UNITS: 슬롯 깊이 Z1
			0 = 절대 1 = 증분
			TENS: 평면 절입 단위 (_MIDA) DXY
			0 = mm 1 = 공구 직경의 %
			HUNDREDS: 모따기를 위한 삽입 깊이 ZFS
			0 = 절대 1 = 증분
20		_UMODE	예비용
21	FS	_FS	모따기 너비 (inc)
22	ZFS	_ZFS	모따기에서 삽입 깊이 (공구 팁) (abs/inc), _AMODE 참조

16.1.23 연장된 홀 - LONGHOLE

프로그래밍

```
LONGHOLE (REAL RTP,REAL RFP,REAL SDIS,REAL _DP,REAL _DPR,
INT NUM,REAL LENG,REAL _CPA,REAL _CPO,REAL RAD,REAL STA1,
REAL INDA,REAL FFD,REAL FFP1,REAL MID,INT _VARI,INT _UMODE,
INT _GMODE,INT _DMODE,INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	RP	RTP	후퇴 평면 (abs)
2	Z0	_RFP	공구 축 원점 (abs)
3	SC	SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
4	Z1	_DP	긴 홀 깊이 (abs)
5		_DPR	Z0 를 기준으로 한 긴 홀 깊이 (inc) (부호 없이 입력)
6		NUM	긴 홀 개수 = 1
7	L	LENG	긴 홀의 길이
8	X0	_CPA	평면의 첫 번째 축 원점
9	Y0	_CPO	평면의 두 번째 축 원점
10		RAD	예비용
11	α0	STA1	회전각
12		INDA	예비용
13	FZ	FFD	깊이 절입 속도
14	F	FFP1	이송 속도
15	DZ	MID	최대 깊이 절입
16		_VARI	가공 형식
			UNITS: 절입 유형
			1 = G1 으로 수직 3 = 왕복
			HUNDRED: 예비용
17		_UMODE	예비용

번호	Param Mask	Param intern	설명
18		_GMODE	형상 모드
			UNITS: 예비용
			TENS: 예비용
			HUNDRED: 가공 선택 또는 시작점 계산만
			0 = 호환 모드 1 = 일반 가공
			THOUSANDS: 원점 치수 지정 , 슬롯 길이
			0 = 중간 1 = 내부 좌측 +L 2 = 내부 우측 -L 3 = 좌측 에지 +L 4 = 우측 에지 -L
19		_DMODE	표시 모드
			UNITS: 가공 평면 G17/18/19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
			TENS: 이송 속도 유형 : G 그룹 (G94/G95), 표면 및 깊이 이송 속도의 경우
			0 = 호환 모드 1 = 싸이클 호출 이전과 같은 G 코드 . 표면 및 깊이 이송 속도의 경우 G94/G95 는 동일
			HUNDREDS: 예비용
			THOUSANDS: 소프트웨어 버전 식별
			1 = 기능 확장 LONGHOLE (원점 치수 지정)
20		_AMODE	대체 모드
			UNITS: 최종 깊이 Z1 (abs/inc)
			0 = 호환성 1 = Z1 (inc) 2 = Z1 (abs)

주

이전 소프트웨어 버전에는 없던 새로운 기능을 포함한 싸이클이 제공됩니다 . 따라서 입력 마스크의 특정 파라미터 (NUM, RAD, INDA) 는 더 이상 표시되지 않습니다 . "MCALL" 을 사용하고 원하는 위치 패턴 (예 : HOLES2) 을 호출하여 한 위치에서 여러 슬롯을 프로그래밍할 수 있습니다 .

16.1.24 나사 밀링 - CYCLE70

프로그래밍

```
CYCLE70(REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _DIATH, REAL
_H1, REAL _FAL, REAL _PIT, INT _NT, REAL _MID, REAL _FFR, INT _TYPTH,
REAL _PA, REAL _PO, REAL _NSP, INT _VARI, INT _PITA, STRING[15]
_PITM, STRING[20] _PTAB, STRING[20] _PTABA, INT _GMODE, INT _DMODE,
INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	RP	_RTP	후퇴 평면 (abs)
2	Z0	_RFP	공구 축 원점 (abs)
3	SC	_SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
4	Z1	_DP	나사 길이 (abs/inc), _AMODE 참조 홀의 베이스에서 런아웃 고려 (최소 1/2 피치)
5	∅	_DIATH	나사의 호칭경
6	H1	_H1	나사 깊이
7	U	_FAL	정삭 여유
8	P	_PIT	피치 (select_PITA: mm, inch, MODULE, threads/inch)
9	NT	_NT	공구 팁에서 톱니의 수 공구 길이는 항상 하단 절삭날을 기준으로 합니다 .
10	DXY	_MID	절삭당 최대 절입 _MID > _H1: 한 번에 전체 절삭
11	F	_FFR	밀링 피드
12		_TYPTH	나사 유형 0 = 내경 나사 1 = 외경 나사
13	X0	_PA	원호 중심 첫 번째 축 (abs)
14	Y0	_PO	원호 중심 두 번째 축 (abs)
15	αS	_NSP	시작 각도 (멀티 스타트 나사)
16		_VARI	가공 형식
			UNITS:
			1 = 황삭 2 = 정삭
			TENS:
			1 = 하향 2 = 상향
			HUNDREDS:
			0 = 오른 나사
			1 = 왼 나사

번호	Param Mask	Param intern	설명
17		_PITA	나사 피치 평가 0 = 호환 모드 1 = 피치 (mm) 2 = 피치 (인치당 스레드 , TPI) 3 = 피치 (inch) 4 = 피치 (MODULE)
18		_PITM	피치 입력을 위한 마커로 사용되는 문자열 (인터페이스만 해당)
19		_RTAB	나사 테이블의 문자열 ("", "ISO", "BSW", "BSP", "UNC") (인터페이스만 해당)
20		_PTABA	나사 테이블에서 선택하기 위한 문자열 (예 : "M 10", "M 12", ...) (인터페이스만 해당)
21		_GMODE	형상 모드
			UNITS: 예비용
			TENS: 예비용
			HUNDREDS: 가공 / 시작점 계산
			0 = 호환 모드 1 = 일반 가공
22		_DMODE	표시 모드
			UNITS: 가공 평면 G17/18/19
			0 = 호환성 , 싸이클 호출 이전에 유효한 평면은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
23		_AMODE	대체 모드
			UNITS: 나사 길이 (_DP)
			0 = 절대 1 = 증분

16.1.25 조각 싸이클 - CYCLE60

프로그래밍

```
CYCLE60(STRING[200] _TEXT, REAL _RTP, REAL _RFP, REAL _SDIS, REAL
_DP, REAL _DPR, REAL _PA, REAL _PO, REAL _STA, REAL _CP1, REAL _CP2,
REAL _WID, REAL _DF, REAL _FFD, REAL _FFP1, INT _VARI, INT _CODEP,
INT _UMODE, INT _GMODE, INT _DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1		_TEXT	조각할 텍스트 (최대 100 자)
2	RP	_RTP	후퇴 평면 (abs)
3	Z0	_RFP	공구 축 원점 (abs)
4	SC	_SDIS	안전 거리 (기준 평면에 추가됨 , 부호 없이 입력)
5	Z1	_DP	깊이 (abs), _AMODE 참조
6	Z1	_DPR	깊이 (inc), _AMODE 참조
7	X0	_PA	평면의 첫 번째 축 원점 (abs) - 직각 , _VARI 참조
	R		원점 , 길이 (반경) - 극좌표 , _VARI 참조
8	Y0	_PO	평면의 두 번째 축 원점 (abs) - 직각 , _VARI 참조
	α0		원점 , 첫 번째 축 기준 각도 - 극좌표 , _VARI 참조
9	α1	_STA	텍스트 방향 , 첫 번째 축 기준으로 텍스트 라인의 각도 , _VARI 참조
10	XM	_CP1	텍스트의 원호 중심 , 평면의 첫 번째 축 (abs) - 직각 , _VARI 참조
	LM		텍스트의 원호 중심 , WNP 기준 길이 (반경) - 극좌표 , _VARI 참조
11	YM	_CP2	: 텍스트의 원호 중심 , 평면의 두 번째 축 (abs) - 직각 , _VARI 참조
	αM		텍스트의 원호 중심 , 첫 번째 축 기준 각도 - 극좌표 , _VARI 참조
12	W	_WID	글자 높이 (부호 없이 입력)
13	DX1 DX2	_DF	글자 간격 / 전체 너비 , _VARI 참조
	α2		틈 각도 , _VARI 참조
14	FZ	_FFD	깊이 절입 속도 , _DMODE 참조
15	F	_FFP1	표면 가공 이송 속도

번호	Param Mask	Param intern	설명
16		_VARI	가공 (조각 텍스트 정렬 및 원점)
			UNITS: 원점
			0: 사각형 1: 극 좌표
			TENS: 텍스트 정렬
			0: 한 라인으로 텍스트 정렬 1: 상향 커브 원호로 텍스트 정렬 2: 하향 커브 원호로 텍스트 정렬
			HUNDREDS: 예비용
			THOUSANDS: : 텍스트 원점 , 수평
			0: 좌측 1: 중앙 2: 우측
			TEN THOUSANDS: 텍스트 원점 , 수직
			0: 하단 1: 중앙 2: 상단
			HUNDRED THOUSANDS: 텍스트 길이
			0: 문자 크기 1: 전체 텍스트 너비 (직선 텍스트만) 2: 틈 각도 (원호 텍스트만 해당)
			MILLION: 원호 중심
			0: 직각 (직교 좌표)
			1: 극 좌표
			TEN MILLION: 미러링
			0 = 호환성
			1 = 미러링 ON
			2 = 미러링 OFF
17		_CODEP	작성할 코드 페이지 번호 (현재는 1252 만)
18		_UMODE	예비용
19		_GMODE	프로그래밍된 형상 데이터 평가 모드
			UNITS: 예비용
			TENS: 예비용
			HUNDREDS: 가공 선택 / 시작점 계산만
			0 = 호환 모드 1 = 일반 가공

번호	Param Mask	Param intern	설명
20		_DMODE	표시 모드
			UNITS: 가공 평면 G17/18/19
			0 = 호환성 , 싸이클 호출 이전에 유효한 평면은 계속 활성 상태 1 = G17 2 = G18 3 = G19
			TENS: 이송 속도 유형 : G 그룹 (G94/G95), 표면 및 깊이 이송 속도의 경우
			0 = 호환 모드 1 = 싸이클 호출 이전과 같은 G 코드 . 표면 및 깊이 이송 속도의 경우 G94/G95 는 동일
21		_AMODE	대체 모드
			UNITS: 최종 깊이 (_DP,_DPR)
			0 = 호환성 1 = 증분 (_DPR) 2 = 절대 (_DP)

16.1.26 형상 호출 - CYCLE62

프로그래밍

```
CYCLE62(STRING[140] _KNAME, INT _TYPE, STRING[32] _LAB1, STRING[32]
_LAB2)
```

파라미터

번호	Param Mask	Param intern	설명
1	PRG/ CON	_KNAME	_TYPE = 2 에서는 형상 이름 또는 서브루틴 이름을 프로그래밍할 필요가 없습니다 .
2		_TYPE	형상 입력 결정 0 = 서브루틴 1 = 형상 이름 2 = 레이블 3 = 서브루틴의 레이블
3	LAB1	_LAB1	레이블 1, 형상 시작
4	LAB2	_LAB2	레이블 2, 형상 끝

16.1.27 경로 밀링 - CYCLE72

프로그래밍

```
CYCLE72(STRING[141] _KNAME, REAL _RTP, REAL _RFP, REAL _SDIS, REAL
_DP, REAL _MID, REAL _FAL, REAL _FALD, REAL _FFP1, REAL _FFD, INT
_VARI, INT _RL, INT _AS1, REAL __LP1, REAL _FF3, INT _AS2, REAL _LP2,
INT _UMODE, REAL _FS, REAL _ZFS, INT _GMODE, INT _DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1		_KNAME	형상 서브루틴 이름
2	RP	_RTP	후퇴 평면 (abs)
3	Z0	_RFP	공구 축 원점 (abs)
4	SC	_SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
5	Z1	_DP	종점 , 최종 깊이 (abs/inc), _AMODE 참조
6	DZ	_MID	최대 깊이 절입 (inc, 부호 없이 입력)
7	UXY	_FAL	정삭 여유 , 평면 (inc), 에지 형상의 공차
8	UZ	_FALD	정삭 여유 깊이 (inc), 베이스에서 공차 (부호 없이 입력)
9	FX	_FFP1	형상에서의 이송 속도
10	FZ	_FFD	깊이 절입 이송 속도 (또는 공간 절입)
11		_VARI	가공 형식
			UNITS: 가공
			1 = 황삭 2 = 정삭 5 = 모따기
			TENS:
			0 = G0 을 사용하여 중간 지점으로 이동 1 = G1 을 사용하여 중간 지점으로 이동
			HUNDREDS:
			0 = 형상의 끝에서 원점까지 후퇴 1 = 형상의 끝에서 원점 +_SDIS 까지 후퇴 2 = 형상의 끝에서 _SDIS 만큼 후진 3 = 형상 끝에서 후퇴 없음 , 형상 피드로 다음 시작점에 접근
			THOUSANDS: 예비용
			TEN THOUSANDS:
			0 = 형상 포워드 가공 1 = 형상 백워드 가공 백워드 가공 제한 : • 최대 170 개 형상 요소 (모따기 또는 라운딩 포함) • (X/Y) 및 F 평면의 값만 평가됩니다 .

번호	Param Mask	Param intern	설명
12		_RL	가공 방향 40 = 형상 중심 (G40, 접근 및 후퇴 : 직선 또는 수직) 41 = 형상 좌측 (G41, 접근 및 후퇴 : 직선 또는 원호) 42 = 형상 우측 (G42, 접근 및 후퇴 : 직선 또는 원호)
13		_AS1	형상 접근 이동
			UNITS:
			1 = 직선 2 = 4 분원 3 = 반원 4 = 수직 접근 및 후퇴
			TENS:
			0 = 마지막 이동 , 평면에서 1 = 마지막 이동 , 공간에서
14	L1	_LP1	접근 경로 또는 접근 반경 (inc, 부호 없이 입력)
15	FZ	_FF3	중간 경로의 이송 속도 (형상의 경우 G94/G95)
16		_AS2	형상 접근 이동 (수직 접근 / 후퇴 아님)
			UNITS:
			1 = 직선 2 = 4 분원 3 = 반원
			TENS:
			0 = 마지막 이동 , 평면에서 1 = 마지막 이동 , 공간에서
17	L2	_LP2	후퇴 경로 또는 후퇴 반경 (inc, 부호 없이 입력)
18		_UMODE	예비용
19	FS	_FS	모따기 너비 (inc)
20	ZFS	_ZFS	모따기에서 삽입 깊이 (공구 팁) (abs/inc), _AMODE 참조
21		_GMODE	프로그래밍된 형상 데이터 평가 모드
			UNITS: 예비용
			TENS: 예비용
			HUNDREDS: 가공 선택 / 시작점 계산만
			0 = 호환 모드 1 = 일반 가공

번호	Param Mask	Param intern	설명
22		_DMODE	표시 모드
			UNITS: 가공 평면 G17/G18/G19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
			TENS: 이송 속도 유형 : G 그룹 (G94/G95), 표면 및 깊이 이송 속도의 경우
			0 = 호환 모드 1 = 싸이클 호출 이전과 같은 G 코드 . 표면 및 깊이 이송 속도의 경우 G94/G95 는 동일
			THOUSANDS:
			0 = 호환 모드 : 형상 이름은 _KNAME 에 있습니다 . 1 = 형상 이름이 CYCLE62 에서 프로그래밍되어 _SC_CONT_NAME 으로 전송됩니다 .
23		_AMODE	대체 모드
			UNITS: 종점 Z1 (_DP)
			0 = 절대 (호환 모드) 1 = 증분
			TENS: 평면 절입 단위
			0 = mm/inch 1 = 예비용
			HUNDREDS: 모따기를 위한 삽입 깊이 (_ZFS)
			0 = 절대 1 = 증분

주

다음 전송 파라미터가 간접적으로 (파라미터로서) 프로그래밍된 경우에는 입력 마스크가 리셋되지 않습니다 .

_VARI, _RL, _AS1, _AS2, _UMODE, _GMODE, _DMODE. _AMODE

16.1.28 형상 포켓 사전 드릴링 - CYCLE64

프로그래밍

```
CYCLE64(STRING[100] _PRG, INT _VARI, REAL _RP, REAL _Z0, REAL _SC,
REAL _Z1, REAL _F, REAL _DXY, REAL _UXY, REAL _UZ, INT _CDIR,
STRING[20] _TR, INT _DR, INT _UMODE, INT _GMODE, INT _DMODE, INT
_AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	PRG	_PRG	이름 드릴링 / 센터링 프로그램
2		_VARI	가공 형식
			UNITS: 예비용
			TENS: 예비용
			HUNDREDS: 예비용
			THOUSANDS: 도피 모드
			0 = 후퇴 평면으로 도피 1 = 원점 + 안전 거리로 도피
3	RP	_RP	후퇴 평면 (abs)
4	Z0	_Z0	원점 (abs)
5	SC	_SC	안전 거리 (원점에 추가됨 , 부호 없이 입력)
6	Z1	_Z1	드릴링 / 센터링 깊이 (_AMODE UNITS 참조)
7	F	_F	드릴링 / 센터링 이송 속도
8	DXY	_DXY	평면 절입 - 단위 (AMODE TENS 참조)
9	UXY	_UXY	정삭 여유 , 평면
10	UZ	_UZ	정삭 여유 , 깊이
11		_CDIR	밀링 방향 0 = 상향 1 = 일반
12	TR	_TR	기준 공구 이름
13	DR	_DR	기준 공구 D 번호
14		_UMODE	예비용
15		_GMODE	프로그래밍된 형상 데이터 평가 모드
			UNITS: 예비용
			TENS: 예비용
			HUNDREDS: 가공 선택 / 시작점 계산만
			0 = 일반 가공 (호환 모드는 필요 없음) 1 = 일반 가공 2 = 예비용

번호	Param Mask	Param intern	설명
25		_DMODE	표시 모드
			UNITS: 가공 평면 G17/18/19
			0 = 호환성 , 싸이클 호출 이전에 유효한 평면은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
			TENS: 테크놀로지 모드
			1 = 사전 드릴링 2 = 센터링
26		_AMODE	대체 모드
			UNITS: 드릴링 / 센터링 깊이 Z1
			0 = 절대 (호환 모드) 1 = 증분
			TENS: : 평면 절입 단위 (_DXY)
			0 = mm 1 = 공구 직경의 %

16.1.29 형상 포켓 밀링 - CYCLE63

프로그래밍

```
CYCLE63(STRING[100] _PRG, INT _VARI, REAL _RP, REAL _Z0, REAL _SC,
REAL _Z1, REAL _F, REAL _FZ, REAL _DXY, REAL _DZ, REAL _UXY, REAL
_UZ, INT _CDIR, REAL _XS, REAL _YS, REAL _ER, REAL _EP, REAL _EW,
REAL _FS, REAL _ZFS, STRING[20] _TR, INT _DR, INT _UMODE, INT _GMODE,
INT _DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	PRG	_PRG	제거 프로그램 이름
2		_VARI	가공 형식
			UNITS: 가공 공정
			1 = 황삭 3 = 베이스 정삭 4 = 에지 정삭 5 = 모따기
			TENS: 절입 유형
			0 = 중앙 절입 1 = 헬리컬 절입 2 = 왕복 절입
			HUNDREDS: 예비용
			THOUSANDS: 도피 모드
			0 = 후퇴 평면으로 도피 1 = 원점 + 안전 거리로 도피
			TEN THOUSANDS: 황삭 및 정삭 베이스의 시작점
			0 = 자동 1 = 수동
3	RP	_RP	후퇴 평면 (abs)
4	Z0	_Z0	공구 축 원점 (abs)
5	SC	_SC	안전 거리 (원점에 추가됨 , 부호 없이 입력)
6	Z1	_Z1	최종 깊이 (_AMODE UNITS 참조)
7	F	_F	황삭 / 정삭 과정에서 평면 이송 속도
8	FZ	_FZ	깊이 절입 속도
9	DXY	_DXY	평면 절입 - 단위 (AMODE TENS 참조)
10	DZ	_DZ	깊이 절입
11	UXY	_UXY	정삭 여유 , 평면
12	UZ	_UZ	정삭 여유 , 깊이
13		_CDIR	밀링 방향 0 = 상향 1 = 일반

번호	Param Mask	Param intern	설명
14	XS	_XS	시작점 X, 절대
15	YS	_YS	시작점 Y, 절대
16	ER	_ER	헬리컬 절입 : 반경
17	EP	_EP	헬리컬 절입 : 피치
18	EW	_EW	왕복 절입 : 최대 절입 각도
19	FS	_FS	모따기를 위한 모따기 너비 (inc)
20	ZFS	_ZFS	모따기할 때 공구 팁의 삽입 깊이 (AMODE HUNDREDS 참조)
21	TR	_TR	잔삭 가공 시 기준 공구 이름
22	DR	_DR	잔삭 가공 시 기준 공구 D 번호
23		_UMODE	예비용
24		_GMODE	프로그래밍된 형상 데이터 평가 모드
			UNITS: 예비용
			TENS: 예비용
			HUNDREDS: 가공 선택 / 시작점 계산만
			0 = 일반 가공 (호환 모드는 필요 없음) 1 = 일반 가공 2 = 예비용
25		_DMODE	표시 모드
			UNITS: 가공 평면 G17/18/19
			0 = 호환성 , 싸이클 호출 이전에 유효한 평면은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
			TENS: 예비용
			HUNDREDS: 테크놀로지 모드
			1 = 포켓 2 = 스피곳
			THOUSANDS: 잔삭 가공
			0 = 아니오 1 = 예
26		_AMODE	대체 모드
			UNITS: 최종 깊이 Z1
			0 = 절대 (호환 모드) 1 = 증분
			TENS: 평면 절입 단위 (_DXY)
			0 = mm 1 = 공구 직경의 %
			HUNDREDS: 모따기를 위한 삽입 깊이 (_ZFS)
			0 = 절대 1 = 증분

16.1.30 선삭 - CYCLE951

프로그래밍

```
CYCLE951(REAL _SPD, REAL _SPL, REAL _EPD, REAL _EPL, REAL _ZPD, REAL
_ZPL, INT _LAGE, REAL _MID, REAL _FALX, REAL _FALZ, INT _VARI, REAL
_RF1, REAL _RF2, REAL _RF3, REAL _SDIS, REAL _FF1, INT _NR, INT
_DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	X0	_SPD	원점 (abs, 항상 직경)
2	Z0	_SPL	원점 (abs)
3	X1	_EPD	종점
4	Z1	_EPL	종점
5	XM α1 α2	_ZPD	중간점 , _DMODE (TENS) 참조
6	ZM α1 α2	_ZPL	중간점 , _DMODE (TENS) 참조
7	위치	_LAGE	선삭 가공 코너 위치 0 = 외부 / 후방 1 = 외부 / 전방 2 = 내부 / 후방 3 = 내부 / 전방
8	D	_MID	절입 시 최대 깊이 절입
9	UX	_FALX	X 의 정삭 여유
10	UZ	_FALZ	Z 의 정삭 여유

번호	Param Mask	Param intern	설명
11		_VARI	가공 형식
			UNITS: 좌표계 상에서 선삭 방향 (길이 또는 너비)
			1 = 세로 2 = 가로
			TENS:
			1 = 정삭 여유까지 황삭 2 = 정삭
			HUNDREDS:
			0 = 형상의 라운딩 포함 , 잔여 코너 없음 1 = 형상의 라운딩 없음
			THOUSANDS:
			0 = 코너 2 에 반경 / 모따기 있음 1 = 코너 2 에 언더컷 있음
			TEN THOUSANDS:
			0 = 가공 후 정지 상태 1 = 시작 위치로 복귀
12	R1/FS1	_RF1	라운딩 반경 또는 모따기 너비 1, _AMODE (TEN THOUSANDS) 참조
13	R2/FS2	_RF2	라운딩 반경 또는 모따기 너비 2, _AMODE (HUNDRED THOUSANDS) 참조
14	R3/FS3	_RF3	라운딩 반경 또는 모따기 너비 3, _AMODE (ONE MILLION) 참조
15	SC	_SDIS	안전 거리
16	F	_FF1	황삭 / 정삭의 이송 속도
17		_NR	선삭 유형 확인 (형상 선택을 위한 수직 소프트 키에 해당): 0 = 선삭 1, 모따기 / 라운딩 없는 90° 코너 1 = 선삭 2, 모따기 / 라운딩 있는 90° 코너 2 = 선삭 3, 모따기 / 라운딩 있는 모든 코너
18		_DMODE	표시 모드
			UNITS: 가공 평면 G17/G18/G19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
			TENS: 입력 _ZPD/_ZPL 의 형태
			0 = Xm/Zm 1 = Xm/α1 2 = Xm/α2 3 = α1/Zm 4 = α2/Zm 5 = α1/α2

번호	Param Mask	Param intern	설명
21		_AMODE	대체 모드
			UNITS: X 방향 중간점
			0 = 절대 , 직경에서 가로 축의 값 1 = 증분 , 반경에서 가로 축의 값
			TENS: Z 방향 중간점
			0 = 절대 1 = 증분
			HUNDREDS: X 방향 종점
			0 = 절대 , 직경에서 가로 축의 값 1 = 증분 , 반경에서 가로 축의 값
			THOUSANDS: Z 방향 종점
			0 = 절대 1 = 증분
			TEN THOUSANDS: 반경 / 모따기 1
			0 = 반경 1 = 모따기
			HUNDRED THOUSANDS: 반경 / 모따기 2
			0 = 반경 1 = 모따기
			MILLION: 반경 / 모따기 3
			0 = 반경 1 = 모따기

16.1.31 홈 - CYCLE930

프로그래밍

```
CYCLE930(REAL _SPD, REAL _SPL, REAL _WIDG, REAL _WIDG2, REAL _DIAG,
REAL _DIAG2, REAL _STA, REAL _ANG1, REAL _ANG2, REAL _RCO1, REAL
_RCI1, REAL _RCI2, REAL _RCO2, REAL _FAL, REAL _IDEP1, REAL _SDIS,
INT _VARI, INT _DN, INT _NUM, REAL _DBH, REAL _FF1, INT _NR, REAL
_FALX, REAL _FALZ, INT _DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	X0	_SPD	평면 축의 원점 (항상 직경)
2	Z0	_SPL	세로 축의 원점
3	B1	_WIDG	홈 하단 너비
4	B2	_WIDG2	홈 상단 너비 (인터페이스만 해당)
5	T1	_DIAG	원점에서 홈 깊이 abs 및 세로 가공의 경우 = 직경 , 기타의 경우 inc
6	T2	_DIAG2	원점 반대편의 홈 깊이 (인터페이스만 해당), abs 및 세로 가공의 경우 = 직경 , 기타의 경우 inc
7	α0	_STA	경사 각도 (-180 ≤ _STA ≤ 180)
8	α1	_ANG1	원점에 따라 결정되는 홈 측면의 측면 각도 1 (0 ≤ _ANG1 < 90)
9	α2	_ANG2	원점 반대편의 측면 각도 2 (0 ≤ _ANG2 < 90)
10	R1/FS1	_RCO1	라운딩 반경 또는 모따기 너비 1, 원점에서 외부
11	R2/FS2	_RCI1	라운딩 반경 또는 모따기 너비 2, 원점에서 내부
12	R3/FS3	_RCI2	라운딩 반경 또는 모따기 너비 3, 원점 반대편 내부
13	R4/FS4	_RCO2	라운딩 반경 또는 모따기 너비 4, 원점 반대편 외부
14	U	_FAL	X 및 Z 방향의 정삭 여유 , _VARI (TEN THOUSANDS) 참조 (부호 없이 입력)
15	D	_IDEP1	삽입 시 최대 깊이 절입 (부호 없이 입력) 0 = 1. 전체 깊이로 직접 절삭 > 0 = 1. Cut_IDEP1, 두 번째 절삭 2 · _IDEP1 등
16	SC	_SDIS	안전 거리 (부호 없이 입력)

번호	Param Mask	Param intern	설명
17		_VARI	가공 형식
			UNITS: 예비용
			TENS: 가공 공정
			1 = 황삭 2 = 정삭 3 = 황삭과 정삭
			HUNDREDS: 세로 / 가로 외부 / 내부 +Z/+Z 및 +X/-X 포지셔닝
			1 = 세로 / 외부 +Z 2 = 가로 / 내부 -X 3 = 세로 / 내부 +Z 4 = 가로 / 내부 +X 5 = 세로 / 외부 -Z 6 = 가로 / 외부 -X 7 = 세로 / 내부 -Z 8 = 가로 / 외부 +X
			THOUSANDS: 원점 위치
			0 = 상단 원점 1 = 하단 원점
			TEN THOUSANDS: 정삭 여유의 효과 정의
			0 = 형상과 평행하는 정삭 여유 U 1 = 분리된 UX 및 UZ 정삭 여유
18		_DN	공구의 두 번째 에지의 D 번호 > 0 = 홈 가공 공구의 두 번째 에지 수정을 위한 D 번호 0 = 두 번째 에지가 프로그래밍되지 않음
19	N	_NUM	홈 개수 (0 = 1 홈)
20	DP	_DBH	홈 간격 (_NUM > 1 일 때만 필요)
21	F	_FF1	이송 속도
22		_NR	홈 형상 식별은 형상 선택용 수직 소프트 키에 해당합니다 . 0 = 모따기 / 라운딩이 없는 90° 측면 1 = 모따기 / 라운딩을 포함한 경사 측면 (α0 사용 안 함) 2 = 1 과 같으나 테이퍼 위에서 (α0 사용)
23	UX	_FALX	X 축의 정삭 여유 , _VARI (TEN THOUSANDS) 참조 (부호 없이 입력)
24	UZ	_FALZ	Z 축의 정삭 여유 , _VARI (TEN THOUSANDS) 참조 (부호 없이 입력)
25		_DMODE	표시 모드
			UNITS: 가공 평면 G17/G18/G19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)

번호	Param Mask	Param intern	설명
26		_AMODE	대체 모드
			UNITS: 홈 상단 치수 지정 (인터페이스만 해당)
			0 = 원점에서 1 = 원점 반대편
			TENS: 깊이
			0 = 절대 1 = 증분
			HUNDREDS: 너비 치수 지정 (인터페이스만 해당)
			0 = 외부 직경에서 (상단) 1 = 내부 직경에서 (하단)
			THOUSANDS: 반경 / 모따기 1 (_RCO1)
			0 = 반경 1 = 모따기
			TEN THOUSANDS: 반경 / 모따기 2 (_RC11)
			0 = 반경 1 = 모따기
			HUNDRED THOUSANDS: 반경 / 모따기 3 (_RC12)
			0 = 반경 1 = 모따기
			MILLIONS POSITION: 반경 / 모따기 4 (_RCO2)
			0 = 반경 1 = 모따기

16.1.32 언더컷 형상 - CYCLE940

CYCLE940 싸이클을 사용하여 다양한 언더컷을 프로그래밍할 수 있습니다 . 일부 경우에는 파라미터 설정에 따라 크게 다릅니다 .

테이블의 추가 열은 어떤 언더컷 유형에 어떤 파라미터가 필요한지를 나타냅니다 . 이것은 싸이클 화면 양식의 수직 선택 소프트 키에 해당합니다 .

- E: 언더컷 형상 E
- F: 언더컷 형상 F
- A-D: DIN 나사 언더컷 (A-D 형상)
- T: 나사 언더컷 (임의의 형상 정의)

프로그래밍

```
CYCLE940(REAL _SPD, REAL _SPL, CHAR _FORM, INT _LAGE, REAL _SDIS,
REAL _FFP,INT _VARI, REAL _EPD, REAL _EPL, REAL _R1, REAL _R2, REAL
_STA, REAL _VRT, REAL _MID, REAL _FAL, REAL _FALX, REAL _FALZ, INT
_PITI, STRING[5] _PTAB, STRING[20] _PTABA, INT _DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	형상 프로그래밍				설명
			E	F	A-D	T	
1	X0	_SPD	x	x	x	x	평면 축의 원점 (항상 직경)
2	Z0	_SPL	x	x	x	x	세로 축의 원점 (abs)
3	FORM	_FORM	x	x	x	x	언더컷 형상 (대문자 , 예 : "T") 선택 , 언더컷 값을 가져와야 할 테이블 A = 외부 , 참조 DIN76, A = 정상 B = 외부 , 참조 DIN76, B = 짧음 C = 외부 , 참조 DIN76, C = 정상 D = 외부 , 참조 DIN76, D = 짧음 E = 참조 DIN509 F = 참조 DIN509 T = 임의의 형상
4	POSITION	_LAGE	x	x	x	x	언더컷의 위치 (평행 Z) 0 = 외부 +Z: ____| 1 = 외부 -Z: |____/ 2 = 내부 +Z: /-----| 3 = 내부 -Z: |-----\
5	SC	_SDIS	x	x	x	x	안전 거리 (inc)
6	F	_FFP	x	x	x	x	가공 속도 (mm/rev)

번호	Param Mask	Param intern	형상 프로그래밍				설명
7		_VARI	-	-	x	x	가공 형식
							UNITS: 가공
							1 = 황삭 2 = 정삭 3 = 황삭 + 정삭
							TENS: 가공 방식
							0 = 형상과 평행 1 = 세로
			언더컷 형상 E 와 F 는 항상 정삭과 같이 단일 통과로 가공됩니다 .				
8	X1	_EPD	x	x	-	-	공차 X (abs/inc), _AMODE 참조
			-	-	-	x	언더컷 깊이 (abs/inc), _AMODE 참조
9	Z1	_EPL	-	x	-	-	공차 Z
			-	-	-	x	언더컷 너비 (abs/inc), _AMODE 참조
10	R1	_R1	-	-	-	x	경사의 라운딩 반경
11	R2	_R2	-	-	-	x	코너의 라운딩 반경
12	α	_STA	-	-	x	x	절입 각도
13	VX	_VRT	x	x	-	-	교차 피드 X (abs/inc), _AMODE 참조
			-	-	x	x	정삭 시의 교차 피드 X (abs/inc), _AMODE 참조
14	D	_MID	-	-	x	x	깊이 절입
15	U	_FAL	-	-	x	x	형상과 평행하는 정삭 여유 , _AMODE 참조
16	UX	_FALX	-	-	x	x	정삭 여유 X
17	UZ	_FALZ	-	-	x	x	정삭 여유 Z
18	P	_PITI	-	-	x	-	피치 선택 , 형상 A-D, M1 ... M68 에 해당
							0 = 0.20 1 = 0.25 2 = 0.30 3 = 0.35 4 = 0.40 5 = 0.45 6 = 0.50 7 = 0.60 8 = 0.70 9 = 0.75 10 = 0.80 11 = 1.00 12 = 1.25 13 = 1.50 14 = 1.75 15 = 2.00 16 = 2.50 17 = 3.00 18 = 3.50 19 = 4.00 20 = 4.50 21 = 5.00 22 = 5.50 23 = 6.00
			x	x	-	-	반경 / 깊이 선택 , 형상 E, F
							0 = 0.6 · 0.3 1 = 1.0 · 0.4 2 = 1.0 · 0.2 3 = 1.6 ··0.3 4 = 2.5 · 0.4 5 = 4.0 · 0.5 6 = 0.4 · 0.2 7 = 0.6 · 0.2 8 = 0.1 · 0.1 9 = 0.2 ··0.1
19		_PTAB					나사 테이블의 문자열 ("", "ISO", "BSW", "BSP", "UNC") (인터페이스만 해당)
20		_PTABA					나사 테이블에서 선택하기 위한 문자열 (예 : "M 10", "M 12", ...) (인터페이스만 해당)

번호	Param Mask	Param intern	형상 프로그래밍				설명
21		_DMODE					표시 모드
			x	x	x	x	UNITS: 가공 평면 G17/18/19
							0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
22		_AMODE					대체 모드
			x	x	-	x	UNITS: 파라미터 _EPD 공차 X 또는 언더컷 깊이
							0 = 절대 (항상 직경) 1 = 증분
			x	x	-	x	TENS: 파라미터 _EPL 공차 Z 또는 언더컷 너비
							0 = 절대 1 = 증분
			x	x	x	x	HUNDREDS: 파라미터 _VRT 교차 피드 X
							0 = 절대 (항상 직경) 1 = 증분
			-	-	x	x	THOUSANDS: 정삭 여유
							0 = 형상과 평행하는 정삭 여유 (_FAL) = 분리된 가공 공차 (_FALX/_FALZ)

16.1.33 나사 선삭 - CYCLE99

프로그래밍

```
CYCLE99(REAL _SPL, REAL _SPD, REAL _FPL, REAL _FPD, REAL _APP, REAL
_ROP, REAL _TDEP, REAL _FAL, REAL _IANG, REAL _NSP, INT _NRC, INT
_NID, REAL _PIT, INT _VARI, INT _NUMTH, REAL _SDIS, REAL _MID, REAL
_GDEP, REAL _PIT1, REAL _FDEP, INT _GST, INT _GUD, REAL _IFLANK, INT
_PITA, STRING[15] _PITM, STRING[20] _PTAB, STRING[20] _PTABA, INT
_DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	Z0	_SPL	원점 (abs)
2	X0	_SPD	원점 (abs, 항상 직경)
3	Z1	_FPL	종점 , _AMODE (UNITS) 참조
4	X1	_FPD	종점 , _AMODE (TENS) 참조
5	LW/ LW2	_APP	나사 접근 , _AMODE (HUNDREDS) 참조 또는 나사 런인 = 나사 런아웃 , _AMODE (HUNDREDS) 참조
6	LR	_ROP	나사 런아웃
7	H1	_TDEP	나사 깊이
8	U	_FAL	X 와 Z 방향 정삭 여유
9	DP	-_IANG	거리 또는 각도로서의 절입 경사 , _AMODE (THOUSANDS) 참조
	αP		> 0 = 플러스 방향 절입 < 0 = 마이너스 방향 절입 0 = 중앙 절입
10	α0	_NSP	시작 각도 옵셋 (" 싱글 스타트 " 에서만 유효)
11	ND	_NRC	황삭 절삭 개수 , _VARI (TEN THOUSANDS) 와 조합
12	NN	_NID	비 절삭 개수
13	P	_PIT	피치 값 , _PITA 참조

번호	Param Mask	Param intern	설명
14		_VARI	가공 형식
			UNITS: 테크놀로지
			1 = 직선 절입을 사용한 외경 나사 2 = 직선 절입을 사용한 내경 나사 3 = 하강 절입을 사용한 외경 나사 , 절삭 단면은 일정함 4 = 하강 절입을 사용한 내경 나사 , 절삭 단면은 일정함
			TENS: 예비용
			HUNDREDS: 절입 유형
			1 = 한 면에서 절입 2 = 대체 면 절입
			THOUSANDS: 예비용
			TEN THOUSANDS: 대체 깊이 절입
			0 = 사전 설정된 황삭 절삭 개수 (_NRC) 1 = 첫 번째 절입의 사전 설정 값 (_MID)
			HUNDRED THOUSANDS: 가공 형식
			1 = 황삭 2 = 정삭 3 = 황삭과 정삭
			MILLION: 멀티 스타트 나사의 가공 순서
			0 = 나사의 오름차순으로 1 = 나사의 내림차순으로
15	N	_NUMTH	나사의 수
16	VR	_SDIS	후퇴 거리 , inc
17	D1	_MID	첫 번째 절입 깊이 , _VARI (TEN THOUSANDS) 참조
18	DA	_GDEP	나사 변경 깊이 0 = 나사 변경 깊이를 준수하지 않음 > 0 = 나사 변경 깊이 준수
19	G 코드	_PIT1	회전당 피치 변경 0 = 피치가 일정함 (G33) > 0 = 피치 증가 (G34) > 0 = 피치 감소 (G35)
20		_FDEP	삽입 깊이 (부호 없이 입력)
21	N1	_GST	시작 나사 N1 = 1...N, _AMODE (HUNDRED THOUSANDS) 참조
22		_GUD	예비용
23		_IFLANK	너비로서의 절입 경사 (인터페이스만 해당)
24		_PITA	피치 단위 (PIT 및 / 또는 MPIT 평가) 0 = 피치 단위 (mm) - MPIT/PIT 평가 1 = 피치 단위 (mm) - PIT 평가 2 = 피치 단위 (TPI) - PIT 평가 (인치당 스레드) 3 = 피치 단위 (inch) - PIT 평가 4 = MODULE- PIT 평가

번호	Param Mask	Param intern	설명
25		_PITM	피치 입력을 위한 마커로 사용되는 문자열 (인터페이스만 해당)[1)]
26		_PTAB	나사 테이블의 문자열 (인터페이스만 해당)[1)]
27		_PTABA	나사 테이블의 선택 문자열 (인터페이스만 해당)[1)]
28		_DMODE	표시 모드
			UNITS: 가공 평면 G17/G18/G19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
			TENS: 나사 유형
			0 = 수직 나사 1 = 단면 나사 2 = 테이퍼 나사
29		_AMODE	대체 모드
			UNITS: Z 방향의 나사 길이
			0 = 절대 1 = 증분
			TENS: X 방향의 나사 길이
			0 = 절대 , 직경에서 가로 축의 값 1 = 증분 , 반경에서 가로 축의 값 2 = α
			HUNDREDS: 접근 / 런인 경로 계산 _APP
			0 = 나사 접근 _APP 1 = 나사 런인 = 나사 런아웃 _APP = -_ROP 2 = 나사 런인 경로 _APP = -_APP 지정
			THOUSANDS: 절입 경사를 각도 또는 너비로 선택
			0 = 절입 각도 _IANG 1= 절입 경사 _IFLANK
			TEN THOUSANDS: 단일 / 다중 나사
			0 = 단일 나사 (시작 각도 옵셋 _NSP 포함) 1 = 다중 나사
			HUNDRED THOUSANDS 시작 나사 _GST
			0 = 전체 가공 1 = 이 나사부터 가공 시작 2 = 이 나사만 가공

주

1) 파라미터 _PITM, _PTAB, _PTABA 는 입력 마스크 나사 테이블에서 나사 선택에만 사용됩니다 .

싸이클 런타임에는 싸이클 정의를 통해 나사 테이블에 액세스할 수 없습니다

16.1.34 나사 체인 - CYCLE98

프로그래밍

```
CYCLE98(REAL _PO1, REAL _DM1, REAL _PO2, REAL _DM2, REAL _PO3, REAL
_DM3, REAL _PO4, REAL _DM4, REAL APP, REAL ROP, REAL TDEP, REAL FAL,
REAL _IANG, REAL NSP, INT NRC, INT NID, REAL _PP1, REAL _PP2, REAL
_PP3, INT _VARI, INT _NUMTH, REAL _VRT, REAL _MID, REAL _GDEP, REAL
_IFLANK, INT _PITA, STRING[15] _PITM1, STRING[15] _PITM2, STRING[15]
_PITM3, INT _DMODE,INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	Z0	_PO1	Z 의 원점 (abs)
2	X1	_DM1	X 방향 원점 (abs), 직경
3	Z1	_PO2	Z 방향 중간점 1 (abs/inc), _AMODE (UNITS) 참조
4	X1	_DM2	X 방향 중간점 1 (abs/inc), _AMODE (TENS) 참조
	X1α		나사 경사 1 (-90° ~ 90°) abs 는 항상 직경 , inc 는 항상 반경
5	Z2	_PO3	Z 방향 중간점 2 (abs/inc), _AMODE (HUNDREDS) 참조
6	X2	_DM3	X 방향 중간점 2 (abs/inc), _AMODE (THOUSANDS) 참조
	X2α		나사 경사 2 (-90° ~ 90°) abs 는 항상 직경 , inc 는 항상 반경
7	Z3	_PO4	Z 방향 종점 (abs/inc), _AMODE (TEN THOUSANDS) 참조
8	X3	_DM4	X 방향 종점 (abs/inc), _AMODE (HUNDRED THOUSANDS) 참조
	X3α		나사 경사 3 (-90° ~ 90°) abs 는 항상 직경 , inc 는 항상 반경
9	LW	APP	나사 접근 (inc, 부호 없이 입력)
10	LR	ROP	나사 런아웃 (inc, 부호 없이 입력)
11	H1	TDEP	나사 깊이 (inc, 부호 없이 입력)
12	U	FAL	X 와 Z 방향 정삭 여유
13	DP	_IANG	거리 또는 각도로서의 절입 경사 , _AMODE (MILLION) 참조
	αP		절입 경사는 파라미터 _VARI (HUNDREDS) 의 설정에 따라 적용됩니다 . _VARI_HUNDERTER = 0 의 정의 - 호환 모드 : > 0 = 한 면에서 측면 절입 0 = 나사에서 수직 절입 < 0 = 대체 면을 포함한 측면 절입 _VARI_HUNDERTER<>0 의 정의 : > 0 = 플러스 방향 절입 0 = 중앙 절입 < 0 = 마이너스 방향 절입
14	α0	NSP	첫 번째 나사의 시작 각도 옵셋
15		NRC	황삭 절삭 개수 , _VARI (TEN THOUSANDS) 참조

번호	Param Mask	Param intern	설명
16	NN	NID	비 절삭 개수
17	P0	_PP1	나사 첫 번째 섹션의 피치 , _PITA 참조
18	P1	_PP2	나사 두 번째 섹션의 피치 , _PITA 참조
19	P2	_PP3	나사 세 번째 섹션의 피치 , _PITA 참조
20		_VARI	가공
			UNITS: 테크놀로지
			1 = 직선 절입을 사용한 외경 나사 2 = 직선 절입을 사용한 내경 나사 3 = 하강 절입을 사용한 외경 나사 , 절삭 단면은 일정함 4 = 하강 절입을 사용한 내경 나사 , 절삭 단면은 일정함
			TENS: 예비용
			HUNDREDS: 절입 유형
			0 = _IANG 의 호환 모드 1 = 한 면에서 절입 2 = 대체 면 절입
			THOUSANDS: 예비용
			TEN THOUSANDS: 대체 깊이 절입
			0 = 호환성 , 사전 설정된 황삭 절삭 개수 (_NRC) 1 = 첫 번째 절입의 사전 설정 값 (_MID)
			HUNDRED THOUSANDS: 가공 형식
			0 = 호환성 (황삭과 정삭) 1 = 황삭 2 = 정삭 3 = 황삭과 정삭
			MILLION: 멀티 스타트 나사의 가공 순서
			0 = 나사의 오름차순으로 1 = 나사의 내림차순으로
21	N	_NUMTH	나사의 수
22		_VRT	후퇴 거리 (inc) 0 = 작동 시스템에 관계 없이 내부적으로 1 mm 의 도피 거리가 사용됩니다 (inch 또는 미터). > 0 = 도피 거리
23	D1	_MID	첫 번째 절입 , _VARI (TEN THOUSANDS) 참조
24	DA	_GDEP	나사 변경 깊이 (" 다중 스타트 " 에서만 유효) 0 = 나사 변경 깊이를 준수하지 않음 > 0 = 나사 변경 깊이 준수
25		_IFLANK	너비로서의 절입 경사 (인터페이스만 해당)
26		_PITA	나사 피치 평가 0 = 피치의 호환 모드 , 이전처럼 작동 시스템에 따라 _PP1 에서 _PP3 까지 평가 (미터 /inch) 1 = 피치 (mm) 2 = 피치 (인치당 스레드 , TPI) 3 = 피치 (inch) 4 = MODULE
27		_PITM1	피치 입력을 위한 마커로 사용되는 문자열 (인터페이스만 해당)

번호	Param Mask	Param intern	설명
28		_PITM2	피치 입력을 위한 마커로 사용되는 문자열 (인터페이스만 해당)
29		_PITM3	피치 입력을 위한 마커로 사용되는 문자열 (인터페이스만 해당)
30		_DMODE	표시 모드
			UNITS: 가공 평면 G17/18/19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
31		_AMODE	대체 모드
			UNITS: 1. Z 방향 중간점 (Z1)
			0 = 절대 1 = 증분
			TENS: 1. X 방향 중간점 (X1)
			0 = 절대 1 = 증분 2 = α
			HUNDREDS: 2. Z 방향 중간점 (Z2)
			0 = 절대 1 = 증분
			THOUSANDS: 2. X 방향 중간점 (X2)
			0 = 절대 1 = 증분 2 = α
			TEN THOUSANDS: Z 방향 종점 (Z3)
			0 = 절대 1 = 증분
			HUNDRED THOUSANDS: X 방향 종점 (X3)
			0 = 절대 1 = 증분 2 = α
			MILLION: 절입 경사를 각도 또는 너비로 선택
			0 = 절입 각도 _IANG 1= 절입 경사 _IFLANK
			TEN MILLIONS: 단일 / 다중 나사
			0 = 호환 모드 (시작 각도 _NSP 가 평가됨) 1 = 단일 나사 (시작 각도 옵셋 _NSP 포함) 2 = 다중 나사

16.1.35 절단 - CYCLE92

프로그래밍

```
CYCLE92(REAL _SPD, REAL _SPL, REAL _DIAG1, REAL _DIAG2, REAL _RC,
REAL _SDIS, REAL _SV1, REAL _SV2, INT _SDAC, REAL _FF1, REAL _FF2,
REAL _SS2, REAL _DIAGM, INT _VARI, INT _DN, INT _DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	X0	_SPD	원점 (abs, 항상 직경)
2	Y0	_SPL	원점 (abs)
3	X1	_DIAG1	속도 감소를 위한 깊이 , _AMODE (UNITS) 참조
4	X2	_DIAG2	최종 깊이 , _AMODE (TENS) 참조
5	R/FS	_RC	라운딩 상태 또는 모따기 너비 , _AMODE (THOUSANDS) 참조
6	SC	_SDIS	안전 거리 (원점에 추가됨 , 부호 없이 입력)
7	S	_SV1	일정한 스핀들 속도 , _AMODE (TEN THOUSANDS) 참조
	V		주속 일정 제어
8	SV	_SV2	주속 일정 제어에서 최대 속도
9	DIR	_SDAC	스핀들 회전 방향 3 = M3 의 경우 4 = M4 의 경우
10	F	_FF1	속도 감소를 위한 깊이만큼 절입
11	FR	_FF2	최종 깊이만큼 감소된 절입
12	SR	_SS2	최종 깊이만큼 감소된 속도
13	XM	_DIAGM	파트 - 케쳐 철회 깊이 (abs, 항상 직경)
14		_VARI	가공 형식
			UNITS: 후퇴
			0 = _SPD+_SDIS 로 후퇴 1 = 끝에서는 후퇴 없음
			TENS: 파트 - 케쳐
			0 = 아니오 . M 명령을 실행하지 않음 1 = 예. CUST_TECHCYC(101)에서 호출 - open drawer, CUST_TECHCYC(102)에서 호출 - close drawer
15		_DN	공구의 두 번째 에지의 D 번호 , 프로그래밍되지 않은 경우 ⇒ D+1
20		_DMODE	표시 모드
			UNITS: 가공 평면 G17/G18/G19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)

번호	Param Mask	Param intern	설명
21		_AMODE	대체 모드
			UNITS: 속도 감소를 위한 깊이 (_DIAG1)
			0 = 절대 , 직경에서 가로 축의 값 1 = 증분 , 반경에서 가로 축의 값
			TENS: 최종 깊이 (_DIAG2)
			0 = 절대 , 직경에서 가로 축의 값 1 = 증분 , 반경에서 가로 축의 값
			HUNDREDS: 예비용
			THOUSANDS: 반경 / 모따기 (_RC)
			0 = 반경 1 = 모따기
			TEN THOUSANDS: 스핀들 속도 / 절삭 속도 (_SV1)
			0 = 일정한 스핀들 속도 1 = 주속 일정 제어

16.1.36 형상 홈 가공 - CYCLE952

프로그래밍

```
CYCLE952(STRING[100] _PRG, STRING[100] _CON, STRING[100] _CONR, INT
_VARI, REAL _F, REAL _FR, REAL _RP, REAL _D, REAL _DX, REAL _DZ, REAL
_UX, REAL _UZ, REAL _U, REAL _U1, INT _BL, REAL _XD, REAL _ZD, REAL
_XA, REAL _ZA, REAL _XB, REAL _ZB, REAL _XDA, REAL _XDB, INT _N, REAL
_DP, REAL _DI, REAL _SC, INT _DN, INT _GMODE, INT _DMODE, INT _AMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1	PRG	_PRG	선삭 가공 프로그램 이름
2	CON	_CON	소재의 업데이트된 형상을 읽어올 프로그램 이름 (잔삭 가공의 경우)
3	CONR	_CONR	소재의 업데이트된 형상 (_AMODE TEN THOUSANDS 참조) 이 기록될 프로그램 이름
4		_VARI	가공 형식
			UNITS: 선삭 유형
			1 = 세로 2 = 가로 3 = 형상과 평행
			TENS: 가공 공정 , (_GMODE HUNDREDS 참조)
			1 = 황삭 2 = 정삭 3 = 전체 가공
			HUNDREDS: 가공 방향
			1 = 가공 방향 X - 2 = 가공 방향 X + 3 = 가공 방향 Z - 4 = 가공 방향 Z +
			THOUSANDS: 절입 방향
			1 = 외부 X- 2 = 내부 X+ 3 = 전면 Z - 4 = 후면 Z +
			TEN THOUSANDS: 정삭 여유의 효과 정의
			0 = 분리된 UX 및 UZ 정삭 여유 1 = 형상과 평행하는 정삭 여유 U
			HUNDRED THOUSANDS: 라운딩
			0 = 호환성 , 자동 라운딩 1 = 형상의 라운딩 있음 2 = 라운딩 없음 3 = 자동 라운딩
			MILLION: 릴리프 절삭
			0 = 홈 가공 시 위치가 평가되지 않음 , - 잔여 소재 및 홈 선삭 , - 나머지 1 = 릴리프 절삭 가공 2 = 릴리프 절삭 가공 안 함
			TEN MILLION: 터닝 센터의 뒤쪽 / 앞쪽
			0 = 터닝 센터의 앞쪽에서 가공 1 = 예비용

번호	Param Mask	Param intern	설명
5	F	_F	황삭 / 정삭의 이송 속도
	FZ		절입 가로 좌표 홈 선삭
6	FR	_FR	릴리프 절삭부로 삽입을 위한 이송 속도 , 황삭
	FX		절입 세로 좌표 홈 선삭
7	RP	_RP	내부 가공을 위한 후퇴 평면 (abs, 항상 직경)
8	D	_D	황삭 절입 (_AMODE UNITS 참조)
9	DX	_DX	X 절입 (_AMODE UNITS 참조)
10	DZ	_DZ	Z 절입 (_AMODE UNITS 참조)
11	UX	_UX	정삭 여유 X, (_VARI TEN THOUSANDS 참조)
12	UZ	_UZ	정삭 여유 Z, (_VARI TEN THOUSANDS 참조)
13	U	_U	형상과 평행한 정삭 여유 , (_VARI TEN THOUSANDS 참조)
14	U1	_U1	정삭 도중의 추가 정삭 여유 (_AMODE THOUSANDS 참조)
15	BL	_BL	소재 정의 1 = 공차 포함 실린더 2 = 정삭된 파트의 형상 공차 3 = 소재의 형상이 제공됨
16	XD	_XD	소재 X 의 정의 (_AMODE HUNDRED THOUSANDS 참조)
17	ZD	_ZD	소재 Z 의 정의 (_AMODE MILLION 참조)
18	XA	_XA	한계 1 X (abs, 항상 직경)
19	ZA	_ZA	한계 1 Z (abs)
20	XB	_XB	한계 2 X (_AMODE TEN MILLION 참조)
21	ZB	_ZB	한계 2 Z (_AMODE HUNDRED MILLION 참조)
22	XDA	_XDA	전면 홈 가공을 위한 홈 가공 한계 1 (abs, 항상 직경)
23	XDB	_XDB	전면 홈 가공을 위한 홈 가공 한계 2 (abs, 항상 직경)
24	N	_N	슬롯 개수
25	DP	_DP	홈 간격 일반 홈 : Z 축 방향 가공 가로 축 홈 : X 축 방향 가공
26	DI	_DI	절입 중단을 위한 거리 0 = 중단 없음 0 > 중단 포함
27	SC	_SC	장애물 회피를 위한 안전 거리 , 증분
28	D2	_DN	공구의 두 번째 에지의 D 번호 , 프로그래밍되지 않은 경우 ⇒ D+1

번호	Param Mask	Param intern	설명
29		_GMODE	형상 모드 (프로그래밍된 형상 데이터 평가)
			UNITS: 예비용
			TENS: 예비용
			HUNDREDS: 가공 선택 / 시작점 계산만
			0 = 일반 가공 (호환 모드는 필요 없음) 1 = 일반 가공 2 = 시작점 계산 - 가공 안 함 (ShopMill/ShopTurn 의 호출에만 해당)
			THOUSANDS: 한계
			0 = 아니오 1 = 예
			TEN THOUSANDS: 한계 1 X 입력
			0 = 아니오 1 = 예
			HUNDRED THOUSANDS: 한계 2 X 입력
			0 = 아니오 1 = 예
			MILLION: 한계 1 Z 입력
			0 = 아니오 1 = 예
			TEN MILLION: 한계 2 Z 입력
			0 = 아니오 1 = 예
30		_DMODE	표시 모드
			UNITS: 가공 평면 G17/18/19
			0 = 호환성 , 싸이클 호출 이전에 유효한 평면은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)
			TENS: 테크놀로지 모드
			1 = 형상 절삭 2 = 형상 홈 가공 3 = 홈 선삭
			HUNDREDS: 잔삭 가공
			0 = 아니오 1 = 예

번호	Param Mask	Param intern	설명
31		_AMODE	대체 모드
			UNITS: 절입 선택
			0 = 형상과 평행한 선삭 가공을 위한 DX 및 DZ 절입 1 = D 절입
			TENS: 절입 방식
			0 = 가변 절삭 깊이 (90 ... 100 %) 1 = 일정 절삭 깊이
			HUNDREDS: 절삭 분할
			0 = 균일 1 = 에지에 정렬
			THOUSANDS: 형상 공차 U1 선택 , 이중 정삭
			0 = 아니오 1 = 예
			TEN THOUSANDS: 소재 선택 업데이트
			0 = 아니오 1 = 예
			HUNDRED THOUSANDS: 소재 XD 에 대한 공차 선택
			0 = 절대 , 직경에서 가로 축의 값 1 = 증분 , 반경에서 가로 축의 값
			MILLION: 소재 ZD 에 대한 공차 선택
			0 = 절대 1 = 증분
			TEN MILLION: 한계 2 XB 선택
			0 = 절대 , 직경에서 가로 축의 값 1 = 증분 , 반경에서 가로 축의 값
			HUNDRED MILLION: 한계 2 ZB 선택
			0 = 절대 1 = 증분

16.1.37 스위블 - CYCLE800

프로그래밍

```
CYCLE800(INT _FR, STRING[32] _TC, INT _ST, INT _MODE, REAL _X0, REAL
_Y0, REAL _Z0, REAL _A, REAL _B, REAL _C, REAL _X1, REAL _Y1, REAL
_Z1, INT _DIR, REAL _FR_I , INT _DMODE)
```

파라미터

번호	Param Mask	Param intern	설명
1		_FR	후퇴 모드 :
			0 = 후퇴 없음 1 = 후퇴 기계 축 Z 2 = 후퇴 기계 축 Z 다음에는 XY 3 = 예비용 4 = 공구 방향으로 최대 후퇴 5 = 공구 방향으로 증분 후퇴
2		_TC	스위블 데이터 레코드의 이름 : "" "" (이름 없음) 스위블 데이터 레코드가 단 하나인 경우 "0" 스위블 데이터 레코드 선택 취소 (스위블 프레임 삭제)
3		_ST	상태 변환
			UNITS:
			0 = 신규 , 스위블 레벨이 삭제되고 현재 파라미터를 사용하여 다시 계산됩니다 . 1 = 추가 , 스위블 레벨이 활성 스위블 레벨에 추가됩니다 .
			TENS: 공구 팁 교체 예 / 아니오 (IBN SWIVEL 기능이 설정된 경우에만 활성 상태)
			0 = 공구 팁을 교체하지 않음 1 = 공구 팁 교체 (TRAORI)
			HUNDREDS: 공구 접근 / 정렬 (공구 스위블 입력 마스크에 기능이 표시됨)
			0 = 공구에 접근하지 않음 1 = 공구에 접근 (반경 밀링이 바람직함) 2 = 선삭 공구 정렬 (IBN 스위블 밀링을 위해 B 축 좌표계를 설정한 경우) 3 = 밀링 공구 정렬 (IBN 스위블 밀링을 위해 B 축 좌표계를 설정한 경우) 9 = 예비용
			THOUSANDS: 내부 "JOG 스위블 " 파라미터
			TEN THOUSANDS: 방향 파라미터 _DIR 참조
			0 = 스위블 " 예 " 1 = 스위블 " 아니오 ", " 마이너스 " 방향 [3)] 2 = 스위블 " 아니오 ", " 플러스 " 방향 [3)]
			HUNDRED THOUSANDS: 방향 파라미터 _DIR 참조
			0 = 호환성 1 = 방향 선택 " 마이너스 " 최적화 [4)] 2 = 방향 선택 " 플러스 " 최적화 [4)]

번호	Param Mask	Param intern	설명
4		_MODE [5]	스위블 모드 : 스위블 각도 및 스위블 순서 평가 (비트 코딩)
			비트 : 7 6
			0 0: 축에 대한 스위블 각도 -> 파라미터 _A, _B, _C 참조 0 1: 입체각 -> 파라미터 _A, _B 참조 [1] 1 0: 투사각 -> 파라미터 _A, _B, _C 참조 [1] 1 1: 다이렉트 로터리 축 스위블 모드 -> 파라미터 _A, _B 참조 [1]
			비트 : 5 4 3 2 1 0 (입체각에는 해당되지 않음)
			x x x x 0 1 X 축 기준으로 첫 번째 회전 _A x x x x 1 0 1. Y 축 기준으로 회전 _A x x x x 1 1 Z 축 기준으로 첫 번째 회전 _A x x 0 1 x x X 축 기준으로 두 번째 회전 _B x x 1 0 x x Y 축 기준으로 두 번째 회전 _B x x 1 1 x x Z 축 기준으로 두 번째 회전 _B 0 1 x x x x X 축 기준으로 세 번째 회전 _C 1 0 x x x x Y 축 기준으로 세 번째 회전 _C 1 1 x x x x Z 축 기준으로 세 번째 회전 _C
5	X0	_X0	회전하기 전의 원점 X
6	Y0	_Y0	회전하기 전의 원점 Y
7	Z0	_Z0	회전하기 전의 원점 Z
8	X(A)	_A	1. _MODE 파라미터 설정에 따른 회전
9	Y(B)	_B	2. _MODE 파라미터 설정에 따른 회전
10	Z(C)	_C	3. _MODE 파라미터 설정에 따른 회전
11	X1	_X1	회전 후의 원점 X
12	Y1	_Y1	회전 후의 원점 Y
13	Z1	_Z1	회전 후의 원점 Z
14	- 또는 +	_DIR	로터리 축의 이송 초기화 (기본값 = -1!): -1 = 로터리 축 1 또는 2 의 보다 작은 값에 포지셔닝 [2] +1 = 로터리 축 1 또는 2 의 보다 큰 값에 포지셔닝 [2] 0 = 스위블하지 않음 (스위블 프레임만 계산) [1] [3]
15	FR	_FR_I	공구 방향 증분에서 후퇴 값 (inc)
16		_DMODE	표시 모드
			UNITS: 가공 평면 G17/G18/G19
			0 = 호환성 , 싸이클 호출 이전에 유효한 레벨은 계속 활성 상태 1 = G17 (싸이클에서만 활성 상태) 2 = G18 (싸이클에서만 활성 상태) 3 = G19 (싸이클에서만 활성 상태)

주

다음 전송 파라미터가 간접적으로 (파라미터로서) 프로그래밍된 경우에는 입력 마스크가 리셋되지 않습니다 . _FR, _ST, _TC, _MODE, _DIR

1) IBN SWIVEL 에서 기능이 설정된 경우에 선택할 수 있습니다 .

2) 로터리 축 1 또는 2 에 대한 방향 기준이 IBN SWIVEL 에 설정된 경우에 선택할 수 있습니다 .

방향 기준이 " 아니오 " 이면 선택 필드가 없습니다 .

3) 스위블 선택 " 아니오 " 가 회색으로 비활성화될 수 있습니다 . SD 55221 Bit 0

스위블 " 아니오 ", " 마이너스 " 방향은 _DIR = 0 및 _ST TEN THOUSANDS = 1 에 해당합니다 .

스위블 " 아니오 ", " 플러스 " 방향은 _DIR = 0 및 _ST TEN THOUSANDS = 2 에 해당합니다 .

4) 방향 기준이 있는 로터리 축이 극점 위치에 있는 경우에도 회전 축 1 또는 2 의 방향 선택이 이루어집니다 (위치 값은 0 과 같음).

5) 코딩 예제 : 축 기준 회전 , 회전 순서 ZYX

이진 : 00011011 십진 : 27

축 식별자 XYZ 는 NC 채널의 기하 축에 해당합니다 . XYZ 축을 기준으로 하는 개별 회전은 허용됩니다. 예: CYCLE800의 1회 호출에서는 ZXZ 축을 기준으로 하는 회전 순서가 허용되지 않습니다 .

16.1.38 고속 가공 설정 - CYCLE832

프로그래밍

```
CYCLE832(_TOL, _TOLM, _V832)
```

주

CYCLE832를 사용하려면 기계를 스타트업할 때 필요한 최적화 작업을 해야 합니다. 여기에는 가공 공정에 관련된 축의 최적화 및 NCU 설정 (사전 제어 , 저크 제한 등) 이 포함됩니다 .

파라미터

번호	Param Mask	Param intern	설명
1	TOL	_TOL	공차
2		_TOLM	테크놀로지
			UNITS:
			0 = 선택 취소 1 = 정삭 2 = 반 정삭 3 = 황삭
3		_V832	버전 CYCLE832
			UNITS:
			0 = 소프트웨어 버전 7.5 까지 1 = HMI sl 소프트웨어 버전 2.6 부터

주

CYCLE832 의 선택을 취소할 때는 공차 파라미터를 0 으로 이송해야 합니다 .

예 : CYCLE832(0,0,1)

CYCLE832 의 선택을 취소하는 경우에는 CYCLE832() 구문도 허용됩니다 .

16.1.39 고속 절삭 (HSC) - CYCLE_HSC

프로그래밍

```
CYCLE_HSC (_Mode, _TOL, _RTOL)
```

파라미터

번호	Param Mask	Param intern	설명	
1		_MODE	가공 유형 (기술)	
			가공 유형 파라미터는 일반 문자열 형태로 CYCLE_HSC 로 전송됩니다 (대소문자 표기 허용).	
			"FINISH" =	정삭
			"SEMIFINISH" =	중삭
			"ROUGH" =	황삭
			"OFF" 또는 "DESELECT" =	CYCLE_HSC 선택 취소
2		_TOL	공차	
			공차는 기하 축의 축 공차에 해당합니다 .	
3		_RTOL	로터리 축 공차	
			로터리 축 공차는 OTOL=_RTOL 명령을 사용하여 NC 로 전송됩니다 . OTOL 은 방향 전환이 활성 상태인 경우에만 유효합니다 (TRAORI).	

주

CYCLE_HSC 는 내부적으로 CYCLE832 (고속 설정) 를 호출합니다 .

주

CYCLE_HSC 를 선택 취소할 때는 추가 파라미터를 지정할 필요가 없습니다 .

예 : CYCLE_HSC("OFF")

예

프로그램 코드	코멘트
G710	
TRAORI	
CYCLE_HSC("FINISH",0.01,3)	; 기하 축의 0.01 mm 공차 및 3 도 로터리 축 공차를 적용한 가공 유형 FINISH

테이블 17

17.1 작업

범례 :

1) 조작의 효과 :
   m 모달
   s 넌모달

2) 세부적인 조작 설명이 포함된 참고 자료 :
   PGsl 프로그래밍 매뉴얼 , 기본 사항
   PGAsl 프로그래밍 매뉴얼 , 작업 계획
   BNMsl 측정 싸이클 프로그래밍 매뉴얼
   BHDsl 조작 매뉴얼 , 선삭
   BHFsl 조작 매뉴얼 , 밀링
   FB1 ( ) 기능 매뉴얼 , 기본 기능 (해당 기능 설명의 약어는 괄호 안에 표시됨)
   FB2 ( ) 기능 매뉴얼 , 확장 기능 (해당 기능 설명의 약어는 괄호 안에 표시됨)
   FB3 ( ) 기능 매뉴얼 , 특수 기능 (해당 기능 설명의 약어는 괄호 안에 표시됨)
   FBSIsl 기능 매뉴얼 , 통합 안전
   FBSY 기능 매뉴얼 , 동기 동작
   FBW 기능 매뉴얼 , 공구 관리

3) 프로그램 시작 시 디폴트 설정 (별도로 프로그램하지 않은 경우 시스템의 공장 출하 설정)

조작 연산	의미	W 1)	설명 참조 2)
:	NC 메인 블록 번호 , 점프 레이블 종료 , 연결 연산자		*PGAsl* 산술 함수 [쪽 64]
*	곱셈 연산자		*PGAsl* 산술 함수 [쪽 64]
+	덧셈 연산자		*PGAsl* 산술 함수 [쪽 64]
-	뺄셈 연산자		*PGAsl* 산술 함수 [쪽 64]
<	비교 연산자 , 보다 작은		*PGAsl* 산술 함수 [쪽 64]
<<	문자열 연결 연산자		*PGAsl* 산술 함수 [쪽 64]
<=	비교 연산자 , 작거나 같음		*PGAsl* 산술 함수 [쪽 64]
=	지정 연산자		*PGAsl* 산술 함수 [쪽 64]

조작 연산	의미	W [1)]	설명 참조 [2)]
>=	비교 연산자 , 크거나 같음		*PGAs/* 산술 함수 [쪽 64]
/	나눗셈 연산자		*PGAs/* 산술 함수 [쪽 64]
/0 … … /7	블록을 건너뜁니다 (1 차 스킵 레벨). 블록을 건너 뜁니다 (8 차 스킵 레벨).		*PGs/*
A	축 이름	m/s	*PGAs/* 공구 방향 프로그램 (A..., B..., C..., LEAD, TILT) [쪽 327]
A2	공구 오리엔테이션: RPY 또는 Euler 각도	s	*PGAs/* 공구 방향 프로그램 (A..., B..., C..., LEAD, TILT) [쪽 327]
A3	공구 오리엔테이션: 방향/표면 법선 벡터 콤포넌트	s	*PGAs/* 공구 방향 프로그램 (A..., B..., C..., LEAD, TILT) [쪽 327]
A4	공구 오리엔테이션 : 블록 시작을 위한 표면 법선 벡터	s	*PGAs/* 페이스 밀링 (3D 밀링 A4, B4, C4, A5, B5, C5) [쪽 335]
A5	공구 오리엔테이션 : 블록 종료를 위한 표면 법선 벡터	s	*PGAs/* 페이스 밀링 (3D 밀링 A4, B4, C4, A5, B5, C5) [쪽 335]
ABS	절대치 (양)		*PGAs/* 산술 함수 [쪽 64]
AC	좌표 / 위치의 절대치	s	*PGs/*
ACC	현재 축 가속의 효과	m	*PGs/*
ACCLIMA	현재 최대 축 가속의 효과	m	*PGs/*
ACN	로터리 축의 절대 치수 , 마이너스 방향으로 접근	s	*PGs/*
ACOS	역코사인 (삼각형 . 함수)		*PGAs/* 산술 함수 [쪽 64]
ACP	로터리 축의 절대 치수 , 양의 방향으로 접근	s	*PGs/*
ACTBLOCNO	'현재 블록 표시 억제' (DISPLOF) 가 활성 상태인 경우에도 알람 블록의 현재 블록 출력		*PGAs/* 현재 블록 디스플레이 억제 (DISPLOF, DISPLON, ACTBLOCNO) [쪽 174]
ADDFRAME	측정된 프레임 포함 및 활성화 가능		*PGAs/, FB1(K2)* 공간의 세 개 측정 지점에서 프레임 계산 (MEAFRAME) [쪽 302]
ADIS	경로 코드 G1, G2, G3, ... 의 라운딩 공차	m	*PGs/*

조작 연산	의미	W [1]	설명 참조 [2]
ADISPOS	급 이송 G0 의 라운딩 공차	m	*PGs/*
ADISPOSA	IPOBRKA 에서 공차 범위 크기	m	*PGAs/* 프로그래밍된 이동 종료 기준 (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) [쪽 278]
ALF	LIFTFAST 각도	m	*PGAs/* 형상으로부터 빠른 후퇴 (SETINT LIFTFAST, ALF) [쪽 125]
AMIRROR	프로그래밍 미러링	s	*PGs/*
AND	Logical AND		*PGAs/* 비교 및 논리 연산자 [쪽 67]
ANG	형상 각도	s	*PGs/*
AP	극점 각도	m/s	*PGs/*
APR	읽기 / 표시 액세스 방지		*PGAs/* 속성 : 사용 권한 (APR, APW, APRP, APWP, APRB, APWB) [쪽 41]
APRB	읽기 액세스 권한 , OPI		*PGAs/* 속성 : 사용 권한 (APR, APW, APRP, APWP, APRB, APWB) [쪽 41]
APRP	읽기 액세스 권한 , 가공 프로그램		*PGAs/* 속성 : 사용 권한 (APR, APW, APRP, APWP, APRB, APWB) [쪽 41]
APW	쓰기 액세스 방지		*PGAs/* 속성 : 사용 권한 (APR, APW, APRP, APWP, APRB, APWB) [쪽 41]
APWB	쓰기 액세스 권한 , OPI		*PGAs/* 속성 : 사용 권한 (APR, APW, APRP, APWP, APRB, APWB) [쪽 41]
APWP	쓰기 액세스 권한 , 가공 프로그램		*PGAs/* 속성 : 사용 권한 (APR, APW, APRP, APWP, APRB, APWB) [쪽 41]
APX	지정된 언어 요소를 실행하기 위한 사용 권한 정의		*PGAs/* 시스템 변수, 사용자 변수 및 NC 언어 명령어의 재 정의 (REDEF) [쪽 31]
AR	틈 각도	m/s	*PGs/*
AROT	프로그래밍 회전	s	*PGs/*
AROTS	입체각 프로그래밍 프레임 회전	s	*PGs/*

조작 연산	의미	W [1]	설명 참조 [2]
SL	매크로 정의		*PGAs/* 매크로 방법 (DEFINE ... AS) [쪽 209]
ASCALE	프로그래밍 배율	s	*PGs/*
ASIN	산술 함수 , 원호 사인		*PGAs/* 산술 함수 [쪽 64]
ASPLINE	아키마 스플라인	m	*PGAs/* 스플라인 보간 (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [쪽 240]
ATAN2	역탄젠트 2		*PGAs/* 산술 함수 [쪽 64]
ATOL	컴프레서 기능을 위한 축 관련 공차 , 오리엔테이션 스무딩 및 스무딩 종류		*PGAs/* 프로그램 가능한 형상 / 오리엔테이션 공차 (CTOL, OTOL, ATOL) [쪽 492]
ATRANS	프로그램 가능한 추가 변환	s	*PGs/*
AX	변수 축 식별자	m/s	*PGAs/* 축 기능 (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) [쪽 667]
AXCTSWE	축 컨테이너 회전		*PGAs/* 축 컨테이너 (AXCTSWE, AXCTSWED, AXCTSWEC) [쪽 675]
AXCTSWEC	축 컨테이너 회전에 대한 인에이블 취소		*PGAs/* 축 컨테이너 (AXCTSWE, AXCTSWED, AXCTSWEC) [쪽 675]
AXCTSWED	축 컨테이너 회전 (스타트업을 위한 명령 변용)		*PGAs/* 축 컨테이너 (AXCTSWE, AXCTSWED, AXCTSWEC) [쪽 675]
AXIS	축 이름 , 축 어드레스		*PGAs/* 사용자 변수 정의 (DEF) [쪽 25]
AXNAME	입력 문자열을 축 이름으로 변환		*PGAs/* 축 기능 (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) [쪽 667]
AXSTRING	문자열 스핀들 번호 변환		*PGAs/* 축 기능 (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) [쪽 667]
AXTOCHAN	특정 채널의 축 요청 . NC 프로그램과 동기 동작에서 가능 .		*PGAs/* 다른 채널로 축 이송 (AXTOCHAN) [쪽 135]
AXTOSPI	축 이름을 스핀들 인덱스로 변환		*PGAs/* 축 기능 (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) [쪽 667]
B	축 이름	m/s	*PGAs/* 공구 방향 프로그램 (A..., B..., C..., LEAD, TILT) [쪽 327]

조작 연산	의미	W [1]	설명 참조 [2]
B2	공구 오리엔테이션: RPY 또는 Euler 각도	s	*PGAs/* 공구 방향 프로그램 (A..., B..., C..., LEAD, TILT) [쪽 327]
B3	공구 오리엔테이션: 방향/표면 법선 벡터 콤포넌트	s	*PGAs/* 공구 방향 프로그램 (A..., B..., C..., LEAD, TILT) [쪽 327]
B4	공구 오리엔테이션 : 블록 시작을 위한 표면 법선 벡터	s	*PGAs/* 페이스 밀링 (3D 밀링 A4, B4, C4, A5, B5, C5) [쪽 335]
B5	공구 오리엔테이션 : 블록 종료를 위한 표면 법선 벡터	s	*PGAs/* 페이스 밀링 (3D 밀링 A4, B4, C4, A5, B5, C5) [쪽 335]
B_AND	Bit AND		*PGAs/* 비교 및 논리 연산자 [쪽 67]
B_OR	Bit OR		*PGAs/* 비교 및 논리 연산자 [쪽 67]
B_NOT	Bit 부정		*PGAs/* 비교 및 논리 연산자 [쪽 67]
B_XOR	Bit 익스클루시브 OR		*PGAs/* 비교 및 논리 연산자 [쪽 67]
BAUTO	다음의 세 점을 사용한 첫 번째 스플라인 섹션 정의	m	*PGAs/* 스플라인 보간 (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [쪽 240]
BLOCK	키워드 TO 와 함께 간접 서브프로그램 호출 시 처리할 프로그램 부분 정의		*PGAs/* 호출 프로그램 일부를 지정하여 서브프로그램 간접 호출 (CALL BLOCK ... TO ...) [쪽 195]
BLSYNC	인터럽트 루틴의 처리는 다음 블록 변경을 시작할 때만 가능		*PGAs/* 인터럽트 루틴 지정 및 시작 (SETINT, PRIO, BLSYNC) [쪽 121]
BNAT [3]	첫 번째 스플라인 블록으로 자연적인 트렌지션	m	*PGAs/* 스플라인 보간 (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [쪽 240]
BOOL	데이터 유형 : 부울 값은 TRUE/FALSE 또는 1/0		*PGAs/* 사용자 변수 정의 (DEF) [쪽 25]
BOUND	값이 정의된 범위 내에 있는지 여부를 테스트합니다 . 값이 같으면 테스트 값을 반환합니다 .		*PGAs/* 변수 최소 , 최대 및 범위 (MINVAL, MAXVAL 및 BOUND) [쪽 71]
BRISK [3]	고속 비스무딩 경로 가속	m	*PGs/*
BRISKA	프로그램한 축에서 brisk 경로 가속 시작		*PGs/*
BSPLINE	B 스플라인	m	*PGAs/* 스플라인 보간 (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [쪽 240]

조작 연산	의미	W [1]	설명 참조 [2]
BTAN	첫 번째 스플라인 블록으로 접선 트렌지션	m	*PGAs/* 스플라인 보간 (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [쪽 240]
C	축 이름	m/s	*PGAs/* 공구 방향 프로그램 (A..., B..., C..., LEAD, TILT) [쪽 327]
C2	공구 오리엔테이션: RPY 또는 Euler 각도	s	*PGAs/* 공구 방향 프로그램 (A..., B..., C..., LEAD, TILT) [쪽 327]
C3	공구 오리엔테이션: 방향/표면 법선 벡터 콤포넌트	s	*PGAs/* 공구 방향 프로그램 (A..., B..., C..., LEAD, TILT) [쪽 327]
C4	공구 오리엔테이션 : 블록 시작을 위한 표면 법선 벡터	s	*PGAs/* 페이스 밀링 (3D 밀링 A4, B4, C4, A5, B5, C5) [쪽 335]
C5	공구 오리엔테이션 : 블록 종료를 위한 표면 법선 벡터	s	*PGAs/* 페이스 밀링 (3D 밀링 A4, B4, C4, A5, B5, C5) [쪽 335]
CAC	절대 위치 접근		*PGAs/* 코드 위치 복귀 (CAC, CIC, CDC, CACP, CACN) [쪽 239]
CACN	마이너스 방향으로 테이블에 열거된 절대치로 접근		*PGAs/* 코드 위치 복귀 (CAC, CIC, CDC, CACP, CACN) [쪽 239]
CACP	양의 방향으로 테이블에 열거된 절대치로 접근		*PGAs/* 코드 위치 복귀 (CAC, CIC, CDC, CACP, CACN) [쪽 239]
CALCDAT	3 지점 또는 4 지점을 사용하여 원호의 반경 및 중심점 계산		*PGAs/* 원 데이터 계산 (CALCDAT) [쪽 725]
CALCPOSI	보호 영역 위반 , 작업 영역 제한 및 소프트웨어 제한 점검		*PGAs/* 보호 영역 위반 , 작업 영역 제한 및 소프트웨어 제한 점검 (CALCPOSI) [쪽 231]
CALL	간접 서브루틴 호출		*PGAs/* 간접 서브프로그램 호출 (CALL) [쪽 194]
CALLPATH	서브루틴 호출에서 프로그래밍 탐색 경로		*PGAs/* 서브프로그램 호출에 대한 확장 검색 경로 (CALLPATH) [쪽 199]
CANCEL	모달 동기 동작 취소		*PGAs/* 동기화 동작 삭제 (CANCEL) [쪽 635]
CASE	조건부 프로그램 분기		*PGAs/* 프로그램 분기 (CASE ... OF ... DEFAULT ...) [쪽 96]
CDC	위치 직접 이동		*PGAs/* 코드 위치 복귀 (CAC, CIC, CDC, CACP, CACN) [쪽 239]
CDOF [3]	충돌 탐지 OFF	m	*PGs/*

조작 연산	의미	W [1]	설명 참조 [2]
CDOF2	충돌 탐지 OFF, 3D 원주 밀링의 경우	m	*PGsl*
CDON	충돌 탐지 ON	m	*PGsl*
CFC [3]	형상에서의 일정한 이송 속도	m	*PGsl*
CFIN	외부 반경이 아닌 내부 반경에서만 일정한 이송 속도	m	*PGsl*
CFINE	미세 옵셋을 프레임 변수에 지정		*PGAsl* 일반 및 미세조정 옵셋 (CFINE, CTRANS) [쪽 298]
CFTCP	공구 중심점의 일정한 이송 속도 (중심점 경로)	m	*PGsl*
CHAN	데이터의 유효 범위 지정		*PGAsl* 사용자 변수 정의 (DEF) [쪽 25]
CHANDATA	채널 데이터 액세스용 채널 번호 설정		*PGAsl* 작업 메모리 (CHANDATA, COMPLETE, INITIAL) [쪽 218]
CHAR	데이터 유형 : ASCII 문자		*PGAsl* 사용자 변수 정의 (DEF) [쪽 25]
CHECKSUM	고정 길이 문자열로 배열을 통한 체크섬 구성		*PGAsl* 배열을 이용한 체크섬 계산 (CHECKSUM) [쪽 154]
CHF	Chamfer: 값 = 모따기 길이	s	*PGsl*
CHKDM	메거진 내의 고유성 검사		*FBW*
CHKDNO	독자적 D 번호를 점검		*PGAsl* D 번호의 자유 할당 : D 번호 확인 (CHKDNO) [쪽 433]
CHR	모따기 : 값 = 이동 방향에서 모따기 길이		*PGsl*
CIC	증분에 의한 위치 접근		*PGAsl* 코드 위치 복귀 (CAC, CIC, CDC, CACP, CACN) [쪽 239]
CIP	중간점을 통한 원호 보간	m	*PGsl*
CLEARM	채널 코디네이션을 위해 1 개 / 여러 개 마커를 리셋		*PGAsl* 프로그램 코디네이션 (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) [쪽 114]
CLRINT	인터럽트 선택 취소 :		*PGAsl* 인터럽트 루틴 할당 삭제 (CLRINT) [쪽 124]
CMIRROR	좌표 축에서 미러링		*PGAsl* 산술 함수 [쪽 64]
COARSEA	" 일반 정위치 정지 " 에 도달하면 이동이 끝남	m	*PGAsl* 프로그래밍된 이동 종료 기준 (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) [쪽 278]

조작 연산	의미	W [1]	설명 참조 [2]
COMPCAD	컴프레서 ON: CAD 프로그램 실행을 위한 최적의 표면 품질	m	*PGAs/* NC 블록 압축 (COMPON, COMPCURV, COMPCAD, COMPOF) [쪽 253]
COMPCURV	컴프레서 ON: 곡률이 일정한 폴리노미얼	m	*PGAs/* NC 블록 압축 (COMPON, COMPCURV, COMPCAD, COMPOF) [쪽 253]
COMPLETE	데이터 읽기 및 쓰기를 위한 제어 명령		PGAsl 작업 메모리 (CHANDATA, COMPLETE, INITIAL) [쪽 218]
COMPOF [3]	컴프레서 OFF	m	*PGAs/* NC 블록 압축 (COMPON, COMPCURV, COMPCAD, COMPOF) [쪽 253]
COMPON	컴프레서 ON		*PGAs/* NC 블록 압축 (COMPON, COMPCURV, COMPCAD, COMPOF) [쪽 253]
CONTDCON	테이블의 형상 디코딩 ON		*PGAs/* 코드화 형상 테이블의 생성 (CONTDCON) [쪽 718]
CONTPRON	기준 전처리 활성화		*PGAs/* 형상 테이블 생성 (CONTPRON) [쪽 712]
CORROF	활성 오버레이 이동 선택이 모두 해제됨		*PGs/*
COS	코사인 (삼각형 . 함수)		*PGAs/* 산술 함수 [쪽 64]
COUPDEF	ELG 그룹 / 동기 스핀들 그룹 정의		*PGAs/* 동기식 스핀들 : 프로그래밍 (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) [쪽 535]
COUPDEL	ELG 그룹 삭제		*PGAs/* 동기식 스핀들 : 프로그래밍 (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) [쪽 535]
COUPOF	ELG 그룹 / 동기 스핀들 페어 ON		*PGAs/* 동기식 스핀들 : 프로그래밍 (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) [쪽 535]
COUPOFS	추종 스핀들 정지로 ELG 그룹 / 동기 스핀들 페어 비활성화		*PGAs/* 동기식 스핀들 : 프로그래밍 (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) [쪽 535]
COUPON	ELG 그룹 / 동기 스핀들 페어 ON		*PGAs/* 동기식 스핀들 : 프로그래밍 (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) [쪽 535]

조작 연산	의미	W [1]	설명 참조 [2]
COUPONC	이전 프로그래밍으로 ELG 그룹 / 동기 스핀들 페어의 활성화 이송		*PGAs/* 동기식 스핀들 : 프로그래밍 (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) [쪽 535]
COUPRES	ELG 그룹 리셋		*PGAs/* 동기식 스핀들 : 프로그래밍 (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) [쪽 535]
CP	경로 모션	m	*PGAs/* 직교 PTP 이송 [쪽 380]
CPRECOF [3]	프로그램 가능한 형상 정확도 OFF	m	*PGs/*
CPRECON	프로그램 가능한 형상 정확도 ON	m	*PGs/*
CPROT	채널 지정 보호 영역 ON/OFF		*PGAs/* 보호 영역 활성화 / 비활성화 (CPROT, NPROT) [쪽 227]
CPROTDEF	채널 지정 보호 영역 정의		*PGAs/* 보호 영역의 정의 (CPROTDEF, NPROTDEF) [쪽 223]
CR	원호 반경	s	*PGs/*
CROT	현재 좌표계의 회전		*PGAs/* 산술 함수 [쪽 64]
CROTS	입체각 프로그래밍 프레임 회전 (지정 축에서 회전)	s	*PGs/*
CRPL	임의의 면에서 프레임 회전		*FB1(K2)*
CSCALE	여러 축의 배율 계수		*PGAs/* 산술 함수 [쪽 64]
CSPLINE	입체 스플라인	m	*PGAs/* 스플라인 보간 (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [쪽 240]
CT	접선 트렌지션에 의한 원호	m	*PGs/*
CTAB	커브 테이블에서 리딩 축 위치에 따라 추종 축 위치를 정의		*PGAs/* 커브 테이블 값 읽기 (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [쪽 513]
CTABDEF	테이블 정의 ON		*PGAs/* 커브 테이블 정의 (CTABDEF, CATBEND) [쪽 502]
CTABDEL	커브 테이블 삭제		*PGAs/* 커브 테이블 삭제 (CTABDEL) [쪽 509]
CTABEND	테이블 정의 OFF		*PGAs/* 커브 테이블 정의 (CTABDEF, CATBEND) [쪽 502]

조작 연산	의미	W [1]	설명 참조 [2]
CTABEXISTS	번호 n 에 해당하는 커브 테이블 점검		*PGAs/* 커브 테이블의 존재 검사 (CTABEXISTS) [쪽 508]
CTABFNO	메모리에서 여전히 설정 가능한 커브 테이블의 개수		*PGAs/* 커브 테이블 : 리소스 사용 확인 (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [쪽 519]
CTABFPOL	메모리에서 여전히 설정 가능한 폴리노미얼 개수		*PGAs/* 커브 테이블 : 리소스 사용 확인 (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [쪽 519]
CTABFSEG	메모리에서 여전히 설정 가능한 커브 세그먼트 개수		*PGAs/* 커브 테이블 : 리소스 사용 확인 (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [쪽 519]
CTABID	n 번째 커브 테이블의 테이블 번호 반환		*PGAs/* 커브 테이블 : 테이블 속성 결정 (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD) [쪽 511]
CTABINV	커브 테이블에서 추종 축 위치에 따라 리딩 축 위치를 정의		*PGAs/* 커브 테이블 값 읽기 (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [쪽 513]
CTABISLOCK	번호 n 에 해당하는 커브 테이블의 잠금 상태 반환		*PGAs/* 커브 테이블 : 테이블 속성 결정 (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD) [쪽 511]
CTABLOCK	삭제 및 덮어쓰기 , 잠금		*PGAs/* 삭제 및 덮어쓰기 방지를 위한 커브 테이블 잠금 (CTABLOCK, CTABUNLOCK) [쪽 510]
CTABMEMTYP	커브 테이블 번호 n 이 생성된 메모리 반환		*PGAs/* 커브 테이블 : 테이블 속성 결정 (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD) [쪽 511]
CTABMPOL	메모리에서 여전히 설정 가능한 폴리노미얼 최대 개수		*PGAs/* 커브 테이블 : 리소스 사용 확인 (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [쪽 519]
CTABMSEG	메모리에서 여전히 설정 가능한 커브 세그먼트의 최대 개수		*PGAs/* 커브 테이블 : 리소스 사용 확인 (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [쪽 519]
CTABNO	SRAM 또는 DRAM 에서 정의한 커브 테이블의 개수		*FB3(M3)*

조작 연산	의미	W [1]	설명 참조 [2]
CTABNOMEM	SRAM 또는 DRAM 에서 정의한 커브 테이블의 개수		*PGAs/* 커브 테이블 : 리소스 사용 확인 (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [쪽 519]
CTABPERIOD	커브 테이블 번호 n 의 테이블 주기성 반환		*PGAs/* 커브 테이블 : 테이블 속성 결정 (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD) [쪽 511]
CTABPOL	메모리에서 이미 사용한 폴리노미얼의 개수		*PGAs/* 커브 테이블 : 리소스 사용 확인 (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [쪽 519]
CTABPOLID	번호 n 에 해당하는 커브 테이블에서 사용한 커브 폴리노미얼의 개수		*PGAs/* 커브 테이블 : 리소스 사용 확인 (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [쪽 519]
CTABSEG	메모리에서 이미 사용한 커브 세그먼트의 개수		*PGAs/* 커브 테이블 : 리소스 사용 확인 (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [쪽 519]
CTABSEGID	번호 n 에 해당하는 커브 테이블에서 사용한 커브 세그먼트의 개수		*PGAs/* 커브 테이블 : 리소스 사용 확인 (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) [쪽 519]
CTABSEV	커브 테이블 분할에서 추종 축의 최종 값 반환		*PGAs/* 커브 테이블 값 읽기 (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [쪽 513]
CTABSSV	커브 테이블 분할에서 추종 축의 초기 값 반환		*PGAs/* 커브 테이블 값 읽기 (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [쪽 513]
CTABTEP	커브 테이블 끝 부분에 있는 리딩 축의 값 반환		*PGAs/* 커브 테이블 값 읽기 (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [쪽 513]
CTABTEV	커브 테이블 끝 부분에 있는 추종 축의 값 반환		*PGAs/* 커브 테이블 값 읽기 (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [쪽 513]
CTABTMAX	커브 테이블에서 추종 축의 최대값 반환		*PGAs/* 커브 테이블 값 읽기 (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [쪽 513]

조작 연산	의미	W [1]	설명 참조 [2]
CTABTMIN	커브 테이블에서 추종 축의 최소값 반환		*PGAs/* 커브 테이블 값 읽기 (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [쪽 513]
CTABTSP	커브 테이블 시작 부분에 있는 리딩 축의 값 반환		*PGAs/* 커브 테이블 값 읽기 (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [쪽 513]
CTABTSV	커브 테이블 시작 부분에 있는 추종 축의 값 반환		*PGAs/* 커브 테이블 값 읽기 (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) [쪽 513]
CTABUNLOCK	삭제 및 덮어쓰기 취소 잠금		*PGAs/* 삭제 및 덮어쓰기 방지를 위한 커브 테이블 잠금 (CTABLOCK, CTABUNLOCK) [쪽 510]
CTOL	컴프레서 기능을 위한 형상 공차 , 오리엔테이션 스무딩 및 스무딩 종류		*PGAs/* 프로그램 가능한 형상 / 오리엔테이션 공차 (CTOL, OTOL, ATOL) [쪽 492]
CTRANS	여러 축에 대한 워크 옵셋		*PGAs/* 일반 및 미세조정 옵셋 (CFINE, CTRANS) [쪽 298]
CUT2D [3]	2D 공구 옵셋	m	*PGs/*
CUT2DF	2D 공구 옵셋 . 공구 옵셋은 현재 프레임 (경사 면) 과 관련하여 적용됨	m	*PGs/*
CUT3DC	3D 공구 옵셋 원주 밀링	m	*PGAs/* 3D 공구 옵셋 활성 (CUT3DC..., CUT3DF...) [쪽 413]
CUT3DCC	한계 표면으로 3D 공구 옵셋 원주 밀링	m	*PGAs/* 3D 공구 옵셋 한계 표면 고려(CUT3DCC, CUT3DCCD) [쪽 423]
CUT3DCCD	차등 공구를 사용한 한계 표면으로 3D 공구 옵셋 원주 밀링	m	*PGAs/* 3D 공구 옵셋 한계 표면 고려(CUT3DCC, CUT3DCCD) [쪽 423]
CUT3DF	3D 공구 옵셋 페이스 밀링	m	*PGAs/* 3D 공구 옵셋 활성 (CUT3DC..., CUT3DF...) [쪽 413]
CUT3DFF	활성 프레임에 따라 일정한 공구 오리엔테이션의 3D 공구 옵셋 페이스 밀링	m	*PGAs/* 3D 공구 옵셋 활성 (CUT3DC..., CUT3DF...) [쪽 413]
CUT3DFS	활성 프레임과 관계 없이 일정한 공구 오리엔테이션의 3D 공구 옵셋 페이스 밀링	m	*PGAs/* 3D 공구 옵셋 활성 (CUT3DC..., CUT3DF...) [쪽 413]
CUTCONOF [3]	일정한 반경 보정 OFF	m	*PGs/*
CUTCONON	일정한 반경 보정 ON	m	*PGs/*

조작 연산	의미	W [1]	설명 참조 [2]
CUTMOD	" 공구에 대한 옵셋 데이터 수정 " 활성화		*PGAs/* 회전 가능한 공구에 대한 절삭 데이터 수정 (CUTMOD) [쪽 449]
CYCLE60	기술 싸이클 : 조각 싸이클		*PGAs/* 조각 싸이클 - CYCLE60 [쪽 773]
CYCLE61	기술 싸이클 : 페이스 밀링		*PGAs/* 페이스 밀링 - CYCLE61 [쪽 749]
CYCLE62	기술 싸이클 : 형상 호출		*PGAs/* 형상 호출 - CYCLE62 [쪽 776]
CYCLE63	기술 싸이클 : 형상 포켓 밀링		*PGAs/* 형상 포켓 밀링 - CYCLE63 [쪽 782]
CYCLE64	기술 싸이클 : 형상 포켓 황삭 - 드릴링		*PGAs/* 형상 포켓 사전 드릴링 - CYCLE64 [쪽 780]
CYCLE70	기술 싸이클 : 나사 밀링		*PGAs/* 나사 밀링 - CYCLE70 [쪽 771]
CYCLE72	기술 싸이클 : 경로 밀링		*PGAs/* 경로 밀링 - CYCLE72 [쪽 777]
CYCLE76	기술 싸이클 : 사각 스피곳 밀링		*PGAs/* 사각 스피곳 밀링 - CYCLE76 [쪽 756]
CYCLE77	기술 싸이클 : 원호 스피곳 밀링		*PGAs/* 원주 밀링 - CYCLE77 [쪽 758]
CYCLE78	기술 싸이클 : 드릴 나사 밀링		*PGAs/* 나사 밀링 - CYCLE78 [쪽 742]
CYCLE79	기술 싸이클 : 다각형 밀링		*PGAs/* 다각형 - CYCLE79 [쪽 760]
CYCLE81	기술 싸이클 : 드릴링 , 센터링		*PGAs/* 드릴링 , 센터링 - CYCLE81 [쪽 731]
CYCLE82	기술 싸이클 : 드릴링 , 카운터보링		*PGAs/* 드릴링 , 카운터보링 - CYCLE82 [쪽 732]
CYCLE83	기술 싸이클 : 심공 드릴링		*PGAs/* 심공 드릴링 - CYCLE83 [쪽 734]
CYCLE84	기술 싸이클 : 리지드 태핑		*PGAs/* 동기 태핑 - CYCLE84 [쪽 737]
CYCLE85	기술 싸이클 : 리밍		*PGAs/* 리밍 - CYCLE85 [쪽 733]
CYCLE86	기술 싸이클 : 보링		*PGAs/* 보링 - CYCLE86 [쪽 736]
CYCLE92	기술 싸이클 : 태핑		*PGAs/* 절단 - CYCLE92 [쪽 799]
CYCLE98	기술 싸이클 : 나사 체인		*PGAs/* 나사 체인 - CYCLE98 [쪽 796]

조작 연산	의미	W [1)]	설명 참조 [2)]
CYCLE99	기술 싸이클 : 나사 턴		*PGAs/* 나사 선삭 - CYCLE99 [쪽 793]
CYCLE800	기술 싸이클 : 스위블		*PGAs/* 스위블 - CYCLE800 [쪽 805]
CYCLE801	기술 싸이클 : 그리드 또는 프레임		*PGAs/* 그리드 또는 프레임 - CYCLE801 [쪽 747]
CYCLE802	기술 싸이클 : 모든 위치		*PGAs/* 자유 프로그래밍 위치 - CYCLE802 [쪽 744]
CYCLE832	기술 싸이클 : 고속 설정		*PGAs/* 고속 가공 설정 - CYCLE832 [쪽 808]
CYCLE840	기술 싸이클 : 보정 척을 사용한 태핑		*PGAs/* 태핑 척을 사용한 태핑 - CYCLE840 [쪽 740]
CYCLE899	기술 싸이클 : 개방형 홈 밀링		*PGAs/* 밀링 개방형 슬롯 - CYCLE899 [쪽 767]
CYCLE930	기술 싸이클 : 홈		*PGAs/* 홈 - CYCLE930 [쪽 787]
CYCLE940	기술 싸이클 : 언더컷 가공		*PGAs/* 언더컷 형상 - CYCLE940 [쪽 790]
CYCLE951	기술 싸이클 : 스톡 제거		*PGAs/* 선삭 - CYCLE951 [쪽 784]
CYCLE952	기술 싸이클 : 형상 홈 가공		*PGAs/* 형상 홈 가공 - CYCLE952 [쪽 801]
CYCLE_HSC	기술 싸이클 : 고속 절삭		*PGAs/* 고속 절삭 (HSC) - CYCLE_HSC [쪽 809]

조작	의미	W [1)]	설명 참조 [2)]
D	공구 옵셋 번호		*PGs/*
D0	D0 를 통해 공구 옵셋 해제		*PGs/*
DAC	절대치 넌모달 직경 프로그램	s	*PGs/*
DC	로터리 축의 절대 치수 . 위치에 직접 접근	s	*PGs/*
DEF	변수 정의		*PGAs/* 사용자 변수 정의 (DEF) [쪽 25]
DEFINE	매크로 정의 키워드		*PGAs/* 매크로 방법 (DEFINE ... AS) [쪽 209]
DEFAULT	CASE 분기에서 분기		*PGAs/* 프로그램 분기 (CASE ... OF ... DEFAULT ...) [쪽 96]
DELAYFSTON	지연 섹션 정지 시작 지정	m	*PGAs/* 프로그램 분기 (CASE ... OF ... DEFAULT ...) [쪽 96]

조작	의미	W [1]	설명 참조 [2]
DELAYFSTOF	지연 섹션 정지 끝 지정	m	*PGAsl* 조건에 따라 중단 가능한 프로그램 섹션 (DELAYFSTON, DELAYFSTOF) [쪽 471]
DELDL	추가 옵셋 삭제		*PGAsl* 추가 옵셋 삭제 (DELDL) [쪽 399]
DELDTG	이동할 거리 삭제		*PGAsl* 이동할 거리 삭제 (DELDTG) [쪽 581]
DELETE	지정된 파일을 삭제 경로 및 파일 식별자로 파일 이름 지정		*PGAsl* 파일 삭제 (DELETE) [쪽 143]
DELTOOLENV	공구 환경 설명 데이터 기록 삭제		*FB1(W1)*
DIACYCOFA	축 관련 모달 직경 프로그래밍 : 싸이클에서 OFF	m	*FB1(P1)*
DIAM90	G90 의 경우 직경 프로그램 , G91 의 경우 반경 프로그램	m	*PGAsl*
DIAM90A	G90 및 AC 의 경우 직경 프로그램 , G91 및 IC 의 경우 반경 프로그램	m	*PGsl*
DIAMCHAN	MD 축 기능에서 직경 프로그래밍 채널 상태로 모든 축 전송		*PGsl*
DIAMCHANA	직경 프로그래밍 채널 상태 전송		*PGsl*
DIAMCYCOF	채널 관련 직경 프로그래밍 : 싸이클에서 OFF	m	*FB1(P1)*
DIAMOF [3]	직경 프로그래밍 : OFF 정상 위치 . 장비 제조업체 사양 참조	m	*PGsl*
DIAMOFA	축 관련 모달 직경 프로그래밍 : OFF 정상 위치 . 장비 제조업체 사양 참조	m	*PGsl*
DIAMON	직경 프로그래밍 : ON	m	*PGsl*
DIAMONA	축 관련 모달 직경 프로그래밍 : ON 활성화 . 장비 제조업체 사양 참조	m	*PGsl*
DIC	상대 넌모달 축 관련 직경 프로그래밍	s	*PGsl*
DILF	후퇴 경로 (길이)	m	*PGsl*
DISABLE	인터럽트 OFF		*PGAsl* 인터럽트 루틴 할당 비활성화 / 다시 활성화 (DISABLE, ENABLE) [쪽 123]
DISC	트렌지션 원 오버슈트 공구 반경 보정	m	*PGsl*
DISCL	고속 절입 모션의 종점과 가공 평면 사이의 공차		*PGsl*

조작	의미	W [1)]	설명 참조 [2)]
DISPLOF	현재 블록 디스플레이 억제		*PGAs/* 현재 블록 디스플레이 억제 (DISPLOF, DISPLON, ACTBLOCNO) [쪽 174]
DISPLON	현재 블록 디스플레이의 억제 취소		*PGAs/* 현재 블록 디스플레이 억제 (DISPLOF, DISPLON, ACTBLOCNO) [쪽 174]
DISPR	리포지셔닝 경로 차이	s	*PGAs/* 형상에 대해 위치 변경(REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [쪽 478]
DISR	리포지셔닝 거리	s	*PGAs/* 형상에 대해 위치 변경(REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [쪽 478]
DITE	나사 런아웃 경로	m	*PGs/*
DITS	나사 런인 경로	m	*PGs/*
DIV	정수 분할		*PGAs/* 산술 함수 [쪽 64]
DL	위치별 추가 공구 옵셋 (DL, 전체 셋업 옵셋) 선택	m	*PGAs/* 추가 옵셋 선택 (DL) [쪽 396]
DO	동기 동작 키워드 . 조건이 충족되면 동작 실행		*PGAs/* 동작 (DO) [쪽 557]
DRFOF	핸드휠 옵셋 (DRF) 해제	m	*PGs/*
DRIVE	속도 의존 경로 가속	m	*PGs/*
DRIVEA	프로그래밍된 축에서 굴곡 가속 특성 커브 활성화		*PGs/*
DYNFINISH	정삭에 대한 다이나믹 응답	m	*PGs/*
DYNNORM	표준 다이나믹 반응	m	*PGs/*
DYNPOS	포지셔닝 모드에 대한 다이나믹 반응 . 태핑	m	*PGs/*
DYNROUGH	황삭에 대한 다이나믹 반응	m	*PGs/*
DYNSEMIFIN	정삭에 대한 다이나믹 반응	m	*PGs/*
DZERO	TO 유닛의 D 번호를 모두 유효하지 않은 것으로 표시		*PGAs/* D 번호의 자유 할당 : 무효화된 D 번호 (DZERO) [쪽 436]

조작	의미	W [1]	설명 참조 [2]
EAUTO	마지막 세 점을 사용한 마지막 스플라인 섹션 정의	m	*PGAsl* 스플라인 보간 (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [쪽 240]
EGDEF	전자 기어 정의		*PGAsl* 전자 기어 정의 (EGDEF) [쪽 527]
EGDEL	추종 축에 대한 커플링 정의 삭제		*PGAsl* 전자 기어 정의 삭제 (EGDEL) [쪽 533]
EGOFC	지속적으로 전자 기어 OFF		*PGAsl* 전자 기어박스 활성화 (EGOFS, EGOFC) [쪽 532]
EGOFS	선택적으로 전자 기어 OFF		*PGAsl* 전자 기어박스 활성화 (EGOFS, EGOFC) [쪽 532]
EGON	전자 기어 ON		*PGAsl* 전자 기어박스 활성화 (EGOFS, EGOFC) [쪽 532]
EGONSYN	전자 기어 ON		*PGAsl* 전자 기어박스 활성화 (EGOFS, EGOFC) [쪽 532]
EGONSYNE	전자 기어 ON. 접근 모드 지정		*PGAsl* 전자 기어박스 활성화 (EGOFS, EGOFC) [쪽 532]
ELSE	IF 조건을 충족하지 않으면 프로그램 분기		*PGAsl* 대체 블록을 가진 프로그램 루프 (IF, ELSE, ENDIF) [쪽 106]
ENABLE	인터럽트 ON		*PGAsl* 인터럽트 루틴 할당 비활성화 / 다시 활성화 (DISABLE, ENABLE) [쪽 123]
ENAT [3]	다음 이송 블록으로 자연적 트렌지션	m	*PGAsl* 스플라인 보간 (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [쪽 240]
ENDFOR	FOR 카운터 루프의 끝 라인		*PGAsl* 카운트 루프 (FOR ... TO ..., ENDFOR) [쪽 109]
ENDIF	IF 분기의 끝 라인		*PGAsl* 대체 블록을 가진 프로그램 루프 (IF, ELSE, ENDIF) [쪽 106]
ENDLABEL	REPEAT 를 사용한 가공 프로그램 반복 종료 라벨		*PGAsl, FB1(K1)* 프로그램 섹션 반복 (REPEAT, REPEATB, ENDLABEL, P) [쪽 98]
ENDLOOP	무한 프로그램 루프 (LOOP) 의 종료 라인 지정		*PGAsl* 연속 프로그램 루프 (LOOP, ENDLOOP) [쪽 108]
ENDPROC	PROC 로 시작된 프로그램의 끝 라인 .		
ENDWHILE	WHILE 루프의 끝 라인		*PGAsl* 루프 시작 시 조건을 가진 프로그램 루프 (WHILE, ENDWHILE) [쪽 111]
ESRR	드라이브에서 드라이브 자율 ESR 후퇴 파라미터화		*PGAsl* 드라이브 통합 후퇴 (ESRS) 설정 [쪽 709]

조작	의미	W [1]	설명 참조 [2]
ESRS	드라이브에서 드라이브 자율 ESR 차단 파라미터화		*PGAs/* 드라이브 통합 정지 (ESRS) 설정 [쪽 708]
ETAN	스플라인 시작에서 다음 이송 블록으로 접선 트렌지션	m	*PGAs/* 스플라인 보간 (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [쪽 240]
EVERY	조건이 FALSE에서 TRUE로 변하면 동기 동작 실행		*PGAs/* 주기적으로 조건 검사 (WHEN, WHENEVER, FROM, EVERY) [쪽 555]
EX	지수 표시에서 값 지정을 위한 키워드		*PGAs/* 사전 정의된 사용자 변수 : 산술 파라미터 (R) [쪽 21]
EXECSTRING	가공 프로그램 라인 실행에서 문자 변수 전송		*PGAs/* 가공 프로그램 라인 간접 프로그램(EXECSTRING) [쪽 63]
EXECTAB	모션 테이블에서 요소 실행		*PGAs/* 가공 프로그램 라인 간접 프로그램(EXECSTRING) [쪽 63]
EXECUTE	프로그램 실행 ON		*PGAs/* 형상 준비 종료 (EXECUTE) [쪽 727]
EXP	지수 함수 ex		*PGAs/* 산술 함수 [쪽 64]
EXTCALL	외부 서브프로그램 실행		*PGAs/* 외부 서브루틴 실행 (EXTCALL) [쪽 201]
EXTCLOSE	쓰기를 위해 열었던 외부 장치 / 파일 종료		*PGAs/* 외부 장치 / 파일로 출력 (EXTOPEN, WRITE, EXTCLOSE) [쪽 696]
EXTERNAL	파라미터 전송을 통한 서브프로그램 선언		*PGAs/* 파라미터 전송 없는 서브프로그램 호출 [쪽 186]
EXTOPEN	쓰기를 위해 채널의 외부 장치 / 파일 열기		*PGAs/* 외부 장치 / 파일로 출력 (EXTOPEN, WRITE, EXTCLOSE) [쪽 696]
F	이송 속도 값 (G4 와 연계하여 F 로 드웰 시간도 프로그래밍)		*PGs/*
FA	축 이송 속도	m	*PGs/*
FAD	소프트 접근 및 후퇴를 위한 절입 속도		*PGs/*
FALSE	논리 상수 : 잘못됨		*PGAs/* 사용자 변수 정의 (DEF) [쪽 25]
FB	넌모달 이송 속도		*PGs/*
FCTDEF	폴리노미얼 기능의 정의		*PGAs/* 온라인 공구 옵셋 (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) [쪽 408]

조작	의미	W [1]	설명 참조 [2]
FCUB	입체 스플라인에 따른 이송 속도 변수	m	*PGAs/* 이송 속도 응답 (FNORM, FLIN, FCUB, FPO) [쪽 462]
FD	핸드휠 오버라이드의 경로 이송 속도	s	*PGs/*
FDA	핸드휠 오버라이드의 축 이송 속도	s	*PGs/*
FENDNORM	코너 감속 OFF	m	*PGAs/* 코너 감속을 사용한 피드 감소 (FENDNORM, G62, G621) [쪽 277]
FFWOF [3]	피드포워드 제어 OFF	m	*PGs/*
FFWON	피드포워드 제어 ON	m	*PGs/*
FGREF	로터리 축의 기준 반경 또는 오리엔테이션 축의 경로 기준 팩터 (벡터 보간)	m	*PGs/*
FGROUP	축의 피드 그룹 정의		*PGs/*
FI	프레임 데이터에 액세스하기 위한 파라미터 : 미세 옵셋		*PGAs/* 프레임 구성 요소 읽기 및 변경 (TR, FI, RT, SC, MI) [쪽 294]
FIFOCTRL	전처리 버퍼 제어	m	*PGAs/* 전처리 메모리를 사용한 프로그램 순서 (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE) [쪽 468]
FILEDATE	파일의 최종 수정 날짜 보고		*PGAs/* 파일 정보 읽기 (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) [쪽 151]
FILEINFO	FILEDATE, FILESIZE, FILESTAT 및 FILETIME 을 열거한 요약 정보 보고		*PGAs/* 파일 정보 읽기 (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) [쪽 151]
FILESIZE	현재 파일 크기 보고		*PGAs/* 파일 정보 읽기 (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) [쪽 151]
FILESTAT	파일의 읽기 , 쓰기 , 실행 , 디스플레이 , 삭제 (rwxsd) 권한 상태 보고		*PGAs/* 파일 정보 읽기 (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) [쪽 151]
FILETIME	파일의 최종 수정 시간 보고		*PGAs/* 파일 정보 읽기 (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) [쪽 151]
FINEA	'미세 정위치 정지'에 도달할 때 모션 종료	m	*PGAs/* 프로그래밍된 이동 종료 기준 (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) [쪽 278]
FL	동기 축의 한계 속도	m	*PGs/*

조작	의미	W [1]	설명 참조 [2]
FLIN	직선 피드 변수	m	*PGAs/* 이송 속도 응답 (FNORM, FLIN, FCUB, FPO) [쪽 462]
FMA	축 다중 이송 속도	m	*PGs/*
FNORM [3]	DIN 66025 에 따른 일반 이송 속도	m	*PGAs/* 이송 속도 응답 (FNORM, FLIN, FCUB, FPO) [쪽 462]
FOCOF	제한된 토크를 통한 이송 OFF	m	*PGAs/* 고정 정지점으로 이동 (FXS, FXST, FXSW, FOCON, FOCOF) [쪽 620]
FOCON	제한된 토크를 통한 이송 ON	m	*PGAs/* 고정 정지점으로 이동 (FXS, FXST, FXSW, FOCON, FOCOF) [쪽 620]
FOR	고정 실행 횟수 카운터 루프		*PGAs/* 카운트 루프 (FOR ... TO ..., ENDFOR) [쪽 109]
FP	고정 정지점 : 접근할 고정 정지점의 수	s	*PGs/*
FPO	폴리노미얼을 통해 프로그래밍된 이송 속도 특성		*PGAs/* 이송 속도 응답 (FNORM, FLIN, FCUB, FPO) [쪽 462]
FPR	로터리 축 이름		*PGs/*
FPRAOF	회전 당 이송 속도 OFF		*PGs/*
FPRAON	회전 당 이송 속도 ON		*PGs/*
FRAME	좌표계 정의를 위한 데이터 유형		*PGAs/* 새 프레임 정의 (DEF FRAME) [쪽 297]
FRC	반경 및 모따기를 위한 이송 속도	s	*PGs/*
FRCM	반경 및 모따기를 위한 이송 속도 , 모달	m	*PGs/*
FROM	일단 조건이 충족되고 동기 동작이 활성 상태인 경우 동작이 실행됨		*PGAs/* 주기적으로 조건 검사 (WHEN, WHENEVER, FROM, EVERY) [쪽 555]
FTOC	미세 공구 옵셋 변경		*PGAs/* 온라인 공구 옵셋 (FTOC) [쪽 591]
FTOCOF [3]	온라인 미세 공구 옵셋 OFF	m	*PGAs/* 온라인 공구 옵셋 (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) [쪽 408]
FTOCON	온라인 미세 공구 옵셋 ON	m	*PGAs/* 온라인 공구 옵셋 (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) [쪽 408]
FXS	고정 정지점으로 이동 ON	m	*PGs/*

조작	의미	W [1]	설명 참조 [2]
FXST	고정 정지점으로 이동을 위한 토크 한계	m	*PGsl*
FXSW	고정 정지점으로 이동을 위한 모니터링 창		*PGsl*
FZ	절삭날 이송 속도	m	*PGsl*

조작 연산	의미	W [1]	설명 참조 [2]
G0	급 이송 직선 보간 (급 이송 모션)	m	*PGsl*
G1 [3]	절삭 이송 직선 보간 (직선 보간)	m	*PGsl*
G2	CW 방향 원호 보간	m	*PGsl*
G3	CCW 방향 원호 보간	m	*PGsl*
G4	드웰 시간 . 프리셋	s	*PGsl*
G5	사면 플런지 연삭	s	*PGAsl* 경사 축 (TRAANG) [쪽 375]
G7	사면 플런지 연삭 동안의 보정 동작	s	*PGAsl* 경사 축 (TRAANG) [쪽 375]
G9	정위치 정지 - 감속	s	*PGsl*
G17 [3]	X/Y 작업 평면 선택	m	*PGsl*
G18	X/Z 작업 평면 선택	m	*PGsl*
G19	Y/Z 작업 평면 선택	m	*PGsl*
G25	하위 작업 영역 제한	s	*PGsl*
G26	상위 작업 영역 제한	s	*PGsl*
G33	일정 리드 나사 절삭	m	*PGsl*
G34	선형 리드 증가 나사 절삭	m	*PGsl*
G35	선형 리드 감소 나사 절삭	m	*PGsl*
G40 [3]	공구 반경 보정 OFF	m	*PGsl*

조작 연산	의미	W [1]	설명 참조 [2]
G41	형상 좌측 공구 반경 보정	m	*PGsl*
G42	형상 우측 공구 반경 보정	m	*PGsl*
G53	현재 워크 옵셋 해제 (넌모달)	s	*PGsl*
G54	첫 번째 워크 옵셋	m	*PGsl*
G55	2. 셋터블 워크 옵셋	m	*PGsl*
G56	3. 셋터블 워크 옵셋	m	*PGsl*
G57	4. 셋터블 워크 옵셋	m	*PGsl*
G58 (840D sl)	축 워크 옵셋 . 절대 , 일반 옵셋	s	*PGsl*
G58 (828D)	5. 셋터블 워크 옵셋	m	*PGsl*
G59 (840D sl)	축 워크 옵셋 . 추가 , 미세 옵셋	s	*PGsl*
G59 (828D)	6. 셋터블 워크 옵셋	m	*PGsl*
G60 [3]	정위치 정지 - 감속	m	*PGsl*
G62	공구 반경 옵셋 활성 시 (G41, G42) 내부 코너에서 코너 감속	m	*PGAsl* 코너 감속을 사용한 피드 감소 (FENDNORM, G62, G621) [쪽 277]
G63	태핑 척을 사용한 태핑	s	*PGsl*
G64	연속 경로 모드	m	*PGsl*
G70	인치계 치수 (길이)	m	*PGsl*
G71 [3]	미터계 치수 (길이)	m	*PGsl*
G74	원점 복귀	s	*PGsl*
G75	고정 정지점 접근	s	*PGsl*
G90 [3]	절대 치수	m/s	*PGsl*

조작 연산	의미	W [1]	설명 참조 [2]
G91	증분 치수	m/s	*PGsl*
G93	역 시간 이송 속도 rpm	m	*PGsl*
G94 [3]	직선 이송 속도 F (mm/min 또는 inch/min 및 °/min)	m	*PGsl*
G95	회전 당 이송 속도 F (mm/rev 또는 inch/rev)	m	*PGsl*
G96	주속 일정 제어 (G95) ON	m	*PGsl*
G97	주속 일정 제어 (G95) OFF	m	*PGsl*
G110	마지막으로 프로그래밍한 지령치 위치에 상대적인 극점 프로그래밍	s	*PGsl*
G111	현재 공작물 좌표계의 원점에 상대적인 극점 프로그래밍	s	*PGsl*
G112	마지막 유효 극점에 상대적인 극점 프로그래밍	s	*PGsl*
G140 [3]	G41/G42 로 정의된 SAR 접근 방향	m	*PGsl*
G141	형상 좌측으로의 SAR 접근 방향	m	*PGsl*
G142	형상 우측으로의 SAR 접근 방향	m	*PGsl*
G143	접선 의존 SAR 접근 방향	m	*PGsl*
G147	직선으로 소프트 접근	s	*PGsl*
G148	직선으로 소프트 후퇴	s	*PGsl*
G153	베이직 프레임을 포함한 현재 프레임 억제	s	*PGsl*
G247	4 분원으로 소프트 접근	s	*PGsl*
G248	4 분원으로 소프트 후퇴	s	*PGsl*
G290	SINUMERIK 모드로 전환 ON	m	*FBW*
G291	ISO2/3 모드로 전환 ON	m	*FBW*
G331	리지드 태핑 , 양의 리드 , CW 방향	m	*PGsl*
G332	리지드 태핑 , 음의 리드 , CCW 방향	m	*PGsl*

조작 연산	의미	W [1]	설명 참조 [2]
G340 [3]	입체 접근 (X, Y, Z (헬릭스))	m	*PGsl*
G341	수직 절입 (Z) 후 평면 절입 (X,Y)	m	*PGsl*
G347	반원으로 소프트 접근	s	*PGsl*
G348	반원으로 소프트 후퇴	s	*PGsl*
G450 [3]	트렌지션 원	m	*PGsl*
G451	등거리의 교차	m	*PGsl*
G460 [3]	접근 및 후퇴 블록의 충돌 탐지 활성화	m	*PGsl*
G461	원호를 TRC 블록에 삽입	m	*PGsl*
G462	직선을 TRC 블록에 삽입	m	*PGsl*
G500 [3]	전체 셋터블 프레임 해제 , 베이직 프레임은 활성 상태	m	*PGsl*
G505~G599	5 ... 99. 셋터블 워크 옵셋	m	*PGsl*
G601 [3]	미세 정위치 정지에서 블록 변경	m	*PGsl*
G602	일반 정위치 정지에서 블록 변경	m	*PGsl*
G603	IPO 블록 끝에서 블록 변경	m	*PGsl*
G621	모든 코너에서 코너 감속	m	*PGAsl* 코너 감속을 사용한 피드 감소 (FENDNORM, G62, G621) [쪽 277]
G641	거리 기준에 따라 스무딩이 이루어지는 연속 경로 모드 (= 프로그램 가능한 라운딩 공차)	m	*PGsl*
G642	정해진 공차 안에서 스무딩 기능을 포함하는 연속 경로 모드	m	*PGsl*
G643	정해진 공차 안에서 스무딩 기능을 포함하는 연속 경로 모드 (블록 내경)	m	*PGsl*
G644	다이나믹 응답을 가능 최대치로 하여 스무딩하는 연속 경로 모드	m	*PGsl*
G645	정해진 공차 안에서 스무딩 및 접선 블록 트렌지션 기능을 포함하는 연속 경로 모드	m	*PGsl*

조작 연산	의미	W [1]	설명 참조 [2]
G700	인치계 치수 (길이 , 이송 속도)	m	*PGsl*
G710 [3]	미터계 치수 (길이 , 이송 속도)	m	*PGsl*
G751	중간점을 통해 고정 정지점 접근	s	PGsl
G810 [3], ..., G819	OEM 사용자를 위한 G 그룹		*PGAsl* OEM 사용자를 위한 특수 기능 (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829) [쪽 276]
G820 [3], ..., G829	OEM 사용자를 위한 G 그룹		*PGAsl* OEM 사용자를 위한 특수 기능 (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829) [쪽 276]
G931	이송 시간으로 지정한 이송 속도	m	
G942	직선 이송 속도와 주속 일정 제어 또는 스핀들 속도 동결	m	
G952	회전 당 이송 속도 및 주속 일정 제어 또는 스핀들 속도 동결	m	
G961	주속 일정 제어 및 직선 이송 속도	m	*PGsl*
G962	직선 또는 회전당 이송 속도 및 주속 일정 제어	m	*PGsl*
G971	스핀들 속도 및 직선 이송 속도 동결	m	*PGsl*
G972	직선 또는 회전 당 이송 속도 및 스핀들 속도 동결	m	*PGsl*
G973	스핀들 속도 제한이 없는 회전 당 이송 속도	m	*PGsl*
GEOAX	새로운 채널 축을 기하 축 1 - 3 으로 할당		*PGAsl* 대체 가능한 기하 축 (GEOAX) [쪽 670]
GET	채널 간 활성 축 교체		*PGAsl* 축 교체, 스핀들 교체 (RELEASE, GET, GETD) [쪽 131]
GETACTT	그룹에서 이름이 동일한 공구 가져오기		*FBW*
GETACTTD	절대 D 번호와 연결된 T 번호 가져오기		*PGAsl* D 번호의 자유 할당 : 지정된 D 번호의 T 번호를 결정 (GETACTTD) [쪽 435]
GETD	채널 간의 직접 축 교체		*PGAsl* 축 교체, 스핀들 교체 (RELEASE, GET, GETD) [쪽 131]
GETDNO	공구 (T) 절삭날 (CE) 의 D 번호 보고		*PGAsl* D 번호의 자유 할당 : D 번호 이름 변경 (GETDNO, SETDNO) [쪽 434]
GETEXET	로드된 T 번호 읽기		*FBW*
GETFREELOC	메거진에서 특정 공구에 사용 가능한 포켓 찾기		*FBW*
GETSELT	선택한 T 번호 보고		*FBW*

조작 연산	의미	W [1]	설명 참조 [2]
GETT	공구 이름의 T 번호 가져오기		*FBW*
GETTCOR	공구 길이 및 / 또는 공구 길이 구성요소 읽기		*FB1(W1)*
GETTENV	T, D 및 DL 번호 읽기		*FB1(W1)*
GOTO	전방으로 점프 후 후방으로 점프 (처음에는 프로그램 종료 방향 그 다음 프로그램 시작 방향)		*PGAs/* 점프 마커로 프로그램 점프 (GOTOB, GOTOF, GOTO, GOTOC) [쪽 93]
GOTOB	후방 점프 (프로그램 시작 방향으로)		*PGAs/* 점프 마커로 프로그램 점프 (GOTOB, GOTOF, GOTO, GOTOC) [쪽 93]
GOTOC	GOTO 과 유사하나 , 알람 14080 ' 점프 대상을 찾을 수 없음 ' 을 억제		*PGAs/* 점프 마커로 프로그램 점프 (GOTOB, GOTOF, GOTO, GOTOC) [쪽 93]
GOTOF	전방 점프 (프로그램 종료 방향으로)		*PGAs/* 점프 마커로 프로그램 점프 (GOTOB, GOTOF, GOTO, GOTOC) [쪽 93]
GOTOS	프로그램 시작으로 점프		*PGAs/* 프로그램 시작으로 점프 (GOTOS) [쪽 92]
GP	위치 특성의 간접 프로그래밍 키워드		*PGAs/* 위치 특성 간접 프로그램 (BP) [쪽 60]
GWPSOF	연삭 휠 원주 속도 일정 제어 (GWPS) 선택 취소	s	*PGs/*
GWPSON	연삭 휠 원주 속도 일정 제어 (GWPS) 선택	s	*PGs/*
H...	PLC 로 보조 기능 출력		*PGs/FB1(H2)*
HOLES1	기술 싸이클 : 홀의 라인		*PGAs/* 여러 구멍으로 이루어진 행 – HOLES1 [쪽 746]
HOLES2	기술 싸이클 : 홀 원호		*PGAs/* 여러 개의 구멍으로 형성된 원 - HOLES2 [쪽 748]
I	보간 파라미터	s	*PGs/*
I1	중간점 좌표	s	*PGs/*
IC	증분 치수	s	*PGs/*
ICYCOF	ICYCOF 이후 고정 싸이클의 전체 블록은 하나의 보간 싸이클로 처리됩니다 .		*PGAs/* 공정 싸이클을 제어 처리 (ICYCOF, ICYCON) [쪽 631]
ICYCON	ICYCON 이후 고정 싸이클의 개별 블록은 별도의 보간 싸이클로 처리됩니다 .		*PGAs/* 공정 싸이클을 제어 처리 (ICYCOF, ICYCON) [쪽 631]
ID	모달 동기 동작의 이름	m	*PGAs/* 적용 범위 및 가공 순서 (ID, IDS) [쪽 553]

조작 연산	의미	W [1]	설명 참조 [2]
IDS	모달 스태틱 동기 동작의 이름		*PGAs/* 적용 범위 및 가공 순서 (ID, IDS) [쪽 553]
IF	가공 프로그램 / 공정 싸이클에서 조건 점프		*PGAs/* 대체 블록을 가진 프로그램 루프 (IF, ELSE, ENDIF) [쪽 106]
INDEX	입력 문자열에서 문자 인덱스 정의		*PGAs/* 문자열에서 문자 / 문자열 검색 (INDEX, RINDEX, MINDEX, MATCH) [쪽 81]
INIPO	전원을 켤 때 변수 초기화		*PGAs/* 사용자 변수 정의 (DEF) [쪽 25]
INIRE	리셋할 때 변수 초기화		*PGAs/* 사용자 변수 정의 (DEF) [쪽 25]
INICF	NewConfig 에서 변수 초기화		*PGAs/* 사용자 변수 정의 (DEF) [쪽 25]
INIT	특정 채널에서 실행할 특정 NC 프로그램 선택		*PGAs/* 프로그램 코디네이션 (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) [쪽 114]
INITIAL	모든 영역에서 INI 파일 생성		*PGAs/* 작업 메모리 (CHANDATA, COMPLETE, INITIAL) [쪽 218]
INT	데이터 유형 : 부호가 있는 정수		*PGAs/* 사용자 변수 정의 (DEF) [쪽 25]
INTERSEC	두 형상 요소 간의 교차 계산		*PGAs/* 형상 요소 2 개 사이의 교차점을 결정 (INTERSEC) [쪽 722]
INVCCW	인벌류트 트레이스 , CCW 방향	m	*PGs/*
INVCW	인벌류트 트레이스 , CW 방향	m	*PGs/*
INVFRAME	프레임에서 역 프레임 계산		*FB1(K2)*
IP	변수 보간 파라미터		*PGAs/* 간접 프로그램 [쪽 56]
IPOBRKA	정지 동작 활성화의 모션 기준	m	*PGAs/* 프로그래밍된 이동 종료 기준 (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) [쪽 278]
IPOENDA	'IPO 정지 ' 에 도달할 때 모션 종료	m	*PGAs/* 프로그래밍된 이동 종료 기준 (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) [쪽 278]
IPTRLOCK	다음 머신 펑션 블록에서 추적 불가 프로그램 섹션의 시작 동결	m	*PGAs/* SERUPRO (IPTRLOCK, IPTRUNLOCK) 에 대한 방지 프로그램 위치 [쪽 476]
IPTRUNLOCK	현재 블록 중단 시점에서 추적 불가 프로그램 섹션의 종료를 설정	m	*PGAs/* SERUPRO (IPTRLOCK, IPTRUNLOCK) 에 대한 방지 프로그램 위치 [쪽 476]

조작 연산	의미	W [1)]	설명 참조 [2)]
ISAXIS	기하 축 1 이 파라미터로 지정되었는지 확인		*PGAsl* 축 기능 (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) [쪽 667]
ISD	삽입 깊이	m	*PGAsl* 3D 공구 옵셋 활성화 (CUT3DC, CUT3DF, CUT3DFS, CUT3DFF, ISD) [쪽 413]
ISFILE	파일이 NCK 애플리케이션 메모리에 있는지 점검		*PGAsl* 파일의 존재 여부 점검 (ISFILE) [쪽 149]
ISNUMBER	입력 문자열을 숫자로 전환 가능한지 점검		*PGAsl* STRING 에서 유형 변환 (NUMBER, ISNUMBER, AXNAME) [쪽 77]
ISOCALL	ISO 언어로 프로그래밍된 프로그램의 간접 호출		*PGAsl* ISO 언어로 프로그래밍된 프로그램의 간접 호출 (ISOCALL) [쪽 196]
ISVAR	전송 파라미터가 NC에서 선언된 변수를 포함하는지 점검		*PGAsl* 함수 호출 ISVAR 과 판독 기계 데이터 배열 색인 [쪽 686]
J	보간 파라미터	s	*PGsl*
J1	중간점 좌표	s	*PGsl*
JERKA	프로그래밍된 축의 MD 를 통한 가속 반응 활성화		
JERKLIM	최대 축 저크의 감소 또는 오버슈트	m	*PGAsl* 백분율 저크 수정 (JERKLIM) [쪽 487]
JERKLIMA	최대 축 저크의 감소 또는 오버슈트	m	*PGsl*
K	보간 파라미터	s	*PGsl*
K1	중간점 좌표	s	*PGsl*
KONT	공구 옵셋에서 형상 주위로 이동	m	*PGsl*
KONTC	연속 커브 폴리노미얼로 접근 / 후퇴	m	*PGsl*
KONTT	연속 접선 폴리노미얼로 접근 / 후퇴	m	*PGsl*
L	서브프로그램 번호	s	*PGAsl* 파라미터 전송 없는 서브프로그램 호출 [쪽 186]
LEAD	리드 각도 1. 공구 오리엔테이션 2. 오리엔테이션 폴리노미얼	m	*PGAsl* 공구 방향 프로그램 (A..., B..., C..., LEAD, TILT) [쪽 327]
LEADOF	마스터 값 커플링 OFF		*PGAsl* 축 관련 리딩 값 결합 (LEADON, LEADOF) [쪽 521]

조작 연산	의미	W [1]	설명 참조 [2]
LEADON	마스터 값 커플링 ON		*PGAsl* 축 관련 리딩 값 결합 (LEADON, LEADOF) [쪽 521]
LENTOAX	활성 공구의 공구 길이 L1, L2 및 L3 을 가로축 , 세로축 및 직교축에 지정하는 데에 대한 정보를 제공		*FB1(W1)*
LFOF [3]	나사 절삭을 위한 고속 후퇴 OFF	m	*PGsl*
LFON	나사 절삭을 위한 고속 후퇴 ON	m	*PGsl*
LFPOS	POLF 를 사용하여 프로그래밍된 절대 축 위치로 POLFMASK 또는 POLFMLIN 을 사용하여 선언된 축의 후퇴	m	*PGsl*
LFTXT	고속 후퇴를 위한 후퇴 이동 평면은 경로 접선 및 현재 공구 방향에서 결정됨	m	*PGsl*
LFWP	고속 후퇴를 위한 후퇴 이동 평면은 현재 작업 평면 (G17/G18/G19) 에 의해 결정됨	m	*PGsl*
LIFTFAST	고속 후퇴		*PGsl* 형상으로부터 빠른 후퇴 (SETINT LIFTFAST, ALF) [쪽 125]
LIMS	G96/G961 및 G97 에 대한 속도 제한	m	*PGsl*
LLI	변수의 하한값		*PGAsl* 속성 : 한계값 (LLI, ULI) [쪽 37]
LN	자연 로그		*PGAsl* 산술 함수 [쪽 64]
LOCK	ID 로 동기 동작 비활성화 (고정 싸이클 정지)		*PGAsl* 해제 , 인에이블 , 리셋 (LOCK, UNLOCK, RESET) [쪽 633]
LONGHOLE	기술 싸이클 : 연장된 홀		*PGAsl* 연장된 홀 - LONGHOLE [쪽 769]
LOOP	무한 루프 개시		*PGAsl* 연속 프로그램 루프 (LOOP, ENDLOOP) [쪽 108]

작업	의미	W [1]	설명 참조 [2]
M0	프로그램 정지		*PGsl*
M1	옵셔널 정지		*PGsl*
M2	프로그램 시작으로 복귀하면서 메인 프로그램 종료		*PGsl*
M3	CW 방향 스핀들 회전		*PGsl*

작업	의미	W [1)]	설명 참조 [2)]
M4	CCW 방향 스핀들 회전		*PGsl*
M5	스핀들 정지		*PGsl*
M6	공구 교환		*PGsl*
M17	서브프로그램 종료		*PGsl*
M19	SD43240 에 입력된 위치로 스핀들 포지셔닝		*PGsl*
M30	프로그램 종료 , M2 와 동일한 효과		*PGsl*
M40	자동 기어 변경		*PGsl*
M41 ... M45	기어 단수 1 ... 5		*PGsl*
M70	축 모드로 전환		*PGsl*
MASLDEF	마스터 / 슬레이브 축 그룹핑 정의		*PGAsl* 마스터 / 슬레이브 그룹 (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) [쪽 546]
MASLDEL	마스터 / 슬레이브 축 그룹핑 해제 및 그룹핑 정의 제거		*PGAsl* 마스터 / 슬레이브 그룹 (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) [쪽 546]
MASLOF	임시 커플링의 해제		*PGAsl* 마스터 / 슬레이브 그룹 (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) [쪽 546]
MASLOFS	자동 커플링의 해제 축 정지로 임시 커플링의 해제		*PGAsl* 마스터 / 슬레이브 그룹 (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) [쪽 546]
MASLON	임시 커플링의 동작		*PGAsl* 마스터 / 슬레이브 그룹 (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) [쪽 546]
MATCH	문자열에서 문자열 검색		*PGAsl* 문자열에서 문자 / 문자열 검색 (INDEX, RINDEX, MINDEX, MATCH) [쪽 81]
MAXVAL	두 변수 중 높은 값 (산술 함수)		*PGAsl* 변수 최소, 최대 및 범위(MINVAL, MAXVAL 및 BOUND) [쪽 71]
MCALL	모달 서브프로그램 호출		*PGAsl* 모달 서브프로그램 호출 (MCALL) [쪽 192]
MEAC	이동할 거리를 삭제하지 않고 연속 측정	s	*PGAsl* 확장된 측정 기능 (MEASA, MEAWA, MEAC) (옵션) [쪽 268]

작업	의미	W [1]	설명 참조 [2]
MEAFRAME	측정 지점에서 프레임 계산		*PGAsl* 공간의 세 개 측정 지점에서 프레임 계산 (MEAFRAME) [쪽 302]
MEAS	터치 프로브를 사용한 측정	s	*PGAsl* 터치 프로브를 통한 측정 (MEAS, MEAW) [쪽 265]
MEASA	이동할 거리를 삭제하고 측정	s	*PGAsl* 확장된 측정 기능 (MEASA, MEAWA, MEAC) (옵션) [쪽 268]
MEASURE	공작물 및 공구 측정을 위한 계산 방법		*FB2(M5)* 터치 프로브를 통한 측정 (MEAS, MEAW) [쪽 265]
MEAW	이동할 거리를 삭제하지 않고 터치 프로브로 측정	s	*PGAsl* 터치 프로브를 통한 측정 (MEAS, MEAW) [쪽 265]
MEAWA	이동할 거리를 삭제하지 않고 측정	s	*PGAsl* 확장된 측정 기능 (MEASA, MEAWA, MEAC) (옵션) [쪽 268]
MI	프레임 데이터에 액세스 : 미러링		*PGAsl* 프레임 구성 요소 읽기 및 변경 (TR, FI, RT, SC, MI) [쪽 294]
MINDEX	입력 문자열에서 문자 인덱스 정의		*PGAsl* 문자열에서 문자 / 문자열 검색 (INDEX, RINDEX, MINDEX, MATCH) [쪽 81]
MINVAL	두 변수 중 낮은 값 (산술 함수)		*PGAsl* 변수 최소, 최대 및 범위(MINVAL, MAXVAL 및 BOUND) [쪽 71]
MIRROR	프로그래밍 미러링	s	*PGAsl*
MMC	HMI 상에서 가공 프로그램으로 부터 대화형 프로그램 창 호출		*PGAsl* 가공 프로그램 (MMC) 에서 창 양방향 호출 [쪽 690]
MOD	모듈로 분할		*PGAsl* 산술 함수 [쪽 64]
MODAXVAL	모듈로 로터리 축의 모듈로 위치 결정		*PGAsl* 축 기능 (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) [쪽 667]
MOV	축 포지셔닝 시작		*PGAsl* 축 시작 / 정지 (MOV) [쪽 600]
MSG	프로그램 가능한 메시지	m	*PGsl*
MVTOOL	공구 이동을 위한 언어 명령		*FBW*
N	NC 블록 번호		*PGsl*
NCK	데이터의 유효 범위 지정		*PGAsl* 사용자 변수 정의 (DEF) [쪽 25]

작업	의미	W [1]	설명 참조 [2]
NEWCONF	수정한 머신 데이터 적용 (' 머신 데이터 활성화 ' 에 해당)		*PGAs/* 가공 데이터 활성화 (NEWCONF) [쪽 137]
NEWT	새 공구 작성		*PGAs/* 프레임 구성 요소 읽기 및 변경 (TR, FI, RT, SC, MI) [쪽 294]
NORM [3]	공구 옵셋을 사용한 시작 지점과 종점의 표준 설정	m	*PGs/*
NOT	논리적 NOT (부정)		*PGAs/* 비교 및 논리 연산자 [쪽 67]
NPROT	기계 지정 보호 영역 ON/OFF		*PGAs/* 보호 영역 활성화 / 비활성화 (CPROT, NPROT) [쪽 227]
NPROTDEF	기계 지정 보호 영역의 정의		*PGAs/* 보호 영역의 정의 (CPROTDEF, NPROTDEF) [쪽 223]
NUMBER	입력 문자열을 숫자로 전환		*PGAs/* STRING 에서 유형 변환 (NUMBER, ISNUMBER, AXNAME) [쪽 77]
OEMIPO1	OEM 보간 1	m	*PGAs/* OEM 사용자를 위한 특수 기능 (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829) [쪽 276]
OEMIPO2	OEM 보간 2	m	*PGAs/* OEM 사용자를 위한 특수 기능 (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829) [쪽 276]
OF	CASE 분기의 키워드		*PGAs/* 프로그램 분기 (CASE ... OF ... DEFAULT ...) [쪽 96]
OFFN	프로그램된 형상의 허용치	m	*PGs/*
OMA1	OEM 어드레스 1	m	
OMA2	OEM 어드레스 2	m	
OMA3	OEM 어드레스 3	m	
OMA4	OEM 어드레스 4	m	
OMA5	OEM 어드레스 5	m	
OR	로직 연산자 , OR 연산		*PGAs/* 비교 및 논리 연산자 [쪽 67]
ORIAXES	기계 축 또는 오리엔테이션 축의 직선 보간	m	*PGAs/* 오리엔테이션 축 프로그래밍 (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) [쪽 338]
ORIAXPOS	로터리 축 위치의 가상 오리엔테이션 축을 통한 오리엔테이션 각도	m	
ORIC [3]	외부 코너에서의 오리엔테이션 변경은 삽입될 원호 블록 위에 오버레이됨	m	*PGAs/* 공구 오리엔테이션 (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) [쪽 427]

작업	의미	W [1]	설명 참조 [2]
ORICONCCW	CCW 방향으로 원주면 보간	m	*PGAsl/FB3(F3)* 테이퍼의 주변 표면을 따라 오리엔테이션 프로그래밍 (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) [쪽 340]
ORICONCW	CW 방향으로 원주면 보간	m	*PGAsl/FB3(F4)* 테이퍼의 주변 표면을 따라 오리엔테이션 프로그래밍 (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) [쪽 340]
ORICONIO	중간 오리엔테이션 설정으로 원주면 보간	m	*PGAsl/FB3(F4)* 테이퍼의 주변 표면을 따라 오리엔테이션 프로그래밍 (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) [쪽 340]
ORICONTO	접선 트렌지션에서 원주면 보간 (최종 오리엔테이션)	m	*PGAsl/FB3(F5)* 테이퍼의 주변 표면을 따라 오리엔테이션 프로그래밍 (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) [쪽 340]
ORICURVE	공구 두 접촉 지점의 모션 지정으로 오리엔테이션 보간	m	*PGAsl/FB3(F6)* 두 개의 접점 방향 지정 (ORICURVE, PO[XH]=, PO[YH]=, PO[ZH]=) [쪽 344]
ORID	오리엔테이션 변경은 원호 블록 이전에 수행됨	m	*PGAsl* 공구 오리엔테이션 (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) [쪽 427]
ORIEULER	Euler 각도를 통한 오리엔테이션 각	m	*PGAsl* 오리엔테이션 축 프로그래밍 (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) [쪽 338]
ORIMKS	기계 좌표계 공구 오리엔테이션	m	*PGAsl* 방향 축 기준 (ORIWKS, ORIMKS) [쪽 336]
ORIPATH	경로 관련 공구 오리엔테이션	m	*PGAsl* 경로 기준 공구 방향 회전 (ORIPATH, ORIPATHS, 회전 각도) [쪽 353]
ORIPATHS	경로와 관련된 공구 오리엔테이션 , 오리엔테이션 특성의 소음이 감소됩니다 .	m	*PGAsl* 경로 기준 공구 방향 회전 (ORIPATH, ORIPATHS, 회전 각도) [쪽 353]
ORIPLANE	평면에서 보간 (ORIVECT 에 일치), 대형 반경 원호 보간	m	*PGAsl* 테이퍼의 주변 표면을 따라 오리엔테이션 프로그래밍 (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) [쪽 340]
ORIRESET	최대 3 개의 오리엔테이션 축이 있는 초기 공구 오리엔테이션		*PGAsl* 오리엔테이션 프로그래밍 변수 및 초기 설정 (ORIRESET) [쪽 326]
ORIROTA	절대 회전 방향 회전 각도	m	*PGAsl* 공구 오리엔테이션 회전 (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) [쪽 348]
ORIROTC	경로 접선과 관계된 접선 회전 벡터	m	*PGAsl* 공구 오리엔테이션 회전 (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) [쪽 348]

작업	의미	W [1]	설명 참조 [2]
ORIROTR	시작과 종료 오리엔테이션 사이의 평면에 관련한 회전 각	m	*PGAs/* 공구 오리엔테이션 회전 (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) [쪽 348]
ORIROTT	오리엔테이션 벡터의 변경에 관련한 회전 각	m	*PGAs/* 공구 오리엔테이션 회전 (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) [쪽 348]
ORIRPY	RPY 각 (XYZ) 을 통한 오리엔테이션 각	m	*PGAs/* 오리엔테이션 축 프로그래밍 (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) [쪽 338]
ORIRPY2	RPY 각 (ZYX) 을 통한 오리엔테이션 각	m	*PGAs/* 오리엔테이션 축 프로그래밍 (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) [쪽 338]
ORIS	오리엔테이션 변경	m	*PGAs/* 공구 오리엔테이션 (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) [쪽 427]
ORISOF [3]	오리엔테이션 특성 스무딩 OFF	m	*PGAs/* 오리엔테이션 특성 스무딩 (ORISON, ORISOF) [쪽 361]
ORISON	오리엔테이션 특성 스무딩 ON	m	*PGAs/* 오리엔테이션 특성 스무딩 (ORISON, ORISOF) [쪽 361]
ORIVECT	대형 반경 원호 보간 (ORIPLANE 과 동일)	m	*PGAs/* 오리엔테이션 축 프로그래밍 (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) [쪽 338]
ORIVIRT1	가상 오리엔테이션 축을 통한 오리엔테이션 각도 (정의 1)	m	*PGAs/* 오리엔테이션 축 프로그래밍 (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) [쪽 338]
ORIVIRT2	가상 오리엔테이션 축을 통한 오리엔테이션 각도 (정의 1)	m	*PGAs/* 오리엔테이션 축 프로그래밍 (ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) [쪽 338]
ORIWKS [3]	공작물 좌표계의 공구 오리엔테이션 각도	m	*PGAs/* 방향 축 기준 (ORIWKS, ORIMKS) [쪽 336]
OS	오실레이션 ON/OFF		*PGAs/* 비동기 오실레이션 (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) [쪽 639]
OSB	오실레이션 : 시작점	m	*FB2(P5)*
OSC	연속 공구 오리엔테이션 스무딩	m	*PGAs/* 공구 오리엔테이션 (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) [쪽 427]
OSCILL	축 : 1 -3 절입 축	m	*PGAs/* 동기 동작에 의한 오실레이션 제어 (OSCILL) [쪽 645]

작업	의미	W [1]	설명 참조 [2]
OSCTRL	오실레이션 옵션	m	*PGAsl* 비동기 오실레이션 (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) [쪽 639]
OSD	SD 로 스무딩 거리를 지정 공구 오리엔테이션 스무딩	m	*PGAsl* 공구 오리엔테이션 (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) [쪽 427]
OSE	오실레이션 끝 위치	m	*PGAsl* 비동기 오실레이션 (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) [쪽 639]
OSNSC	오실레이션 : 스파크아웃 싸이클 개수	m	*PGAsl* 비동기 오실레이션 (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) [쪽 639]
OSOF [3]	공구 오리엔테이션 스무딩 OFF	m	*PGAsl* 비동기 오실레이션 (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) [쪽 639]
OSP1	오실레이션 좌측 반전 지점	m	*PGAsl* 비동기 오실레이션 (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) [쪽 639]
OSP2	오실레이션 우측 반전 지점	m	*PGAsl* 비동기 오실레이션 (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) [쪽 639]
OSS	블록 끝에서의 공구 오리엔테이션 스무딩	m	*PGAsl* 공구 오리엔테이션 (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) [쪽 427]
OSSE	블록 시작과 끝에서의 공구 오리엔테이션 스무딩	m	*PGAsl* 공구 오리엔테이션 (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) [쪽 427]
OST	SD (프로그래밍된 오리엔테이션 특성과의 최대 편차) 를 사용한 각도 공차 지정으로 공구 오리엔테이션 스무딩	m	*PGAsl* 공구 오리엔테이션 (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) [쪽 427]
OST1	오실레이션: 좌측 반전 지점의 정지 지점	m	*PGAsl* 비동기 오실레이션 (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) [쪽 639]
OST2	오실레이션: 우측 반전 지점의 정지 지점	m	*PGAsl* 비동기 오실레이션 (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) [쪽 639]
OTOL	컴프레서 기능에 대한 오리엔테이션 공차 . 오리엔테이션 스무딩 및 스무딩 종류		*PGAsl* 프로그램 가능한 형상 / 오리엔테이션 공차 (CTOL, OTOL, ATOL) [쪽 492]
OVR	속도 옵셋	m	*PGAsl*
OVRA	축 속도 옵셋	m	*PGAsl*
OVRRAP	급 이송 오버라이드	m	*PGAsl*

작업	의미	W [1]	설명 참조 [2]
P	서브프로그램 싸이클 개수		*PGAsl* 프로그램 반복 횟수 (P) [쪽 190]
PAROT	공작물로 공작물 좌표계 정렬	m	*PGsl*
PAROTOF	공작물과 관련된 프레임 회전 해제	m	*PGsl*
PCALL	절대 경로 및 파라미터 전송을 사용하여 서브프로그램 호출		*PGAsl* 경로 지정 및 파라미터를 통한 서브루틴 호출 (PCALL) [쪽 198]
PDELAYOF	펀칭 지연 OFF	m	*PGAsl* 펀칭 및 니블링 활성화 / 비활성화 (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) [쪽 653]
PDELAYON [3]	펀칭 지연 ON	m	*PGAsl* 펀칭 및 니블링 활성화 / 비활성화 (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) [쪽 653]
PHU	변수의 물리 단위		*PGAsl* 사용자 변수 정의 (DEF) [쪽 25]
PL	1. B 스플라인 : 노드 공차 2. 폴리노미얼 보간 : 폴리노미얼 보간을 위한 파라미터 간격 길이	s	*PGAsl* 1. 스플라인 보간 (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [쪽 240] 2. 폴리노미널 보간 (POLY, POLYPATH, PO, PL) [쪽 256]
PM	분 당		*PGsl*
PO	폴리노미얼 보간을 위한 폴리노미얼 계수	s	*PGAsl* 폴리노미널 보간 (POLY, POLYPATH, PO, PL) [쪽 256]
POCKET3	기술 싸이클 : 사각 포켓 밀링		*PGAsl* 사각 포켓 밀링 - POCKET3 [쪽 751]
POCKET4	기술 싸이클 : 원호 포켓 밀링		*PGAsl* 원호 포켓 밀링 – POCKET4 [쪽 754]
POLF	LIFTFAST 후퇴 위치	m	*PGsl/PGAsl*
POLFA	$AA_ESR_TRIGGER 을 사용하여 단일 축의 후퇴 시작 위치	m	*PGsl*
POLFMASK	축 사이의 연결 없는 후퇴를 위한 축 인에이블	m	*PGsl*
POLFMLIN	축 사이가 직선으로 연결된, 후퇴를 위한 축 인에이블	m	*PGsl*
POLY	폴리노미얼 보간	m	*PGAsl* 폴리노미널 보간 (POLY, POLYPATH, PO, PL) [쪽 256]

작업	의미	W [1]	설명 참조 [2]
POLYPATH	AXIS 또는 VECT 축 그룹에 대해 폴리노미얼 보간 선택 가능	m	*PGAsl* 폴리노미널 보간 (POLY, POLYPATH, PO, PL) [쪽 256]
PON	펀칭 ON	m	*PGAsl* 펀칭 및 니블링 활성화 / 비활성화 (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) [쪽 653]
PONS	보간 싸이클 내 펀칭 ON	m	*PGAsl* 펀칭 및 니블링 활성화 / 비활성화 (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) [쪽 653]
POS	축 포지셔닝		*PGsl*
POSA	블록 경계를 가로 지르는 축 포지셔닝		*PGsl*
POSM	메거진 포지셔닝		*FBW*
POSP	섹션에서 축 포지셔닝 (오실레이션)		*PGsl*
POSRANGE	현재 보간된 축의 위치 지령치가 미리 정의된 기준 위치 범위에 있는지 확인합니다 .		*PGAsl* 지정된 참조 범위에서 위치 지정 (POSRANGE) [쪽 599]
POT	제곱 (산술 함수)		*PGAsl* 산술 함수 [쪽 64]
PR	회전 당		*PGsl*
PREPRO	준비를 통한 서브프로그램 확인		*PGAsl* 준비 과정에서 서브프로그램 식별 (PREPRO) [쪽 177]
PRESETON	프로그래밍된 축의 실제 값 설정		*PGAsl* 사전 설정 옵셋 (PRESETON) [쪽 301]
PRIO	프로세스 인트럽트 순서 설정 키워드		*PGAsl* 인터럽트 루틴 지정 및 시작 (SETINT, PRIO, BLSYNC) [쪽 121]
PROC	프로그램의 첫 번째 동작		*PGAsl* 경로 지정 및 파라미터를 통한 서브루틴 호출 (PCALL) [쪽 198]
PTP	지점 간 모션	m	*PGAsl* 직교 PTP 이송 [쪽 380]
PTPG0	G0 으로만의 지점 간 모션 , 그렇지 않으면 CP 사용	m	*PGAsl* TRANSMIT 에 대한 PTP [쪽 385]
PUNCHACC	니블링 시 이송별 가속		*PGAsl* 펀칭 및 니블링 활성화 / 비활성화 (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) [쪽 653]
PUTFTOC	병렬 드레싱을 위한 공구 미세 옵셋		*PGAsl* 온라인 공구 옵셋 (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) [쪽 408]

작업	의미	W [1]	설명 참조 [2]
PUTFTOCF	FCTDEF 로 정의된 병렬 드레싱에 따른 공구 미세 옵셋		*PGAs/* 온라인 공구 옵셋 (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) [쪽 408]
PW	B 스플라인 , 점 무게	s	*PGAs/* 스플라인 보간 (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [쪽 240]
QECLRNOF	상한 돌기 보정 학습 OFF		*PGAs/* 보정 특성 배우기 (QECLRNON, QECLRNOF) [쪽 688]
QECLRNON	상한 돌기 보정 학습 ON		*PGAs/* 보정 특성 배우기 (QECLRNON, QECLRNOF) [쪽 688]
QU	고속 추가 (보조) 기능 출력		*PGs/*
R...	셋터블 어드레스 식별자 및 숫자로 구성된 산술 파라미터		*PGAs/* 사전 정의된 사용자 변수 : 산술 파라미터 (R) [쪽 21]
RAC	넌모달 축 지정 절대 반경 프로그래밍	s	*PGs/*
RDISABLE	리드인 디스에이블		*PGAs/* 리드인 비활성 설정 (RDISABLE) [쪽 579]
READ	지정된 파일에서 하나 또는 그 이상의 라인을 읽고 그 정보를 배열에 보관함		*PGAs/* 파일의 행 읽기 (READ) [쪽 145]
REAL	데이터 유형 : 부호가 있는 실수		*PGAs/* 사용자 변수 정의 (DEF) [쪽 25]
REDEF	머신 데이터 , NC 언어 및 시스템 변수 , 디스플레이될 사용자 그룹 지정 설정		*PGAs/* 시스템 변수 , 사용자 변수 및 NC 언어 명령어의 재 정의 (REDEF) [쪽 31]
RELEASE	축 교체를 위한 기계 축 릴리즈		*PGAs/* 축 교체, 스핀들 교체 (RELEASE, GET, GETD) [쪽 131]
REP	동일 값으로 배열의 모든 요소를 초기화하는 키워드		*PGAs/* 배열 변수 정의 및 초기화 (DEF, SET, REP): [쪽 47]
REPEAT	프로그램 루프 반복		*PGAs/* 프로그램 섹션 반복 (REPEAT, REPEATB, ENDLABEL, P) [쪽 98]
REPEATB	프로그램 라인 반복		*PGAs/* 프로그램 섹션 반복 (REPEAT, REPEATB, ENDLABEL, P) [쪽 98]
REPOSA	전체 축 직선 리포지셔닝	s	*PGAs/* 형상에 대해 위치 변경 (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [쪽 478]
REPOSH	반원으로 리포지셔닝	s	*PGAs/* 형상에 대해 위치 변경 (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [쪽 478]

작업	의미	W [1]	설명 참조 [2]
REPOSHA	모든 축 , 반원 기하 축 리포지셔닝	s	*PGAsl* 형상에 대해 위치 변경 (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [쪽 478]
REPOSL	직선 리포지셔닝	s	*PGAsl* 형상에 대해 위치 변경 (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [쪽 478]
REPOSQ	4 분원 리포지셔닝	s	*PGAsl* 형상에 대해 위치 변경 (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [쪽 478]
REPOSQA	전체 축 직선 리포지셔닝 , 기하 축은 4 분원으로	s	*PGAsl* 형상에 대해 위치 변경 (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [쪽 478]
RESET	고정 싸이클 리셋		*PGAsl* 해제 , 인에이블 , 리셋 (LOCK, UNLOCK, RESET) [쪽 633]
RESETMON	지령치 활성화를 위한 언어 명령		*FBW*
RET	서브프로그램 종료		*PGAsl* 파라미터 지정 가능한 서브프로그램 리턴 점프 (RET ...) [쪽 180]
RIC	넌모달 축 지정 상대적 반경 프로그래밍	s	*PGsl* 축 교체, 스핀들 교체 (RELEASE, GET, GETD) [쪽 131]
RINDEX	입력 문자열에서 문자 인덱스 정의		*PGAsl* 문자열에서 문자 / 문자열 검색 (INDEX, RINDEX, MINDEX, MATCH) [쪽 81]
RMB	블록 시작점으로 리포지셔닝	m	*PGAsl* 형상에 대해 위치 변경 (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [쪽 478]
RME	블록 끝점으로 리포지셔닝	m	*PGAsl* 형상에 대해 위치 변경 (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [쪽 478]
RMI [3]	인터럽트 지점으로 리포지셔닝	m	*PGAsl* 형상에 대해 위치 변경 (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [쪽 478]
RMN	가장 가까운 경로 지점으로 리포지셔닝	m	*PGAsl* 형상에 대해 위치 변경 (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) [쪽 478]
RND	형상 코너 라운딩	s	*PGsl*

작업	의미	W [1]	설명 참조 [2]
RNDM	모달 라운딩	m	*PGs/*
ROT	프로그래밍 회전	s	*PGs/*
ROTS	입체각 프로그래밍 프레임 회전	s	*PGs/*
ROUND	소수 자리수 반올림		*PGAs/* 산술 함수 [쪽 64]
ROUNDUP	입력 값 반올림		*PGAs/* 반올림 (ROUNDUP) [쪽 156]
RP	극점 반경	m/s	*PGs/*
RPL	평면에서 회전	s	*PGs/*
RT	프레임 데이터에 액세스하기 위한 파라미터 : 회전		*PGAs/* 프레임 구성 요소 읽기 및 변경 (TR, FI, RT, SC, MI) [쪽 294]
RTLIOF	직선 보간 없이 G0 이송 (단일 축 보간)	m	*PGs/*
RTLION	G0 로 직선 보간	m	*PGs/*

조작	의미	W [1]	설명 참조 [2]
S	스핀들 속도 (G4, G96/G961 에서는 다른 의미 사용)	m/s	*PGs/*
SAVE	서브프로그램이 호출될 때 정보를 저장하기 위한 속성		*PGAs/* 모달 G 코드 저장 (SAVE) [쪽 167]
SBLOF	싱글 블록 억제		*PGAs/* 싱글 블록 실행 억제 (SBLOF, SBLON) [쪽 168]
SBLON	싱글 블록 억제 취소		*PGAs/* 싱글 블록 실행 억제 (SBLOF, SBLON) [쪽 168]
SC	프레임 데이터에 액세스하기 위한 파라미터 : 스케일링		*PGAs/* 프레임 구성 요소 읽기 및 변경 (TR, FI, RT, SC, MI) [쪽 294]
SCALE	프로그래밍 배율	s	*PGs/*
SCC	G96/G961/G962 로의 가로 축의 선택적 지정 축 이름은 기하 , 채널 또는 기계 축의 형태를 가질 수 있습니다		*PGs/*
SCPARA	서보 파라미터 세트 프로그래밍		*PGAs/* 프로그램 가능한 서보 파라미터 세트 (SCPARA) [쪽 281]

조작	의미	W [1]	설명 참조 [2]
SD	스플라인 각도	s	*PGAsl* 스플라인 보간 (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) [쪽 240]
SEFORM	HMI Advanced 에서 단계 보기를 생성하기 위해 Step 편집기에서 연산 구조화		*PGAsl* 단계 편집기에서 구조화 명령 (SEFORM) [쪽 221]
SET	수록된 값으로 배열의 모든 요소를 초기화하는 키워드		*PGAsl* 배열 변수 정의 및 초기화 (DEF, SET, REP): [쪽 47]
SETAL	알람 설정		*PGAsl* 알람 (SETAL) [쪽 706]
SETDNO	공구 (T) 절삭날 (CE) 의 D 번호 지정		*PGAsl* D 번호의 자유 할당 : D 번호 이름 변경 (GETDNO, SETDNO) [쪽 434]
SETINT	NCK 입력이 있을 때 활성화될 인터럽트 루틴을 정의		*PGAsl* 인터럽트 루틴 지정 및 시작 (SETINT, PRIO, BLSYNC) [쪽 121]
SETM	전용 채널에서 마커 설정		PGAsl 프로그램 코디네이션 (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) [쪽 114]
SETMS	머신 데이터에서 정의된 마스터 스핀들로 리셋		
SETMS(n)	스핀들 n 을 마스터 스핀들로 설정		*PGsl*
SETMTH	마스터 공구 홀더 번호를 설정함		*FBW*
SETPIECE	스핀들에 할당된 모든 공구에 가공 수량 설정		*FBW*
SETTA	마모 그룹에서 공구 활성화		*FBW*
SETTCOR	모든 일반 조건을 고려한 공구 구성요소 수정		*FB1(W1)*
SETTIA	마모 그룹에서 공구 비활성화		*FBW*
SF	나사 절삭의 시작점 옵셋	m	*PGsl*
SIN	사인 (삼각 함수)		*PGAsl* 산술 함수 [쪽 64]
SIRELAY	SIRELIN, SIRELOUT 및 SIRELTIME 로 파라미터화된 안전 기능 활성화		*FBSIsl*
SIRELIN	펑션 블록의 입력 변수 초기화		*FBSIsl*
SIRELOUT	펑션 블록의 출력 변수 초기화		*FBSIsl*
SIRELTIME	펑션 블록의 타이머 초기화		*FBSIsl*
SLOT1	기술 싸이클 : 세로 홈		*PGAsl* 일반 슬롯 - SLOT1 [쪽 762]
SLOT2	기술 싸이클 : 원호 홈		*PGAsl* 원호 슬롯 - SLOT2 [쪽 765]

조작	의미	W [1]	설명 참조 [2]
SOFT	소프트 경로 가속	m	*PGs/*
SOFTA	프로그래밍된 축의 소프트 축 가속 활성화		*PGs/*
SON	니블링 ON	m	*PGAs/* 펀칭 및 니블링 활성화 / 비활성화 (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) [쪽 653]
SONS	보간 싸이클 내 니블링 ON	m	*PGAs/* 펀칭 및 니블링 활성화 / 비활성화 (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) [쪽 653]
SPATH [3]	FGROUP 축의 경로 기준은 호 길이임	m	*PGAs/* 설정 가능한 경로 기준 (SPATH, UPATH) [쪽 262]
SPCOF	스핀들을 마스터 스핀들로 또는 위치 제어에서 속도 제어로 전환	m	*PGs/*
SPCON	스핀들을 마스터 스핀들로 또는 위치 제어에서 속도 제어로 전환	m	*PGAs/*
SPI	스핀들 번호를 축 이름으로 변환		*PGAs/* 축 기능 (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) [쪽 667]
SPIF1 [3]	펀치 / 니블링 바이트 1 을 위한 빠른 NCK 입력 / 출력	m	*FB2(N4)*
SPIF2	펀치 / 니블링 바이트 2 를 위한 빠른 NCK 입력 / 출력	m	*FB2(N4)*
SPLINEPATH	스플라인 그룹화 정의		*PGAs/* 스플라인 그룹화 (SPLINEPATH) [쪽 251]
SPN	블록 당 경로 섹션 수	s	*PGAs/* 자동 경로 분할 [쪽 658]
SPOF [3]	스트로크 OFF, 니블링 , 펀칭 OFF	m	*PGAs/* 펀칭 및 니블링 활성화 / 비활성화 (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) [쪽 653]
SPOS	스핀들 위치	m	*PGs/*
SPOSA	블록 경계의 스핀들 위치	m	*PGs/*
SPP	경로 섹션의 길이	m	*PGAs/* 자동 경로 분할 [쪽 658]
SPRINT	포맷된 입력 문자열 반환		*PGAs/* 문자열 포맷 (SPRINT) [쪽 84]
SQRT	제곱 근 (산술 함수)		*PGAs/* 산술 함수 [쪽 64]

조작	의미	W [1]	설명 참조 [2]
SR	동기 동작을 위한 오실레이션 후퇴 경로	s	*PGsl*
SRA	동기 동작을 위한 외부 입력 축을 포함한 오실레이션 후퇴 경로	m	*PGsl*
ST	동기 동작을 위한 오실레이션 스파크아웃 시간	s	*PGsl*
STA	동기 동작을 위한 오실레이션 스파크아웃 시간 축	m	*PGsl*
START	현 프로그램의 여러 개의 채널에서 선택된 프로그램 동시 시작		*PGAsl* 프로그램 코디네이션 (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) [쪽 114]
STARTFIFO [3]	실행과 동시에 전처리 메모리 채움	m	*PGAsl* 전처리 메모리를 사용한 프로그램 순서 (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE) [쪽 468]
STAT	조인트의 위치	s	*PGAsl* 직교 PTP 이송 [쪽 380]
STOLF	G0 공차 팩터	m	*PGAsl* G0 모션 공차 (STOLF) [쪽 495]
STOPFIFO	가공 정지 ; STARTFIFO 가 탐지되거나 , 전처리 메모리가 채워지거나 프로그램이 종료될 때까지 전처리 메모리를 채웁니다 .	m	*PGAsl* 전처리 메모리를 사용한 프로그램 순서 (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE) [쪽 468]
STOPRE	주 실행에서 준비된 블록이 모두 실행될 때까지 전처리 정지		*PGAsl* 전처리 메모리를 사용한 프로그램 순서 (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE) [쪽 468]
STOPREOF	전처리 정지 취소		*PGAsl* 전처리 중지 취소 (STOPREOF) [쪽 580]
STRING	데이터 유형 : 문자열		*PGAsl* 사용자 변수 정의 (DEF) [쪽 25]
STRINGFELD	프로그래밍 문자열 필드에서 단일 문자 선택		*PGAsl* 단일 문자 선택 (STRINGVAR, STRINGFELD) [쪽 83]
STRINGIS	NC 언어의 현재 범위를 비롯하여 특히 이 명령에 속하는 NC 싸이클 이름 , 사용자 변수 , 매크로 및 레이블 이름을 검사하여 이들 항목의 존재 여부 , 유효성 , 정의되어 있는지 또는 활성 상태인지 여부를 확인함		*PGAsl* NC 언어 (STRINGIS) 의 범위 확인 [쪽 682]
STRINGVAR	프로그래밍 문자열에서 단일 문자 선택		*PGAsl* 단일 문자 선택 (STRINGVAR, STRINGFELD) [쪽 83]
STRLEN	문자열 길이 정의		*PGAsl* 문자열 길이 결정 (STRLEN) [쪽 80]
SUBSTR	입력 문자열에서 문자 인덱스 정의		*PGAsl* 하위 문자열 선택 (SUBSTR) [쪽 82]

조작	의미	W [1]	설명 참조 [2]
SUPA	프로그래밍된 옵셋 , 시스템 프레임 , 핸드휠 옵셋 (DRF), 외부 워크 옵셋 및 오버레이 이동을 포함한 현재의 워크 옵셋 억제	s	*PGs/*
SVC	공구 절삭 속도	m	*PGs/*
SYNFCT	모션 동기 동작의 조건 함수로서 폴리노미얼 평가		*PGAs/* 동기화 기능 (SYNFCT) [쪽 585]
SYNR	변수는 실행과 동시에 읽혀집니다 .		*PGAs/* 사용자 변수 정의 (DEF) [쪽 25]
SYNRW	변수는 실행과 동시에 작성 및 읽혀집니다 .		*PGAs/* 사용자 변수 정의 (DEF) [쪽 25]
SYNW	변수는 실행과 동시에 작성됩니다 .		*PGAs/* 사용자 변수 정의 (DEF) [쪽 25]
T	공구 호출 (머신 데이터에 지정될 때만 변경 . 그렇지 않으면 M6 명령어가 필요함)		*PGs/*
TAN	접선 (삼각 함수)		*PGAs/* 산술 함수 [쪽 64]
TANG	축 그룹화 접선 보정 정의		*PGAs/* 접선 제어 (TANG, TANGON, TANGOF, TLIFT, TANGDEL) [쪽 455]
TANGDEL	축 그룹화 접선 보정에 대한 정의 삭제		*PGAs/* 접선 제어 (TANG, TANGON, TANGOF, TLIFT, TANGDEL) [쪽 455]
TANGOF	접선 보정 OFF		*PGAs/* 접선 제어 (TANG, TANGON, TANGOF, TLIFT, TANGDEL) [쪽 455]
TANGON	접선 보정 ON		*PGAs/* 접선 제어 (TANG, TANGON, TANGOF, TLIFT, TANGDEL) [쪽 455]
TCA (828D: _TCA)	공구 상태와 관계 없이 공구 선택 / 공구 교환		*FBW*
TCARR	공구 홀더 요청 (번호 "m")		*PGAs/* 방향 조정 가능한 공구 홀더에 대한 공구 길이 보정 (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) [쪽 443]
TCI	버퍼에서 메거진으로 공구 로드		*FBW*
TCOABS[3]	현재 공구 홀더의 오리엔테이션에서 공구 길이 요소 결정	m	*PGAs/* 방향 조정 가능한 공구 홀더에 대한 공구 길이 보정 (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) [쪽 443]
TCOFR	활성화된 프레임의 오리엔테이션에서 공구 길이 요소 결정	m	*PGAs/* 방향 조정 가능한 공구 홀더에 대한 공구 길이 보정 (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) [쪽 443]

조작	의미	W [1)]	설명 참조 [2)]
TCOFRX	공구 선택 시 활성 프레임의 공구 오리엔테이션을 결정 (공구 지점 X 방향)	m	*PGAs/* 방향 조정 가능한 공구 홀더에 대한 공구 길이 보정 (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) [쪽 443]
TCOFRY	공구 선택 시 활성 프레임의 공구 오리엔테이션을 결정 (공구 지점 Y 방향)	m	*PGAs/* 방향 조정 가능한 공구 홀더에 대한 공구 길이 보정 (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) [쪽 443]
TCOFRZ	공구 선택 시 활성 프레임의 공구 오리엔테이션을 결정 (공구 지점 Z 방향)	m	*PGAs/* 방향 조정 가능한 공구 홀더에 대한 공구 길이 보정 (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) [쪽 443]
THETA	회전 각도	s	*PGAs/* 공구 오리엔테이션 회전 (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) [쪽 348]
TILT	틸트 각도	m	*PGAs/* 공구 방향 프로그램 (A..., B..., C..., LEAD, TILT) [쪽 327]
TLIFT	접선 제어에서 형상 코너에 중간 블록 삽입		*PGAs/* 접선 제어 (TANG, TANGON, TANGOF, TLIFT, TANGDEL) [쪽 455]
TMOF	공구 모니터링 선택을 해제		*PGAs/* 가공 프로그램에서 연삭 관련 도구 모니터링 (TMON, TMOF) [쪽 665]
TMON	공구 모니터링 활성화		*PGAs/* 가공 프로그램에서 연삭 관련 도구 모니터링 (TMON, TMOF) [쪽 665]
TO	FOR 카운터 루프에서 끝 값 지정		*PGAs/* 카운트 루프 (FOR ... TO ..., ENDFOR) [쪽 109]
TOFF	유효한 공구 길이 콤포넌트 방향에서의 공구 길이 옵셋은 색인에서 지정한 기하 축에 평행함	m	*PGs/*
TOFFL	공구 길이 콤포넌트 L1, L2 또는 L3 방향에서 공구 길이 옵셋	m	*PGs/*
TOFFOF	온라인 공구 옵셋 비활성화		*PGAs/* 온라인 공구 길이 보정 (TOFFON, TOFFOF) [쪽 446]
TOFFON	온라인 공구 길이 옵셋 활성화		*PGAs/* 온라인 공구 길이 보정 (TOFFON, TOFFOF) [쪽 446]
TOFFR	공구 반경 옵셋	m	*PGs/*
TOFRAME	프레임을 회전시켜 공작물 좌표계의 Z축을 공작물 오리엔테이션과 평행하게 정렬함	m	*PGs/*
TOFRAMEX	프레임을 회전시켜 공작물 좌표계의 X축을 공작물 오리엔테이션과 평행하게 정렬함	m	*PGs/*

조작	의미	W [1]	설명 참조 [2]
TOFRAMEY	프레임을 회전시켜 공작물 좌표계의 Y축을 공작물 오리엔테이션과 평행하게 정렬함	m	*PGs/*
TOFRAMEZ	TOFRAME 와 동일	m	*PGs/*
TOLOWER	문자열의 글자를 소문자로 변환		*PGAs/* 소문자/대문자로 변환 (TOLOWER, TOUPPER) [쪽 79]
TOOLENV	메모리에 저장된 공구 데이터 평가에 중요한 현재 상태를 저장		*FB1(W1)*
TOROT	프레임을 회전시켜 공작물 좌표계의 Z축을 공작물 오리엔테이션과 평행하게 정렬함	m	*PGs/*
TOROTOF	공구 방향에서 프레임 회전 OFF	m	*PGs/*
TOROTX	프레임을 회전시켜 공작물 좌표계의 X축을 공작물 오리엔테이션과 평행하게 정렬함	m	*PGs/*
TOROTY	프레임을 회전시켜 공작물 좌표계의 Y축을 공작물 오리엔테이션과 평행하게 정렬함	m	*PGs/*
TOROTZ	TOROT 와 동일	m	*PGs/*
TOUPPER	문자열의 글자를 대문자로 변환		*PGAs/* 소문자/대문자로 변환 (TOLOWER, TOUPPER) [쪽 79]
TOWBCS	기본 좌표계 (BCS) 의 마모값	m	*PGAs/* 활성화된 가공 작업의 좌표계 (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) [쪽 404]
TOWKCS	좌표계 변환을 위한 공구 헤드의 좌표계에서 마모 값 (공구 회전을 통한 기계 좌표계와 다름)	m	*PGAs/* 활성화된 가공 작업의 좌표계 (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) [쪽 404]
TOWMCS	기계 좌표계에서 마모 값	m	*PGAs/* 활성화된 가공 작업의 좌표계 (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) [쪽 404]
TOWSTD	공구 길이 옵셋의 초기 설정 값	m	*PGAs/* 활성화된 가공 작업의 좌표계 (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) [쪽 404]
TOWTCS	공구 좌표계에서 마모값 (공구 홀더에서의 공구 홀더 기준점 T)	m	*PGAs/* 활성화된 가공 작업의 좌표계 (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) [쪽 404]
TOWWCS	공작물 좌표계에서의 마모 값	m	*PGAs/* 활성화된 가공 작업의 좌표계 (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) [쪽 404]
TR	프레임 변수의 옵셋 콤포넌트		*PGAs/* 프레임 구성 요소 읽기 및 변경 (TR, FI, RT, SC, MI) [쪽 294]

조작	의미	W [1]	설명 참조 [2]
TRAANG	경사 축 변환		*PGAs/* 경사 축 (TRAANG) [쪽 375]
TRACON	계단식 변환		*PGAs/* 연결된 변환 (TRACON, TRAFOOF) [쪽 391]
TRACYL	실린더 . 원주면 변환		*PGAs/* 원통 표면 변환 (TRACYL) [쪽 367]
TRAFOOF	채널에서 활성 변환 비활성화		*PGAs/* 연결된 변환 (TRACON, TRAFOOF) [쪽 391]
TRAILOF	비동기 커플 모션 OFF		*PGAs/* 결합 작동 (TRAILON, TRAILOF) [쪽 497]
TRAILON	비동기 커플 모션 ON		*PGAs/* 결합 작동 (TRAILON, TRAILOF) [쪽 497]
TRANS	프로그램 옵셋	s	*PGs/*
TRANSMIT	극 변환 (면 가공)		*PGAs/* 선삭 가공된 부품에 대한 밀링 (TRANSMIT) [쪽 363]
TRAORI	4- 축 , 5- 축 변환 , 일반 변환		*PGAs/* 3 축 , 4 축 및 5 축 변환 (TRAORI) [쪽 324]
TRUE	논리 상수 : 참		*PGAs/* 사용자 변수 정의 (DEF) [쪽 25]
TRUNC	소수 자릿수 잘라내기		*PGAs/* 비교 오류에 대한 정밀도 수정 (TRUNC) [쪽 69]
TU	축 각도	s	*PGAs/* 직교 PTP 이송 [쪽 380]
TURN	나선형의 회전 수	s	*PGs/*
ULI	변수의 상한값		*PGAs/* 속성 : 한계값 (LLI, ULI) [쪽 37]
UNLOCK	ID 로 동기 동작 활성화 (고정 싸이클 지속)		*PGAs/* 해제 , 인에이블 , 리셋 (LOCK, UNLOCK, RESET) [쪽 633]
UNTIL	REPEAT 루프 끝의 조건		*PGAs/* 루프 시작 시 조건을 가진 프로그램 루프 (WHILE, ENDWHILE) [쪽 111]
UPATH	FGROUP 축의 경로 기준은 곡선 파라미터	m	*PGAs/* 설정 가능한 경로 기준 (SPATH, UPATH) [쪽 262]
VAR	키워드 : 파라미터 전송 유형		*PGAs/* 파라미터 전송을 포함한 서브프로그램 호출 (EXTERN) [쪽 188]
VELOLIM	최대 축 속도 감소	m	*PGAs/* 백분율 속도 수정 (VELOLIM) [쪽 488]
VELOLIMA	추종 축의 최대 축 속도 감소 또는 증가	m	*PGs/*

조작	의미	W [1]	설명 참조 [2]
WAITC	축/스핀들에 대해 충족시켜야 할 커플링 블록 변경 기준 대기		*PGAs/* 프로그램 코디네이션 (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) [쪽 114]
WAITE	다른 채널의 프로그램 종료까지 대기		*PGAs/* 프로그램 코디네이션 (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) [쪽 114]
WAITENC	동기 축 위치 또는 복구된 축 위치 대기		*PGAs/* 유효 축 위치 대기 (WAITENC) [쪽 681]
WAITM	지정된 채널에서 마커 대기 . 정위치 정지로 이전 블록 종료		*PGAs/* 프로그램 코디네이션 (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) [쪽 114]
WAITMC	지정된 채널에서 마커 대기. 다른 채널이 마커에 도달하지 않았을 경우에만 정위치 정지		*PGAs/* 프로그램 코디네이션 (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) [쪽 114]
WAITP	포지셔닝 축의 이송 종료 대기		*PGs/*
WAITS	스핀들 위치에 도달할 때까지 대기		*PGs/*
WALCS0	공작물 좌표계 작업 영역 제한 선택이 취소됨	m	*PGs/*
WALCS1	공작물 좌표계 작업 영역 제한 그룹 1 활성 상태	m	*PGs/*
WALCS2	공작물 좌표계 작업 영역 제한 그룹 2 활성 상태	m	*PGs/*
WALCS3	공작물 좌표계 작업 영역 제한 그룹 3 활성 상태	m	*PGs/*
WALCS4	공작물 좌표계 작업 영역 제한 그룹 4 활성 상태	m	*PGs/*
WALCS5	공작물 좌표계 작업 영역 제한 그룹 5 활성 상태	m	*PGs/*
WALCS6	공작물 좌표계 작업 영역 제한 그룹 6 활성 상태	m	*PGs/*
WALCS7	공작물 좌표계 작업 영역 제한 그룹 7 활성 상태	m	*PGs/*
WALCS8	공작물 좌표계 작업 영역 제한 그룹 8 활성 상태	m	*PGs/*
WALCS9	공작물 좌표계 작업 영역 제한 그룹 9 활성 상태	m	*PGs/*
WALCS10	공작물 좌표계 작업 영역 제한 그룹 10 활성 상태	m	*PGs/*
WALIMOF	BCS 작업 영역 제한 OFF	m	*PGs/*

조작	의미	W [1]	설명 참조 [2]
WALIMON [3]	BCS 작업 영역 제한 ON	m	*PGs/*
WHEN	조건을 충족했을 때 동작이 주기적으로 실행됨		*PGAs/* 주기적으로 조건 검사 (WHEN, WHENEVER, FROM, EVERY) [쪽 555]
WHENEVER	조건이 충족될 때마다 동작이 한 번씩 실행됨		*PGAs/* 주기적으로 조건 검사 (WHEN, WHENEVER, FROM, EVERY) [쪽 555]
WHILE	WHILE 프로그램 루프 시작		*PGAs/* 루프 시작 시 조건을 가진 프로그램 루프 (WHILE, ENDWHILE) [쪽 111]
WRITE	파일 시스템에 텍스트 기록 지정된 파일 끝에 블록 부가		*PGAs/* 파일 쓰기 (WRITE) [쪽 138]
WRTPR	연속 경로 모드를 중단하지 않고 가공 작업 지연		*PGAs/*
X	축 이름	m/s	*PGs/*
XOR	로직 배타적 OR		*PGAs/* 비교 및 논리 연산자 [쪽 67]
Y	축 이름	m/s	*PGs/*
Z	축 이름	m/s	*PGs/*

17.2 작업 SINUMERIK 828D 에서 가용성

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
:	●	●	●	●	●	●
*	●	●	●	●	●	●
+	●	●	●	●	●	●
-	●	●	●	●	●	●
<	●	●	●	●	●	●
<<	●	●	●	●	●	●
<=	●	●	●	●	●	●
=	●	●	●	●	●	●
>=	●	●	●	●	●	●
/	●	●	●	●	●	●
/0 … … /7	● ○	● ○	● ○	● ○	● ○	● ○
A	●	●	●	●	●	●
A2	-	-	-	-	-	-
A3	-	-	-	-	-	-
A4	-	-	-	-	-	-
A5	-	-	-	-	-	-
ABS	●	●	●	●	●	●
AC	●	●	●	●	●	●
ACC	●	●	●	●	●	●
ACCLIMA	●	●	●	●	●	●
ACN	●	●	●	●	●	●
ACOS	●	●	●	●	●	●
ACP	●	●	●	●	●	●
ACTBLOCNO	●	●	●	●	●	●
ADDFRAME	●	●	●	●	●	●
ADIS	●	●	●	●	●	●
ADISPOS	●	●	●	●	●	●
ADISPOSA	●	●	●	●	●	●
ALF	●	●	●	●	●	●
AMIRROR	●	●	●	●	●	●
AND	●	●	●	●	●	●
ANG	●	●	●	●	●	●
AP	●	●	●	●	●	●
APR	●	●	●	●	●	●
APRB	●	●	●	●	●	●

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
APRP	●	●	●	●	●	●
APW	●	●	●	●	●	●
APWB	●	●	●	●	●	●
APWP	●	●	●	●	●	●
APX	●	●	●	●	●	●
AR	●	●	●	●	●	●
AROT	●	●	●	●	●	●
AROTS	●	●	●	●	●	●
AS	●	●	●	●	●	●
ASCALE	●	●	●	●	●	●
ASIN	●	●	●	●	●	●
ASPLINE	-	○	-	○	-	○
ATAN2	●	●	●	●	●	●
ATOL	-	●	-	●	-	●
ATRANS	●	●	●	●	●	●
AX	●	●	●	●	●	●
AXCTSWE	-	-	-	-	-	-
AXCTSWEC	-	-	-	-	-	-
AXCTSWED	-	-	-	-	-	-
AXIS	●	●	●	●	●	●
AXNAME	●	●	●	●	●	●
AXSTRING	●	●	●	●	●	●
AXTOCHAN	●	●	●	●	●	●
AXTOSPI	●	●	●	●	●	●
B	●	●	●	●	●	●
B2	-	-	-	-	-	-
B3	-	-	-	-	-	-
B4	-	-	-	-	-	-
B5	-	-	-	-	-	-
B_AND	●	●	●	●	●	●
B_OR	●	●	●	●	●	●
B_NOT	●	●	●	●	●	●
B_XOR	●	●	●	●	●	●
BAUTO	-	○	-	○	-	○
BLOCK	●	●	●	●	●	●
BLSYNC	●	●	●	●	●	●
BNAT	-	○	-	○	-	○
BOOL	●	●	●	●	●	●
BOUND	●	●	●	●	●	●

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
BRISK	●	●	●	●	●	●
BRISKA	●	●	●	●	●	●
BSPLINE	-	○	-	○	-	○
BTAN	-	○	-	○	-	○
C	●	●	●	●	●	●
C2	-	-	-	-	-	-
C3	-	-	-	-	-	-
C4	-	-	-	-	-	-
C5	-	-	-	-	-	-
CAC	●	●	●	●	●	●
CACN	●	●	●	●	●	●
CACP	●	●	●	●	●	●
CALCDAT	●	●	●	●	●	●
CALCPOSI	●	●	●	●	●	●
CALL	●	●	●	●	●	●
CALLPATH	●	●	●	●	●	●
CANCEL	●	●	●	●	●	●
CASE	●	●	●	●	●	●
CDC	●	●	●	●	●	●
CDOF	●	●	●	●	●	●
CDOF2	●	●	●	●	●	●
CDON	●	●	●	●	●	●
CFC	●	●	●	●	●	●
CFIN	●	●	●	●	●	●
CFINE	●	●	●	●	●	●
CFTCP	●	●	●	●	●	●
CHAN	●	●	●	●	●	●
CHANDATA	●	●	●	●	●	●
CHAR	●	●	●	●	●	●
CHECKSUM	●	●	●	●	●	●
CHF	●	●	●	●	●	●
CHKDM	●	●	●	●	●	●
CHKDNO	●	●	●	●	●	●
CHR	●	●	●	●	●	●
CIC	●	●	●	●	●	●
CIP	●	●	●	●	●	●
CLEARM	-	-	-	-	-	-
CLRINT	●	●	●	●	●	●
CMIRROR	●	●	●	●	●	●

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
COARSEA	●	●	●	●	●	●
COMPCAD	-	○	-	○	-	○
COMPCURV	-	○	-	○	-	○
COMPLETE	●	●	●	●	●	●
COMPOF	-	○	-	○	-	○
COMPON	-	○	-	○	-	○
CONTDCON	●	●	●	●	●	●
CONTPRON	●	●	●	●	●	●
CORROF	●	●	●	●	●	●
COS	●	●	●	●	●	●
COUPDEF	○	-	○	-	○	-
COUPDEL	○	-	○	-	○	-
COUPOF	○	-	○	-	○	-
COUPOFS	○	-	○	-	○	-
COUPON	○	-	○	-	○	-
COUPONC	○	-	○	-	○	-
COUPRES	○	-	○	-	○	-
CP	●	●	●	●	●	●
CPRECOF	●	●	●	●	●	●
CPRECON	●	●	●	●	●	●
CPROT	●	●	●	●	●	●
CPROTDEF	●	●	●	●	●	●
CR	●	●	●	●	●	●
CROT	●	●	●	●	●	●
CROTS	●	●	●	●	●	●
CRPL	●	●	●	●	●	●
CSCALE	●	●	●	●	●	●
CSPLINE	-	○	-	○	-	○
CT	●	●	●	●	●	●
CTAB	-	-	-	-	-	-
CTABDEF	-	-	-	-	-	-
CTABDEL	-	-	-	-	-	-
CTABEND	-	-	-	-	-	-
CTABEXISTS	-	-	-	-	-	-
CTABFNO	-	-	-	-	-	-
CTABFPOL	-	-	-	-	-	-
CTABFSEG	-	-	-	-	-	-
CTABID	-	-	-	-	-	-
CTABINV	-	-	-	-	-	-

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
CTABISLOCK	-	-	-	-	-	-
CTABLOCK	-	-	-	-	-	-
CTABMEMTYP	-	-	-	-	-	-
CTABMPOL	-	-	-	-	-	-
CTABMSEG	-	-	-	-	-	-
CTABNO	-	-	-	-	-	-
CTABNOMEM	-	-	-	-	-	-
CTABPERIOD	-	-	-	-	-	-
CTABPOL	-	-	-	-	-	-
CTABPOLID	-	-	-	-	-	-
CTABSEG	-	-	-	-	-	-
CTABSEGID	-	-	-	-	-	-
CTABSEV	-	-	-	-	-	-
CTABSSV	-	-	-	-	-	-
CTABTEP	-	-	-	-	-	-
CTABTEV	-	-	-	-	-	-
CTABTMAX	-	-	-	-	-	-
CTABTMIN	-	-	-	-	-	-
CTABTSP	-	-	-	-	-	-
CTABTSV	-	-	-	-	-	-
CTABUNLOCK	-	-	-	-	-	-
CTOL	-	○	-	○	-	○
CTRANS	●	●	●	●	●	●
CUT2D	●	●	●	●	●	●
CUT2DF	●	●	●	●	●	●
CUT3DC	-	-	-	-	-	-
CUT3DCC	-	-	-	-	-	-
CUT3DCCD	-	-	-	-	-	-
CUT3DF	-	-	-	-	-	-
CUT3DFF	-	-	-	-	-	-
CUT3DFS	-	-	-	-	-	-
CUTCONOF	●	●	●	●	●	●
CUTCONON	●	●	●	●	●	●
CUTMOD	●	●	●	●	●	●
CYCLE...	●	●	●	●	●	●
D	●	●	●	●	●	●
D0	●	●	●	●	●	●
DAC	●	●	●	●	●	●
DC	●	●	●	●	●	●

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
DEF	●	●	●	●	●	●
DEFINE	●	●	●	●	●	●
DEFAULT	●	●	●	●	●	●
DELAYFSTON	●	●	●	●	●	●
DELAYFSTOF	●	●	●	●	●	●
DELDL	●	●	●	●	●	●
DELDTG	●	●	●	●	●	●
DELETE	●	●	●	●	●	●
DELTOOLENV	●	●	●	●	●	●
DIACYCOFA	●	●	●	●	●	●
DIAM90	●	●	●	●	●	●
DIAM90A	●	●	●	●	●	●
DIAMCHAN	●	●	●	●	●	●
DIAMCHANA	●	●	●	●	●	●
DIAMCYCOF	●	●	●	●	●	●
DIAMOF	●	●	●	●	●	●
DIAMOFA	●	●	●	●	●	●
DIAMON	●	●	●	●	●	●
DIAMONA	●	●	●	●	●	●
DIC	●	●	●	●	●	●
DILF	●	●	●	●	●	●
DISABLE	●	●	●	●	●	●
DISC	●	●	●	●	●	●
DISCL	●	●	●	●	●	●
DISPLOF	●	●	●	●	●	●
DISPLON	●	●	●	●	●	●
DISPR	●	●	●	●	●	●
DISR	●	●	●	●	●	●
DITE	●	●	●	●	●	●
DITS	●	●	●	●	●	●
DIV	●	●	●	●	●	●
DL	-	-	-	-	-	-
DO	●	●	●	●	●	●
DRFOF	●	●	●	●	●	●
DRIVE	●	●	●	●	●	●
DRIVEA	●	●	●	●	●	●
DYNFINISH	●	●	●	●	●	●
DYNNORM	●	●	●	●	●	●
DYNPOS	●	●	●	●	●	●

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
DYNROUGH	●	●	●	●	●	●
DYNSEMIFIN	●	●	●	●	●	●
DZERO	●	●	●	●	●	●
EAUTO	-	○	-	○	-	○
EGDEF	-	-	-	-	-	-
EGDEL	-	-	-	-	-	-
EGOFC	-	-	-	-	-	-
EGOFS	-	-	-	-	-	-
EGON	-	-	-	-	-	-
EGONSYN	-	-	-	-	-	-
EGONSYNE	-	-	-	-	-	-
ELSE	●	●	●	●	●	●
ENABLE	●	●	●	●	●	●
ENAT	-	○	-	○	-	○
ENDFOR	●	●	●	●	●	●
ENDIF	●	●	●	●	●	●
ENDLABEL	●	●	●	●	●	●
ENDLOOP	●	●	●	●	●	●
ENDPROC	●	●	●	●	●	●
ENDWHILE	●	●	●	●	●	●
ESRR	●	●	●	●	●	●
ESRS	●	●	●	●	●	●
ETAN	-	○	-	○	-	○
EVERY	●	●	●	●	●	●
EX	●	●	●	●	●	●
EXECSTRING	●	●	●	●	●	●
EXECTAB	●	●	●	●	●	●
EXECUTE	●	●	●	●	●	●
EXP	●	●	●	●	●	●
EXTCALL	●	●	●	●	●	●
EXTCLOSE	●	●	●	●	●	●
EXTERN	●	●	●	●	●	●
EXTOPEN	●	●	●	●	●	●
F	●	●	●	●	●	●
FA	●	●	●	●	●	●
FAD	●	●	●	●	●	●
FALSE	●	●	●	●	●	●
FB	●	●	●	●	●	●
FCTDEF	-	-	-	-	-	-

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
FCUB	●	●	●	●	●	●
FD	●	●	●	●	●	●
FDA	●	●	●	●	●	●
FENDNORM	●	●	●	●	●	●
FFWOF	●	●	●	●	●	●
FFWON	●	●	●	●	●	●
FGREF	●	●	●	●	●	●
FGROUP	●	●	●	●	●	●
FI	●	●	●	●	●	●
FIFOCTRL	●	●	●	●	●	●
FILEDATE	●	●	●	●	●	●
FILEINFO	●	●	●	●	●	●
FILESIZE	●	●	●	●	●	●
FILESTAT	●	●	●	●	●	●
FILETIME	●	●	●	●	●	●
FINEA	●	●	●	●	●	●
FL	●	●	●	●	●	●
FLIN	●	●	●	●	●	●
FMA	-	-	-	-	-	-
FNORM	●	●	●	●	●	●
FOCOF	○	-	○	-	○	-
FOCON	○	-	○	-	○	-
FOR	●	●	●	●	●	●
FP	●	●	●	●	●	●
FPO	-	-	-	-	-	-
FPR	●	●	●	●	●	●
FPRAOF	●	●	●	●	●	●
FPRAON	●	●	●	●	●	●
FRAME	●	●	●	●	●	●
FRC	●	●	●	●	●	●
FRCM	●	●	●	●	●	●
FROM	●	●	●	●	●	●
FTOC	●	●	●	●	●	●
FTOCOF	●	●	●	●	●	●
FTOCON	●	●	●	●	●	●
FXS	●	●	●	●	●	●
FXST	●	●	●	●	●	●
FXSW	●	●	●	●	●	●
FZ	●	●	●	●	●	●

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
G0	●	●	●	●	●	●
G1	●	●	●	●	●	●
G2	●	●	●	●	●	●
G3	●	●	●	●	●	●
G4	●	●	●	●	●	●
G5	●	●	●	●	●	●
G7	●	●	●	●	●	●
G9	●	●	●	●	●	●
G17	●	●	●	●	●	●
G18	●	●	●	●	●	●
G19	●	●	●	●	●	●
G25	●	●	●	●	●	●
G26	●	●	●	●	●	●
G33	●	●	●	●	●	●
G34	●	●	●	●	●	●
G35	●	●	●	●	●	●
G40	●	●	●	●	●	●
G41	●	●	●	●	●	●
G42	●	●	●	●	●	●
G53	●	●	●	●	●	●
G54	●	●	●	●	●	●
G55	●	●	●	●	●	●
G56	●	●	●	●	●	●
G57	●	●	●	●	●	●
G58	●	●	●	●	●	●
G59	●	●	●	●	●	●
G60	●	●	●	●	●	●
G62	●	●	●	●	●	●
G63	●	●	●	●	●	●
G64	●	●	●	●	●	●
G70	●	●	●	●	●	●
G71	●	●	●	●	●	●
G74	●	●	●	●	●	●
G75	●	●	●	●	●	●
G90	●	●	●	●	●	●
G91	●	●	●	●	●	●
G93	●	●	●	●	●	●
G94	●	●	●	●	●	●
G95	●	●	●	●	●	●

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
G96	●	●	●	●	●	●
G97	●	●	●	●	●	●
G110	●	●	●	●	●	●
G111	●	●	●	●	●	●
G112	●	●	●	●	●	●
G140	●	●	●	●	●	●
G141	●	●	●	●	●	●
G142	●	●	●	●	●	●
G143	●	●	●	●	●	●
G147	●	●	●	●	●	●
G148	●	●	●	●	●	●
G153	●	●	●	●	●	●
G247	●	●	●	●	●	●
G248	●	●	●	●	●	●
G290	●	●	●	●	●	●
G291	●	●	●	●	●	●
G331	●	●	●	●	●	●
G332	●	●	●	●	●	●
G340	●	●	●	●	●	●
G341	●	●	●	●	●	●
G347	●	●	●	●	●	●
G348	●	●	●	●	●	●
G450	●	●	●	●	●	●
G451	●	●	●	●	●	●
G460	●	●	●	●	●	●
G461	●	●	●	●	●	●
G462	●	●	●	●	●	●
G500	●	●	●	●	●	●
G505 ...G599	●	●	●	●	●	●
G601	●	●	●	●	●	●
G602	●	●	●	●	●	●
G603	●	●	●	●	●	●
G621	●	●	●	●	●	●
G641	●	●	●	●	●	●
G642	●	●	●	●	●	●
G643	●	●	●	●	●	●
G644	●	●	●	●	●	●
G645	●	●	●	●	●	●
G700	●	●	●	●	●	●

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
G710	●	●	●	●	●	●
G751	●	●	●	●	●	●
G810 ...G819	-	-	-	-	-	-
G820 ...G829	-	-	-	-	-	-
G931	●	●	●	●	●	●
G942	●	●	●	●	●	●
G952	●	●	●	●	●	●
G961	●	●	●	●	●	●
G962	●	●	●	●	●	●
G971	●	●	●	●	●	●
G972	●	●	●	●	●	●
G973	●	●	●	●	●	●
GEOAX	●	●	●	●	●	●
GET	●	●	●	●	●	●
GETACTT	●	●	●	●	●	●
GETACTTD	●	●	●	●	●	●
GETD	●	●	●	●	●	●
GETDNO	●	●	●	●	●	●
GETEXET	●	●	●	●	●	●
GETFREELOC	●	●	●	●	●	●
GETSELT	●	●	●	●	●	●
GETT	●	●	●	●	●	●
GETTCOR	●	●	●	●	●	●
GETTENV	●	●	●	●	●	●
GOTO	●	●	●	●	●	●
GOTOB	●	●	●	●	●	●
GOTOC	●	●	●	●	●	●
GOTOF	●	●	●	●	●	●
GOTOS	●	●	●	●	●	●
GP	●	●	●	●	●	●
GWPSOF	●	●	●	●	●	●
GWPSON	●	●	●	●	●	●
H...	●	●	●	●	●	●
HOLES1	●	●	●	●	●	●
HOLES2	●	●	●	●	●	●
I	●	●	●	●	●	●
I1	●	●	●	●	●	●
IC	●	●	●	●	●	●
ICYCOF	●	●	●	●	●	●

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
ICYCON	●	●	●	●	●	●
ID	●	●	●	●	●	●
IDS	●	●	●	●	●	●
IF	●	●	●	●	●	●
INDEX	●	●	●	●	●	●
INIPO	●	●	●	●	●	●
INIRE	●	●	●	●	●	●
INICF	●	●	●	●	●	●
INIT	-	-	-	-	-	-
INITIAL	●	●	●	●	●	●
INT	●	●	●	●	●	●
INTERSEC	●	●	●	●	●	●
INVCCW	-	-	-	-	-	-
INVCW	-	-	-	-	-	-
INVFRAME	●	●	●	●	●	●
IP	●	●	●	●	●	●
IPOBRKA	●	●	●	●	●	●
IPOENDA	●	●	●	●	●	●
IPTRLOCK	●	●	●	●	●	●
IPTRUNLOCK	●	●	●	●	●	●
ISAXIS	●	●	●	●	●	●
ISD	-	-	-	-	-	-
ISFILE	●	●	●	●	●	●
ISNUMBER	●	●	●	●	●	●
ISOCALL	●	●	●	●	●	●
ISVAR	●	●	●	●	●	●
J	●	●	●	●	●	●
J1	●	●	●	●	●	●
JERKA	●	●	●	●	●	●
JERKLIM	●	●	●	●	●	●
JERKLIMA	●	●	●	●	●	●
K	●	●	●	●	●	●
K1	●	●	●	●	●	●
KONT	●	●	●	●	●	●
KONTC	●	●	●	●	●	●
KONTT	●	●	●	●	●	●
L	●	●	●	●	●	●

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
LEAD 공구 오리엔테이션 오리엔테이션 폴리노미얼	- -	- -	- -	- -	- -	- -
LEADOF	-	-	-	-	-	-
LEADON	-	-	-	-	-	-
LENTOAX	●	●	●	●	●	●
LFOF	●	●	●	●	●	●
LFON	●	●	●	●	●	●
LFPOS	●	●	●	●	●	●
LFTXT	●	●	●	●	●	●
LFWP	●	●	●	●	●	●
LIFTFAST	●	●	●	●	●	●
LIMS	●	●	●	●	●	●
LLI	●	●	●	●	●	●
LN	●	●	●	●	●	●
LOCK	●	●	●	●	●	●
LONGHOLE	-	-	-	-	-	-
LOOP	●	●	●	●	●	●
M0	●	●	●	●	●	●
M1	●	●	●	●	●	●
M2	●	●	●	●	●	●
M3	●	●	●	●	●	●
M4	●	●	●	●	●	●
M5	●	●	●	●	●	●
M6	●	●	●	●	●	●
M17	●	●	●	●	●	●
M19	●	●	●	●	●	●
M30	●	●	●	●	●	●
M40	●	●	●	●	●	●
M41 ... M45	●	●	●	●	●	●
M70	●	●	●	●	●	●
MASLDEF	●	●	●	●	●	●
MASLDEL	●	●	●	●	●	●
MASLOF	●	●	●	●	●	●
MASLOFS	●	●	●	●	●	●
MASLON	●	●	●	●	●	●
MATCH	●	●	●	●	●	●
MAXVAL	●	●	●	●	●	●

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
MCALL	●	●	●	●	●	●
MEAC	-	-	-	-	-	-
MEAFRAME	●	●	●	●	●	●
MEAS	●	●	●	●	●	●
MEASA	-	-	-	-	-	-
MEASURE	●	●	●	●	●	●
MEAW	●	●	●	●	●	●
MEAWA	-	-	-	-	-	-
MI	●	●	●	●	●	●
MINDEX	●	●	●	●	●	●
MINVAL	●	●	●	●	●	●
MIRROR	●	●	●	●	●	●
MMC	●	●	●	●	●	●
MOD	●	●	●	●	●	●
MODAXVAL	●	●	●	●	●	●
MOV	●	●	●	●	●	●
MSG	●	●	●	●	●	●
MVTOOL	●	●	●	●	●	●
N	●	●	●	●	●	●
NCK	●	●	●	●	●	●
NEWCONF	●	●	●	●	●	●
NEWT	●	●	●	●	●	●
NORM	●	●	●	●	●	●
NOT	●	●	●	●	●	●
NPROT	●	●	●	●	●	●
NPROTDEF	●	●	●	●	●	●
NUMBER	●	●	●	●	●	●
OEMIPO1	-	-	-	-	-	-
OEMIPO2	-	-	-	-	-	-
OF	●	●	●	●	●	●
OFFN	●	●	●	●	●	●
OMA1	-	-	-	-	-	-
OMA2	-	-	-	-	-	-
OMA3	-	-	-	-	-	-
OMA4	-	-	-	-	-	-
OMA5	-	-	-	-	-	-
OR	●	●	●	●	●	●
ORIAXES	-	-	-	-	-	-
ORIAXPOS	-	-	-	-	-	-

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
ORIC	-	-	-	-	-	-
ORICONCCW	-	-	-	-	-	-
ORICONCW	-	-	-	-	-	-
ORICONIO	-	-	-	-	-	-
ORICONTO	-	-	-	-	-	-
ORICURVE	-	-	-	-	-	-
ORID	-	-	-	-	-	-
ORIEULER	-	-	-	-	-	-
ORIMKS	-	-	-	-	-	-
ORIPATH	-	-	-	-	-	-
ORIPATHS	-	-	-	-	-	-
ORIPLANE	-	-	-	-	-	-
ORIRESET	-	-	-	-	-	-
ORIROTA	-	-	-	-	-	-
ORIROTC	-	-	-	-	-	-
ORIROTR	-	-	-	-	-	-
ORIROTT	-	-	-	-	-	-
ORIRPY	-	-	-	-	-	-
ORIRPY2	-	-	-	-	-	-
ORIS	-	-	-	-	-	-
ORISOF	-	-	-	-	-	-
ORISON	-	-	-	-	-	-
ORIVECT	-	-	-	-	-	-
ORIVIRT1	-	-	-	-	-	-
ORIVIRT2	-	-	-	-	-	-
ORIWKS	-	-	-	-	-	-
OS	-	-	-	-	-	-
OSB	-	-	-	-	-	-
OSC	-	-	-	-	-	-
OSCILL	-	-	-	-	-	-
OSCTRL	-	-	-	-	-	-
OSD	-	-	-	-	-	-
OSE	-	-	-	-	-	-
OSNSC	-	-	-	-	-	-
OSOF	-	-	-	-	-	-
OSP1	-	-	-	-	-	-
OSP2	-	-	-	-	-	-
OSS	-	-	-	-	-	-
OSSE	-	-	-	-	-	-

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
OST	-	-	-	-	-	-
OST1	-	-	-	-	-	-
OST2	-	-	-	-	-	-
OTOL	-	●	-	●	-	●
OVR	●	●	●	●	●	●
OVRA	●	●	●	●	●	●
OVRRAP	●	●	●	●	●	●
P	●	●	●	●	●	●
PAROT	●	●	●	●	●	●
PAROTOF	●	●	●	●	●	●
PCALL	●	●	●	●	●	●
PDELAYOF	-	-	-	-	-	-
PDELAYON	-	-	-	-	-	-
PHU	●	●	●	●	●	●
PL	- -	○ -	- -	○ -	- -	○ -
PM	●	●	●	●	●	●
PO	-	-	-	-	-	-
POCKET3	●	●	●	●	●	●
POCKET4	●	●	●	●	●	●
POLF	●	●	●	●	●	●
POLFA	●	●	●	●	●	●
POLFMASK	●	●	●	●	●	●
POLFMLIN	●	●	●	●	●	●
POLY	-	-	-	-	-	-
POLYPATH	-	-	-	-	-	-
PON	-	-	-	-	-	-
PONS	-	-	-	-	-	-
POS	●	●	●	●	●	●
POSA	●	●	●	●	●	●
POSM	●	●	●	●	●	●
POSP	●	●	●	●	●	●
POSRANGE	●	●	●	●	●	●
POT	●	●	●	●	●	●
PR	●	●	●	●	●	●
PREPRO	●	●	●	●	●	●
PRESETON	●	●	●	●	●	●
PRIO	●	●	●	●	●	●

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
PROC	●	●	●	●	●	●
PTP	●	●	●	●	●	●
PTPG0	●	●	●	●	●	●
PUNCHACC	-	-	-	-	-	-
PUTFTOC	●	●	●	●	●	●
PUTFTOCF	●	●	●	●	●	●
PW	-	○	-	○	-	○
QECLRNOF	●	●	●	●	●	●
QECLRNON	●	●	●	●	●	●
QU	●	●	●	●	●	●
R...	●	●	●	●	●	●
RAC	●	●	●	●	●	●
RDISABLE	●	●	●	●	●	●
READ	●	●	●	●	●	●
REAL	●	●	●	●	●	●
REDEF	●	●	●	●	●	●
RELEASE	●	●	●	●	●	●
REP	●	●	●	●	●	●
REPEAT	●	●	●	●	●	●
REPEATB	●	●	●	●	●	●
REPOSA	●	●	●	●	●	●
REPOSH	●	●	●	●	●	●
REPOSHA	●	●	●	●	●	●
REPOSL	●	●	●	●	●	●
REPOSQ	●	●	●	●	●	●
REPOSQA	●	●	●	●	●	●
RESET	●	●	●	●	●	●
RESETMON	●	●	●	●	●	●
RET	●	●	●	●	●	●
RIC	●	●	●	●	●	●
RINDEX	●	●	●	●	●	●
RMB	●	●	●	●	●	●
RME	●	●	●	●	●	●
RMI	●	●	●	●	●	●
RMN	●	●	●	●	●	●
RND	●	●	●	●	●	●
RNDM	●	●	●	●	●	●
ROT	●	●	●	●	●	●
ROTS	●	●	●	●	●	●

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
ROUND	●	●	●	●	●	●
ROUNDUP	●	●	●	●	●	●
RP	●	●	●	●	●	●
RPL	●	●	●	●	●	●
RT	●	●	●	●	●	●
RTLIOF	●	●	●	●	●	●
RTLION	●	●	●	●	●	●
S	●	●	●	●	●	●
SAVE	●	●	●	●	●	●
SBLOF	●	●	●	●	●	●
SBLON	●	●	●	●	●	●
SC	●	●	●	●	●	●
SCALE	●	●	●	●	●	●
SCC	●	●	●	●	●	●
SCPARA	●	●	●	●	●	●
SD	-	○	-	○	-	○
SEFORM	●	●	●	●	●	●
SET	●	●	●	●	●	●
SETAL	●	●	●	●	●	●
SETDNO	●	●	●	●	●	●
SETINT	●	●	●	●	●	●
SETM	-	-	-	-	-	-
SETMS	●	●	●	●	●	●
SETMS(n)	●	●	●	●	●	●
SETMTH	●	●	●	●	●	●
SETPIECE	●	●	●	●	●	●
SETTA	●	●	●	●	●	●
SETTCOR	●	●	●	●	●	●
SETTIA	●	●	●	●	●	●
SF	●	●	●	●	●	●
SIN	●	●	●	●	●	●
SIRELAY	-	-	-	-	-	-
SIRELIN	-	-	-	-	-	-
SIRELOUT	-	-	-	-	-	-
SIRELTIME	-	-	-	-	-	-
SLOT1	●	●	●	●	●	●
SLOT2	●	●	●	●	●	●
SOFT	●	●	●	●	●	●
SOFTA	●	●	●	●	●	●

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
SON	-	-	-	-	-	-
SONS	-	-	-	-	-	-
SPATH	●	●	●	●	●	●
SPCOF	●	●	●	●	●	●
SPCON	●	●	●	●	●	●
SPI	●	●	●	●	●	●
SPIF1	-	-	-	-	-	-
SPIF2	-	-	-	-	-	-
SPLINEPATH	-	○	-	○	-	○
SPN	-	-	-	-	-	-
SPOF	-	-	-	-	-	-
SPOS	●	●	●	●	●	●
SPOSA	●	●	●	●	●	●
SPP	-	-	-	-	-	-
SPRINT	●	●	●	●	●	●
SQRT	●	●	●	●	●	●
SR	-	-	-	-	-	-
SRA	-	-	-	-	-	-
ST	-	-	-	-	-	-
STA	-	-	-	-	-	-
START	-	-	-	-	-	-
STARTFIFO	●	●	●	●	●	●
STAT	●	●	●	●	●	●
STOLF	-	-	-	-	-	-
STOPFIFO	●	●	●	●	●	●
STOPRE	●	●	●	●	●	●
STOPREOF	●	●	●	●	●	●
STRING	●	●	●	●	●	●
STRINGFELD	●	●	●	●	●	●
STRINGIS	●	●	●	●	●	●
STRINGVAR	-	-	-	-	-	-
STRLEN	●	●	●	●	●	●
SUBSTR	●	●	●	●	●	●
SUPA	●	●	●	●	●	●
SVC	●	●	●	●	●	●
SYNFCT	●	●	●	●	●	●
SYNR	●	●	●	●	●	●
SYNRW	●	●	●	●	●	●
SYNW	●	●	●	●	●	●

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
T	●	●	●	●	●	●
TAN	●	●	●	●	●	●
TANG	-	-	-	-	-	-
TANGDEL	-	-	-	-	-	-
TANGOF	-	-	-	-	-	-
TANGON	-	-	-	-	-	-
TCA (828D: _TCA)	●	●	●	●	●	●
TCARR	-	●	-	●	-	●
TCI	●	●	●	●	●	●
TCOABS	-	●	-	●	-	●
TCOFR	-	●	-	●	-	●
TCOFRX	-	●	-	●	-	●
TCOFRY	-	●	-	●	-	●
TCOFRZ	-	●	-	●	-	●
THETA	-	-	-	-	-	-
TILT	-	-	-	-	-	-
TLIFT	-	-	-	-	-	-
TMOF	●	●	●	●	●	●
TMON	●	●	●	●	●	●
TO	●	●	●	●	●	●
TOFF	●	●	●	●	●	●
TOFFL	●	●	●	●	●	●
TOFFOF	●	●	●	●	●	●
TOFFON	●	●	●	●	●	●
TOFFR	●	●	●	●	●	●
TOFRAME	●	●	●	●	●	●
TOFRAMEX	●	●	●	●	●	●
TOFRAMEY	●	●	●	●	●	●
TOFRAMEZ	●	●	●	●	●	●
TOLOWER	●	●	●	●	●	●
TOOLENV	●	●	●	●	●	●
TOROT	●	●	●	●	●	●
TOROTOF	●	●	●	●	●	●
TOROTX	●	●	●	●	●	●
TOROTY	●	●	●	●	●	●
TOROTZ	●	●	●	●	●	●
TOUPPER	●	●	●	●	●	●
TOWBCS	-	●	-	●	-	●

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
TOWKCS	-	●	-	●	-	●
TOWMCS	-	●	-	●	-	●
TOWSTD	-	●	-	●	-	●
TOWTCS	-	●	-	●	-	●
TOWWCS	-	●	-	●	-	●
TR	●	●	●	●	●	●
TRAANG	-	-	-	-	○	-
TRACON	-	-	-	-	○	-
TRACYL	○	○	○	○	○	○
TRAFOOF	●	●	●	●	●	●
TRAILOF	●	●	●	●	●	●
TRAILON	●	●	●	●	●	●
TRANS	●	●	●	●	●	●
TRANSMIT	○	○	○	○	○	○
TRAORI	-	●	-	●	-	●
TRUE	●	●	●	●	●	●
TRUNC	●	●	●	●	●	●
TU	●	●	●	●	●	●
TURN	●	●	●	●	●	●
ULI	●	●	●	●	●	●
UNLOCK	●	●	●	●	●	●
UNTIL	●	●	●	●	●	●
UPATH	●	●	●	●	●	●
VAR	●	●	●	●	●	●
VELOLIM	●	●	●	●	●	●
VELOLIMA	●	●	●	●	●	●
WAITC	-	-	-	-	○	-
WAITE	-	-	-	-	-	-
WAITENC	-	-	-	-	-	-
WAITM	-	-	-	-	-	-
WAITMC	-	-	-	-	-	-
WAITP	●	●	●	●	●	●
WAITS	●	●	●	●	●	●
WALCS0	●	●	●	●	●	●
WALCS1	●	●	●	●	●	●
WALCS2	●	●	●	●	●	●
WALCS3	●	●	●	●	●	●
WALCS4	●	●	●	●	●	●
WALCS5	●	●	●	●	●	●

작업	828D 시스템 버전					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	선삭	밀링	선삭	밀링
WALCS6	●	●	●	●	●	●
WALCS7	●	●	●	●	●	●
WALCS8	●	●	●	●	●	●
WALCS9	●	●	●	●	●	●
WALCS10	●	●	●	●	●	●
WALIMOF	●	●	●	●	●	●
WALIMON	●	●	●	●	●	●
WHEN	●	●	●	●	●	●
WHENEVER	●	●	●	●	●	●
WHILE	●	●	●	●	●	●
WRITE	●	●	●	●	●	●
WRTPR	●	●	●	●	●	●
X	●	●	●	●	●	●
XOR	●	●	●	●	●	●
Y	●	●	●	●	●	●
Z	●	●	●	●	●	●

● 표준

○ 옵션

- 사용할 수 없음

17.3 현재 HMI 에서 설정된 언어

아래 테이블은 사용자 인터페이스에서 사용 가능한 언어를 모두 열거한 것입니다 .

현재 설정된 언어는 다음 시스템 변수를 사용하여 가공 프로그램 및 동기 동작에서 쿼리할 수 있습니다 .

$AN_LANGUAGE_ON_HMI = < 값 >

< 값 >	언어	언어 코드
1	독일어 (독일)	GER
2	프랑스어	FRA
3	영어 (영국)	ENG
4	스페인어	ESP
6	이탈리아어	ITA
7	네델란드어	NLD
8	중국어 간체	CHS
9	스웨덴어	SVE
18	헝가리어	HUN
19	핀란드어	FIN
28	체코어	CSY
50	포르투갈어 (브라질)	PTB
53	폴란드어	PLK
55	덴마크어	DAN
57	러시아어	RUS
68	슬로바키아어	SKY
72	루마니아어	ROM
80	중국어 번체	CHT
85	한국어	KOR
87	일본어	JPN
89	터키어	TRK

주

다음 경우 $AN_LANGUAGE_ON_HMI 가 업데이트됩니다 .

- 시스템 부팅 후
- NCK 및 / 또는 PLC 가 리셋된 후
- M2N 범위 내의 다른 NCK 로 전환한 후
- HMI 에서 언어를 변경한 후

부록 A

A.1 약어 목록

A	출력
자동화 시스템	자동화 시스템
ASCII	American Standard Code for Information Interchange(미국 정보교환 표준 코드)
ASIC	특정 통합 회로 적용 : 사용자 전환 회로
ASUB	비동기 서브루틴
AV	작업 계획
STL	성명 목록
BA	운전 모드
모드 그룹	모드 그룹
BB	실행 준비
HMI	인간 기계 인터페이스
BCD	2 진 코드 정수 : 2 진 코드로 인코드된 정수
HHU	핸드 헬드 유닛
BIN	이진 파일 (이진파일)
BIOS	기본적 입력 출력 시스템
BCS	기본 좌표계
UI	사용자 인터페이스
OP	화면 조작반
OPI	화면 조작반 인터페이스 :
CAD	컴퓨터 지원 설계
CAM	컴퓨터 지원 제작
CNC	컴퓨터 수치 제어 장치 : 컴퓨터 수치 제어 장치
COM	통신
CP	통신 프로세서
CPU	중앙 처리 장치 : 중앙 처리 장치
CR	Carriage Return(캐리지 리턴)
흑백 브라운관	흑백 브라운관 수상관
CSB	Central Service Board(중앙 서비스 보드): PLC 모듈
CTS	Clear To Send(송신 허가): 직렬 데이터 인터페이스의 신호
CUTCOM	커터 반경 보정 공구 반경 보정
DAC	Digital-to-Analog Converter (디지털 - 아날로그 변환기)
DB	PLC 내 데이터 블록
DBB	PLC 내 데이터 블록 바이트
DBW	PLC 내 데이터 블록 워드
DBX	PLC 내 데이터 블록 비트
DC	직접 제어 : 한 회전 이내 절대 위치로의 가장 짧은 경로를 통한 로터리 축 이동
DCD	데이터 케리어 탐지

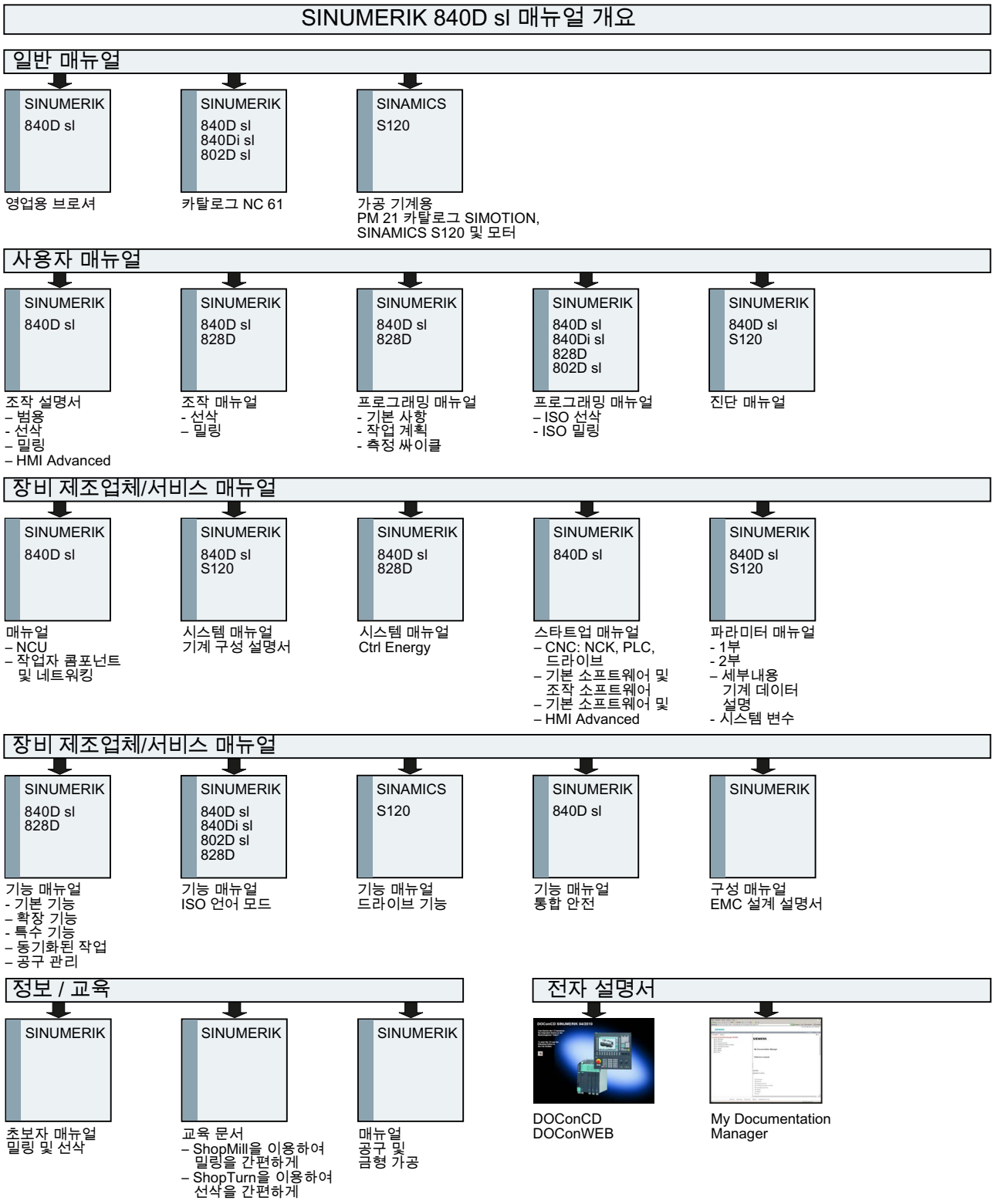
DDE	역학적 데이터 교환
DTE	데이터 터미널 장치
DIN	독일 공업 표준 (Deutsche Industrie Norm)
DIO	데이터 입력 / 출력 : 데이터 전송 디스플레이
DIR	디렉토리 : 디렉토리
DLL	역학적 링크 라이브러리
DOE	데이터 전송 장치
DOS	디스크 운영체제
DPM	이원 포트 메모리
DPR	이원 포트 램
DRAM	Dynamic Random Access Memory(동적 랜덤 액세스 메모리)
DRF	핸드휠 옵셋 기능 : 핸드휠 옵셋 기능 (DRF)
DRY	드라이런 : 드라이런 이송 속도
DSB	싱글 블록 디코딩 : 싱글 블록 디코딩
DW	데이터 워드
E	입력
I/O	입력 / 출력
EIA 코드	특수 펀칭 테이프 코드 , 문자당 구멍 수는 항상 홀수
ENC	엔코더 : 실제 값 엔코더
EPROM	Erasable Programmable Read Only Memory(소거 및 프로그램 가능 읽기 전용 메모리)
에러	인쇄기의 에러
FB	함수 블록
FBS	슬림라인 화면
FC	함수 호출 : PLC 내 함수 블록
FDB	제품 데이터베이스
FDD	플로피 디스크 드라이브
FEPROM	플래쉬 -EPROM: 읽기와 쓰기 메모리
FIFO	First In First Out(선입선출): 동일한 순서로 데이터를 읽고 이러한 데이터가 저장된 어드레스 지정 없이 작동하는 메모리입니다 .
선입선출	미세 보간자
FM	함수 모듈
FPU	유동 지점 유닛 유동 지점 유닛
FRA	프레임 블록
FRAME	데이터 기록 (프레임)
CRC	커터 반경 보정
FST	이송 정지 : 이송 정지
CSF	함수 계획 (PLC 프로그래밍 방법)
BP	기본 프로그램
GUD	전역 사용자 데이터 : 전역 사용자 데이터
HD	하드 디스크 하드 디스크
16 진수	16 진수의 축약

AuxF	보조 기능
HMI	인간 기계 인터페이스: 작업, 프로그래밍 및 시뮬레이션을 위한 SINUMERIK 의 조작자 기능성 .
HMS	고해상도 측정 시스템
MSD	메인 스핀들 드라이브
HW	하드웨어
IBN	스타트업
IF	드라이브 모듈 펄스 가능해짐
IK (GD)	함축적 통신 (전역 데이터)
IKA	보간 보상 : 보간 보상
IM	인터페이스 모듈 상호 연결 모듈
IMR	인터페이스 모듈 수신 : 수신 데이터의 상호 연결 모듈
IMS	인터페이스 모듈 송신 : 송신 데이터의 상호 연결 모듈
INC	증분 : 증분
INI	초기화 데이터 : 초기화 데이터
IPO	보간자
ISA	산업 표준 아키텍처
ISO	국제 표준화 기관
ISO 코드	특수 펀칭 테이프 코드 , 문자당 구멍 수는 항상 짝수
조그 모드	조깅 : 설정 모드
K1 .. K4	채널 1 - 채널 4
C Bus	통신 버스
COR	좌표 회전
LAD	래더 다이어그램 (PLC 프로그래밍 방법)
K_V	서보 게인 팩터
K_{UE}	속도비
LCD	액정화면 : 액정화면
LED	발광 다이오드 : 발광 다이오드
LF	라인 피드
PMS	위치 측정 시스템
LR	위치 제어기
LUD	로컬 사용자 데이터
MB	메가바이트
MD	머신 데이터
MDI	수동 데이터 입력 : 수동 입력
MC	측정 회로
MCS	기계 좌표계
MLFB	기계 판독 가능 제품 명칭
MPF	메인 프로그램 파일 : NC 가공 프로그램 (메인 프로그램)
MPI	다중 포트 인터페이스 다중 포트 인터페이스
MS	마이크로소프트 (소프트웨어 제작업체)
MCP	기계 조작반

NC	수치 제어 : 수치 제어
NCK	수치 제어 커넬 : 블록 준비 , 이송 범위 등의 NC 커넬
NCU	수치 제어 유닛 : NCK 의 하드웨어 유닛
NRK	NCK 운영 시스템 이름
IS	인터페이스 신호
NURBS	Non-Uniform Rational B-Spline
ZO	워크 옵셋
OB	PLC 내 조직 블록
OEM	주문자 상표 부착 방식
OP	Operator Panel: 운영 설정
OPI	화면 조작반 인터페이스 : 화면 조작반 연결용 인터페이스
OPT	옵션 : 옵션
OSI	개방 시스템 상호 연결 : 컴퓨터 통신 표준
P bus	주변기기 버스
PC	개인용 컴퓨터
PCIN	제어기를 사용한 데이터 교환 SW 이름
PCMCIA	개인 컴퓨터 메모리 카드 국제 협회 : 프러그 인 메모리 카드 표준
PCU	PC 유닛 : PC 박스 (컴퓨터 유닛)
PG	프로그래밍 디바이스
PLC	프로그램 가능 논리 제어 : 인터페이스 제어
POS	포지셔닝
RAM	임의 액세스 기억 장치 : 읽고 쓸 수 있는 프로그램 메모리
REF	원점 복귀 기능
REPOS	위치 변경 기능
RISC	Reduced Instruction Set Computer: 간단한 명령 집합 및 고속에서 명령을 처리할 수 있는 기능이 있는 프로세서 유형
ROV	급 이송 오버라이드 : 입력 수정
RPA	R 파라미터 활성 : R 파라미터 값에 대한 NCK 의 메모리 영역
RPY	좌우 진동 , 앞뒤 진동 , 수평진동 좌표계의 회전 유형
RTS	송신 요청 : RTS, 직렬 데이터 인터페이스의 제어 신호
SBL	싱글 블록 : 싱글 블록
SD	셋팅 데이터
SDB	시스템 데이터 블록
SEA	셋팅 데이터 활성 : 셋팅 데이터의 식별자 (파일 종류)
SFB	시스템 함수 블록
SFC	시스템 함수 호출
SK	소프트 키
SKP	SKiP: 블록 스킵
SM	Stepper 모터
SPF	서브루틴 파일 : 서브프로그램
PLC	프로그래밍 가능한 논리 컨트롤러

SRAM	정적 램 (비휘발성)
TNRC	공구 노즈 반경 보정
LEC	리드스크류 에러 보정
SSI	직렬 동기 인터페이스 : 직렬 동기 인터페이스
SW	소프트웨어
SYF	시스템 파일 시스템 파일
TEA	테스팅 데이터 활성 : 머신 데이터 식별자
TO	공구 옵셋 : 공구 옵셋
TOA	공구 옵셋 활성 : 공구 옵셋의 식별자 (파일 종류)
TRANSMIT	밀링 가공에서 선삭 가공으로 변환 : 밀링 작업을 위해 선삭 기계에서 좌표 변환
UFR	사용자 프레임 : 워크 옵셋
SR	서브프로그램
FDD	이송 드라이브
V.24	직렬 인터페이스 (DTE 와 DCE 사이의 교환 라인 정의)
워크	공작물 좌표계
T	공구
TLC	공구 길이 보정
WOP	작업장 지향 프로그래밍
WPD	Workpiece Directory: 공작물 디렉토리
TRC	공구 반경 보정
TO	공구 옵셋
TC	공구 교환
ZOA	제로 옵셋 활성 : 워크 옵셋 데이터의 식별자 (파일 종류)
µC	마이크로 제어기

A.2 개요


SINUMERIK 840D sl 매뉴얼 개요
일반 매뉴얼
SINUMERIK 840D sl
영업용 브로셔
SINUMERIK 840D sl 840Di sl 802D sl
카탈로그 NC 61
SINAMICS S120
가공 기계용
PM 21 카탈로그 SIMOTION, SINAMICS S120 및 모터
사용자 매뉴얼
SINUMERIK 840D sl
조작 설명서
– 범용
- 선삭
– 밀링
– HMI Advanced
SINUMERIK 840D sl 828D
조작 매뉴얼
- 선삭
– 밀링
SINUMERIK 840D sl 828D
프로그래밍 매뉴얼
- 기본 사항
- 작업 계획
- 측정 싸이클
SINUMERIK 840D sl 840Di sl 828D 802D sl
프로그래밍 매뉴얼
– ISO 선삭
- ISO 밀링
SINUMERIK 840D sl S120
진단 매뉴얼
장비 제조업체/서비스 매뉴얼
SINUMERIK 840D sl
매뉴얼
– NCU
– 작업자 콤포넌트 및 네트워킹
SINUMERIK 840D sl S120
시스템 매뉴얼
기계 구성 설명서
SINUMERIK 840D sl 828D
시스템 매뉴얼
Ctrl Energy
SINUMERIK 840D sl
스타트업 매뉴얼
– CNC: NCK, PLC, 드라이브
– 기본 소프트웨어 및 조작 소프트웨어
– 기본 소프트웨어 및
– HMI Advanced
SINUMERIK 840D sl S120
파라미터 매뉴얼
- 1부
- 2부
– 세부내용 기계 데이터 설명
- 시스템 변수
장비 제조업체/서비스 매뉴얼
SINUMERIK 840D sl 828D
기능 매뉴얼
- 기본 기능
– 확장 기능
- 특수 기능
– 동기화된 작업
– 공구 관리
SINUMERIK 840D sl 840Di sl 802D sl 828D
기능 매뉴얼
ISO 언어 모드
SINAMICS S120
기능 매뉴얼
드라이브 기능
SINUMERIK 840D sl
기능 매뉴얼
통합 안전
SINUMERIK
구성 매뉴얼
EMC 설계 설명서
정보 / 교육
SINUMERIK
초보자 매뉴얼
밀링 및 선삭
SINUMERIK
교육 문서
– ShopMill을 이용하여 밀링을 간편하게
– ShopTurn을 이용하여 선삭을 간편하게
SINUMERIK
매뉴얼
공구 및
금형 가공
전자 설명서
DOConCD
DOConWEB
My Documentation Manager

SINUMERIK 828D 매뉴얼 개요

일반 매뉴얼

영업용 브로서

SINUMERIK

EMC 지침

사용자 매뉴얼

SINUMERIK 828D 840D sl

조작 매뉴얼
- 선삭
- 밀링

SINUMERIK 828D 840D sl

프로그래밍 매뉴얼
- 기본 정보
- 생산 계획
- 이지 스크린

SINUMERIK 802D sl 828D 840D sl

프로그래밍 매뉴얼
- ISO 선삭
- ISO 밀링

SINUMERIK 828D SINAMICS S120

진단 매뉴얼

장비 제조업체/서비스 매뉴얼

SINUMERIK 828D

매뉴얼
시운전 매뉴얼
서비스 매뉴얼

SINUMERIK 828D 840D sl

기능 매뉴얼
- 기본 기능
- 확장 기능

SINUMERIK 802D sl 828D 840D sl

기능 매뉴얼
ISO 모드

SINUMERIK 828D SINAMICS S120

리스트 매뉴얼
- 머신 데이터 및 인터페이스 신호
- 상세 파라미터 설명

전자 설명서

DOConCD
DOConWEB

Industry Mall

용어 색인

C 스플라인

C 스플라인은 가장 잘 알려져 있고 널리 사용되는 스플라인입니다. 보간 지점의 전이가 접선과 만곡 지점에서 지속적으로 이뤄집니다 . 3 차 다항식이 사용됩니다 .

CNC

NC 를 참조하십시오 .

COM

통신 구현 및 좌표 지정을 위한 NC 제어 시스템의 구성 요소입니다 .

CPU

중앙 처리 장치 , 참조 → PLC

C 축

공구 스핀들이 제어된 회전 및 위치 지정 이동을 설명하는 축입니다 .

DRF

핸들 인트럽트 기능 (FANUC 기준): 핸드휠을 사용하여 자동 모드에서 증분 워크 옵셋을 생성하는 NC 기능입니다 .

HIGHSTEP

AS300/AS400 시스템의 PLC 용 프로그래밍 옵션 요약

KV

서보 게인 팩터이며 제어 루프 내의 제어 변수입니다 .

MDA

시스템 동작 모드 : 수동 데이터 입력 (MDI). MDI 모드에서 , 현재의 메인 프로그램 또는 서브루틴과 아무런 관련이 없는 개별 프로그램 블록 또는 블록 시퀀스를 입력하고 NC 시작 키를 눌러 실행시킬 수 있습니다 .

NC

수치 제어 : NC 에는 다음과 같은 기계 공구 제어에 대한 모든 구성 요소가 포함되어 있습니다 : → NCK → PLC, HMI → COM.

주

SINUMERIK 840D 제어 시스템에서 보다 정확한 용어 표현은 다음과 같습니다 . 컴퓨터 수치 제어

NCK

수치 제어 커널 : 가공 프로그램을 실행하고 기본적으로 공작 기계의 이동 작업을 좌표로 표시하는 구성 요소입니다 .

NRK

수치 로봇 커널 (NCK 운영체제)

NURBS

제어 시스템에서 이뤄지는 모션 제어 및 경로 보간은 NURBS (**N**on **U**niform **R**ational **B**-**S**plines) 을 토대로 수행됩니다 . 결과적으로 , 제어 시스템에서 통일된 프로세스가 SINUMERIK 840D 의 모든 보간에 맞게 지원됩니다 .

OEM

사용자 인터페이스를 생성하거나 프로세스 지향 기능을 제어 시스템에 통합하려는 장비 제조업체를 위해 SINUMERIK 840D 의 개별 솔루션 (OEM 애플리케이션) 을 구현하는 범위가 제공됩니다 .

PCIN 데이터 전송 프로그램

PCIN 은 직렬 인터페이스 (시리얼 통신) 를 통해 CNC 사용자 데이터(예: 가공 프로그램, 공구 옵셋 등) 을 송수신하는 데 사용되는 보조 프로그램입니다 . PCIN 프로그램은 표준 산업용 PC 의 MS-DOS 에서 실행할 수 있습니다 .

PLC

Programmable **L**ogic **C**ontrol: → 프로그램 가능 논리 제어기. NC 구성 요소: 공작 기계의 제어 논리를 처리하기 위한 프로그램 가능 제어 시스템입니다 .

PLC 프로그래밍

STEP 7 소프트웨어를 사용하여 PLC를 프로그래밍합니다. STEP 7 프로그래밍 소프트웨어는 **WINDOWS** 표준 운영체제를 기반으로 하며 획기적으로 개선된 STEP 5 프로그래밍 기능이 포함되어 있습니다 .

PLC 프로그램 메모리

SINUMERIK 840D: PLC 사용자 프로그램 , 사용자 데이터 및 기본 PLC 프로그램은 모두 PLC 사용자 메모리에 저장됩니다 .

R 파라미터

프로그램에서 원하는 목적에 따라 가공 프로그램의 프로그래머에 의해 이러한 수치 파라미터를 설정하거나 질의할 수 있으며 필요에 따라 변경할 수 있습니다 .

SRT

변환비

TOA 영역

TOA 영역에는 모든 공구 및 매거진 데이터가 저장됩니다 . 기본적으로 이 영역은 데이터를 수신하는 채널 영역과 동기화됩니다. 그러나 머신 데이터를 이용하여 일반 공구 관리 데이터를 여러 채널이 사용할 수 있도록 그 채널들에 하나의 TOA 장치를 공유하도록 지정할 수 있습니다 .

TOA 유닛

각 TOA 영역에는 한 개 이상의 TOA 유닛이 있습니다 . 존재할 수 있는 TOA 유닛 개수는 활성 채널의 최대 개수로 제한됩니다 . TOA 유닛에는 정확히 한 개의 공구 데이터 블록과 매거진 데이터 블록이 존재합니다. 또한 TOA 유닛에 공구 고정 장치 데이터 블록(옵션)이 존재할 수도 있습니다 .

WinSCP

WinSCP 는 파일 전송에 무료로 사용할 수 있는 Windows 용 개방 소스 프로그램입니다 .

가공 채널

채널 구조는 병렬 이송 순서를 통해 유휴 시간을 단축 (예 : 가공과 동시에 로딩 갠트리 이동)하는 데 사용할 수 있습니다. 단, CNC 채널을 반드시 디코딩, 블록 준비 및 보간과 더불어 별도의 CNC 제어 시스템으로 간주해야 합니다 .

가공 프로그램 관리

가공 프로그램 관리는 공작물을 기준으로 편성됩니다. 사용자 메모리의 크기에 따라 관리할 수 있는 프로그램 개수와 데이터 용량이 다릅니다 . 각 파일 (프로그램 및 데이터) 의 이름은 최대 24 개의 영숫자로 지정할 수 있습니다 .

가공 프로그램 블록

줄 바꿈으로 구분된 가공 프로그램의 일부입니다 . 다음과 같이 두 가지 유형이 있습니다 : → 기본 블록 및 하위 블록 .

경로 속도

최대 프로그램 가능 경로 속도는 입력 해상도에 따라 다릅니다 . 예를 들어 0.1mm 의 해상도에서는 최대 프로그램 가능 경로 속도가 1000m/min 입니다 .

경로 이송 속도

경로 이송은 경로 축에 영향을 줍니다 . 관련 기하 축의 이송 속도에 대한 기하학적 합계를 표시합니다 .

경로 축

경로 축에는 끝점을 동시에 시작 , 가속 , 정지 및 도달하는 방법으로 보간자에 의해 제어되는 채널의 모든 가공 축이 포함되어 있습니다 .

경사면 가공

" 경사면 가공 " 기능을 이용하면 기계의 좌표계에 놓지 않고서도 손쉽게 공작물 표면에 드릴링 및 밀링 작업을 실시할 수 있습니다 .

고속 디지털 입 / 출력

디지털 입력은 예를 들어 고속 CNC 프로그램 루틴 (인터럽트 루틴) 를 시작하는 데 사용할 수 있습니다 . 디지털 CNC 출력을 사용하여 고속 프로그램 제어 스위치 기능을 실행할 수 있습니다 (SINUMERIK 840D).

고정 정지점 접근

기계 공구는 공구 변경 지점 , 로딩 지점 , 팰릿 변경 지점 등과 같이 정의된 방법으로 고정 정지점에 접근할 수 있습니다 . 이러한 지점의 상태는 제어 시스템에 저장됩니다 . 제어 시스템은 가능할 때마다 급 이송 시 관련 축을 이송합니다 .

곡률

형상의 곡률 k 는 형상 지점에서 네스팅 원호의 반경 r 에 반비례합니다 (k = 1/r).

공구

가공을 실행하는 공작 기계의 활성 부분입니다 (예: 선삭 공구, 밀링 공구, 드릴, 레이저 빔 등).

공구 반경 보정

원하는 공작물 형상을 직접 프로그래밍하려면 제어 시스템이 사용할 공구의 반경을 고려하여 프로그래밍된 형상과 동일한 거리의 경로를 이송해야 합니다 (G41/G42).

공구 옵셋

경로 계산 시 공구 치수를 고려합니다 .

공구 인선 반경 보정

형상 프로그래밍은 공구가 지정된 것으로 간주합니다 . 이러한 경우가 실제로 발생하지 않기 때문에 사용된 공구의 만곡 반경을 고려할 제어 시스템으로 전송해야 합니다 . 만곡 중심은 만곡 반경에 의해 옵셋된 형상 주위의 거리를 일정하게 유지합니다 .

공작물

공작 기계에 의해 가공될 부품입니다 .

공작물 원점

공작물 원점은 공작물 좌표계에 대한 시작점입니다 . 기계 원점까지의 거리로 정의됩니다 .

공작물 좌표계

공작물 원점에 공작물 좌표계의 시작점이 있습니다 . 공작물 좌표계에서 프로그래밍된 가공 작업 시 이 좌표계가 적용된 치수 및 방향을 사용합니다 .

공작물 형상

만들거나 가공할 공작물의 형상을 설정합니다 .

극 좌표

원점에서의 거리 및 정의된 축의 반경 벡터에 의해 설정된 각도 측면에서 평면의 지점 위치를 정의하는 좌표계입니다 .

급이송

축의 최대 이송 속도입니다 . 예를 들어 공구가 휴지 위치에서 공작물 형상에 접근하거나 공구가 공작물 형상에서 후진하는 경우 급 이송이 사용됩니다 . 급 이송 속도는 가공 데이터 요소를 사용하여 기계별로 설정됩니다 .

기계 고정점

예를 들어 기계 원점과 같이 기계 공구에 의해 고유하게 정의된 지점입니다 .

기계 원점

전체 파생된 측정 시스템을 추적할 수 있는 공작 기계의 고정점입니다 .

기계 조작반

키 , 로타리 스위치 및 LED 와 같은 간단한 표시등과 같은 작동 요소를 가진 공작 기계의 조작반입니다 . PLC 를 통해 공작 기계에 직접 적용하는 데 사용합니다 .

기계 좌표계

공작 기계의 축과 관련된 좌표계입니다 .

기계 축

공작 기계 상에 물리적으로 존재하는 축입니다 .

기본 블록

블록의 선두에 ":" 가 지정된 블록은 가공 프로그램 실행시키기 위해 필요한 모든 파라미터를 포함합니다 .

기본 좌표계

기계 좌표계로 변환에 의해 매핑된 직교 좌표계입니다 .

프로그래머는 가공 프로그램에서 기본 좌표계의 축 이름을 사용합니다. 기본 좌표계는 변환이 수행되지 않을 경우 기계 좌표계와 평행하게 존재합니다 . 두 가지 좌표계의 차이는 축 식별자 (이름) 에 있습니다 .

기본 축

보정 값 계산의 기준을 구성하는 지령치 또는 실제 값 위치의 축입니다 .

기하 축

기하 축은 공작물 좌표계에서 2 차원 또는 3 차원 영역을 설명하는 데 사용됩니다 .

나선형 보간

나선형 보간 기능은 밀링 커터를 사용하여 내부 및 외부 나사를 가공하거나 윤활 홈을 밀링하는 데 적합합니다 .

나선형 보간은 다음 두 가지 이동으로 구성됩니다 :

- 한 평면에서 원호 방향 이동
- 이 평면과 수직 방향으로 선형 이동

네트워크

네트워크는 연결 케이블을 통해 여러 개의 S7-300 스테이션 및 다른 장치 (예 : 프로그래밍 장치) 로 구성된 연결망입니다 . 네트워크에서는 연결된 장치간의 데이터 교환이 이뤄집니다 .

데이터 블록

1. HIGHSTEP 프로그램이 액세스할 수 있는 PLC 데이터 단위입니다 .
2. 데이터 단위 → NC: 데이터 모듈은 전역 사용자 데이터에 대한 데이터 정의를 포함합니다 . 이들 데이터는 정의 되었을 때 곧바로 초기화 됩니다 .

데이터 워드

데이터 블록 내의 2 바이트 데이터 단위입니다 .

동기화

다른 채널에서 특정 가공 지점에서 시퀀스 좌표화를 위한 가공 프로그램의 명령문입니다 .

동기화 동작

1. 보조 기능 출력

   공작물 가공 도중 기술적 기능 (보조 기능) 을 CNC 프로그램에서 PLC 로 출력할 수 있습니다. 예를 들어, 보조 기능을 사용하여 퀼, 그리퍼, 척 클램핑과 같은 공작 기계용 보조 장비를 제어할 수 있습니다 .

2. 보조 기능 고속 출력

   시간이 중시되는 전환 기능의 경우 가공 과정에서 보조 기능 인식 시간을 최소화하고 불필요한 대기 시간을 없앨 수 있습니다 .

동기화 축

동기화 축은 리딩 축 이송에 따라 설정 위치가 계속 변경되고 리딩 축과 동시에 이동하는 갠트리 축입니다 . 조작자와 프로그래머의 관점에서는 동기 축은 존재하지 않는 축입니다 .

동기화 축

동기화 축이 경로를 이동하는 데 걸리는 시간은 기하 축의 경로 이동 시간과 동일합니다 .

동적 피드 포워드 제어

특히 추종 에러로 인한 형상의 부정확성은 동적 가속 종속 피드 포워드 제어를 사용하여 제거할 수 있습니다 . 이렇게 하면 고속의 경로 속도에서도 뛰어난 가공 정확도를 얻을 수 있습니다 . 피드 포워드 제어는 가공 프로그램을 통해 축 관련 기준에서 선택하거나 선택 취소할 수 있습니다 .

드라이브

드라이브는 NC 설정을 기준으로 속도 및 토크 제어를 수행하는 CNC 의 장치입니다 .

라운딩 축

라운딩 축은 그리드 인덱스에 상응하는 각도 위치로 공작물이나 공구를 회전시킵니다 . 공작물이나 공구가 그리드 인덱스에 도달하면 라운딩 축이 정지합니다 .

로드 메모리

로드 메모리는 PLC 의 CPU 314 용 RAM 과 동일합니다 .

로터리 축

로터리 축은 공작물 또는 공구 회전을 정의된 각도 위치에 적용합니다 .

룩 - 어헤드

룩 어헤드 기능은 할당 가능한 이송 블록 수를 미리 확인하여 최적의 가공 속도를 구현하는 데 사용합니다 .

리딩 축

리딩 축은 조작자와 프로그래머의 관점으로 존재하는 갠트리 축입니다 . 따라서 표준 NC 축과 같은 영향을 받을 수 있습니다 .

매크로 기법

단일 식별자에 대한 명령문 집합을 그룹화합니다 . 식별자는 프로그램에서 통합된 명령문 집합을 표시합니다 .

메시지

가공 프로그램에 프로그래밍된 전체 메시지 및 시스템에 의해 감지된 알람은 화면 조작반에 날짜, 시간 및 취소 기준에 대한 해당 기호와 함께 일반 텍스트로 표시됩니다. 알람 및 메시지가 개별적으로 표시됩니다 .

메인 프로그램

" 메인 프로그램 " 은 가공 프로그램을 메인 프로그램과 서브프로그램으로 나누던 시절에 만들어진 용어입니다 . 오늘날의 SINUMERIK NC 언어에서는 이처럼 엄격하게 두 프로그램을 구분하지 않습니다 . 원칙적으로 채널 내의 모든 가공 프로그램을 선택하고 시작할 수 있습니다. 그러면 선택한 프로그램은 프로그램 레벨 0 (메인 프로그램 레벨) 에서 실행됩니다. 추가 가공 프로그램 또는 사이클을 서브프로그램으로써 메인 프로그램에서 호출할 수 있습니다 .

모드 그룹

기술적으로 연관된 축과 스핀들을 한 모드 그룹으로 통합할 수 있습니다. BAG (모드그룹) 상의 축/스핀들은 하나 이상의 채널에 의해 제어될 수 있습니다. 모드 그룹 내의 채널에는 항상 동일 모드 형태가 할당됩니다 .

미러링

미러링은 축 관련 형상의 좌표값 부호를 보관합니다 . 둘 이상의 축을 한 번에 미러링할 수 있습니다 .

미터 측정 시스템

표준화된 단위 체계 : 예를 들어 길이의 경우 mm(밀리미터), m(미터) 입니다 .

반한시 이송 속도

SINUMERIK 840D 를 사용하면 이송 속도 대신 축 이송을 기준으로 블록 경로 이송에 필요한 시간을 프로그래밍할 수 있습니다 (G93).

배율 조정

축 관련 배율 수정을 수행하는 프레임의 구성 요소입니다 .

백래시 보정

볼 스크류의 반전 시 발생되는 기계적 백래시를 보정합니다 . 각 축에 대해 백래시 보정을 개별적으로 입력할 수 있습니다 .

백업 배터리

백업 배터리는 정전 사고가 발생한 경우 지정된 데이터 영역과 비트 메모리 , 타이머 및 카운터가 안전하게 저장될 수 있도록 CPU 에 로드된 사용자 프로그램을 저장합니다 .

변수 정의

변수 정의에는 데이터 유형 정보 및 변수 이름이 포함됩니다 . 변수 이름을 사용하여 변수 값에 액세스할 수 있습니다 .

변환

축의 추가 또는 절대 워크 옵셋입니다 .

보간 보상

보간 보정은 리드스크류 에러 및 측정 시스템 에러 (SSFK, MSFK) 를 보정하기위한 도구입니다 .

보간자

가공 프로그램에 지정된 끝 위치에 대한 정보를 기준으로 개별 축에서 실행할 작동에 대한 중간 값을 정의하는 NCK 의 논리 단위입니다 .

보정 값

엔코더 및 원하는 프로그램 축 위치에 의해 측정된 축 위치 사이 편차입니다 .

보정 메모리

시스템 내에서 공구 옵셋 데이터가 저장되는 데이터 범위입니다 .

보정 척을 사용하지 않고 태핑

이 기능을 사용하여 보정 척 없이 나사를 태핑할 수 있습니다 . 스핀들을 로터리 축 및 드릴링 축으로 하는 보간 방법을 사용하여 예를 들어 블라인드 홀 나사와 같이 정밀한 최종 드릴링 깊이로 나사를 절삭할 수 있습니다 . (요구 사항 : 스핀들이 축 모드여야 합니다 .)

보정 축

보정 값에 의해 수정된 지령치 또는 실제 값을 가진 축입니다 .

보정 테이블

보간 지점이 포함된 테이블입니다 . 기준 축에서 선택한 지점에 대한 보정 축의 보정 값을 제공합니다 .

보조 기능

보조 기능은 장비 제조회사에서 정의한 동작을 실행시키기 위해 PLC 로 파라미터를 전송하기 위한 가공 프로그램을 인에이블 합니다 .

보호 영역

공구 팁이 전달되지 않아야 하는 작업 영역 내의 3 차원 영역입니다 .

부팅

전원 ON 후 시스템 프로그램 로딩 .

블록

" 블록 " 은 프로그램 생성 및 처리에 필요한 파일을 의미하는 용어입니다 .

블록 탐색

디버깅을 위해 또는 프로그램 중단 이후 " 블록 탐색 " 기능을 사용하여 프로그램을 시작 또는 재개할 가공 프로그램의 위치를 선택할 수 있습니다 .

비동기 서브루틴

인터럽트 신호 (예 : " 고속 NC 입력 " 신호) 를 사용하여 현재 프로그램 상태와 독립적인 비동기식으로 시작할 수 있는 가공 프로그램입니다 .

사용자 메모리

가공 프로그램 , 서브루틴 , 코멘트 , 공구 옵셋 및 워크 옵셋 / 프레임 , 채널 및 프로그램 사용자 데이터와 같은 모든 프로그램 및 데이터를 CNC 사용자 메모리에 저장할 수 있습니다 .

사용자 인터페이스

사용자 인터페이스 (UI) 는 화면 형식의 CNC 용 표시 매체입니다 . 수평 및 수직 소프트키를 제공합니다 .

사용자 정의 변수

사용자가 가공 프로그램 또는 데이터 블록 (전역 사용자 데이터) 에서 원하는 용도로 변수를 선언할 수 있습니다 . 정의에는 데이터 유형 정보 및 변수 이름이 포함됩니다 . 시스템 변수를 참조하십시오 .

사용자 프로그램

S7-300 PLC 의 사용자 프로그램은 프로그래밍 언어 STEP 7 을 사용하여 생성됩니다 . 사용자 프로그램에는 모듈식 레이아웃이 있으며 개별 블록으로 구성됩니다 .

기본 블록 유형은 다음과 같습니다 .

- 코드 블록

  이러한 블록에는 STEP 7 명령이 포함됩니다 .

- 데이터 블록

  이러한 블록에는 STEP 7 프로그램 변수가 포함됩니다 .

사전 합치

경로 거리가 사용자가 지정할 수 있는 '델타'라는 끝 위치와 동일한 값에 도달하면 블록 변경이 완료됩니다 .

상위 수준의 CNC 언어

상위 수준의 언어는 다음을 제공합니다 . → 사용자 정의 변수 , → 시스템 변수 , → 매크로 기법 .

상한 돌기 보정

유도로의 마찰 조건이 변하면서 발생하는 4 분원 전이의 형상 오류는 상한 돌기 보정으로 사실상 완벽하게 제거할 수 있습니다 . 상한 돌기 보정의 설정은 회로 테스트를 통해 실행됩니다 .

서브루틴

" 서브프로그램 " 은 가공 프로그램을 메인 프로그램과 서브프로그램으로 나누던 시절에 만들어진 용어입니다 . 오늘날의 SINUMERIK NC 언어에서는 이처럼 엄격하게 두 프로그램을 구분하지 않습니다 . 원칙적으로 다른 가공 프로그램 내의 서브프로그램으로 모든 가공 프로그램 또는 싸이클을 호출할 수 있습니다 . 그러면 호출된 프로그램은 다음 프로그램 레벨 (x+1) (서브프로그램 레벨 (x+1)) 에서 실행됩니다 .

선택하기

특정 공작물을 제작하기 위해 현장에서 작동하는 NC 에 대한 일련의 명령입니다 . 마찬가지로 이 용어는 제공된 비가공 부품에 대한 특정 가공 작업 실행에 적용됩니다 .

셋팅 데이터

시스템 소프트웨어에 의해 정의된 대로 공작 기계의 속성을 NC 와 통신하는 데이터입니다 .

소재

가공하기 이전의 공작물입니다 .

소프트 키

화면 영역에 나타나는 키 이름입니다 . 표시된 소프트 키 선택은 작동 조건에 맞게 동적으로 적용됩니다 . 자유롭게 할당 가능한 기능 키 (소프트 키) 에는 소프트웨어의 정의된 기능이 할당됩니다 .

소프트웨어 한계 스위치

소프트웨어 한계 스위치는 축의 이송 범위를 제한하고 하드웨어 한계 스위치에 의한 갑작스러운 이송 정지를 방지합니다 . 두 개의 값 쌍을 각 축에 대해 지정하고 PLC 를 통해 개별적으로 활성화할 수 있습니다 .

속도 제어

블록당 약간의 이동이 있을 때 적용 가능한 이송 속도를 실행하기 위해 여러 블록에 대한 예측 평가 (룩 어헤드 참조) 를 지정할 수 있습니다 .

스플라인 보간

스플라인 보간을 통해 컨트롤러는 설정된 형상의 몇 가지 지정된 보간 지점으로부터 부드러운 곡선 특성을 생성할 수 있습니다 .

스핀들 정위치 정지

예를 들어 , 특정 위치에서 추가 가공을 실시할 수 있도록 지정된 각도 위치에서 공작물 스핀들이 정지합니다 .

시스템 메모리

시스템 메모리는 다음과 같은 데이터가 저장되는 CPU 의 메모리입니다 .

- 운영체제가 필요로 하는 데이터
- 연산 시간 , 카운터 , 마커

시스템 변수

프로그래머가 가공 프로그램에 입력하지 않고 존재하는 변수입니다 . 문자 **$** 를 데이터 유형 및 변수 이름 앞에 표시하여 정의됩니다 . 사용자 정의 변수를 참조하십시오 .

식별자

DIN 66025에 따라 변수(산술 변수, 시스템 변수, 사용자 변수), 서브루틴, 키워드 및 여러 주소 문자를 가진 워드에 대한 식별자 (이름) 를 사용합니다 . 이러한 식별자는 블록 형식과 관련된 워드와 동일한 의미를 가집니다 . 식별자는 고유해야 합니다 . 다른 개체와 동일한 식별자를 사용할 수 없습니다 .

싸이클

공작물에 대해 반복적인 가공 작업 실행을 위한 보호된 서브루틴입니다 .

안전 기능

제어 시스템에는 공작물 , 공구 또는 기계에 대한 손상을 상당히 방지하도록 시기 적절한 방법으로 CNC, PLC 및 기계의 결함을 감지하는 영구 활성 모니터링 기능이 장착되어 있습니다 . 결함이 발생할 경우 가공 작업이 중단되고 드라이브가 정지됩니다 . 고장 원인이 기록되고 알람으로 출력됩니다 . 동시에 PLC 에 CNC 알람이 트리거 되었음을 통지합니다 .

알람

모든 메시지 및 알람은 화면 조작반 상에 날짜 , 시간 및 취소 기준에 대한 해당 기호와 함께 일반 텍스트로 표시됩니다 . 알람 및 메시지가 개별적으로 표시됩니다 .

1. 가공 프로그램에서 알람 및 메시지 :

   알람 및 메시지는 가공 프로그램에서 직접 일반 텍스트로 표시할 수 있습니다 .

2. PLC 알람 및 메시지

   기계의 알람 및 메시지는 PLC 프로그램에서 일반 텍스트로 표시할 수 있습니다 . 이러한 목적으로 추가 기능 블록 패키지가 필요 없습니다 .

압축파일

외부 메모리 장치의 파일 및 / 또는 디렉토리를 출력합니다 .

연결 케이블

연결 케이블은 양쪽 끝에 커넥터가 달려 있는 2 선 케이블입니다 . 연결 케이블은 다중점 인터페이스 (MPI) 를 통해 CPU 를 프로그래밍 장치나 다른 CPU 에 연결하는 역할을 합니다 .

연속 경로 모드

연속 경로 제어 모드의 목적은 가공 프로그램 블록 경계에서 급격한 감속을 방지하고 가능한 일정한 경로 속도를 사용하여 다음 블록으로 전환하는 데 있습니다 .

오버라이드

수동 또는 프로그램 가능 제어 기능이며 이를 사용하여 특정 공작물 또는 재료에 적용하기 위한 프로그래밍된 이송 속도 또는 속도를 재지정할 수 있습니다 .

완성 부품 형상

완성 공작물 형상입니다 . 비가공 부품을 참조하십시오 .

외부 제로 옵셋

PLC 에 의해 지정된 제로 옵셋

워크 옵셋

기존 영점 및 프레임을 참조하여 좌표계에 대한 새 원점을 지정합니다 .

1. 설정 가능

   SINUMERIK 840D: 각 CNC 축에 대해 설정 가능 워크 옵셋의 구성 가능한 수를 사용할 수 있습니다 . G 코드를 통해 선택된 옵셋이 적용됩니다 .

2. 외부

   공작물 워크 옵셋의 위치를 정의하는 모든 옵셋 외에도 핸드휠 (DRF 옵셋) 이나 PLC 를 사용하여 외부 워크 옵셋을 재지정할 수 있습니다 .

3. 프로그램 가능

   `TRANS` 명령문을 사용하여 모든 경로 및 위치 지정 축에 대해 워크 옵셋을 프로그래밍할 수 있습니다 .

원점

공작 기계 상에서 축 측정 시스템의 기준이 되는 위치입니다 .

원호 보간

공구가 제공된 이송 속도에서 형상의 지정된 지점 사이를 원을 그리며 이동하고 그에 따라 공작물이 가공됩니다 .

위치 지정 축

공작 기계에서 보조 이동을 수행하는 축입니다 (예 : 공구 매거진 , 팰릿 이송). 위치 지정 축은 경로 축으로 보간되지 않는 축입니다 .

이송 범위

선형 축의 최대 허용 가능한 이송 범위는 ± 9 도입니다 . 절대값은 선택한 입력 및 위치 제어 해상도와 측정 단위 (인치 또는 미터) 에 따라 달라집니다 .

인치 측정 시스템

인치 및 인치 분수 단위로 거리를 정의하는 측정 시스템입니다 .

인터럽트 루틴

인터럽트 루틴은 가공 프로세스의 이벤트 (외부 신호) 에 의해 시작할 수 있는 특정 서브루틴입니다 . 현재 작동 중인 가공 프로그램 블록이 중단되고 중단 지점에서의 축 위치가 자동으로 저장됩니다 .

자동

시스템의 동작 모드 (DIN 에 따라 블록 시퀀스 동작): 가공 프로그램을 지속적으로 선택 및 실행하는 NC 시스템의 작동 모드입니다 .

작동 모드

SINUMERIK 제어 시스템의 작동 개념입니다 . 다음과 같은 모드가 정의됩니다 . → 조그 → MDA → 자동

작업 메모리

RAM 메모리는 애플리케이션 프로그램 처리 시 프로세서가 액세스하는 CPU 내의 메인 작업 메모리입니다 .

작업 영역

기계 공구의 물리적 디자인을 고려하여 공구 팁을 3 차원 영역으로 이동할 수 있습니다 . 보호 영역을 참조하십시오 .

작업 영역 제한

작업 영역 제한을 지원하여 한계 스위치 이외에 축 이송 범위를 추가로 제한할 수 있습니다 . 축당 하나의 값 쌍을 사용하여 보호된 작업 영역을 설명할 수 있습니다 .

저크 제한을 통한 가속

기계의 가속 응답을 최적화하는 동시에 기계 구성품을 보호하기 위해 가공 프로그램에서 급가속 및 연속 (저크 없음) 가속 간에 전환할 수 있습니다 .

전송 속도 (Baud rate)

데이터 전송 속도 (Bit/s) 입니다 .

전체 리셋

전체 리셋을 수행할 경우 다음 CPU 메모리가 삭제됩니다 .

- → 작업 메모리
- 로드 메모리의 읽기 / 쓰기 영역
- → 시스템 메모리
- → 백업 메모리

절대치

축 이동 대상은 현재 활성 좌표계 (MCS/WCS) 의 원점을 기준으로 한 치수에 의해 정의됩니다. 참조 → 증분치

접지

고장난 경우에도 위험한 접촉 전압이 전달되지 않는 장치의 연결된 모든 비활성 부품에 대해 접지합니다 .

정위치 정지

정위치 정지 명령을 프로그래밍하면 블록에서 지정된 위치로 정확하게 복귀하고 필요한 경우 매우 느리게 복귀하도록 할 수 있습니다 . 접근 시간을 줄이기 위해 , → 급이송 및 절삭 이송에 대해 정위치 정지 한계가 정의됩니다 .

정위치 정지 제한

모든 경로 축이 해당 정위치 정지 제한에 도달할 경우 제어 시스템은 정확한 대상 지점에 도달한 것처럼 반응합니다 . 가공 프로그램의 블록 선독이 수행됩니다 .

정해진 방향으로 공구 후퇴

RETTOOL: (예를 들어 , 공구가 고장난 경우) 가공이 중단되면 프로그램 명령을 사용하여 사용자가 지정한 방향으로 지정된 거리까지 공구를 이송할 수 있습니다 .

제한 속도

최대 / 최소 스핀들 속도 : 최대 스핀들 속도는 머신 데이터 , PLC 또는 셋팅 데이터를 사용하여 제한할 수 있습니다 .

조그 모드

시스템 동작 모드 (설정 모드): 조그 모드에서 기계를 설정할 수 있습니다 . 개별 축 및 스핀들은 방향 키를 통해 조그 모드에서 이송할 수 있습니다 . JOG 모드에서의 축 기능은 다음과 같습니다 . → 원점 복귀 , → 위치 지정 및 프리셋 (실제 값 설정).

좌표계

기계 좌표계 (MCS), 공작물 좌표계 (WCS) 를 참조하십시오 .

주변 장치 모듈

I/O 모듈은 CPU 및 프로세스 간의 링크를 표시합니다 .

I/O 모듈은 다음과 같습니다 .

- → 디지털 입 / 출력 모듈
- → 아날로그 입 / 출력 모듈
- → 시뮬레이터 모듈

주소

어드레스는 특정 연산자 또는 연산자 범위의 인식자 입니다 . 예 , 입력 , 출력 , 등 .

중간 블록

선택한 공구 옵셋 (G41/G42) 에 의한 작동을 제한된 수의 중간 블록으로 중단할 수 있으며 공구 옵셋을 계속해서 정확하게 보정할 수 있습니다 . 제어 시스템이 미리 인식하는 허용 가능한 중간 블록 수는 시스템 파라미터로 설정할 수 있습니다 .

증분

증분 값에 기반한 이송 경로 길이 지정입니다 . 증분 값은 셋팅 데이터로 저장하거나 적절한 키 (예 : 10, 100, 1000, 10000) 를 통해 선택할 수 있습니다 .

증분치

증분치 : 축 이송 대상은 이미 도달된 지점을 기준으로 이동할 위치 및 방향을 기준으로 정의됩니다 . 절대치를 참조하십시오 .

직렬 RS-232-C 인터페이스

데이터 입 / 출력용으로 PCU 20 에는 한 개의 V.24 직렬 인터페이스 (RS232) 가 있는 반면 PCU 50/70 에는 두 개의 V.24 인터페이스가 있습니다 . 가공 프로그램 , 제조회사 및 사용자 데이터를 이 인터페이스를 통해 로드하고 저장할 수 있습니다 .

직선 보간

공구는 공작물을 가공하는 동안 대상 지점까지 직선 방향으로 이송합니다 .

직선 축

로터리 축과 반대로 직선 축은 직선으로 이동합니다 .

진단

1. 제어 시스템의 작동 영역입니다 .
2. 제어 시스템에는 다음과 같은 서비스를 위한 테스트 기능 뿐만 아니라 자체 진단 프로그램이 있습니다 . 상태 , 알람 및 서비스 디스플레이

채널

채널은 다른 채널과 무관하게 가공 프로그램을 처리할 수 있다는 사실을 기준으로 특성화됩니다 . 채널은 할당된 축 및 스핀들만 제어합니다 . 여러 채널에서 실행중인 가공 프로그램의 상태를 동기화를 통해 좌표화 할 수 있습니다 .

축

기능 범위에 따라 CNC 축은 다음과 같이 세분화됩니다 .

- 축 : 경로 축 보간
- 보조 축 : 축 관련 이송 속도를 가진 비보간 이송 축 및 위치 지정 축 . 보조 축은 예를 들어 공구 피더 , 공구 매거진과 같이 실제 가공과 관련 없습니다 .

축 식별자 (이름)

축 이름은 DIN 66217 에 정의된 바와 같이 직교 좌표로 X, Y 및 Z 로 정의됩니다 . 직교 좌표계 .

X, Y 및 Z 주위를 회전하는 로터리 축은 A, B 및 C 를 사용하여 식별됩니다 . 지정된 축과 평행하게 위치한 추가 축은 다른 문자를 사용하여 지정할 수 있습니다 .

축 이름

축 식별자 (이름) 를 참조하십시오 .

축 주소

축 식별자 (이름) 를 참조하십시오 .

치수 지정 , 미터 및 인치 단위

가공 프로그램에서 위치 및 리드 값을 인치 단위로 프로그래밍할 수 있습니다 . 프로그램 가능한 치수 (G70/G71) 와 관계 없이 제어기가 기본 시스템으로 설정됩니다 .

키 스위치

기계 조작반에 있는 키 스위치에는 제어 시스템의 운영체제에 의해 기능이 할당된 4 가지 위치가 있습니다 . 키 스위치에는 서로 다른 세 가지 색상의 키가 있는데 , 사용자가 지정된 위치에서 제거할 수 있습니다 .

키워드

가공 프로그램에 대해 프로그래밍 언어로 정의된 의미를 가진 지정된 표시의 워드입니다 .

텍스트 편집기

편집기를 참조하십시오 .

편집기

편집기는 프로그램 / 텍스트 / 프로그램 블록을 생성 , 편집 , 확장 , 결합 및 가져올 수 있습니다 .

폴리노미얼 보간

폴리노미널 보간을 이용하면 **직선, 포물형, 지수 함수**(SINUMERIK 840D)와 같은 다양한 곡선 특성을 생성할 수 있습니다 .

표준 싸이클

자주 반복되는 가공 작업을 위해 다음과 같은 표준 사이클이 제공됩니다 :

- 드릴링 / 밀링에 대한 사이클
- - 선삭 가공

사용 가능한 사이클은 " 프로그램 " 조작 영역의 < 싸이클 지원 > 메뉴에 나열되어 있습니다 . 원하는 가공 사이클을 선택하면 값 할당에 필요한 파라미터가 일반 텍스트로 표시됩니다 .

프레임

프레임은 직교 좌표계를 다른 직교 좌표계로 변환하는 산술적 규칙입니다 . 프레임은 다음과 같은 구성요소를 포함하고 있습니다 : → 워크 옵셋 , → 회전 , → 배율 , → 미러링 .

프로그래밍 키

가공 프로그램에 대해 프로그래밍 언어로 정의된 의미를 가진 지정된 표시의 워드입니다 .

프로그램 가능 논리 제어

PLC(프로그램 가능 논리 제어) 는 전자식 제어 시스템으로 제어 장치에 프로그램으로 저장된 기능입니다 . 장치의 레이아웃 및 배선은 제어 기능에 영향을 주지 않습니다 . 프로그램 가능 논리 제어기의 구조는 컴퓨터와 동일하며 메모리가 있는 CPU(중앙 모듈), 입 / 출력 모듈 및 내부 버스 시스템으로 구성됩니다 . 주변 장치 및 프로그래밍 언어는 제어 기술 요구 사항에 맞게 적용됩니다 .

프로그램 가능 작업 영역 제한

공구의 이동 공간을 프로그래밍된 제한에 의해 정의된 공간으로 제한합니다 .

프로그램 가능한 프레임

프로그램 가능한 프레임을 사용하여 가공 프로그램이 실행되는 동안 새 좌표계 출력 지점을 동적으로 정의할 수 있습니다 . 새 프레임을 사용한 절대 정의 및 기존 시작점을 기준으로 한 추가 정의 사이에 차이가 있습니다 .

프로그램 레벨

채널에서 시작된 가공 프로그램은 프로그램 레벨 0 (메인 프로그램 레벨) 의 메인 프로그램으로 실행됩니다 . 메인 프로그램에서 호출된 가공 프로그램은 프로그램 레벨 1 ... n 까지 서브프로그램으로 실행됩니다 .

프로그램 블록

프로그램 블록에는 가공 프로그램의 메인 프로그램 및 서브루틴이 포함되어 있습니다 .

피드 오버라이드

프로그래밍된 속도는 기계 조작반 또는 PLC 를 통해 설정된 현재 속도에 의해 재지정됩니다 (0 - 200%). 또한 이송 속도는 가공 프로그램의 프로그램 가능한 백분율 요소 (1-200%) 에 의해 수정될 수 있습니다 .

피치 에러 보정

이송에 관련된 리드 스크류의 기계적 부정확성을 보정합니다 . 제어 시스템은 보정에 대해 저장된 편차 값을 사용합니다 .

하위 블록

위치 데이터와 같은 시퀀스 정보가 포함된 블록입니다 . 일반적으로 블록 번호는 "N" 으로 표시됩니다 .

형상

공작물 좌표계에서 공작물에 대한 설명입니다 .

공작물 형상입니다 .

형상

공작물 좌표계에서 공작물에 대한 설명입니다 .

공작물 형상입니다 .

형상 모니터링

형상 정확도 측정 방법으로 정의 가능한 오차 범위 내에서 다음 에러가 모니터링됩니다 . 예를 들어 적용할 수 없는 높은 에러로 인해 드라이브가 오버로드될 수 있습니다 . 이러한 경우 알람이 출력되고 축이 정지합니다 .

형상으로부터 고속 후퇴

가공이 중단되면 CNC 가공 프로그램을 통해 동작이 초기화되어 공구가 즉시 현재 가공 중인 공작물 형상에서 후퇴합니다. 후퇴 각도 및 후퇴 거리는 파라미터로 지정할 수 있습니다. 고속 후퇴가 끝난 이후에도 인터럽트 루틴을 실행할 수 있습니다 (SINUMERIK 840D).

회전

특정 각도 주위를 좌표계가 회전하도록 정의한 프레임의 구성 요소입니다 .

인덱스

Symbols

, 67, 73, 78
* (산술 함수), 64
/ (산술 함수), 64
+ (산술 함수), 64
=(b2, b3, b4, b5), 340
=(xe, x2, x3, x4, x5), 344
=(ye, y2, y3, y4, y5), 344
=(ze, z2, z3, z4, z5), 344
=(a2, a3, a4, a5), 340
== (비교 연산자), 67
> (비교 연산자), 67
>= (관계 연산자), 67
$AA_ATOL, 494
$AA_COUP_ACT, 461, 500, 526
$AA_LEAD_SP, 526
$AA_LEAD_SV, 526
$AC_ACT_PROG_NET_TIME, 692
$AC_ACTUAL_PARTS, 695
$AC_AXCTSWA, 679
$AC_AXCTSWE, 679
$AC_BLOCKTYPE, 573
$AC_BLOCKTYPEINFO, 573
$AC_CTOL, 494
$AC_CUT_INV, 453
$AC_CUTMOD, 453
$AC_CUTMOD_ANG, 453
$AC_CUTTING_TIME, 691
$AC_CYCLE_TIME, 691
$AC_FIFO1, 571
$AC_MARKER, 566
$AC_OLD_PROG_NET_TIME, 692
$AC_OLD_PROG_NET_TIME_COUNT, 692
$AC_OPERATING_TIME, 691
$AC_OTOL, 494
$AC_PARAM, 567
$AC_PROG_NET_TIME_TRIGGER, 692
$AC_REQUIRED_PARTS, 695
$AC_SMAXVELO, 490
$AC_SMAXVELO_INFO, 490
$AC_SPECIAL_PARTS, 695
$AC_SPLITBLOCK, 573
$AC_STOLF, 496
$AC_TIMER, 570
$AC_TOTAL_PARTS, 695
$AN_AXCTAS, 679
$AN_AXCTSWA, 679
$AN_LANGUAGE_ON_HMI, 882
$AN_POWERON_TIME, 691
$AN_SETUP_TIME, 691
$MC_COMPESS_VELO_TOL, 467
$P_AD, 454
$P_CTOL, 494
$P_CUT_INV, 453
$P_CUTMOD, 453
$P_CUTMOD_ANG, 453
$P_OTOL, 494
$P_SIM, 276
$P_STOLF, 496
$P_SUBPAR, 162
$P_TECCYCLE, 629
$R, 567
$Rn, 567
$SA_LEAD_TYPE, 525, 526
$SC_PA_ACTIV_IMMED, 230
$SN_PA_ACTIV_IMMED, 230
$TC_CARR1...14, 437
$TC_DP1, 393
$TC_DP10, 394
$TC_DP11, 394
$TC_DP12, 394
$TC_DP13, 394
$TC_DP14, 394
$TC_DP15, 394
$TC_DP16, 394
$TC_DP17, 394
$TC_DP18, 394
$TC_DP19, 394
$TC_DP2, 393
$TC_DP20, 394
$TC_DP21, 394
$TC_DP22, 394
$TC_DP23, 394
$TC_DP24, 394
$TC_DP25, 394
$TC_DP3, 393
$TC_DP4, 393
$TC_DP5, 393
$TC_DP6, 394
$TC_DP7, 394
$TC_DP8, 394
$TC_DP9, 394
$TC_ECPxy, 398
$TC_SCPxy, 398
$TC_TPG1 ... 9, 665, 666

Numerics

0 문자 , 75
3D 공구 반경 보정 , 413
    등거리의 3D 교차점 , 421
    외부 / 내부 코너 , 421
    원주 밀링 , 415
    전이 원 , 421
    페이스 밀링 , 416
3D 공구 옵셋 , 417
    경로 곡률 , 419
    경로를 따른 보정 , 419
    공구 오리엔테이션 , 427
    교차 절차 , 422
    삽입 깊이 , 419
    한계 표면을 사용한 3D 원주 밀링 , 423
3D 페이스 밀링 , 335
    표면 법선 벡터를 사용한 경로 곡선 , 335
3lp, 64
4 분원 오류 보정
    재인지 , 689
5 축 변환
    LEAD 및 TILT 를 통해 공구 방향 프로그램 , 333
    방향 벡터 프로그램 , 333

A

A 스플라인 , 246
A1, A2, 437
A2, 327
A3, 327
A4, 327, 335
A5, 327, 335
A6, 340
A7, 340
ABS, 64
ACC, 540
ACOS, 64
ACTBLOCNO, 174
ACTFRAME, 285
ADISPOSA, 278
Alarm
    -number, 706
ALF, 125, 127
AND, 67
APR, 41
APRB, 41
APRP, 41
APW, 41
APWB, 41
APWP, 41
AS, 209
ASIN, 64
ASPLINE, 240
ASUB, 119
ATAN2, 64
ATOL, 492
AV, 535
AX, 667
AXCTSWE, 675
AXCTSWEC, 675
AXCTSWED, 675
AXIS, 25
AXNAME, 77, 667
AXSTRING, 667
AXTOCHAN, 135
AXTOSPI, 667

B

B 스플라인 , 247
B_AND, 67
B_NOT, 67
B_OR, 67
B_XOR, 67
B2, 327
B3, 327
B4, 327, 335
B5, 327, 335
B6, 340
B7, 340
BAUTO, 240
BFRAME, 285
BLOCK, 195
BLSYNC, 121
BNAT, 240
BOOL, 25
BOUND, 71
BP, 60
BSPLINE, 240
BTAN, 240

C

C, 455
C 스플라인 , 248
C2, 327
C3, 327
C4, 327, 335
C5, 327, 335
C6, 340
C7, 340
CAC, 239

CACN, 239
CACP, 239
CALCDAT, 725
CALL, 194
CALLPATH, 199, 217
CANCEL, 635
CASE, 96
CDC, 239
CFINE, 298
CHAN, 25
CHANDATA, 218
CHAR, 25
CHECKSUM, 154
CHKDNO, 433
CIC, 239
CLEARM, 114, 618
CLRINT, 124
CMIRROR, 64, 291
COARSE, 535
COARSEA, 278
COMCAD, 253
COMPCAD, 358
COMPCURV, 253, 358
COMPLETE, 218
COMPOF, 253, 358
COMPON, 253, 358, 467
CONTDCON, 718
CONTPRON, 712
COS, 64
COUPDEF, 535
COUPDEL, 535
Couple, 455
COUPOF, 535
COUPOFS, 535
COUPON, 535
COUPONC, 535
COUPRES, 535
CP, 380
CPROT, 227
CPROTDEF, 223
CROT, 64, 291
CSCALE, 64, 291
CSPLINE, 240
CT, 675
CTAB, 513
CTABDEF, 502
CTABDEL, 509
CTABEND, 502
CTABEXISTS, 508
CTABFNO, 519
CTABFPOL, 519
CTABFSEG, 519
CTABID, 511
CTABINV, 513
CTABISLOCK, 511
CTABLOCK, 510
CTABMEMTYP, 511
CTABMPOL, 519
CTABMSEG, 519
CTABNO, 519
CTABNOMEM, 519
CTABPERIOD, 511
CTABPOL, 519
CTABPOLID, 519
CTABSEG, 519
CTABSEGID, 519
CTABSEV, 513
CTABSSV, 513
CTABTEP, 513
CTABTEV, 513
CTABTMAX, 513
CTABTMIN, 513
CTABTSP, 513
CTABTSV, 513
CTABUNLOCK, 510
CTOL, 492
CTRANS, 64, 291, 298
CUT3DC, 413, 419
CUT3DCC, 423
CUT3DCCD, 423
CUT3DF, 413
CUT3DFF, 413
CUT3DFS, 413
CUTMOD, 449
CYCLE_HSC, 809
CYCLE60, 773
CYCLE61, 749
CYCLE62, 776
CYCLE63, 782
CYCLE64, 780
CYCLE70, 771
CYCLE72, 777
CYCLE76, 756
CYCLE77, 758
CYCLE78, 742
CYCLE79, 760
CYCLE800, 805
CYCLE801, 747
CYCLE802, 744
CYCLE81, 731
CYCLE82, 732
CYCLE83, 734
CYCLE832, 808
CYCLE84, 737

CYCLE840, 740
CYCLE85, 733
CYCLE86, 736
CYCLE899, 767
CYCLE92, 799
CYCLE930, 787
CYCLE940, 790
CYCLE951, 784
CYCLE952, 801
CYCLE98, 796
CYCLE99, 793

D

D 번호
  Check, 433
  이름 변경 , 434
  자유롭게 할당 , 433
DEF, 25, 47, 626
DEFAULT, 96
DEFINE, 626
DEFINE ... AS, 209
DELAYFSTOF, 471
DELAYFSTON, 471
DELDL, 399
DELDTG, 581
DELETE, 143
DISABLE, 123
DISPLOF, 174
DISPLON, 174
DISPR, 478
DIV, 64
DL, 396
DO, 557
DV, 535

E

EAUTO, 240
EG
  전자 기어 , 527
EGDEF, 527
EGDEL, 533
EGOFC, 532
EGOFS, 532
EGON, 529
EGONSYN, 529
EGONSYNE, 529
ELSE, 106
ENABLE, 123
ENAT, 240
ENDFOR, 109
ENDIF, 106
ENDLABEL, 98
ENDLOOP, 108
ENDPROC, 589
ENDWHILE, 111
ESRR, 709
ESRS, 708
ETAN, 240
EVERY, 555
EXECSTRING, 63
EXECTAB, 724
EXECUTE, 223, 727
EXP, 64
EXTCALL, 201
EXTCLOSE, 696
EXTERN, 188
EXTOPEN, 696

F

F10, 223
F3, 688
FA, 535, 605
FALSE, 25
Faxis, 455, 521
FCTDEF, 408, 582
FCUB, 462
FENDNORM, 277
FGROUP 축 , 262
FIFO 변수 , 571
FIFOCTRL, 468
FILEDATE, 151
FILEINFO, 151
FILESIZE, 151
FILESTAT, 151
FILETIME, 151
FINE, 535
FINEA, 278
FLIN, 462
FNORM, 462
FOCOF, 620
FOCON, 620
FOR, 109
FPO, 462
FPR, 533
FRAME, 25
FROM, 555
FTOC, 591
FTOCOF, 408
FTOCON, 408
FXS, 620

FXST, 620
FXSW, 620

G

G 코드
    간접 프로그램 , 59
G0 공차 팩터 , 495
G05, 378
G07, 378
G40, 413
G450, 421
G451, 421
G62, 277
G621, 277
G810 ...G819, 276
G820 ...G829, 276
GEOAX, 670
GET, 131
GETACTTD, 435
GETD, 131
GETDNO, 434
GOTO, 93
GOTOB, 93
GOTOC, 93
GOTOF, 93
GOTOS, 92
GUD, 25, 214
GUD 변수
    동기 동작 가능 , 563

H

HOLES1, 746
HOLES2, 748

I

I1,I2, 437
ICYCOF, 631
ICYCON, 631
ID, 553
IDS, 553
IF, 93, 106
IFRAME, 285
II1,II2, 647
INDEX, 81
INICF, 25
INIPO, 25
INIRE, 25
INIT, 114
INITIAL, 218
INITIAL_INI, 218
INT, 25
INTERSEC, 722
IPOBRKA, 278
IPOENDA, 278
IPOSTOP, 535
IPTRLOCK, 476
IPTRUNLOCK, 476
ISAXIS, 667
ISD, 413, 419
ISFILE, 149
ISNUMBER, 77
ISOCALL, 196
ISVAR, 686

J

JERKLIM, 487

L

L..., 186
Label, 98
Laxis, 455, 521
LEAD, 327
LEADOF, 521
LIFTFAST, 125
LLI, 37
LLIMIT, 582
LN, 64
LOCK, 633
LONGHOLE, 769
LOOP, 108
LUD, 25

M

M, 439
$TC_CARR18, 438, 442
M17, 178
M30, 178
MASLDEF, 546
MASLDEL, 546
MASLOF, 546
MASLOFS, 546
MASLON, 546
MATCH, 81
MAXVAL, 71
MCALL, 192
MD20800, 178

MD37400, 461
MEAC, 268
MEAFRAME, 302, 306
MEAS, 265
MEASA, 268
MEAW, 265
MEAWA, 268
MINDEX, 81
MINVAL, 71
MIRROR, 285
MMC, 690
MOD, 64
MODAXVAL, 667
MOV, 600
MPF, 214, 688
MU, 377
MZ, 377

N

NC 블록 컴프레서 , 253
NCK, 25
NCU 전역 기본 프레임 , 306
NCU 전역 설정 가능한 프레임 , 306
NEWCONF, 137
NOC, 535
NOT, 67
NPROT, 227
NPROTDEF, 223
NUMBER, 77
NUT= 각도 , 340

O

OEM 주소 , 276
OEM 함수 , 276
OEMIPO1/2, 276
OFFN, 363, 367
OMA1 ... OMA5, 276
ORIAXES, 338, 354
ORIC, 427
ORICONCCW, 340, 354
ORICONCW, 340, 354
ORICONIO, 340, 354
ORICONTO, 340, 354
ORICURVE, 344, 354
ORID, 427
ORIEULER, 338, 354
ORIMKS, 336, 338
ORIPATH, 353
ORIPATHS, 353, 356
ORIPLANE, 340, 354
ORIRESET(A, B, C), 326
ORIROTA, 348
ORIROTC, 348, 354
ORIROTR, 348
ORIROTT, 348
ORIRPY, 338, 354
ORIRPY2, 338
ORIS, 427
ORISOF, 361
ORISON, 361
ORIVECT, 338, 354
ORIVIRT1, 338, 354
ORIVIRT2, 338, 354
ORIWKS, 336, 338
OS, 639
OSB, 639
OSC, 427
OSCILL, 645, 648
OSCTRL, 639
OSD, 427
OSE, 639
OSNSC, 639
OSOF, 427
OSP1, 639
OSP2, 639
OSS, 427
OSSE, 427
OST, 427
OST1, 639
OST2, 639
OTOL, 492
OVRA, 540

P

P..., 190
PCALL, 198
PDELAYOF, 653
PDELAYON, 653
PFRAME, 285
PHI, 340, 347
PO, 340, 347, 353
PHU, 39
PL, 240, 256
PO, 256
POCKET3, 751
POCKET4, 754
POLY, 256
POLYPATH, 256
PON, 662
PONS, 653

POS, 597
POSFS, 535
POSP, 645
POSRANGE, 599
POT, 64
PREPRO, 177
PRESETON, 301, 607
PRIO, 121, 125
PRLOC, 25
PROC, 164
PO, 340, 347, 353
PSI, 340, 347
PTP, 380, 385
PTPG0, 385
PUD, 25
PUNCHACC, 653
PUTFTOC, 408
PUTFTOCF, 408
PW, 240

Q

QEC, 688
QECDAT, 688
QECLRN, 688
QECLRNOF, 688
QECLRNON, 688
QECTEST, 688

R

R 파라미터 , 567
R..., 21, 23
RDISABLE, 579
READ, 145
REAL, 25
REDEF, 31
Refpos, 599
RELEASE, 131
REP, 47, 617
REPEAT, 98, 112
REPEATB, 98
REPOS, 119
REPOSA, 478
REPOSH, 478
REPOSHA, 478
REPOSL, 478
REPOSQ, 478
REPOSQA, 478
RESET, 633
RET, 179, 180
RINDEX, 81
RMB, 478
RME, 478
RMI, 478
RMN, 478
ROUND, 64
ROUNDUP, 156

S

S1, S2, 535
SAVE, 167
SBLOF, 168
SBLON, 168
SCPARA, 281
SD, 240
SD41700, 676
SD42475, 359
SD42476, 359
SD42477, 359
SD42678, 361
SD42680, 361
SD42900, 402
SD42910, 402
SD42920, 403
SD42930, 403
SD42935, 406
SD42940, 407, 452
SD42984, 450
SEFORM, 221
SET, 47, 617
SETAL, 619, 706
SETDNO, 434
SETINT, 121
SETM, 114, 618
SIEMENS cycles, 706
SIN, 64
SLOT1, 762
SLOT2, 765
SON, 653, 661, 662
SONS, 653
SPATH, 262
SPF, 214, 688
SPI, 667
SPIF1, 653
SPIF2, 653
SPLINEPATH, 251
SPN, 658
SPOF, 653
SPOS, 535
SPP, 658
SPRINT, 84

SQRT, 64
START
, 114
STARTFIFO, 468
STAT, 380, 385
STOLF, 495
STOPFIFO, 468
STOPRE, 468
STOPREOF, 580
STRING, 25
STRINGIS, 682
STRINGVAR, 83
STRLEN, 80
SUBSTR, 82
SYNFCT, 585
SYNFCT() 산정 기능 , 585
SYNR, 25
SYNRW, 25
SYNW, 25

T

TAN, 64
TANG, 455
TANGDEL, 455
TANGOF, 455
TANGON, 455
TCARR, 443
TCOABS, 443
TCOFR, 443
TCOFRX, 443
TCOFRY, 443
TCOFRZ, 443
THETA, 347, 348
THETA 를 통한 방향 벡터의 회전 프로그램 , 348
PO, 347, 353
TILT, 327
TLIFT, 455
TMOF, 665
TMON, 665
TOFFOF, 446, 594
TOFFON, 446, 594
TOLOWER, 79
TOUPPER, 79
TOWBCS, 404
TOWKCS, 404
TOWMCS, 404
TOWSTD, 404
TOWTCS, 404
TOWWCS, 404
TRAANG, 375
TRACON, 391
TRACYL, 367, 373
TRACYL 변환 , 368
TRAFOOF, 390
TRAILOF, 497
TRAILON, 497
TRANSMIT, 363, 366, 385
TRANSMIT 변환 , 364
TRANSMIT 에 대한 PTP, 385
TRAORI, 321, 324
TRUE, 25
TRUNC, 64
TU, 380, 385

U

U1,U2, 647
ULI, 37
ULIMIT, 582
UNLOCK, 633
UNTIL, 112
UPATH, 262

V

V1,V2, 437
VAR, 165
VELOLIM, 488

W

WAIT, 114
WAITC, 535
WAITE, 114
WAITENC, 681
WAITM, 114
WAITMC, 114
WHEN, 555
WHEN-DO, 648
WHENEVER, 555
WHENEVER-DO, 648
WHILE, 111
Winlimit, 599
WRITE, 138, 696

X

xe, ye, ze, 344
PO, 344, 347
XH YH ZH, 344
xi, yi, zi, 344

XOR, 67

Y

PO, 344, 347

Z

PO, 344, 347
α, 375
가공 시간 , 691
가용성
    시스템 의존 , 5
가장 가까운 경로점에서 접근 , 484
각도 기준 , 542
간격 제어 , 588
간접 프로그램 , 60
    G 코드 , 59
    어드레스 , 56
값에 따른 호출 파라미터
    고정 사이클의 경우 , 630
개요
    채널에서 프레임 활성화 , 308
검색 경로
    서브프로그램 호출 , 216
    서브프로그램 호출 시 , 160
    프로그램 가능한 검색 경로 , 199
격자 / 프레임 위치 패턴 - CYCLE801, 747
결정된 구조 , 438
결합 동작 , 612
결합 상태 , 526
경로
    상대 , 115
    절대 , 114
경로 구간 , 658
경로 기준
    셋팅 가능 , 262
경로 기준 방향
    방향 벡터의 회전 , 354
    중간 블록 삽입 , 356
경로 기준 오리엔테이션
    공구 오리엔테이션 회전 , 354
    공구의 회전 , 353
경로 밀링 - CYCLE72, 777
경로 분할 , 658, 662
경로 접선 각도 , 623
경로 지정 및 파라미터를 통한 서브프로그램 , 198
경로 축을 대상으로 경로 분할 , 661
경사 축 (TRAANG), 317
경사 축 프로그램
    G05, G07, 378
고속 가공 설정 – CYCLE832, 808
고속 절삭 - CYCLE_HSC, 809
고정 사이클 , 626
    주기적 처리 시간 제어 (ICYCOF), 631
고정 정지점 , 620
공구
    - 길이 보정 , 443
    모니터링 , 연삭 관련 , 665
    - 반경 보정 , 400
    - 보정 메모리 , 393
    오리엔테이션 , 스무딩 , 361
    - 옵셋 , 추가 , 396
    - 파라메타 , 393
    - 프레임 변경에 대한 방향 , 445
공구 반경 보정
    코너 감속 , 277
    한계 표면을 사용하지 않는 3D 원주 밀링 , 422
공구 오리엔테이션 , 427
공구 옵셋
    마모값에 대한 좌표계 , 404
    보정 메모리 , 393
    온라인 , 408, 591
공구 홀더 , 443
    구조 , 437
    데이터 삭제 / 변경 / 읽기 , 442
    전향적 가능 , 443
공작물
    계수기 , 695
    - 디렉토리 , 215
    - 메인 디렉토리 , 214
공정 싸이클
    IF 점검 구조 , 632
    계단식 , 632
    무조건 점프 , 633
    비모달식 동기화 동작 , 632
    점프 명령 (GOTOP, GOTOF, GOTOB), 633
    초기값의 기본 파라미터 , 630
공차
    G0 의 경우 , 495
관계 연산자 , 67
구조
    결정된 , 442
극 변환 , 317
기본 설정 축 식별자 , 565
기하 축
    대체 , 670
끝각 , 349
나사 밀링 - CYCLE70, 771
나사 밀링 - CYCLE78, 742
나사 선삭 - CYCLE99, 793
나사 체인 - CYCLE98, 796
내부 코너에서 코너 감속 , 277

논리 연산자 , 67
니블링 , 653, 658
다각형 - CYCLE79, 760
단일 문자 선택 , 83
단일 위치 , 337
단일 축 동작 , 663
대기 마커 , 618
대체 가능한 기하 축 , 670
동기
    미세조정 , 537
    일반 , 537
동기 동작
    삭제 , 635
    축 포지셔닝 , 597
    취소 , 635
동기 스핀들
    속도비 SRT, 541
    페어 정의 , 540
동기 태핑 - CYCLE84, 737
동기식 스핀들 , 534
    쌍 , 534
동기식 오실레이션
    검사 , 보간 싸이클 , 650
    다음 부분 절입 , 651
    동기화 동작 , 648
    반전 지점 범위에서 절입 , 649
    반전 지점에서 정지 , 650
    오실레이션 축 및 절입 축 할당 , 648
    절입 이동 , 649
    절입 정의 , 648
동기화 동작
    구문 , 552
    동작 , 557
    동작 개요 , 576
    명령어 요소 , 552
    사전 처리 변수 , 560
    적용 범위 , 553
    조건 , 555
    주 실행 변수 , 560
동기화 동작 파라미터 , 567
드릴링 - CYCLE82, 732
또는 , 67
라인 위치 패턴 - HOLES1, 746
런타임
    점검 구조의 응답 , 106
레이저 출력 제어 , 584
로타리 축
    거리 벡터 I1, I2, 437
    방향 벡터 V1, V2, 437
로타리 축의 각도 옵셋 / 각도 증분 , 440
로타리 축의 옵셋 , 440
로타리 축의 최소 위치 / 최대 위치 , 440
리드 각도 , 329
리드인 디스에이블 , 579
리딩 축 , 521
리밍 - CYCLE85, 733
링크 축 , 678
마모값 , 398
마스터 값
    커플링 , 614
마스터 값 결합
    리딩 축과 추종 축 동기화 , 524
    실제값과 지령치 결합 , 525
마스터 값 시뮬레이션 , 525
마스터 값 커플링
    스태틱 동기 동작에서 , 522
    실제값과 지령치 커플링 , 521
마찰 , 688
마커 변수 , 566
매크로 , 209
메모리
    작업 , 218
    전처리 , 468
    프로그램 메모리 , 213
명령 축 , 596
모든 코너에서 코너 감속 , 277
문자열
    - 길이 , 80
    - 연결 , 78
    - 연산 , 75
    포맷 , 84
미세조정 옵셋 , 298
밀링 개방형 슬롯 - CYCLE77, 767
밀링 공구
    -reference point (FH), 420
    -tip (FS), 420
밀링 공구 형태 , 417
반올림 , 156
반전
    지점 , 645
방향 능력이 있는 공구 홀더 , 437
    공구 홀더 번호 , 439
    시스템 변수 , 438
방향 보간 , 355
방향 특성 유연화 , 357
방향 프로그램 , 355
배열 , 47
    -element, 47
배열 인덱스 , 50
배열 정의 , 47
백래시 , 688
변수
    사용자 정의 , 25
    유형 변환 , 74, 76

- 이름 , 27
이름 , 32
정의 , 25
종류 , 25
변환
3 축 , 4 축 및 5 축 변환 (TRAORI), 314
3 축 , 4 축 변환 , 324
경사 축 , 375
방향 변환 , 314
연결 , 391
연결된 변환 , 315
운동성과 관계 없이 초기 공구 방향 설정 , 314
운동학적 변환 , 315
변환 관련 제한사항 , 389
변환 루틴 , 562
변환 유형
일반 작동 , 313
변환 , 5 축
Euler 각도 방식 프로그램 , 331
LEAD/TILT 를 사용한 프로그래밍 , 327
RPY 각도 방식 프로그램 , 332
표면 법선 벡터로 경로 곡선 프로그램 , 335
보링 - CYCLE86, 736
보정 특성 배우기 , 688
보조 기능 , 578, 658
보호
영역 , 223
분모 다항식 , 260
블록 표시 , 196
억제 , 174
비동기 오실레이션 , 639
비틀림 (Torsion), 688
사각 스피곳 - CYCLE76, 756
사각 포켓 - POCKET3, 751
사전 설정 옵셋 , 301
산술 파라미터
- 숫자 n, 21
숫자 n, 23
산술 파라미터 (R), 21, 23
삽입 깊이 , 419
삽입 깊이 (ISD), 413
상한 돌기 보정
인지 작업 활성화 , 688
학습 비활성화 , 688
서보 파라미터 세트
프로그램 가능 , 281
서브프로그램 , 157
- 반복 , 190
이름 , 158
- 파라미터 전송 없는 호출 , 186
- 파라미터 전송을 통한 호출 , 188
파라미터 지정 가능한 리턴 점프 , 180
프로그램 가능한 검색 경로 , 199
- 호출 , 간접 , 194
- 호출 , 모달 , 192
서브프로그램 모달 호출 , 192
선삭 - CYCLE951, 784
설정값 , 398
센터링 - CYCLE81, 731
소프트웨어 제한 스위치 , 605
속도 커플링 , 537
스무딩
오리엔테이션 특성 , 361
스위블 - CYCLE800, 805
스위치 OFF 위치 , 544
스테이션 / 위치 변경 , 675
스톡 제거 , 711
스플라인
- 보간 , 240
- 유형 , 246
스플라인 그룹화 , 251
스핀들
- 교체 , 131
스핀들 이동 , 611
시간 사용 점검 , 624
시뮬레이션 , 276
시스템
의존 가용성 , 5
시스템 변수 , 560
식별 번호 , 553
실린더 표면 커브 변환 , 367
실제값 메모리 프리셋 , 607
실제값 커플링 , 537
심공 드릴링 - CYCLE83, 734
싱글 블록
- 억제 , 168
싸이클
사용자 싸이클 파라미터화 , 205
싸이클 알람 , 706
알람 , 706
동기 동작에 대한 반응 , 638
어드레스
간접 프로그램 , 56
언더컷 - CYCLE940, 790
연결
문자열 , 78
연속 루프 , 108
연장된 홀 - LONGHOLE, 769
오리엔테이션
- 보간 , 342
- 축 , 341
오리엔테이션 변환 TRAORI
기계 구조 , 316
오리엔테이션 프로그래밍 , 326

오리엔테이션 프로그래밍의 변수 , 326
이송 및 오리엔테이션 이동 , 316
일반 5/6 축 변환 , 317
오리엔테이션 축 , 327, 336, 338
오리엔테이션 특성 유연화 , 354
오리엔테이션 프로그램 , 339
오버라이드
결과 , 624
현재 , 623
오실레이션
동기식 오실레이션 , 645
동기화 동작을 이용한 제어 , 645
부분 절입 , 647
비동기 , 639
비동기 오실레이션 , 639
오실레이션 동작
반전 범위 , 647
반전 지점 , 647
반전 지점에서 절입 , 650
절입 억제 , 647
온라인 공구 길이 보정 , 446, 594
옵셋 메모리 , 393
외부 워크 옵셋 , 300
운동학적 유형 , 442
운동학적 유형 M, 442
운동학적 유형 P, 442
운동학적 유형 T, 442
운전 모드
측정 도중 , 272
워크 옵셋
PRESETON, 301
외부 워크 옵셋 , 300
원 데이터
계산 , 725
원주 밀링 , 415
원주 밀링 - CYCLE77, 758
원주 밀링 (3D)
한계 표면 사용 , 423
원통 표면 변환 , 317
원호 슬롯 - SLOT2, 765
원호 위치 패턴 - HOLES2, 748
원호 포켓 - POCKET4, 754
위치 동기 , 535
위치 변경
새로운 공구를 사용하여 접근 , 485
재접근 지점 , 482
위치 특성
간접 프로그램 , 60
이동 위치 지정 , 596
이동 종료 기준
프로그램 가능 , 278
이동할 거리 삭제 , 273, 581
이동할 거리 예약 삭제 , 581
이송
축 , 646
이송 속도
이동 , 650
축 , 605
인터럽트 루틴 , 119
모달 G 코드 저장 , 120
비활성화 / 활성화 , 123
삭제 , 124
새롭게 할당 , 122
프로그래밍 이송 방향 , 127
프로그램 가능한 이송 방향 , 126
할당 및 시작 , 121
형상으로부터 고속 후퇴 , 125
후퇴 이동 , 127
일반 슬롯 - SLOT1, 762
일반 옵셋 , 298
자동 "GET", 134
자동 경로 분할 , 658
자동 중단 지점 , 477
자유 프로그래밍 위치 - CYCLE802, 744
작업
목록 , 811
작업 메모리 , 218
데이터 영역 , 218
잔여 시간
공작물 , 693
저크
옵셋 , 487
적응식 제어 (배가), 587
적응식 제어 (증가), 586
전자 기어 , 527
전처리
메모리 , 468
전처리 정지 , 580
전체 기본 프레임 , 309, 310
절단 - CYCLE92, 799
절삭날 번호 , 433
점검
구조 , 105
점프
- 대상 , 93
- 마커 , 94
마커 , 98
- 명령 , 93
- 조건 , 94
프로그램 시작으로 , 92
점프 문구
CASE, 96
접선 제어 , 455
조각 싸이클 - CYCLE60, 773

좌표계 변환 TRANSMIT, TRACYL 및 TRAANG, 317
주변 밀링 , 414
중첩 깊이
점검 구조 , 105
지령치 커플링 , 537
직교 PTP 이송 , 318
채널 관련 프레임 , 307
채널에서 첫 번째 기본 프레임 , 307
채널에서 현재 첫 번째 기본 프레임 , 309
초기 공구 오리엔테이션 설정 ORIRESET, 326
초기화
배열 , 47
배열 변수 , 617
초기화 프로그램 , 218
추적 불가능한 섹션의 획득 및 찾기 , 477
추종 축 , 521
축
경사 (TRAANG), 375
- 교체 , 131
로컬 , 678
직접 적용 , 131
커플 모션 , 499
클램핑 , 675
축 관련 마스터 값 커플링 , 521
축 관련 이송 , 605
축 교체 , 135
동기화 없이 , 133
변수 응답 설정 , 134
사전 조건 , 133
사전 처리 중지 없음 , 134
축 분리 , 134
축 적용 , 134
축 대체
동기화 동작을 이용한 확보 및 해제 , 601
축 시작 / 정지 , 600
축 좌표 , 606
축 컨테이너 , 675
축 포지셔닝
지정된 기준 위치 , 599
출력
외부 장치 / 파일로 , 696
측정 작업 상태 , 274
측정하기 , 616
카운트 루프 , 109
커플 축 조합 , 497
커플링 동작 , 497
다이나믹 한계 , 500
커플링 상태 , 500
커플링 팩터 , 497
커플링 형식 , 537
컴프레서 , 253
타이머 변수 , 570
태핑 척을 사용한 태핑 - CYCLE840, 740
트리거 이벤트
측정 도중 , 272
틸트 각도 , 329
파라미터
기계 , 393
서브프로그램 호출 시 전송 , 161
- 서브프로그램 호출 전송 , 188
실제 , 160
유형 , 160
파일
- 정보 , 151
펀칭 , 653, 658
페이스 밀링 , 413, 416
페이스 밀링 - CYCLE61, 749
폴리노미널 정의 , 582
폴리노미얼 계수 , 257
폴리노미얼 보간 , 256
분모 다항식 , 260
프레임
FRAME 체인 , 312
프레임 체인 , 296
할당 , 296
호출 , 295
프레임 계산
MEAFRAME, 302
프레임 구성 요소
FI, 294
MI, 294
SC, 294
TR, 294
프레임 구성 요소 RT, 294
프레임 변수 , 283
G 명령 G54 - G599 에 할당 , 290
값 할당 , 291
미리 정의된 프레임 변수 , 285, 295
새 프레임 정의 , 297
워크 옵셋 G54 - G599, 289
좌표 변환 호출 , 283
프로그래밍 명령
목록 , 811
프로그램
런타임 , 691
- 메모리 , 215
반복 , 190
분기 , 96
점프 , 93
초기화 , 218
프로그램 루프
IF loop, 106
REPEAT 루프 , 112
WHILE 루프 , 111

루프의 끝 , 108
카운트 루프 , 109
프로그램 메모리 , 213
파일 유형 , 214
표준 디렉토리 , 214
프로그램 섹션
반복 , 98
프로그램 섹션 반복
간접 프로그램 호출을 통한 , 195
프로그램 코디네이션
채널 수 , 116
채널 이름 , 116
프로브 상태 , 274
필요 시간
동기화 동작 , 624
한계 표면을 사용한 3D 원주 밀링 , 422
현재 NCU 전역 기본 프레임 , 308
현재 설정 가능한 프레임 , 310
현재 시스템 프레임 , 308
현재 전체 프레임 , 311
현재 채널 기본 프레임 , 309
현재 프로그램 가능한 프레임 , 310
형상
위치 변경 , 478
- 준비 , 712
- 코딩 , 718
- 테이블 , 712, 718
형상 요소
이동 , 724
형상 준비
고장 피드백 신호 , 727
형상 포켓 사전 드릴링 - CYCLE63, 782
형상 포켓 사전 드릴링 - CYCLE64, 780
형상 호출 - CYCLE62, 776
형상 홈 가공 - CYCLE952, 801
형상으로부터 고속 후퇴 , 125
홀드 블록 , 477
홈 - CYCLE930, 787
확장된 측정 함수 , 380
회전 각도 , 349
회전 각도 1, 2, 437
회전 벡터 보간 , 348, 355
회전하는 선형 축을 통한 변환 , 323
후속 동작 , 656