Cat. No.
I52E-ES-03

OMRON

Sistema Motion Control Trajexia

TJ1-MC04
TJ1-MC16

MANUAL DE PROGRAMACIÓN

Nota

Los productos OMRON se fabrican para su uso conforme a procedimientos adecuados, por un operador cualificado, y sólo para el fin descrito en este manual.
Las convenciones que aparecen a continuación se utilizan para indicar y clasificar las precauciones que aparecen en este manual. Preste atención siempre a la información que aparece en ellas. Su incumplimiento podría conllevar lesiones físicas o daños materiales.

Definición de información preventiva

PELIGRO

Indica una situación de peligro inmediato que, de no evitarse, puede ocasionar la muerte o lesiones graves.

ADVERTENCIA

Indica una situación de peligro potencial que, de no evitarse, puede ocasionar la muerte o lesiones graves.

Precaución

Indica una situación potencialmente peligrosa que, de no evitarse, puede ocasionar lesiones físicas o daños materiales menores o moderados.

Marcas comerciales y copyrights

PROFIBUS es una marca registrada de PROFIBUS International.
MECHATROLINK es una marca registrada de Yaskawa Corporation.
DeviceNet es una marca registrada de Open DeviceNet Vendor Assoc INC.
CIP es una marca registrada de Open DeviceNet Vendor Assoc INC.
Trajexia es una marca registrada de OMRON.
Motion Perfect es una marca registrada de Trio Motion Technology Ltd.

© OMRON, 2007

Reservados todos los derechos. Se prohíbe la reproducción, almacenamiento en sistemas de recuperación o transmisión total o parcial, por cualquier forma o medio (mecánico, electrónico, fotocopiado, grabación u otros) sin la previa autorización por escrito de OMRON. No se asume responsabilidad alguna con respecto al uso de la información contenida en el presente manual. Asimismo, dado que OMRON mantiene una política de constante mejora de sus productos de alta calidad, la información contenida en el presente manual está sujeta a modificaciones sin previo aviso. En la preparación de este manual se han adoptado todas las precauciones posibles. No obstante, OMRON no se hace responsable de ningún error u omisión. Tampoco se hace responsable de los posibles daños resultantes del uso de la información contenida en esta publicación.

Acerca de este manual

En este manual se describen la instalación y la operación del sistema Motion Control Trajexia.
Antes de intentar instalar o utilizar las unidades Motion Control Trajexia, se recomienda leer detenidamente el presente manual, así como toda la documentación afín relacionada en la tabla, con el objeto de familiarizarse perfectamente con la información facilitada. Asegúrese de leer las precauciones presentadas en la siguiente sección.

Nombre	Cat. No.	Contenido
Sistema Motion Control Trajexia GUÍA RÁPIDA DE REFERENCIA	I50E	Se describe cómo familiarizarse rápidamente con Trajexia en una configuración de prueba para mover un solo eje con MECHATROLINK-II.
MANUAL DE REFERENCIA DE HARDWARE del sistema Motion Control Trajexia	I51E	Se describen la instalación y las especificaciones de hardware de las unidades Trajexia y se explica la filosofía del sistema Trajexia.
Sistema Motion Control Trajexia MANUAL DE PROGRAMACIÓN	I52E	Se describen los comandos BASIC que se utilizan para la programación de Trajexia, se explican los protocolos de comunicaciones y el software Trajexia, se ofrecen ejemplos prácticos e información para la detección y corrección de errores.
Manual del servodriver Sigma-II	SIEP S800000 15	Se describen la instalación y la operación de los servodrivers Sigma-II.
Manual del servodriver de la serie JUNMA	TOEP-C71080603 01-OY	Se describen la instalación y la operación de los servodrivers JUNMA.
Manual de JUSP-NS115	SIEP C71080001	Se describen la instalación y la operación del módulo de aplicación de MECHATROLINK-II.
Manual de Sigma-III con interfaz MECHATROLINK	SIEP S800000 11	Se describen la instalación y la operación de los servodrivers Sigma-III con interfaz MECHATROLINK
Variador V7	TOEP C71060605 02-OY	Se describen la instalación y la operación de los variadores V7.
Variador F7Z	TOE S616-55 1-OY	Se describen la instalación y la operación de los variadores F7Z.
Variador G7	TOE S616-60	Se describen la instalación y la operación de los variadores G7.
Interfaz MECHATROLINK SI-T para G7 y F7	SIBP-C730600-08	Se describen la instalación y la operación de las interfaces MECHATROLINK para los variadores G7 y F7.
Interfaz MECHATROLINK ST-T/V7 para V7	SIBP-C730600-03	Se describen la instalación y la operación de las interfaces MECHATROLINK para los variadores V7.
Módulos de E/S MECHATROLINK	SIE C887-5	Se describen la instalación y la operación de los módulos de entrada y salida MECHATROLINK y el repetidor MECHATROLINK-II.
Comandos de comunicaciones de SYSMAC series CS/CJ	W342	Se describen el protocolo de comunicaciones FINS y los comandos FINS.

ADVERTENCIA
Asegúrese de leer y comprender la información incluida en este manual; en caso contrario, pueden producirse daños personales o incluso la muerte, daños en el producto o fallos del mismo. Antes de iniciar cualquiera de los procedimientos y las operaciones indicadas, lea cada sección en su totalidad y asegúrese de comprender toda la información incluida en ella y en las secciones relacionadas.

Funciones compatibles por versiones de unidad

Durante el desarrollo de Trajexia, se han incorporado nuevas funciones a la unidad controladora después de su lanzamiento al mercado.
Estas funciones están implementadas en el firmware y FPGA de la unidad controladora.
En la tabla siguiente se muestra un resumen de las funciones correspondientes en relación con la versión de firmware y FPGA de TJ1-MC__.

Funcionalidad	**Versión de firmware de TJ1-MC__**	**Versión de FPGA de TJ1-MC__**
Compatibilidad completa con TJ1-FL02	V1.6509	21 y superior
Compatibilidad con comandos BASIC FINS_COMMS	V1.6509	Todas las versiones
Compatibilidad con TJ1-DRT	V1.6509	Todas las versiones
Compatibilidad con TJ1-MC04 y TJ1-ML04	V1.6607	21 y superior

Compruebe las versiones de firmware y FPGA de TJ1-MC__

Conecte TJ1-MC__ al software Trajexia Tools. Consulte el manual de programación.
Abra la ventana de terminal y escriba los siguientes comandos:

Escriba `PRINT VERSION` en la ventana de terminal. Este comando devuelve el número de versión de firmware actual del controlador Motion.
Escriba `PRINT FPGA_VERSION SLOT(-1)` en la ventana de terminal. Este comando devuelve el número de versión de FPGA actual de TJ1-MC__.

1 Advertencias y precauciones de seguridad 16
1.1 Perfil de usuario 16
1.2 Precauciones generales 16
1.3 Precauciones de seguridad 16
1.4 Precauciones del entorno de servicio 17
1.5 Precauciones de uso 18
1.6 Precauciones de montaje de la unidad 21

2 Sistema Trajexia 22
2.1 Introducción 22
2.1.1 Hardware de Trajexia 23
2.1.2 Este manual 23
2.2 Programación multitarea en BASIC 23
2.3 Programación BASIC 24
2.3.1 Instrucciones de eje, sistema y tarea 24
2.3.2 Áreas de memoria 24
2.3.3 Estructuras de datos y variables 25
2.3.4 Especificaciones matemáticas 27
2.4 Ejecución de Motion Control 28
2.4.1 Generador de Motion Control 28
2.4.2 Secuencia 29
2.4.3 Carga de movimientos 29
2.5 Interfaz de línea de comandos 30
2.6 Programas BASIC 30
2.6.1 Gestión de programas 30
2.6.2 Compilacion de programas 31
2.6.3 Ejecución del programa 31

3 Comandos BASIC 33
3.1 Categorías 33
3.1.1 Comandos de eje 33
3.1.2 Parámetros de eje 34
3.1.3 Comandos y parámetros de comunicaciones 36
3.1.4 Constantes 36
3.1.5 Comandos, funciones y parámetros de E/S 36
3.1.6 Funciones y operandos matemáticos 37
3.1.7 Comandos de programa 37
3.1.8 Comandos de control de programa 38
3.1.9 Parámetros y modificadores de slot 38
3.1.10 Comandos y funciones del sistema 38

3.1.11 Parámetros del sistema 39
3.1.12 Comandos y parámetros de tareas 40
3.2 Todos los comandos BASIC 41
3.2.1 + (suma) 41
3.2.2 – (resta) 41
3.2.3 * (multiplicación) 41
3.2.4 / (división) 41
3.2.5 ^ (potencia) 42
3.2.6 = (es igual a) 42
3.2.7 = (asignación) 42
3.2.8 <> (no es igual a) 42
3.2.9 > (es mayor que) 43
3.2.10 >= (es mayor o igual que) 43
3.2.11 < (es menor que) 43
3.2.12 <= (es menor o igual que) 43
3.2.13 $ (entrada hexadecimal) 44
3.2.14 ' (campo de comentarios) 44
3.2.15 : (separador de instrucciones) 44
3.2.16 # 44
3.2.17 ABS 45
3.2.18 ACC 45
3.2.19 ACCEL 45
3.2.20 ACOS 45
3.2.21 ADD_DAC 46
3.2.22 ADDAX 46
3.2.23 ADDAX_AXIS 47
3.2.24 AIN 47
3.2.25 AND 47
3.2.26 AOUT 48
3.2.27 ASIN 48
3.2.28 ATAN 48
3.2.29 ATAN2 48
3.2.30 ATYPE 49
3.2.31 AUTORUN 49
3.2.32 AXIS 49
3.2.33 AXIS_DISPLAY 50
3.2.34 AXIS_ENABLE 50
3.2.35 AXISSTATUS 50
3.2.36 B_SPLINE 51
3.2.37 BASE 51

3.2.38 BASICERROR 52
3.2.39 BATTERY_LOW 53
3.2.40 BREAK_RESET 53
3.2.41 CAM 53
3.2.42 CAMBOX 55
3.2.43 CANCEL 56
3.2.44 CHECKSUM 56
3.2.45 CHR 56
3.2.46 CLEAR 57
3.2.47 CLEAR_BIT 57
3.2.48 CLEAR_PARAMS 57
3.2.49 CLOSE_WIN 57
3.2.50 CLUTCH_RATE 57
3.2.51 COMMSERROR 58
3.2.52 COMMSTYPE 58
3.2.53 COMPILE 58
3.2.54 CONNECT 59
3.2.55 CONSTANT 59
3.2.56 CONTROL 59
3.2.57 COPY 60
3.2.58 COS 60
3.2.59 CREEP 60
3.2.60 D_GAIN 60
3.2.61 D_ZONE_MAX 61
3.2.62 D_ZONE_MIN 61
3.2.63 DAC 61
3.2.64 DAC_OUT 61
3.2.65 DAC_SCALE 61
3.2.66 DATE 62
3.2.67 DATE$ 62
3.2.68 DATUM 62
3.2.69 DATUM_IN 63
3.2.70 DAY 64
3.2.71 DAY$ 64
3.2.72 DECEL 64
3.2.73 DEFPOS 64
3.2.74 DEL 65
3.2.75 DEMAND_EDGES 65
3.2.76 DEVICENET 65
3.2.77 DIR 67

3.2.78 DISABLE_GROUP 67
3.2.79 DISPLAY 67
3.2.80 DPOS 68
3.2.81 DRIVE_ALARM 68
3.2.82 DRIVE_CLEAR 69
3.2.83 DRIVE_CONTROL 69
3.2.84 DRIVE_INPUTS 70
3.2.85 DRIVE_MONITOR 70
3.2.86 DRIVE_READ 71
3.2.87 DRIVE_RESET 71
3.2.88 DRIVE_STATUS 72
3.2.89 DRIVE_WRITE 73
3.2.90 EDIT 73
3.2.91 ELSE 73
3.2.92 ELSEIF 73
3.2.93 ENCODER 74
3.2.94 ENCODER_BITS 74
3.2.95 ENCODER_CONTROL 74
3.2.96 ENCODER_ID 75
3.2.97 ENCODER_RATIO 75
3.2.98 ENCODER_READ 75
3.2.99 ENCODER_STATUS 76
3.2.100 ENCODER_TURNS 76
3.2.101 ENCODER_WRITE 76
3.2.102 ENDIF 76
3.2.103 ENDMOVE 77
3.2.104 EPROM 77
3.2.105 ERROR_AXIS 77
3.2.106 ERROR_LINE 77
3.2.107 ERRORMASK 78
3.2.108 ETHERNET 78
3.2.109 EX 79
3.2.110 EXP 79
3.2.111 FALSE 79
3.2.112 FAST_JOG 79
3.2.113 FASTDEC 80
3.2.114 FE 80
3.2.115 FE_LATCH 80
3.2.116 FE_LIMIT 80
3.2.117 FE_LIMIT_MODE 81

3.2.118 FE_RANGE 81
3.2.119 FHOLD_IN 81
3.2.120 FHSPEED 82
3.2.121 FINS_COMMS 82
3.2.122 FLAG 84
3.2.123 FLAGS 84
3.2.124 FOR..TO..STEP..NEXT 85
3.2.125 FORWARD 86
3.2.126 FPGA_VERSION 86
3.2.127 FRAC 86
3.2.128 FRAME 86
3.2.129 FREE 87
3.2.130 FS_LIMIT 87
3.2.131 FWD_IN 87
3.2.132 FWD_JOG 88
3.2.133 GET 88
3.2.134 GLOBAL 89
3.2.135 GOSUB..RETURN 89
3.2.136 GOTO 89
3.2.137 HALT 90
3.2.138 HEX 90
3.2.139 HLM_COMMAND 90
3.2.140 HLM_READ 91
3.2.141 HLM_STATUS 92
3.2.142 HLM_TIMEOUT 93
3.2.143 HLM_WRITE 94
3.2.144 HLS_NODE 95
3.2.145 HW_PSWITCH 95
3.2.146 I_GAIN 96
3.2.147 IDLE 96
3.2.148 IEEE_IN 96
3.2.149 IEEE_OUT 96
3.2.150 IF..THEN..ELSE..ENDIF 97
3.2.151 IN 97
3.2.152 INDEVICE 98
3.2.153 INITIALISE 98
3.2.154 INPUT 99
3.2.155 INT 99
3.2.156 INVERT_IN 99
3.2.157 INVERT_STEP 100

3.2.158 INVERTER_COMMAND 100
3.2.159 INVERTER_READ 101
3.2.160 INVERTER_WRITE 102
3.2.161 JOGSPEED 103
3.2.162 KEY 103
3.2.163 LAST_AXIS 104
3.2.164 LINKAX 104
3.2.165 LINPUT 104
3.2.166 LIST 105
3.2.167 LIST_GLOBAL 105
3.2.168 LN 106
3.2.169 LOCK 106
3.2.170 MARK 106
3.2.171 MARKB 107
3.2.172 MECHATROLINK 107
3.2.173 MERGE 108
3.2.174 MHELICAL 109
3.2.175 MOD 109
3.2.176 MOTION_ERROR 109
3.2.177 MOVE 110
3.2.178 MOVEABS 111
3.2.179 MOVECIRC 112
3.2.180 MOVELINK 114
3.2.181 MOVEMODIFY 115
3.2.182 MPOS 116
3.2.183 MSPEED 116
3.2.184 MTYPE 116
3.2.185 NAIO 117
3.2.186 NEG_OFFSET 117
3.2.187 NEW 117
3.2.188 NEXT 117
3.2.189 NIO 117
3.2.190 NOT 118
3.2.191 NTYPE 118
3.2.192 OFF 118
3.2.193 OFFPOS 118
3.2.194 ON 119
3.2.195 ON.. GOSUB 119
3.2.196 ON.. GOTO 119
3.2.197 OP 119

3.2.198 OPEN_WIN 120
3.2.199 OR 120
3.2.200 OUTDEVICE 121
3.2.201 OUTLIMIT 121
3.2.202 OV_GAIN 121
3.2.203 P_GAIN 122
3.2.204 PI 122
3.2.205 PMOVE 122
3.2.206 POS_OFFSET 123
3.2.207 POWER_UP 123
3.2.208 PRINT 123
3.2.209 PROC 124
3.2.210 PROC_STATUS 124
3.2.211 PROCESS 125
3.2.212 PROCNUMBER 125
3.2.213 PROFIBUS 125
3.2.214 PSWITCH 126
3.2.215 RAPIDSTOP 127
3.2.216 READ_BIT 127
3.2.217 REG_POS 128
3.2.218 REG_POSB 128
3.2.219 REGIST 128
3.2.220 REMAIN 130
3.2.221 REMOTE_ERROR 130
3.2.222 RENAME 130
3.2.223 REP_DIST 131
3.2.224 REP_OPTION 131
3.2.225 REPEAT..UNTIL 131
3.2.226 RESET 132
3.2.227 RETURN 132
3.2.228 REV_IN 132
3.2.229 REV_JOG 133
3.2.230 REVERSE 133
3.2.231 RS_LIMIT 133
3.2.232 RUN 133
3.2.233 RUN_ERROR 134
3.2.234 RUNTYPE 134
3.2.235 S_REF 135
3.2.236 S_REF_OUT 135
3.2.237 SCOPE 136

3.2.238 SCOPE_POS 137
3.2.239 SELECT 137
3.2.240 SERVO 137
3.2.241 SERVO_PERIOD 137
3.2.242 SET_BIT 138
3.2.243 SETCOM 138
3.2.244 SGN 139
3.2.245 SIN 139
3.2.246 SLOT 139
3.2.247 SPEED 139
3.2.248 SQR 140
3.2.249 SRAMP 140
3.2.250 PASO 140
3.2.251 STEP_RATIO 140
3.2.252 STEPLINE 141
3.2.253 STOP 141
3.2.254 SYSTEM_ERROR 142
3.2.255 T_REF 142
3.2.256 TABLE 143
3.2.257 TABLEVALUES 143
3.2.258 TAN 144
3.2.259 THEN 144
3.2.260 TICKS 144
3.2.261 TIME 144
3.2.262 TIME$ 145
3.2.263 TO 145
3.2.264 TRANS_DPOS 145
3.2.265 TRIGGER 145
3.2.266 TROFF 145
3.2.267 TRON 146
3.2.268 TRUE 146
3.2.269 TSIZE 146
3.2.270 UNITS 147
3.2.271 UNLOCK 147
3.2.272 UNTIL 147
3.2.273 VERIFY 147
3.2.274 VERSION 147
3.2.275 VFF_GAIN 148
3.2.276 VP_SPEED 148
3.2.277 VR 148

3.2.278 VRSTRING 149
3.2.279 WA 149
3.2.280 WAIT IDLE 150
3.2.281 WAIT LOADED 150
3.2.282 WAIT UNTIL 150
3.2.283 WDOG 151
3.2.284 WHILE..WEND 151
3.2.285 XOR 152

4 Protocolos de comunicaciones 153
4.1 Interfaces disponibles 153
4.2 Ethernet 153
4.2.1 Comunicación con Trajexia directamente desde el ordenador 154
4.2.2 Comunicación con Trajexia de forma remota 155
4.2.3 Protocolo Trajexia Tools 156
4.2.4 Protocolo de servidor FINS 156
4.2.5 Protocolo de cliente FINS 158
4.3 Protocolo serie 158
4.3.1 Maestro Host Link 158
4.3.2 Esclavo Host Link 163
4.3.3 Protocolo definido por el usuario 165
4.4 PROFIBUS 167
4.4.1 Introducción 167
4.4.2 Configuración de las comunicaciones 167
4.4.3 Estado de las comunicaciones 172
4.5 DeviceNet 173
4.5.1 Introducción 173
4.5.2 Configuración de las comunicaciones 173
4.5.3 Estado de las comunicaciones 178
4.6 MECHATROLINKII 179

5 Interfaz Trajexia Tools 180
5.1 Introducción 180
5.2 Especificaciones y conexiones 180
5.2.1 Especificaciones del PC 180
5.2.2 Instalación del software Trajexia Tools 181
5.2.3 Conexión a la TJ1-MC__ 186
5.3 Proyectos 191
5.3.1 Proyectos de Trajexia Tools 191
5.3.2 Ventana de comprobación de proyecto 192

5.4 Ventana de aplicación de Trajexia Tools 194
5.4.1 Panel de control....194
5.4.2 Barra de menú....194
5.4.3 Barra de herramientas 195
5.5 Descripciones de menús 196
5.5.1 Menú Project (Proyecto) 196
5.5.2 Menú Controller (Controlador) 198
5.5.3 Menú Program (Programa)....203
5.5.4 Menú Tools (Herramientas) 205
5.5.5 Menú Options (Opciones)....221
5.5.6 Menú Window (Ventana) 224
5.5.7 Menú Help (Ayuda)....224

6 Ejemplos y sugerencias 225
6.1 Procedimientos 225
6.1.1 Programa de inicio 225
6.1.2 Ajustes de ganancia 229
6.1.3 Ajuste del parámetro de eje UNITS y la relación de engranaje....239
6.1.4 Asignación de entradas y salidas del servodriver 251
6.1.5 Búsqueda de origen 253
6.1.6 Registro 259
6.1.7 Seguimiento y monitorización 269
6.2 Ejemplos prácticos....279
6.2.1 Programa shell 279
6.2.2 Programa de inicialización 283
6.2.3 Programa de un solo eje 286
6.2.4 Posición con detección de producto 287
6.2.5 Posición en una cuadrícula 289
6.2.6 Programa de alimentador de bolsas 291
6.2.7 Tabla CAM en un programa 293
6.2.8 Programa de sierra pendular 295
6.2.9 Programa de corrección 298

7 Detección y corrección de errores 300
7.1 Tensión y herramientas de análisis 300
7.2 TJ1-MC__ 300
7.2.1 Errores de sistema 300
7.2.2 Errores de eje 300
7.2.3 Errores de unidad 301
7.2.4 Errores de configuración 302
7.2.5 Sustituya la batería....302

7.3 TJ1-PRT 302
7.3.1 Errores de sistema 302
7.3.2 Problemas de comunicaciones de datos de E/S 303
7.4 TJ1-DRT 304
7.4.1 Errores de sistema 304
7.4.2 Problemas de comunicaciones de datos de E/S 304
7.5 TJ1-ML__ 304
7.5.1 Errores de sistema 304
7.5.2 Error de bus 304
7.6 TJ1-FL02 305
7.6.1 Errores de sistema 305

1 Advertencias y precauciones de seguridad

1.1 Perfil de usuario

Este manual está dirigido al personal con conocimientos de sistemas eléctricos (ingenieros eléctricos o equivalentes) que estén encargados del diseño, instalación y gestión de los sistemas e instalaciones de automatización industrial.

1.2 Precauciones generales

El usuario debe utilizar el producto con arreglo a las especificaciones de rendimiento descritas en este manual.
Consulte a su representante de OMRON antes de utilizar el producto en alguna situación no contemplada en este manual o de emplearlo en sistemas de control nuclear, sistemas ferroviarios, sistemas de aviación, vehículos, equipos de seguridad, plantas petroquímicas y otros sistemas, máquinas o equipos que puedan tener una repercusión grave en vidas o propiedades en caso de uso incorrecto.

1.3 Precauciones de seguridad

ADVERTENCIA
No intente desmontar la unidad ni toque ninguna de las piezas internas mientras esté conectada la alimentación.
Esto podría provocar una descarga eléctrica.

ADVERTENCIA
No toque ningún terminal o bloque de terminales mientras esté conectada la alimentación.
En caso contrario podrían producirse descargas eléctricas.

ADVERTENCIA
Bajo ninguna circunstancia cortocircuite los terminales positivo y negativo de las baterías, ni cargue, desarme, deforme, aplique presión ni queme las baterías.
Las baterías pueden explotar, quemarse o derramar líquido.

ADVERTENCIA
El usuario debe tomar medidas de protección a prueba de fallos para garantizar la seguridad en caso de que no se reciban señales o que éstas sean incorrectas o anómalas debido a cortes momentáneos de corriente u otras causas.
En caso de no hacerlo pueden producirse graves accidentes.

ADVERTENCIA
El usuario deberá instalar por su cuenta circuitos de parada de emergencia, de bloqueo y de limitación, así como otras medidas de seguridad similares, en circuitos externos (es decir, no en el controlador Motion Trajexia).
En caso de no hacerlo pueden producirse graves accidentes.

ADVERTENCIA
Cuando la salida de 24 Vc.c. (fuente de alimentación de E/S de TJ1) esté sobrecargada o cortocircuitada, puede producirse una caída de tensión que provoque la desconexión de las salidas.
Para evitar dichos problemas, deben incorporarse al sistema medidas de prevención externas que garanticen la seguridad.

ADVERTENCIA
Las salidas de TJ1 se desconectarán debido a la sobrecarga de los transistores de salida (protección). Para evitar dichos problemas, deben incorporarse al sistema medidas de prevención externas que garanticen la seguridad.

ADVERTENCIA
TJ1 desconectará WDOG cuando su función de autodiagnóstico detecte un error. Para evitar dichos problemas, deben incorporarse al sistema medidas de prevención externas que garanticen la seguridad.

ADVERTENCIA
Con el objeto de garantizar la seguridad del sistema en caso de producirse una anomalía como consecuencia de un funcionamiento incorrecto de TJ1 o de cualquier otro factor externo que afecte a éste, incorpore a los circuitos externos, es decir no al controlador Motion Trajexia (denominado “TJ1”), medidas de seguridad.
En caso de no hacerlo pueden producirse graves accidentes.

ADVERTENCIA
No intente desarmar, reparar o modificar ninguna unidad. Cualquier intento de hacerlo puede provocar desperfectos, descargas eléctricas e incluso incendios.

Precaución
Compruebe las condiciones de seguridad de la unidad de destino antes de transferir un programa a otra unidad o de editar la memoria. La realización de cualquiera de estos procesos sin confirmar las condiciones de seguridad puede provocar lesiones.

Precaución
No se realizará automáticamente una copia de seguridad de los programas de usuario escritos para la unidad Motion Control en la memoria flash de TJ1 (función de memoria flash).

Precaución
Preste especial atención a la polaridad (+/–) cuando realice el cableado de la fuente de alimentación de c.c. Una conexión errónea puede provocar el funcionamiento incorrecto del sistema.

Precaución
Apriete los tornillos del bloque de terminales de la fuente de alimentación de c.a. al par de apriete especificado en este manual. Los tornillos flojos pueden provocar incendios o un funcionamiento incorrecto.

1.4 Precauciones del entorno de servicio

Precaución
No utilice la unidad en ninguno de los siguientes lugares. Esto podría provocar un funcionamiento incorrecto, descargas eléctricas o incendios.
- Lugares expuestos a la luz directa del sol.
- Lugares expuestos a temperaturas o grados de humedad fuera de los rangos determinados en las especificaciones.
- Lugares expuestos a condensación como resultado de cambios drásticos de temperatura.
- Lugares expuestos a gases corrosivos o inflamables.
- Lugares con altas cantidades de polvo (especialmente polvo de hierro) o sal.
- Lugares expuestos al contacto con agua, aceite o productos químicos.
- Lugares sometidos a golpes o vibraciones.

Precaución
En el momento de realizar instalaciones en los siguientes lugares, adopte las suficientes medidas de seguridad.
Unas medidas inadecuadas o insuficientes pueden provocar un funcionamiento incorrecto.
- Lugares expuestos a electricidad estática u otras formas de ruido.
- Lugares expuestos a fuertes campos electromagnéticos.
- Lugares expuestos a una posible radioactividad.
- Lugares cercanos a fuentes de alimentación.

Precaución
El entorno de servicio del sistema TJ1 puede tener un efecto muy importante en la vida útil y en la fiabilidad del sistema.
Los entornos de funcionamiento inadecuados pueden provocar un funcionamiento incorrecto, averías y otros problemas imprevistos en el sistema TJ1.
Asegúrese de que el entorno de funcionamiento cumple las condiciones especificadas, tanto durante la instalación como durante toda la vida del sistema.

1.5 Precauciones de uso

ADVERTENCIA
No arranque el sistema hasta que haya comprobado que los ejes están presentes y son del tipo correcto.
Los números de los ejes flexibles cambiarán si se producen errores de la red MECHATROLINK-II durante el arranque o si cambia la configuración de dicha red.

ADVERTENCIA
Asegúrese de que el programa del usuario puede ejecutarse correctamente antes de ejecutarlo en la unidad.
De lo contrario puede producirse un funcionamiento imprevisto.

Precaución
Utilice siempre la tensión de alimentación indicada en este manual.
Una tensión incorrecta puede provocar un funcionamiento incorrecto o un incendio.

Precaución
Adopte las medidas apropiadas para garantizar que se suministra la alimentación con la tensión y frecuencia nominal especificada.
Tenga especial cuidado en lugares en los que la alimentación eléctrica sea inestable.
Una alimentación inapropiada puede provocar un funcionamiento incorrecto.

Precaución
Instale disyuntores externos y tome otras medidas de seguridad frente a cortocircuitos en cableados externos.
En caso de que no se tomen las suficientes medidas de seguridad para prevenir cortocircuitos, puede producirse un incendio.

Precaución
No aplique a las unidades de entrada una tensión superior a la tensión nominal de entrada.
Un exceso de tensión puede provocar un incendio.

Precaución
No aplique tensión ni conecte cargas a las unidades de salida que superen la capacidad de conmutación máxima.
Los excesos de tensión o de carga pueden provocar incendios.

Precaución
Durante la realización de pruebas de aislamiento de tensión, desconecte el terminal de puesta a tierra.
De lo contrario, puede producirse un incendio.

Precaución
Al instalar las unidades, conéctelas siempre a una toma de tierra de clase 3 (hasta 100Ω o menos).
El no conectar a una toma de tierra de clase 3 puede provocar descargas eléctricas.

Precaución
Desconecte siempre la alimentación del sistema antes de proceder a realizar cualquiera de las siguientes tareas.
De lo contrario, puede producirse un funcionamiento incorrecto o descargas eléctricas.
- Montaje o desmontaje de las unidades expansoras, CPUs u otras unidades.
- Ensamblado de las unidades.
- Configuración de los interruptores DIP o de los interruptores rotativos.
- Conexión de cables o cableado.
- Conexión o desconexión de los conectores.

Precaución
Asegúrese de que todos los tornillos de montaje, de terminal y del conector de cable están apretados al par especificado en este manual.
El uso de un par inapropiado puede provocar un funcionamiento incorrecto.

Precaución
Durante el cableado, deje pegada la etiqueta de protección al polvo a la unidad.
Si se quita la etiqueta de protección al polvo puede provocar un funcionamiento incorrecto.

Precaución
Una vez concluido el cableado, retire la etiqueta de protección al polvo para permitir una adecuada disipación térmica.
Dejar la etiqueta de protección al polvo pegada puede provocar un funcionamiento incorrecto.

Precaución
Utilice terminales de crimpar para el cableado. No conecte cables trenzados pelados directamente a los terminales.
La conexión de cables trenzados desnudos puede provocar un incendio.

Precaución
Antes de conectar la alimentación eléctrica, vuelva a comprobar el cableado.
Un cableado incorrecto puede provocar un incendio.

Precaución
Realice un cableado correcto.
Un cableado incorrecto puede provocar un incendio.

Precaución
Monte la unidad sólo después de haber comprobado exhaustivamente el bloque de terminales.

Precaución
Asegúrese de que los bloques de terminales, los cables de expansión y demás elementos con dispositivos de bloqueo están situados adecuadamente.
De lo contrario, podría producirse un funcionamiento incorrecto.

Precaución
Confirme que no se producirá ningún efecto negativo adverso en el sistema antes de cambiar el modo de operación del sistema. En caso de no hacerlo, puede producirse un funcionamiento imprevisto.

Precaución
Reanude el funcionamiento sólo cuando haya transferido a la nueva CPU el contenido de las memorias VR y tabla necesarias para el funcionamiento.
En caso de no hacerlo, puede producirse un funcionamiento imprevisto.

Precaución
Cuando sustituya alguna pieza, asegúrese de comprobar que los valores nominales de la nueva pieza sea la correcta. De lo contrario podría producirse un funcionamiento incorrecto o un incendio.

Precaución
No tire de los cables ni los doble más allá de sus límites naturales. Los cables podrían romperse.

Precaución
Antes de tocar el sistema, toque antes un objeto metálico conectado a tierra para descargarse de la electricidad estática que pudiera haber acumulado.
De lo contrario, se podría producir un funcionamiento incorrecto o daños.

Precaución
Los cables UTP no están apantallados. En los entornos expuestos a ruidos, utilice cables de par trenzado apantallado (STP) y concentradores adecuados para un entorno FA.
No instale cables de par trenzado con líneas de alta tensión.
No instale cables de par trenzado cerca de dispositivos que generen ruido.
No instale cables de par trenzado en lugares que estén expuestos a altos niveles de humedad.
No instale cables de par trenzado en lugares que estén expuestos a polvo y suciedad excesivos o a vapores de grasa u otros contaminantes.

Precaución
Utilice cables de conexión dedicados, según lo especificado en los manuales de operación, para conectar las unidades. El uso de cables RS-232C estándar para ordenador puede provocar fallos en los dispositivos externos o la unidad de Motion Control.

Precaución
Las salidas pueden permanecer conectadas debido a un funcionamiento incorrecto de las salidas transistor incorporadas u otros circuitos internos. Para evitar dichos problemas, deben incorporarse al sistema medidas de prevención externas que garanticen la seguridad.

Precaución
TJ1 comenzará a funcionar en modo RUN cuando se conecte la alimentación si un programa BASIC está configurado en modo de ejecución automática.

1.6 Precauciones de montaje de la unidad

Precaución
Instale la unidad correctamente.
Una instalación incorrecta de la unidad puede provocar un funcionamiento incorrecto.

Precaución
Asegúrese de montar la terminación incluida con la TJ1-MC__ en la unidad instalada más a la derecha.
A menos que la terminación se monte correctamente, la TJ1 no funcionará de forma adecuada.

2 Sistema Trajexia

2.1 Introducción

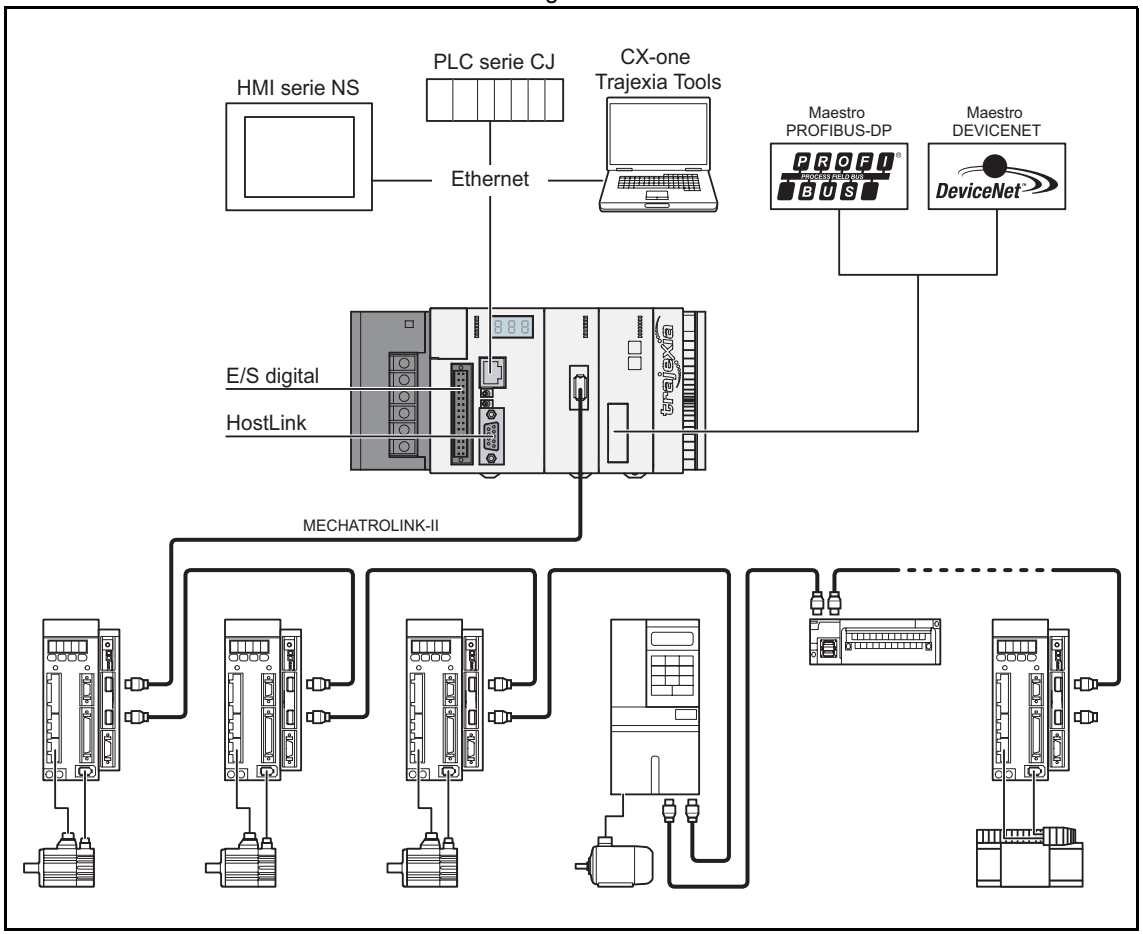
Trajexia es una plataforma Motion Control de OMRON que ofrece el rendimiento y la facilidad de uso de un sistema Motion Control dedicado.

Trajexia es un sistema modular independiente que permite la máxima flexibilidad y escalabilidad. En el corazón de Trajexia se encuentra el coordinador de Motion Control multitarea TJ1. Reforzado con un DSP de 32 bits, puede realizar tareas de Motion Control, como levas electrónicas, engranaje electrónico, control de registro e interpolación, todo ello mediante simples comandos Motion.

Trajexia ofrece un control perfecto de hasta 16 ejes mediante un bus de Motion Control MECHATROLINK-II, o control analógico o de impulsos tradicional, con posicionamiento independiente, control de par y velocidad para cada eje. Y además su potente juego de instrucciones de Motion Control hace que la programación se realice de forma intuitiva y sencilla.

Puede seleccionar de entre una amplia gama de los mejores servosistemas rotativos, lineales, Direct-Drives y variadores. El sistema es escalable hasta 16 ejes y 8 dispositivos no ejes (variadores o módulos de E/S).

fig. 1



2.1.1 Hardware de Trajexia

El hardware de Trajexia se describe en el manual de referencia de hardware de Trajexia. Se recomienda leer dicho manual en primer lugar.
El sistema Trajexia ofrece estas ventajas:

Conectividad directa mediante Etherne
El puerto integrado Ethernet de Trajexia proporciona conectividad directa y rápida a PCs, PLCs, HMIs y otros dispositivos, a la vez que ofrece acceso completo a los drivers mediante un bus de Motion Control MECHATROLINK-II. Permite mensajes explícitos a través de Ethernet y mediante MECHATROLINK-II con el fin de proporcionar transparencia completa hasta el nivel de actuador, lo que posibilita el acceso remoto.

Protección de los programas
El método de cifrado de Trajexia garantiza la protección y confidencialidad de sus programas.

Puerto serie y E/S locales
Un puerto serie proporciona conexion directa con cualquier PLC de OMRON, HMI o cualquier otro dispositivo de campo. Con las 16 entradas y 8 salidas completamente configurables incluidas en el controlador puede adaptar el sistema Trajexia al diseño de su máquina.

Maestro MECHATROLINK-II
El maestro MECHATROLINK-II realiza el control de un máximo de 16 dispositivos, a la vez que ofrece una completa transparencia en todo el sistema. MECHATROLINK-II ofrece la velocidad de comunicación y la precisión esenciales para garantizar un perfecto control de los servos del sistema. El tiempo de ciclo de Motion Control se puede seleccionar entre 0,5 ms, 1 ms ó 2 ms.

TJ1-FL02 (unidad de eje flexible)
La TJ1-FL02 permite el completo control de dos actuadores mediante una salida analógica o tren de impulsos. El módulo admite los principales protocolos de encoder absolutos, con lo que se puede conectar un encoder externo al sistema.

Drivers y variadores
Hay disponible una amplia variedad de servos rotativos, lineales y Direct-Drive así como variadores para ajustarse a sus necesidades de tamaño compacto, rendimiento y fiabilidad. Los variadores conectados a MECHATROLINK-II están controlados con el mismo tiempo de ciclo de actualización que los servodrivers.

E/S remotas
Las E/S del bus Motion Control MECHATROLINK-II proporcionan la posibilidad de expansión del sistema a la vez que mantienen los dispositivos en un solo bus Motion Control.

PROFIBUS-DP
El esclavo PROFIBUS-DP permite conectar una red PROFIBUS en la máquina.

DeviceNet
El esclavo DeviceNet permite conectar una red DeviceNet en la máquina.

2.1.2 Este manual

Este manual de programación ofrece información dedicada a:
- La descripción y el uso de los comandos BASIC
- Los protocolos de comunicaciones necesarios para el sistema Trajexia
- El uso y la descripción de los componentes del interfaz de Trajexia Tools
- Ejemplos de programación y buenas prácticas de programación
- Detección y corrección de errores y búsqueda de fallos

2.2 Programación multitarea en BASIC

Las unidades TJ1-MC__ (CPU del Controlador Motion) ofrecen la posibilidad de programación multitarea en lenguaje de programación BASIC. El lenguaje Motion Control se basa en un lenguaje BASIC de token y los programas se compilan en el formato de token antes de ejecutarse.

La multitarea es fácil de configurar y usar, y permite la programación de máquinas muy complejas. La multitarea ofrece a la TJ1-MC__ una ventaja considerable sobre sistemas monotarea equivalentes. Permite aplicaciones modulares en las que los procesos conectados lógicamente se pueden agrupar en el mismo programa de tarea, lo que simplifica la arquitectura y el diseño del código.
La TJ1-MC__ puede ejecutar hasta 14 programas si lo permite el tamaño de la memoria. El usuario controla la ejecución de los programas mediante BASIC. Los comandos, funciones y parámetros BASIC presentados aquí se pueden encontrar en el capítulo 3.

2.3 Programación BASIC

El lenguaje BASIC consta, entre otros elementos, de comandos, funciones y parámetros. Estas instrucciones BASIC son los componentes que se proporcionan para controlar la operación de la TJ1-MC__.
Los comandos son palabras reconocidas por el procesador que llevan a cabo una determinada acción pero no devuelven un valor. Por ejemplo, **PRINT** es una palabra reconocida que provocará que el valor de las funciones o variables siguientes se impriman en un determinado dispositivo de salida.
Las funciones son palabras reconocidas por el procesador que llevan a cabo una determinada acción y devuelven un valor relacionado con dicha acción. Por ejemplo, **ABS** toma el valor de su parámetro y devuelve el valor absoluto del mismo para que lo utilice otra función o comando. Por ejemplo, **ABS(-1)** devolverá el valor 1, que puede usar el comando **PRINT** para generar una cadena que se enviará a un determinado dispositivo.
Los parámetros son palabras reconocidas por el procesador que contienen un determinado valor. Este valor se puede leer y, si no es de sólo lectura, escribir. Los parámetros se utilizan para determinar y monitorizar el comportamiento del sistema. Por ejemplo, **ACCEL** determina la aceleración de un movimiento para un determinado eje.

2.3.1 Instrucciones de eje, sistema y tarea

Los comandos, las funciones y los parámetros se aplican a cualquiera de los ejes, las tareas en ejecución o el sistema general.

Instrucciones de eje

Los comandos y los parámetros de eje se aplican a uno o varios ejes. Los parámetros de eje determinan y monitorizan el modo en que un eje reacciona a los comandos indicados y su comportamiento en el mundo exterior. Cada eje tiene un conjunto de parámetros, por lo que todos los ejes pueden funcionar independientemente entre sí. Los comandos pueden controlar uno o varios ejes simultáneamente, mientras que cada eje tiene su propio comportamiento. Los parámetros se resetean a sus valores predeterminados en cada arranque.
Los comandos y parámetros trabajan sobre algun eje o grupo de ejes, lo que se especifica con el comando **BASE**. El comando **BASE** se utiliza para cambiar este grupo de ejes y cada tarea tiene su propio grupo que se puede cambiar en cualquier momento. El eje predeterminado es 0.
Los comandos o parámetros también se pueden programar para funcionar temporalmente en un eje si se incluye la función **AXIS** como modificador en el comando dependiente de eje. Una asignación temporal solo es efectiva para un comando o parámetro si tras el se incluye el modificador **AXIS**.

Instrucciones de tarea

Los parámetros de tarea se aplican a una sola tarea. Los parámetros de tarea monitorizan la tarea para por ejemplo, el tratamiento de errores. El modificador **PROC** permite al usuario acceder a un parámetro de una determinada tarea. Sin **PROC** se supone la tarea actual. El comando **BASE** (consulte más arriba) es específico de la tarea y se puede utilizar con el modificador **PROC**.

Instrucciones de sistema

Estas instrucciones rigen todas las funciones del sistema y son básicamente todas las instrucciones que no pertenecen a los dos primeros grupos.

2.3.2 Áreas de memoria

En la unidad Motion Control Trajexia se pueden identificar tres áreas de memoria principales:
- Memoria de E/S.
- Memoria VR.
- Memoria TABLE.

Memoria de E/S

La memoria de E/S se utiliza para contener el estado de los dispositivos de entrada y salida conectados al sistema Trajexia. Se divide en dos subáreas: una para la memoria de E/S digitales y otra para la memoria de E/S analógicas. La memoria de E/S digitales contiene los estados de entrada y salida de los dispositivos de E/S digitales. Su capacidad es de 256 bits (puntos de entrada) para la entrada y de 256 bits (puntos de salida) para las salidas. A las entradas de esta memoria se puede acceder mediante el comando **IN**. A las salidas se puede acceder mediante el comando **OUT**.
La memoria de E/S analógicas contiene los valores de entrada y salida de los dispositivos de E/S analógicas. Su capacidad es de 36 canales de entrada y 36 canales de salida. A los canales de entrada analógica se puede acceder mediante el comando **AIN**. A los canales de salida analógica se puede acceder mediante el comando **AOUT**.

Memoria VR

La memoria VR se utiliza habitualmente si algunos datos tienen que ser globales, lo que significa que se puede acceder a ellos desde todos los programas del proyecto a la vez. El tamaño de esta memoria es de 1.024 entradas de la direccion 0 la 1.023. Una entrada de memoria se direcciona con la macro **VR(x)** donde **x** es el direccion de la entrada de memoria VR. La memoria VR está accesible para lectura y escritura. La escritura se lleva a cabo mediante la asignación matemática con el comando **=** en el programa. El contenido de esta memoria se retiene en la memoria RAM alimentada por baterías y se conserva durante la desconexión. La memoria VR también se conserva al cambiar la batería, si se realiza rápidamente.

Memoria TABLE

TABLE se utiliza habitualmente si algunos datos tienen que ser globales, lo que significa que se puede acceder a ellos desde todos los programas del proyecto a la vez. Mientras que la memoria VR se utiliza para propósitos similares con el fin de definir datos y valores globales, la memoria TABLE se utiliza para cantidades mucho mayores de datos globales, por lo que es necesario organizarlos en un determinado orden. Por este motivo, la memoria TABLE se utiliza habitualmente para almacenar tablas de datos, perfiles de Motion Control, registrar datos, etc. Algunos comandos BASIC que proporcionan datos de este tipo y tamaño, por ejemplo **SCOPE**, **CAM**, **CAMBOX** etc., requieren el uso de la memoria TABLE para escribir sus resultados. El tamaño de esta memoria es de 64.000 entradas desde la direccion 0 a 63.999. A la memoria TABLE se puede acceder para lectura y también para escritura, pero la forma en que se accede es distinta para estas dos operaciones. Antes leer una determinada entrada de memoria TABLE, se tiene que definir y escribir en primer lugar con el comando **TABLE(x, valor1, valor2,…)** donde **x** es la primera direccion de memoria TABLE que se definirá y **valor1**, **valor2**, ... son los valores que se escriben en la memoria TABLE en las direcciones x, x + 1, ... Una entrada de memoria TABLE, una vez definida y escrita, se puede leer con el comando **TABLE(x)**, donde **x** es la direccion de la entrada de memoria TABLE. Un intento de leer una entrada de memoria TABLE sin definir da como resultado un mensaje de error de la TJ1-MC__. El contenido de la memoria TABLE se retiene en la memoria RAM alimentada por baterías y se conserva durante la desconexión. La memoria TABLE también se conserva al cambiar la batería, si se realiza rápidamente.

2.3.3 Estructuras de datos y variables

Los programas BASIC pueden almacenar datos numéricos en varios tipos de variables. Algunas variables tienen funciones predefinidas, como los parámetros de eje y los parámetros del sistema; otras variables están a disposición del programador para que las defina según lo necesite en la programación. Las variables de TABLE, globales y locales de la TJ1-MC__ se explican en esta sección. Además, también se especificará el uso de etiquetas.

Variables de TABLE

TABLE es una estructura de matriz que contiene una serie de números. Estos números se utilizan, por ejemplo, para especificar posiciones en el perfil para un comando **CAM** o **CAMBOX**. También se pueden utilizar para almacenar datos para un uso posterior, por ejemplo, para almacenar los parámetros utilizados para definir un programa que se procesará.
TABLE es común a todas las tareas de la TJ1-MC__. Esto significa que los valores escritos en TABLE desde una tarea se puede leer en otras.
Los valores de TABLE se pueden escribir y leer con el comando **TABLE**.
La longitud máxima de la matriz es de 64.000 elementos, de **TABLE(0)** a **TABLE(63999)**. La matriz TABLE se inicializa hasta el máximo elemento definido.

Variables globales

Las variables globales, definidas en la memoria VR, son comunes a todas las tareas de la TJ1-MC__. Esto significa que si un programa que se ejecuta en la tarea 2 establece VR(25) en un determinado valor, otro programa que

se ejecute en otra tarea distinta puede leer el mismo valor de VR(25). Esto resulta muy útil para sincronizar dos o más tareas, pero se debe tener cuidado para evitar que varios programas escriban en la misma variable simultáneamente. El controlador tiene 1.024 variables globales, de VR(0) a VR(1023). Las variables se leen y escriben con el comando **VR**.

A los datos de TABLE y VR se puede acceder desde las distintas tareas en ejecución. Al utilizar las variables de VR o TABLE, asegúrese de utilizar sólo una tarea para escribir en una determinada variable. Esto evitará que dos programas escriban inesperadamente en una misma variable.

Variables locales

Las variables con nombre o las variables locales se pueden declarar en los programas y son locales a la tarea. Esto significa que dos o más programas que se ejecuten en tareas distintas pueden utilizar el mismo nombre de variable, pero sus valores pueden ser distintos. Las variables locales no se pueden leer desde cualquier tarea, excepto en la que se han declarado. Las variables locales siempre se borran cuando se inicia un programa. Las variables locales se pueden borrar utilizando el comando **CLEAR** o **RESET**. Se puede declarar un máximo de 255 variables locales. Sólo los 16 primeros caracteres del nombre son significativos. Las variables locales no definidas devolverán cero. Las variables locales no se pueden declarar en la línea de comandos.

Etiquetas

Los programas BASIC se ejecutan en orden descendente a través de las líneas. Las etiquetas se pueden utilizar para alterar este flujo de ejecución con los comandos BASIC **GOTO** y **GOSUB**. Para definir una etiqueta, debe aparecer como la primera instrucción en una línea y debe terminar en dos puntos (:). Las etiquetas pueden ser cadenas de caracteres de cualquier longitud, pero sólo los 15 primeros caracteres son significativos.

Uso de variables y etiquetas

Cada tarea tiene sus propias etiquetas y variables locales. Por ejemplo, consulte los dos programas mostrados a continuación:

```
inicio:
    FOR a = 1 to 100
        MOVE(a)
        WAIT IDLE
    NEXT a
GOTO inicio
```

```
inicio:
    a=0
    REPEAT
        a = a + 1
        PRINT a
    UNTIL a = 300
GOTO inicio
```

Estos dos programas se pueden ejecutar simultáneamente en tareas distintas y disponer de su propia versión de la variable **a** y la etiqueta **inicio**. Si necesita contener datos comunes entre dos o más programas, se deben usar variables VR. O, alternativamente, si se debe contener una gran cantidad de datos, se puede utilizar la memoria TABLE.
Para que un programa sea más legible al utilizar una variable VR global, se pueden adoptar dos planteamientos. El primero consiste en utilizar una variable local con nombre como una constante en la variable VR.
No obstante, la variable constante local se debe declarar en cada programa mediante la variable VR global. Con este planteamiento, el ejemplo siguiente muestra cómo usar VR(3) para contener un parámetro longitud común a varios programas:

```
inicio:
    GOSUB inicial
    VR(longitud) = x
    ...
    ...

Inicial:
    longitud = 3
    RETURN
```

```
inicio:
    GOSUB inicial
    MOVE(VR(longitud))
    PRINT(VR(longitud))
    ...

Inicial:
    longitud = 3
    RETURN
```

El otro planteamiento es incluso más legible y utiliza el comando **GLOBAL** para declarar el nombre como una referencia a una de las variables VR globales. Posteriormente, el nombre se puede utilizar desde el programa que contiene la definición **GLOBAL** y todos los demás programas. Tenga en cuenta que el programa que contiene la definición **GLOBAL** se debe ejecutar antes de que el nombre se use en los demás programas. La mejor práctica es definir nombres globales en el programa de inicio. Con este planteamiento, el ejemplo anterior cambia a:

```
'La declaración en el programa de inicio
GLOBAL longitud, 3

'En los demás programas ejecutados después del programa
de inicio
inicio:                         inicio:
    longitud = x                    MOVE(longitud)
    ...                             PRINT(longitud)
    ...                             ...
```

2.3.4 Especificaciones matemáticas

Formato de número

La TJ1-MC__ tiene dos formatos principales para los valores numéricos: coma flotante de precisión simple y entero de precisión simple.
El formato de coma flotante de precisión simple es un valor de 32 bits. Tiene un campo de exponente de 8 bits, un bit de signo y campo de fracción de 23 bits con un 1 implícito como el bit 24. Los números de coma flotante tienen un rango válido de $\pm 5{,}9 \times 10^{-39}$ a $\pm 3{,}4 \times 10^{38}$.
Los enteros son fundamentalmente número de coma flotante con un exponente cero. Esto implica que los enteros tienen un ancho de 24 bits. El rango de enteros, por lo tanto, se indica de –16.777.216 a 16.777.215. Los valores numéricos fuera de este rango serán de coma flotante.

Todos los cálculos matemáticos se realizan en formato de coma flotante. Esto implica que en los cálculos de valores mayores los resultados tendrán una precisión limitada. El usuario debe tenerlo en cuenta al desarrollar la aplicación Motion Control.

Formato hexadecimal

La TJ1-MC__ admite la asignación e impresión de valores hexadecimales. Un número hexadecimal se introduce anteponiendo al número el carácter $. El rango válido es de 0x0 a 0xFFFFFF. Ejemplo:

```
>> VR(0)=$FF
>> PRINT VR(0)
255.0000
```

Un valor se puede imprimir en hexadecimal utilizando la función **HEX**. Los valores negativos dan como resultado el valor hexadecimal de complemento a 2 (24 bits). El rango válido va de –8.388.608 a 16.777.215. Ejemplo:

```
>> TABLE(0,-10,65536)
>> PRINT HEX(TABLE(0)),HEX(TABLE(1))
FFFFF6 10000
```

Posicionamiento

Para el posicionamiento, la TJ1-MC__ redondeará hacia arriba si la distancia de borde de encoder fraccionario excede de 0,9. De lo contrario, el valor fraccionario se redondeará hacia abajo. La posición medida interna y la posición solicitada de los ejes, representadas por los parámetros de eje **MPOS** y **DPOS**, tienen contadores de 32 bits.

Comparación de coma flotante

La función de comparación considera una pequeña diferencia entre valores como igual para evitar resultados de comparación inesperados. Por lo tanto, dos valores cualesquiera cuya diferencia sea menor que $1{,}19 \times 10^{-6}$ se consideran iguales.

Prioridad

La prioridad de los operadores se indica a continuación:

1. Menos unario, **NOT**
2. ^
3. / *
4. **MOD**
5. + –
6. = <> > >= <= <
7. **AND OR XOR**
8. Izquierda a derecha

La mejor forma de garantizar la prioridad de los distintos operadores es mediante el uso de paréntesis.

2.4 Ejecución de Motion Control

Cada tarea de la TJ1-MC__ tiene un conjunto de búferes que contiene la información de los comandos Motion Control indicados.

2.4.1 Generador de Motion Control

El generador de Motion Control tiene dos búferes de Motion Control para cada eje. Un búfer denominado **MTYPE**, contiene el movimiento actual, que es el movimiento que se está ejecutando actualmente en el eje. El otro búfer denominado **NTYPE**, contiene el movimiento siguiente, que se ejecuta después de que haya acabado el movimiento real.
Consulte el capítulo 2.8 “Búferes de movimiento” en el manual de referencia de Trajexia para obtener una explicación detallada.

Los programas BASIC están separados del programa generador del perfil de movimiento, que controla los movimientos de los ejes. El generador de perfiles de movimiento tiene funciones separadas para cada eje, por lo que cada uno puede programarse con sus propios parámetros de eje (por ejemplo, velocidad o aceleración) y moverse independiente y simultáneamente o bien se pueden enlazar entre sí con comandos especiales.
Cuando se está procesando un comando de movimiento, el generador de perfiles de movimiento espera hasta que ha terminado el movimiento y se vacía el búfer del eje requerido; a continuación, carga estos búferes con la información del movimiento siguiente.

Si los búferes de tareas están llenos, la ejecución del programa se pone en pausa hasta que los búferes vuelven a estar disponibles. Esto también se aplica a la tarea de línea de comandos y no se pueden indicar comandos durante dicho período. Trajexia Tools se desconectará en un caso de este tipo. El parámetro de tarea **PMOVE** se establecerá en TRUE cuando los búferes de tareas estén llenos y se restablecerán a FALSE cuando los búferes de tareas vuelvan a estar disponibles.

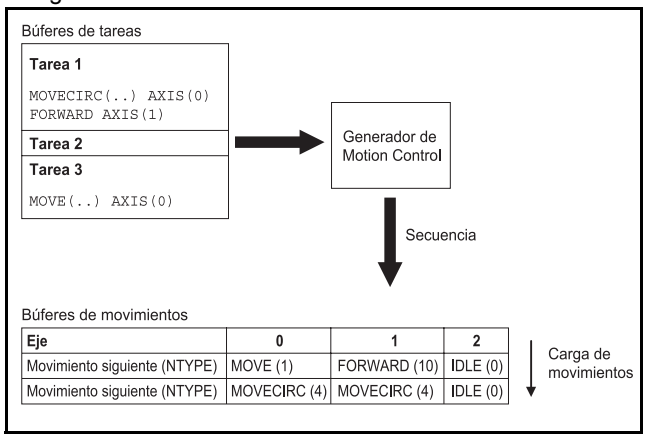
2.4.2 Secuencia

En cada interrupción de ciclo de servo (consulte la sección 2.6.3), el generador de Motion Control examina los búferes **NTYPE** para comprobar si hay alguno disponible. Si lo hay, comprueba si en los búferes de tareas hay un movimiento que espere para cargarse. Si se puede cargar un movimiento, se cargan los datos de los ejes específicos desde los búferes de tareas en los búferes **NTYPE** y los búferes de tareas correspondientes se marcan como inactivos. Este proceso se denomina secuencia.

2.4.3 Carga de movimientos

Una vez completada la secuencia, se comprueban los búferes **MTYPE** por si se puede cargar algún movimiento. Si los búferes **MTYPE** necesarios están disponibles, se carga el movimiento desde los búferes **NTYPE** en los búferes **MTYPE** y los búferes **NTYPE** se marcan como inactivos. Este proceso se denomina carga de movimientos. Si hay un movimiento válido en los búferes **MTYPE**, se procesa. Cuando el movimiento se ha completado, los búferes **MTYPE** se marcan como inactivos.

fig. 2



2.5 Interfaz de línea de comandos

La interfaz de línea de comandos proporciona una interfaz directa para que el usuario ejecute comandos y acceda a parámetros del sistema. Utilice la ventana de terminal en Trajexia Tools cuando la TJ1-MC__ esté conectada. Consulte información detallada en la sección 5.5.4.
La TJ1-MC__ coloca los últimos 10 comandos indicados en la línea de comandos en un búfer. Al pulsar las teclas de cursor arriba y abajo se recorrerá el búfer para volver a ejecutar el comando.

2.6 Programas BASIC

La TJ1-MC__ puede almacenar hasta 14 programas en memoria, siempre que no se exceda la capacidad de la memoria. La TJ1-MC__ admite instrucciones de tratamiento simples para gestionar estos archivos de programa en vez del sistema de archivos DOS de un ordenador.
El paquete de software Trajexia Tools se utiliza para almacenar y cargar programas en un ordenador, y desde él, para archivarlos, imprimirlos y editarlos. También dispone de varias funciones de monitorización y depuración de controlador. Consulte el capítulo 5.

2.6.1 Gestión de programas

Trajexia Tools crea automáticamente un proyecto que contiene los programas que se utilizarán para una aplicación. Los programas del proyecto se mantienen en el controlador y en el ordenador. Siempre que se crea o edita un programa, Trajexia Tools edita ambas copias para disponer siempre de una copia de seguridad precisa fuera del controlador en cualquier momento. Trajexia Tools comprueba que las dos versiones del proyecto sean idénticas mediante una comprobación de redundancia cíclica. Si son distintas, Trajexia Tools permite copiar la versión de TJ1-MC__ en disco o viceversa.
Los programas del ordenador se almacenan en archivos de texto ASCII. Por lo tanto, se pueden imprimir, editar o copiar con un editor de textos.
Los programas de código fuente se mantienen en la TJ1-MC__ en formato token y, como resultado, los tamaños de los programas serán menores en la TJ1-MC__ si se comparan con los mismos programas en el ordenador.

Almacenamiento de programas

Los programas de la TJ1-MC__ se retienen en la memoria RAM alimentada por baterías y se conservan durante la desconexión. Es similar a la memoria VR y TABLE. El contenido de la memoria RAM de programas se conserva cuando la batería está en la TJ1-MC__. Los programas también se conservan al cambiar la batería, si se realiza rápidamente. Para conservar los programas sin la batería durante un período más largo, los programas se deben copiar en la memoria flash del controlador con el comando **EPROM** y leerse durante el encendido, lo que se determina con el parámetro del sistema **POWER_UP**.

Comandos de programa

La TJ1-MC__ tiene una serie de comandos BASIC que permiten la creación, manipulación y eliminación de programas. Trajexia Tools proporciona botones que también realizan estas operaciones, por lo que el uso de estos comandos no se requiere en los programas.

Comando	Función
SELECT	Selecciona un programa para editarlo, eliminarlo, etc.
NEW	Elimina el programa seleccionado, un programa especificado o todos los programas.
DIR	Enumera el directorio de todos los programas.
COPY	Duplica un programa especificado.
RENAME	Cambia el nombre de un programa especificado.
DEL	Elimina el programa seleccionado actual o un programa especificado.
LIST	Enumera el programa seleccionado actual o un programa especificado.

2.6.2 Compilacion de programas

El sistema TJ1-MC__ compila los programas automáticamente cuando es necesario. Normalmente no es necesario forzar al TJ1-MC__ para que compile los programas, sino que éstos se compilan mediante el menú **Program** (Programa) de Trajexia Tools.
El sistema TJ1-MC__ compila automáticamente los programas en las siguientes ocasiones:

- El programa seleccionado se compila antes de ejecutarse si se ha editado.
- El programa seleccionado se compila si se ha editado antes de cambiar del programa seleccionado a otro.
- El programa seleccionado se compila mediante el comando **COMPILE**.

Durante la compilación se comprueban la sintaxis y la estructura del programa. Si la compilación no se realiza correctamente, se ofrecerá un mensaje y no se generará código de programa. En el cuadro del directorio de Trajexia Tools aparecerá una cruz roja.
Los programas no se pueden ejecutar cuando se producen errores de compilación. Los errores se deben corregir y el programa se tiene que volver a compilar.
El proceso de compilación también incluye las siguientes acciones:

- Quitar comentarios.
- Compilar los números al formato de procesador interno.
- Convertir las expresiones al formato RPN (Reverse Polish Notation) para su ejecución.
- Calcular previamente las ubicaciones de variables.
- Calcular e integrar destinos de estructura de lazo.

Como el proceso de compilación requiere memoria libre, se pueden producir errores de compilación inesperados cuando la cantidad de memoria libre no sea suficiente.

2.6.3 Ejecución del programa

La temporización de la ejecución de las diferentes tareas y la actualización de las E/S de la TJ1-MC__ giran alrededor del tiempo de ciclo de servo del sistema. El tiempo de ciclo de servo se determina mediante el parámetro del sistema **SERVO_PERIOD**. La TJ1-MC__ tendrá un tiempo de ciclo de servo de 0,5, 1,0 ó 2,0 ms.

Actualización de E/S

El estado de E/S de la TJ1-MC__ se actualiza al comienzo de cada ciclo de servo.

- El estado capturado de las entradas digitales se transfiere a la variable de entrada del sistema **IN**. Tenga en cuenta que se trata del estado capturado en el ciclo de servo anterior.
- Se actualizan las salidas analógicas de las referencias de velocidad.
- Las salidas digitales se actualizan según el estado de la variable de salida del sistema **OP**.
- Se captura el estado de las entradas digitales.

Tenga en cuenta que no se lleva a cabo ningún procesamiento automático de las señales de E/S, excepto para el registro. Esto implica que todas las acciones se deben programar en los programas BASIC.

Comandos relevantes

Trajexia Tools proporciona varias formas de ejecutar, poner en pausa y parar los programas con los botones del panel de control y las ventanas de edición. En la línea de comandos se pueden indicar los siguientes comandos para controlar la ejecución.

Comando	Función
RUN	Ejecuta el programa seleccionado actual o un programa especificado, opcionalmente en un número de tarea especificado.
STOP	Para el programa seleccionado actual o un programa especificado.
HALT	Para todos los programas del sistema.
PROCESS	Muestra todas las tareas en ejecución.

El usuario puede asignar explícitamente la prioridad de tarea en la que se espera que se ejecute el programa BASIC. Cuando un programa de usuario se ejecuta sin una asignación de tarea explícita, se le asigna la prioridad de tarea más alta disponible.

Configuración de programas para ejecutarse al inicio

Los programas se pueden configurar para ejecutarse automáticamente a distintas prioridades cuando se conecta la alimentación. Si es necesario, el ordenador se puede dejar conectado como una interfaz de operador o se puede quitar y los programas se ejecutan de forma independiente.
Los programas se configuran en Trajexia Tools para ejecutarse al inicio mediante la selección **Set Power Up Mode...** (Configurar modo de encendido) en el menú **Program** (Programa). Esta operación establece el programa que se ejecuta automáticamente y su prioridad. Esto también se puede llevar a cabo mediante el comando BASIC **RUNTYPE**. El estado actual se puede ver con el comando **DIR**.
Para obtener más información sobre el control de programas, la multitarea y los tiempos de ciclo, consulte las secciones 2.2 y 2.3 del manual de referencia de hardware de Trajexia.

3 Comandos BASIC

3.1 Categorías

En esta sección se enumeran todos los comandos BASIC divididos en categorías. Las categorías son:

- Comandos de eje.
- Parámetros de eje.
- Comandos y parámetros de comunicaciones.
- Constantes.
- Comandos, funciones y parámetros de E/S.
- Funciones y operaciones matemáticas.
- Comandos de programa.
- Comandos de control de programa.
- Parámetros y modificadores de entrada.
- Comandos y funciones del sistema.
- Parámetros del sistema.
- Comandos y parámetros de tareas.

Las listas sólo son guías de referencia rápida. En la próxima sección se ofrece una descripción completa de los comandos en orden alfabético.

3.1.1 Comandos de eje

Nombre	Descripción
ACC	Cambia los parámetros **ACCEL** y **DECEL** al mismo tiempo.
ADD_DAC	Suma el valor de **DAC** de un eje a la salida analógica del eje base.
ADDAX	Define un vínculo a un eje superpuesto. Todos los movimientos de posición solicitada para el eje superpuesto se agregarán a los movimientos que se están ejecutando actualmente.
B_SPLINE	Expande el perfil almacenado en la memoria **TABLE** con la función matemática de b-spline.
BASE	Se utiliza para establecer el eje base al que se aplican los comandos y parámetros.
CAM	Mueve un eje según los valores de un perfil de movimiento almacenados en la matriz de variables TABLE.
CAMBOX	Mueve un eje según los valores de un perfil de movimiento almacenados en la matriz de variables TABLE. El movimiento está vinculado a un movimiento medido de otro eje para formar un engranaje electrónico de software variable continuamente.
CANCEL	Cancela el movimiento en un eje.
CONNECT	Conecta la posición solicitada de un eje a los movimientos medidos del eje especificado para el **eje_impulsor** con el fin de producir un engranaje electrónico.
DATUM	Realiza una de las 7 secuencias de búsqueda de origen para situar un eje en una posición absoluta o restablecer un error de Motion Control.
DEFPOS	Define la posición actual como una nueva posición absoluta.
DISABLE_GROUP	Agrupa los ejes entre sí para desactivar errores.
DRIVE_ALARM	Monitoriza la alarma actual.
DRIVE_CLEAR	Borra el estado de alarma del servodriver.
DRIVE_READ	Lee el parámetro especificado del servodriver.
DRIVE_RESET	Realiza un reset del servodriver.
DRIVE_WRITE	Escribe un valor específico en el parámetro especificado del servodriver.
ENCODER_READ	Lee un parámetro del encoder absoluto EnDat.
ENCODER_WRITE	Escribe en un parámetro del encoder absoluto EnDat.
FORWARD	Mueve un eje continuamente hacia delante a la velocidad definida en el parámetro **SPEED**.
HW_PSWITCH	Activa y desactiva el interruptor hardware en la salida 0 de la TJ1-FL02 cuando se alcanzan las posiciones predefinidas.
MECHATROLINK	Inicializa el bus MECHATROLINK-II y efectúa varias operaciones en las estaciones MECHATROLINK-II conectadas al bus.

Nombre	Descripción
MHELICAL	Interpola 3 ejes ortogonales en un movimiento helicoidal.
MOVE	Mueve uno o varios ejes a la velocidad, aceleración y deceleración solicitadas hasta la posición especificada como incremento desde la posición actual.
MOVEABS	Mueve uno o varios ejes a la velocidad, aceleración y deceleración solicitadas hasta la posición especificada como posición absoluta.
MOVECIRC	Interpola 2 ejes ortogonales en un arco circular.
MOVELINK	Crea un movimiento lineal en el eje base vinculado a un engranaje por software hasta la posición medida de un eje de vínculo.
MOVEMODIFY	Cambia la posición final absoluta de un eje que esta ejecutando un movimiento lineal (**MOVE** o **MOVEABS**).
RAPIDSTOP	Cancela el movimiento actual en todos los ejes.
REGIST	Captura una posición de eje cuando se detecta una entrada de registro o se detecta la marca Z del encoder.
REVERSE	Mueve un eje continuamente hacia atrás a la velocidad definida en el parámetro **SPEED**.
STEP_RATIO	Establece la relación para la salida de un eje paso a paso.

3.1.2 Parámetros de eje

Nombre	Descripción
ACCEL	Contiene la velocidad de aceleración de eje.
ADDAX_AXIS	Contiene el número del eje al que está vinculado actualmente el eje base mediante **ADDAX**.
ATYPE	Contiene el tipo de eje.
AXIS_DISPLAY	Selecciona la información que se representa mediante los LEDs de la cubierta frontal de la TJ1-FL02.
AXIS_ENABLE	Activa y desactiva un eje concreto independientemente de los otros eje.
AXISSTATUS	Contiene el estado de eje.

Nombre	Descripción
CLOSE_WIN	Define el fin del periodo en el que se espera una marca de registro.
CLUTCH_RATE	Define el cambio en la relación de conexión al utilizar el comando **CONNECT**.
CREEP	Contiene la velocidad a marcha lenta.
D_GAIN	Contiene la ganancia de control derivativo.
DAC_SCALE	Establece la escala y polaridad aplicadas a los valores de **DAC**.
DATUM_IN	Contiene el número de entrada que se utilizará como la entrada de origen.
DECEL	Contiene la velocidad de deceleración de eje.
DEMAND_EDGES	Contiene el valor actual del parámetro de eje **DPOS** en flancos de encoder.
DPOS	Contiene la posición solicitada que se ha generado mediante los comandos de movimiento.
DRIVE_CONTROL	Selecciona los datos que se monitorizarán con **DRIVE_MONITOR** para los ejes conectados mediante el bus MECHATROLINK-II. Para los ejes conectados mediante TJ1-FL02, **DRIVE_CONTROL** establece las salidas de la TJ1-FL02.
DRIVE_INPUTS	Contiene los datos de E/S del driver conectado al bus MECHATROLINK-II. Los datos se actualizan cada ciclo de servo.
DRIVE_MONITOR	Monitoriza los datos de E/S del servodriver conectado al bus MECHATROLINK-II. Los datos se actualizan cada ciclo de servo.
DRIVE_STATUS	Contiene el estado actual del servodriver.
ENCODER	Contiene una copia sin procesar del registro de hardware del encoder.
ENCODER_BITS	Define el número de bits para el encoder absoluto conectado a la TJ1-FL02.
ENCODER_CONTROL	Controla el modo operativo del encoder absoluto EnDat.
ENCODER_ID	Devuelve el valor ID del encoder absoluto conectado a la TJ1-FL02.
ENCODER_RATIO	Define el valor de escala para los contajes de encoder entrantes.
ENCODER_STATUS	Devuelve el estado del encoder absoluto Tamagawa.

Nombre	Descripción
ENCODER_TURNS	Devuelve el contaje de multigiro del encoder absoluto.
ENDMOVE	Contiene la posición del final del movimiento actual.
ERRORMASK	Contiene el valor de máscara que determina para que estado de eje se produce un **MOTION_ERROR**.
FAST_JOG	Contiene el número de entrada que se utilizará como la entrada de jog rápido.
FASTDEC	Define la velocidad de deceleración de rampa a cero cuando se alcanza un final de carrera o posición de eje.
FE	Contiene el error de seguimiento.
FE_LATCH	Contiene el valor FE que ha provocado que el eje coloque el controlador en el estado **MOTION_ERROR**.
FE_LIMIT	Contiene el error de seguimiento permitido máximo.
FE_LIMIT_MODE	Define cómo FE influye en el estado **MOTION_ERROR**.
FE_RANGE	Contiene el límite de rango de advertencia de error de seguimiento.
FHOLD_IN	Contiene el número de entrada que se utilizará como la entrada de retención de realimentación.
FHSPEED	Contiene la velocidad de retención de realimentación.
FS_LIMIT	Contiene la posición absoluta del límite de software de marcha directa.
FWD_IN	Contiene el número de entrada que se utilizará como una entrada de límite de marcha directa.
FWD_JOG	Contiene el número de entrada que se utilizará como una entrada de marcha directa jog.
I_GAIN	Contiene la ganancia de control integral.
INVERT_STEP	Cambia la lógica del circuito de salida paso a paso.
JOGSPEED	Permite ajustar la velocidad jog.
LINKAX	Contiene el número del eje de vínculo durante cualquier movimiento vinculado.
MARK	Detecta el evento de registro principal en una entrada de registro.
MARKB	Detecta el evento de registro secundario en una entrada de registro.

Nombre	Descripción
MERGE	Es un interruptor software que se puede utilizar para activar o desactivar la combinación de movimientos consecutivos.
MPOS	Es la posición del eje tal como lo mide el encoder.
MSPEED	Representa el cambio en la posición medida en el último período de servo.
MTYPE	Contiene el tipo de movimiento que se está ejecutando actualmente.
NTYPE	Contiene el tipo del movimiento en el búfer de movimiento siguiente.
OFFPOS	Contiene un offset que se aplicará a la posición solicitada sin que se vea afectado el movimiento de ninguna otra forma.
OPEN_WIN	Define el comienzo del periodo en el que se espera una marca de registro.
OUTLIMIT	Contiene el límite que restringe la salida de referencia de velocidad desde la TJ1-MC__.
OV_GAIN	Contiene la ganancia de control de velocidad de salida.
P_GAIN	Contiene la ganancia de control proporcional.
REG_POS	Contiene la posición en la que se ha producido un evento de registro.
REG_POSB	Contiene la posición en la que se ha producido el evento de registro secundario.
REMAIN	Es la distancia restante hasta el final del movimiento actual.
REMOTE_ERROR	Devuelve el número de errores en la conexión MECHATROLINK-II del servodriver.
REP_DIST	Contiene o establece la distancia de repetición.
REP_OPTION	Controla la aplicación del parámetro de eje REP_DIST.
REV_IN	Contiene el número de entrada que se utilizará como una entrada de límite de marcha inversa.
REV_JOG	Contiene el número de entrada que se utilizará como una entrada de marcha inversa jog.
RS_LIMIT	Contiene la posición absoluta del límite de software de marcha inversa.
S_REF	Contiene el valor de referencia de velocidad que se aplica cuando el eje está en lazo abierto.

Nombre	Descripción
S_REF_OUT	Contiene el valor de referencia de velocidad aplicado al servodriver para lazo abierto y cerrado.
SERVO	Determina si el eje funciona bajo control de servo o de lazo abierto.
SPEED	Contiene la velocidad solicitada en unidades/s.
SRAMP	Contiene el factor de curva S.
T_REF	Contiene el valor de referencia de par que se aplica al servomotor.
TRANS_DPOS	Contiene la posición solicitada de eje en la salida de la transformación de trama.
UNITS	Contiene el factor de conversión de unidad.
VERIFY	Selecciona distintos modos de operación en un eje de salida paso a paso.
VFF_GAIN	Contiene la ganancia de control de realimentación positiva de velocidad.
VP_SPEED	Contiene la velocidad de perfil de velocidad.

3.1.3 Comandos y parámetros de comunicaciones

Nombre	Descripción
FINS_COMMS	Envía los comandos FINS de lectura de memoria y de escritura de memoria a una unidad de servidor FINS designada.
HLM_COMMAND	Ejecuta un comando Host Link específico en el esclavo.
HLM_READ	Lee datos del esclavo Host Link en la matriz de variables VR o TABLE.
HLM_STATUS	Representa el estado del último comando de maestro Host Link.
HLM_TIMEOUT	Define el tiempo de espera de maestro Host Link.
HLM_WRITE	Escribe datos en el esclavo Host Link desde la matriz de variables VR o TABLE.
HLS_NODE	Define el número de unidad de esclavo para el protocolo de esclavo Host Link.
SETCOM	Establece las comunicaciones serie.

3.1.4 Constantes

Nombre	Descripción
FALSE	Igual al valor numérico 0.
OFF	Igual al valor numérico 0.
ON	Igual al valor numérico 1.
PI	Igual al valor numérico 3,1416.
TRUE	Igual al valor numérico –1.

3.1.5 Comandos, funciones y parámetros de E/S

Nombre	Descripción
GET	Espera a que llegue un solo carácter y asigna el código ASCII del carácter a la variable.
IN	Devuelve el valor de las entradas digitales.
INDEVICE	El parámetro define el dispositivo de entrada predeterminado.
INPUT	Espera que se reciba una cadena y asigna el valor numérico a una variable.
KEY	Devuelve TRUE o FALSE en función de si se recibe un carácter.
LINPUT	Espera una cadena y la coloca en las variables VR.
OP	Establece una o varias salidas o devuelve el estado de las primeras 24 salidas.
OUTDEVICE	Define el dispositivo de salida predeterminado.
PRINT	Envía una serie de caracteres a un puerto serie.
PSWITCH	Activa una salida cuando se alcanza una posición predefinida y la desactiva cuando se alcanza una segunda posición.

3.1.6 Funciones y operandos matemáticos

Nombre	Descripción
+ (SUMA)	Suma dos expresiones.
– (RESTA)	Resta dos expresiones.
*** (MULTIPLICACIÓN)**	Multiplica dos expresiones.
/ (DIVISIÓN)	Divide dos expresiones.
^ (POTENCIA)	Eleva una expresión a la potencia de la otra expresión.
= (ES IGUAL A)	Comprueba si dos expresiones son iguales.
= (ASIGNACIÓN)	Asigna una expresión a una variable.
<> (NO ES IGUAL A)	Comprueba si dos expresiones son diferentes.
> (ES MAYOR QUE)	Comprueba si en dos expresiones, la de la izquierda es mayor que la de la derecha.
>= (ES MAYOR O IGUAL QUE)	Comprueba si en dos expresiones, la de la izquierda es mayor o igual que la de la derecha.
< (ES MENOR QUE)	Comprueba si en dos expresiones, la de la izquierda es menor que la de la derecha.
<= (ES MENOR O IGUAL QUE)	Comprueba si en dos expresiones, la de la izquierda es menor o igual que la de la derecha.
ABS	Devuelve el valor absoluto de una expresión.
ACOS	Devuelve el arco coseno de una expresión.
AND	Realiza una operación AND en los bits correspondientes de las partes enteras de dos expresiones.
ASIN	Devuelve el arco seno de una expresión.
ATAN	Devuelve el arco tangente de una expresión.
ATAN2	Devuelve el arco tangente del número complejo distinto de cero que se compone de dos expresiones.
COS	Devuelve el coseno de una expresión.
EXP	Devuelve el valor exponencial de una expresión.
FRAC	Devuelve la parte fraccionaria de una expresión.
IEEE_IN	Devuelve el número de coma flotante en formato IEEE, representado por 4 bytes.
IEEE_OUT	Devuelve el byte único extraído del número de coma flotante en formato IEEE.
INT	Devuelve la parte entera de una expresión.
LN	Devuelve el logaritmo natural de una expresión.
MOD	Devuelve el resto de dos expresiones.
NOT	Realiza una operación NOT en los bits correspondientes de la parte entera de la expresión.
OR	Realiza una operación OR entre los bits correspondientes de las partes enteras de dos expresiones.
SGN	Devuelve el signo de una expresión.
SIN	Devuelve el seno de una expresión.
SQR	Devuelve la raíz cuadrada de una expresión.
TAN	Devuelve la tangente de una expresión.
XOR	Realiza una función XOR entre los bits correspondientes de las partes enteras de dos expresiones.

3.1.7 Comandos de programa

Nombre	Descripción
' (CAMPO DE COMENTARIOS)	Permite que no se ejecute una línea.
: (SEPARADOR DE INSTRUCCIONES)	Permite más instrucciones en una línea.
AUTORUN	Inicia todos los programas que se han configurado para ejecutarse al inicio.
COMPILE	Compila el programa actual.
COPY	Copia un programa existente del controlador Motion en un nuevo programa.

Nombre	Descripción
DEL	Elimina un programa del controlador Motion.
DIR	Muestra una lista de los programas del controlador Motion, su tamaño y su RUNTYPE en la salida estándar.
EDIT	Permite modificar un programa utilizando un terminal VT100.
EPROM	Almacena un programa en la memoria flash.
LIST	Imprime el programa en la salida estándar.
NEW	Elimina todas las líneas del programa del controlador Motion.
PROCESS	Devuelve el estado de ejecución y el número de cada tarea actual.
RENAME	Cambia el nombre de un programa del controlador Motion.
RUN	Ejecuta un programa.
RUNTYPE	Determina si un programa se ejecuta al inicio y en qué tarea se ejecutará.
SELECT	Especifica el programa actual.
STEPLINE	Ejecuta una sola línea de un programa.
STOP	Detiene la ejecución del programa.
TROFF	Suspende un seguimiento en la línea actual y reanuda la ejecución normal del programa.
TRON	Crea un punto de ruptura en un programa.

3.1.8 Comandos de control de programa

Nombre	Descripción
FOR..TO..STEP..NEXT	Lazo que permite que un segmento de programa se repita con una variable incrementada o decrementada.
GOSUB..RETURN	Salta a una subrutina en la línea situada justo después de la etiqueta. La ejecución del programa vuelve a la instrucción siguiente después de indicar “RETURN” en página 132.
GOTO	Salta a la línea que contiene la etiqueta.
IF..THEN..ELSE..ENDIF	Controla el flujo del programa según los resultados de la condición.
ON.. GOSUB o **ON.. GOTO**	Permite un salto condicional a una de varias etiquetas.
REPEAT..UNTIL	Lazo que permite que el segmento de programa se repita hasta que la condición sea “TRUE” en página 146.
WHILE..WEND	Lazo que permite que el segmento de programa se repita hasta que la condición sea **FALSE**.

3.1.9 Parámetros y modificadores de slot

Nombre	Descripción
COMMSTYPE	Contiene el tipo de unidad conectada a un slot del controlador.
FPGA_VERSION	Devuelve la versión FPGA de la unidad con **número_de_unidad** en un sistema de controlador.
SLOT	Es un modificador que especifica el número de Slot de la unidad.

3.1.10 Comandos y funciones del sistema

Nombre	Descripción
$ (ENTRADA HEXADECIMAL)	Asigna un número hexadecimal a una variable.
AXIS	Define el eje para un comando, lectura de parámetro de eje o asignación a un determinado eje.
BASICERROR	Se utiliza para ejecutar una rutina específica cuando se produce un error en un comando BASIC.
CLEAR	Borra todas las variables globales y locales de la tarea actual.
CLEAR_BIT	Borra el bit especificado de la variable VR especificada.
CLEAR_PARAMS	Borra todos los parámetros y variables almacenados en la EPROM flash a sus valores predeterminados.

Nombre	Descripción
CONSTANT	Declara una constante para usarla en un programa BASIC.
DATE$	Imprime la fecha actual como una cadena.
DAY$	Imprime el día actual como una cadena.
DEVICENET	Configura la TJ1-DRT (unidad esclava DeviceNet) para el intercambio de datos o devuelve el estado de intercambio de la TJ1-DRT.
ETHERNET	Lee y establece varios parámetros del puerto Ethernet de la TJ1-MC__.
EX	Efectúa un reset del controlador.
FLAG	Establece y lee un banco de 32 bits.
FLAGS	Lee y establece los indicadores especificados con FLAGS como un bloque.
FREE	Devuelve la cantidad de memoria disponible.
GLOBAL	Declara una referencia a una de las variables VR.
HALT	Para la ejecución de todos los programas que se están ejecutando actualmente.
INITIALISE	Establece todos los ejes y parámetros a sus valores predeterminados.
INVERT_IN	Invierte los canales de entrada 0 – 31 en el software.
INVERTER_COMMAND	Lee las E/S y borra la alarma del variador de frecuencia.
INVERTER_READ	Lee el parámetro, la alarma, la velocidad y la referencia de par del variador de frecuencia.
INVERTER_WRITE	Escribe en el parámetro, la velocidad y la referencia de par del variador de frecuencia.
LIST_GLOBAL	Muestra todas las variables GLOBAL y CONSTANT.
LOCK	Impide que los programas se vean o se modifiquen.
PROFIBUS	Configura la TJ1-PRT (unidad esclava PROFIBUS-DP) para intercambiar datos de E/S con el maestro y devuelve el estado de la TJ1-PRT.
READ_BIT	Devuelve el valor del bit especificado de la variable VR especificada.

Nombre	Descripción
RESET	Restablece todas las variables locales de una tarea.
SCOPE	Los programas del sistema almacenan automáticamente hasta 4 parámetros por cada período de muestreo en la matriz de variables TABLE.
SET_BIT	Establece el bit especificado de la variable VR especificada en uno.
TABLE	Escribe y lee datos de la matriz de variables TABLE.
TABLEVALUES	Devuelve la lista de valores de la memoria TABLE.
TIME$	Imprime la hora actual como una cadena.
TRIGGER	Inicia un comando **SCOPE** establecido anteriormente.
VR	Escribe y lee datos de las variables globales (VR).
VRSTRING	Combina los valores de la memoria VR para que se puedan imprimir como una cadena.
WA	Retiene la ejecución del programa el número de milisegundos especificado.
WAIT IDLE	Suspende la ejecución del programa hasta que el eje base ha terminado de ejecutar su movimiento actual y cualquier movimiento del búfer.
WAIT LOADED	Suspende la ejecución del programa hasta que el eje base no tiene movimientos en búfer distintos del movimiento que se está ejecutando actualmente.
WAIT UNTIL	Evalúa repetidamente la condición hasta que es **TRUE**.

3.1.11 Parámetros del sistema

Nombre	Descripción
AIN	Contiene el valor del canal analógico.
AOUT	Contiene el valor del canal analógico.
BATTERY_LOW	Devuelve el estado actual del estado de la batería.
CHECKSUM	Contiene el codigo checksum de los programas en la RAM.
COMMSERROR	Contiene todos los errores de comunicaciones que se han producido desde la última vez que se inicializó.

Nombre	Descripción
CONTROL	Contiene el tipo de la TJ1-MC__ en el sistema.
D_ZONE_MAX	Controla la salida DAC en combinación con el valor de error de seguimiento.
D_ZONE_MIN	Controla la salida DAC en combinación con el valor de error de seguimiento.
DATE	Establece o devuelve la fecha actual que contiene el reloj de tiempo de real.
DAY	Configura o devuelve el día actual.
DISPLAY	Determina los canales de E/S que se mostrarán en los LEDs del panel frontal.
ERROR_AXIS	Contiene el número del eje que ha provocado el error de Motion.
FRAME	Especifica la trama operativa de las transformaciones de trama.
LAST_AXIS	Contiene el número del último eje que ha procesado el sistema.
MOTION_ERROR	Contiene un indicador de error para los errores de Motion de eje.
NAIO	Devuelve el número de canales analógicos conectados al bus MECHATROLINK-II.
NEG_OFFSET	Aplica un offset negativo a la señal DAC desde el lazo de servo.
NIO	Contiene el número de entradas y salidas conectadas al sistema.
POWER_UP	Determina si los programas se deben leer desde la EPROM flash en el encendido o se debe realizar un reset.
POS_OFFSET	Aplica un offset positivo a la señal DAC desde el lazo de servo.
SCOPE_POS	Contiene la posición TABLE actual en la que el comando SCOPE almacena actualmente su primer parámetro.
SERVO_PERIOD	Establece el período de ciclo de servo de la TJ1-MC__.
SYSTEM_ERROR	Contiene los errores de sistema desde la última inicialización.
TIME	Devuelve la hora actual que contiene el reloj de tiempo de real.
TSIZE	Devuelve el tamaño de la tabla definida actualmente.
VERSION	Devuelve el número de versión del firmware del controlador.
WDOG	El interruptor software que activa los servodrivers.

3.1.12 Comandos y parámetros de tareas

Nombre	Descripción
ERROR_LINE	Contiene el número de la línea que ha provocado el último error del programa BASIC.
PMOVE	Contiene el estado actual de los búferes de tarea.
PROC	Permite que se acceda a un parámetro de proceso desde un determinado proceso.
PROC_STATUS	Devuelve el estado del proceso especificado.
PROCNUMBER	Contiene el número de la tarea en la que se está ejecutando el programa seleccionado actualmente.
RUN_ERROR	Contiene el número del último error BASIC que se ha producido en la tarea especificada.
TICKS	Contiene el contaje actual de los impulsos del reloj de tareas.

3.2 Todos los comandos BASIC

3.2.1 + (suma)

Tipo	Función matemática
Sintaxis	**expresión1 + expresión2**
Descripción	El operador **+** suma dos expresiones.
Argumentos	• **expresión1** Cualquier expresión BASIC válida. • **expresión2** Cualquier expresión BASIC válida.
Ejemplo	**resultado = 4 + 3** Asigna el valor 7 a la variable **resultado**.
Consulte también	N/D

3.2.2 – (resta)

Tipo	Función matemática
Sintaxis	**expresión1 – expresión2**
Descripción	El operador **–** resta **expresión2** a **expresión1**.
Argumentos	• **expresión1** Cualquier expresión BASIC válida. • **expresión2** Cualquier expresión BASIC válida.
Ejemplo	**resultado = 10 – 2** Asigna el valor 8 a la variable **resultado**.
Consulte también	N/D

3.2.3 * (multiplicación)

Tipo	Función matemática
Sintaxis	**expresión1 * expresión2**
Descripción	El operador ***** multiplica dos expresiones.
Argumentos	• **expresión1** Cualquier expresión BASIC válida. • **expresión2** Cualquier expresión BASIC válida.
Ejemplo	**resultado = 3 * 7** Asigna el valor 21 a la variable **resultado**.
Consulte también	N/D

3.2.4 / (división)

Tipo	Función matemática
Sintaxis	**expresión1/expresión2**
Descripción	El operador/divide **expresión1** entre **expresión2**.
Argumentos	• **expresión1** Cualquier expresión BASIC válida. • **expresión2** Cualquier expresión BASIC válida.
Ejemplo	**resultado = 11/4** Asigna el valor 2,75 a la variable **resultado**.
Consulte también	N/D

3.2.5 ^ (potencia)

Tipo	Función matemática
Sintaxis	**expresión_1 ^ expresión_2**
Descripción	El operador de potencia **^** eleva **expresión_1** a la potencia de **expresión_2**. Esta operación utiliza algoritmos de coma flotante y puede ofrecer pequeñas desviaciones para los cálculos de enteros.
Argumentos	• **expresión_1** Una expresión BASIC. • **expresión_2** Una expresión BASIC.
Ejemplo	**resultado = 2^5** Esto asigna el valor 32 a la variable **resultado**.
Consulte también	N/D

3.2.6 = (es igual a)

Tipo	Función matemática
Sintaxis	**expresión1 = expresión2**
Descripción	El operador **=** devuelve **TRUE** si **expresión1** es igual a **expresión2**, de lo contrario devuelve **FALSE**.
Argumentos	• **expresión1** Cualquier expresión BASIC válida. • **expresión2** Cualquier expresión BASIC válida.
Ejemplo	**IF a = 10 THEN GOTO etiqueta1** Si la variable **a** contiene un valor igual a 10, la ejecución del programa continúa en la etiqueta **etiqueta1**. De lo contrario, la ejecución del programa continúa con la instrucción siguiente.
Consulte también	N/D

3.2.7 = (asignación)

Tipo	Función matemática
Sintaxis	**variable = expresión**
Descripción	El operador **=** asigna el valor de la expresión a la variable.
Argumentos	• **variable** Un nombre de variable. • **expresión** Cualquier expresión BASIC válida.
Ejemplo	**var = 18** Asigna el valor 18 a la variable **var**.
Consulte también	N/D

3.2.8 <> (no es igual a)

Tipo	Función matemática
Sintaxis	**expresión1 <> expresión2**
Descripción	El operador **<>** devuelve **TRUE** si **expresión1** no es igual a **expresión2**, de lo contrario devuelve **FALSE**.
Argumentos	• **expresión1** Cualquier expresión BASIC válida. • **expresión2** Cualquier expresión BASIC válida.
Ejemplo	**IF a <> 10 THEN GOTO etiqueta1** Si la variable **a** contiene un valor distinto de 10, la ejecución del programa continúa en la etiqueta **etiqueta1**. De lo contrario, la ejecución del programa continúa con la instrucción siguiente.
Consulte también	N/D

3.2.9 > (es mayor que)

Tipo	Función matemática
Sintaxis	**expresión1 > expresión2**
Descripción	El operador **>** devuelve **TRUE** si **expresión1** es mayor que **expresión2**, de lo contrario devuelve **FALSE**.
Argumentos	• **expresión1** Cualquier expresión BASIC válida. • **expresión2** Cualquier expresión BASIC válida.
Ejemplo	**IF a > 10 THEN GOTO etiqueta1** Si la variable **a** contiene un valor mayor que 10, la ejecución del programa continúa en la etiqueta **etiqueta1**. De lo contrario, la ejecución del programa continúa con la instrucción siguiente.
Consulte también	N/D

3.2.10 >= (es mayor o igual que)

Tipo	Función matemática
Sintaxis	**expresión1 >= expresión2**
Descripción	El operador **>=** devuelve **TRUE** si **expresión1** es mayor o igual que **expresión2**, de lo contrario devuelve **FALSE**.
Argumentos	• **expresión1** Cualquier expresión BASIC válida. • **expresión2** Cualquier expresión BASIC válida.
Ejemplo	**IF a >= 10 THEN GOTO etiqueta1** Si la variable **a** contiene un valor mayor o igual que 10, la ejecución del programa continúa en la etiqueta **etiqueta1**. De lo contrario, la ejecución del programa continúa con la instrucción siguiente.
Consulte también	N/D

3.2.11 < (es menor que)

Tipo	Función matemática
Sintaxis	**expresión1 < expresión2**
Descripción	El operador **<** devuelve **TRUE** si **expresión1** es menor que **expresión2**, de lo contrario devuelve **FALSE**.
Argumentos	• **expresión1** Cualquier expresión BASIC válida. • **expresión2** Cualquier expresión BASIC válida.
Ejemplo	**IF a < 10 THEN GOTO etiqueta1** Si la variable **a** contiene un valor menor que 10, la ejecución del programa continúa en la etiqueta **etiqueta1**. De lo contrario, la ejecución del programa continúa con la instrucción siguiente.
Consulte también	N/D

3.2.12 <= (es menor o igual que)

Tipo	Función matemática
Sintaxis	**expresión1 <= expresión2**
Descripción	El operador **<=** devuelve **TRUE** si **expresión1** es menor o igual a **expresión2**, de lo contrario devuelve **FALSE**.
Argumentos	• **expresión1** Cualquier expresión BASIC válida. • **expresión2** Cualquier expresión BASIC válida.
Ejemplo	**IF a <= 10 THEN GOTO etiqueta1** Si la variable **a** contiene un valor menor o igual que 10, la ejecución del programa continúa en la etiqueta **etiqueta1**. De lo contrario, la ejecución del programa continúa con la instrucción siguiente.
Consulte también	N/D

3.2.13 $ (entrada hexadecimal)

Tipo	Comando de sistema
Sintaxis	**$número_hexadecimal**
Descripción	El comando **$** convierte al número que le sigue en un número hexadecimal.
Argumentos	• **número_hexadecimal** Un número hexadecimal (que consta de los caracteres 0 – 9 y A – F). **número_hexadecimal** va de 0 a FFFFFF.
Ejemplo	**>>TABLE(0,$F,$ABCD)** **>>print TABLE(0),TABLE(1)** **15.0000 43981.0000**
Consulte también	**HEX (PRINT)**

3.2.14 ' (campo de comentarios)

Tipo	Comando de programa
Sintaxis	**'**
Descripción	**'** marca todo lo que le sigue en una línea como comentarios y no como código de programa. El comentario no se ejecuta cuando se ejecuta el programa. Puede utilizar **'** al comienzo de una línea o después de una instrucción válida.
Argumentos	N/D
Ejemplo	**' Esta línea no se imprime** **PRINT "Inicio"**
Consulte también	N/D

3.2.15 : (separador de instrucciones)

Tipo	Comando de programa
Sintaxis	**:**
Descripción	El separador de instrucciones **:** separa varias instrucciones BASIC en una línea. Se puede utilizar en la línea de comandos y en programas.
Argumentos	N/D
Ejemplo	**PRINT "ESTA LÍNEA": GET bajo : PRINT "REALIZA TRES ACCIONES"**
Consulte también	N/D

3.2.16

Tipo	Carácter especial
Sintaxis	**#**
Descripción	El símbolo **#** se utiliza para especificar un canal de comunicaciones que se utilizará para los comandos de entrada/salida serie. Nota: los canales de comunicaciones mayores que 3 sólo se utilizarán cuando se ejecute el software Trajexia Tools.
Argumentos	N/D
Ejemplo	**PRINT #1,"RS232"** **PRINT #2,"Puerto 2"**
Ejemplo	**IF KEY #1 THEN GET #1,k** Comprobar teclado en el puerto RS232
Consulte también	N/D

3.2.17 ABS

Tipo	Función matemática
Sintaxis	**ABS(expresión)**
Descripción	La función **ABS** devuelve el valor absoluto de una expresión.
Argumentos	• **expresión** Cualquier expresión BASIC válida.
Ejemplo	**IF ABS(A) > 100 THEN PRINT "A está fuera del rango –100 ... 100"**
Consulte también	N/D

3.2.18 ACC

Tipo	Comando de eje
Sintaxis	**ACC(velocidad)**
Descripción	Establece la aceleración y la deceleración al mismo tiempo. Este comando proporciona un método rápido para configurar **ACCEL** y **DECEL**. Se recomienda establecer las velocidades de aceleración y deceleración con los parámetros de eje **ACCEL** y **DECEL**.
Argumentos	• **velocidad** La velocidad de aceleración en unidades/s^2. Puede definir las unidades con el parámetro de eje **UNITS**.
Ejemplo	**ACC(100)** Establece **ACCEL** y **DECEL** en 100 unidades/s^2.
Consulte también	**ACCEL**, **DECEL**, **UNITS**

3.2.19 ACCEL

Tipo	Parámetro de eje
Sintaxis	**ACCEL = expresión**
Descripción	El parámetro de eje **ACCEL** contiene la velocidad de aceleración de eje. La velocidad se establece en unidades/s^2. El parámetro puede tener cualquier valor positivo, incluido el cero.
Argumentos	N/D
Ejemplo	**BASE(0)** **ACCEL = 100 ' Establecer la velocidad de aceleración** **PRINT "Velocidad de aceleración: ";ACCEL;" mm/s/s"** **ACCEL AXIS(2) = 100 ' Establece la velocidad de aceleración para el eje (2)**
Consulte también	**ACCEL**, **DECEL**, **UNITS**

3.2.20 ACOS

Tipo	Función matemática
Sintaxis	**ACOS(expresión)**
Descripción	La función **ACOS** devuelve el arco coseno de la expresión. El valor de la expresión debe estar entre –1 y 1. El resultado en radianes está entre 0 y PI. Los valores de entrada que estén fuera del rango devolverán 0.
Argumentos	• **expresión** Cualquier expresión BASIC válida.
Ejemplo	**>> PRINT ACOS(-1)** **3.1416**
Consulte también	N/D

3.2.21 ADD_DAC

Tipo: Comando de eje

Sintaxis: **ADD_DAC(eje)**

Descripción: El comando **ADD_DAC** puede proporcionar control de realimentación dual, lo que permite el uso de un encoder secundario en el servoeje. El comando permite la salida de 2 lazos de servo que se sumen para determinar la referencia de velocidad para el servodriver.
Este comando normalmente se utiliza en aplicaciones como alimentadores por rodillo donde se necesita un encoder secundario para compensar el deslizamiento.
Para utilizar **ADD_DAC** es necesario que los dos ejes con realimentación física se vinculen a un eje común en el que se ejecuten los movimientos requeridos. Normalmente esto se consigue mediante la ejecución de los movimientos en uno de los dos ejes y con **ADDAX** o **CONNECT** para producir una posición solicitada coincidente (**DPOS**) para ambos ejes. Se tienen que definir ganancias de lazo de servo para ambos ejes. Las salidas de lazo de servo se suman a la salida de referencia de velocidad del servoeje. Utilice **ADD_DAC(-1)** para cancelar el vínculo.
ADD_DAC funciona en el eje base predeterminado (establecido con **BASE**) a menos que se utilice **AXIS** para especificar un eje base temporal.
Nota:
1. Tenga en cuenta que las ganancias de lazo de control para ambos ejes se tienen que determinar con cuidado. Como se utilizan encoders con resoluciones distintas, las ganancias no son idénticas.
2. Establezca el parámetro **OUTLIMIT** al mismo valor para ambos ejes vinculados.

Argumentos:
- **eje**
  El eje desde el que se sumará la salida de referencia de velocidad al eje base. Establezca el argumento en –1 para cancelar el vínculo y volver a la operación normal.

Ejemplo:

```
BASE(0)
OUTLIMIT AXIS(1) = 15000
ADD_DAC(1) AXIS(0)
ADDAX(0) AXIS(1)
WDOG = ON
SERVO AXIS(0) = ON
SERVO AXIS(1) = ON
' Ejecutar movimientos en el eje 0
```

Este ejemplo muestra el control del eje 0 del servodriver con control de realimentación dual mediante los ejes 0 y 1.

Ejemplo:

```
BASE(0)
OUTLIMIT AXIS(1) = 15000
ADD_DAC(1) AXIS(0)
ADDAX(0) AXIS(1)
WDOG = ON
SERVO = OFF
S_REF = 0
BASE(1)
SERVO = ON
' Ejecutar movimientos en el eje 1
```

Este ejemplo muestra el control del eje 0 del servodriver utilizando sólo realimentación de encoder en el eje 1.

Consulte también: **AXIS**, **ADDAX**, **OUTLIMIT**

3.2.22 ADDAX

Tipo: Comando de eje

Sintaxis: **ADDAX(eje)**

Descripción: El comando **ADDAX** toma los cambios de posición solicitada del eje superpuesto según lo especificado por el argumento de eje y los suma a cualquier movimiento que se ejecute en el eje en el que se emita el comando.
Después de que se haya emitido el comando **ADDAX**, permanecerá el vínculo entre los dos ejes hasta que se rompa. Utilice **ADDAX(-1)** para cancelar el vínculo de eje. **ADDAX** permite que un eje realice los movimientos especificados para 2 ejes sumados. Se pueden realizar combinaciones de más de dos ejes si también se aplica **ADDAX** al eje superpuesto.
ADDAX funciona en el eje base predeterminado (establecido con **BASE**) a menos que se utilice **AXIS** para especificar un eje base temporal.

Argumentos:
- **eje**
  El eje que se definirá como un eje superpuesto. Establezca el argumento en –1 para cancelar el vínculo y volver a la operación normal.

Ejemplo

```
FORWARD ' Definir movimiento continuo
ADDAX(2) ' Agregar eje 2 para corrección
REPEAT
   GOSUB obteneroffset ' Obtener el offset que se aplicará
   MOVE(offset) AXIS(2)
UNTIL IN(2) = ON ' Hasta que se realice la corrección
```

Las piezas se colocan en una cinta en movimiento continuo y se cogen más allá de la línea. Un sistema de detección ofrece una indicación de si una pieza está delante o detrás de su posición nominal, así como la distancia.
En este ejemplo, se supone que el eje 0 es el eje base, ejecuta un movimiento directo continuo y se utiliza un movimiento superpuesto en el eje 2 para aplicar offsets según el offset calculado en una subrutina.

Consulte también **AXIS**, **OUTLIMIT**

ADVERTENCIA
Tenga en cuenta que el uso de varios comandos **ADDAX** en un sistema puede crear un lazo peligroso cuando, por ejemplo, un eje está vinculado a otro y viceversa. Esto puede provocar inestabilidad en el sistema.

3.2.23 ADDAX_AXIS

Tipo Parámetro de eje (sólo lectura)

Sintaxis **ADDAX_AXIS**

Descripción El parámetro de eje **ADDAX_AXIS** devuelve el número del eje al que está vinculado actualmente el eje base mediante **ADDAX**.

Argumentos N/D

Ejemplo

```
>> BASE(0)
>> ADDAX(2)
>> PRINT ADDAX_AXIS
2.0000
```

Consulte también **ADDAX**, **AXIS**

3.2.24 AIN

Tipo Parámetro de sistema

Sintaxis **AIN(canal_analógico)**

Descripción Los canales de entrada analógicos de +/–10 V se proporcionan al conectar módulos JEPMC-AN2900 o modulos analógicos en la cabecera GRT1-ML2 en el bus MECHATROLINK-II.
Nota: Se comprueba el valor de entrada analógica para garantizar que es superior a cero aunque siempre debe ser positivo. Esto se realiza para permitir que cualquier ruido en la señal entrante podría dar un valor negativo y provocar un error porque una velocidad negativa no es válida para ningún tipo de movimiento, excepto **FORWARD** o **REVERSE**.

Argumentos **canal_analógico**.
Número de canal de entrada analógica 0.31

Ejemplo

```
MOVE(-5000)
 REPEAT
  a = AIN(1)
  IF a<0 THEN a = 0
  SPEED = a * 0.25
 UNTIL MTYPE = 0
```

La velocidad de una línea de producción se rige por la velocidad con la que se introduce el material en ella. La alimentación de material se realiza mediante una disposición de lazo abierto que se ajusta con un dispositivo de detección de altura por ultrasonidos. La salida del sensor ultrasónico está en el rango de 0 V a 4 V, donde la salida está a 4 V cuando el lazo está en su máxima longitud.

Consulte también N/D

3.2.25 AND

Tipo Operación matemática

Sintaxis **expresión1 AND expresión2**

Descripción El operador **AND** realiza la función **AND** lógica en los bits correspondientes de las partes enteras de dos expresiones BASIC válidas.
La función **AND** lógica entre dos bits se define del siguiente modo:
0 AND 0 = 0
0 AND 1 = 0
1 AND 0 = 0
1 AND 1 = 1

Argumentos
- **expresión1**
  Cualquier expresión BASIC válida.
- **expresión2**
  Cualquier expresión BASIC válida.

Ejemplo **VR(0) = 10 AND (2.1 * 9)**
Los paréntesis se evalúan en primer lugar, pero sólo la parte entera del resultado, 18, se utiliza para la operación **AND**. Por lo tanto, esta expresión equivale a lo siguiente:
VR(0) = 10 AND 18
El operador **AND** es un operador de bit y, por lo tanto, la acción binaria es la siguiente:
01010 AND 10010 = 00010
Por lo tanto, **VR(0)** contendrá el valor 2.

Ejemplo **IF MPOS AXIS(0) > 0 AND MPOS AXIS(1) > 0 THEN GOTO ciclo1**

Consulte también N/D

3.2.26 AOUT

Tipo Parámetro de sistema

Sintaxis **AOUT(canal_analógico)**

Descripción Este comando establece el valor de salida de los canales analógicos de +/–10 V que se proporcionan al conectar módulos JEPMC-AN2910 o modulos analogicos conectados a la cabecera GRT1-ML2 en el bus MECHATROLINK-II. El rango del valor seleccionado es [–32.000, 32.000] para el rango de tensión [–10 V, 10 V].

Argumentos
- **canal_analógico**
  Número de canal de salida analógica 0.31.

Ejemplo No hay ejemplo.

Consulte también N/D

3.2.27 ASIN

Tipo Función matemática

Sintaxis **ASIN(expresión)**

Descripción La función **ASIN** devuelve el arco seno del argumento. El argumento debe tener un valor entre –1 y 1. El resultado en radianes está entre –PI/2 y PI/2. Los valores de entrada que estén fuera de este rango devuelven 0.

Argumentos
- **expresión**
  Cualquier expresión BASIC válida.

Ejemplo **>> PRINT ASIN(-1)**
–1.5708

Consulte también N/D

3.2.28 ATAN

Tipo Función matemática

Sintaxis **ATAN(expresión)**

Descripción La función **ATAN** devuelve el arco tangente del argumento. En **expresión** puede haber cualquier valor. El resultado en radianes está entre –PI/2 y PI/2.

Argumentos
- **expresión**
  Cualquier expresión BASIC válida.

Ejemplo **>> PRINT ATAN(1)**
0.7854

Consulte también N/D

3.2.29 ATAN2

Tipo Función matemática

Sintaxis **ATAN2(expresión1,expresión2)**

Descripción La función **ATAN2** devuelve el arco tangente del número complejo distinto de cero **(expresión2, expresión1)**, que es el equivalente al ángulo entre un punto con la coordenada **(expresión1, expresión2)** y el eje x.
Si **expresión2 >= 0**, el resultado es igual al valor de **ATAN(expresión1/expresión2)**.
El resultado en radianes estará entre –PI y PI.

Argumentos
- **expresión1**
  Cualquier expresión BASIC válida.
- **expresión2**
  Cualquier expresión BASIC válida.

Ejemplo
>> PRINT ATAN2(0,1)
0.0000

Consulte también N/D

3.2.30 ATYPE

Tipo Parámetro de eje

Sintaxis **ATYPE = valor**

Descripción El parámetro de eje **ATYPE** establece el tipo del eje. Los valores válidos dependen del módulo TJ1 al que está conectado el servodriver que controla el eje. Consulte la tabla siguiente. Los parámetros **ATYPE** los establece el sistema en el inicio. Para los ejes controlados por los servodrivers conectados al sistema mediante el bus MECHATROLINK-II, el valor ATYPE predeterminado es 41 (velocidad Mechatro). Para los ejes controlados por los servodrivers conectados al sistema mediante la TJ1-FL02, el valor ATYPE predeterminado es 44 (servoeje flexible).

Argumentos N/D

Ejemplo
ATYPE AXIS(1) = 45
Este comando establecerá el eje 1 como el eje de salida del encoder de eje flexible.

Consulte también **AXIS**

Tipo de AXIS	Valor ATYPE	Unidad TJ1 aplicable
Virtual	0	Todas
Posición de Mechatro	40	TJ1-ML__ (unidad maestra MECHATROLINK-II)
Velocidad Mechatro	41	TJ1-ML__
Par de Mechatro	42	TJ1-ML__
Eje flexible salida paso a paso	43	TJ1-FL02
Servoeje flexible	44	TJ1-FL02
Salida de encoder de eje flexible	45	TJ1-FL02
Eje flexible Encoder Tamagawa absoluto	46	TJ1-FL02
Eje flexible encoder EnDat absoluto	47	TJ1-FL02
Eje flexible encoder SSI absoluto	48	TJ1-FL02

3.2.31 AUTORUN

Tipo Comando de programa

Sintaxis **AUTORUN**

Descripción El comando **AUTORUN** inicia todos los programas que se han configurado para ejecutarse al inicio.

Argumentos N/D

Ejemplo No hay ejemplo.

Consulte también **RUNTYPE**

3.2.32 AXIS

Tipo Comando de sistema

Sintaxis **AXIS(número_eje)**

Descripción El modificador **AXIS** define el eje para un solo comando Motion Control o una sola lectura/escritura de parámetro de eje en un determinado eje. **AXIS** sólo es efectivo para el comando o línea de programa en que está programado. Utilice el comando **BASE** para cambiar el eje base para todas las líneas de comandos.

Argumentos
- **número_eje**
  Cualquier expresión BASIC válida que especifique el número de eje.

Ejemplo **BASE(0)**
PRINT VP_SPEED AXIS(2)

Ejemplo **MOVE(300) AXIS(0)**

Ejemplo **REPDIST AXIS(1) = 100**

Consulte también **BASE**

3.2.33 AXIS_DISPLAY

Tipo Parámetro de eje

Sintaxis **AXIS_DISPLAY = valor**

Descripción El parámetro de eje **AXIS_DISPLAY** permite que diferentes datos se muestren mediante los LEDs de la cubierta frontal de la TJ1-FL02. Los LEDs afectados por este ajuste de parámetro son dos LEDs amarillos que muestran el estado de eje. El valor predeterminado de este parámetro al inicio para todos los ejes es de 0. Los valores válidos se muestran en la tabla siguiente.

Argumentos N/D

Ejemplo **AXIS_DISPLAY AXIS(2) = 2**
Este comando mostrará el estado de OUT 0 y OUT 1 asignados al eje 2.

Consulte también N/D

AXIS_DISPLAY valor	0	1	2	3
A0	REG 0	AUX IN	OUT 0	ENCODER A
A1	REG 1	ENCODER Z	OUT 1	ENCODER B
B0	REG 0	AUX IN	OUT 0	ENCODER A
B1	REG 1	ENCODER Z	OUT 1	ENCODER B

3.2.34 AXIS_ENABLE

Tipo Parámetro de eje

Sintaxis **AXIS_ENABLE = ON/OFF**

Descripción El parámetro de eje **AXIS_ENABLE** se utiliza para activar o desactivar un eje determinado independientemente de los demás. Este parámetro se puede activar o desactivar para cada eje individualmente. El valor predeterminado al inicio es activado en todos los ejes. El eje se activará si AXIS_ENABLE para dicho eje está activado y también lo está **WDOG**. Para los ejes MECHATROLINK-II, al desactivar AXIS_ENABLE se desactivará la señal de run (Etapa de potencia) del servodriver al motor. Para los ejes de servoeje flexible, al desactivar AXIS_ENABLE se forzará ambas salidas de tensión a 0. Para los ejes de salida paso a paso flexible y de salida de encoder, al desactivar AXIS_ENABLE se bloqueará la generación de impulsos en las salidas.

Argumentos N/D

Ejemplo **AXIS_ENABLE AXIS(3) = OFF**
Este comando desactivará el eje 3 independientemente de los demás ejes del sistema.

Consulte también **AXIS**, **DISABLE_GROUP**

3.2.35 AXISSTATUS

Tipo Parámetro de eje (sólo lectura)

Sintaxis **AXISSTATUS**

Descripción El parámetro de eje **AXISSTATUS** contiene el estado del eje. Las definiciones del parámetro de eje **AXISSTATUS** se muestran en la tabla siguiente.
El parámetro **AXISSTATUS** se utiliza para el tratamiento de los errores de Motion Control de la unidad.

Argumentos N/D

Ejemplo **IF (AXISSTATUS AND 16)>0 THEN PRINT "En límite de marcha directa"**

Consulte también **AXIS**, **ERRORMASK**

Número De bit	Descripción	Valor	Carácter (tal como se usa en Trajexia Tools)
0	–	1	-
1	Advertencia de error de seguimiento	2	W
2	Error de comunicaciones de servodriver	4	A
3	Alarma de servodriver	8	M
4	Límite de marcha directa	16	F
5	Límite de marcha inversa	32	R
6	Buscando origen	64	D
7	Entrada de retención de realimentación	128	H
8	Límite de error de seguimiento	256	E
9	Límite de software de marcha directa	512	X
10	Límite de software de marcha inversa	1.024	Y
11	Cancelación de movimiento	2.048	C
12	Sobrevelocidad de salida de encoder	4.096	O

3.2.36 B_SPLINE

Tipo: Comando de eje

Sintaxis: **B_SPLINE(tipo, entrada_datos, entrada_número, salida_datos, nº expansión)**

Descripción: Expande un perfil existente almacenado en el área TABLA mediante la función matemática de b-spline por un factor de expansión configurable a otra área de TABLE.
Se utiliza de forma idónea donde el perfil **CAM** de origen es demasiado tosco y se tiene que extrapolar en un mayor número de puntos.

Argumentos:
- **tipo**
  Reservado para expansión futura. Defínalo siempre en 1.
- **entrada_datos**
  Ubicación en TABLE donde está almacenado el perfil de origen.
- **entrada_número**
  Número de puntos del perfil de origen.
- **salida_datos**
  Ubicación en TABLE donde se almacenará el perfil expandido.
- **relación_expansión**
  La relación de expansión, es decir, si el perfil de origen es 100 puntos y **relación_expansión** se define en 10, el perfil resultante será 1.000 puntos (100 * 10).

Ejemplo: No hay ejemplo.

Consulte también: N/D

3.2.37 BASE

Tipo: Comando de eje

Sintaxis:
BASE
BASE(eje_1 [, eje_2 [, eje_3 [, eje_4 [, eje_...]]]])
BA
BA(eje_1 [, eje_2 [, eje_3 [, eje_4 [, eje_...]]]])

Descripción El comando **BASE** se utiliza para establecer el eje base predeterminado o para establecer un grupo de secuencias de eje especificado. Todos los comandos de Motion Control y parámetros de eje posteriores se aplicarán al eje base o el grupo de eje especificado a menos que el comando **AXIS** se utilice para especificar un eje base temporal. El eje base es efectivo hasta que se cambia de nuevo con **BASE**.
Cada proceso BASIC puede tener su propio grupo de ejes y cada programa puede establecer su propio grupo de ejes independientemente. Utilice el modificador **PROC** para acceder al parámetro para una determinada tarea.
La agrupación de orden **BASE** se puede definir mediante la asignación explícita del orden de los ejes. Este orden se utiliza para interpolación en movimientos lineales y circulares de varios ejes. El valor predeterminado del grupo de ejes base es (0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15) al inicio o cuando un programa se empieza a ejecutar en una tarea. El comando **BASE** sin argumentos devuelve la agrupación de orden base actual.

Nota: si el comando **BASE** no especifica todos los ejes, el comando **BASE** "rellenará" los valores restantes automáticamente. En primer lugar, rellenará los restantes ejes por encima del último valor declarado y, después, rellenará los ejes restantes en secuencia.
Por lo tanto, **BASE(2,6,10) establecerá la matriz interna de 16 ejes a: 2,6,10,11,12,13,14,15,0,1,3,4,5,7,8,9.**

Argumentos El comando puede aceptar hasta 16 argumentos.

- **eje_i**
  El número del eje definido como el eje base y cualquier eje posterior del orden de grupo para los movimientos multieje.

Ejemplo
```
BASE(1)
UNITS = 2000 ' Definir el factor de conversión para el eje 1
SPEED = 100 ' Definir la velocidad para el eje 1
ACCEL = 5000 ' Establecer la velocidad de aceleración para el eje 1
BASE(2)
UNITS = 2000 ' Definir el factor de conversión para el eje 2
SPEED = 125 ' Definir la velocidad para el eje 2
ACCEL = 10000 ' Establecer la velocidad de aceleración para el eje 2
```
Se puede programar cada eje con su propia velocidad, aceleración y otros parámetros.

Ejemplo
```
BASE(0)
MOVE(100,-23.1,1250)
```
En este ejemplo, los ejes 0, 1 y 2 se moverán a las posiciones especificadas a la velocidad y aceleración definidas para el eje 0. **BASE(0)** define el eje base en el eje 0, que determina los tres ejes que utiliza **MOVE** así como la velocidad y la aceleración.

Ejemplo
```
>> BASE(0,2,1)
```
En la línea de comandos, el orden de grupo base se puede mostrar si se escribe **BASE**.

Ejemplo
```
>> RUN "PROGRAM",3
>> BASE PROC(3)(0,2,1)
```
Utilice el modificador **PROC** para mostrar el orden de grupo base de una determinada tarea.

Ejemplo
```
>> BASE(2)
>> PRINT BASE
2.0000
```
Al imprimir **BASE** se devolverá el eje base seleccionado actualmente.

Consulte también **AXIS**

3.2.38 BASICERROR

Tipo Comando de sistema

Sintaxis **BASICERROR**

Descripción El comando **BASICERROR** se puede utilizar para ejecutar una rutina cuando se produce un error en tiempo de ejecución en un programa. **BASICERROR** sólo se puede usar como parte de un comando **ON ... GOSUB** u **ON ... GOTO**. Este comando es necesario que se ejecute una vez en el programa BASIC. Si se utilizan varios comandos, sólo es efectivo el último ejecutado.

Argumentos N/D

Ejemplo

```
ON BASICERROR GOTO rutina_error
...
num_error = 1
STOP
rutina_error:
IF num_error = 0 THEN
  PRINT "El error ";RUN_ERROR[0];
  PRINT " se ha producido en la línea ";ERROR_LINE[0]
ENDIF
STOP
```

Si se produce un error en un comando BASIC en este ejemplo, se ejecutará la rutina de error.
La instrucción **IF** aparece para evitar que el programa vaya a la rutina de error cuando se pare normalmente.

Consulte también: **ERROR_LINE**, **ON**, **RUN_ERROR**.

3.2.39 BATTERY_LOW

Tipo: Parámetro de sistema (sólo lectura)

Sintaxis: **BATTERY_LOW**

Descripción: Este parámetro devuelve el estado actual del estado de la batería.
Si **BATTERY_LOW = ON**, se tiene que cambiar la batería.
Si **BATTERY_LOW = OFF**, el estado de la batería es correcto.

Argumentos: N/D

Ejemplo: No hay ejemplo.

Consulte también: N/D

3.2.40 BREAK_RESET

Tipo: Comando de sistema

Sintaxis: **BREAK_RESET "nombre_programa"**

Descripción: Lo utiliza Trajexia Tools para quitar todos los puntos de ruptura del programa especificado.

Argumentos:
- **nombre_programa**
  El nombre del programa del que desea quitar todos los puntos de ruptura.

Ejemplo: **BREAK_RESET "pruebasimple"**
Quitará todos los puntos de ruptura del programa **pruebasimple**.

Consulte también: N/D

3.2.41 CAM

Tipo: Comando de eje

Sintaxis: **CAM(punto_inicial, punto_final, multiplicador_tabla, distancia)**

Descripción: El comando **CAM** se utiliza para generar el movimiento de un eje siguiendo un perfil de posición que está almacenado en una matriz de variables TABLE. Los valores TABLE son posiciones absolutas relativas al punto inicial y se especifican en flancos de encoder. La matriz TABLE se especifica con el comando **TABLE**.
El movimiento se puede definir con cualquier número de puntos de 2 a 64.000. La TJ1-MC__ se mueve continuamente entre los valores de TABLE para permitir que una serie de puntos definan un perfil suavizado. Se pueden ejecutar dos o más comandos **CAM** simultáneamente con valores idénticos o superpuestos en la matriz TABLE. El perfil TABLE se recorre una vez.
CAM requiere que el elemento inicial de la matriz TABLE tenga valor cero.
El argumento de distancia junto con los parámetros **SPEED** y **ACCEL** determinan la velocidad de movimiento por la matriz TABLE. Tenga en cuenta que para seguir el perfil **CAM** exactamente, el parámetro **ACCEL** del eje debe ser al menos 1.000 veces mayor que el parámetro **SPEED**.
CAM funciona en el eje base predeterminado (establecido con **BASE**) a menos que se utilice **AXIS** para especificar un eje base temporal.

Argumentos

- **punto_inicial**
  La dirección del primer elemento de la matriz TABLE que se utilizará. Al poder especificar el punto inicial, la matriz TABLE puede contener varios perfiles u otra información.
- **punto_final**
  La dirección del elemento final de la matriz TABLE.
- **multiplicador_tabla**
  El valor de multiplicador de tabla se utiliza para ajustar a escala los valores almacenados en TABLE. Como los valores de tabla se especifican en flancos de encoder, utilice este argumento para establecer los valores, por ejemplo, para el factor de conversión de unidad (establecido por el parámetro **UNITS**).
- **distancia**
  Factor indicado en unidades del usuario que controla la velocidad del movimiento a través de la tabla. El tiempo que se tarda en ejecutar el comando **CAM** depende de la velocidad de eje actual y esta distancia. Por ejemplo, supongamos que el sistema está programado en mm y que la velocidad está definida en 10 mm/s y la aceleración es suficientemente alta. Si se especifica una distancia de 100 mm, **CAM** tardará 10 segundos en ejecutarse.
  El parámetro **SPEED** del eje base permite la modificación de la velocidad del movimiento cuando se utiliza el movimiento **CAM**.

Nota: cuando se está ejecutando el comando **CAM**, el parámetro **ENDMOVE** está configurado en el final del movimiento anterior.

Ejemplo

Supongamos que se necesita un movimiento para seguir la ecuación de posición t(x) = x * 25 + 10.000 (1 – cos(x)). Aquí, x se expresa en grados. Este ejemplo corresponde a una matriz TABLE que proporciona una oscilación simple superpuesta con una velocidad constante. Para cargar la matriz TABLE y recorrerla continuamente, se debe utilizar el código siguiente:

```
GOSUB camtable
lazo:
  CAM(1,19,1,200)
  GOTO lazo
```

La subrutina **camtable** carga los datos de la tabla siguiente en la matriz TABLE.

Posición de TABLE	Grados	Valor
1	0	0
2	20	1.103
3	40	3.340
4	60	6.500
5	80	10.263
6	100	14.236
7	120	18.000
8	140	21.160
9	160	23.396
10	180	24.500
11	200	24.396
12	220	23.160
13	240	21.000
14	260	18.236
15	280	15.263
16	300	12.500
17	320	10.340
18	340	9.103
19	360	9.000

Consulte también

ACCEL, **AXIS**, **CAMBOX**, **SPEED**, **TABLE**.

3.2.42 CAMBOX

Tipo: Comando de eje

Sintaxis: **CAMBOX(punto_inicial, punto_final, multiplicador_tabla, distancia_vínculo, eje_vínculo [, opción_vínculo [, posición_vínculo]])**

Descripción: El comando **CAMBOX** se utiliza para generar movimiento de un eje siguiendo un perfil de posición de la matriz de variables TABLE. El movimiento está vinculado a un movimiento medido de otro eje para formar un engranaje electrónico de software variable continuamente. Los valores TABLE corresponden a una posición absoluta relativa al punto inicial y se especifican en flancos del encoder.
La matriz TABLE se especifica con el comando **TABLE**. El movimiento se puede definir con cualquier número de puntos de 2 a 64.000. Al poder especificar el punto inicial, la matriz TABLE puede contener varios perfiles u otra información. La TJ1-MC__ se mueve continuamente entre los valores de TABLE para permitir que una serie de puntos definan un perfil suavizado.
Se pueden ejecutar dos o más comandos **CAMBOX** simultáneamente con valores idénticos o superpuestos en la matriz TABLE.
El comando **CAMBOX** requiere que el elemento inicial de TABLE tenga valor cero. Tenga en cuenta también que el comando **CAMBOX** permite recorrer TABLE hacia atrás así como hacia delante según la dirección del eje maestro.
El argumento **opción_vínculo** se puede utilizar para especificar diferentes opciones para iniciar el comando y especificar un comando **CAM** continuo. Por ejemplo, si **opción_vínculo** se define en 4, el comando **CAMBOX** actúa como un elemento **CAM** “físico”.
CAMBOX funciona en el eje base predeterminado (establecido con **BASE**) a menos que se utilice **AXIS** para especificar un eje base temporal.
Nota: mientras se está ejecutando **CAMBOX**, el parámetro **ENDMOVE** se configurará en el final del movimiento anterior. El parámetro de eje **REMAIN** contendrá el resto de la distancia en el eje de vínculo.

Argumentos:
- **punto_inicial**
  La dirección del primer elemento de la matriz TABLE que se utilizará.
- **punto_final**
  La dirección del elemento final de la matriz TABLE.
- **multiplicador_tabla**
  El valor de multiplicador de tabla se utiliza para ajustar a escala los valores almacenados en TABLE. Como los valores de TABLE se especifican en flancos de encoder, utilice este argumento para establecer los valores, por ejemplo, para el factor de conversión de unidad (establecido por el parámetro UNITS).
- **distancia_vínculo**
  La distancia en unidades de usuario que el eje de vínculo se debe mover para completar el movimiento de salida especificado. La distancia de vínculo se debe especificar como una distancia positiva.
- **eje_vínculo**
  El eje al que se vinculará.
  opción_vínculo
  Consulte la siguiente tabla.
- **posición_vínculo**
  La posición absoluta donde se iniciará el comando **CAMBOX** cuando **opción_vínculo** se establece en 2.

Ejemplo: No hay ejemplo.

Consulte también:
- **AXIS**, **CAM**, **REP_OPTION**, **TABLE**

Valor de opción_vínculo	Descripción
1	El vínculo comienza cuando se produce un registro en el eje de vínculo.
2	El vínculo comienza en una posición absoluta del eje de vínculo (consulte **posición_vínculo**).
4	**CAMBOX** se repite automáticamente y de forma bidireccional. Esta opción se cancela si se establece el bit 1 del parámetro **REP_OPTION** (**REP_OPTION = REP_OPTION OR 2**).
5	Combinación de las opciones 1 y 4.
6	Combinación de las opciones 2 y 4.

3.2.43 CANCEL

Tipo	Comando de eje
Sintaxis	**CANCEL[(1)]** **CA[(1)]**
Descripción	El comando **CANCEL** cancela el movimiento actual en un eje. Se decelerarán los movimientos con perfil de velocidad (**FORWARD**, **REVERSE**, **MOVE**, **MOVEABS**, **MOVECIRC**, **MHELICAL** y **MOVEMODIFY**) a la velocidad de deceleración tal como se define en el parámetro **DECEL** y, a continuación, se paran. Otros movimientos se pararán inmediatamente. El comando **CANCEL** cancela el contenido del búfer de movimientos actual (**MTYPE**). El comando **CANCEL(1)** cancela el contenido del búfer de movimiento siguiente (**NTYPE**) sin que se vea afectado el actual en el búfer **MTYPE**. **CANCEL** funciona en el eje base predeterminado (establecido con **BASE**) a menos que se utilice **AXIS** para especificar un eje base temporal. Nota: • **CANCEL** sólo cancela el movimiento que se ejecuta actualmente. Si hay más movimientos en búfer, se cargarán. • Durante la deceleración del movimiento actual, se omitirán los comandos **CANCEL** adicionales. • **CANCEL(1)** sólo cancela el movimiento que está en búfer actualmente. Cualquier movimiento almacenado en los búferes de tareas que se indique con la variable **PMOVE** se puede cargar en el búfer tan pronto como se cancele el movimiento en el búfer.
Argumentos	N/D
Ejemplo	**FORWARD** **WA(10000)** **CANCEL**
Ejemplo	**MOVE(1000)** **MOVEABS(3000)** **CANCEL ' Cancelar el movimiento a 3000 e ir a 4000 en su lugar.** **MOVEABS(4000)** Tenga en cuenta que el comando **MOVEMODIFY** es una mejor solución para modificar puntos finales de movimientos en este caso.
Consulte también	**AXIS**, **MTYPE**, **NTYPE**, **PMOVE**, **RAPIDSTOP**

3.2.44 CHECKSUM

Tipo	Parámetro de sistema (sólo lectura)
Sintaxis	**CHECKSUM**
Descripción	El parámetro **CHECKSUM** contiene el código de comprobación de los programas en RAM. Al inicio, el código de comprobación se vuelve a calcular y se compara con el valor retenido anteriormente. Si el código de comprobación es incorrecto, el programa no se ejecutará.
Argumentos	N/D
Ejemplo	No hay ejemplo.
Consulte también	N/D

3.2.45 CHR

Tipo	Comando de E/S
Sintaxis	**CHR(x)**
Descripción	El comando **CHR** se utiliza para enviar caracteres ASCII individuales a los que se hace referencia por número. **PRINT CHR(x)**; equivale a **PUT(x)** en otras versiones de BASIC.
Argumentos	• **x** Una expresión BASIC.
Ejemplo	**>>PRINT CHR(65);** **A**
Consulte también	N/D

3.2.46 CLEAR

Tipo	Comando de sistema
Sintaxis	**CLEAR**
Descripción	El comando **CLEAR** restablece todas las variables VR globales a 0. Cuando se utiliza en un programa, también se restablecen las variables locales de la tarea actual a 0.
Argumentos	N/D
Ejemplo	No hay ejemplo.
Consulte también	• **RESET**, **VR**

3.2.47 CLEAR_BIT

Tipo	Comando de sistema
Sintaxis	**CLEAR_BIT(número_bit, número_vr)**
Descripción	El comando **CLEAR_BIT** restablece el bit especificado en la variable VR especificada a 0. Otros bits de la variable mantienen sus valores.
Argumentos	• **número_bit** El número del bit que se restablecerá. Rango: 0 – 23. • **número_vr** El número de la variable VR cuyo bit se restablecerá. Rango: 0 – 1.023.
Ejemplo	No hay ejemplo.
Consulte también	**READ_BIT**, **SET_BIT**, **VR**.

3.2.48 CLEAR_PARAMS

Tipo	Comando de sistema
Sintaxis	**CLEAR_PARAMS**
Descripción	Borra todas las variables y parámetros almacenados en la EPROM flash a sus valores predeterminados. CLEAR_PARAM no se puede realizar si el controlador está bloqueado.
Argumentos	N/D
Ejemplo	No hay ejemplo.
Consulte también	N/D

3.2.49 CLOSE_WIN

Tipo	Parámetro de eje
Sintaxis	**CLOSE_WIN** **CW**
Descripción	El parámetro de eje **CLOSE_WIN** define el final del periodo dentro o fuera del que se espera una marca de registro. El valor se indica en unidades del usuario.
Argumentos	N/D
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**, **OPEN_WIN**, **REGIST**, **UNITS**.

3.2.50 CLUTCH_RATE

Tipo	Parámetro de eje
Sintaxis	**CLUTCH_RATE**
Descripción	El parámetro de eje **CLUTCH_RATE** define el cambio en la relación de conexión al utilizar el comando **CONNECT**. La relación se define como la cantidad de relación por segundo. El valor predeterminado se establece en un valor alto (1.000.000) para garantizar la compatibilidad con la unidades TJ1-MC__ anteriores. Nota: la operación que utiliza **CLUTCH_RATE** no es determinista en la posición. Si es necesario, utilice el comando **MOVELINK** en su lugar para evitar una diferencia de fase innecesaria entre el eje base y el eje vinculado.
Argumentos	N/D
Ejemplo	**CLUTCH_RATE = 4** Este ajuste implica que al indicar **CONNECT(4,1)** se tardará un segundo en alcanzar la conexión completa.

Consulte también **AXIS**, **CONNECT**, **MOVELINK**.

3.2.51 COMMSERROR

Tipo Parámetro de sistema (sólo lectura)

Sintaxis **COMMSERROR**

Descripción El parámetro **COMMSERROR** contiene los errores de comunicaciones que se han producido desde la última vez que se inicializó.
Los bits de **COMMSERROR** se indican en la siguiente tabla.

Argumentos N/D

Ejemplo No hay ejemplo.

Consulte también N/D

Bit	Descripción	Ubicación del error
8	Datos Rx de puerto 1 preparados	Puerto serie 1
9	Desbordamiento de Rx de puerto 1	Puerto serie 1
10	Error de paridad de puerto 1	Puerto serie 1
11	Error de trama Rx de puerto 1	Puerto serie 1
12	Datos Rx de puerto 2 preparados	Puerto serie 2
13	Desbordamiento de Rx de puerto 2	Puerto serie 2
14	Error de paridad de puerto 2	Puerto serie 2
15	Error de trama Rx de puerto 2	Puerto serie 2

3.2.52 COMMSTYPE

Tipo Parámetro de entrada

Sintaxis **COMMSTYPE SLOT(número_unidad)**

Descripción Este parámetro devuelve el tipo de una unidad del controlador.
En la tabla siguiente se enumeran los valores de devolución.

Argumentos
- **número_unidad**
  Los números de unidad van de 0 a 6, siendo 6 la unidad situada inmediatamente a la derecha de TJ1-MC__.

Ejemplo No hay ejemplo.

Consulte también N/D

Valor de devolución	Descripción
0	Unidad no utilizada
31	TJ1-ML__
33	TJ1-FL02
34	TJ1-PRT
35	TJ1-DRT

3.2.53 COMPILE

Tipo Comando de programa

Sintaxis **COMPILE**

Descripción El comando **COMPILE** fuerza la compilación del programa actual. El software del sistema compila automáticamente los programas antes de su ejecución o cuando se selecciona otro programa.

Argumentos N/D

Ejemplo No hay ejemplo.

Consulte también N/D

3.2.54 CONNECT

Tipo: Comando de eje

Sintaxis: **CONNECT(relación, eje_impulsor)**
CO(relación, eje_impulsor)

Descripción: El comando **CONNECT** Conecta la posición solicitada del eje base a los movimientos medidos del eje especificado por **eje_impulsor** con el fin de lograr un engranaje electrónico.
La relación se puede cambiar en cualquier momento si se ejecuta otro comando **CONNECT** en el mismo eje. Para cambiar el eje impulsor, primero se tiene que cancelar el comando **CONNECT**. El comando **CONNECT** con otro eje impulsor se omitirá. El comando **CONNECT** se puede cancelar con un comando **CANCEL** o **RAPIDSTOP**. El parámetro de eje **CLUTCH_RATE** se puede utilizar para establecer una velocidad de cambio de conexión específica.
CONNECT funciona en el eje base predeterminado (establecido con **BASE**) a menos que se utilice **AXIS** para especificar un eje base temporal.

Argumentos:
- **relación**
  La relación de conexión del engranaje. La relación se especifica como la relación en flancos de encoder (no unidades). Contiene el número de flancos que se requiere que el eje base mueva por incremento de flanco del eje impulsor. El valor de relación puede ser positivo o negativo y tiene una resolución de dieciséis bits fraccionarios.
- **eje_impulsor**
  El eje maestro que accionará el eje base.

Ejemplo: En un sistema de alimentación de prensa, se requiere que un rodillo gire a una velocidad que sea una cuarta parte de la velocidad medida desde un encoder montado en la cinta transportadora entrante. El rodillo está cableado en el eje 0. Un canal de entrada monitoriza los impulsos del encoder desde la cinta transportadora y forma el eje 1. Se puede utilizar este código:

```
BASE(1)
SERVO = OFF ' Este eje se utiliza para monitorizar la cinta transportadora
BASE(0)
SERVO = ON
CONNECT(0.25,1)
```

Consulte también: **AXIS**, **CANCEL**, **CLUTCH_RATE**, **CONNECT**, **RAPIDSTOP**.

3.2.55 CONSTANT

Tipo: Comando de sistema

Sintaxis: **CONSTANT "nombre", valor**

Descripción: Declara el nombre como una constante para utilizar desde el programa que contiene la definición **CONSTANT** y todos los demás programas en el proyecto de Trajexia Tools.
Nota: el programa que contiene la definición **CONSTANT** se debe ejecutar antes de que el nombre se use en los demás programas. Además, sólo dicho programa se debe ejecutar en el momento en que se ejecuta CONSTANT, de lo contrario, aparecerá un error de programa y el programa se parará cuando intente ejecutar este comando. Para un inicio rápido, el programa también debe ser el único proceso que se ejecuta al inicio.
Cuando se declara CONSTANT, la declaración permanece activa hasta el próximo reset de TJ1-MC__ al desconectar y conectar la alimentación o mediante la ejecución del comando EX.
Se puede declarar un máximo de 128 instrucciones **CONSTANT**.

Argumentos:
- **nombre**
  Cualquier nombre definido por el usuario que contiene caracteres alfanuméricos en minúsculas, numéricos o subrayados.
- **valor**
  El valor asignado a **nombre**.

Ejemplo:

```
CONSTANT "nak",$15
CONSTANT "boton_inicio",5
IF IN(boton_inicio) = ON THEN OP(led1,ON)
IF key_char = nak THEN GOSUB no_ack_received
```

Consulte también: N/D

3.2.56 CONTROL

Tipo: Parámetro de sistema (sólo lectura)

Sintaxis: **CONTROL**

Descripción: El parámetro **CONTROL** Contiene el tipo de la TJ1-MC__ en el sistema.
El valor de este parámetro del sistema para la TJ1-MC__ es 262.

Argumentos	N/D
Ejemplo	No hay ejemplo.
Consulte también	N/D

3.2.57 COPY

Tipo	Comando de programa
Sintaxis	**COPY nombre_programa nuevo_nombre_programa**
Descripción	El comando **COPY** copia un programa existente del controlador en un nuevo programa con el nombre especificado. El nombre de programa se puede especificar sin comillas. Nota: este comando se ha implementado para un terminal (VT100) offline. En Trajexia Tools, los usuarios pueden seleccionar el comando desde el menú **Program** (Programa).
Argumentos	• **nombre_programa** Nombre del programa que se copiará. • **nuevo_nombre_programa** Nombre que se usará para el nuevo programa.
Ejemplo	**>> COPY "programa" "nuevoprograma"**
Consulte también	**DEL**, **NEW**, **RENAME**.

3.2.58 COS

Tipo	Función matemática
Sintaxis	**COS(expresión)**
Descripción	La función **COS** devuelve el coseno de la expresión. Los valores de entrada se expresan en radianes y pueden tener cualquier valor. El valor de resultado estará en el rango de –1 a 1.
Argumentos	• **expresión** Cualquier expresión BASIC válida.
Ejemplo	**>> PRINT COS(0)** **1.0000**
Consulte también	N/D

3.2.59 CREEP

Tipo	Parámetro de eje
Sintaxis	**CREEP**
Descripción	El parámetro de eje **CREEP** contiene la velocidad a marcha lenta del eje. La velocidad a marcha lenta se utiliza para la parte lenta de una secuencia de búsqueda de origen. **CREEP** puede tener cualquier valor positivo, incluido 0. La velocidad a marcha lenta se introduce en unidades con el factor de conversión de unidades **UNITS**. Por ejemplo, si el factor de conversión de unidades se establece en el número de flancos/pulgada de encoder, la velocidad se establece en pulgadas.
Argumentos	N/D
Ejemplo	**BASE(2)** **CREEP = 10** **SPEED = 500** **DATUM(4)** **CREEP AXIS(1) = 10** **SPEED AXIS(1) = 500** **DATUM(4) AXIS(1)**
Consulte también	**AXIS**, **DATUM**, **UNITS**.

3.2.60 D_GAIN

Tipo	Parámetro de eje
Sintaxis	**D_GAIN**
Descripción	El parámetro de eje **D_GAIN** contiene la ganancia derivatida del eje. La aportación de salida de derivada se calcula multiplicando el cambio en el error de seguimiento por **D_GAIN**. El valor predeterminado es 0. Incorpore la ganancia de derivada a un sistema para producir una respuesta más suave y el uso de una mayor ganancia proporcional. Los valores altos pueden provocar oscilación. Nota: la ganancia de servo sólo se debe cambiar cuando **SERVO** está en OFF.

Argumentos	N/D
Ejemplo	No hay ejemplo.
Consulte también	• **AXIS**, **I_GAIN**, **OV_GAIN**, **P_GAIN**, **VFF_GAIN**.

3.2.61 D_ZONE_MAX

Tipo	Parámetro de sistema
Sintaxis	**D_ZONE_MAX = valor**
Descripción	Funciona en combinación con **D_ZONE_MIN** para fijar la salida DAC en cero cuando se completa el movimiento solicitado y la magnitud del error de seguimiento es menor que el valor de **D_ZONE_MIN**. El lazo de servo se reactivará cuando el error de seguimiento esté por encima del valor **D_ZONE_MAX** o se inicie un nuevo movimiento.
Argumentos	N/D
Ejemplo	**D_ZONE_MIN = 3** **D_ZONE_MAX = 10** Con estos dos parámetros configurados tal como se indica más arriba, la salida DAC se fijará en cero cuando el movimiento esté completo y el error de seguimiento esté por debajo de 3. Cuando se reinicia un movimiento o si el error de seguimiento está por encima de un valor de 10, se reactivará el lazo de servo.
Consulte también	**D_ZONE_MIN**.

3.2.62 D_ZONE_MIN

Tipo	Parámetro de sistema
Sintaxis	**D_ZONE_MIN = valor**
Descripción	Funciona en combinación con **D_ZONE_MAX** para fijar la salida DAC en cero cuando se completa el movimiento solicitado y la magnitud del error de seguimiento es menor que el valor de D_ZONE_MIN. El lazo de servo se reactivará cuando el error de seguimiento esté por encima del valor **D_ZONE_MAX** o se inicie un nuevo movimiento.
Argumentos	N/D
Ejemplo	**D_ZONE_MIN = 3** **D_ZONE_MAX = 10** Con estos dos parámetros configurados tal como se indica más arriba, la salida DAC se fijará en cero cuando el movimiento esté completo y el error de seguimiento esté por debajo de 3. Cuando se reinicia un movimiento o si el error de seguimiento está por encima de un valor de 10, se reactivará el lazo de servo.
Consulte también	**D_ZONE_MAX**.

3.2.63 DAC

Consulte **S_REF**.

3.2.64 DAC_OUT

Consulte **S_REF_OUT**.

3.2.65 DAC_SCALE

Tipo	Parámetro de eje
Sintaxis	**DAC_SCALE**
Descripción	El parámetro tiene dos propósitos: 1. Se establece en el valor 16 en el encendido en los ejes integrados del sistema. Escala los valores aplicados al **DAC** de mayor resolución, por lo que las ganancias necesarias en el eje son similares a las requeridas en otros controladores. 2. **DAC_SCALE** se puede configurar en un valor negativo (–16) para invertir la polaridad de la señal de salida DAC. Cuando el servo está en OFF, la magnitud de **DAC_SCALE** no es importante ya que la tensión aplicada se controla mediante el parámetro **DAC**. No obstante, la polaridad se sigue invirtiendo mediante **DAC_SCALE**.
Argumentos	N/D
Ejemplo	**DAC_SCALE AXIS(3) = –16**
Consulte también	**DAC**, **S_REF**.

3.2.66 DATE

Tipo	Parámetro de sistema
Sintaxis	**DATE**
Descripción	Devuelve o establece la fecha actual que contiene el reloj de tiempo real del Trajexia. El número se puede introducir en formato DD:MM:AA o DD:MM:AAAA.
Argumentos	N/D
Ejemplo	**DATE = 20:10:05** o bien **DATE = 20:10:2005**
Ejemplo	**>>PRINT DATE** **36956** Se imprime el número que representa la fecha actual. Se trata del número de días transcurridos desde el 1 de enero de 1.900, siendo éste el 1.
Consulte también	N/D

3.2.67 DATE$

Tipo	Comando de sistema
Sintaxis	**DATE$**
Descripción	Imprime la fecha actual DD/MM/AA como una cadena en el puerto. Se indica una descripción de año de 2 dígitos.
Argumentos	N/D
Ejemplo	**PRINT #1,DATE$** Por ejemplo, se imprimirá la fecha con el formato: 20/10/05
Consulte también	N/D

3.2.68 DATUM

Tipo	Comando de eje
Sintaxis	**DATUM(secuencia)**
Descripción	El comando **DATUM** realiza una de las 6 búsquedas de origen para situar un eje en una posición absoluta y también para restablecer los errores de seguimiento: **DATUM** utiliza la velocidad a marcha lenta y la velocidad solicitada para la búsqueda de origen. La velocidad a marcha lenta en las secuencias se establece con el parámetro de eje **CREEP** y la velocidad solicitada se establece con el parámetro de eje **SPEED**. El número de entrada del interruptor de referencia, que se utiliza para las secuencias 3 a 6, se establece mediante el parámetro **DATUM_IN**. **DATUM** funciona en el eje base predeterminado (establecido con **BASE**) a menos que se utilice **AXIS** para especificar un eje base temporal. Nota: la entrada de origen establecida con el parámetro **DATUM_IN** es activa por baja, es decir, el interruptor de origen se establece cuando la entrada está desconectada. Las entradas de retención de realimentación, marcha jog inversa, marcha jog directa y límites de marcha directa e inversa también son activas por baja. Las entradas activas por baja se utilizan para permitir el cableado a prueba de fallos.
Argumentos	• **secuencia** Consulte la siguiente tabla.
Ejemplo	No hay ejemplo.
Consulte también	**ACCEL**, **AXIS**, **AXISSTATUS**, **CREEP**, **DATUM_IN**, **DECEL**, **MOTION_ERROR**, **SPEED**.

Valor de secuencia	Descripción
0	El comando **DATUM(0)** borrará el error de Motion Control. La posición medida actualmente se establece como la posición solicitada (esto resulta muy útil en los ejes paso a paso con verificación de posición). DATUM(0) también borra el error de seguimiento que ha excedido la condición FE_LIMIT en el registro **AXISSTATUS** para TODOS los ejes. Establece estos bits en AXISSTATUS a cero: Bit 1 : advertencia de error de seguimiento. Bit 2 : error de comunicaciones de driver remoto. Bit 3 : error de driver remoto. Bit 8 : límite de error de seguimiento excedido. Bit 11 : cancelación de movimiento. Tenga en cuenta que el estado no se puede borrar si sigue estando presente la causa del problema.
1	El eje se mueve a la velocidad de marcha lenta hacia delante hasta que se encuentra el marcador Z. A continuación, la posición solicitada se restablece a cero y la posición medida se corrige para mantener el error de seguimiento.
2	El eje se mueve a la velocidad de marcha lenta hacia atrás hasta que se encuentra el marcador Z. A continuación, la posición solicitada se restablece a cero y la posición medida se corrige para mantener el error de seguimiento.
3	El eje se mueve a la velocidad solicitada hacia delante hasta que se alcanza el interruptor de referencia. A continuación, el eje se mueve hacia atrás a la velocidad de marcha lenta hasta que se restablece el interruptor de referencia. A continuación, la posición solicitada se restablece a cero y la posición medida se corrige para que se mantenga el error de seguimiento.
4	El eje se mueve a la velocidad solicitada hacia atrás hasta que se alcanza el interruptor de referencia. A continuación, el eje se mueve hacia delante a la velocidad de marcha lenta hasta que se restablece el interruptor de referencia. A continuación, la posición solicitada se restablece a cero y la posición medida se corrige para que se mantenga el error de seguimiento.
5	El eje se mueve a la velocidad solicitada hacia delante hasta que se alcanza el interruptor de referencia. A continuación, el eje se va hacia atrás a la velocidad de marcha lenta hasta que se restablece el interruptor de referencia. El eje continúa a la velocidad de marcha lenta hasta que se encuentra el marcador Z del encoder. A continuación, la posición solicitada se restablece a cero y la posición medida se corrige para que se mantenga el error de seguimiento.
6	El eje se mueve a la velocidad solicitada hacia atrás hasta que se alcanza el interruptor de referencia. A continuación, el eje se mueve hacia delante a la velocidad a marcha lenta hasta que se restablece el interruptor de referencia. El eje continúa a la velocidad de marcha lenta hasta que se encuentra el marcador Z del encoder. A continuación, la posición solicitada se restablece a cero y la posición medida se corrige para que se mantenga el error de seguimiento.

3.2.69 DATUM_IN

Tipo: Parámetro de eje

Sintaxis: **DATUM_IN**
DAT_IN

Descripción: El parámetro de eje **DATUM_IN** contiene el número de entrada que se utilizará como la entrada de interruptor de referencia para el comando **DATUM**. El rango de entrada válido va de 0 a 31. Los valores de 0 a 15 representan entradas presentes físicamente del conector de E/S de la TJ1-MC__ y son comunes para todos los ejes. Los valores 16 a 27 representan entradas de software que se pueden utilizar libremente en programas y comandos, como, por ejemplo, IN y OP. También son comunes para todos los ejes. Los valores 28 a 31 se asignan directamente a las entradas de driver presentes en el conector CN1 y son únicos para cada eje. Las entradas de driver que se asignan a las entradas 28 a 31 dependen del ajuste Pn81E del parámetro de servodriver. El ajuste recomendado es Pn81E = 0x4321, con la siguiente asignación:
Nota: la entrada de origen es activa por baja, es decir, el interruptor de origen se establece cuando la entrada está desconectada. Las entradas de retención de realimentación, marcha jog inversa, marcha jog directa y límites de marcha directa e inversa también son activas por baja. Las entradas activas por baja se utilizan para permitir el cableado a prueba de fallos.

Sigma II
- entrada 28: CN1-40
- entrada 29: CN1-41
- entrada 30: CN1-42
- entrada 31: CN1-43

Sigma III
- entrada 28: CN1-13
- entrada 29: CN1-7
- entrada 30: CN1-8
- entrada 31: CN1-9

Junma	• entrada 26: CN1-2 • entrada 27: CN1-1 Para obtener más información sobre el ajuste del parámetro de driver Pn81E, consulte el manual del servodriver. De forma predeterminada, el parámetro se establece en –1 y no hay seleccionada ninguna entrada.
Argumentos	N/D
Ejemplo	**DATUM_IN AXIS(0) = 5**
Consulte también	**AXIS**, **DATUM**.

3.2.70 DAY

Tipo	Parámetro de sistema
Sintaxis	**DAY**
Descripción	Devuelve el día actual como un número de 0 a 6; el domingo es 0. **DAY** se puede establecer mediante asignación.
Argumentos	N/D
Ejemplo	**>>DAY = 3** **>>? DAY** **3.0000**
Consulte también	N/D

3.2.71 DAY$

Tipo	Comando de sistema
Sintaxis	**DAY$**
Descripción	Imprime el día actual como una cadena.
Argumentos	N/D
Ejemplo	**>>DAY = 3** **>>? DAY$** **Miércoles**
Consulte también	N/D

3.2.72 DECEL

Tipo	Parámetro de eje
Sintaxis	**DECEL**
Descripción	El parámetro de eje **DECEL** contiene la velocidad de deceleración de eje. La velocidad se establece en unidades/s^2. El parámetro puede tener cualquier valor positivo, incluido 0.
Argumentos	N/D
Ejemplo	**DECEL = 100 ' Establecer la velocidad de deceleración** **PRINT " La velocidad de deceleración es ";DECEL;" mm/s/s"**
Consulte también	**ACCEL**, **AXIS**, **UNITS**.

3.2.73 DEFPOS

Tipo	Comando de eje
Sintaxis	**DEFPOS(pos_1 [, pos_2 [, pos_3 [, pos_4 [, ...]]]])** **DP(pos_1 [, pos_2 [, pos_3 [, pos_4 [, ...]]])**
Descripción	El comando **DEFPOS** define la posición solicitada actual (**DPOS**) como una nueva posición absoluta. La posición medida (**MPOS**) se cambiará en consecuencia para mantener el error de seguimiento. **DEFPOS** normalmente se utiliza después de una secuencia de búsqueda de origen (consulte el comando DATUM), estableciendo la posición actual en 0. **DEFPOS** se puede utilizar en cualquier momento. Como alternativa, también se puede usar el parámetro de eje **OFFPOS**. Este parámetro se puede utilizar para realizar un ajuste relativo de la posición actual. **DEFPOS** funciona en el eje base predeterminado (establecido con **BASE**) a menos que se utilice **AXIS** para especificar un eje base temporal. Nota: los cambios en la posición de eje mediante DEFPOS u OFFPOS se efectúan en la siguiente actualización de servo. Esto puede provocar problemas potenciales cuando se inicia un movimiento en el mismo período de servo que DEFPOS u OFFPOS. En el siguiente ejemplo se muestra cómo se puede usar el parámetro **OFFPOS** para evitar este problema. Los comandos DEFPOS se convierten internamente a offsets de posición **OFFPOS**, que proporciona una forma simple de evitar el problema mediante programación del siguiente modo: **DEFPOS(100): WAIT UNTIL OFFPOS = 0: MOVEABS(0)**

Argumentos El comando puede aceptar hasta 16 argumentos.

- **pos_i**
  La posición absoluta del eje (base + i) en unidades de usuario. Consulte el comando **BASE** para la agrupación de los ejes.

Ejemplo
BASE(2)
DATUM(5)
BASE(1)
DATUM(4)
WAIT IDLE
DEFPOS(–1.000,–3.500)
La última línea define la posición actual en (–1.000,–3.500) en unidades de usuario. La posición actual se habrá restablecido a (0,0) por los dos comandos **DATUM**.

Consulte también **AXIS**, **DATUM**, **DPOS**, **OFFPOS**, **MPOS**, **UNITS**.

3.2.74 DEL

Tipo Comando de programa

Sintaxis
DEL [nombre_programa]
RM [nombre_programa]

Descripción El comando **DEL** elimina un programa del controlador. Se puede utilizar **DEL** sin un nombre de programa para eliminar el programa seleccionado actualmente (mediante **SELECT**). El nombre de programa también se puede especificar sin comillas. Con **DEL ALL** se eliminan todos los programas. **DEL** también se puede utilizar para eliminar la tabla: **DEL "TABLE"**. El nombre **"TABLE"** debe estar entre comillas.
Nota: este comando se ha implementado para un terminal (VT100) offline. En Trajexia Tools, los usuarios pueden seleccionar el comando desde el menú **Program** (Programa).

Argumentos

- **nombre_programa**
  Nombre del programa que se eliminará.

Ejemplo **>> DEL programa_anterior**

Consulte también **COPY**, **NEW**, **RENAME**, **SELECT**, **TABLE**.

3.2.75 DEMAND_EDGES

Tipo Parámetro de eje (sólo lectura)

Sintaxis **DEMAND_EDGES**

Descripción El parámetro de eje **DEMAND_EDGES** contiene el valor actual del parámetro de eje **DPOS** en unidades de flanco de encoder.

Argumentos N/D

Ejemplo No hay ejemplo.

Consulte también **AXIS**, **DPOS**.

3.2.76 DEVICENET

Tipo Comando de sistema

Sintaxis
DEVICENET(número_unidad, 2, 1, salidas_inicio_VR, número_salidas, entradas_inicio_VR, número_entradas)
DEVICENET(número_unidad, 4, 0)

Descripción La función 2 de **DEVICENET** configura la TJ1-DRT para el intercambio de datos con la unidad maestra DeviceNet y define las áreas de la memoria VR donde se realiza el intercambio de E/S. La función 4 de **DEVICENET** devuelve el estado de intercambio de datos de la TJ1-DRT. Consulte en la tabla la descripción de los bits en el canal de estado de intercambio de datos.

Argumentos

- **número_unidad**
  Especifica el número de unidad de la TJ1-DRT en el sistema Trajexia.
- **salidas_inicio_VR**
  La dirección de inicio de la memoria VR del controlador donde se encuentran los datos de salida del maestro DeviceNet.
- **número_salidas**
  El número de canales de salida desde el maestro DeviceNet en la memoria VR.
- **entradas_inicio_VR**
  La dirección de inicio de la memoria VR del controlador donde se encuentran los datos de entrada para el maestro DeviceNet.
- **número_entradas**
  El número de canales de entrada al maestro DeviceNet en la memoria VR.

Ejemplo **DEVICENET (0,2,1,10,16,150,31)**
En este ejemplo, la TJ1-DRT está configurada para intercambiar datos con el maestro DeviceNet con 16 canales de salida (recibidos del maestro) situados en VR(10) a VR(25) y 31 canales de entrada (enviados al maestro) situados en VR(150) a VR(180).

Consulte también N/D

Bit	Valor	Descripción
0	0	**DEVICENET (número_unidad, 2, ...)** no se ha ejecutado todavía
	1	**DEVICENET (número_unidad, 2, ...)** se ha ejecutado sin errores
1	0	Sin conexión de E/S de DeviceNet
	1	Conexión de E/S de DeviceNet en funcionamiento
2	0	Las variables VR del rango de datos de salida se han actualizado
	1	Las variables VR del rango de datos de salida todavía no se han actualizado
3	0	El tamaño de conexión de E/S de DeviceNet coincide con el comando **DEVICENET (número_unidad, 2,…)**
	1	El tamaño de conexión de E/S de DeviceNet no coincide con el comando **DEVICENET (número_unidad, 2,…)**
4-7	0	Siempre cero
8	0	Alimentación de red correcta
	1	Fallo de alimentación de red
9	0	No se ha producido BUSOFF
	1	Se ha producido BUSOFF
10	0	Sin error de duplicación de dirección de nodo
	1	Error de duplicación de dirección de nodo

3.2.77 DIR

Tipo	Comando de programa
Sintaxis	**DIR** **LS**
Descripción	El comando **DIR** muestra una lista de los programas que contiene el controlador, el tamaño de memoria y el valor de **RUNTYPE**. DIR también muestra el tamaño de memoria disponible, el modo de encendido y el programa seleccionado actual del controlador.
Argumentos	N/D
Ejemplo	No hay ejemplo.
Consulte también	**FREE**, **POWER_UP**, **PROCESS**, **RUNTYPE**, **SELECT**.

3.2.78 DISABLE_GROUP

Tipo	Comando de eje
Sintaxis	**DISABLE_GROUP(-1)** **DISABLE_GROUP(eje_1 [, eje_2 [, ...]])**
Descripción	Se utiliza para agrupar cualquier lista de ejes entre sí para desactivar errores. Si se crea un grupo de ejes, cuando se produce un error en **uno**, a todos se les desactiva **AXIS_ENABLE** y **SERVO**. Se pueden crear varios grupos, aunque un eje no puede pertenecer a más de un grupo. Todas las agrupaciones se pueden borrar con **DISABLE_GROUP(-1)**.
Argumentos	• **eje_i** Una expresión BASIC que se evalúa en un número de eje.
Ejemplo	**DISABLE_GROUP(-1)** **DISABLE_GROUP(0,1,2,3)** **DISABLE_GROUP(4,5,6,7)** **WDOG = ON** **STOP** **enable_b:** **FOR ax = 4 TO 7** **AXIS_ENABLE AXIS(ax) = ON** **NEXT ax** Un sistema de 8 ejes requiere que los ejes 4 a 7 sigan en funcionamiento si los ejes 0 a 3 tienen un error y viceversa. Los ejes se agrupan con **DISABLE_GROUP**. Nota: sólo para usar con MECHATROLINK-II.
Consulte también	N/D

3.2.79 DISPLAY

Tipo	Parámetro de sistema
Sintaxis	**DISPLAY = valor**
Descripción	Determina los canales de E/S que se mostrarán en los LEDs del panel frontal. El parámetro **DISPLAY** se puede utilizar para seleccionar el banco de E/S que se debe mostrar. Los valores se muestran en la tabla siguiente.
Argumentos	N/D
Ejemplo	**DISPLAY = 5** Muestra las salidas 8-15.
Consulte también	N/D

valor	Descripción
0	Entradas 0 a 7 (predeterminado)
1	Entradas 8 a 15
2	Entradas 16 a 23
3	Entradas 24 a 31

valor	Descripción
4	Salidas 0 a 7 (no se utilizan en Trajexia)
5	Salidas 8 a 15
6	Salidas 16 a 23
7	Salidas 24 a 31

3.2.80 DPOS

Tipo: Parámetro de eje (sólo lectura)

Sintaxis: **DPOS**

Descripción: El parámetro de eje **DPOS** contiene la posición solicitada en unidades de usuario, que se genera mediante los comandos de movimiento en el servocontrol. Cuando el controlador está en lazo abierto (**SERVO = OFF**), la posición medida (**MPOS**) se copiará en **DPOS** con el fin de mantener un error de seguimiento 0.
El rango de la posición solicitada se controla con los parámetros de eje **REP_DIST** y **REP_OPTION**. El valor se puede ajustar sin realizar un movimiento mediante el comando **DEFPOS** o el parámetro de eje **OFFPOS**. **DPOS** se restablece a 0 al iniciarse.

Argumentos: N/D

Ejemplo:
```
>> PRINT DPOS AXIS(0)
34.0000
```
La línea anterior devolverá la posición solicitada en unidades de usuario.

Consulte también: **AXIS**, **DPOS**, **DEFPOS**, **DEMAND_EDGES**, **FE**, **MPOS**, **REP_DIST**, **REP_OPTION**, **OFFPOS**, **UNITS**.

3.2.81 DRIVE_ALARM

Tipo: Comando de eje

Sintaxis: **DRIVE_ALARM(VR)**

Descripción: La función DRIVE_ALARM lee la alarma actual del servodriver que está conectado al sistema mediante MECHATROLINK-II. Tras una ejecución correcta, el comando devuelve –1 y almacena el valor en la ubicación de memoria VR especificada por el parámetro VR. Si el comando no puede ejecutarse, se devuelve el valor 0. El comando se ejecuta en el driver para el eje base definido por BASE. El eje base puede cambiar con el modificador AXIS, igual que con los demás comandos y parámetros de eje.
Este comando espera la respuesta del eje. La ejecución del comando puede ser lenta y de duración variable. Si necesita una respuesta rápida, no utilice este comando.

Argumentos:
- **VR**
  El valor de alarma se almacena en la dirección VR tras una ejecución correcta.

Ejemplo:
```
IF NOT DRIVE_ALARM(10) AXIS(2) THEN
  PRINT "Error al leer la alarma del servodriver"
ELSE
  IF VR(10) = 0 THEN
    PRINT "Servodriver en buen estado"
  ELSE
    PRINT "Código de alarma de servo: "; VR(10)
  ENDIF
ENDIF
```
Este ejemplo lee una alarma del eje impulsor 2 del servodriver y presenta dicha información al usuario.

Consulte también: N/D

3.2.82 DRIVE_CLEAR

Tipo	Comando de eje
Sintaxis	**DRIVE_CLEAR**
Descripción	El comando **DRIVE_CLEAR** borra el estado de alarma del servodriver conectado mediante el bus MECHATROLINK-II. Este comando no puede borrar todos los estados de alarma posibles. Algunas alarmas sólo se pueden cancelar si se desconecta la alimentación (tanto de la TJ1-MC__ como del servodriver) y se vuelve a conectar.
Argumentos	N/D
Ejemplo	No hay ejemplo.
Consulte también	**DRIVE_STATUS**.

Precaución
Asegúrese de que no hay ninguna unidad de parámetros o software de ordenador personal conectado al servodriver al ejecutar este comando. De lo contrario, la tarea de programa se pondrá en pausa hasta que se quite la conexión del otro dispositivo al servodriver.

3.2.83 DRIVE_CONTROL

Tipo	Parámetro de eje
Sintaxis	**DRIVE_CONTROL**
Descripción	Cuando se aplica al eje controlado por el servodriver conectado al sistema mediante el bus MECHATROLINK-II, este parámetro selecciona los datos que se monitorizarán mediante **DRIVE_MONITOR** según la tabla siguiente. Si un servodriver para el eje está conectado mediante la TJ1-FL02, este parámetro establece las salidas de la TJ1-FL02. Estableza el bit 8 de este parámetro para activar OUT 0 para un eje. Establezca el bit 9 de este parámetro para activar OUT 1 para un eje. Tenga en cuenta que el comando **HW_PSWITCH** utiliza las mismas salidas.
Argumentos	N/D
Ejemplo	**DRIVE_CONTROL AXIS(2) = 256** En este ejemplo, OUT 0 se activa para el eje 2, conectado mediante la TJ1-FL02.
Consulte también	N/D

Código	Descripción
2	Error de seguimiento (se trata del FE real cuando se utiliza ATYPE = 40)
8	Velocidad de realimentación (con Atype = 41 unidades = velocidad máx./ 40000000H, con otro Atype unidades = unidades referencia/s)
9	Velocidad de comando (unidades iguales a velocidad de realimentación)
10	Velocidad objetivo (unidades iguales a velocidad de realimentación)
11	Referencia de par (fuerza) (con Atype = 42 unidades = par máx./40000000H, con otro Atype unidades = % por encima de par nominal)
14	Monitorización seleccionada con Pn813.0. Resulta útil para monitorizar los monitores de servo (Unxxx)
15	Monitorización seleccionada con Pn813.1. Resulta útil para monitorizar los monitores de servo (Unxxx)

3.2.84 DRIVE_INPUTS

Tipo	Parámetro de eje
Sintaxis	**DRIVE_INPUTS**
Descripción	Este parámetro monitoriza el estado de las entradas del servodriver conectado mediante el bus MECHATROLINK-II. El valor de parámetro se actualiza cada ciclo **SERVO_PERIOD**. Se trata de un canal de operación de bits con los bits tal como se enumeran en la tabla siguiente. El ajuste recomendado es: Pn81E = 4321 y Pn511 = 654x.
Argumentos	N/D
Ejemplo	Todas las entradas se pueden monitorizar en este canal del modo que se indica a continuación (servodriver Sigma-II): **CN1-40 DRIVE_INPUTS bit 12** **CN1-41 DRIVE_INPUTS bit 13** **CN1-42 DRIVE_INPUTS bit 14** **CN1-43 DRIVE_INPUTS bit 15** **CN1-44 DRIVE_INPUTS bit 06** **CN1-45 DRIVE_INPUTS bit 07** **CN1-46 DRIVE_INPUTS bit 08**
Ejemplo	Todas las entradas se pueden monitorizar en este canal del modo que se indica a continuación (servodriver Junma): **CN1-1 DRIVE_INPUTS bit 6** **CN1-2 DRIVE_INPUTS bit 2** **CN1-3 DRIVE_INPUTS bit 1** **CN1-4 DRIVE_INPUTS bit 0**
Consulte también	N/D

Número de bit	Descripción de Sigma-II	Descripción de Junma
0	P_OT	P_OT
1	N_OT	N_OT
2	Señal DEC (se selecciona con Pn511.0)	/DEC
3	Fase A de encoder	N/C
4	Fase B de encoder	N/C
5	Fase C de encoder	N/C
6	Señal EXT1 (se selecciona con Pn511.1)	/EXT1
7	Señal EXT2 (se selecciona con Pn511.2)	N/C
8	Señal EXT3 (se selecciona con Pn511.3)	N/C
9	Salida de freno BRK	/BK
10	Reservado	E_STP
11	Reservado	N/C
12	IO12 (señal de entrada CN1 seleccionada en Pn81E.0)	N/C
13	IO13 (señal de entrada CN1 seleccionada en Pn81E.1)	N/C
14	IO14 (señal de entrada CN1 seleccionada en Pn81E.2)	N/C
15	IO15 (señal de entrada CN1 seleccionada en Pn81E.3)	N/C

3.2.85 DRIVE_MONITOR

Tipo	Parámetro de eje
Sintaxis	**DRIVE_MONITOR**
Descripción	Este parámetro contiene los datos monitorizados del servodriver conectado al sistema mediante el bus MECHATROLINK-II. Los datos que se monitorizarán se seleccionan con **DRIVE_CONTROL** y se pueden mostrar en el ámbito de Trajexia Tools o utilizarse en un programa. Los datos monitorizados se actualizan cada ciclo **SERVO_PERIOD**.
Argumentos	N/D
Ejemplo	No hay ejemplo.
Consulte también	N/D

3.2.86 DRIVE_READ

Tipo	Comando de eje
Sintaxis	**DRIVE_READ(parámetro,tamaño,VR)**
Descripción	La función **DRIVE_READ** lee el parámetro especificado del servodriver que está conectado al sistema Trajexia mediante el bus MECHATROLINK-II. Tras una ejecución correcta, este comando devuelve –1 y coloca el valor leído en la ubicación de memoria VR especificada por el parámetro VR. Si el comando no puede ejecutarse, se devuelve el valor 0. El comando se ejecuta en el driver para el eje base definido con **BASE**. Se puede cambiar con el modificador **AXIS**, igual que con los demás comandos y parámetros de eje. Nota: este comando espera la respuesta del eje, por lo tanto, la ejecución es lenta y de duración variable. No utilice este comando junto con otros comandos que requieran una ejecución rápida. Nota: al ejecutar un comando **DRIVE_READ** se desactivará temporalmente el display del panel frontal del servodriver. Nota: **DRIVE_READ** devuelve –1 si se realiza correctamente. También devuelve –1 sin ningún parámetro leído si el número de parámetro no existe o tiene un tamaño erróneo.
Argumentos	• **parámetro** El número del parámetro que se leerá. Tenga en cuenta que los números de parámetro son hexadecimales. El formato de los datos se puede encontrar en el manual del servodriver. • **tamaño** Para la mayoría de los parámetros, el tamaño normalmente es de 2 bytes. Algunos parámetros especiales pueden tener una longitud de 4 bytes. Los tamaños de cada parámetro se pueden encontrar en el manual del servodriver. • **VR** La dirección VR donde el parámetro leído se almacena tras una ejecución correcta.

Ejemplo	**IF DRIVE_READ($100,2,1) THEN**
	PRINT "La ganancia del lazo de velocidad es: ";VR(1)
	ELSE
	PRINT "No se puede leer la ganancia del lazo de velocidad"
	ENDIF

Consulte también	**DRIVE_WRITE**, **HEX**, **$ (ENTRADA HEXADECIMAL)**.

Precaución
Asegúrese de que no hay ninguna unidad de parámetros o software de ordenador personal conectado al servodriver al ejecutar este comando. De lo contrario, la tarea de programa se pondrá en pausa hasta que se quite la conexión del otro dispositivo al servodriver.

3.2.87 DRIVE_RESET

Tipo	Comando de eje
Sintaxis	**DRIVE_RESET**
Descripción	El comando **DRIVE_RESET** realiza el reset del servodriver conectado mediante el bus MECHATROLINK-II.
Argumentos	N/D
Ejemplo	No hay ejemplo.
Consulte también	N/D

Precaución
Asegúrese de que no hay ninguna unidad de parámetros o software de ordenador personal conectado al servodriver al ejecutar este comando. De lo contrario, la tarea de programa se pondrá en pausa hasta que se quite la conexión del otro dispositivo al servodriver.

3.2.88 DRIVE_STATUS

Tipo: Parámetro de eje (sólo lectura)

Sintaxis: **DRIVE_STATUS**

Descripción: Para los ejes MECHATROLINK-II, este parámetro se establece desde el campo STATUS en la trama de comunicaciones MECHATROLINK-II y se actualiza cada período de servo. Estos bits se pueden ver en la ventana de configuración de Intelligent drives en Trajexia Tools y se pueden utilizar en los programas. En la tabla siguiente se ofrece la explicación de cada bit. (Nota: sólo se enumeran los bits relevantes para los ejes MECHATROLINK-II.) Para obtener una explicación detallada de estos bits de estado, consulte el manual MECHATROLINK-II.
Para los ejes flexibles, este parámetro contiene el estado de registro y las entradas auxiliares, así como la selección de registro. En la segunda tabla siguiente se ofrece la explicación de cada bit. (Nota: sólo se enumeran los bits relevantes para los ejes flexibles.)

Argumentos: N/D

Ejemplo: **PRINT DRIVE_STATUS AXIS(4)**
Este comando imprimirá el valor actual de **DRIVE_STATUS** para el eje (4).

Ejemplo:
```
BASE(3)
ATYPE = 44
  IF (DRIVE_STATUS AND 32) = 32 THEN
  PRINT "La entrada REG 0 está en ON para el eje (3)"
ENDIF
```

Consulte también: **AXIS**, **MARK**, **MARKB**, **REGIST**.

Bit	Descripción (MECHATROLINK-II)
0	Alarma
1	Advertencia
2	Listo
3	Servo ON
4	Alimentación conectada
5	Bloqueo de máquina
6	Posición de inicio
7	En posición/velocidad
8	Salida finalizada
9	Límite de par
10	Registro finalizado
11	En límite de rango/velocidad

Bit	Descripción (eje flexible)
0	MARK
1	MARKB
2	Valor actual seleccionado de REG-0
3	Valor actual seleccionado de REG-1
4	Valor actual de AUX IN
5	Valor actual de REG 0
6	Valor actual de REG 1

3.2.89 DRIVE_WRITE

Tipo: Comando de eje

Sintaxis: **DRIVE_WRITE(parámetro, tamaño, valor [,modo])**

Descripción: La función **DRIVE_WRITE** escribe en el parámetro especificado del servodriver mediante el bus MECHATROLINK-II. Tras una ejecución correcta, este comando devuelve –1. Si el comando no puede ejecutarse, se devuelve el valor 0. El comando se ejecuta en el driver para el eje base definido con **BASE**. Se puede cambiar con el modificador **AXIS**, igual que con los demás comandos y parámetros de eje. Para algunos parámetros, el driver se tiene que apagar y volver a encender. El comando **DRIVE_RESET** se puede utilizar para esta finalidad.
Nota: este comando espera la respuesta del eje, por lo tanto, la ejecución es lenta y de duración variable. No utilice este comando junto con otros comandos que requieran una ejecución rápida.
Nota: al ejecutar un comando **DRIVE_WRITE** se desactivará temporalmente el display del panel frontal del servodriver.
Nota: **DRIVE_WRITE** devuelve –1 si se realiza correctamente. También devuelve –1 si ningún parámetro es leído si el número de parámetro no existe o tiene un tamaño erróneo.

Argumentos:
- **parámetro**
  El número del parámetro en el que se escribirá. Tenga en cuenta que los números de parámetro son hexadecimales. El formato de los datos se puede encontrar en el manual del servodriver. Consulte el formato en dicho manual.
- **tamaño**
  Para la mayoría de los parámetros, el tamaño normalmente es de 2 bytes. Algunos parámetros especiales pueden tener una longitud de 4 bytes. Los tamaños de cada parámetro se pueden encontrar en el manual del servodriver.
- **valor**
  El valor que se escribirá en el parámetro de driver.
- **modo**
  El modo de escritura. Valores posibles: 0 (u omitido), escribir y almacenar en RAM; 1, escribir y almacenar en EPROM.

Ejemplo:
```
IF DRIVE_WRITE($100,2,90) THEN
  PRINT "La nueva ganancia del lazo de velocidad es: 90"
ELSE
  PRINT "No se puede escribir la ganancia del lazo de velocidad en la RAM"
ENDIF
```

Consulte también:
- **DRIVE_READ**, **DRIVE_RESET**, **$ (ENTRADA HEXADECIMAL)**

Precaución
Asegúrese de que no hay ninguna unidad de parámetros o software de ordenador personal conectado al servodriver al ejecutar este comando. De lo contrario, la tarea de programa se pondrá en pausa hasta que se quite la conexión del otro dispositivo al servodriver.

3.2.90 EDIT

Tipo: Comando de programa

Sintaxis:
EDIT [número_línea]
ED [número_línea]

Descripción: El comando **EDIT** inicia el editor de pantalla integrado que permite modificar un programa del controlador mediante un terminal VT100. Se editará el programa seleccionado actualmente.
Los comandos del editor son los siguientes:
Este comando se ha implementado para un terminal (VT100) offline.
En Trajexia Tools, los usuarios pueden seleccionar el comando desde el menú **Program** (Programa).
- Editor rápido: [CTRL] K y D
- Eliminar línea: [CTRL] Y

Argumentos:
- **número_línea**
  El número de la línea en la que se iniciará la edición.

Ejemplo: No hay ejemplo.

Consulte también: **SELECT**.

3.2.91 ELSE

Consulte **IF..THEN..ELSE..ENDIF**.

3.2.92 ELSEIF

Consulte **IF..THEN..ELSE..ENDIF**.

3.2.93 ENCODER

Tipo	Parámetro de eje (sólo lectura)
Sintaxis	**ENCODER**
Descripción	El parámetro de eje **ENCODER** contiene una copia sin procesar del encoder. El parámetro de eje **MPOS** contiene la posición medida que se calcula a partir del valor de **ENCODER** automáticamente, permitiendo desbordamientos y offsets.
Argumentos	N/D
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**, **MPOS**.

3.2.94 ENCODER_BITS

Tipo	Parámetro de eje
Sintaxis	**ENCODER_BITS = valor**
Descripción	Este parámetro de eje configura la interfaz del número de bits de encoder para los ejes de encoder absoluto SSI y EnDat de eje flexible. El parámetro sólo se aplica a los ejes de los valores 47 y 48 de **ATYPE**. Cuando se aplican a un encoder absoluto EnDat del eje flexible, los bits 0 a 7 del parámetro se deben configurar en el número total de bits de encoder. Los bits 8 a 14 se deben establecer al número de bits multigiro que se utilizarán. Cuando se aplican a un encoder absoluto SSI del eje flexible, los bits 0 a 5 del parámetro se deben configurar en el número de bits de encoder. El bit 6 debe ser 1 para operación binaria, ó 0 para código Gray. Nota: si se utiliza un eje de encoder absoluto de eje flexible, es fundamental establecer este parámetro para el eje antes de configurar ATYPE.
Argumentos	N/D
Ejemplo	**ENCODER_BITS = 25 + (256 * 12)** **ATYPE = 47** En este ejemplo se utiliza un encoder EnDat de 25 bits, que tiene 12 bits para el valor de multigiro y 13 bits para una revolución.
Ejemplo	**ENCODER_BITS = 12 + (64 * 1)** **ATYPE = 48** En este ejemplo se utiliza un encoder SSI de 12 bits (4.096 posiciones por revolución) con tipo de salida binaria.
Consulte también	**AXIS**.

3.2.95 ENCODER_CONTROL

Tipo	Parámetro de eje
Sintaxis	**ENCODER_CONTROL = valor**
Descripción	El parámetro **ENCODER_CONTROL** sólo es aplicable al eje EnDat absoluto de eje flexible con un valor 47 de ATYPE. Este parámetro controla el modo en que el encoder EnDat vuelve a su posición. El encoder se puede configurar para volver cíclicamente a su posición o se puede configurar en un modo de lectura/escritura de parámetros. El valor predeterminado después de la inicialización es el modo de retorno de posición cíclico. Para obtener más información, consulte la especificación de interfaz del encoder absoluto EnDat.
Argumentos	N/D
Ejemplo	**ENCODER_CONTROL AXIS(1) = 0** Este comando establece el modo de retorno de posición cíclico.
Ejemplo	**ENCODER_CONTROL AXIS(1) = 1** Este comando establece el modo de lectura/escritura de parámetros.
Consulte también	**AXIS**, **ENCODER**, **ENCODER_BITS**.

3.2.96 ENCODER_ID

Tipo: Parámetro de eje (sólo lectura)

Sintaxis: **ENCODER_ID**

Descripción: Este parámetro devuelve el valor ID de un encoder absoluto para el eje. Este parámetro sólo es aplicable al eje Tamagawa absoluto de eje flexible con el valor 46 de **ATYPE**.
Devuelve el parámetro ENID desde el encoder, que está configurado en 17. Para obtener más información, consulte la especificación de interfaz de encoder absoluto Tamagawa.
Si se aplica a un eje de valor ATYPE distinto de 46, este parámetro devuelve 0.

Argumentos: N/D

Ejemplo:
>>PRINT ENCODER_ID AXIS (1)
17.0000
Este comando imprimirá el valor de ID de encoder absoluto para el eje 1.

Consulte también: **AXIS**, **ENCODER**, **ENCODER_BITS**.

3.2.97 ENCODER_RATIO

Tipo: Parámetro de eje

Sintaxis: **ENCODER_RATIO(denominador,numerador)**

Descripción: Permite que se ajuste la escala del contaje de encoder entrante a un número no entero mediante la ecuación:
MPOS = (numerador)/(denominador) x entrada de flancos de encoder
A diferencia de los parámetros **UNITS**, **ENCODER_RATIO** afecta a **MOVECIRC** y **CAMBOX**.

Nota: se deben evitar relaciones grandes ya que provocarán la pérdida de resolución o una suavidad más reducida en el movimiento. El contaje de encoder físico real es la resolución básica del eje y el uso de este comando puede reducir la capacidad del controlador Motion para alcanzar todas las posiciones de forma precisa.
Nota: ENCODER_RATIO no reemplaza a UNITS. Utilice **ENCODER_RATIO** sólo cuando sea absolutamente necesario. Para todos los demás ajustes a escala de eje utilice **UNITS**.

Argumentos:
- **denominador**
  Un número entre 0 y 16.777.215 que se utiliza para definir el denominador en la ecuación anterior.
- **numerador**
  Un número entre 0 y 16.777.215 que se utiliza para definir el numerador en la ecuación anterior.

Ejemplo:
ENCODER_RATIO(8192,7200)
UNITS = 20
Una mesa rotativa tiene un servomotor conectado directamente a su centro de rotación. Se monta un encoder en la parte trasera del servomotor y devuelve un valor de 8.192 contajes por revolución. La aplicación requiere que la mesa esté calibrada en grados, pero un grado es un número entero de contajes.

Consulte también: N/D

3.2.98 ENCODER_READ

Tipo: Comando de eje

Sintaxis: **ENCODER_READ(dirección)**

Descripción: El comando **ENCODER_READ** sólo es aplicable a un eje EnDat absoluto de eje flexible con un valor 47 de **ATYPE**. El parámetro devuelve un parámetro de encoder de 16 bits almacenado en una dirección especificada. Los bits 8 a 15 de la dirección son los ajustes de campo MRS de EnDat y los bits 0 a 7 son el offset dentro del bloque MRS de EnDat. Si se produce un error de CRC, este comando devolverá –1. Para obtener más información, consulte la especificación de interfaz del encoder absoluto EnDat.

Argumentos:
- **dirección**
  Especifica el campo MRS de EnDat que se leerá.

Ejemplo:
VR(100) = ENCODER_READ($A10D) AXIS(7)
Este comando leerá el número de bits de encoder y pondrá dicho valores en la ubicación de memoria VR(10).

Consulte también: **AXIS**, **ENCODER**, **ENCODER_BITS**.

3.2.99 ENCODER_STATUS

Tipo	Parámetro de eje (sólo lectura).
Sintaxis	**ENCODER_STATUS**
Descripción	Este parámetro devuelve el estado del encoder absoluto. Este parámetro sólo es aplicable al eje Tamagawa absoluto de eje flexible con el valor 46 de **ATYPE**. Devuelve el campo de estado SF y el campo de error de encoder ALMC. El campo SF está en los bits 0 a 7, mientras que el campo ALMC está en los bits 8 a 15. Para obtener más información, consulte la especificación de interfaz de encoder absoluto Tamagawa. Si se aplica a un eje de valor **ATYPE** distinto de 46, este parámetro devuelve 0.
Argumentos	N/D
Ejemplo	**PRINT (ENCODER_STATUS AXIS (1) AND 255)** Este comando imprimirá el campo SF del encoder absoluto Tamagawa para el eje 1.
Consulte también	**AXIS**, **ENCODER**, **ENCODER_BITS**.

3.2.100 ENCODER_TURNS

Tipo	Parámetro de eje (sólo lectura)
Sintaxis	**ENCODER_TURNS**
Descripción	El parámetro **ENCODER_TURNS** devuelve el número de contaje multigiro del encoder. Sólo es aplicable al eje Tamagawa absoluto de eje flexible con el valor 46 de **ATYPE** y al eje EnDat absoluto de eje flexible con el valor 47 de **ATYPE**. Los datos de multigiro no se aplican automáticamente al parámetro de eje **MPOS** después de la inicialización. El programador de la aplicación debe aplicarlos desde el programa mediante los comandos **OFFPOS** o **DEFPOS** según sea necesario. Si se aplica a un eje de valor **ATYPE** distinto de 46 ó 47, el parámetro devuelve 0.
Argumentos	N/D
Ejemplo	**PRINT ENCODER_TURNS AXIS (1)** Este comando imprimirá los contajes de multigiro de encoder absoluto para el eje 1.
Consulte también	**AXIS**, **ENCODER**, **ENCODER_BITS**.

3.2.101 ENCODER_WRITE

Tipo	Comando de eje
Sintaxis	**ENCODER_WRITE(dirección, valor)**
Descripción	El comando **ENCODER_WRITE** sólo es aplicable a un eje EnDat absoluto de eje flexible con un valor 47 de **ATYPE**. El comando escribe el parámetro de encoder especificado por la dirección. Los bits 8 a 15 de la dirección son los ajustes de campo MRS de EnDat y los bits 0 a 7 son el offset dentro del bloque MRS de EnDat. Si se produce un error de CRC, este comando devolverá 0. Al escribir en la dirección 0 se realiza una función de reset de encoder. Para obtener más información, consulte la especificación de interfaz del encoder absoluto EnDat. Para escribir correctamente un parámetro de encoder con este comando, el parámetro **ENCODER_WRITE** se debe configurar en 1, modo de lectura/escritura de parámetros de encoder.
Argumentos	• **dirección** Especifica el campo MRS de EnDat en el que se escribirá. • **valor** Una expresión BASIC.
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**, **ENCODER**, **ENCODER_BITS**, **ENCODER_CONTROL**.

3.2.102 ENDIF

Consulte **IF..THEN..ELSE..ENDIF**.

3.2.103 ENDMOVE

Tipo	Parámetro de eje
Sintaxis	**ENDMOVE**
Descripción	El parámetro de eje **ENDMOVE** contiene la posición del final del movimiento actual en unidades del usuario. Si el parámetro de eje **SERVO** está en ON, el parámetro **ENDMOVE** se puede escribir para producir un cambio de paso en la posición solicitada (**DPOS**). Nota: Como la posición medida no se cambia inicialmente, se debe considerar el límite de error de seguimiento (**FE_LIMIT**). Si el cambio de la posición solicitada es demasiado grande, se superará el límite.
Argumentos	N/D
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**, **DPOS**, **FE_LIMIT**, **UNITS**.

3.2.104 EPROM

Tipo	Comando de programa
Sintaxis	**EPROM**
Descripción	El comando **EPROM** almacena los programas BASIC en la memoria RAM respaldada por las baterías de la TJ1-MC__ en la memoria EPROM flash. Con el parámetro de sistema POWER_UP se controla si los programas almacenados en la memoria EPROM flash se copian en la RAM al inicio. Nota: Trajexia Tools ofrece este comando como un botón en el panel de control. Además, en las pantallas emergentes se solicitará escribir los datos de programa en la memoria flash.
Argumentos	N/D
Ejemplo	No hay ejemplo.
Consulte también	**POWER_UP**, **RUNTYPE**.

3.2.105 ERROR_AXIS

Tipo	Parámetro de sistema (sólo lectura)
Sintaxis	**ERROR_AXIS**
Descripción	El parámetro de eje **ERROR_AXIS** contiene el número del eje que ha provocado el error de movimiento. Los errores de movimiento se producen cuando el estado **AXISSTATUS** de uno de los ejes coincide con el ajuste de **ERRORMASK**. En este caso, el interruptor de activación (**WDOG**) se desactivará, el parámetro **MOTION_ERROR** tendrá el valor 1 y el parámetro **ERROR_AXIS** contendrá el número del primer eje que tenga el error.
Argumentos	N/D
Ejemplo	No hay ejemplo.
Consulte también	**AXISSTATUS**, **ERRORMASK**, **MOTION_ERROR**, **WDOG**.

3.2.106 ERROR_LINE

Tipo	Parámetro de tarea (sólo lectura)
Sintaxis	**ERROR_LINE**
Descripción	El parámetro **ERROR_LINE** contiene el número de la línea que ha provocado el último error de tiempo de ejecución BASIC en la tarea de programa. Este valor sólo es válido cuando el parámetro **BASICERROR** es **TRUE**. Cada tarea tiene su propio parámetro **ERROR_LINE**. Utilice el modificador **PROC** para acceder al parámetro para una determinada tarea. Sin **PROC** se supondrá la tarea actual.
Argumentos	N/D
Ejemplo	**>> PRINT ERROR_LINE PROC(4)** **23.0000**
Consulte también	**BASICERROR**, **PROC**, **RUN_ERROR**.

3.2.107 ERRORMASK

Tipo	Parámetro de eje
Sintaxis	**ERRORMASK**
Descripción	El parámetro de eje **ERRORMASK** contiene un valor de máscara al que se le aplica la operación **AND** bit a bit con el parámetro de eje **AXISSTATUS** en cada ciclo de servo para determinar si se ha producido un error de movimiento. Cuando se produce un error de movimiento, el interruptor de activación (**WDOG**) se desactivará, el parámetro **MOTION_ERROR** tendrá el valor 1 y el parámetro **ERROR_AXIS** contendrá el número del primer eje que tenga el error. Compruebe el parámetro **AXISVALUES** para determinar las asignaciones de bit de estado. El ajuste predeterminado de **ERRORMASK** es 268.
Argumentos	N/D
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**, **AXISSTATUS**, **MOTION_ERROR**, **WDOG**.

Precaución
El usuario debe definir los casos en que se genera un error de movimiento. Para una operación segura, se recomienda generar un error de movimiento cuando el error de seguimiento haya excedido su límite en todos los casos. This is done by setting bit 8 of **ERRORMASK**

3.2.108 ETHERNET

Tipo	Comando de sistema
Sintaxis	**ETHERNET(función, número_unidad, parámetro [,valores])**
Descripción	El comando **ETHERNET** se utiliza para leer y establecer determinadas funciones de las comunicaciones Ethernet. El comando **ETHERNET** se debe introducir en la línea de comandos con Trajexia Tools en modo desconectado a través del puerto serie 0. Nota: tendrá que apagar y encender el Trajexia para activar los nuevos parámetros.
Argumentos	• **función** 0 = lectura, 1 = escritura. • **número_unidad** –1. • **parámetro** 0 = dirección IP; 2 = máscara de subred; 3 = dirección MAC; 8 = gateway; 11 = caché ARP (sólo lectura). • **Valores** El parámetro necesario para una operación de escritura.
Ejemplo	**ETHERNET(1,-1,0,192,200,185,2)** Establezca la dirección IP de Trajexia en 192.200.185.002.
Consulte también	N/A

3.2.109 EX

Tipo	Comando de sistema
Sintaxis	**EX[(opción)]**
Descripción	Realiza el reset del controlador como si se hubiera vuelto a encender. Hay dos tipos de reset realizados por el comando **EX** . **EX** sin el argumento o **EX(0)** realiza el reset de software del controlador. **EX(1)** realiza el reset de hardware del controlador
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	N/A

3.2.110 EXP

Tipo	Función matemática
Sintaxis	**EXP(expresión)**
Descripción	La función **EXP** devuelve el valor exponencial de la expresión.
Argumentos	• **expresión** Cualquier expresión BASIC válida.
Ejemplo	**>> print exp(1.0)** **2.7183**
Consulte también	N/A

3.2.111 FALSE

Tipo	Constante (sólo lectura)
Sintaxis	**FALSE**
Descripción	La constante **FALSE** devuelve el valor numérico 0.
Argumentos	N/A

Ejemplo	**prueba:**
	res = IN(0) OR IN(2)
	IF res = FALSE THEN
	PRINT "Las entradas están desconectadas"
	ENDIF

Consulte también	N/A

3.2.112 FAST_JOG

Tipo	Parámetro de eje
Sintaxis	**FAST_JOG**
Descripción	El parámetro de eje **FAST_JOG** contiene el número de entrada que se utilizará como la entrada de jog rápido. El número puede ser de 0 a 7. De forma predeterminada, el parámetro se establece en –1 y no hay seleccionada ninguna entrada. La entrada de jog rápido controla la velocidad jog entre dos velocidades. Si se establece la entrada de jog rápido, para la operación jog se utilizará la velocidad tal como se indica mediante el parámetro de eje **SPEED**. Si no se establece la entrada, se utilizará la velocidad indicada mediante el parámetro de eje **JOGSPEED**. Nota: Esta entrada se activa por baja.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**, **FWD_JOG**, **JOGSPEED**, **REV_JOG**, **SPEED**.

3.2.113 FASTDEC

Tipo	Parámetro de eje
Sintaxis	**FASTDEC**
Descripción	El valor predeterminado es cero. Si se especifica un parámetro **FASTDEC** distinto de cero, el eje decelerará a cero con rampa a esta velocidad de deceleración cuando se alcance un final de carrera o posición de eje.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	N/A

3.2.114 FE

Tipo	Parámetro de eje (sólo lectura)
Sintaxis	**FE**
Descripción	El parámetro de eje **FE** contiene el error de posición en unidades de usuario. Se calcula mediante la posición solicitada (parámetro de eje **DPOS**) menos la posición medida (parámetro de eje **MPOS**). El valor del error de seguimiento se puede comprobar si se utilizan los parámetros de eje **FE_LIMIT** y **FE_RANGE**.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**, **DPOS**, **FE_LIMIT**, **FE_RANGE**, **MPOS**, **UNITS**.

3.2.115 FE_LATCH

Tipo	Parámetro de eje (sólo lectura)
Sintaxis	**FE_LATCH**
Descripción	Contiene el valor **FE** inicial que ha provocado que el eje coloque el controlador en el estado MOTION_ERROR. Este valor sólo se establece cuando **FE** excede **FE_LIMIT** y el parámetro **SERVO** se ha establecido en 0. **FE_LATCH** se restablece a 0 cuando el parámetro **SERVO** del eje se vuelve a establecer en 1.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	N/A

3.2.116 FE_LIMIT

Tipo	Parámetro de eje
Sintaxis	**FE_LIMIT**
Descripción	El parámetro de eje **FE_LIMIT** contiene el límite del error de seguimiento máximo permitido en unidades de usuario. Cuando se excede, se establece el bit 8 del parámetro **AXISSTATUS** del eje. Si el parámetro **ERRORMASK** se ha configurado correctamente, se generará un error de movimiento. Este límite se utiliza como protección contra situaciones de fallo, como, por ejemplo, un bloqueo mecánico, pérdida de la realimentación del encoder, etc.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**, **AXISSTATUS**, **ERRORMASK**, **FE**, **FE_RANGE**, **UNITS**.

3.2.117 FE_LIMIT_MODE

Tipo	Parámetro de eje
Sintaxis	**FE_LIMIT_MODE = valor**
Descripción	Cuando este parámetro se establece en 0, el eje provocará un error **MOTION_ERROR** inmediatamente si **FE** excede el valor **FE_LIMIT**. Si **FE_LIMIT_MODE** se establece en 1, el eje sólo generará un error **MOTION_ERROR** cuando **FE** exceda **FE_LIMIT** durante 2 períodos de servo consecutivos. Esto significa que si **FE_LIMIT** se excede sólo durante un período de servo, se omitirá. El valor predeterminado de **FE_LIMIT_MODE** es 0.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	N/A

3.2.118 FE_RANGE

Tipo	Parámetro de eje
Sintaxis	**FE_RANGE**
Descripción	El parámetro de eje **FE_RANGE** contiene el límite del rango de advertencia del error de seguimiento en unidades de usuario. Cuando el error de seguimiento excede este valor en un servoeje, se activará el bit 1 del parámetro de eje **AXISSTATUS**. Este rango se utiliza como primera indicación de las situaciones de fallo en la aplicación (compárese con **FE_LIMIT**).
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**, **AXISSTATUS**, **ERRORMASK**, **FE**, **UNITS**.

3.2.119 FHOLD_IN

Tipo	Parámetro de eje
Sintaxis	**FHOLD_IN** **FH_IN**
Descripción	El parámetro de eje **FHOLD_IN** contiene el número de entrada que se utilizará como la entrada de retención de realimentación. El rango de entrada válido va de 0 a 31. Los valores de 0 a 15 representan entradas presentes físicamente del conector de E/S de la TJ1-MC__ y son comunes para todos los ejes. Los valores 16 a 27 representan entradas de software que se pueden utilizar libremente en programas y comandos, como, por ejemplo, IN y OP. También son comunes para todos los ejes. Los valores 28 a 31 se asignan directamente a las entradas de driver presentes en el conector CN1 y son únicos para cada eje. Las entradas de driver que se asignan a las entradas 28 a 31 dependen del ajuste Pn81E del parámetro de servodriver. El ajuste recomendado es Pn81E = 0x4321, con la siguiente asignación: De forma predeterminada, el parámetro se establece en –1 y no hay seleccionada ninguna entrada. Nota: Esta entrada se activa por baja.
Sigma II	• entrada 28: CN1-40 • entrada 29: CN1-41 • entrada 30: CN1-42 • entrada 31: CN1-43
Sigma III	• entrada 28: CN1-13 • entrada 29: CN1-7 • entrada 30: CN1-8 • entrada 31: CN1-9
Junma	• entrada 26: CN1-2 • entrada 27: CN1-1
	Para obtener más información sobre el ajuste del parámetro de driver Pn81E, consulte el manual del servodriver. De forma predeterminada, el parámetro se establece en –1 y no hay seleccionada ninguna entrada.

Si se establece un número de entrada y se establece la entrada de retención de realimentación, la velocidad del movimiento del eje se cambia al valor seleccionado en el parámetro de eje **FHSPEED**. El movimiento actual no se cancela. Además, se establece el bit 7 del parámetro **AXISSTATUS**. Cuando se vuelve a hacer un reset de la entrada, cualquier movimiento en curso cuando se estableció la entrada volverá a la velocidad programada. Nota: esta característica sólo funciona en movimientos con velocidad controlada. Los movimientos que no tienen controlada la velocidad (**CAMBOX**, **CONNECT** y **MOVELINK**) no se ven afectados.

Argumentos N/A

Ejemplo No hay ejemplo.

Consulte también **AXIS**, **AXISSTATUS**, **UNITS**.

3.2.120 FHSPEED

Tipo Parámetro de eje

Sintaxis **FHSPEED**

Descripción El parámetro de eje **FHSPEED** contiene la velocidad de retención de realimentación. Este parámetro se puede establecer en un valor en unidades de usuario/s a la velocidad que se moverá el eje cuando se active la entrada de retención de realimentación. El movimiento actual no se cancela. **FHSPEED** puede tener cualquier valor positivo, incluido 0. El valor predeterminado es 0. Nota: esta característica sólo funciona en movimientos con velocidad controlada. Los movimientos que no tienen controlada la velocidad (**CAMBOX**, **CONNECT** y **MOVELINK**) no se ven afectados.

Argumentos N/A

Ejemplo No hay ejemplo.

Consulte también **AXIS**, **AXISSTATUS**, **FHOLD_IN**, **UNITS**.

3.2.121 FINS_COMMS

Tipo Comando de comunicaciones

Sintaxis **FINS_COMMS(tipo, red, nodo, unidad, área_remota, offset_remoto, longitud, área_local, offset_local, tiempo de espera [, ip1, ip2, ip3, ip4])**

Descripción FINS (Factory Interface Network Service) es un protocolo de comunicaciones propio de OMRON. En Trajexia se ha implementado un subconjunto de este protocolo. El protocolo FINS se ha implementado con la intención de permitir una comunicación perfecta con otros dispositivos (PLCs, HMIs, etc.) y software (CX-Drive, CX-Server, etc.) de OMRON. Para obtener más información sobre el protocolo de comunicaciones FINS, consulte la sección 4.2.4 y el Manual de referencia de comandos de comunicaciones, número de catálogo W342-E1, secciones 3 y 5.
Trajexia incorpora funciones de cliente FINS, por lo que puede iniciar las comunicaciones FINS con dispositivos esclavos FINS mediante **FINS_COMMS**. Se ha implementado los comandos FINS 0101 (leer memoria) y FINS 0102 (escribir memoria). Con FINS 0101, la memoria se puede leer desde otros dispositivos con la función de servidor FINS. FINS 0102 se puede utilizar para escribir datos en dispositivos con la función de servidor FINS.
Este comando devuelve uno de los siguientes valores, según el resultado de la ejecución:
–1: el comando se ha ejecutado correctamente.
0: el comando ha fallado.
1: la solicitud no se ha enviado porque el cliente o el protocolo FINS están ocupados.
2: uno o varios de los parámetros de la solicitud no son válidos.
3: área de memoria de origen no válida.
4: se ha enviado la solicitud, pero no se ha recibido ninguna respuesta del servidor remoto en el período de tiempo de espera.
5: código de respuesta de error recibido del servidor remoto.

Argumentos

- **tipo**
  El tipo del comando FINS. 0 significa FINS 0101, leer la memoria desde el servidor FINS. 1 significa FINS 0102, escribir la memoria en el servidor remoto.
- **red**
  La red de destino. Para obtener más detalles, consulte el Manual de referencia de comandos de comunicaciones, número de catálogo W342-E1, sección 3.
- **nodo**
  El nodo del servidor FINS de destino. Para obtener más detalles, consulte el Manual de referencia de comandos de comunicaciones, número de catálogo W342-E1, sección 3.
- **unidad**
  El número de unidad del servidor FINS de destino. Para obtener más detalles, consulte el Manual de referencia de comandos de comunicaciones, número de catálogo W342-E1, sección 3.
- **área_remota**
  El área de memoria a la que se accede en el servidor FINS de destino. Rango: 128..255. Tenga en cuenta que esta área debe tener uno de los siguientes valores si el destino es otro sistema Trajexia: 0xB0: valor VR entero; 0x82: valor TABLE entero; 0xC2: valor TABLE flotante.
- **offset_remoto**
  El offset de memoria en el servidor FINS de destino. Rango: 0...65.535. Tenga en cuenta que este rango estará más limitado a las direcciones TABLE o VR máxima si el destino es otro sistema Trajexia.
- **longitud**
  El número de elementos que se transferirán. El rango dependerá de la longitud de trama FINS y las funciones del cliente y los servidores remotos. El rango para un sistema Trajexia es un valor entero de 1 a 700 ó un valor de coma flotante de 1 a 350.
- **área_local**
  El área de memoria local (origen). Tenga en cuenta que esta área debe tener uno de los siguientes valores si el destino es otro sistema Trajexia: 0x00: valor VR entero; 0x01: valor TABLE entero; 0x02: valor TABLE flotante.
- **offset_local**
  El offset del primer valor del área de memoria local (origen). El rango depende del tamaño de matriz VR o TABLE y el valor de la longitud del dato.
- **tiempo de espera**
  El número de milisegundos que se esperará una respuesta del servidor FINS de destino antes de que se agote el tiempo de espera.
- **IP1, IP2, IP3, IP4**
  Parámetros opcionales que definen la dirección IP del servidor remoto (destino). Estos argumentos se deben utilizar si tanto el sistema Trajexia como el servidor FINS de destino no pertenecen a la misma red.

Ejemplo

Un sistema Trajexia y un PLC CJ1 de OMRON con una unidad de Ethernet CJ1W-ETN11 están conectados a la misma red. La dirección IP del sistema Trajexia es 192.168.0.5. La dirección IP de la unidad Ethernet del PLC es 192.168.0.12.
Cuando se ejecuta el comando **FINS_COMMS(0,0,12,0,$82, 1000,20,0,500,5000,192,168,0,12)**, se leen 20 canales (**longitud = 20**) del área de memoria DM del PLC (**área_remota = $82**), a partir de DM1000 (**offset_remoto = 1000**) y se escriben en la memoria VR de Trajexia en formato de entero (**área_local = 0**), a partir de VR(500) (**offset_local = 500**). Por lo tanto, los valores del rango de memoria DM1000 a DM1019 del PLC se colocan en la memoria Trajexia VR(500) a VR(519) de Trajexia. El tiempo de espera se establece en 5 segundos.
Cuando se ejecuta el comando **FINS_COMMS(1,0,12,0,$80, 50,10,0,300,3000,192,168,0,12)**, se escriben 10 canales (**longitud = 10**) de la memoria VR de Trajexia como enteros (**área_local = 0**), a partir de VR(300) (**offset_local = 300**) en el área CIO del PLC (**área_remota = $80**), a partir de CIO50 (**offset_remoto = 50**). Por lo tanto, los valores del rango de memoria VR(300) a VR(309) de Trajexia se colocan en la memoria CIO50 a CIO59 del PLC. El tiempo de espera se establece en 3 segundos.

Consulte también

N/A

3.2.122 FLAG

Tipo	Comando de sistema
Sintaxis	**FLAG(número_indicador [,valor])**
Descripción	El comando **FLAG** se utiliza para establecer y leer un banco de 32 bits indicadores. El comando **FLAG** se puede utilizar con uno o dos parámetros. Cuando se especifica un parámetro, se devuelve el estado del bit indicador especificado. Si se especifican dos parámetros, el indicador especificado se establece en el valor del segundo parámetro. El comando **FLAG** se proporciona por compatibilidad con controladores anteriores y no se recomienda para programas nuevos.
Argumentos	• **número_indicador** El número de indicador es un valor de 0 a 31. • **valor** Si se especifica, se trata del estado para establecer el indicador especificado en ON u OFF. También se puede escribir como 1 ó 0.
Ejemplo	**FLAG(27,ON)** Pone el bit de indicador 27 en ON.
Consulte también	N/A

3.2.123 FLAGS

Tipo	Comando de sistema
Sintaxis	**FLAGS([valor])**
Descripción	Lee y establece los indicadores especificados por **FLAGS** como un bloque. El comando **FLAGS** se proporciona por compatibilidad con controladores anteriores y no se recomienda para programas nuevos. Los 32 bits de indicador se pueden leer con **FLAGS** y establecerse con **FLAGS(valor)**.
Argumentos	• **valor** El decimal equivalente del patrón de bits al que se deben establecer los indicadores. Consulte la siguiente tabla.
Ejemplo	**FLAGS(146) ' 2 + 16 + 128** Establece los indicadores 1, 4 y 7 en ON y todos los demás en OFF.
Ejemplo	**IF (FLAGS and 8) <>0 then GOSUB otra_parte** Prueba si se ha establecido el indicador 3.
Consulte también	N/A

Número de bit	Valor decimal
0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128

3.2.124 FOR..TO..STEP..NEXT

Tipo: Comando de control de programa

Sintaxis:

```
FOR variable = inicio TO fin [STEP incremento]
  comandos
NEXT variable
```

Descripción: El lazo **FOR ... NEXT** permite que el segmento de programa entre las instrucciones **FOR** y **NEXT** se repita una serie de veces.
Al entrar en este lazo, la variable se inicializa en el valor de inicio y, a continuación, se ejecuta el bloque de comandos. Al llegar al comando **NEXT**, la variable aumenta según el incremento especificado después de **STEP**. El valor **STEP** puede ser positivo o negativo; si se omite, se supone que el valor es 1.
Mientras la variable sea menor o igual que el valor de fin, el bloque de comandos se ejecuta repetidamente hasta que la variable sea mayor que fin, tras lo cual la ejecución del programa seguirá después de **NEXT**.
Nota: las instrucciones **FOR ... NEXT** se pueden anidar hasta 8 niveles en un programa BASIC.

Argumentos:

- **variable**
  Una expresión BASIC.
- **inicio**
  Una expresión BASIC.
- **fin**
  Una expresión BASIC.
- **incremento**
  Una expresión BASIC.
- **comandos**
  Uno o varios comandos BASIC.

Ejemplo:

```
FOR opnum = 8 TO 13
  OP(opnum,ON)
NEXT opnum
```

Este lazo activa las salidas 8 a 13.

Ejemplo:

```
lazo:
  FOR dist = 5 TO –5 STEP –0.25
    MOVEABS(dist)
    GOSUB seleccionar
  NEXT dist
```

El incremento de **STEP** puede ser positivo o negativo.

Ejemplo:

```
lazo1:
  FOR I1 = 1 TO 8
  lazo2:
    FOR I2 = 1 TO 6
      MOVEABS(I1 * 100,I2 * 100)
      GOSUB 1000
    NEXT I2
  NEXT I1
```

Las instrucciones **FOR..TO..STEP..NEXT** pueden estar anidadas (hasta una profundidad de 8 niveles) siempre que los comandos **FOR** y **NEXT** interiores estén dentro del lazo **FOR..TO..STEP..NEXT** exterior.

Consulte también: **REPEAT..UNTIL**, **WHILE..WEND**.

3.2.125 FORWARD

Tipo	Comando de eje
Sintaxis	**FORWARD** **FO**
Descripción	El comando **FORWARD** mueve un eje continuamente hacia delante a la velocidad definida en el parámetro de eje **SPEED**. La velocidad de aceleración se define mediante el parámetro de eje **ACCEL**. **FORWARD** funciona en el eje base predeterminado (establecido con **BASE**) a menos que se utilice **AXIS** para especificar un eje base temporal. Nota: El movimiento directo se puede parar si se ejecuta el comando **CANCEL** o **RAPIDSTOP**, o si se alcanza el límite de marcha directa.
Argumentos	N/A
Ejemplo	**inicio:** **FORWARD** **WAIT UNTIL IN(0) = ON ' Esperar la señal de parada** **CANCEL**
Consulte también	**AXIS**, **CANCEL**, **RAPIDSTOP**, **REVERSE**, **UNITS**.

3.2.126 FPGA_VERSION

Tipo	Parámetro de entrada
Sintaxis	**FPGA_VERSION SLOT(número_unidad)**
Descripción	Este parámetro devuelve la versión de FPGA de la unidad indicada con número_de_unidad en un sistema de control.
Argumentos	• **número_unidad** Los números de unidad van de –1 a 6, incluido el 0, siendo –1 la TJ1-MC__ y 0 la unidad situada inmediatamente a la derecha de TJ1-MC__.
Ejemplo	N/A
Consulte también	N/A

3.2.127 FRAC

Tipo	Función matemática
Sintaxis	**FRAC(expresión)**
Descripción	La función **FRAC** devuelve la parte fraccionaria de la expresión.
Argumentos	• **expresión** Cualquier expresión BASIC válida.
Ejemplo	**>> PRINT FRAC(1.234)** **0.2340**
Consulte también	N/A

3.2.128 FRAME

Tipo	Parámetro de sistema
Sintaxis	**FRAME = valor**
Descripción	Se utiliza para especificar la trama que se acciona cuando se emplean transformaciones de trama. Las transformaciones de trama se utilizan para permitir que se especifiquen movimiento en una trama de coordenadas multieje de referencia que no corresponda de forma idéntica a los ejes. Un ejemplo lo constituye un brazo robótico SCARA con ejes unidos. Para que el extremo del brazo robótico realice movimientos en línea recta en X-Y, los motores se tienen que mover en un patrón determinado por la geometría del motor. Las transformaciones de trama para realizar funciones de este tipo se tienen que compilar del código fuente en lenguaje C y cargarse en el software del sistema del controlador. Póngase en contacto con OMRON si tiene que realizar esta operación. Se puede especificar un sistema de máquina con varias tramas distintas. La trama de **FRAME** actualmente activa se especifica con el parámetro de sistema **FRAME**. La trama predeterminada de **FRAME** es 0 que corresponde a una transformación uno a uno.
Argumentos	N/A
Ejemplo	**FRAME = 1**
Consulte también	N/A

3.2.129 FREE

Tipo	Función de sistema
Sintaxis	**FREE**
Descripción	La función **FREE** devuelve la cantidad restante de memoria disponible para los programas de usuario y los elementos de matriz TABLE. Nota: cada línea ocupa un mínimo de 4 caracteres (bytes) de memoria. Esto corresponde a la longitud de esta línea, la longitud de la línea anterior, el número de espacios al comienzo de la línea y un solo paso de comando. Los comandos adicionales necesitan un byte por paso; la mayoría de los datos restantes se guardan en ASCII. La TJ1-MC__ compila los programas antes de que se ejecuten. Esto significa que se necesita el doble de memoria para poder ejecutar un programa.
Argumentos	N/A
Ejemplo	**>> PRINT FREE** **47104.0000**
Consulte también	N/A

3.2.130 FS_LIMIT

Tipo	Parámetro de eje
Sintaxis	**FS_LIMIT** **FSLIMIT**
Descripción	El parámetro de eje **FS_LIMIT** contiene la posición absoluta del límite de software de marcha directa en unidades de usuario. Se puede establecer un límite de software para movimiento directo desde el programa para controlar el rango de trabajo de la máquina. Cuando este límite se alcanza, el TJ1-MC__ decelerará hasta 0 y, a continuación, cancelará el movimiento. El bit 9 del parámetro de eje **AXISSTATUS** se pondrá en ON mientras la posición de eje sea mayor que **FS_LIMIT**.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**, **AXISSTATUS**, **UNITS**.

3.2.131 FWD_IN

Tipo	Parámetro de eje
Sintaxis	**FWD_IN**
Descripción	El parámetro de eje **FWD_IN** contiene el número de entrada que se utilizará como entrada de límite de marcha directa. El rango de entrada válido va de 0 a 31. Los valores de 0 a 15 representan entradas presentes físicamente del conector de E/S de la TJ1-MC__ y son comunes para todos los ejes. Los valores 16 a 27 representan entradas de software que se pueden utilizar libremente en programas y comandos, como, por ejemplo, IN y OP. También son comunes para todos los ejes. Los valores 28 a 31 se asignan directamente a las entradas de driver presentes en el conector CN1 y son únicos para cada eje. Las entradas de driver que se asignan a las entradas 28 a 31 dependen del ajuste Pn81E del parámetro de servodriver. El ajuste recomendado es Pn81E = 0 x 4321, con la siguiente asignación: Si se establece un número de entrada y se alcanza el límite, se parará cualquier movimiento directo en dicho eje. También se establecerá el bit 4 de **AXISSTATUS**. Nota: esta entrada se activa por baja.
Sigma II	• entrada 28: CN1-40 • entrada 29: CN1-41 • entrada 30: CN1-42 • entrada 31: CN1-43
Sigma III	• entrada 28: CN1-13 • entrada 29: CN1-7 • entrada 30: CN1-8 • entrada 31: CN1-9
Junma	• entrada 26: CN1-2 • entrada 27: CN1-1 Para obtener más información sobre el ajuste del parámetro de driver Pn81E, consulte el manual del servodriver. De forma predeterminada, el parámetro se establece en –1 y no hay seleccionada ninguna entrada.
Argumentos	N/A
Ejemplo	No hay ejemplo.

Consulte también **AXIS**, **AXISSTATUS REV_IN**.

3.2.132 FWD_JOG

Tipo Parámetro de eje

Sintaxis **FWD_JOG**

Descripción El parámetro de eje **FWD_JOG** contiene el número de entrada que se utilizará como entrada de marcha directa jog. La entrada se puede establecer de 0 a 7. De forma predeterminada, el parámetro se establece en –1 y no hay seleccionada ninguna entrada.
Nota: esta entrada se activa por baja.

Argumentos N/A

Ejemplo No hay ejemplo.

Consulte también **AXIS**, **FAST_JOG**, **JOGSPEED**, **REV_JOG**.

3.2.133 GET

Tipo Comando de E/S

Sintaxis **GET [#n,] variable**

Descripción El comando **GET** asigna el código ASCII de un carácter recibido a una variable. Si el búfer del puerto serie está vacío, la ejecución del programa se pondrá en pausa hasta que se haya recibido un carácter. Los canales 5 a 7 son canales lógicos que están superpuestos en el puerto de programación 0 cuando se utiliza Trajexia Tools.
Nota: el canal 0 está reservado para la conexión a Trajexia Tools y/o la interfaz de línea de comandos. Tenga en cuenta que este canal puede dar problemas para esta función.

Argumentos
- **n**
  El dispositivo de entrada especificado. Cuando se omite este argumento, se utilizará el puerto especificado por **INDEVICE**. Consulte la siguiente tabla.
- **variable**
  El nombre de la variable que recibirá el código ASCII.

Ejemplo **GET#5, k**
Esta línea almacena el carácter ASCII recibido en el canal de puerto 5 de Trajexia Tools en **k**.

Consulte también **INDEVICE INDEVICE**, **INPUT**, **KEY**, **LINPUT**

Número de dispositivo de entrada	**Descripción**
0	Puerto de programación 0
1	Puerto 1 comunicación serie RS-232C
2	Puerto 2 comunicación serie RS-422/485
5	Canal de usuario 5 de puerto 0 de Trajexia Tools
6	Canal de usuario 6 de puerto 0 de Trajexia Tools
7	Canal de usuario 7 de puerto 0 de Trajexia Tools

3.2.134 GLOBAL

Tipo	Comando de sistema
Sintaxis	**GLOBAL "nombre", número_vr**
Descripción	Declara el nombre como una referencia a una de las variables VR globales. Posteriormente, el nombre se puede utilizar desde el programa que contiene la definición **GLOBAL** y todos los demás programas en el proyecto de Trajexia Tools. Nota: el programa que contiene la definición **GLOBAL** se debe ejecutar antes de que el nombre se use en los demás programas. Además, sólo dicho programa se debe ejecutar en el momento en que se ejecuta GLOBAL, de lo contrario, aparecerá el error de programa y el programa se parará cuando intente ejecutar este comando. Para un inicio rápido, el programa también debe ser el único proceso que se ejecuta al inicio. Cuando se declara GLOBAL, la declaración permanece activa hasta el próximo reset de TJ1-MC__ al desconectar y conectar la alimentación o mediante la ejecución del comando EX. En los programas que utilizan el comando **GLOBAL** definido, **nombre** tiene el mismo significado que **VR(número_vr)**. No utilice la sintaxis: **VR(nombre)**. Se puede declarar un máximo de 128 instrucciones **GLOBAL**.
Argumentos	• **nombre** Cualquier nombre definido por el usuario que contiene caracteres alfanuméricos en minúsculas, numéricos o subrayados. • **número_vr** El número de VR que se asociará a **nombre**.
Ejemplo	**GLOBAL "srew_pitch",12** **GLOBAL "ratio1",534** **ratio1 = 3.56** **screw_pitch = 23.0** **PRINT screw_pitch, ratio1**
Consulte también	N/A

3.2.135 GOSUB..RETURN

Tipo	Comando de control de programa
Sintaxis	**GOSUB etiqueta** **...** **RETURN**
Descripción	La estructura **GOSUB** permite saltar a una subrutina. **GOSUB** almacena la posición de la línea después del comando **GOSUB** y, a continuación, va a la etiqueta especificada. Tras llegar a la instrucción **RETURN**, la ejecución del programa vuelve a la posición almacenada. Nota: las subrutinas en cada tarea se pueden anidar hasta 8 niveles.
Argumentos	• **etiqueta** Una etiqueta válida que aparece en el programa. Una etiqueta no válida provocará un error de compilación antes de la ejecución del programa. Las etiquetas pueden ser cadenas de caracteres de cualquier longitud, pero sólo los 15 primeros caracteres son significativos.
Ejemplo	**principal:** **GOSUB rutina** **GOTO principal** **rutina:** **PRINT "Posición medida=";MPOS;CHR(13);** **RETURN**
Consulte también	**GOTO**

3.2.136 GOTO

Tipo	Comando de control de programa
Sintaxis	**GOTO etiqueta**
Descripción	La estructura **GOTO** permite un salto en la ejecución del programa. **GOTO** cambia la ejecución del programa a la línea que contiene la etiqueta.

Argumentos
- **etiqueta**
  Una etiqueta válida que aparece en el programa. Una etiqueta no válida provocará un error de compilación antes de la ejecución del programa. Las etiquetas pueden ser cadenas de caracteres de cualquier longitud, pero sólo los 15 primeros caracteres son significativos.

Ejemplo

```
lazo:
  PRINT "Posición medida = ";MPOS;CHR(13);
  GOTO lazo
```

Consulte también **GOSUB..RETURN**

3.2.137 HALT

Tipo: Comando de sistema

Sintaxis: **HALT**

Descripción: El comando **HALT** para la ejecución de todas las tareas de programa que se están ejecutando actualmente. El comando se puede utilizar tanto en la línea de comandos como en los programas. El comando **STOP** se puede utilizar para parar una sola tarea de programa.

Argumentos: N/A

Ejemplo: No hay ejemplo.

Consulte también: **PROCESS**, **STOP**.

3.2.138 HEX

Tipo: Comando de E/S

Sintaxis: **HEX**

Descripción: Este comando se utiliza en una instrucción de impresión para enviar un número en formato hexadecimal.

Argumentos: N/A

Ejemplo: **PRINT#5,HEX(IN(8,16))**

Consulte también: N/A

3.2.139 HLM_COMMAND

Tipo: Comando de comunicaciones

Sintaxis: **HLM_COMMAND(comando, puerto [, nodo [, área_mc/modo [, offset_mc]]])**

Descripción: El comando **HLM_COMMAND** realiza una operación de comando Host Link específica en uno o en todos los esclavos Host Link del puerto seleccionado. La ejecución del programa se pondrá en pausa hasta que se haya recibido la cadena de respuesta o haya transcurrido el tiempo de espera. El tiempo de espera se especifica mediante el parámetro **HLM_TIMEOUT**. El estado de la transferencia se puede monitorizar con el parámetro **HLM_STATUS**.
Notas:
- Al utilizar **HLM_READ**, asegúrese de configurar el protocolo de maestro Host Link con el comando **SETCOM**.
- Los comandos de maestro Host Link se deben ejecutar desde una tarea de programa únicamente para evitar problemas de temporización multitarea.

Argumentos
- **comando**
  La selección de la operación Host Link que se realizará.
  Consulte la primera tabla más abajo.
- **puerto**
  El puerto serie especificado. 1 = puerto serie RS-232C 1; 2 = puerto serie RS-422A 2.
- **nodo** (para **HLM_MREAD**, **HLM_TEST**, **HLM_ABORT** y **HLM_STWR**)
  El número de nodo esclavo al que se enviará el comando Host Link.
  Rango: [0, 31].
- **modo** (para **HLM_STWR**)
  El modo operativo de CPU especificado. 0 = modo PROGRAM;
  2 = modo MONITOR; 3 = modo RUN.
- **área_mc** (para **HLM_MREAD**)
  La selección de memoria de la TJ1-MC__ de la que se leerán los datos de envío. Consulte la segunda tabla más abajo.
- **offset_mc** (para **HLM_MREAD**)
  La dirección del área de memoria de TJ1-MC__ especificada de la que se realizará la lectura. Rango para variables VR: [0, 1.023]. Rango para variables TABLE: [0, 63.999].

Ejemplo **HLM_COMMAND(HLM_MREAD,1,12,MC_VR,233)**
Este comando lee el código de modelo de CPU del esclavo Host Link con la dirección de nodo 12 conectado al puerto RS-232C. El resultado se escribe en VR(233).
Si el esclavo conectado es un PC C200HX, VR(233) contendrá el valor 12 (hexadecimal) tras una ejecución correcta.

Ejemplo **HLM_COMMAND(HLM_TEST,2,23)**
PRINT HLM_STATUS PORT(2)
Este comando comprobará las comunicaciones Host Link con el esclavo Host Link (nodo 23) conectado al puerto RS-422A.
Si el parámetro **HLM_STATUS** contiene el valor 0, las comunicaciones están operativas.

Ejemplo **HLM_COMMAND(HLM_INIT,2)**
HLM_COMMAND(HLM_ABORT,2,4)
Estos dos comandos realizan las operaciones **INITIALIZE** y **ABORT** de Host Link en el puerto 2 RS-422A. El esclavo tiene el número de nodo 4.

Ejemplo **HLM_COMMAND(HLM_STWR,2,0,2)**
Cuando los dato se tienen que escribir en un PC mediante Host Link, la CPU no puede estar en modo RUN. El comando **HLM_COMMAND** se puede utilizar para definirlo en el modo MONITOR. El esclavo tiene la dirección de nodo 0 y está conectado al puerto RS-232C.

Consulte también **HLM_READ**, **HLM_COMMAND**, **HLM_STATUS**, **HLM_TIMEOUT**, **HLS_NODE**, **HLM_WRITE**, **SETCOM**.

Valor de comando	Descripción
HLM_MREAD (o valor 0)	Realiza el comando **MODEL READ** (**MM**) de PC Host Link para leer el código de modelo de CPU. El resultado se escribe en la variable de TJ1-MC__ especificada por **área_mc** y **offset_mc**.
HLM_TEST (o valor 1)	Realiza el comando **TEST** (**TS**) de Host Link para comprobar las comunicaciones correctas mediante el envío de la cadena “MCW151 TEST STRING” y la comprobación de la cadena devuelta. Compruebe el resultado en el parámetro **HLM_STATUS**.
HLM_ABORT (o valor 2)	Realiza el comando **ABORT** (**XZ**) de Host Link para anular el comando de Host Link que se está procesando actualmente. El comando **ABORT** no recibe una respuesta.
HLM_INIT (o valor 3)	Realiza el comando **INITIALIZE** (******) de Host Link para inicializar el procedimiento de control de transmisión de todas las unidades esclavas.
HLM_STWR (o valor 4)	Realiza el comando **STATUS WRITE** (**SC**) de Host Link para cambiar el modo operativo de la CPU.

Valor de área_mc	Área de datos
MC_TABLE (o valor 8)	Matriz de variables TABLE
MC_VR (o valor 9)	Matriz de variables globales (VR)

3.2.140 HLM_READ

Tipo Comando de comunicaciones

Sintaxis **HLM_READ(puerto, nodo, área_pc, offset_pc, longitud, área_mc, offset_mc)**

Descripción El comando **HLM_READ** lee datos de un esclavo Host Link mediante el envío de una cadena de comando Host Link que contiene el nodo especificado del esclavo al puerto serie. Los datos de respuesta recibidos se escribirán en las variables VR o TABLE. Cada canal de datos se transferirá a una variable. La longitud de datos máxima es de 30 canales (transferencia de trama individual). La ejecución del programa se pondrá en pausa hasta que se haya recibido la cadena de respuesta o haya transcurrido el tiempo de espera. El tiempo de espera se especifica mediante el parámetro **HLM_TIMEOUT**. El estado de la transferencia se puede monitorizar con el parámetro **HLM_STATUS**.
Notas:
- Al utilizar **HLM_READ**, asegúrese de configurar el protocolo de maestro Host Link con el comando **SETCOM**.
- Los comandos de maestro Host Link se deben ejecutar desde una tarea de programa únicamente para evitar problemas de temporización multitarea.

Argumentos
- **puerto**
  El puerto serie especificado. 1 = puerto serie RS-232C 1; 2 = puerto serie RS-422A 2.
- **nodo**
  El número de nodo esclavo al que se enviará el comando Host Link. Rango: [0, 31].
- **área_pc**
  La selección de memoria de PC para el comando de Host Link. Consulte la primera tabla más abajo.
- **offset_pc**
  La dirección del área de memoria de PC especificada de la que se realizará la lectura. Rango: [0, 9.999].
- **longitud**
  El número de canales de datos que se transferirán. Rango: [1, 30].
- **área_mc**
  La selección de memoria de la TJ1-MC__ de la que se leerán los datos de envío. Consulte la segunda tabla más abajo.
- **offset_mc**
  La dirección del área de memoria de TJ1-MC__ especificada en la que se realizará la escritura. Rango para variables VR: [0, 1.023]. Rango para variables TABLE: [0, 63.999].

Ejemplo **HLM_READ(2,17,PLC_DM,120,20,MC_TABLE,4000)**
En este ejemplo se muestra cómo leer 20 canales de las direcciones 120-139 del área DM de PC en las direcciones 4000-4019 de TABLE de la TJ1-MC__. El PC tiene la dirección 17 de nodo esclavo y está conectado al puerto RS-422A.

Consulte también **HLM_COMMAND**, **HLM_STATUS**, **HLM_TIMEOUT**, **HLS_NODE**, **HLM_WRITE**, **SETCOM**.

Valor de área_pc	Área de datos	Comando Host Link
PLC_DM (o valor 0)	Área DM	RD
PLC_IR (o valor 1)	Área CIO/IR	RR
PLC_LR (o valor 2)	Área LR	RL
PLC_HR (o valor 3)	Área HR	RH
PLC_AR (o valor 4)	Área AR	RJ
PLC_EM (o valor 6)	Área EM	RE

Valor de área_mc	Área de datos
MC_TABLE (o valor 8)	Matriz de variables TABLE
MC_VR (o valor 9)	Matriz de variables globales (VR)

3.2.141 HLM_STATUS

Tipo Parámetro de comunicaciones

Sintaxis **HLM_STATUS PORT(n)**

Descripción El parámetro **HLM_STATUS** contiene el estado del último comando de maestro Host Link enviado al puerto especificado. El parámetro indicará el estado de los comandos **HLM_READ**, **HLM_WRITE** y **HLM_COMMAND**. Los bits de estado se definen en la tabla siguiente.
HLM_STATUS contendrá el valor 0 si no se ha producido ningún problema. En el caso de un valor distinto de 0, en el programa BASIC del usuario se tiene que programar una acción adecuada como, por ejemplo, un reintento o una parada de emergencia.
Cada puerto tiene un parámetro **HLM_STATUS**. El modificador **PORT** es necesario para especificar el puerto.

Argumentos
- **n**
  El puerto serie especificado. 1 = puerto 1 serie RS-232C 1; 2 = puerto 2 serie RS-422A

Ejemplo
```
>> HLM_WRITE(1,28,PLC_EM,50,25,MC_VR,200)
>> PRINT HEX(HLM_STATUS PORT(1))
1
```
Aparentemente, la CPU está en modo RUN y no acepta la operación de escritura.

Ejemplo **>> HLM_COMMAND(HLM_TEST,2,0)**
>> PRINT HLM_STATUS PORT(2)
256.0000
Se ha producido un error de tiempo de espera.

Consulte también **HLM_READ**, **HLM_COMMAND**, **HLM_TIMEOUT**, **HLS_NODE**, **HLM_WRITE**, **SETCOM**.

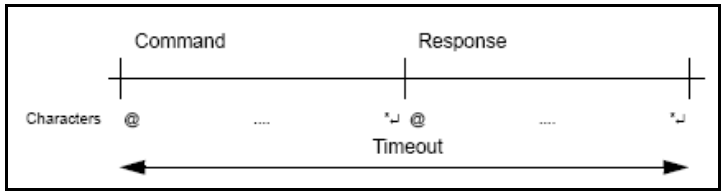
Bit	Nombre	Descripción
0 – 7	Código de fin	El código final puede ser el definido por el esclavo Host Link (problema en la cadena de comando enviada) o un código final definido porque el maestro Host Link ha encontrado un problema (problema en la cadena de respuesta recibida).
8	Error de tiempo de espera	Se producirá un error de tiempo de espera si no se ha recibido ninguna respuesta en el tiempo de espera. Esto indica que se han interrumpido las comunicaciones.
9	Comando no reconocido	Este estado indica que el esclavo no ha reconocido el comando y ha devuelto una respuesta IC.

3.2.142 HLM_TIMEOUT

Tipo Parámetro de comunicaciones

Sintaxis **HLM_TIMEOUT**

Descripción El parámetro **HLM_TIMEOUT** especifica el tiempo de espera fijo para el protocolo de maestro Host Link para ambos puertos serie. Se producirá un error de tiempo de espera cuando el tiempo necesario para enviar el comando y recibir la respuesta del esclavo sea mayor que el tiempo especificado con este parámetro. El parámetro se aplica a los comandos **HLM_READ**, **HLM_WRITE** y **HLM_COMMAND**. El parámetro **HLM_TIMEOUT** se especifica en períodos de servo.



Argumentos N/A

Ejemplo **>> HLM_TIMEOUT = 2000**
Considere que el período de servo de la TJ1-MC__ está definido en 500 ms (SERVO_PERIOD = 500). Para ambos puertos serie, el tiempo de espera de maestro Host Link se ha establecido en 1 s.

Consulte también **HLM_READ**, **HLM_COMMAND**, **HLM_STATUS**, **HLS_NODE**, **HLM_WRITE**, **SETCOM SERVO_PERIOD**.

3.2.143 HLM_WRITE

Tipo: Comando de comunicaciones

Sintaxis: **HLM_WRITE(puerto, nodo, área_pc, offset_pc, longitud, área_mc, offset_mc)**

Descripción: El comando **HLM_WRITE** escribe datos desde la TJ1-MC__ en un esclavo Host Link mediante el envío de una cadena de comando Host Link que contiene el nodo especificado del esclavo al puerto serie. Los datos de respuesta recibidos se escribirán de las variables VR o TABLE. Cada variable definirá el canal o los datos que se transferirán. La longitud de datos máxima es de 29 canales (transferencia de trama individual).
La ejecución del programa se pondrá en pausa hasta que se haya recibido la cadena de respuesta o haya transcurrido el tiempo de espera. El tiempo de espera se especifica mediante el parámetro **HLM_TIMEOUT**. El estado de la transferencia se puede monitorizar con el parámetro **HLM_STATUS**.
Notas:

- Al utilizar **HLM_WRITE**, asegúrese de configurar el protocolo de maestro Host Link con el comando **SETCOM**.
- Los comandos de maestro Host Link se deben ejecutar desde una tarea de programa únicamente para evitar problemas de temporización multitarea.

Argumentos

- **puerto**
  El puerto serie especificado. 1 = puerto 1 serie RS-232C; 2 = puerto 2 serie RS-422A
- **nodo**
  El número de nodo esclavo al que se enviará el comando Host Link. Rango: [0, 31].
- **área_pc**
  La selección de memoria de PC para el comando Host Link. Consulte la primera tabla más abajo.
- **offset_pc**
  La dirección del área de memoria de PC especificada en la que se realizará la escritura. Rango: [0, 9.999].
- **longitud**
  El número de canales de datos que se transferirán. Rango: [1, 29].
- **área_mc**
  La selección de memoria de la TJ1-MC__ de la que se leerán los datos de envío. Consulte la segunda tabla más abajo.
- **offset_mc**
  La dirección del área de memoria de TJ1-MC__ especificada de la que se realizará la lectura. Rango para variables VR: [0, 1.023]. Rango para variables TABLE: [0, 63.999].

Ejemplo: **HLM_WRITE(1,28,PLC_EM,50,25,MC_VR,200)**
En este ejemplo se muestra cómo escribir 25 canales de las direcciones VR 200-224 de la TJ1-MC__ en el área EM de PC 50-74. El PC tiene la dirección 28 de nodo esclavo y está conectado al puerto RS-232C.

Consulte también: **HLM_READ**, **HLM_COMMAND**, **HLM_STATUS**, **HLM_TIMEOUT**, **HLS_NODE**, **SETCOM**.

Valor de área_pc	Área de datos	Comando Host Link
PLC_DM (o valor 0)	Área DM	RD
PLC_IR (o valor 1)	Área CIO/IR	RR
PLC_LR (o valor 2)	Área LR	RL

Valor de área_pc	Área de datos	Comando Host Link
PLC_HR (o valor 3)	Área HR	RH
PLC_AR (o valor 4)	Área AR	RJ
PLC_EM (o valor 6)	Área EM	RE

Valor de área_mc	Área de datos
MC_TABLE (o valor 8)	Matriz de variables de tabla
MC_VR (o valor 9)	Matriz de variables globales (VR)

3.2.144 HLS_NODE

Tipo: Parámetro de comunicaciones

Sintaxis: **HLS_NODE**

Descripción: El parámetro **HLS_NODE** define el número de unidad de esclavo para el protocolo de esclavo Host Link. La TJ1-MC__ sólo responderá a las cadenas de comando de maestro Host Link con el número de unidad según lo especificado en este parámetro. El rango válido para este parámetro es [0, 31]. El valor predeterminado es 0.

Argumentos: N/A

Ejemplo: No hay ejemplo.

Consulte también: N/A

3.2.145 HW_PSWITCH

Tipo: Comando de eje

Sintaxis: **HW_PSWITCH(modo, dirección, estado_salida, inicio_tabla, fin_tabla)**

Descripción: El comando **HW_PSWITCH** activa la salida OUT 0 para el eje cuando se alcanza la posición medida de eje predefinido y desactiva la salida cuando se alcanza otra posición medida. Las posiciones se definen como una secuencia en la memoria TABLE en el rango de **inicio_tabla** a **fin_tabla** y al ejecutarse el comando **HW_PSWITCH** se almacenan en la cola FIFO.
Este comando sólo es aplicable a los ejes flexibles con los valores 43, 44 y 45 de **ATYPE**.
El comando se puede utilizar con uno o 5 parámetros. Sólo se necesita un parámetro para desactivar el interruptor o borrar la cola FIFO. Se necesitan los cinco parámetros para activar el interruptor.
Después de cargar la cola FIFO y recorrer la secuencia de posiciones de ella, si se tiene que ejecutar de nuevo la misma secuencia, se debe borrar la cola FIFO antes de ejecutar el comando **HW_PSWITCH** con los mismos parámetros.

Argumentos:
- **modo**
  0 = desactivar interruptor; 1 = activar y cargar FIFO; 2 = borrar FIFO.
- **dirección**
  0 = descendiendo; 1 = ascendiendo.
- **estado_salida**
  Estado de salida para establecer en la primera posición de la cola FIFO: ON u OFF.
- **inicio_tabla**
  Dirección de inicio de la secuencia de table.
- **fin_tabla**
  Dirección de finalización de la secuencia de table.

Ejemplo: **HW_PSWITCH(1, 1, ON, 21, 50)**
Este comando cargará la tabla FIFO con 30 posiciones, almacenas en la memoria TABLE a partir de **TABLE(21)** en dirección ascendente. Cuando se alcanza la posición almacenada en **TABLE(21)**, la salida OUT 0 se pondrá en ON y, alternativamente, en OFF y en ON al alcanzar las posiciones siguientes de la secuencia, hasta llegar a la posición almacenada en **TABLE(50)**.

Ejemplo: **HW_PSWITCH(0)**
Este comando desactivará el interruptor si se ha activado anteriormente, pero no borrará la cola FIFO.

Ejemplo	**HW_PSWITCH(2)** Este comando borrará la cola FIFO si se ha cargado anteriormente.
Consulte también	**AXIS**

3.2.146 I_GAIN

Tipo	Parámetro de eje
Sintaxis	**I_GAIN**
Descripción	El parámetro de eje **I_GAIN** contiene la ganancia integral del eje. La aportación a la salida de la integral se calcula multiplicando las sumas de los errores de seguimiento con el valor del parámetro **I_GAIN**. El valor predeterminado es 0. Al incorporar la ganancia integral a un servosistema se reducen los errores de posición en reposo y al moverse continuamente. Puede producir o aumentar los sobreimpulsos y las oscilaciones y, por lo tanto, sólo resulta adecuada para sistemas que funcionen a velocidad constante y con aceleraciones lentas. Nota: para evitar inestabilidad, las ganancias de servo sólo se deben cambiar cuando **SERVO** está en OFF.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**D_GAIN**, **I_GAIN**, **OV_GAIN**, **P_GAIN**, **VFF_GAIN**.

3.2.147 IDLE

Consulte **WAIT IDLE**.

3.2.148 IEEE_IN

Tipo	Función matemática
Sintaxis	**IEEE_IN(byte0,byte1,byte2,byte3)**
Descripción	La función **IEEE_IN** devuelve el número en coma flotante representado por 4 bytes que normalmente se han recibido a través de comunicaciones.
Argumentos	• **byte0** – byte3 Cualquier combinación de valores de 8 bits que represente un número de coma flotante IEEE válido.
Ejemplo	**VR(20) = IEEE_IN(b0,b1,b2,b3)**
Consulte también	N/A

3.2.149 IEEE_OUT

Tipo	Función matemática
Sintaxis	**byte_n = IEEE_OUT(valor, n)**
Descripción	La función **IEEE_OUT** devuelve un solo byte en formato IEEE extraído de un valor en coma flotante para la transmisión a través de un sistema de bus. La función normalmente se llamará 4 veces para extraer cada byte de uno en uno. Nota: El byte 0 es el byte alto del formato en coma flotante IEEE de 32 bits.
Argumentos	• **valor** Cualquier variable o parámetro de coma flotante BASIC. • **n** El número de byte (0 – 3) que se extraerá.
Ejemplo	**V = MPOS AXIS(2)** **byte0 = IEEE_OUT(V, 0)** **byte1 = IEEE_OUT(V, 1)** **byte2 = IEEE_OUT(V, 2)** **byte3 = IEEE_OUT(V, 3)**
Consulte también	N/A

3.2.150 IF..THEN..ELSE..ENDIF

Tipo: Comando de control de programa

Sintaxis:
IF condición_1 THEN comandos {ELSEIF condición_i THEN comandos}
[ELSE comandos] ENDIF
IF condición_1 THEN comandos

Descripción: Esta estructura controla el flujo del programa según los resultados de la condición. Si la condición es **TRUE** se ejecutan los comandos que siguen a **THEN** hasta **ELSEIF**, **ELSE** o **ENDIF**. Si la condición es **FALSE** y el comando de una subestructura **ELSEIF** posterior es **TRUE**, se ejecutan los comandos de dicha subestructura. Si todas las condiciones son **FALSE**, se ejecutarán los comandos que siguen a **ELSE** o el programa se reanudará en la línea después de **ENDIF** si no se incluye ningún **ELSE**. **ENDIF** se utiliza para marcar el final del bloque condicional.
Nota: las secuencias **IF..THEN..ELSE..ENDIF** se pueden anidar sin ningún tipo de límite. Para una construcción **IF..THEN** de varias líneas, no debe haber ninguna instrucción después de **THEN**. Una construcción de una sola línea no debe utilizar **ENDIF**.

Argumentos:
- **condición_i**
  Una expresión lógica.
- **comandos**
  Uno o varios comandos BASIC.

Ejemplo:
```
IF MPOS > (0.22 * VR(0)) THEN GOTO longitud_superada
```

Ejemplo:
```
IF IN(0) = ON THEN
  contaje = contaje + 1
  PRINT "CONTAJES = ";contaje
  fallo = 0
ELSE
  fallo = fallo + 1
ENDIF
```

Ejemplo:
```
IF IN(parar) = ON THEN
  OP(8,ON)
  VR(indicador_ciclo) = 0
ELSEIF IN(inicio_ciclo) = ON THEN
  VR(indicador_ciclo) = 1
ELSEIF IN(paso1) = ON THEN
  VR(indicador_ciclo) = 99
ENDIF
```

Ejemplo:
```
IF car_tecla = $31 THEN
  GOSUB car_1
ELSEIF car_tecla = $32 THEN
  GOSUB car_2
ELSEIF car_tecla = $33 THEN
  GOSUB car_3
ELSE
  PRINT "Carácter desconocido"
ENDIF
```

Consulte también: N/A

3.2.151 IN

Tipo: Función de E/S

Sintaxis:
IN(número_entrada [, número_entrada_final])
IN

Descripción: La función **IN** devuelve el valor de las entradas digitales.
- **IN(número_entrada, número_entrada_final)** devolverá la suma binaria del grupo de entradas. Los dos argumentos deben ser menores que 24.
- **IN(número_entrada)** con un valor para **número_entrada** menor que 32 devolverá el valor del canal concreto.
- **IN** (sin argumentos) devolverá la suma binaria de las primeras 24 entradas (como en **IN(0,23)**).

Argumentos:
- **número_entrada**
- **El número de la entrada para la que se devolverá un valor. Valor: un entero entre 0 y 31.**
- **número_entrada_final**
  El número de la última entrada para la que se devolverá un valor. Valor: un entero entre 0 y 31.

Ejemplo Se pueden utilizar las siguientes líneas para mover la posición establecida en un thumb wheel multiplicada por un factor. El thumb wheel está conectado a las entradas 4, 5, 6 y 7, y ofrece la salida en BCD.

```
lazomovimiento:
MOVEABS(IN(4,7) * 1.5467)
WAIT IDLE
GOTO lazomovimiento
```

El comando **MOVEABS** se construye del siguiente modo:
Paso 1: **IN(4,7)** obtendrá un número entre 0 y 15.
Paso 2: El número se multiplica por 1,5467 para obtener la distancia necesaria.
Paso 3: Se realiza un movimiento absoluto a esta posición.

Ejemplo En este ejemplo se prueba una sola entrada:

```
prueba:
  WAIT UNTIL IN(4) = ON   'La cinta transportadora está en posición
                          'cuando se encuentra en ON
  GOSUB lugar
```

Consulte también **OP**.

3.2.152 INDEVICE

Tipo Parámetro de E/S

Sintaxis **INDEVICE**

Descripción El parámetro **INDEVICE** define el dispositivo de entrada predeterminado. Este dispositivo se seleccionará para los comandos de entrada cuando se omita la opción **#n**. El parámetro **INDEVICE** es específico de la tarea. Los valores admitidos se enumeran en la tabla siguiente.

Argumentos N/A

Ejemplo No hay ejemplo.

Consulte también **GETGET**, **INPUT**, **LINPUT**, **KEY**.

Valor	Descripción
0	Puerto de programación 0 (predeterminado)
1	Puerto 1 comunicacion serie RS-232C
2	Puerto 2 comunicación serie RS-422A/485
5	Canal de usuario 5 de puerto 0 de Trajexia Tools
6	Canal de usuario 6 de puerto 0 de Trajexia Tools
7	Canal de usuario 7 de puerto 0 de Trajexia Tools

3.2.153 INITIALISE

Tipo Comando de sistema

Sintaxis **INITIALISE**

Descripción Establece todos los ejes y los parámetros de sistema y proceso a sus valores predeterminados. Los parámetros también se restablecen cada vez que se enciende el controlador o cuando se ejecuta un comando **EX** (reset por software). En Trajexia Tools, el menú **Reset the controller...** (Restablecer controlador), en el menú **Controller** (Controlador), realiza la función equivalente al comando **EX**.

Argumentos N/A

Ejemplo No hay ejemplo.

Consulte también
- **EX**

3.2.154 INPUT

Tipo	Comando de E/S
Sintaxis	**INPUT [#n], variable {, variable }**
Descripción	El comando **INPUT** asignará valores de cadena de entrada numérica a las variables especificadas. Se pueden solicitar varios valores de cadena de entrada en una sola línea separados por comas o en varias líneas separados por retorno de carro. La ejecución del programa estará en pausa hasta que se termine la cadena con un retorno de carro después de que se haya asignado la última variable. Si la cadena no es válida, al usuario se le mostrará un mensaje de error y se repetirá la tarea. La cantidad máxima de entradas en una sola línea no tiene más límite que la longitud de línea. Los canales 5 a 7 son canales lógicos que están superpuestos en el puerto 0 de programación RS-232C cuando se utiliza Trajexia Tools. Nota: el canal 0 está reservado para la conexión a Trajexia Tools y/o la interfaz de línea de comandos. Tenga en cuenta que este canal puede dar problemas para esta función.
Argumentos	• **n** El dispositivo de entrada especificado. Cuando se omite este argumento, se utilizará el puerto especificado por INDEVICE. • **variable** La variable en la que se escribirá.
Ejemplo	Considere el siguiente programa para recibir datos del terminal. **INPUT#5, num** **PRINT#5, "CONTAJE DE LOTES=";num[0]** Una respuesta posible en el terminal podría ser: **123<CR>** **CONTAJE DE LOTES = 123**
Consulte también	**INDEVICE**, **GET**, **LINPUT**, **KEY**

3.2.155 INT

Tipo	Función matemática
Sintaxis	**INT(expresión)**
Descripción	La función **INT** devuelve la parte entera de la expresión. Nota: Para redondear un número positivo al valor entero más próximo, tome la función **INT** del valor más 0,5. Del mismo modo, para redondear un valor negativo, reste 0,5 al valor antes de aplicar **INT**.
Argumentos	• **expresión** Cualquier expresión BASIC válida.
Ejemplo	**>> PRINT INT(1.79)** **1.0000**
Consulte también	N/A

3.2.156 INVERT_IN

Tipo	Comando de sistema
Sintaxis	**INVERT_IN(entrada,on/off)**
Descripción	El comando **INVERT_IN** permite que los canales 0 a 31 se inviertan individualmente en software. Esto resulta importante ya que estos canales de entrada se pueden asignar para activar funciones como, por ejemplo, la retención de realimentación. La función **INVERT_IN** establece la inversión para un canal en ON u OFF. Sólo se puede aplicar a las entradas 0 a 31.
Argumentos	• **de reset** Una expresión BASIC.
Ejemplo	**>>? IN(3)** **0.0000** **>>INVERT_IN(3,ON)** **>>? IN(3)** **1.0000**
Consulte también	N/A

3.2.157 INVERT_STEP

Tipo	Parámetro de eje
Sintaxis	**INVERT_STEP**
Descripción	**INVERT_STEP** se utiliza para invertir la lógica en el circuito de salida para motores paso a paso. Esto puede ser necesario para conectar algunos drivers paso a paso. La lógica electrónica en el generador de impulsos paso a paso de Trajexia supone que el flanco descendente de la salida de paso es el flanco activo que da como resultado el movimiento del motor. Esto resulta adecuado para la mayoría de los controladores paso a paso. Al establecer **INVERT_STEP = ON** se invierte la lógica de forma que el flanco ascendente de la señal de paso se tomará como flanco activo. **INVERT_STEP** se debe establecer, si es necesario, antes de activar el controlador con **WDOG = ON**. El valor predeterminado es OFF. Nota: si el ajuste es incorrecto, un motor paso a paso puede perder la posición en un paso al cambiar de dirección.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	N/A

3.2.158 INVERTER_COMMAND

Tipo	Comando de sistema
Sintaxis	**INVERTER_COMMAND(módulo, estación, 7, señales_operación)** **INVERTER_COMMAND(módulo, estación, 1, número_alarma)**
Descripción	**INVERTER_COMMAND** controla las entradas y borra la alarma del variador de frecuencia conectado al sistema mediante el bus MECHATROLINK-II. Existen dos funciones de **INVERTER_COMMAND**: • 1: Borra una alarma. • 7: Controla las señales de operación. Para usar un variador mediante MECHATROLINK-II, debe configurar el comando de marcha y de referencia del variador en la opción de comunicaciones: • Variador MV/V7: N3 = 3; N4 = 9 • Variador F7/G7: B1 – 01 = 3; B1 – 02 = 3. Asegúrese de que el firmware del variador admite la tarjeta de comunicaciones MECHATROLINK-II. El comando devuelve –1 si se ha ejecutado correctamente y 0 si ha fallado. El comando enviado al variador corresponderá a los bits indicados en la tabla siguiente.
Argumentos	• **módulo** El número de la TJ1-ML__ a la que está conectado el variador. • **estación** El número de estación MECHATROLINK-II del variador. • **número_alarma** El número de la alarma. Consulte el manual del variador. • **señales_operación** Un valor de operación (Trajexia internamente lo tratará a nivel de bits) para controlar las señales de operación. Consulte la siguiente tabla.
Ejemplo	No hay ejemplo.
Consulte también	N/A

Bit	Valor	Comando	Descripción
0	Hexadecimal	1	Marcha directa
1	Hexadecimal	2	Marcha inversa
2	Hexadecimal	4	Entrada multifunción 3 de variador
3	Hexadecimal	8	Entrada multifunción 4 de variador
4	Hexadecimal	10	Entrada multifunción 5 de variador
5	Hexadecimal	20	Entrada multifunción 6 de variador
6	Hexadecimal	40	Entrada multifunción 7 de variador
7	Hexadecimal	80	Entrada multifunción 8 de variador (sólo G7)
8	Hexadecimal	100	Fallo externo
9	Hexadecimal	200	Reset de fallo
10	Hexadecimal	400	Entrada multifunción 9 de variador (sólo G7)
11	Hexadecimal	800	Entrada multifunción 10 de variador (sólo G7)
12	Hexadecimal	1.000	Entrada multifunción 11 de variador (sólo G7)
13	Hexadecimal	2.000	Entrada multifunción 12 de variador (sólo G7)
14	Hexadecimal	4.000	Borrado de datos del historial de fallos
15	Hexadecimal	8.000	Comando BB externo

3.2.159 INVERTER_READ

Tipo: Comando de sistema

Sintaxis:
INVERTER_READ(módulo, estación,0, número_parámetro, tamaño_parámetro, VR)
INVERTER_READ(módulo, estación, 1, número_alarma, VR)
INVERTER_READ(módulo, estación, 2, VR)
INVERTER_READ(módulo, estación, 3, VR)
INVERTER_READ(módulo, estación, 4, desde, longitud, VR)

Descripción: **INVERTER_READ** lee el parámetro, la referencia de velocidad, la referencia de par o la alarma del variador de frecuencia conectado al sistema mediante el bus MECHATROLINK-II.
Existen cinco funciones de **INVERTER_READ**:

- 0: lee un parámetro de variador.
- 1: lee una alarma de variador.
- 2: lee la referencia de velocidad.
- 3: lee la referencia de par.
- 4: lee las entradas de variador.

Para usar un variador mediante MECHATROLINK-II, debe configurar el comando de marcha y de referencia del variador en la opción de comunicaciones:

- Variador MV/V7: N3 = 3; N4 = 9
- Variador F7/G7: B1 – 01 = 3; B1 – 02 = 3

Asegúrese de que el firmware del variador admite la tarjeta de comunicaciones MECHATROLINK-II.
El comando devuelve 1 si se ha ejecutado correctamente y 0 si ha fallado.
El resultado, si lo hubiera, se devuelve en la variable VR seleccionada.

Argumentos
- **módulo**
  El número de la TJ1-ML__ a la que está conectado el variador.
- **estación**
  El número de estación MECHATROLINK-II del variador.
- **número_parámetro**
  El número del parámetro que se leerá. Consulte el manual del variador.
- **tamaño_parámetro**
  El tamaño del parámetro que se leerá, 2 ó 4 bytes. Consulte el manual del variador.
- **VR**
  La dirección de la memoria VR de la TJ-MC__ donde se coloca la información leída. Cuando la función es **4**, el resultado se devuelve como un valor de operación de bits. Consulte la siguiente tabla.
- **número_alarma**
  El número de la alarma que se leerá. Consulte el manual del variador.
- **desde**
  La dirección inicial de la entrada que se leerá.
- **longitud**
  La longitud de la entrada que se leerá.

Ejemplo: No hay ejemplo.

Consulte también: N/A

Bit	Valor	Comando	Descripción
0	Hexadecimal	1	Marcha directa
1	Hexadecimal	2	Marcha inversa
2	Hexadecimal	4	Entrada multifunción 3 de variador
3	Hexadecimal	8	Entrada multifunción 4 de variador
4	Hexadecimal	10	Entrada multifunción 5 de variador
5	Hexadecimal	20	Entrada multifunción 6 de variador
6	Hexadecimal	40	Entrada multifunción 7 de variador
8	Hexadecimal	100	Fallo externo
9	Hexadecimal	200	Reset de fallo
14	Hexadecimal	4.000	Borrado de datos del historial de fallos
15	Hexadecimal	8.000	Comando BB externo

3.2.160 INVERTER_WRITE

Tipo: Comando de sistema

Sintaxis:
INVERTER_WRITE(módulo, estación, 0, número_parámetro, tamaño_parámetro, VR, modo)
INVERTER_WRITE(módulo, estación, 2, valor)
INVERTER_WRITE(módulo, estación, 3, valor)

Descripción: **INVERTER_WRITE** escribe el parámetro, la referencia de velocidad o la referencia de par del variador de frecuencia conectado al sistema mediante el bus MECHATROLINK-II.
Existen tres funciones de **INVERTER_WRITE**:
- 0: escribe un parámetro de variador.
- 2: escribe la referencia de velocidad.
- 3: escribe la referencia de par.

Para usar un variador mediante MECHATROLINK-II, debe configurar el comando de marcha y de referencia del variador en la opción de comunicaciones:
- Variador MV/V7: N3 = 3; N4 = 9
- Variador F7/G7: B1 – 01 = 3; B1 – 02 = 3

Asegúrese de que el firmware del variador admite la tarjeta de comunicaciones MECHATROLINK-II.
El comando devuelve –1 si se ha ejecutado correctamente y 0 si ha fallado.
El resultado, si lo hubiera, se devuelve en la variable VR seleccionada.

Argumentos
- **módulo**
  El número de la TJ1-ML__ a la que está conectado el variador.
- **estación**
  El número de estación MECHATROLINK-II del variador.
- **número_parámetro**
  El número del parámetro que se escribirá.
  Consulte el manual del variador.
- **tamaño_parámetro**
  El tamaño del parámetro que se escribirá, 2 ó 4 bytes.
  Consulte el manual del variador.
- **VR**
  La dirección en la memoria VR de la TJ1-MC__ donde se encuentra el nuevo valor del parámetro.
- **modo**
  0 = sólo escribir; 1= escribir e intro; 2 = escribir y configurar.
- **valor**
  El nuevo valor que está escrito.

Ejemplo: No hay ejemplo.

Consulte también: N/A

Si tiene que transferir numerosos parámetros a la vez, la forma más eficaz consiste en usar el MODO 0 para todos los parámetros menos el último y MODO 1 para este último.
MODO 0 se ejecuta más rápido que MODO 1.

3.2.161 JOGSPEED

Tipo: Parámetro de eje

Sintaxis: **JOGSPEED**

Descripción: El parámetro **JOGSPEED** establece la velocidad jog en unidades de usuario para un eje. Se ejecutará una operación jog cuando se haya declarado una entrada jog para un eje y dicha entrada esté baja. Hay disponibles una entrada jog de marcha directa y una entrada jog de marcha inversa para cada eje si se establece **FWD_JOG** y **REV_JOG** respectivamente. La velocidad de la operación jog se puede controlar con la entrada **FAST_JOG**.

Argumentos: N/A

Ejemplo: No hay ejemplo.

Consulte también: **AXIS AXIS**, **FAST_JOG**, **FWD_JOG**, **REV_JOG**, **UNITS**.

3.2.162 KEY

Tipo: Parámetro de E/S

Sintaxis: **KEY [#n]**

Descripción: El parámetro **KEY** devuelve **TRUE** o **FALSE** en función de si se ha recibido un carácter en el búfer de puerto serie o no se ha recibido. El resultado de **TRUE** será reseteado cuando el carácter se lea con el comando **GET**.
Los canales 5 a 7 son canales lógicos que están superpuestos en el puerto de programación 0 cuando se utiliza Trajexia Tools.
Nota: el canal 0 está reservado para la conexión a Trajexia Tools y/o la interfaz de línea de comandos. Tenga en cuenta que este canal puede dar problemas para esta función.

Argumentos
- **n**
  El dispositivo de entrada especificado. Cuando se omite este argumento, se utilizará el puerto especificado por **INDEVICE**. Consulte la siguiente tabla.

Ejemplo:
WAIT UNTIL KEY#1
GET#1, k
Tenga en cuenta que al utilizar **KEY#1** en una ecuación puede requerir paréntesis en la instrucción; en este caso: **WAIT UNTIL (KEY#1) = TRUE**.

Consulte también
- **GET**

Valor	Dispositivo de entrada
0	Puerto de programación 0
1	Puerto 1 comunicacion serie RS-232C
2	Puerto 2 comunicación serie RS-422A/485
5	Trajexia Tools puerto 0 Canal de usuario 5
6	Trajexia Tools puerto 0 Canal de usuario 6
7	Trajexia Tools puerto 0 Canal de usuario 7

3.2.163 LAST_AXIS

Tipo: Parámetro de sistema

Sintaxis: **LAST_AXIS** (sólo lectura)

Descripción: El parámetro **LAST_AXIS** contiene el número del último eje que ha procesado el sistema.
La mayoría de los sistemas no utilizan todos los ejes disponibles. Por lo tanto, se perdería tiempo en programar la tarea de los movimientos inactivos en todos los ejes que no están en uso. Para evitar esta situación de alguna forma, la TJ1-MC__ programará la tarea de los movimientos en los ejes de 0 a **LAST_AXIS**, donde **LAST_AXIS** es el número del eje más alto para el que se ha procesado un comando **AXIS** o **BASE**, el que sea mayor de los dos.

Argumentos: N/A

Ejemplo: No hay ejemplo.

Consulte también: **AXIS**, **BASE**.

3.2.164 LINKAX

Tipo: Parámetro de eje (sólo lectura)

Sintaxis: **LINKAX**

Descripción: Devuelve el número del eje al que está vinculado el eje durante los movimientos vinculados. Los movimientos vinculados son en los que la posición solicitada es en funcion de la posición de otro eje.

Argumentos: N/A

Ejemplo: No hay ejemplo.

Consulte también: **CONNECT**, **CAMBOX**, **MOVELINK**.

3.2.165 LINPUT

Tipo: Comando de E/S

Sintaxis: **LINPUT [#n,] variable_vr**

Descripción: El comando **LINPUT** asigna el código ASCII de los caracteres a una matriz de variables empezando con la variable VR especificada. La ejecución del programa estará en pausa hasta que se termine la cadena con un retorno de carro, que también se almacena. El controlador no repite la cadena.
Los canales 5 a 7 son canales lógicos que están superpuestos en el puerto de programación 0 cuando se utiliza Trajexia Tools.
Nota: el canal 0 está reservado para la conexión a Trajexia Tools y/o la interfaz de línea de comandos. Tenga en cuenta que este canal puede dar problemas para este comando.

Argumentos:
- **N**
  El dispositivo de entrada especificado. Cuando se omite este argumento, se utilizará el puerto especificado por INDEVICE. Consulte la siguiente tabla.
- **variable_vr**
  La primera variable VR en la que se escribirá.

Ejemplo Considere la siguiente línea de un programa.
LINPUT#5, VR(0)
Al introducir START<CR> se obtendrá
VR(0) = 83S
VR(1) = 84T
VR(2) = 65A
VR(3) = 82R
VR(4) = 84T
VR(5) = 13<CR>

Consulte también
- **GET**
- **INPUT**
- **VR**

Valor	Dispositivo de entrada
0	Puerto 0 de programación
1	Puerto 1 comunicación serie RS-232C
2	Puerto 2 comunicacion serie RS-422A/485
5	Canal de usuario 5 de puerto 0 de Trajexia Tools
6	Canal de usuario 6 de puerto 0 de Trajexia Tools
7	Canal de usuario 7 de puerto 0 de Trajexia Tools

3.2.166 LIST

Tipo Comando de programa (sólo línea de comandos de Trajexia Tools)

Sintaxis **LIST ["nombre_programa"]**
TYPE ["nombre_programa"]

Descripción Usar sólo con la ventana de terminal. **LIST** se utiliza como un comando inmediato (línea de comandos) únicamente y no se debe utilizar en programas. El comando **LIST** imprime el programa seleccionado actual o el programa especificado por **nombre_programa**. El nombre de programa también se puede especificar sin comillas. Si se omite el nombre de programa, se enumerará el programa seleccionado actual.
Nota: Este comando se ha implementado para un terminal (VT100) offline. Con Trajexia Tools los usuarios pueden utilizar la ventana de terminal.

Argumentos
- **nombre_programa**
  El programa que se imprimirá.

Ejemplo No hay ejemplo.

Consulte también **SELECT**.

3.2.167 LIST_GLOBAL

Tipo Comando de sistema (sólo terminal)

Sintaxis **LIST_GLOBAL**

Descripción Cuando se ejecuta desde la línea de comandos (canal de terminal 0) devuelve todos los parámetros **GLOBAL** y **CONSTANT** seleccionados actualmente.

Argumentos N/A

Ejemplo En una aplicación donde se han configurado los siguientes valores de GLOBAL y CONSTANT:

CONSTANT "cortadora", 23
GLOBAL "cinta_transportadora",5

```
>>LIST_GLOBAL
 Global VR
 ---------------- ----
 cinta_transportadora 5
 Constant Value
 ---------------- -------
 cortadora 23.0000
```

Consulte también N/A

3.2.168 LN

Tipo Función matemática

Sintaxis **LN(expresión)**

Descripción La función **LN** devuelve el logaritmo natural de la expresión. El valor de la expresión de entrada debe ser mayor que 0.

Argumentos
- **expresión**
  Cualquier expresión BASIC válida.

Ejemplo **>> PRINT LN(10)**
2.3026

Consulte también N/A

3.2.169 LOCK

Tipo Comando de sistema

Sintaxis **LOCK(código)**
UNLOCK(código)

Descripción El comando **LOCK** impide que se visualice, modifique o elimine el programa por parte de personal que no conozca el código de seguridad. El comando **UNLOCK** permite desbloquear el estado bloqueado. El número de código puede ser cualquier entero y se almacena en formato codificado. **LOCK** es siempre un comando inmediato y sólo se puede emitir cuando el estado del sistema es **UNLOCKED**.

Argumentos
- **código**
  Cualquier entero con 7 dígitos como máximo.

Ejemplo **>> LOCK(561234)**
Los programas no se pueden modificar ni visualizar.
>> UNLOCK(561234)
El sistema ahora está desbloqueado.

Consulte también N/A

Precaución
Se debe recordar el código de seguridad, ya que será necesario para desbloquear el sistema. Sin el código de seguridad, el sistema no se puede recuperar.

3.2.170 MARK

Tipo Parámetro de eje (sólo lectura)

Sintaxis **MARK**

Descripción El parámetro **MARK** se establece en **FALSE** cuando se ha ejecutado el comando **REGIST** y se establece en **TRUE** cuando se produce el evento de registro principal.

Argumentos N/A

Ejemplo
IF MARK AXIS(1) THEN
PRINT "Se ha producido el evento de registro principal para el eje 1"
ENDIF

Consulte también **AXIS**, **REGIST**, **REG_POS**.

3.2.171 MARKB

Tipo Parámetro de eje (sólo lectura)

Sintaxis **MARKB**

Descripción El parámetro **MARKB** se establece en **FALSE** cuando se ha ejecutado el comando **REGIST** y se establece en **TRUE** cuando se produce el evento de registro secundario.

Argumentos N/A

Ejemplo
IF MARKB AXIS(2) THEN
PRINT "Se ha producido el evento de registro secundario para el eje 2"
ENDIF

Consulte también **AXIS**, **REGIST**, **REG_POSB**.

3.2.172 MECHATROLINK

Tipo Comando de sistema

Sintaxis **MECHATROLINK(unidad,0)**
Detecta y conecta dispositivos en la unidad MECHATROLINK-II **unidad**. Es necesario utilizarlo para realizar un reset de la red por un problema de comunicaciones y para volver a detectar servos que no se han detectado (por ejemplo, cuando la letra A de la palabra **AXISSTATUS** está en mayúsculas de color rojo).
MECHATROLINK(unidad,3,VR)
Devuelve el número de los dispositivos detectados después de un comando **MECHATROLINK(unidad,0)**. Lo utiliza el programa **STARTUP** para comprobar que el número de módulos MECHATROLINK-II detectados corresponde al previsto.
MECHATROLINK(unidad,4,estación,VR)
Devuelve la dirección del dispositivo MECHATROLINK-II en el número de “estación”. Los números de estación se encuentran en una secuencia de 0 a x para todos los dispositivos conectados. Se devuelve –1 si no hay asignado ningún dispositivo a dicha estación. Lo utiliza el programa **STARTUP** para comprobar que el número de módulos MECHATROLINK-II detectados corresponde al previsto.
MECHATROLINK(unidad,5,estación,VR)
Lee y borra el contaje de mensajes perdidos. Un dispositivo MECHATROLINK-II que no sea eje no informa automáticamente de un problema de red, por lo que puede utilizar este comando para sondear el variador y los módulos de E/S con el fin de comprobar que la red está activa. Este programa se utilizará en el programa SHELL propuesto.
Nota:

- Puede utilizar el comando **MECHATROLINK(unidad,5,estación,VR)** para monitorizar el estado de un dispositivo durante la ejecución de un programa. Si el contenido de la dirección VR es mayor que 0, se produce un error de comunicaciones con el dispositivo y éste puede funcionar incorrectamente. Puede utilizar este comando para parar el programa cuando el dispositivo tenga un error.

Descripción Nota: Este comando tiene dos formatos, según la función que se requiera: funciones de maestro y esclavo.
Todas las funciones **MECHATROLINK** devuelven **TRUE** (–1) si el comando se ha realizado correctamente o **FALSE** (0) si ha fallado el comando.
Las funciones se dividen en 2 tipos, funciones **MASTER** que actúan en una unidad y funciones **STATION** que actúan en una **dirección_estación** específica de una unidad concreta.
Todas las funciones que recuperan un valor lo almacenan en la variable **VR** indicada en el último parámetro. Si este parámetro tiene el valor –1, se imprime en el puerto de línea de comandos.
Notas:

- Si falla un comando MECHATROLINK-II, la estación MECHATROLINK-II pasará al estado de advertencia/alarma. Todos los comandos posteriores volverán a este estado de advertencia/alarma, aunque el comando en cuestión se ejecute correctamente.
- El estado de advertencia/alarma sólo se puede borrar mediante el comando **ALM_CLR**.
- No hay ningún subcomando **ALM_CLR**, por lo que, para enviar **ALM_CLR**, debe acceder al modo commissioning.

Argumentos N/A

Ejemplo No hay ejemplo.

Consulte también N/A

3.2.173 MERGE

Tipo Parámetro de eje

Sintaxis **MERGE**

Descripción El parámetro **MERGE** es un interruptor software que se puede utilizar para activar o desactivar la combinación de movimientos consecutivos.
Si **MERGE** está en ON y el movimiento siguiente ya está en el búfer de movimiento siguiente (**NTYPE**), el eje no decelerará de rampa a velocidad 0, sino que cargar el movimiento siguiente, lo que permite una combinación perfecta. El ajuste predeterminado de **MERGE** es OFF.
Es responsabilidad del programador garantizar que la combinación es razonable. Por ejemplo, la combinación de un movimiento de marcha directa con otro de marcha inversa provocará un intento instantáneo de cambio de dirección.
MERGE sólo funcionará si se cumplen las siguientes condiciones:

1. Sólo los movimientos con perfil de velocidad **MOVE**, **MOVEABS**, **MOVECIRC**, **MHELICAL**, **REVERSE**, **FORWARD** y **MOVEMODIFY** se pueden combinar entre sí. No se pueden combinar con movimientos vinculados **CONNECT**, **MOVELINK** y **CAMBOX**.
2. Hay un movimiento en el búfer de movimiento siguiente (**NTYPE**).
3. El grupo de ejes no cambia para movimientos multieje.

Al combinar movimientos multieje, sólo se tiene que establecer el parámetro de eje **MERGE** del eje base.
Nota: si los movimientos son cortos, se debe establecer una velocidad de deceleración alta para evitar que la TJ1-MC__ decelera anticipadamente al final del movimiento almacenado en el búfer.

Argumentos N/A

Ejemplo **MERGE = OFF ' Decelerar al final de cada movimiento**
MERGE = ON ' Los movimientos se combinarán si es posible

Consulte también **AXIS**.

3.2.174 MHELICAL

Tipo	Comando de eje
Sintaxis	**MHELICAL(fin1, fin2, centro1, centro2, dirección, distancia3)** **MH(fin1, fin2, centro1, centro2, dirección, distancia3)**
Descripción	Realiza un movimiento helicoidal, es decir, mueve 2 ejes ortogonales de modo que se produce un arco circular en el punto de herramienta con un movimiento lineal simultáneo en un tercer eje. Los primeros 5 parámetros son similares a los de un comando **MOVECIRC()**. El sexto parámetro define el movimiento lineal simultáneo. Los valores de fin 1 y centro 1 están en el eje **BASE** actual. Los valores de fin 2 y centro 2 están en el eje siguiente. La escala de las 4 primeras distancias y el sexto parámetro se ajustan según el factor de conversión de unidad actual para cada eje.
Argumentos	• **fin1** Posición en el eje **BASE** en la que se finalizará. • **fin2** Posición en el eje siguiente de la matriz **BASE** en la que se finalizará. • **centro1** Posición en el eje **BASE** sobre la que se realizará el movimiento. • **centro2** Posición en el eje siguiente de la matriz **BASE** sobre la que se realizará el movimiento. • **dirección** La **dirección** es un interruptor software que determina si el arco se interpola en el sentido de las agujas del reloj o en sentido contrario. El parámetro se establece en 0 ó 1. Consulte **MOVECIRC**. • **distancia3** La distancia que se recorrerá en el tercer eje en el eje de matriz **BASE** en unidades de usuario.
Ejemplo	No hay ejemplo.
Consulte también	**MOVECIRC**.

3.2.175 MOD

Tipo	Función matemática
Sintaxis	**expresión1 MOD expresión2**
Descripción	La función **MOD** devuelve el resto de **expresión2** entre **expresión1**. Esta función tomará la parte entera de cualquier entrada que no sea un entero.
Argumentos	• **expresión1** Cualquier expresión BASIC válida. • **expresión2** Cualquier expresión BASIC válida.
Ejemplo	**>> PRINT 122 MOD 13** **5.0000**
Consulte también	N/A

3.2.176 MOTION_ERROR

Tipo	Parámetro de sistema (sólo lectura)
Sintaxis	**MOTION_ERROR**
Descripción	El parámetro **MOTION_ERROR** contiene un patrón de bits que muestra los ejes que tienen un error de movimiento. Por ejemplo, si los ejes 2 y 6 tienen un error de movimiento, el valor de MOTION_ERROR será 68 (4 + 64). Los errores de movimiento se producen cuando el estado **AXISSTATUS** de uno de los ejes coincide con el ajuste de **ERRORMASK**. En este caso, el interruptor de activación (**WDOG**) se desactivará, **MOTION_ERROR** contiene un patrón de bits que muestra todos los ejes que tienen el error de movimiento y el parámetro **ERROR_AXIS** contendrá el número del primer eje que tenga el error. Los errores de movimiento se pueden borrar si se ejecuta un comando **DATUM(0)**.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**, **AXISSTATUS**, **DATUM**, **ERROR_AXIS**, **ERRORMASK**, **WDOG**.

3.2.177 MOVE

Tipo: Comando de eje

Sintaxis:
MOVE(distancia_1 [, distancia_2 [, distancia_3 [, distancia_4 [, ...]]]])
MO(distancia_1 [, distancia_2 [, distancia_3 [, distancia_4 [, ...]]])

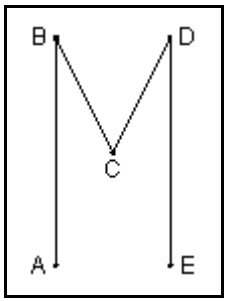
Descripción: El comando **MOVE** mueve uno o varios ejes a la velocidad, aceleración y deceleración solicitadas hasta una posición especificada como incremento desde la posición actual. En los movimientos multieje, el movimiento se interpola y la velocidad, aceleración y deceleración se toman del eje base. La escala de las distancias especificadas se ajusta con el factor de conversión de unidades del parámetro de eje **UNITS**. Por ejemplo, si un eje tiene 4.000 flancos de encoder/mm, el número de unidades para dicho eje se establecerá en 4.000 y **MOVE**(12.5) realizará un movimiento de 12,5 mm.
MOVE funciona en el grupo de ejes base predeterminado (establecido con **BASE**) a menos que se utilice **AXIS** para especificar un eje base temporal. El argumento **distancia_1** se aplica al eje base, **distancia_2** se aplica al eje siguiente, etc. Si se cambia de eje entre comandos **MOVE** individuales, se pueden obtener un movimiento multieje sin interpolar y no sincronizado. Los movimientos incrementales se pueden combinar para obtener movimientos de trayectoria continua con perfil mediante la activación del parámetro de eje **MERGE**.
Si se considera un movimiento de 2 ejes, las velocidades individuales se calculan con las ecuaciones siguientes. Con el comando **MOVE(**x_1, x_2**)** y la velocidad con perfil v_p tal como se ha calculado a partir de los parámetros **SPEED**, **ACCEL** y **DECEL** según el eje base y la distancia multieje total L = **SQR(**$x_1^2 + x_2^2$**)**.
La velocidad individual v_i para el eje i en cualquier momento del movimiento se calcula del siguiente modo: $v_i = (x_i * v_p)/L$.

Argumentos: El comando puede aceptar hasta 16 argumentos.
- **distancia_i**
  La distancia que se recorrerá por cada eje i en unidades de usuario empezando con el eje base.

Ejemplo: Un sistema funciona con un factor de conversión de unidades de 1 y tiene un encoder de 1.000 líneas. Por lo tanto, es necesario utilizar el siguiente comando para realizar 10 giros en el motor. Un encoder de 1.000 líneas ofrece 4.000 flancos/giro.
MOVE(40000)

Ejemplo:
MOVE(10) AXIS(0)
MOVE(10) AXIS(1)
MOVE(10) AXIS(2)
En este ejemplo, los ejes 0, 1 y 2 se mueven independientemente (sin interpolación). Cada eje se moverá a su velocidad programada y otros parámetros de eje.

Ejemplo: Un plóter X-Y puede escribir texto en cualquier posición dentro de su radio de trabajo. Los caracteres individuales se definen como una secuencia de movimientos relativos a un punto de inicio por lo que se pueden utilizar los mismos comandos sin importar la posición de trazado. La subrutina de comandos para la letra M podría ser la siguiente:
MOVE(0,12) ' A -> B
MOVE(3,-6) ' B -> C
MOVE(3,6) ' C -> D
MOVE(0,-12) ' D -> E



Consulte también: **AXIS**, **MOVEABS**, **UNITS**.

3.2.178 MOVEABS

Tipo: Comando de eje

Sintaxis: **MOVEABS(distancia_1 [, distancia_2 [, distancia_3 [, distancia_4 [, ...]]]])**
MA(distancia_1 [, distancia_2 [, distancia_3 [, distancia_4 [, ...]]]])

Descripción: El comando **MOVEABS** mueve uno o varios ejes a la velocidad, aceleración y deceleración solicitadas hasta una posición especificada como posición absoluta, es decir, como referencia al origen. En los movimientos multieje, el movimiento se interpola y la velocidad, aceleración y deceleración se toman del eje base.
La escala de las distancias especificadas se ajusta con el factor de conversión de unidades del parámetro de eje **UNITS**. Por ejemplo, si un eje tiene 4.000 flancos de encoder/mm, el número de unidades para dicho eje se establecerá en 4.000 y **MOVEABS**(12.5) realizará un movimiento de 12,5 mm desde el origen.**MOVEABS** funciona en el grupo de ejes base predeterminado (establecido con **BASE**) a menos que se utilice **AXIS** para especificar un eje base temporal. El argumento **distancia_1** se aplica al eje base, **distancia_2** se aplica al eje siguiente, etc. Si se cambia de eje entre comandos **MOVE** individuales, se pueden obtener un movimiento multieje sin interpolar y no sincronizado. Los movimientos absolutos se pueden combinar para obtener movimientos de trayectoria continua con perfil mediante la activación del parámetro de eje **MERGE**.
Si se considera un movimiento de 2 ejes, las velocidades individuales se calculan con las ecuaciones siguientes. Con el comando **MOVE(**ax_1, ax_2**)**, la posición actual (ay_1,ay_2) y la velocidad con perfil v_p tal como se ha calculado a partir de los parámetros **SPEED**, **ACCEL** y **DECEL** según el eje base y la distancia multieje total L = **SQR(**$x_1^2 + x_2^2$**)**, donde $x_1 = ax_i - ay_i$.
La velocidad individual del eje en cualquier momento del movimiento se calcula del siguiente modo: $v_i = (x_i \times v_p)/L$.

Argumentos: El comando puede aceptar hasta 16 argumentos.

- **distancia_i**
  La posición a la que se moverá cada eje *i* en unidades de usuario empezando con el eje base.

Ejemplo: **MOVEABS(20,350)**
Un plóter X-Y tiene un carrusel de plumillas cuya posición es fija en relación con el origen del plóter. Para cambiar de plumilla, un movimiento absoluto a la posición del carrusel encontrará el destino independientemente de la posición de trazado cuando se ejecute el comando.

Ejemplo: Un palet consta de una cuadrícula de 6 por 8 en la que una máquina de embalaje inserta los bidones de gasolina con una separación de 85 mm. Los bidones se recogen de un punto fijo. La primera posición del palet se define como la posición (0,0) con el comando DEFPOS. La parte del programa para colocar los bidones en el palet es la siguiente:

```
lazox:
  FOR x = 0 TO 5
lazoy:
    FOR y = 0 TO 7
    MOVEABS(-340,-516.5) ' Movimiento hasta el punto de recogida
    GOSUB recoger ' Ir a la subrutina de recogida
    PRINT "MOVER A POSICIÓN: ";x * 6 + y + 1
    MOVEABS(x * 85, y * 85)
    GOSUB colocar ' Ir a la subrutina de colocación
    NEXT y
  NEXT x
```

Consulte también: **AXIS**, **MOVE**, **MOVEABS**, **UNITS**.

3.2.179 MOVECIRC

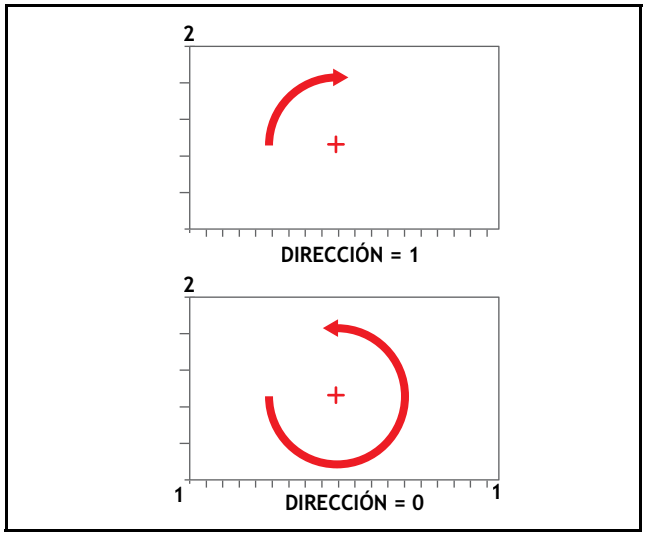
Tipo: Comando de eje

Sintaxis: **MOVECIRC(fin_1,fin_2,centro_1,centro_2,dirección)**
MC(fin_1,fin_2,centro_1,centro_2,dirección)

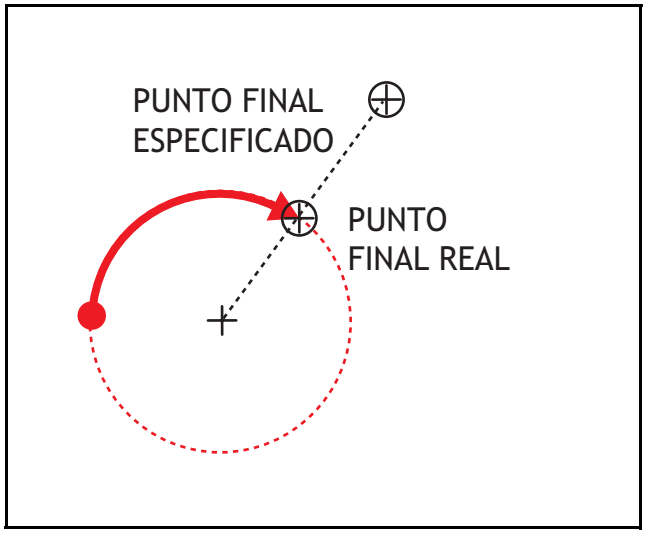
Descripción: El comando **MOVECIRC** interpola 2 ejes ortogonales en un arco circular. La trayectoria del movimiento se determina mediante los 5 argumentos, que son incrementales desde la posición actual.
Los argumentos **fin_1** y **centro_1** se aplican al eje base y **fin_2** y **centro_2** se aplican al eje siguiente. Todos los argumentos se indican en unidades de usuario de cada eje. La velocidad de movimiento a lo largo del arco circular se establece con los parámetros **SPEED**, **ACCEL** y **DECEL** del eje base.
MOVECIRC funciona en el grupo de ejes base predeterminado (establecido con **BASE**) a menos que se utilice **AXIS** para especificar un eje base temporal.
Para que **MOVECIRC** se ejecute correctamente, los dos ejes que se mueven en el arco circular deben tener el mismo número de impulsos de encoder por distancia de eje lineal. Si no es así, en la mayoria de los casos se pueden ajustar las escalas de encoder con el ajuste de los parámetros de eje **ENCODER_RATIO** para el eje.

Argumentos:

- **fin_1**
  La posición final del eje base.
- **fin_2**
  La posición final del eje siguiente.
- **centro_1**
  La posición sobre la que se moverá el eje base.
- **centro_2**
  La posición sobre la que se moverá el eje siguiente.
- **dirección**
  Un interruptor software que determina si el arco se interpola en el sentido de las agujas del reloj o en sentido contrario. Valor: 0 ó 1.
  Si los dos ejes implicados en el movimiento forman un eje a la derecha, establezca la dirección a 0 para producir movimiento positivo sobre el tercer eje ortogonal (posiblemente imaginario). Si los dos ejes implicados en el movimiento forman un eje a la izquierda, establezca la dirección a 0 para producir movimiento negativo sobre el tercer eje ortogonal (posiblemente imaginario). Consulte la siguiente tabla.



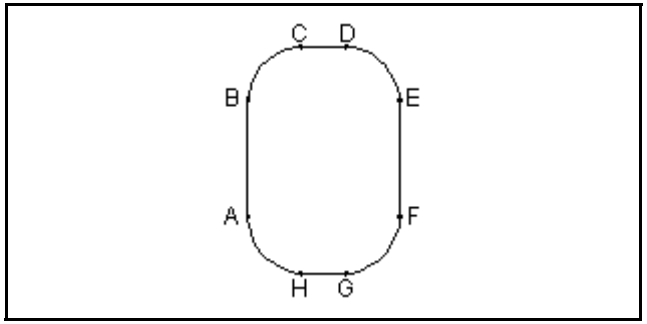
Nota: El comando **MOVECIRC** calcula el radio y el ángulo total de rotación desde el centro y el punto final. Si el punto final no se encuentra en la trayectoria calculada, el movimiento simplemente termina en el final calculado y no en el punto final especificado. Es responsabilidad del programador garantizar que los dos puntos corresponden a puntos correctos en un círculo.



Ejemplo

La secuencia de comandos para trazar la letra 0 podría ser la siguiente:

MOVE(0,6) ' Mover A -> B
MOVECIRC(3,3,3,0,1) ' Mover B -> C
MOVE(2,0) ' Mover C -> D
MOVECIRC(3,-3,0,-3,1) ' Mover D -> E
MOVE(0,-6) ' Mover E -> F
MOVECIRC(-3,-3,-3,0,1) ' Mover F -> G
MOVE(-2,0) ' Mover G -> H
MOVECIRC(-3,3,0,3,1) ' Mover H -> A



Consulte también

AXIS, **ENCODER_RATIO**, **UNITS**

Dirección	Eje a la derecha	Eje a la izquierda
1	Negativo	Positivo
0	Positivo	Negativo

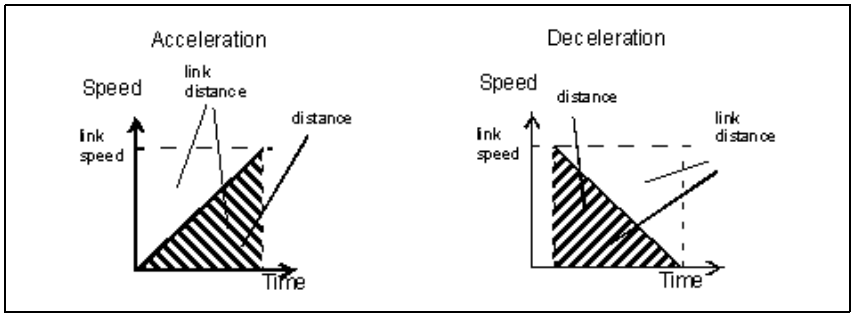
3.2.180 MOVELINK

Tipo: Comando de eje

Sintaxis: **MOVELINK(distancia, distancia_vínculo, aceleración_vínculo, deceleración_vínculo, eje_vínculo [, opción_vínculo [, posición_vínculo]])**
ML(distancia, distancia_vínculo, aceleración_vínculo, deceleración_vínculo, eje_vínculo [, opción_vínculo [, posición_vínculo]])

Descripción: El comando **MOVELINK** crea un movimiento lineal en el eje base vinculado a un engranaje por software hasta la posición medida de un eje de vínculo. El eje de vínculo se puede mover en cualquier dirección para impulsar el movimiento de salida.
Los parámetros muestran la distancia que el eje base recorre para una determinada distancia del eje de vínculo (**distancia_vínculo**). La distancia de eje de vínculo se divide en tres fases que se aplican al movimiento del eje base. Estas partes son la aceleración, la velocidad constante y la deceleración. Las distancias de aceleración y deceleración de vínculo se especifican mediante los parámetros **aceleración_vínculo** y **deceleración_vínculo**.
La distancia de vínculo de velocidad constante se deriva de la distancia de vínculo total y estos dos parámetros.
Las tres fases se pueden dividir en comandos **MOVELINK** independientes o se pueden agrupar en uno solo.
Tenga en cuenta las dos reglas siguientes al configurar el comando **MOVELINK**.
Regla 1: en una fase de aceleración y deceleración con velocidad coincidente, la **distancia_vínculo** debe ser el doble de la distancia. Consulte la figura.
Regla 2: en una fase de velocidad constante con velocidades coincidentes, los dos ejes recorren la misma distancia, por lo que la distancia que se mueve debe ser igual a la **distancia_vínculo**.
MOVELINK funciona en el grupo de ejes base predeterminado (establecido con **BASE**) a menos que se utilice **AXIS** para especificar un eje base temporal. El eje definido para **eje_vínculo** acciona el eje base.
Nota: Si la suma de **aceleración_vínculo** y **deceleración_vínculo** es mayor que **distancia_vínculo**, se reducen en proporción con el fin de igualar la suma a **distancia_vínculo**.

Argumentos



- **distancia**
  La distancia incremental en unidades de usuario para mover el eje base, como resultado del movimiento de **distancia_vínculo** medida en el eje de vínculo.
- **distancia_vínculo**
  La distancia incremental positiva en unidades de usuario que es necesaria medida en el eje de vínculo para que dé como resultado el movimiento de distancia en el eje base.
- **aceleración_vínculo**
  La distancia incremental positiva en unidades de usuario en el eje de vínculo sobre el que acelerará el eje base.
- **deceleración_vínculo**
  La distancia incremental positiva en unidades de usuario en el eje de vínculo sobre el que decelerará el eje base.
  Nota: si la suma de los parámetros 3 y 4 es mayor que el parámetro 2, ambos se reducen proporcionalmente hasta que se igualan al parámetro 2.
- **eje de vínculo**
  El eje al que se vinculará.
- **opción_vínculo**
  Consulte la siguiente tabla.
- **posición_vínculo**
  La posición absoluta donde se iniciará el comando **MOVELINK** cuando **opción_vínculo** se establece en 2.

Nota: El comando utiliza BASE() y AXIS(), y los factores de conversión de unidades de un modo similar a otros comandos MOVE.
Nota: El eje de “vínculo” se puede mover en cualquier dirección para impulsar el movimiento de salida. Las distancias de vínculo especificadas siempre son positivas.

Ejemplo

Una sierra pendular corta un rollo de papel cada 160 m a la vez que se mueve a la velocidad del papel. La sierra puede recorrer hasta 1,2 m, de los que 1 m se utiliza en este ejemplo. La distancia del papel se mide con un encoder y el factor de conversión de unidades se ha configurado para que ofrezca unidades en metros en ambos ejes. El eje 1 es el eje de vínculo.

MOVELINK(0,150,0,0,1) ' esperar distancia
MOVELINK(0.4,0.8,0.8,0,1) ' acelerar
MOVELINK(0.6,1.0,0,0.8,1) ' igualar velocidad y, después, decelerar
WAIT UNTIL NTYPE = 0 ' esperar hasta que se haya iniciado el último movimiento
OP(8,ON) ' activar cortadora
MOVELINK(-1,8.2,0.5,0.5,1) ' volver atrás

En este programa, la TJ1-MC__ espera a que el rollo expulse 150 m en la primera línea. A esta distancia, la sierra acelera a la velocidad del papel, va en marcha libre hasta parada a la misma velocidad y, a continuación, delecera hasta pararse en un recorrido de 1 m. Este movimiento se especifica mediante dos comandos **MOVELINK** independientes. A continuación, el programa espera a que se borre el búfer de movimiento siguiente, **NTYPE = 0**. Esto indica que la fase de aceleración está completa. Las distancias en el eje de vínculo (**distancia_vínculo**) en los comandos **MOVELINK** son 150, 0,8, 1,0 y 8,2, lo que suma 160 m.

Para garantizar que las velocidades y las posiciones de la sierra y el papel coinciden durante la tarea de corte, los argumentos del comando **MOVELINK** deben ser correctos. Sugerencia: trate las fases de aceleración, velocidad constante y deceleración de forma independiente. Tal como se ha indicado, las fases de aceleración y deceleración requieren que la **distancia_vínculo** sea el doble de la distancia. Ambas fases se pueden especificar como:

MOVELINK(0.4,0.8,0.8,0,1) ' Este movimiento es todo aceleración
MOVELINK(0.4,0.8,0,0,8,1) ' Este movimiento es todo deceleración

en una fase de velocidad constante con velocidades coincidentes, los dos ejes recorren la misma distancia, por lo que la distancia que se mueve debe ser igual a la distancia de vínculo. La fase de velocidad constante se puede especificar del siguiente modo:

MOVELINK(0.2,0.2,0,0,1) ' Es todo velocidad constante

El comando **MOVELINK** permite que se agreguen las tres secciones mediante la suma de **distancia**, **distancia_vínculo**, **aceleración_vínculo** y **deceleración_vínculo** para cada fase con el fin de producir el comando del siguiente modo.

MOVELINK(1,1.8,0.8,0.8,1)

En el programa anterior, la fase de aceleración se ha programado de forma independiente. Esto se realiza para que se pueda efectuar una acción al final de la fase de aceleración.

MOVELINK(0.4,0.8,0.8,0,1)
MOVELINK(0.6,1.0,0,0.8,1)

Consulte también

AXIS, **UNITS**, **REP_OPTION**.

Opción de vínculo	Descripción
1	El vínculo comienza cuando el evento de registro se produce en el eje de vínculo.
2	El vínculo comienza en una posición absoluta en el eje de vínculo (consulte **posición_vínculo**).
4	**MOVELINK** se repite automáticamente y de forma bidireccional. Esta opción se cancela si se establece el bit 1 del parámetro **REP_OPTION** (es decir, **REP_OPTION = REP_OPTION OR 2**).
5	Combinación de las opciones 1 y 4.
6	Combinación de las opciones 2 y 4.

3.2.181 MOVEMODIFY

Tipo

Comando de eje

Sintaxis

MOVEMODIFY(posición)
MM(posición)

Descripción

El comando **MOVEMODIFY** cambia la posición final absoluta del movimiento lineal de eje único actual (**MOVE** o **MOVEABS**). Si no hay ningún movimiento actual o el actual no es lineal, **MOVEMODIFY** se trata como un comando **MOVEABS**. El parámetro **ENDMOVE** contendrá la posición del final del movimiento actual en unidades del usuario.

MOVEMODIFY funciona en el eje base predeterminado (establecido con **BASE**) a menos que se utilice **AXIS** para especificar un eje base temporal.

Argumentos

- **Posición**
  La posición absoluta que se configurará como el nuevo final del movimiento.

Ejemplo

No hay ejemplo.

Consulte también

AXIS, **MOVE**, **MOVEABS**, **UNITS**.

3.2.182 MPOS

Tipo	Parámetro de eje (sólo lectura)
Sintaxis	**MPOS**
Descripción	El parámetro **MPOS** es la posición medida del eje en unidades de usuario tal como se deriva del encoder. Este parámetro se puede configurar mediante el comando **DEFPOS**. El parámetro de eje **OFFPOS** también se puede utilizar para desplazar el punto de origen. **MPOS** se restablece a 0 al iniciarse. El rango de la posición medida se controla con los parámetros de eje **REP_DIST** y **REP_OPTION**.
Argumentos	N/A
Ejemplo	**WAIT UNTIL MPOS >= 1250** **SPEED = 2.5**
Consulte también	**UNITS**, **AXIS**, **DEFPOS**, **ENCODER**, **FE**, **OFFPOS**, **REP_DIST**, **REP_OPTION**, **UNITS**.

3.2.183 MSPEED

Tipo	Parámetro de eje (sólo lectura)
Sintaxis	**MSPEED**
Descripción	El parámetro **MSPEED** contiene la velocidad medida en unidades/s. Se calcula tomando el cambio en la posición medida en unidades de usuario en el último período de servo y dividiéndolo entre el período de servo (en segundos). El período de servo se configura con el parámetro **SERVO_PERIOD**. **MSPEED** representa una instantánea de la velocidad y las fluctuaciones importantes que se puedan producir, en concreto a velocidades bajas. Puede merecer la pena calcular el promedio de varias lecturas si se requiere un valor estable a velocidades bajas.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**, **SERVO_PERIOD**, **VP_SPEED**, **UNITS**.

3.2.184 MTYPE

Tipo	Parámetro de eje (sólo lectura)
Sintaxis	**MTYPE**
Descripción	El parámetro **MTYPE** contiene el tipo de movimiento que se está ejecutando actualmente. Los valores posibles se indican en la tabla siguiente. **MTYPE** se puede utilizar para determinar si un movimiento ha terminado o si se ha producido una transición de un tipo de movimiento a otro. Un tipo de movimiento no inactivo no significa necesariamente que el eje se esté moviendo realmente. Puede ser una parte de velocidad 0 a lo largo de un movimiento o una interpolación con otro eje sin moverse él mismo.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**, **NTYPE**.

Número de movimiento	Tipo de movimiento
0	**IDLE**(sin movimiento)
1	**MOVE**
2	**MOVEABS**
3	**MHELICAL**
4	**MOVECIRC**
5	**MOVEMODIFY**
10	**FORWARD**
11	**REVERSE**
12	**DATUM**
13	**CAM**
14	JOG_FORWARD, consulte **FWD_JOG**
15	JOG_REVERSE, consulte **REV_JOG**
20	**CAMBOX**
21	**CONNECT**
22	**MOVELINK**

3.2.185 NAIO

Tipo	Parámetro de sistema (sólo lectura)
Sintaxis	**NAIO**
Descripción	Este parámetro devuelve el número de canales de entrada analógica conectados en el bus de expansión MECHATROLINK-II. Por ejemplo, una TJ1-MC__ devolverá 8 si hay dos unidades AN2900 conectadas, ya que cada una tiene 4 canales de entrada analógica.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	N/A

3.2.186 NEG_OFFSET

Tipo	Parámetro de sistema
Sintaxis	**NEG_OFFSET = valor**
Descripción	Para el funcionamiento de Piezo. Permite aplicar un offset negativo a la señal **DAC** de salida desde el lazo de servo. El offset se aplica después de la función **DAC_SCALE**. Un offset de 327 representará un offset de 0,1 voltios. Se sugiere utilizar un offset del 65% al 70% del valor necesario para realizar el movimiento de etapa en una situación de lazo abierto.
Argumentos	• **valor** Una expresión BASIC.
Ejemplo	No hay ejemplo.
Consulte también	N/A

3.2.187 NEW

Tipo	Comando de programa
Sintaxis	**NEW ["nombre_programa"]**
Descripción	El comando **NEW** elimina todas las líneas del programa del controlador. Se puede utilizar **NEW** sin un nombre de programa para eliminar el programa seleccionado actualmente (mediante **SELECT**). El nombre de programa también se puede especificar sin comillas. Con **NEW ALL** se eliminan todos los programas. El comando también se puede utilizar para eliminar la tabla: **NEW "TABLE"** El nombre **"TABLE"** debe estar entre comillas. Nota: Este comando se ha implementado para un terminal (VT100) offline. En Trajexia Tools, los usuarios pueden seleccionar el comando desde el menú Program (Programa).
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**COPY**, **DEL**, **RENAME**, **SELECT**, **TABLE**

3.2.188 NEXT

Consulte **FOR..TO..STEP..NEXT**.

3.2.189 NIO

Tipo	Parámetro de sistema
Sintaxis	**NIO**
Descripción	Devuelve el número de entradas/salidas instaladas en el sistema o conectadas en el bus de expansión MECHATROLINK-II. Una TJ-MC__ sin ninguna E/S MECHATROLINK-II devolverá **NIO = 32**. Las entradas integradas son los canales del 0 al 15. Las salidas integradas son los canales del 8 al 15. Los canales del 16 al 27 se pueden utilizar como E/S “virtuales” que están conectadas entre sí. Los canales del 28 al 31 están reservados para que cada eje pueda utilizar los canales de entrada de driver MECHATROLINK-II para las funciones de control de eje.
Argumentos	N/A

Ejemplo	No hay ejemplo.
Consulte también	N/A

3.2.190 NOT

Tipo	Operación matemática
Sintaxis	**NOT expresión**
Descripción	El operador **NOT** realiza la función **NOT** lógica en todos los bits de la parte entera de la expresión. La función **NOT** lógica se define del modo que se indica en la tabla siguiente.
Argumentos	• **expresión** Cualquier expresión BASIC válida.
Ejemplo	**>> PRINT 7 AND NOT 1** **6.0000**
Consulte también	N/A

Bit	Resultado
0	1
1	0

3.2.191 NTYPE

Tipo	Parámetro de eje (sólo lectura)
Sintaxis	**NTYPE**
Descripción	El parámetro **NTYPE** Contiene el tipo del movimiento en el búfer de movimiento siguiente. Una vez terminado el movimiento actual, se ejecutará el movimiento que se encuentra en el búfer **NTYPE**. Los valores son los mismos que los del parámetro de eje **MTYPE**. **NTYPE** se borra con el comando **CANCEL(1)**.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**, **MTYPE**.

3.2.192 OFF

Tipo	Constante (sólo lectura)
Sintaxis	**OFF**
Descripción	La constante **OFF** devuelve el valor numérico 0.
Argumentos	N/A
Ejemplo	**OP (palanca,OFF)** **La línea anterior establece la salida denominada palanca en OFF.**
Consulte también	N/A

3.2.193 OFFPOS

Tipo	Parámetro de eje
Sintaxis	**OFFPOS**
Descripción	El parámetro **OFFSPOS** contiene un offset que se aplicará a la posición solicitada (**DPOS**) sin que se vea afectado el movimiento de ninguna otra forma. La posición medida se cambiará en consecuencia para mantener el error de seguimiento. **OFFPOS** ajusta de forma efectiva la posición 0 del eje. El sistema restablecerá a 0 el valor definido en **OFFPOS** a medida que se cargue el offset. Nota: el offset se aplica en el siguiente período de servo. Se pueden ejecutar otros comandos antes del siguiente período de servo. Asegúrese de que en estos comandos no se suponga que se ha producido el desplazamiento de posición. Esto se puede realizar mediante la instrucción **WAIT UNTIL** (consulte el ejemplo).
Argumentos	N/A
Ejemplo	En las siguientes líneas se define la posición solicitada actual como 0. **OFFPOS = -DPOS** **WAIT UNTIL OFFPOS = 0 ' Esperar hasta que se aplique** Este ejemplo equivale a DEFPOS(0).
Consulte también	**AXIS**, **DEFPOS**, **DPOS**, **MPOS**, **UNITS**.

3.2.194 ON

Tipo	Constante (sólo lectura)
Sintaxis	**ON**
Descripción	La constante **ON** devuelve el valor numérico 1.
Argumentos	N/A
Ejemplo	**OP (palanca,ON)** La línea anterior establece la salida denominada palanca en **ON**.
Consulte también	N/A

3.2.195 ON.. GOSUB

Tipo	Comando de control de programa
Sintaxis	**ON expresión GOSUB etiqueta {, etiqueta }**
Descripción	Las estructuras **ON..GOSUB** y **ON..GOTO** permiten efectuar un salto condicional. La expresión entera se utiliza para seleccionar una etiqueta de la lista. Si la expresión tiene un valor 1, se utiliza la primera etiqueta; para el valor 2 se utiliza la segunda etiqueta, etc. Una vez seleccionada la etiqueta, se realiza el salto **GOSUB** de subrutina a dicha etiqueta. Nota: si la expresión no es válida, no se lleva a cabo ningún salto.
Argumentos	• **expresión** Cualquier expresión BASIC válida. • **etiqueta** Cualquier etiqueta válida del programa.
Ejemplo	**REPEAT** **GET#5,car** **UNTIL 1<=car and car<=3** **ON car GOSUB mover, parar, cambiar**
Consulte también	**GOSUB..RETURN**, **GOTO**.

3.2.196 ON.. GOTO

Tipo	Comando de control de programa
Sintaxis	**ON expresión GOTO etiqueta[,etiqueta[,...]]**
Descripción	Se evalúa la expresión y la parte entera se utiliza para seleccionar una etiqueta de la lista. Si la expresión tiene el valor 1, se utiliza la primera etiqueta; si es 2 se utiliza la segunda etiqueta, etc. Si el valor de la expresión es menor que 1 o mayor que el número de etiquetas, se produce un error. Una vez seleccionada la etiqueta, se realiza el salto **GOTO** de subrutina a dicha etiqueta.
Argumentos	• **expresión** Cualquier expresión BASIC válida. • **etiqueta** Cualquier etiqueta válida del programa.
Ejemplo	**REPEAT** **GET #1,car** **UNTIL 1<=car and car<=3** **ON car GOTO mover,parar,cambiar**
Consulte también	N/A

3.2.197 OP

Tipo	Comando de E/S
Sintaxis	**OP(número_salida, valor)** **OP(patrón_binario)** **OP**

Descripción El comando **OP** establece una o varias salidas o devuelve el estado de las primeras 24 salidas. **OP** tiene tres formatos distintos según el número de argumentos.

- El comando **OP(número_salida,valor)** establece un solo canal de salida. El rango de **número_salida** está entre 8 y 256, y **valor** es el valor que se enviará, 0 ó 1.
- El comando **OP(patrón_binario)** establece el patrón binario en las 24 salidas según el valor definido por **patrón_binario**.
- La función **OP (sin argumentos)** devuelve el estado de las primeras 24 salidas. Esto permite que se establezcan varias salidas sin que se dañen las demás que no se cambian.

Nota: las primeras 8 salidas (de 0 a 7) no existen físicamente en la TJ1-MC__. No se puede escribir en ellas y siempre devuelven 0.

Argumentos
- **número_salida**
  El número de la salida que se establecerá.
- **valor**
  El valor que se enviará, tanto OFF como ON.
  Todos los valores distintos de 0 se consideran ON.
- **patrón_binario**
  El equivalente entero del patrón binario que se enviará.

Ejemplo
OP(12,1)
OP(12,ON)
Las dos líneas son equivalentes.

Ejemplo
OP(18 * 256)
Esta línea establece el patrón de bits 10010 en las primeras 5 salidas físicas; las salidas 13 a 17 se borrarán. El patrón de bits se desplaza 8 bits al realizar la multiplicación por 256 para establecer las primeras salidas disponibles ya que las salidas 0 a 7 no existen.

Ejemplo
VR(0) = OP
VR(0) = VR(0) AND 65280
OP(VR(0))
Esta rutina establece las salidas 8 a 15 en **ON** y todas las demás en OFF.
El programa anterior también se puede escribir del siguiente modo:
OP(OP AND 65280)

Ejemplo
val = 8 ' El valor que se establecerá
mask = OP AND NOT(15 * 256) ' Obtener el estado y máscara actuales
OP(mask OR val * 256) ' Establecer el valor en OP(8) a OP(11)
Esta rutina establece el valor **val** en las salidas 8 a 11 sin que se vean afectadas las demás salidas mediante el uso de enmascaramiento.

Consulte también **IN**.

3.2.198 OPEN_WIN

Tipo Parámetro de eje

Sintaxis
OPEN_WIN
OW

Descripción El parámetro **OPEN_WIN** define el principio del periodo ventana dentro o fuera del que se espera una marca de registro. El valor se indica en unidades del usuario.

Argumentos N/A

Ejemplo No hay ejemplo.

Consulte también **CLOSE_WIN**, **REGIST**, **UNITS**.

3.2.199 OR

Tipo Operación matemática

Sintaxis **expresión1 OR expresión2**

Descripción El operador **OR** realiza la función **OR** lógica entre los bits correspondientes de las partes enteras de dos expresiones BASIC válidas.
La función **OR** lógica entre dos bits se define del modo que se indica en la tabla siguiente.

Argumentos
- **expresión1**
  Cualquier expresión BASIC válida.
- **expresión2**
  Cualquier expresión BASIC válida.

Ejemplo **Ejemplo 1:**
Resultado = 10 OR (2.1 * 9)
Los paréntesis se evalúan en primer lugar, pero sólo la parte entera del resultado, 18, se utiliza para la operación. Por lo tanto, esta expresión equivale a lo siguiente:
resultado = 10 OR 18

Por lo tanto, el resultado contendrá el valor 26.
Ejemplo 2:

Ejemplo **Resultado = 10 OR 18**
El operador **OR** es un operador de bit y, por lo tanto, la acción binaria que se realiza es:
01010 OR 10010 = 11010

Ejemplo **IF KEY OR VR(0) = 2 THEN GOTO etiqueta**

Consulte también N/A

Bit 1	Bit 2	Resultado
0	0	0
0	1	1
1	0	1
1	1	1

3.2.200 OUTDEVICE

Tipo Parámetro de E/S

Sintaxis **OUTDEVICE**

Descripción El parámetro **OUTDEVICE** define el dispositivo de salida predeterminado. Este dispositivo se seleccionará para el comando **PRINT** cuando se omita la opción **#n**. El parámetro **OUTDEVICE** es específico de la tarea. Los valores admitidos se enumeran en la tabla siguiente.

Argumentos N/A

Ejemplo No hay ejemplo.

Consulte también **PRINT**.

Valor	Descripción
0	Puerto de programación 0 (predeterminado)
1	Puerto 1 comunicacion serie RS-232C
2	Puerto 2 comunicacion serie RS-422A/485
5	Canal de usuario 5 de puerto 0 de Trajexia Tools
6	Canal de usuario 6 de puerto 0 de Trajexia Tools
7	Canal de usuario 7 de puerto 0 de Trajexia Tools

3.2.201 OUTLIMIT

Tipo Parámetro de eje

Sintaxis **OUTLIMIT**

Descripción El límite de salida restringe la salida solicitad desde un eje a un valor menor que el máximo. El valor necesario varía según la salida solicitada máxima que sea posible. Si la salida de tensión se generan mediante valores de DAC de 16 bits, un parámetro **OUTLIMIT** de 32.767 producirá el rango completo de +/–10 V. Un eje de velocidad MECHATROLINK-II tiene una solicitud máxima de 32 bits.

Argumentos N/A

Ejemplo No hay ejemplo.

Consulte también **AXIS**, **S_REF**, **S_REF_OUT**, **SERVO**.

3.2.202 OV_GAIN

Tipo Parámetro de eje

Sintaxis **OV_GAIN**

Descripción El parámetro **OV_GAIN** contiene la ganancia de velocidad de salida.
La aportación de salida de velocidad de salida se calcula multiplicando el cambio en la posición medida por el valor del parámetro **OV_GAIN**.
El valor predeterminado es 0.
Incorporar ganancia de velocidad de salida a un sistema es el equivalente mecánico a añadir amortiguación. Es probable que se produzca una respuesta más suave y permitir el uso de una mayor ganancia proporcional de la que se utilizaría, pero a costa de mayores errores de seguimiento. Los valores altos pueden provocar oscilaciones y producir elevados errores de seguimiento.
Nota: para evitar inestabilidad, las ganancias de servo sólo se deben cambiar cuando **SERVO** está en OFF.

Argumentos N/A

Ejemplo No hay ejemplo.

Consulte también **D_GAIN**, **I_GAIN**, **P_GAIN**, **VFF_GAIN**.

3.2.203 P_GAIN

Tipo Parámetro de eje

Sintaxis **P_GAIN**

Descripción El parámetro **P_GAIN** contiene la ganancia proporcional. La aportación de salida proporcional se calcula multiplicando el error de seguimiento por el valor del parámetro **P_GAIN**. El valor predeterminado de **P_GAIN** para el eje de velocidad Mechatro (**ATYPE = 41**) es 131.072. El valor predeterminado para el servoeje flexible (**ATYPE = 44**) es 1,0.
La ganancia proporcional establece la rigidez de la respuesta del servo.
Los valores que sean demasiado altos provocarán oscilación. Los valores que sean demasiado bajos provocarán errores de seguimiento grandes.
Nota: para evitar inestabilidad, las ganancias de servo sólo se deben cambiar cuando **SERVO** está en OFF.

Argumentos N/A

Ejemplo No hay ejemplo.

Consulte también **D_GAIN**, **I_GAIN**, **OV_GAIN**, **VFF_GAIN**.

3.2.204 PI

Tipo Constante (sólo lectura)

Sintaxis **PI**

Descripción La constante **PI** devuelve el valor numérico 3,1416.

Argumentos N/A

Ejemplo
```
circunferencia = 100
PRINT "Radio = ";circunferencia/(2 * PI)
```

Consulte también N/A

3.2.205 PMOVE

Tipo Parámetro de tarea (sólo lectura)

Sintaxis **PMOVE**

Descripción El parámetro **PMOVE** contiene el estado actual de los búferes de tarea. El parámetro **TRUE** si los búferes de tarea están ocupados y **FALSE** si están vacíos. Cuando la tarea ejecuta un comando de movimiento, ésta carga la información movimiento en los búferes de movimiento de tarea. Los búferes pueden contener una instrucción de movimiento por cualquier grupo de ejes. **PMOVE** se establecerá en **TRUE** cuando se haya completado la carga de los búferes. Cuando se produzca la siguiente interrupción de servo, el generador de Motion Control carga el movimiento en el búfer de siguiente movimiento (**NTYPE**) de los ejes requeridos si están disponibles. Cuando se ha realizado esta segunda transferencia, **PMOVE** se borra a 0 hasta que se ejecute otro movimiento en la tarea.
Cada tarea tiene su propio parámetro **PMOVE**. Utilice el modificador **PROC** para acceder al parámetro para una determinada tarea.
Sin **PROC** se supondrá la tarea actual.

Argumentos N/A

Ejemplo No hay ejemplo.

Consulte también **NTYPE**, **PROC**.

3.2.206 POS_OFFSET

Tipo	Parámetro de sistema
Sintaxis	**POS_OFFSET = valor**
Descripción	Para el funcionamiento de Piezo. Esta palabra clave permite aplicar un offset positivo a la señal **DAC** de salida desde el lazo de servo. El offset se aplica después de la función **DAC_SCALE**. Un offset de 327 representará un offset de 0,1 voltios. Se sugiere utilizar un offset del 65% al 70% del valor necesario para realizar el movimiento de etapa en una situación de lazo abierto.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	N/A

3.2.207 POWER_UP

Tipo	Parámetro de sistema
Sintaxis	**POWER_UP**
Descripción	Este parámetro se utiliza para determinar si los programas se deben leer desde la EPROM flash en el encendido o se debe realizar un reset por software (**EX**). Son posibles dos valores: 0: utilizar los programas de la RAM respaldada por baterías; 1: copiar los programas de la EPROM flash del controlador en RAM. Los programas se seleccionan individualmente para ejecutarse al inicio con el comando **RUNTYPE**. Notas: • **POWER_UP** siempre es un comando inmediato y, por lo tanto, no se puede incluir en los programas. • Este valor normalmente lo establece Trajexia Tools.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**EPROM**

3.2.208 PRINT

Tipo	Comando de E/S
Sintaxis	**PRINT [#n,] expresión {, expresión}** **? [#n,] expresión {, expresión}**
Descripción	El comando **PRINT** envía una serie de caracteres a los puertos serie. **PRINT** puede enviar parámetros, cadenas ASCII fijas y caracteres ASCII individuales. Al utilizar **PRINT #n**, se puede seleccionar cualquier puerto al que se le enviará la información. Los diferentes elementos que se imprimirán se pueden colocar en la misma línea separados por una coma o punto y coma. Un separador de coma en el comando de impresión coloca una tabulación entre los elementos impresos. El separador de punto y coma imprime el siguiente elemento sin espacios entre los elementos impresos. El ancho del campo en el que se imprime un número se puede configurar con el uso de [w,x] después del número que se imprimirá. El ancho de la columna se indica con w y el número de decimales se especifica con x. Al usar el parámetro [x] únicamente, se adopta el ancho predeterminado y se especifican los decimales que se imprimirán. Los números están alineados a la derecha en el campo y los caracteres iniciales que no se utilicen se rellenan con espacios. Si el número es demasiado largo, el campo se rellenará con asteriscos para indicar que no había suficiente espacio para mostrar el número. El ancho de campo máximo permitido es de 127 caracteres. El comando ****, barra diagonal inversa, se puede utilizar para imprimir un solo carácter ASCII.
Argumentos	• **N** El dispositivo de salida especificado. Cuando se omite este argumento, se utilizará el puerto especificado por **OUTDEVICE**. Consulte la siguiente tabla. • **expresión** La expresión que se imprimirá.
Ejemplo	**PRINT "SE PUEDEN IMPRIMIR LETRAS EN MAYÚSCULAS y minúsculas"**
Ejemplo	Si VR(1) = 6 y **variab** = 1.5, la salida de impresión será la siguiente: **PRINT 123.45,VR(1)-variab** **123.4500 4.5000**
Ejemplo	**longitud:** **PRINT "DISTANCIA = ";mpos** **DISTANCIA = 123.0000** En este ejemplo se utiliza un separador de punto y coma. No pasa a la siguiente columna, lo que ofrece al programador más libertad sobre dónde se colocan los elementos de impresión.

Ejemplo **PRINT VR(1)[4,1];variab[6,2]**
6.0 1.50

Ejemplo **params:**
PRINT "DISTANCIA = ";mpos[0];" VELOCIDAD = ";v[2];
DISTANCIA = 123 VELOCIDAD = 12.34

Ejemplo **PRINT "ELEMENTO ";total" DE ";limit;CHR(13);**

Ejemplo **>> PRINT HEX(15),HEX(-2)**
F FFFFA

Consulte también **$ (ENTRADA HEXADECIMAL)**, **OUTDEVICE**.

Valor	Descripción
0	Puerto de programación 0 (predeterminado)
1	Puerto 1 comunicación serie RS-232C
2	Puerto 2 comunicación serie RS-422A/485
5	Canal de usuario 5 de puerto 0 de Trajexia Tools
6	Canal de usuario 6 de puerto 0 de Trajexia Tools
7	Canal de usuario 7 de puerto 0 de Trajexia Tools

3.2.209 PROC

Tipo Comando de tarea

Sintaxis **PROC(número_tarea)**

Descripción El modificador **PROC** permite leer o escribir un parámetro de proceso desde un determinado proceso. Si se omite, se supondrá la tarea actual.

Argumentos
- **número_tarea**
  El número de la tarea a la que se accederá.

Ejemplo **WAIT UNTIL PMOVE PROC(3) = 0**

Consulte también N/A

3.2.210 PROC_STATUS

Tipo Parámetro de tarea

Sintaxis **PROC_STATUS**

Descripción El parámetro **PROC_STATUS** devuelve el estado del proceso o tarea especificados. El parámetro se utiliza con el modificador **PROC** y puede devolver los valores enumerados en la tabla siguiente.

Argumentos N/A

Ejemplo **WAIT UNTIL PROC_STATUS PROC(3) = 0**

Consulte también **PROCNUMBER**, **PROC**.

Valor	Descripción
0	Proceso parado
1	Proceso en ejecución
2	Proceso paso a paso
3	Proceso en pausa

3.2.211 PROCESS

Tipo	Comando de programa
Sintaxis	**PROCESS**
Descripción	El comando **PROCESS** devuelve la lista de estados de todas las tareas en ejecución con su número de tarea.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**HALT**, **RUN**, **STOP**.

3.2.212 PROCNUMBER

Tipo	Parámetro de tarea (sólo lectura)
Sintaxis	**PROCNUMBER**
Descripción	El parámetro **PROCNUMBER** contiene el número de la tarea en la que se está ejecutando el programa seleccionado actualmente. **PROCNUMBER** suele ser necesario cuando se ejecutan varias copias de un programa en diferentes tareas.
Argumentos	N/A
Ejemplo	**MOVE(longitud) AXIS(PROCNUMBER)**
Consulte también	**PROC_STATUS**, **PROC**.

3.2.213 PROFIBUS

Tipo	Comando de sistema
Sintaxis	**PROFIBUS(número_unidad, 2,1,salidas_inicio_VR,número_salidas, entradas_inicio_VR,número_entradas)** **PROFIBUS(número_unidad,4,0)**
Descripción	La función 2 de **PROFIBUS** configura la TJ1-PRT para el intercambio de datos con la unidad maestra PROFIBUS-DP y define las áreas de la memoria VR donde se realiza el intercambio de E/S. La función 4 de **PROFIBUS** devuelve el estado de intercambio de datos de la TJ1-PRT. Consulte en la tabla la descripción de los bits en el canal de estado de intercambio de datos.
Argumentos	• **número_unidad** Especifica el número de unidad de la TJ1-PRT en el sistema Trajexia. • **salidas_inicio_VR** La dirección de inicio de la memoria VR del controlador donde se encuentran los datos de salida del maestro PROFIBUS-DP. • **número_salidas** El número de canales de salida desde el maestro PROFIBUS-DP en la memoria VR. • **entradas_inicio_VR** La dirección de inicio de la memoria VR del controlador donde se encuentran los datos de entrada para el maestro PROFIBUS-DP. • **número_entradas** El número de canales de entrada al maestro PROFIBUS-DP en la memoria VR.
Ejemplo	**PROFIBUS (0,2,1,10,16,150,31)** En este ejemplo, la TJ1-PRT está configurada para intercambiar datos con el maestro PROFIBUS-DP con 16 canales de salida (recibidos del maestro) situados en VR(10) a VR(25) y 31 canales de entrada (enviados al maestro) situados en VR(150) a VR(180).
Consulte también	N/A

Bit	Valor	Descripción
0	0	Configuración fallida del intercambio de datos de E/S
	1	Intercambio de datos de E/S configurado correctamente
1	0	Datos de E/S no disponibles
	1	Datos de E/S disponibles
2	0	Intercambio de datos activo en modo OPERATE
	1	Intercambio de datos activo en modo CLEAR

3.2.214 PSWITCH

Tipo: Comando de E/S

Sintaxis: **PSWITCH(interruptor, activar [, eje, número_salida, estado_salida, establecer_posición, reset_posición])**

Descripción: El comando **PSWITCH** Activa una salida cuando se alcanza una posición predefinida y la desactiva cuando se alcanza una segunda posición.
Las posiciones se especifican como posiciones absolutas medidas.
Hay 16 interruptores de posición, cada uno de los cuales se puede asignar a cualquier eje. A cada interruptor se le asignan sus propias posiciones de ON y OFF, así como el número de salida.
El comando se puede utilizar con 2 argumentos o los 7. Con sólo 2 argumentos, se puede desactivar un determinado interruptor.
Los comandos **PSWITCH** se calculan en cada ciclo de servo y el resultado de salida se aplica al hardware. Por lo tanto, el tiempo de respuesta es de 1 periodo de ciclo de servo aproximadamente.
Nota: una salida puede permanecer activada si ya lo estaba cuando se desactivó el comando **PSWITCH**. El comando **OP** se puede utilizar para desactivar una salida del siguiente modo:
PSWITCH(2,OFF) OP(14,OFF) ' Desactivar el interruptor físico que controla OP 14
Nota: los interruptores físicos que se utilizan con **PSWITCH** no son interruptores de hardware rápidos, por lo que la conmutación se realiza por software, lo que puede introducir retrasos en la operación. La conmutación por hardware rápida sólo se puede utilizar con ejes conectados mediante la TJ1-FL02. Utilice el comando **HW_PSWITCH**.

Argumentos
- **conmutador**
  El número de interruptor. Rango: [0,15].
- **activar**
  La activación de interruptor. Rango: [on, off].
- **eje**
  El número del eje que proporcionará la entrada de posición.
- **número_salida**
  La salida física que se establecerá. Rango: [8,31].
- **estado_salida**
  El estado que se enviará. Rango: [on, off].
- **establecer_posición**
  La posición absoluta en unidades de usuario en la que se establece la salida.
- **reset_posición**
  La posición absoluta en unidades de usuario en la que se hace un reset de la salida.

Ejemplo

Un eje rotativo tiene una leva accionada por interruptor que se tiene que cambiar para piezas de trabajo de distinto tamaño. También hay un interruptor de proximidad en el eje para indicar el TDC de la máquina. Con una leva mecánica, el cambio de un trabajo a otro consume tiempo. Esto se puede facilitar mediante el uso de PSWITCH como un interruptor de leva de software. El interruptor de proximidad está cableado a la entrada 7 y la salida es la 11. El eje se controla mediante el eje 0. El motor tiene un encoder de 900 ppr. La salida debe estar activada desde las 80 unidades.
PSWITCH utiliza el factor de conversión de unidades para permitir que las posiciones se establezcan en unidades cómodas. Primero se debe calcular y establecer el factor de conversión de unidades. Cada impulso en un encoder ofrece cuatro flancos para que los cuente la TJ1-MC__. Por lo tanto, hay 3.600 flancos/rev. o 10 flancos/grado. Si establece el factor de conversión de unidades a 10, puede trabajar en grados.
A continuación tiene que determinar un valor para todos los argumentos de **PSWITCH**.
interruptor: el número de interruptor puede ser cualquier interruptor que no esté en uso. En este ejemplo, utilizará el número 0.
activación: el interruptor debe estar activado para que funcione; establezca la activación en 1.
eje: el eje se controla mediante el eje 0.
número_salida: la salida que se va a controlar es la 11.
estado_salida: la salida debe estar activada, por lo que debe establecerlo en 1.
establecer_posición: la salida se produce a las 80 unidades.
rspos: la salida debe estar activada durante un período de 120 unidades.
Todo esto se puede poner en las siguientes líneas de código BASIC:

```
interruptor:
  UNITS AXIS(0) = 10 ' Establecer el factor de conversión de unidades
  REPDIST = 360
  REP_OPTION = ON
  PSWITCH(0,ON,0,11,ON,80,200)
```

Este programa utiliza la distancia de repetición establecida en 360 grados y la opción de repetición activada de modo que la posición de eje se mantendrá entre 0 y 360 grados.

Consulte también

HW_PSWITCH, **OP**, **UNITS**.

3.2.215 RAPIDSTOP

Tipo

Comando de eje

Sintaxis

RAPIDSTOP
RS

Descripción

El comando **RAPIDSTOP** cancela el movimiento actual en todos los ejes del búfer de movimientos actual (**MTYPE**). Se decelerarán los movimientos para comandos de movimiento con perfil de velocidad (**MOVE**, **MOVEABS**, **MOVEMODIFY**, **FORWARD**, **REVERSE**, **MOVECIRC** y **MHELICAL**) hasta pararse con la velocidad de deceleración tal como se define mediante el parámetro **DECEL**. Los movimientos de los demás comandos se pararán inmediatamente.
Notas:

- **RAPIDSTOP** sólo cancela los movimientos que se ejecutan actualmente. Si hay más movimientos almacenados en los búferes de movimiento siguiente (**NTYPE**) o los búferes de tareas, se cargarán.
- Durante la deceleración de los movimientos actuales, se omitirán los comandos **RAPIDSTOP** adicionales.

Argumentos

N/A

Ejemplo

No hay ejemplo.

Consulte también

CANCEL, **MTYPE**, **NTYPE**.

3.2.216 READ_BIT

Tipo

Comando de sistema

Sintaxis

READ_BIT(número_bit, número_vr)

Descripción

El comando **READ_BIT** devuelve el valor del bit especificado de la variable VR especificada, 0 ó 1.

Argumentos

- **número_bit**
  El número del bit que se leerá. Rango: [0,23].
- **número_vr**
  El número de la variable VR cuyo bit se leerá. Rango: [0,1.023].

Ejemplo

No hay ejemplo.

Consulte también

CLEAR_BIT, **SET_BIT**.

3.2.217 REG_POS

Tipo	Parámetro de eje (sólo lectura)
Sintaxis	**REG_POS**
Descripción	El parámetro **REG_POS** almacena la posición en unidades de usuario en la que se ha producido el evento de registro principal.
Argumentos	N/A
Ejemplo	**PRINT REG_POS AXIS(2)** De este modo se imprimirá la posición de registro en unidades de usuario para el eje 2.
Consulte también	**AXIS**, **MARK**, **REGIST**.

3.2.218 REG_POSB

Tipo	Parámetro de eje (sólo lectura)
Sintaxis	**REG_POSB**
Descripción	El parámetro **REG_POSB** almacena la posición en unidades de usuario en la que se ha producido el evento de registro secundario.
Argumentos	N/A
Ejemplo	**PRINT REG_POSB AXIS(2)** De este modo se imprimirá la posición de registro en unidades de usuario para el eje 2.
Consulte también	**AXIS**, **MARKB**, **REGIST**.

3.2.219 REGIST

Tipo	Comando de eje
Sintaxis	**REGIST(modo)**
Descripción	El comando **REGIST** configura la operación de registro. El comando captura una posición de eje cuando se detecta una señal de registro. Con una TJ1-FL02, la captura se realiza mediante hardware, por lo que los retrasos de software no afectan a la precisión de la posición que se captura. Con un eje MECHATROLINK-II, la captura se realiza por parte del servodriver. Para un eje flexible, un comando **REGIST** puede capturar dos posiciones de registro mediante entradas de registro independientes. Cuando se ha producido un evento de registro principal, el parámetro de eje **MARK** se establece en ON y la posición se almacena en el parámetro de eje **REG_POS**. Para el evento de registro secundario, el parámetro de eje **MARKB** se establece en ON y la posición se almacena en el parámetro de eje **REG_POSB**. Este comando sólo es aplicable a los ejes flexibles con los valores 43, 44 y 45 de **ATYPE**. El registro MECHATROLINK-II se puede realizar con las entradas de registro externo EXT1, EXT2, EXT3 o con la fase Z de encoder A diferencia de los ejes flexibles, sólo se puede capturar la posición de registro. Cuando se ha producido un evento de registro, el parámetro de eje **MARK** se establece en ON y la posición se almacena en el parámetro de eje **REG_POS**. El comando **REGIST** permite capturar un evento de registro en la entrada de registro concreta. Cuando se ha producido un evento de registro, el parámetro de eje **MARK** se establece en ON y la posición se almacena en el parámetro de eje **REG_POS**. Las señales de registro EXT1, EXT2 y EXT3 deben estar asignada a entradas CN1 con el parámetro del servo driver Pn511. Por ejemplo, Pn511 = 654x establece las conexiones de EXT1 al pin 44 de CN1, EXT2 al pin 45 de CN1 y EXT3 al pin 46 de CN1. En la tabla siguiente se muestra cómo configurar las entradas externas individualmente. Nota: para configurar las señales de registro EXT1, EXT2 y EXT3 se utilizan los números de parámetro Pn511.1, Pn511.2 y Pn511.3, respectivamente. Pn511.0 no se utiliza. Consulte el manual del usuario del servodriver para obtener más detalles.

Señal de registro	Número de parámetro	Valor de parámetro	Descripción
EXT 1	Pn511.1	0 a 3	No se utiliza.
		4	Entrada del pin 44 de CN1 (flanco ascendente).
		5	Entrada del pin 45 de CN1 (flanco ascendente).
		6	Entrada del pin 46 de CN1 (flanco ascendente).
		7	La señal siempre está en OFF.
		8	La señal siempre está en ON.
		9 a C	No se utiliza.
		D	Entrada del pin 44 de CN1 (flanco descendente).
		E	Entrada del pin 45 de CN1 (flanco descendente).
		F	Entrada del pin 46 de CN1 (flanco descendente).
EXT 2	Pn511.2	Como en EXT 1	Como en EXT 1
EXT 3	Pn511.3	Como en EXT 1	Como en EXT 1

Los intervalos inclusivos permiten que el registro se produzca sólo dentro de un intervalo especificado de posiciones de eje. Con esta función de intervalo, los eventos de registro se omiten si la posición medida del eje no es mayor que el parámetro de eje **OPEN_WIN** y es menor que el parámetro **CLOSE_WIN**.
Los intervalos exclusivos permiten que el registro se produzca sólo fuera del intervalo especificado de posiciones de eje. Con esta función de intervalo, los eventos de registro se omiten si la posición medida del eje no es menor que el parámetro de eje **OPEN_WIN** y es mayor que el parámetro **CLOSE_WIN**.

Argumentos
- **modo**

El parámetro de modo especifica la entrada de registro y el evento que se utilizarán así como el flanco de señal en el que se produce el evento de registro. El modo de parámetro también especifica el uso de la función de intervalo y el filtrado.
El parámetro de modo varía entre MECHATROLINK-II y el eje flexible. Las funciones para cada bit en el parámetro de modo se explican en las tablas siguientes.

Ejemplo **REGIST(4 + 1) AXIS (1)**
Este comando selecciona el evento de registro principal que se produce en el flanco ascendente de la señal de entrada REG 0 para el eje 1.

Ejemplo **REGIST(48 + 64 + 128 + 512 + 1024) AXIS(2)**
Este comando selecciona el evento de registro secundario que se produce en el flanco descendente de la señal de entrada AUX IN con la función de intervalo inclusivo y filtrado para el eje 2.

Consulte también **AXIS**, **MARK**, **MARKB**, **REG_POS**, **REG_POSB**, **OPEN_WIN**, **CLOSE_WIN**.

Bit	Función (eje flexible)
1, 0	El registro principal se produce para: • 00: marca Z del encoder • 01: entrada REG 0 • 10: entrada REG 1 • 11: entrada AUX IN
2	Establezca este bit para utilizar el evento de registro principal
3	El evento de registro principal se produce en la señal: • 0: flanco ascendente • 1: flanco descendente
5, 4	El registro secundario se produce para: • 00: marca Z del encoder • 01: entrada REG 0 • 10: entrada REG 1 • 11: entrada AUX IN
6	Establezca este bit para utilizar el evento de registro secundario
7	El evento de registro secundario se produce en la señal: • 0: flanco ascendente • 1: flanco descendente
9, 8	Opción de función de intervalo: • 00: sin intervalo • 01: intervalo inclusivo • 10: intervalo inclusivo • 11: intervalo exclusivo
10	Establezca este bit para utilizar la función de filtrado

Bit	Función (MECHATROLINK-II)
1, 0	El registro principal se produce para: • 00: marca Z del encoder • 01: entrada EXT1 (pin de CN1 programado con Pn511.1) • 10: entrada EXT2 (pin de CN1 programado con Pn511.2) • 11: entrada EXT3 (pin de CN1 programado con Pn511.3)
2-7	No se utiliza
9, 8	Opción de función de intervalo: • 00: sin intervalo • 01: intervalo inclusivo • 10: intervalo inclusivo • 11: intervalo exclusivo
10	No se utiliza

3.2.220 REMAIN

Tipo: Parámetro de eje (sólo lectura)

Sintaxis: **REMAIN**

Descripción: El parámetro **REMAIN** contiene la distancia restante hasta el final del movimiento actual. Se puede consultar para averiguar la cantidad del movimiento que se ha realizado. **REMAIN** se define en unidades de usuario.

Argumentos: N/A

Ejemplo: Para cambiar la velocidad a un valor más lento a 5 mm desde el final de un movimiento:

```
inicio:
  SPEED = 10
  MOVE(45)
  WAIT UNTIL REMAIN < 5
  SPEED = 1
  WAIT IDLE
```

Consulte también: **AXIS**, **UNITS**

3.2.221 REMOTE_ERROR

Tipo: Parámetro de eje

Sintaxis: **REMOTE_ERROR**

Descripción: Devuelve el número de errores en el vínculo de comunicaciones digitales de un driver.

Argumentos: N/A

Ejemplo:

```
>>PRINT REMOTE_ERROR
1.0000
```

Consulte también: N/A

3.2.222 RENAME

Tipo: Comando de programa

Sintaxis: **RENAME "nombre_programa_anterior" "nombre_programa_nuevo"**

Descripción: El comando **RENAME** cambia el nombre de un programa del directorio de la TJ1-MC__. Los nombres de programa también se pueden especificar sin comillas.
Nota: Este comando se ha implementado para un terminal (VT100) offline. En Trajexia Tools, los usuarios pueden seleccionar el comando desde el menú **Program** (Programa).

Argumentos:
- **nombre_programa_anterior**
  El nombre actual del programa.
- **nuevo_nombre_programa**
  El nombre nuevo del programa.

Ejemplo: **RENAME "coche" "automóvil"**

Consulte también: **COPY**, **DEL**, **NEW**.

3.2.223 REP_DIST

Tipo	Parámetro de eje
Sintaxis	**REP_DIST**
Descripción	El parámetro **REP_DIST** contiene la distancia de repetición, que es el rango permitido de movimiento para un eje antes de que se corrijan la posición solicitada (**DPOS**) y la posición medida (**MPOS**). **REP_DIST** se define en unidades de usuario. El rango exacto se controla mediante **REP_OPTION**. El parámetro **REP_DIST** puede tener cualquier valor positivo distinto de 0. Cuando la posición medida ha alcanzado su límite, la TJ1-MC__ ajustará las posiciones absolutas sin afectar al movimiento en curso o el algoritmo de servo. Tenga en cuenta que la posición solicitada puede estar fuera del rango porque la posición medida se utiliza para activar el ajuste. Para cada caso (**DEFPOS**, **OFFPOS**, **MOVEABS**, **MOVEMODIFY**) que define una posición fuera del rango, la posición final se redefinirá dentro del rango. El valor predeterminado para todos los ejes es 5.000.000.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**, **DPOS**, **MPOS**, **REP_OPTION**, **UNITS**.

3.2.224 REP_OPTION

Tipo	Parámetro de eje
Sintaxis	**REP_OPTION**
Descripción	El parámetro **REP_OPTION** controla la aplicación del parámetro de eje **REP_DIST** y la opción de repetición de los comandos de eje **CAMBOX** y **MOVELINK**. El valor predeterminado es 0. Consulte la tabla siguiente.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**, **CAMBOX**, **MOVELINK**, **REP_DIST**.

Bit	Descripción
0	El rango de distancia repetida se controla mediante el bit 0 del parámetro **REP_OPTION**. • Si el bit 0 de **REP_OPTION** está en OFF, el rango de las posiciones solicitada y media estará entre **-REP_DIST** y **REP_DIST**. • Si el bit 0 de **REP_OPTION** está en ON, el rango de las posiciones solicitada y media estará entre 0 y **REP_DIST**.
1	La opción de repetición automática de los comando **CAMBOX** y **MOVELINK** se controlan mediante el bit 1 del parámetro **REP_OPTION**. El bit se establece en ON para solicitar al software del sistema que finalice la opción de repetición automática. Cuando el software del sistema ha establecido la opción en OFF, borra automáticamente el bit 1 de **REP_OPTION**.

3.2.225 REPEAT..UNTIL

Tipo	Comando de control de programa
Sintaxis	**REPEAT** **comandos** **UNTIL condición**

Descripción	La estructura **REPEAT ... UNTIL** permite que el segmento de programa entre las instrucciones **REPEAT** y **UNTIL** se repita una serie de veces hasta que la condición sea **TRUE**. Nota: la construcción **REPEAT ... UNTIL** se puede anidar indefinidamente.
Argumentos	• **comandos** Cualquier conjunto válido de comandos BASIC. • **condición** Cualquier expresión lógica BASIC válida.
Ejemplo	Una cinta transportadora cataloga cada 100 mm a una velocidad de 1.000 mm/s, espera 0,5 s y, a continuación, repite el ciclo hasta que un contador señaliza que se pare activando la entrada 4. ciclo: **SPEED = 1000** **REPEAT** **MOVE(100)** **WAIT IDLE** **WA(500)** **UNTIL IN(4) = ON**
Consulte también	**FOR..TO..STEP..NEXT**, **WHILE..WEND**.

3.2.226 RESET

Tipo	Comando de sistema
Sintaxis	**RESET**
Descripción	El comando **RESET** establece el valor de todas las variables locales de la tarea BASIC actual en 0.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**CLEAR**.

3.2.227 RETURN

Consulte **GOSUB..RETURN**.

3.2.228 REV_IN

Tipo	Parámetro de eje
Sintaxis	**REV_IN**
Descripción	El parámetro de eje **REV_IN** contiene el número de entrada que se utilizará como entrada de límite de marcha inversa. El número se puede establecer de 0 a 7 y 19. El rango de entrada válido va de 0 a 31. Los valores de 0 a 15 representan entradas presentes físicamente del conector de E/S de la TJ1-MC__ y son comunes para todos los ejes. Los valores 16 a 27 representan entradas de software que se pueden utilizar libremente en programas y comandos, como, por ejemplo, IN y OP. También son comunes para todos los ejes. Los valores 28 a 31 se asignan directamente a las entradas de driver presentes en el conector CN1 y son únicos para cada eje. Las entradas de driver que se asignan a las entradas 28 a 31 dependen del ajuste Pn81E del parámetro de servodriver. El ajuste recomendado es Pn81E = 0x4321, con la siguiente asignación: Si se establece un número de entrada y se alcanza el límite, se parará cualquier movimiento inverso en dicho eje. También se establecerá el bit 5 del parámetro de eje **AXISSTATUS**. Nota: esta entrada se activa por baja.
Sigma II	• entrada 28: CN1-40 • entrada 29: CN1-41 • entrada 30: CN1-42 • entrada 31: CN1-43
Sigma III	• entrada 28: CN1-13 • entrada 29: CN1-7 • entrada 30: CN1-8 • entrada 31: CN1-9
Junma	• entrada 26: CN1-2 • entrada 27: CN1-1
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**, **AXISSTATUS**, **FWD_IN**.

3.2.229 REV_JOG

Tipo	Parámetro de eje
Sintaxis	**REV_JOG**
Descripción	El parámetro **REV_JOG** contiene el número de entrada que se utilizará como entrada de marcha inversa jog. La entrada puede ser de 0 a 7. De forma predeterminada, el parámetro se establece en –1 y no hay seleccionada ninguna entrada. Nota: esta entrada se activa por baja.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**, **FAST_JOG**, **FWD_JOG**, **JOGSPEED**, **UNITS**.

3.2.230 REVERSE

Tipo	Comando de eje
Sintaxis	**REVERSE** **RE**
Descripción	El comando **REVERSE** mueve un eje continuamente hacia atrás a la velocidad definida en el parámetro **SPEED**. La velocidad de aceleración se define mediante el parámetro de eje **ACCEL**. **REVERSE** funciona en el eje base predeterminado (establecido con **BASE**) a menos que se utilice **AXIS** para especificar un eje base temporal. Nota: El movimiento inverso se puede parar si se ejecuta el comando **CANCEL** o **RAPIDSTOP**, o si se alcanza el límite de marcha inversa, el límite de inhibición o el límite de retorno al origen.
Argumentos	N/A
Ejemplo	**volver:** **REVERSE** **WAIT UNTIL IN(0) = ON ' Esperar la señal de parada** **CANCEL**
Consulte también	**AXIS**, **CANCEL**, **FORWARD**, **RAPIDSTOP**.

3.2.231 RS_LIMIT

Tipo	Parámetro de eje
Sintaxis	**RS_LIMIT** **RSLIMIT**
Descripción	El parámetro **RS_LIMIT** contiene la posición absoluta del límite de software de marcha inversa en unidades de usuario. Se puede establecer un límite de software para movimiento inverso desde el programa para controlar el rango de trabajo de la máquina. Cuando este límite se alcanza, al TJ1-MC__ decelerará hasta 0 y, a continuación, cancelará el movimiento. El bit 10 del parámetro de eje **AXISSTATUS** se pondrá en ON mientras la posición de eje sea menor que **RS_LIMIT** o esté por debajo de ese valor.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**, **FS_LIMIT**, **UNITS**.

3.2.232 RUN

Tipo	Comando de programa
Sintaxis	**RUN ["nombre_programa" [, número_tarea]]**
Descripción	El comando **RUN** ejecuta el programa de la TJ1-MC__ especificado con **nombre_programa**. **RUN** con la especificación de nombre de programa ejecutará el programa seleccionado actual. El nombre de programa también se puede especificar sin comillas. El número de tarea especifica dónde se ejecutará el programa. Si se omite el número de tarea, el programa se ejecuta en la tarea más alta disponible. **RUN** se puede incluir en un programa para ejecutar otro programa. Nota: La ejecución continúa hasta que se produce una de las siguientes situaciones: • No hay más líneas para ejecutar. • **HALT** se escribe en la línea de comandos para parar todos los programas. • **STOP** se escribe en la línea de comandos para parar un solo programa. • Se encuentra el comando **STOP** en el programa. • Se produce un error de tiempo de ejecución.

Argumentos
- **nombre_programa**
  Cualquier nombre de programa válido.
- **número_tarea**
  Cualquier número de tarea válido. Rango: [1,14].

Ejemplo
>> SELECT "PROGRAMA"
PROGRAMA seleccionado
>> RUN
En este ejemplo se ejecuta el programa seleccionado actualmente.

Ejemplo
RUN "salchicha"
En este ejemplo se ejecuta el programa denominado **salchicha**.

Ejemplo
RUN "salchicha",3
En este ejemplo se ejecuta el programa denominado **salchicha** en la tarea 3.

Consulte también
HALT, **STOP**.

3.2.233 RUN_ERROR

Tipo
Parámetro de tarea (sólo lectura)

Sintaxis
RUN_ERROR

Descripción
El parámetro **RUN_ERROR** contiene el número del último error BASIC en tiempo de ejecución que se ha producido en la tarea especificada. Cada tarea tiene su propio parámetro **RUN_ERROR**. Utilice el modificador **PROC** para acceder al parámetro para una determinada tarea. Sin **PROC** se supondrá la tarea actual.

Argumentos
N/A

Ejemplo
>> PRINT RUN_ERROR PROC(5)
9.0000

Consulte también
BASICERROR, **ERROR_LINE**, **PROC**.

3.2.234 RUNTYPE

Tipo
Comando de programa

Sintaxis
RUNTYPE "nombre_programa", ejecución_automática [, número_tarea]

Descripción
El comando **RUNTYPE** determina si el programa, especificado por **nombre_programa**, se ejecuta automáticamente al inicio o no se ejecuta, y la tarea en que se ejecutará. El número de tarea es opcional. Si se omite, el programa se ejecutará en la tarea más alta disponible.
El estado **RUNTYPE** actual de cada programa se muestra cuando se ejecuta un comando **DIR**. Si un programa tiene errores de compilación, no se ejecutará ningún programa al inicio. Para establecer **RUNTYPE** con Trajexia Tools, seleccione **Set Power-up mode** (Configurar modo de encendido) en el menú **Program** (Programa).
Nota: La ejecución del comando **EPROM** es necesario para almacenar los nuevos ajustes de **RUNTYPE** en la memoria flash. De lo contrario, los nuevos ajustes se perderán cuando se desconecte la alimentación.

Argumentos
- **nombre_programa**
  El nombre del programa cuyo valor de **RUNTYPE** se va a establecer.
- **ejecución_automática**
  0 = ejecutar manualmente mediante comando; 1 = ejecutar automáticamente en el encendido. Todos los valores distintos de cero se considerán 1.
- **número_tarea**
  El número de la tarea en que se ejecutará el programa. Rango: [1, 14].

Ejemplo
>> RUNTYPE nombre_programa,1,3
Esta línea establece que el programa **nombre_programa** se ejecute automáticamente en el inicio en la tarea 3.

Ejemplo
>> RUNTYPE nombre_programa,0
Esta línea establece el programa **nombre_programa** para que se ejecute manualmente.

Consulte también
AUTORUN, **EPROM**, **EX**.

3.2.235 S_REF

Tipo	Parámetro de eje
Sintaxis	**DAC** **S_REF**
Descripción	Este parámetro contiene el valor de referencia de velocidad que se aplica directamente al servodriver cuando el eje está en lazo abierto (**SERVO = OFF**). El rango de este parámetro se define por el número de bits disponibles. Para los ejes MECHATROLINK, **S_REF** ocupa 32 bits, por lo que el rango disponible es [–2.147.483.648, 2.147.483.648], que corresponde a un rango de tensión [–10 V, 10 V]. Para los ejes flexibles, **S_REF** ocupa 16 bits, por lo que el rango disponible es [–32.768, 32.767], que corresponde a un rango de tensión [–10 V, 10 V]. Estos rangos se pueden limitar con el parámetro **OUTLIMIT**. La referencia de velocidad real depende del servomotor. Para determinar la referencia de velocidad en revoluciones por minuto (RPM), multiplique el valor del parámetro por el valor del parámetro **S_RATE**. El valor que se está aplicando actualmente al driver se puede leer con el parámetro de eje **S_REF_OUT**.
Argumentos	N/A
Ejemplo	**WDOG = ON** **SERVO = OFF** **cuadrado:** **S_REF AXIS(0) = 2000** **WA(250)** **S_REF AXIS(0) = –2000** **WA(250)** **GOTO cuadrado** Estas líneas se pueden utilizar para forzar una onda cuadrada de movimiento positivo y negativo con un período de 500 ms aproximadamente en el eje 0.
Consulte también	**AXIS**, **S_REF_OUT**, **OUTLIMIT**, **SERVO**.

3.2.236 S_REF_OUT

Tipo	Parámetro de eje (sólo lectura)
Sintaxis	**DAC_OUT** **S_REF_OUT**
Descripción	El parámetro **S_REF_OUT** contiene el valor de referencia de velocidad aplicado al servodriver para lazo abierto y cerrado. En lazo cerrado (**SERVO = ON**), el algoritmo de control Motion enviará una señal de referencia de velocidad determinada por los ajustes de ganancia de control y el error de seguimiento. La posición del servomotor se determina mediante los comandos de eje. En lazo abierto (**SERVO = OFF**), la señal de referencia de velocidad se determina mediante el parámetro de eje **S_REF**. La referencia de velocidad real depende del servomotor. Para determinar la referencia de velocidad en revoluciones por minuto (RPM), multiplique el valor del parámetro **S_REF** por el valor del parámetro **S_RATE**.
Argumentos	N/A
Ejemplo	**>> PRINT S_REF_OUT AXIS(0)** **288.0000**
Consulte también	**AXIS**, **S_REF**, **OUTLIMIT**, **SERVO**.

3.2.237 SCOPE

Tipo: Comando de sistema

Sintaxis: **SCOPE(control, período, inicio_tabla, parar_tabla, P0 [, P1 [, P2 [, P3]]])**

Descripción: El comando **SCOPE** programa el sistema para que almacene automáticamente hasta 4 parámetros cada período de muestreo. El almacenamiento de los datos comenzará tan pronto como se haya ejecutado el comando **TRIGGER**.
El período de muestreo puede ser cualquier múltiplo del período de servo.
Los parámetros se almacenan en la matriz TABLE y se pueden leer en un ordenador o escribir en un archivo para un análisis posterior con la opción **Create Table file** (Crear archivo de tabla) en el menú **File** (Archivo).
La posición de TABLE actual del primer parámetro que se escribe mediante **SCOPE** se puede leer del parámetro **SCOPE_POS**.
Notas:

1. Trajexia Tools utiliza el comando **SCOPE** cuando se ejecuta la función de osciloscopio.
2. Con el fin de reducir el tiempo de cálculo para escribir los datos de tiempo real, el comando **SCOPE** escribe datos sin procesar en la matriz TABLE. Por ejemplo
   a) Los parámetros se escriben en flancos de encoder (por segundo) y, por lo tanto, no se compensan por el factor de conversión **UNITS**.
   b) El parámetro **MSPEED** se escribe como el cambio en flancos de encoder por período de servo.
3. Aplicaciones como los comandos **CAM**, **CAMBOX** y **SCOPE** utilizan la misma matriz TABLE como el área de datos.

Argumentos

- **control**
  Activar o desactivar el control de la ejecución de **SCOPE**. Si se activa, el comando **SCOPE** está preparado para ejecutarse tan pronto como se ejecuta el comando **TRIGGER**.
- **período**
  El número de períodos de servo entre muestreos de datos.
- **inicio_tabla**
  La dirección del primer elemento de la matriz TABLE para empezar a almacenar datos.
- **parar_tabla**
  La dirección del último elemento de la matriz TABLE que se utilizará.
- **P0**
  Primer parámetro que se almacenará.
- **P1**
  Segundo parámetro opcional que se almacenará.
- **P2**
  Tercer parámetro opcional que se almacenará.
- **P3**
  Cuarto parámetro opcional que se almacenará.

Ejemplo: **SCOPE(ON,10,0,1000,MPOS AXIS(1),DPOS AXIS(1))**
Este ejemplo programa la función **SCOPE** para almacenar el parámetro **MPOS** para el eje 1 y el parámetro **DPOS** para el eje 1 cada 10 ciclos de servo. El parámetro **MPOS** se almacenará en las ubicaciones de TABLE 0 a 499; los parámetros **DPOS**, en las ubicaciones de TABLE 500 a 999. La función **SCOPE** volverá a almacenar en el principio hasta que se pare. El muestreo no comenzará hasta que se ejecute el comando **TRIGGER**.

Ejemplo: **SCOPE(OFF)**
La línea anterior desactiva la función de ámbito.

Consulte también: **SCOPE_POS**, **TABLE**, **TRIGGER**.

3.2.238 SCOPE_POS

Tipo	Parámetro de sistema (sólo lectura)
Sintaxis	**SCOPE_POS**
Descripción	El parámetro **SCOPE_POS** contiene la posición TABLE actual en la que el comando **SCOPE** almacena actualmente su primer parámetro.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**SCOPE**.

3.2.239 SELECT

Tipo	Comando de programa
Sintaxis	**SELECT "nombre_programa"**
Descripción	El comando **SELECT** especifica el programa actual para editar, ejecutar, listar, etc. **SELECT** crea un nuevo programa si el nombre introducido no existe. El nombre de programa también se puede especificar sin comillas. Cuando se selecciona un programa, los comandos **COMPILE**, **DEL**, **EDIT**, **LIST**, **NEW**, **RUN**, **STEPLINE**, **STOP** y **TROFF** se aplicarán al programa seleccionado actualmente a menos que se especifique un programa en la línea de comandos. Cuando se selecciona otro programa, se compilará el seleccionado anteriormente. El programa seleccionado no se puede cambiar cuando se está ejecutando un programa. Nota: Este comando se ha implementado para un terminal (VT100) offline. Trajexia Tools selecciona automáticamente los programas cuando se hace clic en su entrada en lista del panel de control.
Argumentos	N/A
Ejemplo	**>> SELECT "PROGRAMA"** **PROGRAMA seleccionado** **>> RUN**
Consulte también	**COMPILE**, **DEL**, **EDIT**, **LIST**, **NEW**, **RUN**, **STEPLINE**, **STOP**, **TROFF**.

3.2.240 SERVO

Tipo	Parámetro de eje
Sintaxis	**SERVO**
Descripción	El parámetro **SERVO** determina si el eje base funciona bajo servocontrol (**SERVO = ON**) o lazo abierto (**SERVO = OFF**). En lazo cerrado, el algoritmo de control enviará una señal de referencia de velocidad determinada por los ajustes de ganancia de control y el error de seguimiento. La posición del servomotor se determina mediante los comandos de eje. En lazo abierto, la señal de referencia de velocidad se determina completamente mediante el parámetro de eje **S_REF**.
Argumentos	N/A
Ejemplo	**SERVO AXIS(0) = ON ' El eje 0 está bajo servocontrol** **SERVO AXIS(1) = OFF ' El eje 1 funciona en lazo abierto**
Consulte también	**AXIS**, **FE_LIMIT**, **S_REF**, **S_REF_OUT**, **WDOG**.

3.2.241 SERVO_PERIOD

Tipo	Parámetro de sistema
Sintaxis	**SERVO_PERIOD**
Descripción	El parámetro **SERVO_PERIOD** establece el período de ciclo de servo de la TJ1-MC__. La temporización de la ejecución de las tareas de programa y la actualización de los datos de control y las E/S de la unidad dependen de este ajuste. El parámetro se define en microsegundos. La TJ1-MC__ se puede configurar en ciclo de servo de 0,5, 1,0 ó 2,0 ms. Consulte la siguiente tabla.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**DRIVE_RESET**.

Valor	Descripción
500	0,5 ms
1.000	1,0 ms
2.000	2,0 ms

Precaución
Cuando se ha configurado el parámetro, se debe realizar un apagado o reset por software (con **EX**) del sistema. En caso de no hacerlo puede producirse un comportamiento indefinido.

3.2.242 SET_BIT

Tipo: Comando de sistema

Sintaxis: **SET_BIT(número_bit, número_vr)**

Descripción: El comando **SET_BIT** establece el bit especificado de la variable VR especificada en uno. Los demás bits de la variable mantendrán sus valores.

Argumentos:
- **número_bit**
  El número del bit que se establecerá. Rango: [0,23].
- **número_vr**
  El número de la variable VR cuyo bit se establecerá. Rango: [0,1.023].

Ejemplo: No hay ejemplo.

Consulte también: **CLEAR_BIT**, **READ_BIT**, **VR**.

3.2.243 SETCOM

Tipo: Comando de comunicaciones

Sintaxis: **SETCOM(velocidad_baudios, bits_datos, bits_parada, paridad, número_puerto, modo)**

Descripción: El comando **SETCOM** establece las comunicaciones de los puertos serie. El comando activará los protocolos Host Link o definirá la comunicación de propósito general.
Los puertos serie tiene 9.600 baudios, 7 bits de datos, 2 bits de parada, paridad par y XON/XOFF activados para la comunicación de propósito general de forma predeterminada. Estos ajustes predeterminados se recuperan al inicio.

Argumentos:
- **velocidad_baudios**
  1.200, 2.400,4.800, 9.600,19.200, 38.400
- **bits_datos**
  7, 8
- **bits_parada**
  1, 2
- **paridad**
  0 = ninguna; 1 = impar; 2 = par.
- **número_puerto**
  Consulte la primera tabla más abajo.
- **modo**
  Seleccione uno de los modos enumerados en la segunda tabla de más abajo para los puertos serie 1 y 2.

Ejemplo: No hay ejemplo.

Consulte también: N/A

valor_número_puerto	Descripción
1	Puerto 1 comunicación serie RS-232C
2	Puerto 2 comunicación serie RS-422A/485

Modo	Descripción
0	Comunicaciones de propósito general (sin mecanismo XON/XOFF)

Modo	Descripción
5	Protocolo de esclavo Host Link
6	Protocolo maestro Host Link

3.2.244 SGN

Tipo	Función matemática
Sintaxis	**SGN(expresión)**
Descripción	La función **SGN** devuelve el signo de un número. Devuelve el valor 1 para valores positivos (incluido el 0) y –1 para valores negativos.
Argumentos	• **expresión** Cualquier expresión BASIC válida.
Ejemplo	**>> PRINT SGN(-1.2)** **–1.0000**
Consulte también	N/A

3.2.245 SIN

Tipo	Función matemática
Sintaxis	**SIN(expresión)**
Descripción	La función **SIN** devuelve el seno de la expresión. Los valores de entrada se expresan en radianes y pueden tener cualquier valor. El valor de resultado estará en el rango de –1 a 1.
Argumentos	• **expresión** Cualquier expresión BASIC válida.
Ejemplo	**>> PRINT SIN(PI/2)** **1.0000**
Consulte también	N/A

3.2.246 SLOT

Tipo	Modificador de entrada
Sintaxis	**SLOT**
Descripción	El modificador especifica el número de unidad para un parámetro como **COMMSTYPE**. Los números de unidad de Trajexia van de 0 a 6.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	N/A

3.2.247 SPEED

Tipo	Parámetro de eje
Sintaxis	**SPEED**
Descripción	El parámetro **SPEED** contiene la velocidad solicitada en unidades/s. Puede tener cualquier valor positivo (incluido el 0). La velocidad solicitada es la máxima para los comandos de movimiento con perfil de velocidad.
Argumentos	N/A
Ejemplo	**SPEED = 1000** **PRINT "Establecer velocidad = ";SPEED**
Consulte también	**ACCEL**, **AXIS**, **DATUM**, **DECEL**, **FORWARD**, **MOVE**, **MOVEABS**, **MOVECIRC**, **MOVEMODIFY**, **REVERSE**, **UNITS**.

3.2.248 SQR

Tipo	Función matemática
Sintaxis	**SQR(expresión)**
Descripción	La función **SQR** devuelve la raíz cuadrada de la expresión. La expresión debe tener un valor positivo (incluido el 0).
Argumentos	• **expresión** Cualquier expresión BASIC válida.
Ejemplo	**>> PRINT SQR(4)** **2.0000**
Consulte también	N/A

3.2.249 SRAMP

Tipo	Parámetro de eje
Sintaxis	**SRAMP**
Descripción	El parámetro **SRAMP** contiene el factor de curva S. El factor de curva S controla la cantidad de redondeo aplicado a los perfiles trapezoidales. Con el valor 0 no se establece redondeo. Con el valor 10 se establece el redondeo máximo. El valor predeterminado del parámetro es 0. **SRAMP** se aplica a los comandos **FORWARD**, **MOVE**, **MOVEABS**, **MOVECIRC**, **MHELICAL** y **REVERSE**. Notas: • El uso de curvas S aumenta el tiempo necesario para que se complete el movimiento. • El factor de curva S no se debe cambiar mientras esté en curso un movimiento.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**AXIS**.

3.2.250 PASO

Consulte **FOR..TO..STEP..NEXT**.

3.2.251 STEP_RATIO

Tipo	Comando de eje
Sintaxis	**STEP_RATIO(contaje_salida, contaje_posición_solicitada)**
Descripción	Este comando configura una relación para la salida de pulsos de un eje de motor paso a paso. Cada período de servo, se envía el número de pasos mediante la función **STEP_RATIO** antes de que vaya a la salida de pulsos de paso. Salida de contaje de pulsos = (numerador)/(denominador) * **MPOS**. **STEP_RATIO** afecta a **MOVECIRC** y **CAMBOX**. Notas: • La función **STEP_RATIO** actúa antes de dividir entre 16 el factor en el eje paso a paso. • Se deben evitar relaciones grandes ya que provocarán la pérdida de resolución o una suavidad más reducida en el movimiento. El tamaño de paso físico real x 16 es la resolución básica del eje y el uso de este comando puede reducir la capacidad del controlador Motion para alcanzar todas las posiciones de forma precisa. • **STEP_RATIO** no reemplaza a **UNITS**. No utilice **STEP_RATIO** para eliminar el factor x16 en el eje paso a paso ya que se producirá un control de frecuencia de paso deficiente.
Argumentos	• **denominador** Un número entero entre 0 y 16.777.215 que se utiliza para definir el denominador en la ecuación anterior. • **numerador** Un número entero entre 0 y 16.777.215 que se utiliza para definir el numerador en la ecuación anterior.

Ejemplo Dos ejes están configurados como X e Y, pero los pasos de los ejes por mm no son los mismos. Los movimientos interpolados requieren valores de UNITS idénticos en ambos ejes para mantener la velocidad de trayectoria constante y para que MOVECIRC funcione correctamente. El eje con la resolución menor se cambia para que coincida con el eje de resolución de paso mayor y mantener la mejor precisión para ambos ejes.

```
' Eje 0: 500 contajes por mm (31,25 pasos por mm)
' Eje 1: 800 contajes por mm (50,00 pasos por mm)
BASE(0)
STEP_RATIO(500,800)
UNITS = 800
BASE(1)
UNITS = 800
```

Consulte también N/A

3.2.252 STEPLINE

Tipo Comando de programa

Sintaxis **STEPLINE ["nombre_programa" [, número_tarea]]**

Descripción El comando **STEPLINE** ejecuta una línea (es decir, “pasos”) del programa especificado por **nombre_programa**. El nombre de programa también se puede especificar sin comillas. Si **STEPLINE** se ejecuta sin nombre de programa en la línea de comandos, se irá paso a paso en el programa seleccionado actual. Si **STEPLINE** se ejecuta sin nombre de programa en un programa, se irá paso a paso en dicho programa.
Si el programa se especifica, se irá paso a paso en todas las apariciones de este programa. Se iniciará una nueva tarea cuando no haya copia del programa en ejecución. Si la tarea también se especifica, sólo se irá paso a paso en el programa que se ejecuta en la tarea especificada. Si no hay ninguna copia del programa que se ejecuta en la tarea especificada, se iniciará uno en ella.

Argumentos
- **nombre_programa**
  El nombre del programa en el que se irá paso a paso.
- **número_tarea**
  El número de la tarea con el programa en el que se irá paso a paso.
  Rango: [1,14].

Ejemplo **>> STEPLINE "cinta"**

Ejemplo **>> STEPLINE "matemáticas",2**

Consulte también **RUN**, **SELECT**, **STOP**, **TROFF**, **TRON**.

3.2.253 STOP

Tipo Comando de programa

Sintaxis **STOP ["nombre_programa" [, número_tarea]]**

Descripción El comando **STOP** detendrá la ejecución del programa especificado con **nombre_programa**. Si se omite el nombre de programa, se detendrá el programa seleccionado actualmente. El nombre de programa también se puede especificar sin comillas.
En el caso de varias ejecuciones de un solo programa en distintas tareas, se puede utilizar **número_tarea** para especificar la tarea específica que se parará.

Argumentos
- **nombre_programa**
  El nombre del programa que se parará.
- **número_tarea**
  El número de la tarea con el programa que se parará.
  Rango: [1,14].

Ejemplo **>> STOP nombre_programa**

Ejemplo En este ejemplo no se ejecutarán las líneas desde etiqueta.

```
STOP
etiqueta:
  PRINT var
  RETURN
```

Consulte también **HALT**, **RUN**, **SELECT**.

3.2.254 SYSTEM_ERROR

Tipo	Parámetro de sistema (sólo lectura)
Sintaxis	**SYSTEM_ERROR**
Descripción	El parámetro **SYSTEM_ERROR** contiene los errores de sistema que se han producido en el sistema TJ1 desde la última vez que se inicializó. Los bits del parámetro **SYSTEM_ERROR** se indican en la siguiente tabla.
Argumentos	N/A.
Ejemplo	No hay ejemplo.
Consulte también	N/A

Bit	Descripción
0	Error BASIC
1	Error de batería baja
2 – 7	Reservado para uso futuro
8	Error de configuración de unidad (cualquier unidad del sistema)
9	Error de configuración de dispositivo (cualquier dispositivo del sistema)
10 – 15	Reservado para uso futuro
16	Error de pérdida de unidad (cualquier unidad del sistema)
17	Terminación no instalada
18	Error de pérdida de dispositivo (cualquier dispositivo del sistema)

3.2.255 T_REF

Tipo	Parámetro de eje
Sintaxis	**T_REF** **DAC**
Descripción	El parámetro **T_REF** contiene el valor de referencia de par que se aplicará al servomotor. El rango de este parámetro se define por el número de bits disponibles. Para los ejes MECHATROLINK, **T_REF** ocupa 32 bits, por lo que el rango disponible es [–2.147.483.648, 2.147.483.648], que corresponde a un rango de tensión [–10 V, 10 V]. Para los ejes flexibles, **T_REF** ocupa 16 bits, por lo que el rango disponible es [–32.768, 32.767], que corresponde a un rango de tensión [–10 V, 10 V]. Estos rangos se pueden limitar con el parámetro **OUTLIMIT**. La referencia de par real depende del servomotor.
Argumentos	N/A
Ejemplo	**T_REF AXIS(0) = 1000**
Consulte también	**AXIS**, **S_REF**.

3.2.256 TABLE

Tipo: Comando de sistema

Sintaxis: **TABLE(dirección, valor {, valor})**
TABLE(dirección)

Descripción: El comando **TABLE** carga datos en la matriz TABLE y los lee de ella. La matriz TABLE tiene una longitud máxima de 64.000 elementos. Los valores de TABLE son números de coma flotante con fracciones. La matriz TABLE también se puede utilizar para contener información, como una alternativa a las variables. El comando **TABLE** tiene dos formatos.

- **TABLE(dirección, valor{, valor})** escribe una secuencia de valores en la matriz TABLE. La ubicación del elemento se especifica por dirección. La secuencia puede tener una longitud máxima de 20 elementos.
- **TABLE(dirección)** devuelve el valor de TABLE en dicha entrada.

Un valor de la matriz TABLE puede ser de sólo lectura si un valor de dicho número o superior se ha escrito anteriormente en la matriz TABLE. Por ejemplo, la impresión de **TABLE(1001)** provocaría un mensaje de error si la ubicación de TABLE más alta que se ha escrito en TABLE es la ubicación 1.000. El tamaño de TABLE total se indica mediante el parámetro **TSIZE**. Tenga en cuenta que este valor es uno más que el más alto que la dirección de elemento definida. La matriz TABLE se puede eliminar con **DEL "TABLE"** o **NEW "TABLE"** en la línea de comandos.

Notas:

- Aplicaciones como los comandos **CAM**, **CAMBOX** y **SCOPE** en Trajexia Tools utilizan la misma matriz TABLE como el área de datos. No utilice la misma área de datos para propósitos distintos.
- A los datos de TABLE y VR se puede acceder desde todas las distintas tareas en ejecución. Para evitar problemas producidos porque dos tareas de programa escriban inesperadamente en una variable global, escriba los programas de modo que sólo un programa escriba en la variable global cada vez.
- Los datos TABLE y VR en la RAM se perderán cuando se desconecte la alimentación.

Argumentos:

- **dirección**
  La primera ubicación de la matriz TABLE que se leerá o escribirá.
  Rango: [0,63.999]
- **valor**
  El valor que se escribirá en la ubicación indicada y en las posteriores.

Ejemplo: **TABLE(100,0,120,250,370,470,530,550)**
La línea anterior carga una tabla interna del modo que se indica a continuación.

Ejemplo: La siguiente línea imprimirá el valor en la ubicación 1.000.
>> PRINT TABLE(1000)

Consulte también: **CAM**, **CAMBOX**, **DEL**, **NEW**, **SCOPE**, **TSIZE**, **VR**.

Entrada de tabla	Valor
100	0
101	120
102	250
103	370
104	470
105	530
106	550

3.2.257 TABLEVALUES

Tipo: Comando de sistema

Sintaxis: **TABLEVALUES(dirección, número_de_puntos, formato)**

Descripción: Devuelve una lista de puntos de TABLE empezando en el número especificado. Por el momento sólo se admite un formato y es el texto delimitado por comas. Nota: **TABLEVALUES** se proporciona principalmente para que Trajexia Tools permita el acceso rápido a los bancos de los valores de TABLE.

Argumentos:

- **dirección**
  Número del primer punto que se devolverá
- **número_de_puntos**
  El número total de puntos que se devolverán
- **formato**
  Formato para la lista

Ejemplo: No hay ejemplo.

Consulte también: N/A

3.2.258 TAN

Tipo	Función matemática
Sintaxis	**TAN(expresión)**
Descripción	La función **TAN** devuelve la tangente de la expresión. Se supone que la expresión está en radianes.
Argumentos	• **expresión** Cualquier expresión BASIC válida.
Ejemplo	**>> print TAN(PI/4)** **1.0000**
Consulte también	N/A

3.2.259 THEN

Consulte **IF..THEN..ELSE..ENDIF**.

3.2.260 TICKS

Tipo	Parámetro de tarea
Sintaxis	**TICKS**
Descripción	El parámetro **TICKS** contiene el contaje actual de los impulsos del reloj de tareas. **TICKS** es un contador de 32 bits que disminuye en cada ciclo de servo. El valor de **TICKS** se puede escribir y leer. Se puede utilizar para medir tiempos de ciclo, agregar retardos de tiempo, etc. Cada tarea tiene su propio parámetro **TICKS**. Utilice el modificador **PROC** para acceder al parámetro para una determinada tarea. Sin **PROC** se supondrá la tarea actual.
Argumentos	N/A

Ejemplo	**retardo:**
	TICKS = 3000
	OP(9,ON)
	prueba:
	IF TICKS <= 0 THEN
	OP(9,OFF)
	ELSE
	GOTO prueba
	ENDIF

Consulte también	N/A

3.2.261 TIME

Tipo	Parámetro de sistema
Sintaxis	**TIME**
Descripción	Devuelve tiempo del reloj de tiempo de real. El tiempo devuelto es el número de segundos transcurridos desde medianoche 00:00:00.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	N/A

3.2.262 TIME$

Tipo	Comando de sistema
Sintaxis	**TIME$**
Descripción	Imprimir la hora actual tal como la define el reloj de tiempo real como una cadena con formato de 24 horas.
Argumentos	N/A
Ejemplo	**>>? TIME$** **14/39/02**
Consulte también	N/A

3.2.263 TO

Consulte **FOR..TO..STEP..NEXT**.

3.2.264 TRANS_DPOS

Tipo	Parámetro de eje (sólo lectura)
Sintaxis	**TRANS_DPOS**
Descripción	Posición solicitada de eje en la salida de la transformación de trama. **TRANS_DPOS** normalmente es igual a **DPOS** en cada eje. Por lo tanto, la transformación de trama es equivalente a 1:1 para cada eje. Para algunas configuraciones de maquinaria puede resultar útil instalar una transformación de trama que no sea 1:1. Normalmente son máquinas como brazos robóticos o máquinas con movimientos parasitarios en los ejes. Las transformaciones de trama se tienen que escribir específicamente en lenguaje C y se deben descargar en el controlador. Es esencial ponerse en contacto con OMRON si se desea instalar transformaciones de trama.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**FRAME**.

3.2.265 TRIGGER

Tipo	Comando de sistema
Sintaxis	**TRIGGER**
Descripción	El comando **TRIGGER** inicia un comando **SCOPE** configurado anteriormente. Nota: Trajexia Tools utiliza **TRIGGER** automáticamente para su función de osciloscopio.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**SCOPE**.

3.2.266 TROFF

Tipo	Comando de programa
Sintaxis	**TROFF ["nombre_programa"]**
Descripción	El comando **TROFF** suspende un seguimiento en la línea actual y reanuda la ejecución normal del programa especificado con **nombre_programa**. El nombre de programa también se puede especificar sin comillas. Si se omite el nombre de programa, se supondrá el programa seleccionado.
Argumentos	• **nombre_programa** El nombre del programa del que se suspenderá el seguimiento.
Ejemplo	**>> TROFF "líneas"**
Consulte también	**SELECT**, **TRON**.

3.2.267 TRON

Tipo	Comando de programa
Sintaxis	**TRON**
Descripción	El comando **TRON** crea un punto de ruptura en un programa que suspenderá su ejecución en la línea siguiente al comando **TRON**. A continuación, el programa se puede ejecutar línea a línea con el comando **STEPLINE**. Notas: • La ejecución del programa se puede reanudar sin utilizar el comando **STEPLINE** mediante la ejecución del comando **TROFF**. • El modo de seguimiento se puede parar con un comando **STOP** o **HALT**. • Trajexia Tools resalta las líneas que contienen **TRON** en las ventanas **Edit** (Edición) y **Debug** (Depuración).
Argumentos	N/A
Ejemplo	**TRON** **MOVE(0,10)** **MOVE(10,0)** **TRON** **MOVE(0,-10)** **MOVE(-10,0)**
Consulte también	**SELECT**, **TROFF**.

3.2.268 TRUE

Tipo	Constante (sólo lectura)
Sintaxis	**TRUE**
Descripción	La constante **TRUE** devuelve el valor numérico –1.
Argumentos	N/A

Ejemplo	**prueba:**
	t = IN(0) AND IN(2)
	IF t = TRUE THEN
	PRINT "Las entradas están en ON"
	ENDIF

Consulte también	N/A

3.2.269 TSIZE

Tipo	Parámetro de sistema (sólo lectura)
Sintaxis	**TSIZE**
Descripción	El parámetro **TSIZE** devuelve el tamaño de la matriz TABLE, que es uno más que el elemento TABLE más alto definido actualmente. **TSIZE** se restablece a 0 cuando se elimina la matriz TABLE con **DEL "TABLE"** o **NEW "TABLE"** en la línea de comandos.
Argumentos	N/A
Ejemplo	En el siguiente ejemplo se supone que no se ha escrito una ubicación mayor que 1.000 en la matriz TABLE. **>> TABLE(1000,3400)** **>> PRINT TSIZE** **1001.0000**
Consulte también	**DEL**, **NEW**, **TABLE**.

3.2.270 UNITS

Tipo	Parámetro de eje
Sintaxis	**UNITS**
Descripción	El parámetro **UNITS** contiene el factor de conversión de unidad. El factor de conversión de unida permite que el usuario defina una unidad de usuario más cómoda, como m, mm o revoluciones de motor, mediante la especificación de la cantidad de flancos de encoder que se incluirán en una unidad de usuario. Los parámetros de eje como velocidad, aceleración o deceleración y los comandos de eje se especifican en estas unidades de usuario. Nota: El parámetro **UNITS** puede ser cualquier valor distinto de cero, pero se recomienda diseñar sistemas con un número entero de impulsos de encoder por unidad de usuario. El cambio de **UNITS** afectará a todos los parámetros de eje que dependen de **UNITS** para mantener la misma dinámica para el sistema.
Argumentos	N/A
Ejemplo	Una disposición de equipo de cables tiene una separación de 5 mm y un encoder de 1.000 revoluciones/impulso. Se deben establecer las unidades para que los movimientos se puedan especificar en mm. Con 1.000 impulsos/rev. se generarán 1.000 x 4 = 4.000 flancos/rev. Una revolución equivale a 5 mm. Por lo tanto, hay 4.000/5 = 800 flancos/mm. Así pues, **UNITS** se define del siguiente modo: **>> UNITS = 1000 * 4/5**
Consulte también	**AXIS**, **ENCODER_RATIO**.

3.2.271 UNLOCK

Consulte **LOCK**.

3.2.272 UNTIL

Consulte **REPEAT..UNTIL**.

3.2.273 VERIFY

Tipo	Parámetro de eje
Sintaxis	**VERIFY**
Descripción	El parámetro verify de eje se utiliza para seleccionar distintos modos de operación en un eje de encoder paso a paso. • **VERIFY = OFF** El circuito de contaje de encoder está conectado a las señales de hardware **STEP** y **DIRECTION**, por lo que se cuentan como si fueran señales de encoder. Esto resulta muy útil para el registro, ya que el circuito de registro puede funcionar, por tanto, en un eje paso a paso. • **VERIFY = ON** El circuito de encoder está conectado a la señal A, B, Z externa Nota: en la TJ1-FL02 cuando **VERIFY = OFF**, el circuito de contaje de encoder está configurado para aceptar las señales **STEP** y **DIRECTION** conectadas a las entradas A y B del encoder. Si **VERIFY = ON**, el circuito de encoder está configurado para la entrada de cuadratura habitual. Asegúrese de que las entradas de encoder no exceden los 5 voltios.
Argumentos	N/A
Ejemplo	**VERIFY AXIS(3) = ON**
Consulte también	N/A

3.2.274 VERSION

Tipo	Parámetro de sistema (sólo lectura)
Sintaxis	**VERSION**
Descripción	El parámetro **VERSION** devuelve el número de versión de firmware actual del sistema actual instalado en la TJ1-MC__.
Argumentos	N/A
Ejemplo	**>> PRINT VERSION** **1.6100**
Consulte también	N/A

3.2.275 VFF_GAIN

Tipo	Parámetro de eje
Sintaxis	**VFF_GAIN**
Descripción	El parámetro **VFF_GAIN** contiene la ganancia de realimentación positiva de velocidad. La aportación a la salida de la realimentación positiva de velocidad se calcula multiplicando el cambio en la posición solicitada por el valor del parámetro **VFF_GAIN**. El valor predeterminado es 0. Al añadir ganancia de realimentación positiva de velocidad a un sistema se reduce el error de seguimiento durante un movimiento al aumentar la salida proporcionalmente con la velocidad. Nota: para evitar inestabilidad, las ganancias de servo sólo se deben cambiar cuando **SERVO** está en OFF.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**D_GAIN**, **I_GAIN**, **OV_GAIN**, **P_GAIN**.

3.2.276 VP_SPEED

Tipo	Parámetro de eje (sólo lectura)
Sintaxis	**VP_SPEED**
Descripción	El parámetro **VP_SPEED** contiene la velocidad del perfil de velocidad en unidades de usuario/s. La velocidad de perfil de velocidad es una velocidad interna que se acelera o decelera a medida que se perfila el movimiento.
Argumentos	N/A
Ejemplo	**' Esperar hasta el comando de velocidad** **MOVE(100)** **WAIT UNTIL SPEED = VP_SPEED**
Consulte también	**AXIS**, **MSPEED**, **UNITS**.

3.2.277 VR

Tipo	Comando de sistema
Sintaxis	**VR(dirección)**
Descripción	El comando VR lee o escribe el valor de una variable global (VR). Estas variables VR contienen números reales y se pueden utilizar fácilmente como un elemento o como una matriz de elementos. La TJ1-MC__ tiene un total de 1.024 variables VR. Las variables VR se pueden utilizar para distintos propósitos en la programación BASIC. Las variables VR se comparten globalmente entre las tareas y se pueden utilizar para las comunicaciones entre las tareas. Notas: • A los datos de TABLE y VR se puede acceder desde todas las tareas en ejecución. Para evitar problemas producidos porque dos tareas de programa escriban inesperadamente en una misma variable global, escriba los programas de modo que sólo un programa escriba en la variable global cada vez. • Los datos TABLE y VR en la RAM se perderán cuando se desconecte la alimentación.
Argumentos	• **dirección** La dirección de la variable VR. Rango: [0,1.023].
Ejemplo	En el siguiente ejemplo, el valor 1,2555 se coloca en la variable VR 15. La variable local **val** se utiliza para asignar un nombre a la variable global localmente: **val = 15** **VR(val) = 1.2555**

Ejemplo

Un soporte de transferencia tiene 10 posiciones de colocación en una fila. Cada posición puede en cualquier momento estar llena o vacía. VR(101) a VR(110) para contener una matriz de diez unos y ceros con el fin de señalizar que las posiciones están llenas (1) o vacías (0). El soporte baja la carga en la primera posición libre. Parte del programa para llevar a cabo esta acción sería como sigue:

```
moverp:
MOVEABS(115) ' Mover a la primera posición de colocación
FOR VR(0) = 101 TO 110
IF (VR(VR(0)) = 0) THEN GOSUB cargar
MOVE(200) ' 200 es el espaciado entre posiciones
NEXT VR(0)
PRINT "Todas las posiciones están llenas"
WAIT UNTIL IN(3) = ON
GOTO moverp

cargar: ' Colocar carga en posición y marcar matriz
OP(15,OFF)
VR(VR(0)) = 1
RETURN
```

Las variables están respaldadas por una batería, por lo que el programa se podría diseñar para almacenar el estado de la máquina cuando la alimentación está desconectada. Evidentemente, sería necesario proporcionar un medio de restablecer por completo la siguiente intervención manual.

Ejemplo

```
lazo: ' Asignar VR(65) a VR(0) multiplicado por la posición medida del eje 1
VR(65) = VR(0) * MPOS AXIS(1)
PRINT VR(65)
GOTO lazo
```

Consulte también

CLEAR_BIT, **READ_BIT**, **SET_BIT**, **TABLE**.

3.2.278 VRSTRING

Tipo

Comando de sistema

Sintaxis

VRSTRING(inicio_vr)

Descripción

Combina el contenido de una matriz de variables VR() para que se puedan imprimir como una cadena de texto. Se enviarán todos los caracteres imprimibles y la cadena se terminará en el primer carácter nulo que se encuentre (es decir, VR(n) contiene 0).

Argumentos

- **inicio_vr**
  número del primer VR() de la matriz de caracteres.

Ejemplo

PRINT #5,VRSTRING(100)

Consulte también

N/A

3.2.279 WA

Tipo

Comando de sistema

Sintaxis

WA(tiempo)

Descripción

El comando **WA** pone en pausa la ejecución del programa el número de milisegundos especificado en tiempo. El comando sólo se puede utilizar en un programa.

Argumentos

- **tiempo**
  El número de segundos que se retendrá la ejecución del programa.

Ejemplo

Las siguientes líneas pondrán en ON la salida 7 dos segundos después de desconectar la salida 1.

```
OP(1,OFF)
WA(2000)
OP(7,ON)
```

Consulte también

N/A

3.2.280 WAIT IDLE

Tipo	Comando de sistema
Sintaxis	**WAIT IDLE**
Descripción	El comando **WAIT IDLE** suspende la ejecución del programa hasta que el eje base ha terminado de ejecutar su movimiento actual y cualquier movimiento del búfer. El comando sólo se puede utilizar en un programa. **WAIT IDLE** funciona en el eje base predeterminado (establecido con **BASE**) a menos que se utilice **AXIS** para especificar un eje base temporal. Nota: La ejecución de **WAIT IDLE** no significa necesariamente que el eje será estacionario en un sistema de servomotor.
Argumentos	N/A
Ejemplo	**MOVE(1000)** **WAIT IDLE** **PRINT "Movimiento realizado"** La instrucción de impresión se imprime al final del movimiento.
Ejemplo	**MOVE(1000)** **WAIT UNTIL MTYPE = 0** **PRINT "Movimiento finalizado"** La instrucción de impresión se imprime, la mayoría de las veces ANTES de que comience el movimiento y, en ocasiones, cuando el movimiento ha finalizado.
Explicación	Los programas y la secuencia Motion Control funcionan en paralelo y no están sincronizados. Un ciclo completo se puede producir antes de que el movimiento se cargue en el búfer. El programa ejecuta **MOVE(1000)** pero el movimiento no se carga en el búfer hasta el inicio de la siguiente “secuencia Motion Control” por lo que al comprobar **MTYPE = 0**, es **0** porque el movimiento TODAVÍA NO SE HA INICIADO, no porque ha finalizado.
Consulte también	**AXIS**, **WAIT LOADED**.

Nota:
WAIT IDLE es un comando diseñado especificamente para esperar hasta que el movimiento anterior haya finalizado. Se ocupa del retraso desde que se ha ejecutado el comando anterior en el programa hasta que el comando se carga correctamente en el búfer de movimiento.

3.2.281 WAIT LOADED

Tipo	Comando de sistema
Sintaxis	**WAIT LOADED**
Descripción	El comando **WAIT LOADED** suspende la ejecución del programa hasta que el eje base no tiene movimientos en búfer distintos del movimiento que se está ejecutando actualmente. El comando sólo se puede utilizar en un programa. Resulta útil para activar eventos al comienzo de un movimiento o al final cuando se guardan juntos varios movimientos en búfer. **WAIT LOADED** funciona en el eje base predeterminado (establecido con **BASE**) a menos que se utilice **AXIS** para especificar un eje base temporal.
Argumentos	N/A
Ejemplo	**' Poner en ON la salida 8 al comienzo de MOVE(500) y en OFF al final** **MOVE(800)** **MOVE(500)** **WAIT LOADED** **OP(8,ON)** **MOVE(400)** **WAIT LOADED** **OP(8,OFF)**
Consulte también	**AXIS**, **WAIT IDLE**

3.2.282 WAIT UNTIL

Tipo	Comando de sistema
Sintaxis	**WAIT UNTIL condición**
Descripción	El comando **WAIT UNTIL** evalúa repetidamente la condición hasta que es **TRUE**. Tras ello, continuará la ejecución del programa. El comando sólo se puede utilizar en un programa.
Argumentos	• **condición** Cualquier expresión lógica BASIC válida.

Ejemplo	En este ejemplo, el programa espera hasta que la posición medida en el eje 0 excede 150 y, a continuación, inicia un movimiento en el eje 1. **WAIT UNTIL MPOS AXIS(0)>150** **MOVE(100) AXIS(1)**
Ejemplo	Las expresiones evaluadas pueden ser tan complejas como se desee siempre que sigan la sintaxis BASIC, por ejemplo: **WAIT UNTIL DPOS AXIS(2) <= 0 OR IN(1) = ON** Con la línea anterior se espera hasta que la posición solicitada del eje 2 sea menor o igual a 0 ó que la entrada 1 esté en ON.
Consulte también	N/A

3.2.283 WDOG

Tipo	Parámetro de sistema
Sintaxis	**WDOG**
Descripción	El parámetro **WDOG** contiene el interruptor software que activa el servodriver con la señal de entrada **RUN** (servo ON). El servodriver activado controlará el servomotor según los valores de referencia de velocidad y par. **WDOG** se puede activar o desactivar mediante control del programa, en la línea de comandos y en el botón de control de Trajexia Tools. El servodriver se desactivará automáticamente cuando se produzca el error **MOTION_ERROR**. Los errores de movimiento se producen cuando el estado **AXISSTATUS** de uno de los ejes coincide con el ajuste de **ERRORMASK**. En este caso, el interruptor software (**WDOG**) se desactivará, el parámetro **MOTION_ERROR** tendrá el valor 1 y el parámetro **ERROR_AXIS** contendrá el número del primer eje que tenga el error. Nota: Trajexia Tools puede ejecutar automaticamente el parámetro **WDOG** cuando se haga clic en el botón **Drives Enable** (Activación de drivers) en el panel de control.
Argumentos	N/A
Ejemplo	No hay ejemplo.
Consulte también	**AXISSTATUS**, **ERROR_AXIS**, **ERRORMASK**, **MOTION_ERROR**, **SERVO**.

3.2.284 WHILE..WEND

Tipo	Comando de control de programa
Sintaxis	**WHILE condición** **Comandos** **WEND**
Descripción	La estructura **WHILE ... WEND** permite que el segmento de programa entre las instrucciones **WHILE** y **WEND** se repita una serie de veces hasta que la condición sea **FALSE**. En este caso, la ejecución del programa continuará después de **WEND**. Nota: los lazos **WHILE ... WEND** se pueden anidar sin ningún tipo de límite.
Argumentos	• **condición** Cualquier expresión lógica BASIC válida.
Ejemplo	**WHILE IN(12) = OFF** **MOVE(200)** **WAIT IDLE** **OP(10,OFF)** **MOVE(-200)** **WAIT IDLE** **OP(10,ON)** **WEND**
Consulte también	**FOR..TO..STEP..NEXT**, **REPEAT..UNTIL**

3.2.285 XOR

Tipo	Operación matemática
Sintaxis	**expresión1 XOR expresión2**
Descripción	El operador **XOR** realiza la función **XOR** lógica entre los bits correspondientes de las partes enteras de dos expresiones BASIC válidas. La función **XOR** lógica entre dos bits se define del modo que se indica en la tabla siguiente.
Argumentos	• **expresión1** Cualquier expresión BASIC válida. • **expresión2** Cualquier expresión BASIC válida.
Ejemplo	**VR(0) = 10 XOR 18** El operador **XOR** es un operador de bit y, por lo tanto, la acción binaria que se realiza es la siguiente: **01010 XOR 10010 = 11000**. El resultado es 24.
Consulte también	N/A

Bit 1	Bit 2	Resultado
0	0	0
0	1	1
1	0	1
1	1	0

4 Protocolos de comunicaciones

4.1 Interfaces disponibles

Las unidades Trajexia disponen de estas interfaces para comunicarse:

Unidad	Interfaz	Protocolo	Comentario
TJ1-MC__	Ethernet	Protocolo Trajexia Tools	Para programar, monitorizar y depurar el proyecto con Trajexia Tools.
		Servidor FINS	Para comunicarse con cualquier maestro FINS, por ejemplo, PLC, HMI u ordenador personal.
		Cliente FINS	Para comunicarse con cualquier servidor FINS, por ejemplo, PLC u otra unidad Trajexia.
	Serie	Maestro Host Link	Para comunicarse con cualquier esclavo Host Link, por ejemplo, un PLC de OMRON.
		Esclavo Host Link	Para comunicarse con cualquier maestro Host Link, normalmente HMI.
		Definido por el usuario	Este protocolo se crea y gestiona mediante comandos BASIC.
TJ1-PRT	PROFIBUS	Esclavo PROFIBUS DP-V0	Para intercambiar variables de canal con cualquier maestro PROFIBUS.
TJ1-DRT	DeviceNet	DeviceNet	Para intercambiar variables de canal con cualquier maestro DeviceNet.
TJ1-ML__	MECHATROLINK	MECHATROLINK	Para comunicar con los esclavo MECHATROLINK admitidos. Este protocolo es transparente para el usuario.

4.2 Ethernet

La TJ1-MC__ tiene un puerto Ethernet estándar de 10/100 Mbps. Puede utilizar un cable Ethernet plano o cruzado para conectar la TJ1-MC__ a un PC. Para configurar esta interfaz, establezca estos parámetros:

Elemento	Valor predeterminado	Comentario
Dirección IP	192.168.0.250	Establezca una dirección IP que sea única en la red.
Máscara de subred	255.255.255.0	Establezca la misma subred que utiliza la LAN.
Gateway	0.0.0.0	El gateway es necesario para tener acceso remoto desde otra LAN.

Asegúrese de que la dirección IP del PC está en el mismo rango que la TJ1-MC__: si la dirección IP de la TJ1-MC__ es *aaa.bbb.ccc.ddd*, la dirección IP del PC debe ser *aaa.bbb.ccc.xxx*, donde *xxx* va de 000 a 255 y es distinto de *ddd*.
Puede cambiar la dirección IP de la TJ1-MC__ para que coincida con la dirección del PC si se conecta a él mediante un concentrador o conmutador de red. Por ejemplo, si la dirección IP del PC es 192.200.185.001, puede establecer la dirección IP de la TJ1-MC__ en 192.200.185.002.

Nota
La TJ1-MC__ no dispone de funcionalidad DHCP y, por lo tanto, no puede asignar una dirección IP a un PC.

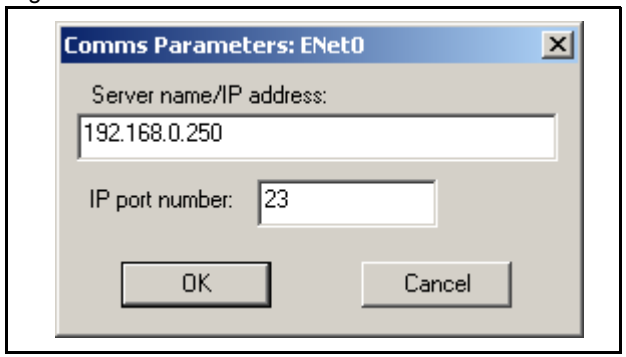
La submáscara de red de la TJ1-MC__ es genérica. No es necesario que coincida con la máscara de subred del PC.
Utilice el comando **Ethernet** para leer o escribir los ajustes de Ethernet.
Es necesario apagar y volver a encender las unidades para que los cambios surtan efecto.
Puede consultar la dirección IP de la TJ1-MC__ con la línea de comandos de Trajexia Tools y con el comando **Ethernet**: escriba el comando **Ethernet(0, -1, 0)** en la línea de comandos y se mostrará en ella la dirección IP de la TJ1-MC__.

Nota
Tiene que desconectar y conectar la alimentación del sistema Trajexia antes de que el cambio de la dirección IP surta efecto.

4.2.1 Comunicación con Trajexia directamente desde el ordenador

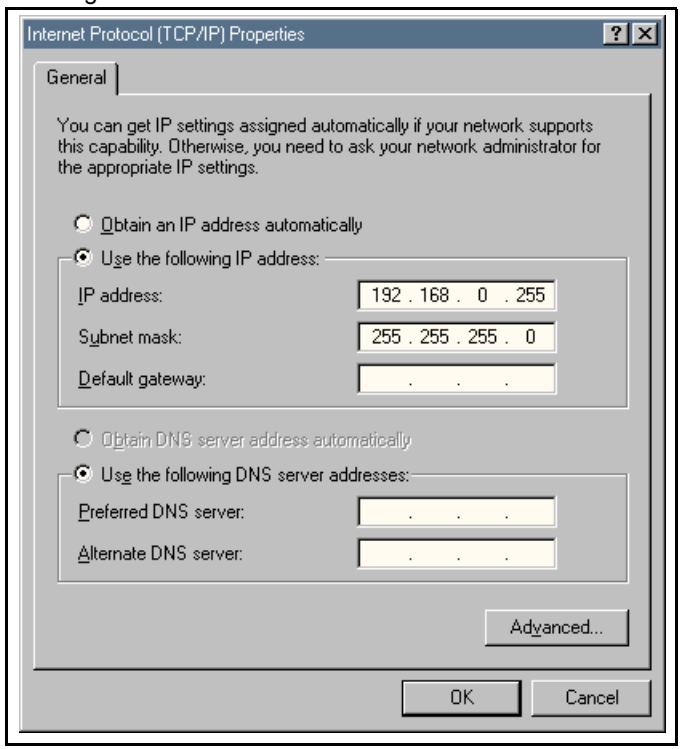
1. No cambie los ajustes Ethernet de Trajexia.
2. Establezca los ajustes de Trajexia Tools tal como se muestran.

fig. 1



3. Establezca los ajustes del ordenador tal como se muestran.

fig. 2



4.2.2 Comunicación con Trajexia de forma remota

En este ejemplo se muestra cómo conectarse a un sistema Trajexia local desde un ordenador en una ubicación remota. Suponga que los ajustes Ethernet del sistema Trajexia son:

- 10.83.50.70 es la dirección IP asignada de Trajexia.
- 255.255.240.0 es la máscara de subred local.
- 10.83.48.1 es el gateway local.
- El servidor asigna una dirección IP a los ordenadores automáticamente.

1. Establezca la dirección IP, la máscara de subred y el gateway desde la línea de comandos de la ventana de terminal en Trajexia con:
   Ethernet(1,-1,0,10,83,50,70)
   Ethernet(1,-1,2,255,255,240,0)
   Ethernet(1,-1,8,10,83,48,1)

2. Compruebe que los ajustes del sistema Trajexia local y el ordenador remotos son los que se muestran.

Tras el encendido, la TJ1-MC__ muestra alternativamente la dirección IP y la máscara de subred. Tras cada reconexión del cable Ethernet, el display sólo muestra la dirección IP.

fig. 3

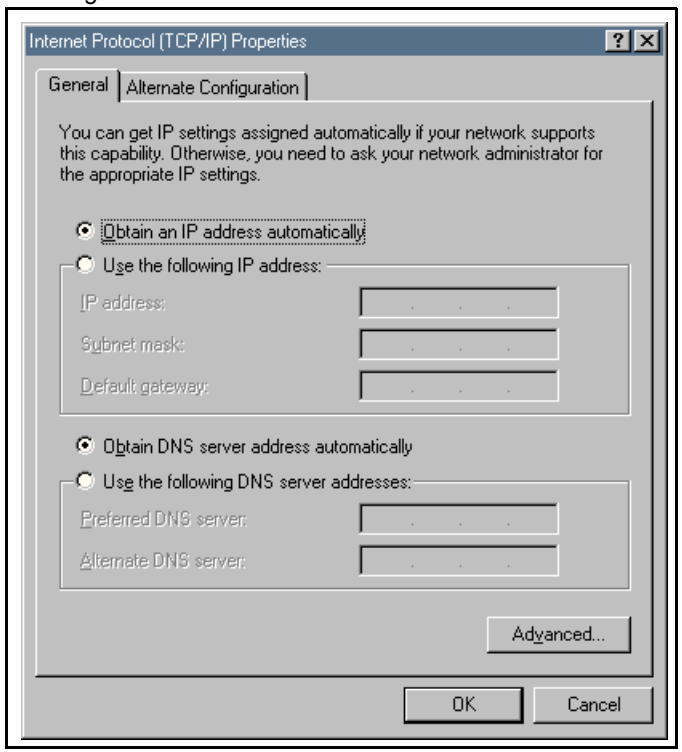
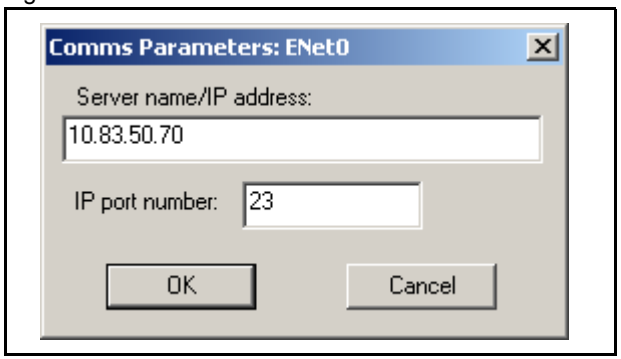


fig. 4



4.2.3 Protocolo Trajexia Tools

El protocolo Trajexia Tools lo utiliza Trajexia Tools para programar, monitorizar y depurar la TJ1-MC__.

Trajexia Tools utiliza un protocolo Telnet. De forma predeterminada, esta conexión utiliza el puerto 23. Si no se puede acceder a este puerto, se puede cambiar el número de puerto con el comando **Ethernet(1,-1,4,nuevo_puerto_n)**.

A diferencia de los comandos Ethernet estándar, este comando surte efecto inmediatamente después de su ejecución. El puerto cambia al valor predeterminado en el encendido. Por lo tanto, este comando se tiene que incluir en cualquier programa que se ejecute en el encendido.

El protocolo Trajexia Tools es TCP únicamente.

4.2.4 Protocolo de servidor FINS

FINS (Factory Interface Network Service) es un protocolo de comunicaciones propio de OMRON. En Trajexia se ha implementado un subconjunto de este protocolo. Consulte el manual de referencia de comandos de comunicaciones (W342-E1).

El protocolo FINS permite una comunicación perfecta con otros dispositivos OMRON, como PLCs, HMIs y CX-Drive.
El protocolo de servidor FINS no requiere ajustes de configuración.

ADVERTENCIA
Como la TJ1-MC__ se puede comunicar con diferentes fuentes a la vez, los comandos de dos fuentes pueden interferirse entre sí.

De forma predeterminada, esta conexión utiliza el puerto 9.600.
Si no se puede acceder a este puerto, se puede cambiar el puerto con el comando **Ethernet(1,-1,12,nuevo_puerto_n)**.

A diferencia de los comandos Ethernet estándar, este comando surte efecto inmediatamente después de su ejecución. El puerto cambia al valor predeterminado en el encendido. Por lo tanto, este comando se tiene que incluir en cualquier programa que se ejecute en el encendido.

Los comandos FINS permiten las comunicaciones entre los nodos de diferentes redes. Un dispositivo maestro FINS puede leer y escribir las variables VR y las variables de memoria TABLE de Trajexia con los comandos FINS. Estos comandos utilizan la conexión Ethernet de la TJ1-MC__.
El protocolo de servidor FINS es UDP únicamente.

Nota
La longitud máxima de un comando FINS a través de una conexión Ethernet es de 2.012 bytes.

Trajexia utiliza estos comandos FINS.
- 0101 (leer memoria)
- 0102 (escribir memoria)

Comando de lectura

El comando **read** FINS tiene este formato:

01 01	..	.. ..	00	.. ..
código_comando	**tipo_variable**	**dirección_inicio**	**Fijo**	**contaje_elementos**

Los parámetros pueden tener los siguientes valores:

Parámetro	Valores (hexadecimal)
código_comando	01 01
tipo_variable	• 82 (memoria TABLE en formato de entero de 16 bits) • C2 (memoria TABLE en formato de coma flotante IEEE de 32 bits) • B0 (memoria VR en formato de entero de 16 bits)
dirección_inicio	0 <= **dirección_inicio** <= número de variables – 1 <= FFFF

Parámetro	Valores (hexadecimal)
contaje_elementos	1 <= **contaje_elementos** <= número de variables – **dirección_inicio**

La TJ1-MC__ responde con estos códigos:

Estado	Código de respuesta (hexadecimal)	Descripción
Todos los elementos válidos	0000	OK
Tipo_variable no válido	1101	Sin tipo de área
Dirección_inicio no válida	1103	Error de designación de rango de direcciones
Número de elementos no válido	1104	Dirección fuera de rango

Si **tipo_variable** es 82 o B0 y el código de respuesta es 0000, la TJ1-MC__ responde con:

01	01	00	00			
código_comando		**código_respuesta**		**canal_1**	**canal_2**	**...**

Si **tipo_variable** es C2 y el código de respuesta es 0000, la TJ1-MC__ responde con:

01	01	00	00		
código_comando		**código_respuesta**		**canal_doble_1**	**...**

Nota
Los canales y canales dobles devueltos están en formato big-endian.

Comando de escritura

El comando **write** FINS tiene estos formatos:

- Si **tipo_variable** es 82 ó B0:

01	02	..	..	..	00	..	..	..	..	..	..
código_ comando		**tipo_ variable**	**dirección_ inicio**		**fijo**	**total_canales**		**canal 1**		**..**	

- Si **tipo_variable** es C2:

01	02	C2	..	..	00	..	..	..	..	..	..	..
código_ comando		**tipo_ variable**	**dirección_ inicio**		**fijo**	**total_canales**		**canal doble 1**				**..**

- Si **tipo_variable** es 30:

01	02	30	..	..	00	..	..	..	..	
código_ comando		**tipo_ variable**	**dirección_ inicio**		**número_ bit**	**total_bits**		**bit**		

Los parámetros pueden tener los siguientes valores:

Parámetro	Valores
código_comando	01 02
tipo_variable	• 82 (memoria TABLE en formato de entero de 16 bits) • C2 (memoria TABLE en formato de coma flotante IEEE de 32 bits) • B0 (memoria VR en formato de entero de 16 bits) • 30 (memoria VR en formato de bits)
dirección_inicio	0 <= **dirección_inicio** <= número de variables – 1 <= FFFF
total_canales	1 <= **total_canales** <= tamaño memoria – **dirección_inicio** + 1
total_bits	1
bit	00 ó 01

La TJ1-MC__ responde con estos códigos:

Estado	Código de respuesta (hexadecimal)	Descripción
Todos los elementos válidos	0000	Sí
Tipo_variable no válido	1101	Sin tipo de área
Dirección_inicio no válida	1103	Error de designación de rango de direcciones
Número_bit no válido	1103	Error de designación de rango de direcciones
Número de elementos no válido (totales)	1104	Dirección fuera de rango

4.2.5 Protocolo de cliente FINS

Trajexia puede iniciar la comunicación FINS con el comando BASIC **FINS_COMMS**. Consulte la descripción del comando para obtener información detallada.

Se admiten los comandos de lectura de memoria (0101) y de escritura de memoria (0102).

Esta funcionalidad resulta útil para comunicarse con un PLC de OMRON, otro sistema Trajexia o un PC que ejecute software de aplicación de servidor FINS.

Con el comando de lectura de memoria, la memoria se puede leer desde otros dispositivos con la función de servidor FINS. El comando de escritura se puede utilizar para escribir datos en dispositivos con la función de servidor FINS. El comando devuelve uno de los siguientes valores, según el resultado de la ejecución:

–1 El comando se ha ejecutado correctamente.

0 El comando ha fallado.

1 La solicitud no se ha enviado porque el cliente o el protocolo FINS están ocupados.

2 Uno o varios de los parámetros de la solicitud no son válidos.

3 Área de memoria de origen no válida.

4 Se ha enviado la solicitud, pero no se ha recibido ninguna respuesta del servidor remoto en el período de tiempo de espera.

5 Se ha recibido un código de respuesta de error del servidor remoto.

4.3 Protocolo serie

La TJ1-MC__ tiene un conector DB-9 que incluye dos puertos serie:
- Puerto 1: RS232
- Puerto 2: RS422 o RS485, según los ajustes del interruptor

Consulte el manual de referencia de hardware de Trajexia para obtener detalles.

Ambos puertos admiten independientemente estos protocolos:
- Maestro Host Link
- Esclavo Host Link
- Protocolo definido por el usuario

Nota
El puerto serie (puerto 1) NO SE PUEDE utilizar para programar la unidad.

4.3.1 Maestro Host Link

Si la TJ1-MC__ es el maestro Host Link, puede enviar comandos BASIC a un esclavo Host Link, por ejemplo, un PC. Al enviar un comando BASIC a un esclavo Host Link, la ejecución del siguiente comando BASIC espera hasta que el esclavo Host Link envía una respuesta.

Puede utilizar estos comandos BASIC:

Comando BASIC	Descripción
HLM_COMMAND	**HLM_COMMAND** ejecuta un comando Host Link específico en el esclavo.
HLM_READ	**HLM_READ** lee datos del esclavo Host Link en la memoria VR o TABLE.

Comando BASIC	Descripción
HLM_STATUS	**HLM_STATUS** ofrece el estado del último comando de maestro Host Link.
HLM_TIMEOUT	**HLM_TIMEOUT** define el tiempo de espera de maestro Host Link.
HLM_WRITE	**HLM_WRITE** escribe datos en el esclavo Host Link desde la memoria VR o TABLE.
SETCOM	**SETCOM** configura el puerto de comunicaciones serie y activa los protocolos Host Link.

Comandos

Para el protocolo maestro Host Link se admiten los siguientes comandos Host Link:

Tipo	Código de cabecera	Nombre	Función
Memoria de E/S lectura	**RR**	**CIO AREA READ**	Lee el número especificado de canales a partir del canal **CIO/IR** designado.
	RL	**LR AREA READ**	Lee el número especificado de canales a partir del canal **LR** designado.
	RH	**HR AREA READ**	Lee el número especificado de canales a partir del canal **HR** designado.
	RD	**DM AREA READ**	Lee el número especificado de canales a partir del canal **DM** designado.
	RJ	**AR AREA READ**	Lee el número especificado de canales a partir del canal **AR** designado.
	RE	**EM AREA READ**	Lee el número especificado de canales a partir del canal **EM** designado.
Memoria de E/S escritura	**WR**	**CIO AREA WRITE**	Escribe los datos especificados en unidades de canal a partir del canal **CIO/IR** designado.
	WL	**LR AREA WRITE**	Escribe los datos especificados en unidades de canal a partir del canal **LR** designado.
	WH	**HR AREA WRITE**	Escribe los datos especificados en unidades de canal a partir del canal **HR** designado.
	WD	**DM AREA WRITE**	Escribe los datos especificados en unidades de canal a partir del canal **DM** designado.
	WJ	**AR AREA WRITE**	Escribe los datos especificados en unidades de canal a partir del canal **AR** designado.
	WE	**EM AREA WRITE**	Escribe los datos especificados en unidades de canal a partir del canal **EM** designado.

Tipo	Código de cabecera	Nombre	Función
Unidad CPU estado	**SC**	**STATUS WRITE**	Cambia el modo operativo de la CPU.
Comprobación	**TS**	**TEST**	Devuelve un solo bloque, sin modificar, que se ha enviado desde el maestro.
Lectura de código de modelo de PC	**MM**	**PC MODEL READ**	Lee el código de modelo de la CPU.
Host Link Comunicaciones procesamiento	**XZ**	**ABORT** (sólo comando)	Anula la operación que se realiza mediante un comando Host Link y vuelve al estado inicial.
	**	**INITIALIZE** (sólo comando)	Inicializa los procedimientos de control de transferencia para todas las unidades Host Link.
	IC	Comando indefinido (sólo respuesta)	Se trata de la respuesta cuando el código de encabezado de comando no es válido.

Nota
El protocolo Host Link sólo admite comandos C. No admite FINS.

El protocolo maestro Host Link admite los comandos únicamente en una sola trama. En la tabla siguiente se muestra cómo utilizar el protocolo Host Link con los comandos BASIC y el modo operativo de la CPU (RUN, MON o PROG) para el que es válido cada comando.

Cabecera Código	Nombre	Comando BASIC requerido	RUN	MON	PRG
RR	**CIO AREA READ**	**HLM_READ**	Válido	Válido	Válido
RL	**LR AREA READ**	**HLM_READ**	Válido	Válido	Válido
RH	**HR AREA READ**	**HLM_READ**	Válido	Válido	Válido
RD	**DM AREA READ**	**HLM_READ**	Válido	Válido	Válido
RJ	**AR AREA READ**	**HLM_READ**	Válido	Válido	Válido
RE	**EM AREA READ**	**HLM_READ**	Válido	Válido	Válido
WR	**CIO AREA WRITE**	**HLM_WRITE**	No válido	Válido	Válido
WL	**LR AREA WRITE**	**HLM_WRITE**	No válido	Válido	Válido
WH	**HR AREA WRITE**	**HLM_WRITE**	No válido	Válido	Válido
WD	**DM AREA WRITE**	**HLM_WRITE**	No válido	Válido	Válido
WJ	**AR AREA WRITE**	**HLM_WRITE**	No válido	Válido	Válido
WE	**EM AREA WRITE**	**HLM_WRITE**	No válido	Válido	Válido
SC	**STATUS CHANGE**	**HLM_COMMAND**	Válido	Válido	Válido
TS	**TEST**	**HLM_COMMAND**	Válido	Válido	Válido
MM	**PC MODEL READ**	**HLM_COMMAND**	Válido	Válido	Válido
XZ	**ABORT** (sólo comando)	**HLM_COMMAND**	Válido	Válido	Válido
**	**INITIALIZE** (sólo comando)	**HLM_COMMAND**	Válido	Válido	Válido

Cabecera Código	Nombre	Comando BASIC requerido	RUN	MON	PRG
IC	Comando indefinido (sólo respuesta)	–	Válido	Válido	Válido

Precaución
Debe ejecutar los comandos de maestro Host Link desde una tarea de programa únicamente para evitar problemas de temporización multitarea.

Precaución
Los comando de maestro Host Link proporcionan las herramientas para intercambiar datos con el esclavo Host Link. El programa de usuario debe contener rutinas de tratamiento de errores adecuadas para tratar los errores de comunicaciones y efectuar reintentos si es necesario.

Códigos de fin

Son los códigos de fin definidos en el parámetro **HLM_STATUS**:

Código de fin	Descripción	Motivo probable	Solución
$00	Finalización normal	No existe ningún problema.	N/A
$01	No ejecutable en modo RUN	El comando que se ha enviado no se puede ejecutar cuando el PC está en modo RUN.	Compruebe la relación entre el comando y el modo de PC.
$13	Error de FCS	El FCS es erróneo.	Influencia del ruido; vuelva a transferir el comando.
$14	Format error (Error de formato)	• El formato de comando es erróneo. • Se ha dividido un comando que no se puede dividir. • La longitud de trama es menor que la longitud del comando aplicable.	Compruebe el formato y vuelva a transferir el comando.
$15	Error de datos de número de entrada	Los datos están fuera del rango especificado o son demasiado largos.	Corrija los argumentos del comando y vuelva a transferirlo.
$18	Error de longitud de trama	Se ha excedido la longitud de trama máxima de 131 bytes.	Compruebe el comando y vuelva a transferirlo.
$19	No ejecutable	No ha obtenido derechos de acceso.	Obtenga los derechos de acceso.
$21	No ejecutable debido a un error de la CPU.	El comando no se puede ejecutar debido a que se ha producido un error de la CPU.	Desconecte y vuelva a conectar la alimentación de la CPU.
$100	Tiempo de espera ACK de esclavo Host Link	–	–
$200	Error de dirección de comando IC	–	–

Configuración

Necesita el comando **SETCOM** para configurar el puerto serie de la TJ1-MC__ para el protocolo maestro Host Link. Configure el comando del siguiente modo:

SETCOM(velocidad_baudios, bits_datos, bits_parada, paridad, puerto, 6)

Después de haber establecido este comando, puede utilizar los comandos **HLM_READ**, **HLM_WRITE** y **HLM_COMMAND** para leer y escribir datos con Host Link.

Tiempo de espera

El mecanismo de tiempo de espera se ha implementado para impedir que la tarea BASIC esté en pausa durante mucho tiempo debido a comunicaciones deficientes o a su ausencia. El parámetro **HLM_TIMEOUT** especifica el período de tiempo de espera. Este período es el tiempo máximo que la tarea de programa espera después de que ha enviado el comando para recibir una respuesta.
Si transcurre el período de tiempo de espera, **HLM_STATUS** contiene el estado del comando y la tarea BASIC continúa.
El parámetro **HLM_TIMEOUT** especifica el período de tiempo de espera para todos los comandos y todos los puertos.

Estado

El parámetro **HLM_STATUS** contiene el estado del último comando de maestro Host Link enviado al puerto especificado. El parámetro indica el estado de los comandos **HLM_READ**, **HLM_WRITE** y **HLM_COMMAND**. Los bits de estado son:

Bit	Nombre	Descripción
0-7	Código de fin	El código de fin es: • el código de fin definido por el esclavo Host Link, cuando se ha producido un problema en la cadena de datos del comando enviado, o bien • un código de fin definido por el maestro Host Link, cuando se ha producido un problema en la cadena de datos de la respuesta recibida.
8	Error de tiempo de espera	Se produce un error de tiempo de espera si no se recibe ninguna respuesta en el período de tiempo de espera. Esto indica que se han perdido las comunicaciones.
9	Comando no reconocido	Este estado indica que el esclavo no ha reconocido el comando y ha devuelto una respuesta IC.

HLM_STATUS contiene el valor 0 si no se ha producido ningún problema. En el caso de un valor distinto de cero, se tiene que programar una acción adecuada como, por ejemplo, un reintento o una parada de emergencia, en el programa BASIC del usuario. Cada puerto tiene un parámetro HLM_STATUS. Se necesita el modificador **PORT** para especificar el puerto.

Ejemplos

En estos ejemplos se supone la siguiente configuración:

- Un sistema Trajexia con una TJ1-MC__.
- Un PC esclavo, con la dirección de nodo 13.
- Una conexión desde el puerto serie de la TJ1-MC__ al PC. El puerto serie utiliza comunicaciones RS422.

Ejemplo	Lectura de datos desde el PC con **HLM_READ**.
Código BASIC	**' Configurar el maestro Host Link para el puerto 2** **SETCOM(9600,7,2,2,2,6)** **' Dirección de origen: CIO/IR 002** **' Cantidad de datos: 2 canales** **' Dirección de destino: VR(0)** **HLM_READ(2,13,PLC_IR,2,2,MC_VR,0)**
Host Link comunicaciones	• Desde el maestro Host Link al esclavo Host Link: @13RR0002000242* • Desde el esclavo Host Link al maestro Host Link: @13RR000101010241*
Resultado	• Dirección VR = 0: valor = 257,0000 • Dirección VR = 1: valor = 258,0000

Ejemplo	Escritura de datos en el PC con **HLM_WRITE**.
Código BASIC	**' Dirección de origen: TABLE(18)** **' Cantidad de datos: 2 canales** **' Dirección de destino: LR 014** **TABLE(18,$0701,$0702)** **HLM_WRITE(2,13,PLC_LR,14,2,MC_TABLE,18)**
Host Link comunicaciones	• Desde el maestro Host Link al esclavo Host Link: @13WL0014070107025F* • Desde el esclavo Host Link al maestro Host Link: @13WL0059*
Resultado	• Dirección LR = 0: valor = 701 (hexadecimal) • Dirección LR = 1: valor = 702 (hexadecimal)

Ejemplo	Envíe el comando **TS** (prueba) al PC con **HLM_COMMAND**.
Código BASIC	**HLM_COMMAND(HLM_TEST,2,13)**
Comunicaciones Host Link	• Desde el maestro Host Link al esclavo Host Link: @13TSMCW151 TEST STRING2A* • Desde el esclavo Host Link al maestro Host Link: @13TSMCW151 TEST STRING2A*
Resultado	**HLM_STATUS PORT(2) = 0**, que implica una comunicación correcta.

Ejemplo	Configure el PC en modo MON con **HLM_COMMAND**.
Código BASIC	**HLM_COMMAND(HLM_STWR,2,13,2)**
Host Link comunicaciones	• Desde el maestro Host Link al esclavo Host Link: @13SC0250* • Desde el esclavo Host Link al maestro Host Link: @13SC0052*
Resultado	El PC se ejecuta en modo MON. Tenga en cuenta que esto es necesario para escribir datos en el PC con **HLM_WRITE**.

Ejemplo	Lectura del código de modelo de PC con **HLM_COMMAND** (tiempo de espera).
Código BASIC	**HLM_TIMEOUT = 500** **' Dirección de destino: VR(100)** **HLM_COMMAND(HLM_MREAD,2,13,MC_VR,100)**
Comunicaciones Host Link	• Desde el maestro Host Link al esclavo Host Link: @13MM42* • Desde el esclavo Host Link al maestro Host Link: sin respuesta
Resultado	Como el maestro no ha recibido una respuesta del PC, **HLM_STATUS PORT(2)** tiene el valor 256 (está establecido el bit 8) después de 500 ciclos de servo.

4.3.2 Esclavo Host Link

Si la TJ1-MC__ es el esclavo Host Link, un maestro Host Link (por ejemplo, un terminal programable) puede leer datos de la TJ1-MC__ y escribir datos en ella. La asignación entre el esclavo y el maestro es:

Memoria de TJ1-MC__	Asignación de Host Link	Rango de direcciones
VR	**CIO**	0 a 1.023
TABLE	**DM**	0 a 63.999

Puede utilizar estos comandos BASIC:

Comando BASIC	Descripción
SETCOM	**SETCOM** configura el puerto de comunicaciones serie y activa los protocolos Host Link.
HLS_NODE	**HLS_NODE** define el número de unidad de esclavo para el protocolo de esclavo Host Link.
HLS_MODEL	**HLS_MODEL** define el código de modelo de TJ1-MC__ para el protocolo de esclavo Host Link.

Comandos

Los comandos admitidos para el protocolo esclavo Host Link se indican en la tabla siguiente. El protocolo admite la transferencia de trama simple y la transferencia de trama múltiple.

Tipo	Código de cabecera	Nombre	Función
Lectura de memoria de E/S	**RR**	**CIO AREA READ**	Lee el número especificado de canales de la memoria VR a partir del canal designado.
	RD	**DM AREA READ**	Lee el número especificado de canales de la memoria TABLE a partir del canal designado.

Tipo	Código de cabecera	Nombre	Función
Memoria de E/S escritura	**WR**	**CIO AREA WRITE**	Escribe los datos especificados en unidades de canal en la memoria VR a partir del canal designado.
	WD	**DM AREA WRITE**	Escribe los datos especificados en unidades de canal en la memoria TABLE a partir del canal designado.
Comprobación	**TS**	**TEST**	Devuelve un solo bloque, sin modificar, que se ha enviado desde el maestro.
Lectura del código de modelo de PC	**MM**	**PC MODEL READ**	Lee el código de modelo de la TJ1-MC__ según lo especificado por el parámetro **HLS_MODEL**.
Lectura y registro del área de memoria de E/S	**QQMR**	**REGISTER I/O MEMORY**	Registra la tabla de E/S con el contenido de la configuración de E/S real.
	QQIR	**READ I/O MEMORY**	Lee a la vez todos los canales/bits de la memoria de E/S registrada.
Procesamiento de comunicaciones Host Link	**XZ**	**ABORT** (sólo comando)	Anula la operación que se realiza mediante un comando Host Link y vuelve al estado inicial.
	**	**INITIALIZE** (sólo comando)	Inicializa los procedimientos de control de transferencia para todas las unidades Host Link.
	IC	Comando indefinido (sólo respuesta)	Se trata de la respuesta cuando el código de encabezado de comando no es válido.

Códigos de fin

Son los códigos de fin de respuesta que se devuelven en la trama de respuesta:

Código de fin	Descripción	Motivo probable	Solución
0	Finalización normal	No existe ningún problema.	N/A
13	Error de FCS	El FCS es erróneo.	Compruebe el método de cálculo de FCS. Si ha habido influencias del ruido; vuelva a transferir el comando.
14	Format error (Error de formato)	• El formato de comando es erróneo. • Se ha dividido un comando que no se puede dividir. • La longitud de trama es menor que la longitud del comando aplicable.	Compruebe el formato y vuelva a transferir el comando.
15	Error de datos de número de entrada	Los datos están fuera del rango especificado o son demasiado largos.	Corrija los argumentos del comando y vuelva a transferirlo.
18	Error de longitud de trama	Se ha excedido la longitud de trama máxima de 131 bytes.	Compruebe los datos y vuelva a transferir el comando.
19	No ejecutable	Se ha ejecutado un lote de memoria de E/S cuando no estaban registrados elementos para leer.	Registre los elementos para leer antes de intentar una lectura por lotes.
A3	Se ha anulado debido a un error FCS en los datos de transmisión	Se ha producido un error FCS en la trama segunda o posterior.	Corrija los datos del comando y vuelva a transferirlo.

Código de fin	Descripción	Motivo probable	Solución
A4	Se ha anulado debido a un error de formato en los datos de transmisión	El formato de comando no coincide con el número de bytes en la trama segunda o posterior.	Corrija los datos del comando y vuelva a transferirlo.
A5	Se ha anulado debido a un error de datos de número de entrada en los datos de transmisión	Se ha producido un error de datos de número de entrada en la trama segunda o posterior, o bien un error de longitud de datos.	Corrija los datos del comando y vuelva a transferirlo.
A8	Se ha anulado debido a un error longitud de trama en los datos de transmisión	La longitud de la trama segunda o posterior ha excedido el máximo de 128 bytes.	Corrija los datos del comando y vuelva a transferirlo.

Configuración

Necesita el comando **SETCOM** para configurar el puerto serie de la TJ1-MC__ para el protocolo esclavo Host Link. Configure el comando del siguiente modo:

SETCOM(velocidad_baudios, bits_datos, bits_parada, paridad, puerto, 5)

Después de haber establecido este comando, la TJ1-MC__ responde a los comandos Host Link desde el maestro con el número de nodo especificado. Puede establecer este número de nodo con el parámetro **HLS_NODE**.

Ejemplo

En este ejemplo se supone la siguiente configuración:

- Un sistema Trajexia con una TJ1-MC__.
- Un terminal programable NS8.
- Una conexión desde el puerto serie de la TJ1-MC__ al terminal programable. El puerto serie utiliza comunicaciones RS232C.

Ejemplo — Configuración del esclavo Host Link.

Código BASIC

```
' Definir el nodo de esclavo Host Link
HLS_NODE = 15
' Definir el código de modelo de esclavo Host Link
HLS_MODEL = $FA
' Configurar el esclavo Host Link para el puerto 1
SETCOM(9600,7,2,2,1,5)
```

Resultado — La TJ1-MC__ puede comunicarse con el terminal programable.

4.3.3 Protocolo definido por el usuario

Puede implementar un protocolo de comunicaciones definido por el usuario con estos comandos:

Comando BASIC	Descripción
SETCOM	**SETCOM** configura el puerto de comunicaciones serie y activa los protocolos Host Link.
GET	**GET** asigna el código ASCII de un carácter recibido a una variable.
INPUT	**INPUT** asigna valores de cadena de entrada numérica a las variables especificadas.
KEY	**KEY** devuelve **TRUE** o **FALSE** si un carácter se ha recibido o no se ha recibido.
LINPUT	**LINPUT** asigna el código ASCII de los caracteres recibidos a una matriz de variables.
PRINT	**PRINT** envía una serie de caracteres a un dispositivo de salida.

Ejemplo

Se supone una configuración con:

- Un sistema Trajexia con una TJ1-MC__.
- Un sistema de visión F500 de OMRON.
- Una conexión desde el puerto serie de la TJ1-MC__ al F500. El puerto serie utiliza comunicaciones RS232 (puerto 1).

Este programa envía un comando de visión a través del puerto serie, lee la respuesta desde el sistema de visión, la escribe en las variables VR e imprime los resultados en la ventana de terminal de Trajexia Tools.

```
' En el programa STARTUP
' Configuración del puerto RS232 para el sistema de visión
SETCOM(38400,8,1,0,1,0)
' En el programa de aplicación
lazo:
   ' Activar flanco ascendente en el sistema virtual
   WAIT UNTIL IN(30)=0
   WAIT UNTIL IN(30)=1
   ' Borrar pantalla
   PRINT CHR(27);"[2J"

   ' Borrar búfer
   GOSUB borrar_bufer

   ' Enviar comando al puerto serie según VR(10)
   IF comando_vision=medida_v THEN
      PRINT #1, "M"
      PRINT ">> M"
   ELSEIF comando_vision=fecha_v THEN
      PRINT #1, "FECHA"
      PRINT ">> FECHA"
   ELSEIF comando_vision=escena_v THEN
      PRINT #1,"ESCENA ";n_escena
      PRINT ">> ESCENA"
   ENDIF

   'Comprobar respuesta
   GOSUB leer_bufer

GOTO lazo
   leer_bufer:
   contaje=0
   estado_respuesta=0
   k=-1
   TICKS=5000
   REPEAT
      IF KEY#1 THEN
         contaje=contaje+1
         GET#1, k
         'PRINT k;contaje
         TABLE(contaje,k)
         'PRINT contaje
      ENDIF
   UNTIL TICKS<0 'OR k=13
   PRINT "Recibidos ";contaje[0];" caracteres"
   FOR i=1 TO contaje
      IF TABLE(i)<>13 THEN
         PRINT CHR(TABLE(i))
      ELSE
         PRINT "'cr'"
      ENDIF
   NEXT i
   IF TICKS<0 THEN
      PRINT "Se ha agotado el tiempo de espera en la
comunicación con F500"
      estado_respuesta=3
   ELSEIF TABLE(contaje-2)=79 AND TABLE(contaje-1)=75 THEN
      PRINT "Respuesta correcta"
      estado_respuesta=1
   ELSE
      PRINT "Respuesta incorrecta"
      estado_respuesta=2
   ENDIF
   PRINT "El estado de la respuesta es :";estado_respuesta[0]
RETURN
borrar_bufer:
   PRINT "Borrando..."
   WHILE KEY#1
      GET#1,k
      PRINT k
   WEND
   PRINT "Borrado."
RETURN
```

4.4 PROFIBUS

4.4.1 Introducción

PROFIBUS es un estándar de bus de campo abierto internacional. La TJ1-PRT permite que el sistema Trajexia se comunique con una red PROFIBUS. Intercambia datos entre el maestro PROFIBUS y la TJ1-MC__. Para ello, utiliza las variables VR de Trajexia.

4.4.2 Configuración de las comunicaciones

La TJ1-PRT tiene dos selectores de número de nodo. Puede utilizar los selectores de número de nodo para asignar una dirección de red PROFIBUS a la TJ1-PRT. Debe asignar una dirección a la TJ1-PRT antes de conectar la alimentación del sistema Trajexia.

Para inicializar la TJ1-PRT, utilice el comando BASIC **PROFIBUS**:

PROFIBUS(número_unidad, 2, 1, inicio_salida, contaje_salidas, inicio_entrada, contaje_entradas)

donde:
- **número_unidad** es el número de la unidad TJ1-PRT.
- **inicio_salida** es la dirección de inicio del rango de datos de salida de las variables VR.
- **contaje_salidas** es el número de variables VR en el rango de datos de salida.
- **inicio_entrada** es la dirección de inicio del rango de datos de entrada de las variables VR.
- **contaje_entradas** es el número de variables VR en el rango de datos de entrada.

Nota
El número máximo de variables VR para intercambiar datos es 122.

Después de haber ejecutado el comando **PROFIBUS(número_unidad, 2, ...)**, las matrices de datos se intercambian automáticamente. Los datos intercambiados entre la TJ1-PRT y el maestro PROFIBUS tienen formato entero de 16 bits. Cada canal intercambiado va de –32.768 a 32.767.

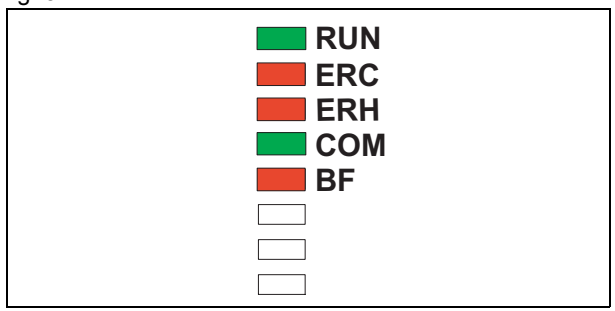
Una variable VR puede contener un número de 24 bits y también puede contener fragmentos. El intercambio con el maestro PROFIBUS no admite valores fuera del rango –32.768...32.767 y fragmentos.

Una secuencia de ejemplo para configurar la unidad TJ1-PRT es la siguiente:

1. Configure el número de unidad con dos interruptores rotativos de la unidad TJ1-PRT.
2. Conecte la alimentación al sistema. Se ilumina el LED **RUN**. El LED **ERH** parpadea.
3. Cree un programa BASIC que contenga el comando **PROFIBUS(2,2,1,10,7,150,3)**. En este ejemplo el sistema inicializa una unidad TJ1-PRT con el número 2. El sistema envía siete canales de salida desde el maestro a las posiciones 10 a 16 de VR y tres canales de entrada desde las posiciones 150 a 152 de VR al maestro.
4. Si la configuración es correcta, se iluminan los LEDs **RUN** y **COMM**. La comunicación ahora está activa.

Para configurar la CJ1-PRM21 con CX-PROFIBUS, realice estos pasos:

fig. 5

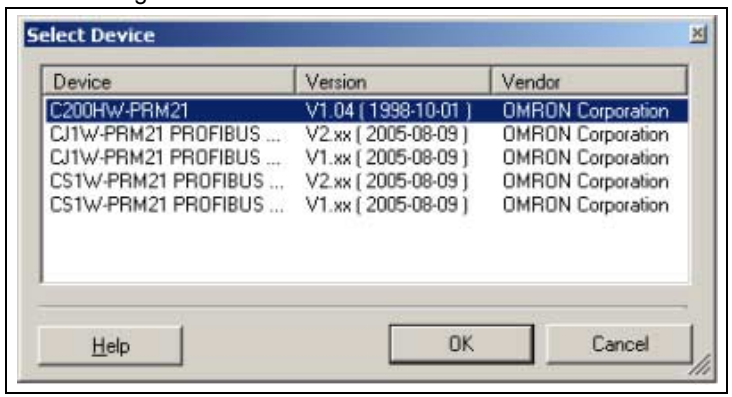


1. Inicie la herramienta de programación CX-PROFIBUS.
2. Haga clic con el botón derecho del ratón en el árbol MyNetwork (Mi red).
3. Seleccione **Add Device...** (Agregar dispositivo).

fig. 6

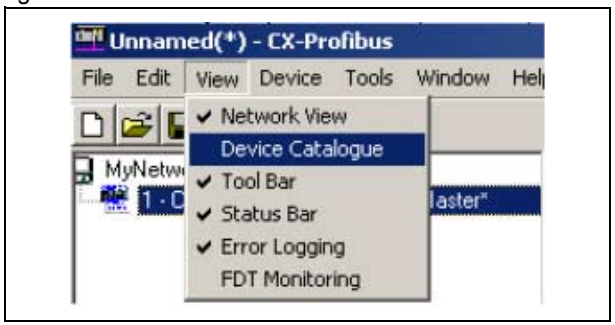
4. Seleccione la placa de maestro PROFIBUS.
5. Haga clic en **OK** (Aceptar).

fig. 7



6. Abra **Device Catalogue** (Catálogo de dispositivos) en el menú **View** (Ver).

fig. 8



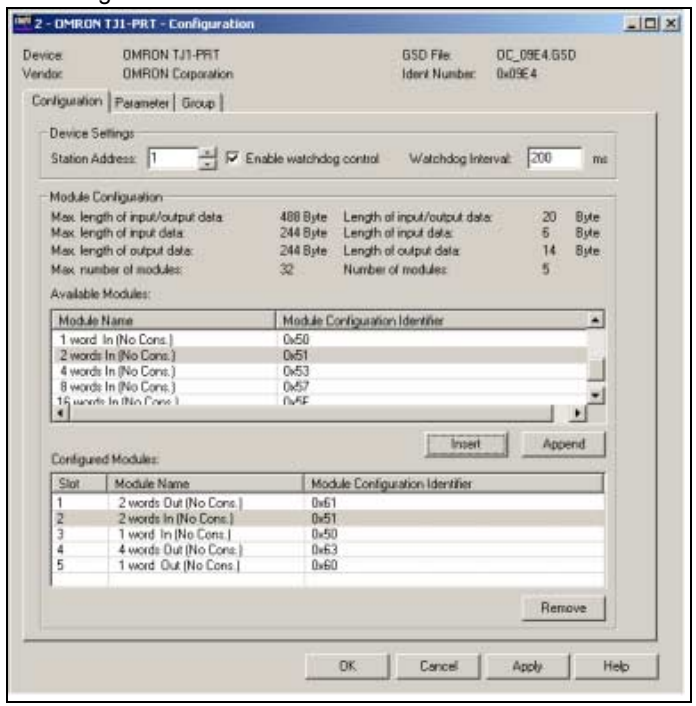
7. Haga clic en **Install GSD Files...** (Instalar archivos GSD). El archivo GSD está en el CD de Trajexia Tools. También se puede encontrar en el centro de descarga del sitio Web de OMRON.
8. Haga clic en **Update** (Actualizar). La TJ1-PRT se muestra en la lista.
9. Seleccione la TJ1-PRT de OMRON en la lista y haga clic en **Add Device** (Agregar dispositivo).

fig. 9

10. Haga doble clic en el módulo de esclavo TJ1-PRT en el árbol MyNetwork (Mi red).
11. Defina el número de nodo en el campo **Station Address** (Dirección de estación).
12. Agregue (con **Insert** [Insertar]) los módulos de entrada y salida a la lista de configuración siguiente.
13. Asegúrese de que la cantidad de canales de entrada y de salida de los módulos seleccionados son iguales a la cantidad seleccionada con el comando PROFIBUS.
14. Haga clic en **OK** (Aceptar).

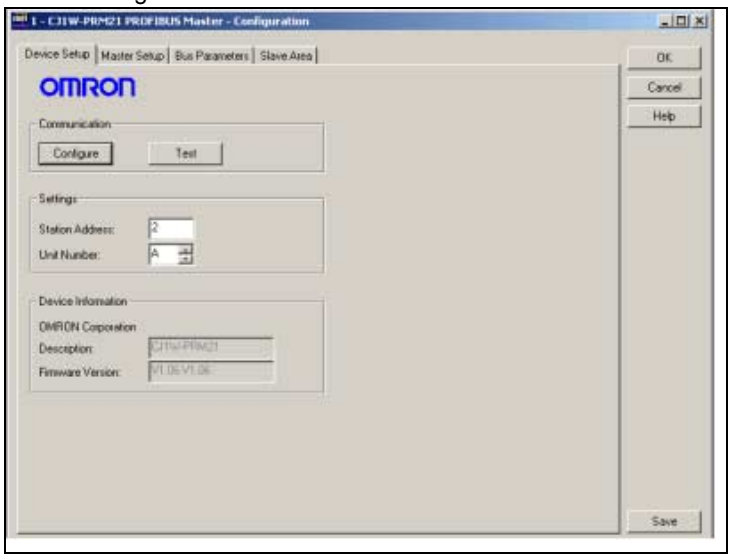
Para configurar la CJ1W-PRM21 con CX-PROFIBUS, realice estos pasos:

fig. 10



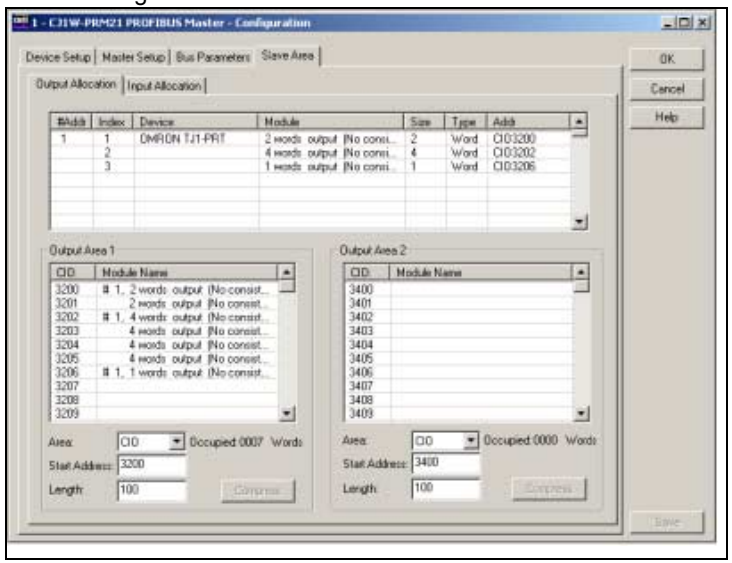
1. Haga doble clic en el módulo maestro en el árbol MyNetwork (Mi red).
2. Configure los campos **Station Address** (Dirección de estación) y **Unit Number** (Número de unidad).

fig. 11



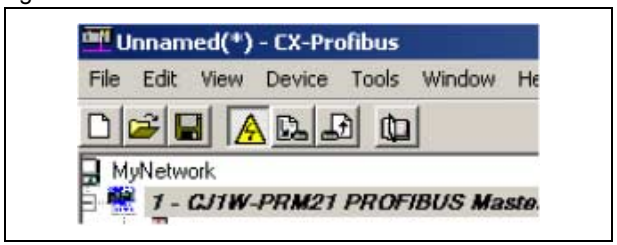
3. Seleccione la ficha **Slave area** (Área de esclavo).
4. Configure el campo **Start Address** (Dirección de inicio) de **Output Area 1** (Área de salida 1) y **Input Area 1** (Área de entrada 1).
5. Guarde el proyecto.

fig. 12



6. Haga clic en el botón de la barra de herramientas **Device Online/ Offline (Toggle)** (Dispositivo online/offline [Conmutador]) para ir online.
7. Haga clic en el botón de la barra de herramientas **Device Download** (Descarga de dispositivo) para descargar los parámetros.

fig. 13



4.4.3 Estado de las comunicaciones

TJ1-PRT puede proporcionar información de estado a la TJ1-MC__. Puede recuperar la información de estado en BASIC con el comando **PROFIBUS** (número_unidad,4,0). El resultado proporciona la siguiente información:

Bit	Valor	Descripción
0	0	Configuración fallida del intercambio de datos de E/S
	1	Intercambio de datos de E/S configurado correctamente
1	0	Datos de E/S no disponibles
	1	Datos de E/S disponibles
2	0	Intercambio de datos activo en modo OPERATE
	1	Intercambio de datos activo en modo CLEAR

4.5 DeviceNet

4.5.1 Introducción

DeviceNet es un estándar de bus de campo abierto internacional que está basado en el protocolo CAN. La TJ1-DRT permite que el sistema Trajexia se comunique con una red DeviceNet. Intercambia datos entre un maestro DeviceNet y la TJ1-MC__. Para ello, utiliza las variables VR de Trajexia.

4.5.2 Configuración de las comunicaciones

La TJ1-DRT tiene dos selectores de número de nodo. Puede utilizar los selectores de número de nodo para asignar un número de nodo a la TJ1-DRT.

Los números de nodo DeviceNet van de 0 a 63. Si selecciona un número de nodo con los selectores de número de nodo que supere este rango, seleccionará el número de nodo definido por software. Los modos que activan el ajuste de software son del 64 al 99.

Para inicializar la TJ1-DRT, utilice el comando BASIC **DEVICENET**:

DEVICENET(número_unidad, 2, 1, inicio_salida, contaje_salidas, inicio_entrada, contaje_entradas)

donde:
- **número_unidad** es el número de la unidad TJ1-DRT.
- **inicio_salida** es la dirección de inicio del rango de datos de salida de las variables VR.
- **contaje_salidas** es el número de variables VR en el rango de datos de salida.
- **inicio_entrada** es la dirección de inicio del rango de datos de entrada de las variables VR.
- **contaje_entradas** es el número de variables VR en el rango de datos de entrada.

Nota
El número máximo de variables VR para intercambiar datos es 32.

Note.
Si utiliza un maestro DeviceNet OMRON, se recomienda que seleccione contaje_entrada o contaje_salida con un valor de 4,8,16 ó 32 para las variables VR.

Después de haber ejecutado el comando **DEVICENET(número_unidad, 2, ...)**, las matrices de datos se intercambian automáticamente. Los datos intercambiados entre la TJ1-DRT y el maestro DeviceNet tienen formato entero de 16 bits. Cada canal intercambiado va de –32.768 a 32.767.

Una variable VR puede contener un número de 24 bits y también puede contener fragmentos. El intercambio con el maestro DeviceNet no admite valores fuera del rango de –32.768 a 32.767 ó fragmentos.

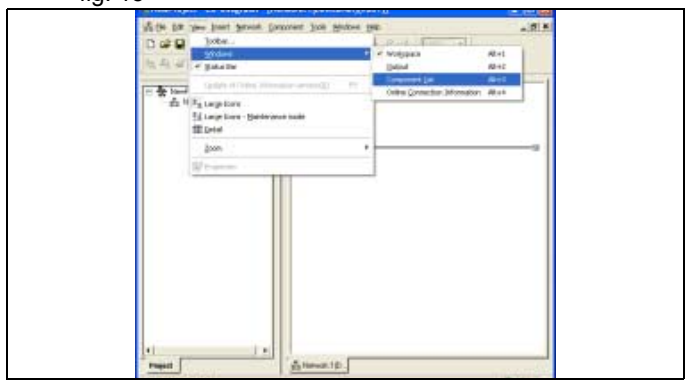
Configuración de la red DeviceNet
Para configurar el maestro DeviceNet CJ1W/CS1W-DRM21 de OMRON para intercambiar variables VR con el sistema Trajexia, realice estos pasos:

1. Inicie CX-Integrater en la herramienta de programación CX-ONE.
2. Seleccione Red en el menú Insert (Insertar).
3. Seleccione DeviceNet en la pantalla Select Network (Seleccionar red). Se muestra la vista de red.

fig. 14

4. Seleccione CJ1W-DRM21 en la lista de adaptadores de comunicaciones de OMRON.

fig. 15



5. Arrastre y coloque la CJ1W-DRM21 en la ventana de red.

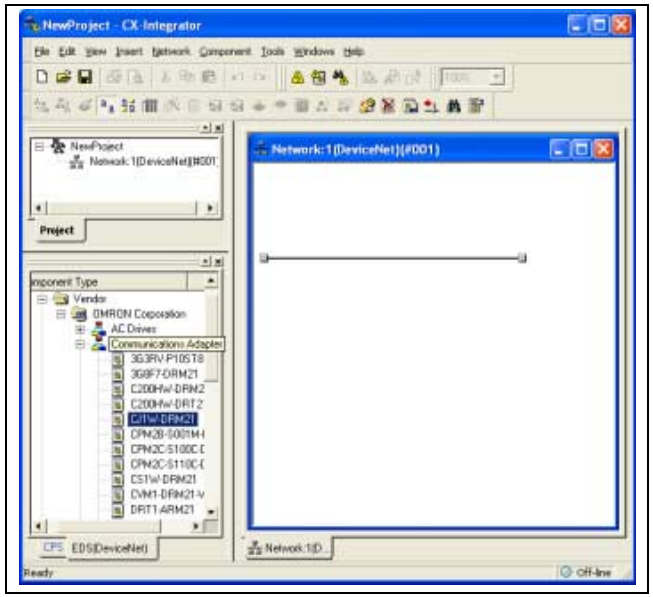
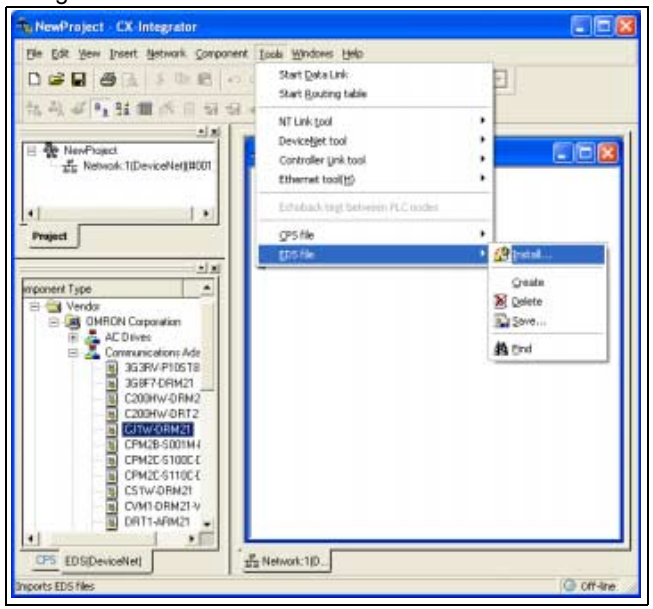


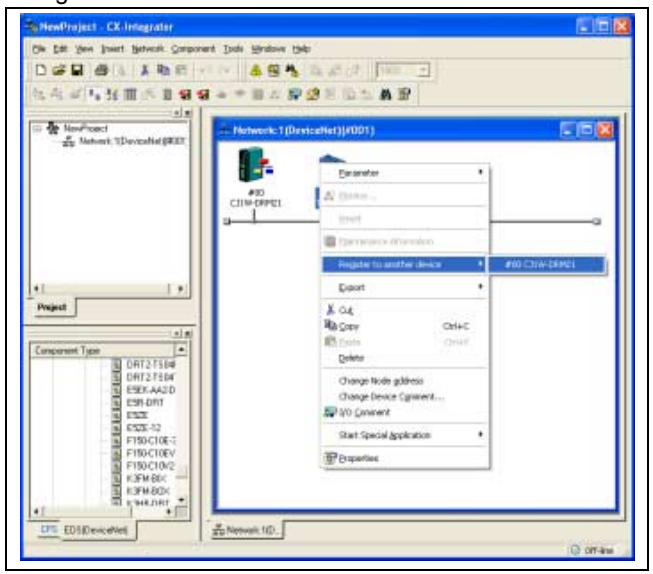
fig. 16

6. Instale el archivo **EDS** desde CX-Integrator.
7. Seleccione No en el cuadro de diálogo. El icono no es necesario.



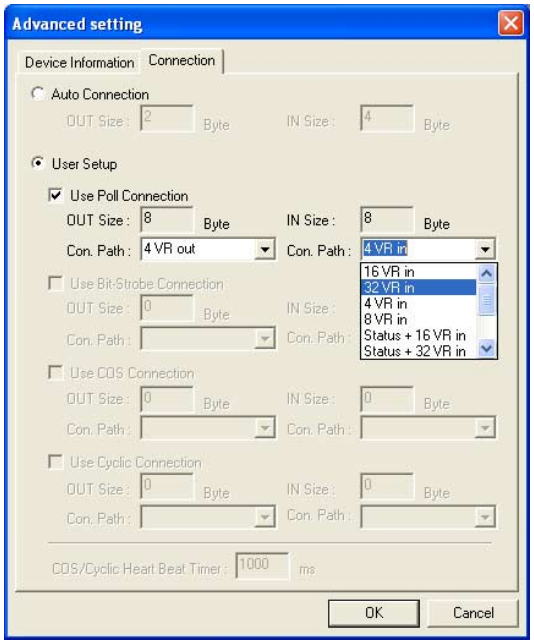
8. Registre el esclavo en el maestro, haga clic con el botón derecho del ratón en el icono de #01TJ1-DRT.
9. Haga doble clic en el icono del maestro.
10. Seleccione el dispositivo TJ1-DRT.

fig. 17



11. Haga clic en Advanced Setup (Configuración avanzada).
12. Haga clic en la ficha Connection (Conexión).
13. Haga clic en User Setup (Configuración de usuario).
14. Haga clic en Use Poll Connection
15. Seleccione Con.Path (Ruta de conexión).
16. Seleccione el número de variables que han seleccionado para la comunicación DeviceNet.
17. Haga clic en OK (Aceptar) para confirmar todos los cuadros de diálogo.
18. Seleccione Work Online (Trabajar online) en el menú Network (Red).
19. Seleccione Parameter (Parámetro) en el menú Component (Componente).
20. Haga clic con el botón derecho en el icono del maestro.
21. Seleccione Parameter Download (Descarga de parámetros).

fig. 18



4.5.3 Estado de las comunicaciones

TJ1-DRT puede proporcionar información de estado a la TJ1-MC__ y al maestro DeviceNet. Puede recuperar la información de estado en BASIC con el comando DeviceNet (número_unidad,4,0). El resultado proporciona la siguiente información:

Bit	Valor	Descripción
0	0	DeviceNet (número_unidad, 2, ...) no ejecutado todavía
	1	DeviceNet (número_unidad, 2, ...) se ha ejecutado sin errores
1	0	Sin conexión de E/S de DeviceNet
	1	Conexión de E/S de DeviceNet en funcionamiento
2	0	Las variables VR del rango de datos de salida se han actualizado
	1	Las variables VR del rango de datos de salida todavía no se han actualizado
3	0	El tamaño de conexión de E/S de DeviceNet coincide con el comando DeviceNet(número_unidad,2,…)
	1	El tamaño de conexión de E/S de DeviceNet no coincide con el comando DeviceNet(número_unidad,2,…)
4-7	0	Siempre cero
8	0	Alimentación de red correcta
	1	Fallo de alimentación de red
9	0	No se ha producido ningún BUSOFF
	1	Se ha producido BUSOFF
10	0	Sin error de duplicación de dirección de nodo
	1	Error de duplicación de dirección de nodo
11-	0	Reservado

Puede recuperar la información de estado en el maestro DeviceNet si selecciona una ruta de conexión que incluya la información de estado. La información de estado incluye un bit. El bit 2 indica que la tensión de red está por debajo del nivel establecido en los parámetros DeviceNet de TJ1-DRT. Puede establecer los parámetros de DeviceNet de TJ1-DRT con un configurador de DeviceNet. El nivel predeterminado es 11 V.

4.6 MECHATROLINKII

El protocolo MECHATROLINK-II es un bus serie creado para controlar el movimiento de una forma determinista.

El número de dispositivos MECHATROLINK-II determina el tiempo de ciclo de intercambio de datos:

- De 1 a 4 dispositivos el tiempo de ciclo puede ser 0,5 ms, 1 ms ó 2 ms.
- De 5 a 8 dispositivos el tiempo de ciclo puede ser 1 ms ó 2 ms.
- De 9 a 16 dispositivos el tiempo de ciclo es de 2 ms.

La transmisión cíclica tiene dos etapas:

- La TJ1-ML__ envía el comando de referencia a los esclavos MECHATROLINK.
- Los esclavos mandan respuesta e información de estado a la TJ1-ML__.

MECHATROLINK-II utiliza un reloj de sincronización y mensajes de difusión para garantizar que todos los esclavos ejecutan los comandos al mismo tiempo.

Además, se transfiere otra información a una velocidad inferior, por ejemplo, la lectura y escritura de parámetros.

Hay comandos BASIC específicos que se dirigen a las unidades esclavas MECHATROLINK directamente.

- **DRIVE_CLEAR**: este comando restablece una alarma en un servodriver MECHATROLINK mediante un mensaje MECHATROLINK.
- **OP(45,ON)**: este comando activa una salida en un módulo de E/S MECHATROLINK remoto.

5 Interfaz Trajexia Tools

5.1 Introducción

Trajexia Tools es la herramienta de programación que permite programar el sistema Trajexia. Este software permite que el ingeniero de aplicaciones gestione los proyectos Trajexia y edite los programas. Incluye algunas herramientas útiles que se describen posteriormente en este capítulo tales como ejecución, parada y recorrido paso a paso de programas individuales, adición de puntos de ruptura, ejecución de comandos directos para leer y escribir variables, funciones de osciloscopio y programar los servodrivers. La conexión a la TJ1-MC__ se realiza a través de Ethernet. Es necesario establecer los ajustes de comunicaciones antes de conectarse a una unidad. La herramienta de programación Trajexia Tools se ha diseñado para funcionar online con una TJ1-MC__.
Trajexia Tools incluye:

- Herramienta de programación para TJ1-MC__ (Motion Perfect 2)
- CX-Server
- CX-Drive para programar y configurar los servodrivers y el variador

Trajexia Tools se puede utilizar para programar, mediante comunicaciones serie, otros controladores Motion de OMRON: C200HW-MC402E, R88A-MCW151-E y R88A-MCW151-DRT-E.

5.2 Especificaciones y conexiones

5.2.1 Especificaciones del PC

Las especificaciones de PC para el uso de Trajexia Tools son:

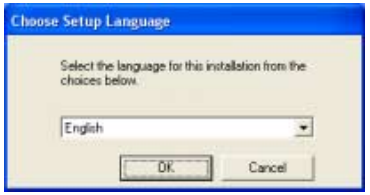
Descripción	Especificación mínima	Especificación recomendada
CPU	Pentium a 300 MHz	Pentium a 1 GHz
RAM	64 MB	256 MB
Espacio en disco duro	140 MB	140 MB
Sistema operativo	Windows TM 98	Windows XP
Pantalla	800 x 600 256 colores	1.024 x 768 Color de 24 bits
Comunicaciones	Ethernet 10BaseT	Ethernet 100BaseT
Internet	Explorer V5.0	Explorer V6.0

Utilice la versión más reciente de Trajexia Tools. Las actualizaciones están disponibles en el distribuidor local. El software también está disponible en el sitio Web de Trajexia: www.trajexia.com.

5.2.2 Instalación del software Trajexia Tools

1. Inserte el CD de Trajexia Tools en la unidad de CD-ROM del PC.
2. El programa de instalación de Trajexia Tools se inicia automáticamente.
3. Si el programa de instalación de Trajexia Tools no se inicia automáticamente, hágalo manualmente: ejecute **setup.exe** en el directorio raíz del CD.
4. Seleccione el idioma que desee utilizar en la lista desplegable. Haga clic en **OK** (Aceptar).

fig. 1



5. Se muestra la ventana del programa de instalación de Trajexia Tools. Haga clic en **Next** (Siguiente).

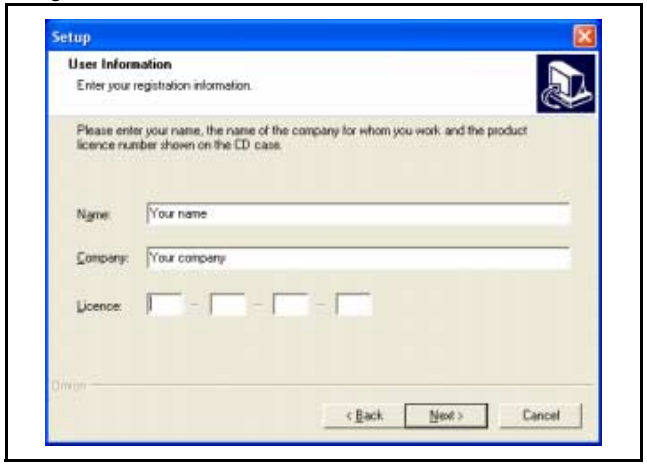
fig. 2

6. Haga clic en **Yes** (Sí) para aceptar el contrato de licencia y continuar.

fig. 3

7. Escriba su nombre en el campo **Name** (Nombre).
8. Escriba el nombre de la empresa en el campo **Company** (Empresa).
9. Escriba el número de licencia en los campos **Licence** (Licencia). El número de licencia de usuario está en la etiqueta de la caja del CD de Trajexia Tools.
10. Haga clic en **Next** (Siguiente).

fig. 4

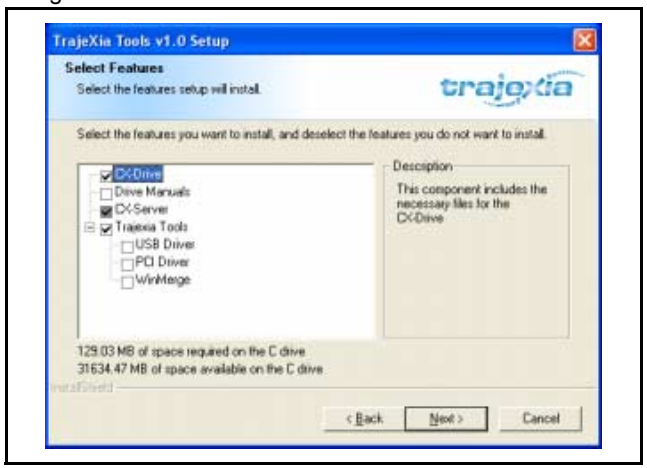


11. Haga clic en **Yes** (Sí).

fig. 5

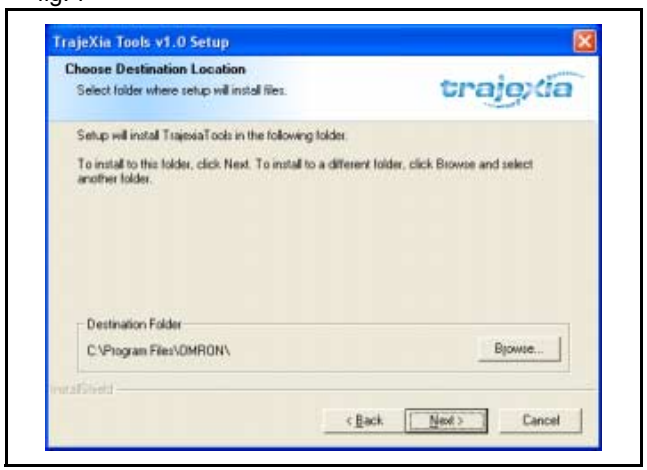
12. Haga clic en **Next** (Siguiente).

fig. 6



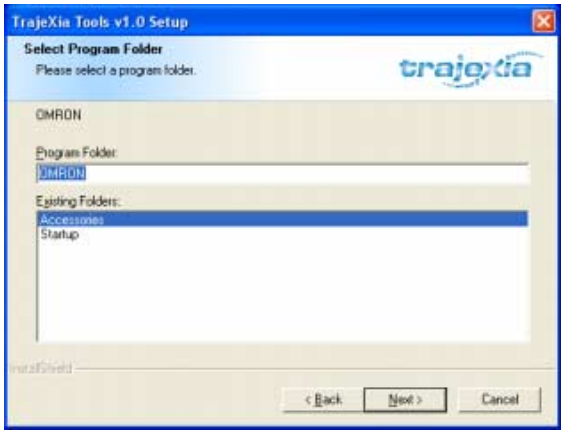
13. Haga clic en **Next** (Siguiente).

fig. 7



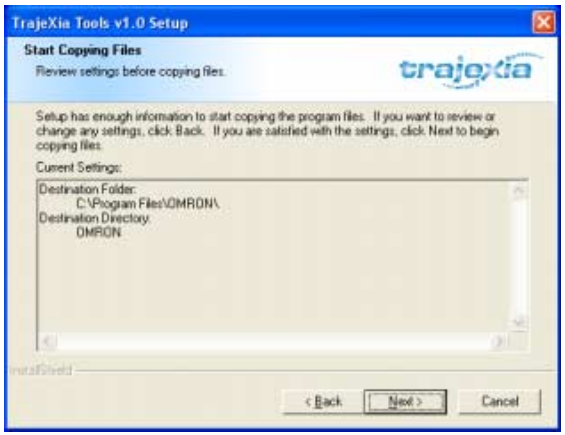
14. Haga clic en **Next** (Siguiente).

fig. 8



15. Haga clic en **Next** (Siguiente).
16. El programa de instalación de Trajexia Tools copia los archivos en el PC. Esto puede durar unos minutos.

fig. 9



17. Haga clic en **Finish** (Finalizar). Se muestra la ventana del archivo léame de CX-Drive. Cierre esta ventana.

fig. 10

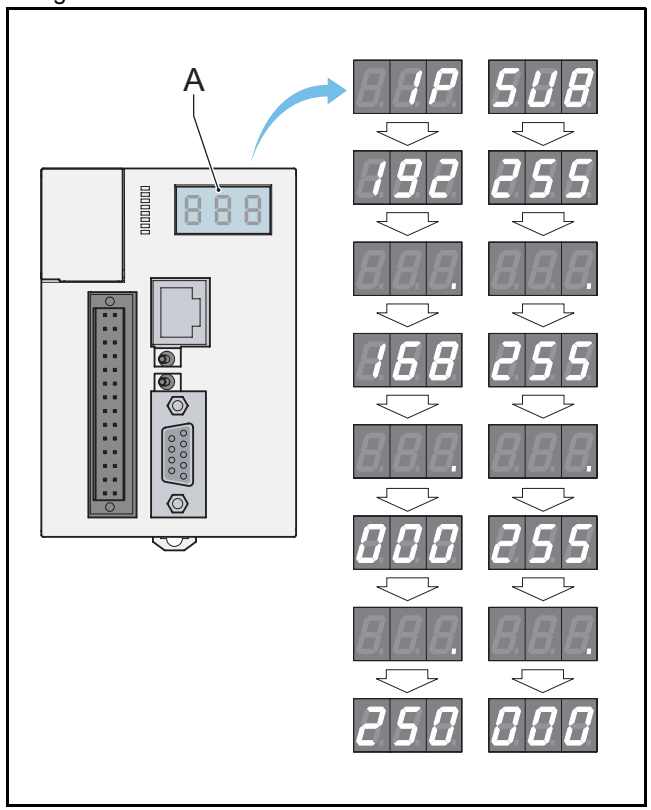
5.2.3 Conexión a la TJ1-MC__

Necesita un cable Ethernet de conexión o cruzado para conectar el PC a la TJ1-MC__.

Nota
Si trabaja offline, se puede utilizar el simulador. La simulación permite a Trajexia Tools conectarse a un controlador virtual. Ésta es la forma de programar offline. El “simulador” no reconoce todos los comandos específicos para la TJ1-MC__.

1. Conecte el sistema Trajexia a la alimentación principal.
2. Si necesita volver a ver la dirección IP y la máscara de subred de la TJ1-MC__, desconecte la alimentación del sistema Trajexia y vuelva a conectarla.
3. Conecte el cable Ethernet al puerto Ethernet del PC.
4. Conecte el cable Ethernet al puerto Ethernet de la TJ1-MC__. La dirección IP de la TJ1-MC__ se muestra 4 veces en el display LED.
5. Al iniciar el software Trajexia Tools, se intenta comunicar con el controlador. Al iniciar Trajexia Tools por primera vez, los ajustes de comunicaciones no son los adecuados, por lo que tendrá que cancelar (consulte la figura 12) y establecer sus ajustes de comunicaciones.

fig. 11



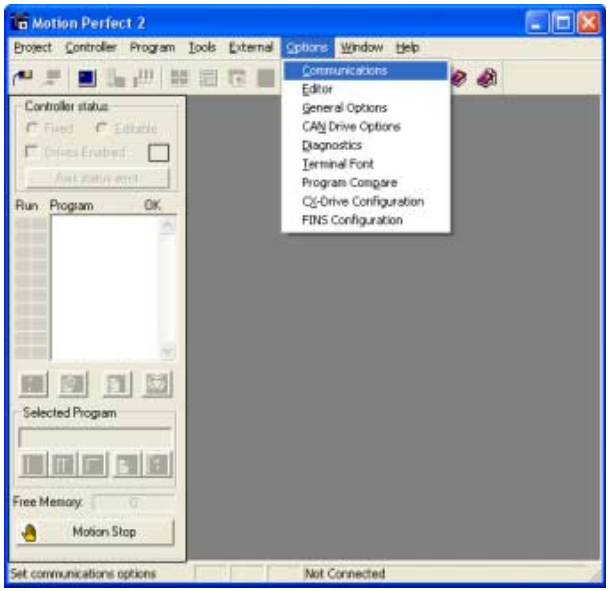
6. Inicie el programa Trajexia Tools en el PC.
   Seleccione en el menú **Inicio** de Windows:
   - **Programas**
   - **OMRON**
   - **Trajexia Tools**
   - **Trajexia Tools**

7. Se muestra la ventana de inicio **Trajexia Tools**. Espere hasta que esté visible el botón **Cancel** (Cancelar). A continuación, haga clic en **Cancel** (Cancelar).

fig. 12

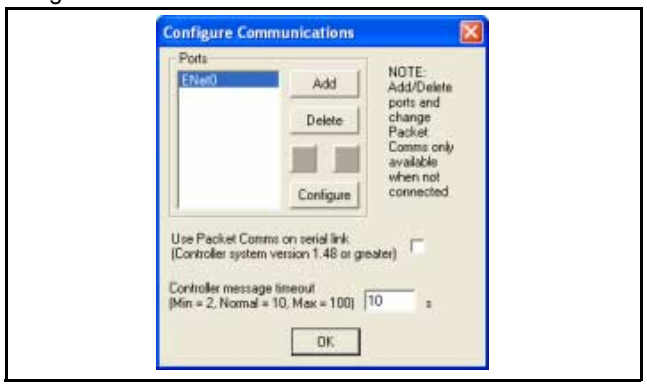
8. Seleccione el menú:
   - **Options** (Opciones)
   - **Communications** (Comunicaciones)

fig. 13



9. Asegúrese de que **ENet0** en la lista está seleccionado.
10. Haga clic en **Configure** (Configurar).

fig. 14



11. Escriba **192.168.0.250** en el campo **Server name/IP address** (Nombre de servidor/Dirección IP).
12. Haga clic en **OK** (Aceptar).

fig. 15

13. Haga clic en **OK** (Aceptar).

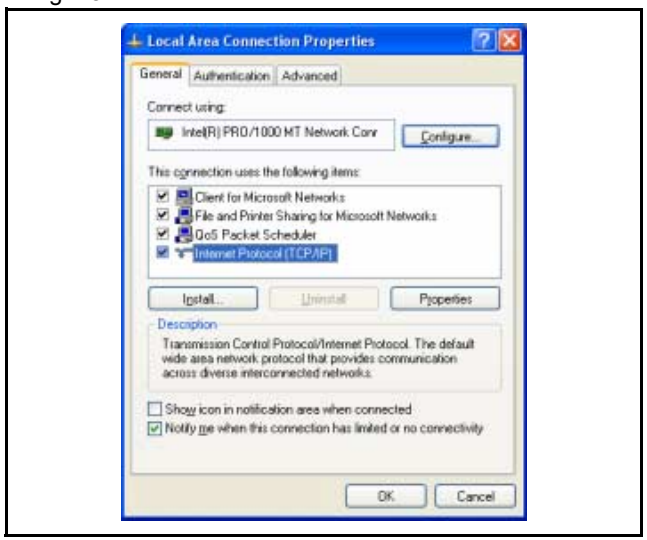
fig. 16

14. Abra el Panel de control de Windows en el PC.
15. Haga doble clic en el icono **Conexiones de red**.
16. Haga clic con el botón derecho en el icono **Conexión de área local**. Haga clic en el menú **Propiedades**.

fig. 17

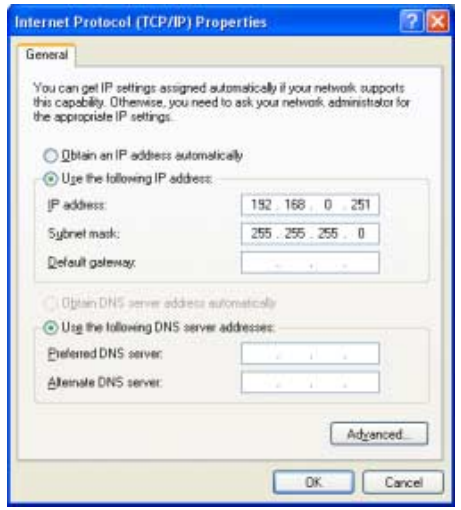
17. Haga clic en la ficha **General**.
18. Seleccione **Protocol Internet (TCP/IP)** en la lista.
19. Haga clic en **Propiedades**.

fig. 18



20. Haga clic en la ficha **General**.
21. Seleccione **Usar la siguiente dirección IP**.
22. Escriba la dirección **192.168.0.251** en el campo **Dirección IP**.
23. Escriba **255.255.255.0** en el campo **Máscara de subred**.
24. Haga clic en **OK** (Aceptar).
25. Haga clic en **OK** (Aceptar).
26. Cierre la pantalla Conexiones de red.

fig. 19



5.3 Proyectos

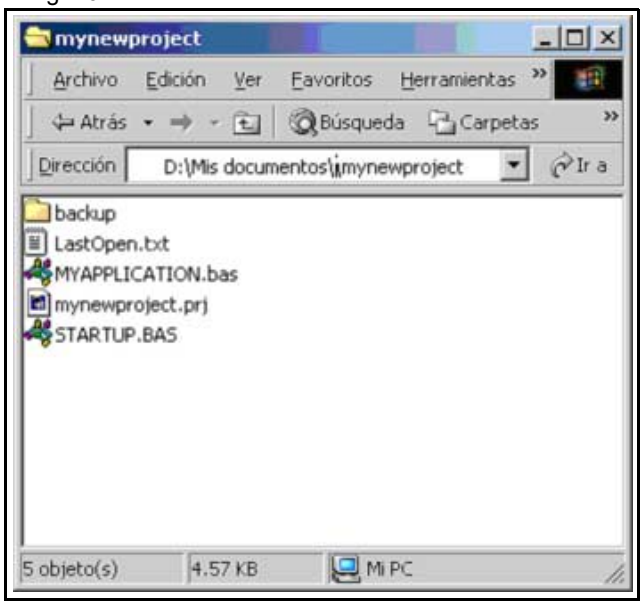
5.3.1 Proyectos de Trajexia Tools

Los proyectos facilitan el proceso de diseño y desarrollo de una aplicación. En el PC que se utiliza para programar el sistema hay disponibles una copia en el disco duro de todos los programas, parámetros y datos.
El usuario define un proyecto, Trajexia Tools mantiene la coherencia entre el proyecto en el PC y el sistema Trajexia.
Los programas que se editan se duplican automáticamente en el PC.

Un proyecto de Trajexia Tools consta de un archivo **proyecto.prj** que contiene la configuración de Trajexia, los parámetros del servodriver y un conjunto de programas **.bas**. Estos archivos se almacenan en una carpeta con el mismo nombre del archivo **.prj**.

Nota:
Un programa creado en un ordenador no se puede abrir correctamente en otro. Para evitar este problema, copie todo el directorio de proyecto de un ordenador al otro.

fig. 20



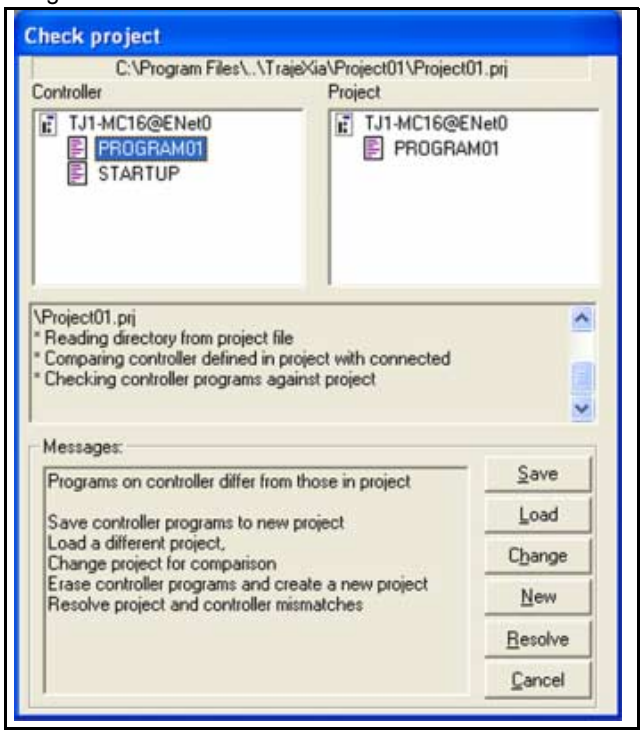
5.3.2 Ventana de comprobación de proyecto

Trajexia Tools inicia la ventana Check Project (Comprobar proyecto) cuando se conecta al sistema Trajexia. Se realiza una comparación entre los archivos de programa del sistema Trajexia y los archivos de programa del PC.

Si los archivos de programa son distintos, en la ventana de comprobación de proyecto se muestra:

- **Save** (Guardar)
- **Load** (Cargar)
- **Change** (Cambiar)
- **New** (Nuevo)
- **Resolve** (Resolver)
- **Cancel** (Cancelar)

fig. 21



Save (Guardar)

Carga el proyecto que se encuentra en el sistema Trajexia en el PC. Se sobrescribe un proyecto del mismo nombre en el PC. Antes de guardar en el PC, asegúrese primero de que el programa del PC tiene una copia de seguridad.

Load (Cargar)

Descarga el proyecto que se encuentra en el PC en el sistema Trajexia.
Se sobrescribe el proyecto del sistema Trajexia. Antes de cargar en el sistema Trajexia, asegúrese primero de que el proyecto del sistema Trajexia tiene una copia de seguridad.

Change (Cambiar)

Permite abrir un proyecto que está en el PC distinto del proyecto predeterminado.
Si el proyecto del sistema Trajexia y el PC no son iguales, puede utilizar **Change** (Cambiar) para seleccionar otro proyecto en el PC. Trajexia Tools vuelve a comparar ambos proyectos. Esto es necesario al trabajar en varias aplicaciones con proyectos distintos.

New (Nuevo)

Elimina el proyecto que está en el sistema Trajexia e inicia un nuevo proyecto en el PC. Trajexia Tools crea un nuevo directorio con el nombre de proyecto que contiene el nuevo archivo de proyecto. El nombre del directorio debe ser el mismo que el nombre del proyecto; de lo contrario, no se podrá abrir el proyecto.

Resolve (Resolver)

Compara el proyecto que está en Trajexia Tools con el proyecto que está en el PC. Esta opción ofrece la posibilidad de **Save** (Guardar), **Load** (Cargar) o **Examine** (Examinar) las diferencias individualmente por cada programa individual dentro del proyecto. Esta opción permite una modificación de un programa offline mediante el simulador y una descarga del mismo programa a la TJ1-MC__. Esta opción también permite que varias personas trabajen en mismo proyecto simultáneamente.

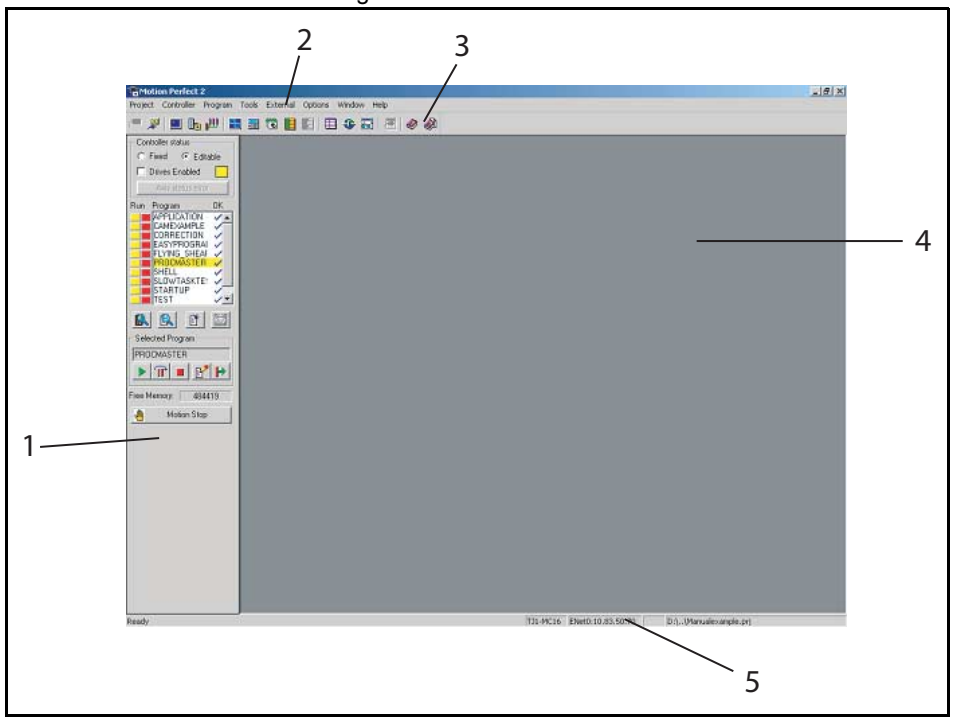
Cancel (Cancelar)

Para el proceso de conexión. Trajexia Tools se inicia en modo desconectado.

5.4 Ventana de aplicación de Trajexia Tools

La ventana de aplicación de Trajexia Tools consta de estas partes:
1. Panel de control
2. Barra de menú
3. Barra de herramientas
4. Espacio de trabajo
5. Barra de estado

fig. 22



5.4.1 Panel de control

El panel de control ofrece una forma rápida y sencilla de acceder a los controles más utilizados para gestionar y poner en marcha un proyecto.

5.4.2 Barra de menú

La barra de menú tiene estos elementos:
- **Project** (Proyecto)
- **Controller** (Controlador)
- **Program** (Programa)
- **Tools** (Herramientas)
- **External** (Externo)
- **Options** (Opciones)
- **Window** (Ventana)
- **Help** (Ayuda)

Los menús se describen en detalle en la sección “Descripciones de menús” (p. 196).

5.4.3 Barra de herramientas

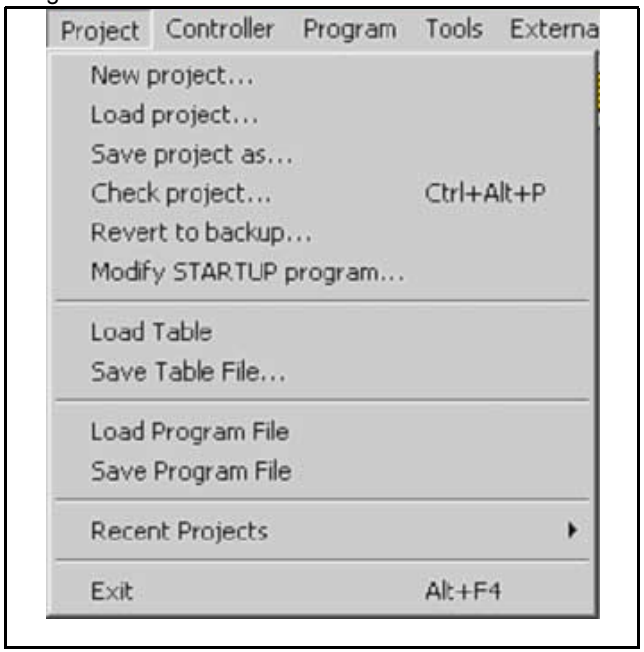
Connect (Conectar)	Conecta Trajexia Tools al sistema Trajexia. Consulte Connect (Conectar) en “Menú Controller (Controlador)” (p. 198).
Disconnect (Desconectar)	Desconecta Trajexia Tools del sistema Trajexia. Consulte Disconnect (Desconectar) en “Menú Controller (Controlador)” (p. 198).
Terminal	Editor de línea de comandos. Consulte Terminal en “Menú Tools (Herramientas)” (p. 205).
Axis Parameters (Parámetros de eje)	Consulte Axis Parameters (Parámetros de eje) en “Menú Tools (Herramientas)” (p. 205).
Intelligent drives (Variadores inteligentes)	Consulte Intelligent drives (Variadores inteligentes) en “Menú Tools (Herramientas)” (p. 205).
Oscilloscope (Osciloscopio)	El osciloscopio de software se puede utilizar para realizar un seguimiento de los parámetros de eje y de movimiento. Ayuda al desarrollo del programa y a la puesta en marcha del sistema. Consulte Oscilloscope (Osciloscopio) en “Menú Tools (Herramientas)” (p. 205).
Keypad (Teclado)	No implementado en Trajexia.
Jog Axis (Eje jog)	Esta ventana permite al usuario mover manualmente los ejes en el Trajexia. Consulte Jog Axis (Eje jog) en “Menú Tools (Herramientas)” (p. 205).
Digital IO (E/S digitales)	Consulte Digital IO Status (Estado de E/S digitales) en “Menú Tools (Herramientas)” (p. 205).
Analog input (Entrada analógica)	Consulte Analog input (Entrada analógica) en “Menú Tools (Herramientas)” (p. 205).
TABLE values (Valores TABLE)	Consulte los valores TABLE y VR en “Menú Tools (Herramientas)” (p. 205).
VR values (Valores VR)	Consulte los valores TABLE y VR en “Menú Tools (Herramientas)” (p. 205).
Watch variables (Variables de inspección)	No implementado en Trajexia.
Simulator (Simulador)	No implementado completamente en Trajexia.
Trajexia Tools Help (Ayuda de Trajexia Tools)	Abre la ayuda de Trajexia Tools.
Trio BASIC Help (Ayuda de Trio BASIC)	Abre la ayuda de Trio BASIC.

5.5 Descripciones de menús

5.5.1 Menú Project (Proyecto)

El menú **Project** (Proyecto) permite crear, cargar y guardar proyectos de Trajexia Tools.

fig. 23



New project (Nuevo proyecto)
Elimina el proyecto que está en el sistema Trajexia e inicia un nuevo proyecto en el PC. Trajexia Tools crea un nuevo directorio con el nombre de proyecto que contiene el nuevo archivo de proyecto. El nombre del directorio debe ser el mismo que el nombre del proyecto; de lo contrario, no se podrá abrir el proyecto.

Load project (Cargar proyecto)
Abre un proyecto que está en el PC. Trajexia Tools descarga el proyecto que se encuentra en el PC en el sistema Trajexia. Se sobrescribe el proyecto del sistema Trajexia. Antes de cargar en el sistema Trajexia, asegúrese primero de que el proyecto del sistema Trajexia tiene una copia de seguridad.

Save project as... (Guardar proyecto como)
Carga el proyecto que se encuentra en el sistema Trajexia en el PC y lo guarda como un directorio en el disco duro del PC.

Check project (Comprobar proyecto)
Se realiza una comparación entre el proyecto del sistema Trajexia y el PC. Se comparan las sumas de comprobación y el contenido de los programas.

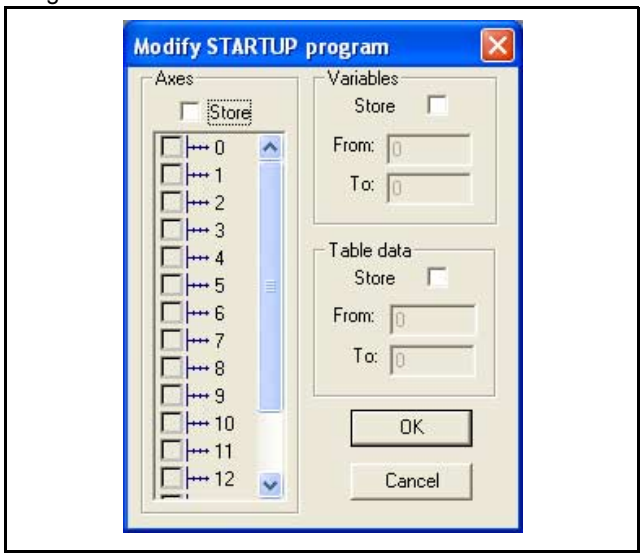
Revert to backup... (Volver a copia de seguridad)
Siempre que Trajexia Tools se conecta al MC16 compara el proyecto del controlador y el del PC y, a continuación, realiza una copia de seguridad. **Revert to backup** (Volver a copia de seguridad) se puede utilizar cuando desee cancelar todas las modificaciones efectuadas en el proyecto y en los programas BASIC mientras está conectado al controlador.

De este modo, los programas BASIC se cambiarán por las versiones del directorio de copia de seguridad.

Modify STARTUP program (Modificar programa STARTUP)

El programa **Startup** comprueba el número de nodos de un sistema MECHATROLINK-II en el proyecto. Utilice la opción **Modify STARTUP program** (Modificar programa STARTUP) para cambiar un programa de inicio creado mediante la ventana **Intelligent Drives** (Variadores inteligentes).

fig. 24



Load table (Cargar tabla)

Se puede cargar una lista de valores de tabla desde un archivo externo con la extensión ***.lst** o ***.bas**. Importa los valores y los almacena en valores TABLE.

Save table file... (Guardar archivo de tabla)

Guarda un archivo ***.lst** o ***.bas** desde los valores TABLE en el directorio de proyecto.

Load program file (Cargar archivo de programa)

Carga un archivo que contiene código que se puede ejecutar en una tarea.

Save program file (Guardar archivo de programa)

Guarda el archivo de programa en formato **.txt**.

Recent projects (Proyectos recientes)

Puede abrir los proyectos recientes que se han editado con el software Trajexia Tools.

Exit (Salir)

Cierra la aplicación Trajexia Tools.

5.5.2 Menú Controller (Controlador)

El menú Controller (Controlador) permite configurar las comunicaciones entre el PC y el sistema Trajexia, así como controlar dicho sistema.

fig. 25

Connet (Conectar)

Se conecta con el sistema Trajexia e inicia el administrador de proyectos. Disponible si Trajexia Tools se desconecta del sistema.

Disconnect (Desconectar)

Se desconecta del sistema Trajexia. Disponible cuando Trajexia Tools está conectado al sistema.

Connect to simulator (Conectar con simulador)

No implementado completamente para Trajexia.

Reset the controller (Reset del controlador)

Realiza un reset por software del sistema Trajexia. La aplicación Trajexia Tools se desconecta del sistema Trajexia.

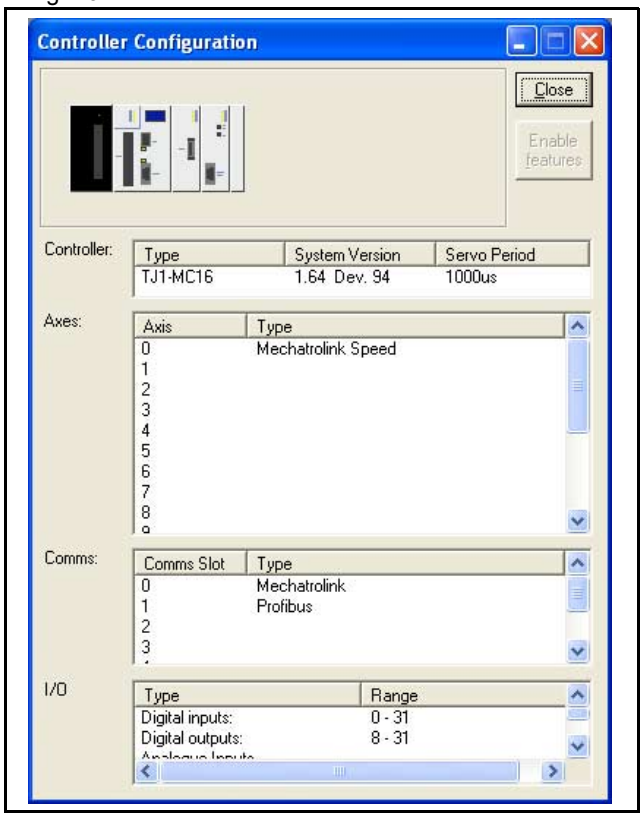
Recover project from EPROM (Recuperar proyecto de EPROM)

Efectúa un reset del sistema Trajexia y restaura los programas que se encuentran en la EPROM en el PC.

Controller configuration (Configuración del controlador)

Muestra la pantalla de configuración del hardware del controlador que está conectado al PC.
Controlador (Controlador): el PC está conectado a un controlador Motion Trajexia (TJ1-MC__) con software 1.64 Dev. 94. El período de servo es de 1.000 µs.
Axis (Eje): muestra los ejes que están disponibles.
Comms (Comunicaciones): la función de comunicaciones del controlador Motion Trajexia.
I/O (E/S): el tipo y el rango de las salidas y entradas digitales, analógicas y virtuales.

fig. 26



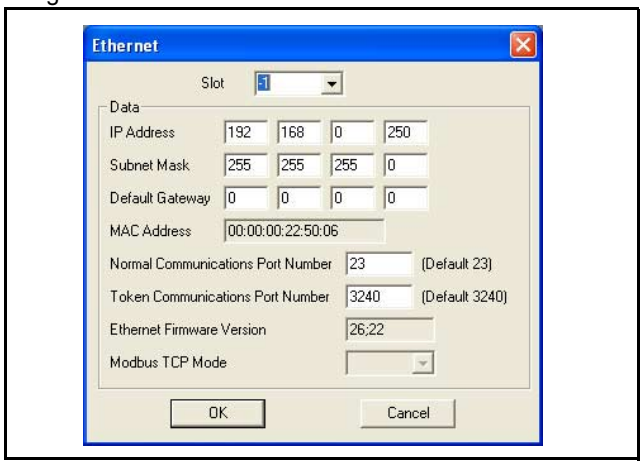
CANIO

No implementado para Trajexia.

Ethernet configuration (Configuración Ethernet)

Permite cambiar la configuración Ethernet y la dirección IP del hardware del controlador.
Slot (Entrada): siempre –1 para Trajexia.
IP address (Dirección IP): la dirección IP del controlador Motion Trajexia. No es la misma que la dirección IP del PC.
Subnet Mask (Máscara de subred): la máscara de subred del controlador Motion Trajexia y la del PC deben ser la misma.
Default gateway (Gateway predeterminado): un nodo de la red que sirve de entrada a otra red. Sólo es necesario si Trajexia tiene que comunicarse con un dispositivo de otra subred.
MAC address (Dirección MAC): dirección de control de acceso a medios, una dirección de hardware que identifica de forma exclusiva a cada nodo de la red. Esta dirección es de sólo lectura.
Normal Communications Port Number (Número de puerto de comunicaciones normal): el puerto TCP utilizado para comunicarse con Trajexia Tools.
Token Communications Port Number el puerto TCP utilizado para comunicarse con el control ActiveX Motion de PC.

fig. 27



Enable features (Activar funciones)

No implementado para Trajexia.

Enable editing (Activar edición)

Configura Trajexia para trabajar con la versión RAM de los programas. En este modo, los programas se pueden editar. Trajexia mantiene los programas almacenados en RAM (y las variables globales) mediante la batería de respaldo. Esta opción cambia internamente el parámetro **POWER_UP**.
Si **POWER_UP** está configurado en 0, Trajexia utiliza al inicio los programas almacenados en RAM mediante la batería de respaldo, incluso si se han guardado en la memoria flash. Esta función sólo está disponible cuando **POWER_UP = 1**.

Fix project into EPROM (Fijar proyecto en EPROM)

Copiar los programas del controlador en la EPROM flash. Se sobrescriben todos los programas que se encuentran actualmente en la EPROM. La función cambia **POWER_UP** a 1 y la RAM se sobrescribe con el contenido de la EPROM tras el encendido. En este modo, los programas no se pueden editar. Esta función sólo está disponible cuando **POWER_UP = 0**.

Load System Software (Cargar software de sistema)

Trajexia tiene una EPROM flash para almacenar los programas de usuario y el software del sistema. Utilice la opción Load System Software (Cargar software de sistema) para actualizar el software de sistema a una versión más reciente.
Se abre un cuadro de diálogo para asegurarse de que realiza una copia de seguridad del proyecto y de que desea continuar.
Se abre un selector de archivos estándar. Seleccione el archivo que necesite.

OMRON recomienda cargar una nueva versión del software de sistema únicamente cuando así lo recomiende el distribuidor de OMRON.

Precaución
No cargue software que no esté especificado para el controlador Motion Trajexia. Cargue únicamente versiones que estén diseñadas específicamente para su uso con Trajexia.
Todas las demás versiones no funcionan.

Se abre un cuadro de diálogo para asegurarse de que desea continuar. Pulse OK (Aceptar) para comenzar. El proceso de la EPROM flash durará 7 minutos aproximadamente.

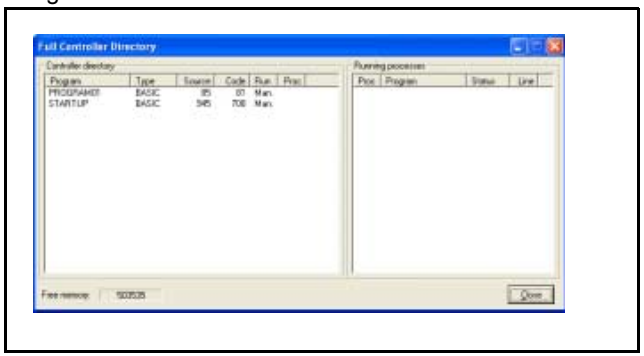
Precaución
No pare el proceso de actualización de software.
Una interrupción en el proceso de comunicaciones dañará la unidad Trajexia. Si no puede recuperar la unidad Trajexia después del proceso de la EPROM flash, póngase en contacto con su representante de ventas.

Cuando la descarga esté completa, una suma de comprobación confirma que el proceso de la EPROM flash es correcto.
Para completar el proceso, seleccione **Yes** (Sí) en el cuadro de diálogo de confirmación.
Abra la ventana **Controller Configuration** (Configuración de controlador) para comprobar la nueva versión del sistema.

Full directory (Directorio completo)

Se utiliza para comprobar el contenido y la estructura de archivos del directorio TJ1-MC__.

fig. 28



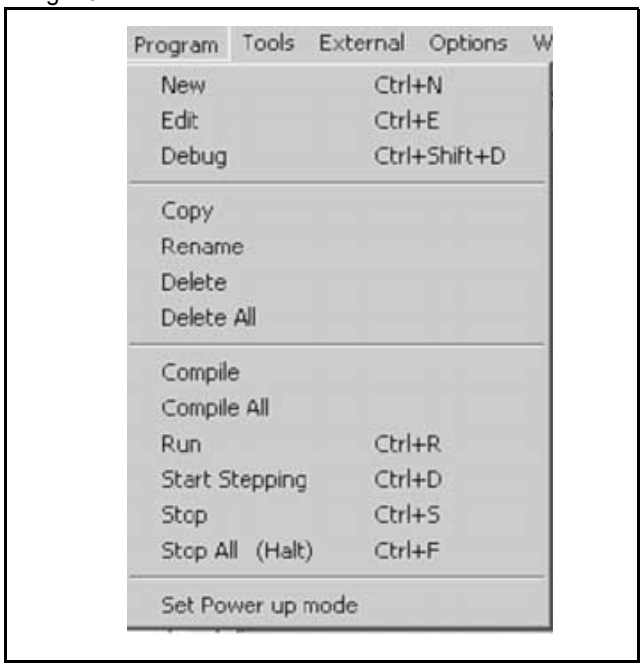
Lock/Unlock (Bloquear/Desbloquear)

Bloquear el sistema Trajexia para impedir el acceso no autorizado. Cuando el sistema Trajexia está bloqueado no es posible enumerar, editar o guardar ninguno de los programas Trajexia. El sistema Trajexia no está disponible a través del software Trajexia Tools, aunque los cuadros de diálogo de terminal y de desbloqueo están disponibles.
Escriba el número 7 para bloquear el sistema. Utilice el mismo número para desbloquearlo.

5.5.3 Menú Program (Programa)

El menú Program (Programa) contiene elementos de menú que permiten la programación de un proyecto Trajexia.

fig. 29



New (Nuevo)
Crea un nuevo programa.

Edit (Editar)
Abre un proyecto para su edición.
También puede iniciar el editor desde el panel de control. Desde el menú de programa primero se mostrará un cuadro de diálogo de selección de archivo para confirmar el archivo que desea editar.
El editor de Trajexia Tools está diseñado para actuar de forma similar a cualquier editor de textos simple que se encuentra en un PC. Las operaciones estándar, como las funciones de edición por bloques, búsqueda y reemplazo de texto e impresión, también se admiten y cumplen las teclas de método abreviado de Windows estándar. Además, proporciona resaltado de sintaxis BASIC, formato de programa y funciones de depuración de programas.

Debug (Depurar)
Comprueba la sintaxis de un programa y ofrece soluciones posibles. El programa se abre en un modo de seguimiento especial que lo ejecuta línea a línea. Puede establecer puntos de ruptura en el programa para ejecutarlo hasta que se llegue a un punto de ruptura. La línea de código actual está resaltada en la ventana de depuración.
Cuando el programa se ejecuta en modo de depuración, cualquier editor abierto se configura en dicho modo y es de sólo lectura.

Copy (Copiar)
Copia el contenido de un programa en otro programa.

Rename (Cambiar nombre)
Cambia el nombre de un programa.

Delete (Eliminar)
Elimina el programa de la estructura de archivos.

Delete all (Eliminar todo)
Elimina todos los programas de la estructura de archivos.

Compile (Compilar)
Compila el programa actual del proyecto.

Compile all (Compilar todo)
Compila todos los programas del proyecto.

Run (Ejecutar)
Ejecuta el programa actual en el proceso especificado.

Start stepping (Iniciar paso a paso)
Ejecuta el programa actual en el proceso especificado en el modo paso a paso (línea a línea).

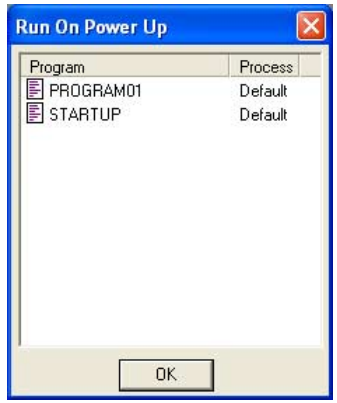
Stop (Parar)
El comando Stop (Parar) para el programa del controlador TJ1-MC__. No es lo mismo que parar el movimiento. El programa se para al final del ciclo de CPU. Los servomotores mantienen su posición.

Stop all (Parar todo)
El comando Stop all (Parar todo) para todos los programas del controlador TJ1-MC__. No es lo mismo que parar el movimiento. Los programas se paran al final del ciclo de CPU. Los servomotores mantienen su posición.

Set power up mode (Establecer modo de encendido)

Es posible hacer que los programas de la TJ1-MC__ se ejecuten automáticamente cuando se inicie el sistema. Seleccione **Set Powerup Mode** (Establecer modo de encendido) para abrir el cuadro de diálogo **Run On Power Up** (Ejecutar en el encendido). Seleccione el programa que desee ejecutar automáticamente. Aparecerá un pequeño menú desplegable a la derecha de la ventana. Si desea que Trajexia asigne el proceso en el que se ejecutará, elija el predeterminado como el número de proceso. También puede seleccionar específicamente el proceso.

fig. 30



5.5.4 Menú Tools (Herramientas)

A las herramientas de Trajexia Tools se puede acceder mediante el menú Tools (Herramientas) o el botón de la barra de herramientas.

fig. 31

Axis parameters (Parámetros de eje)

La ventana Axis Parameters (Parámetros de eje) permite monitorizar y cambiar los parámetros de Motion Control de cualquier eje del sistema Trajexia.
La ventana contiene los parámetros divididos en dos bancos:

- Banco 1 (mitad superior de la ventana): contiene parámetros que puede cambiar el usuario.
- Banco 2 (mitad inferior de la ventana): contiene parámetros que establece el software del sistema Trajexia, a medida que el sistema procesa los comandos y monitoriza el estado de las entradas externas.

El separador que divide los dos bancos de datos se puede mover con el ratón.
Cuando el usuario cambia un parámetro de unidad, todos los parámetros que utilizan este valor de parámetro se vuelven a leer y ajustar mediante este factor.
Algunos ejemplos de los tipos de parámetros que están afectados por este parámetro son:

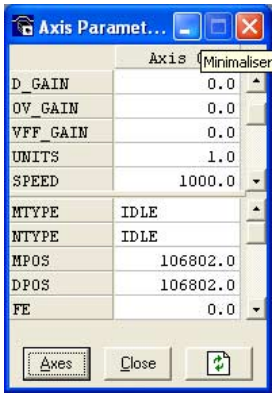
- **SPEED**
- **ACCEL**
- **MPOS**

El parámetro **axistatus** muestra el estado actual del eje. El color de los caracteres en el valor de parámetro indica el estado:

- Verde: ningún error
- Rojo: Error

Carácter	Descripción
W	Alarma de FE
A	Error de comunicaciones de driver
M	Error de driver remoto
F	Límite de marcha directa
R	Límite de marcha inversa
D	Entrada de referencia

fig. 32



Carácter	Descripción
H	Entrada de retención de realimentación
E	Error de seguimiento
X	Límite de software de marcha directa
Y	Límite de software de marcha inversa
C	Cancelación de movimiento
O	Error de encoder

Las opciones de la ventana Axis Parameters (Parámetros de eje) son:
Axes (Ejes): selecciona los ejes para los que se muestran los datos.
Refresh (Actualizar): para reducir la carga en el sistema Trajexia, los parámetros del banco 1 son de sólo lectura cuando la pantalla se muestra por primera vez o cuando se establece el parámetro. Si se cambia un valor de parámetro, el valor mostrado puede ser incorrecto. El botón Refresh (Actualizar) fuerza a que Trajexia Tools vuelva a leer la selección completa.

Nota:
Si cambia un valor de parámetro, debe actualizar la pantalla antes de realizar otro cambio.

Intelligent drives (Variadores inteligentes)

La opción Intelligent drives ofrece acceso a los programas de configuración y de inicio de cada uno de los equipos que se pueden configurar.
La herramienta de intelligent drives muestra la configuración de Trajexia detectada en el encendido. Al hacer clic en la unidad, aparecerá la ficha siguiente.
En la parte superior de la ventana se muestra la TJ1-MC__ con sus distintas unidades. En el ejemplo:

- TJ1-MC__ con el número de unidad –1
- TJ1-ML__ con el número de unidad 0
- TJ1-PRT con el número de unidad 1
- TJ1-FL02 con el número de unidad 2

Si hay varias TJ1-ML__ en el sistema, se muestran más fichas.
Con la opción **Modify STARTUP program** (Modificar programa STARTUP) se crea un programa STARTUP para la configuración detectada.
En la ficha de la TJ1-ML__ se puede ver la información correspondiente a los esclavos MECHATROLINK-II detectados (incluidos los módulos de variador y E/S).
Al hacer clic en el botón **Config** (Configurar) (sólo disponible para servo y variador), aparece la siguiente ventana:
Ficha **Status** (Estado):

- **Drive ID/Motor ID/Firmware Version** (ID de variador/ID de motor/Versión de firmware): muestra información del servodriver y del servomotor.
- **Drive Status** (Estado de variador) muestra el contenido del canal **DRIVE_STATUS** para dicho eje.
- **Drive I/O** (E/S de variador) muestra el contenido del canal **DRIVE_INPUTS** para dicho eje.
- **Drive Clear** (Borrado del servodrive) ejecuta **DRIVE_CLEAR** (borrado de alarma de servodriver) para dicho eje.
- **Drive Reset** (Reset del servodrive) ejecuta **DRIVE_RESET** (encendido de software) para dicho eje.
- **Drive monitor** (Monitor de servodrive) selecciona el monitor que se actualizará en **DRIVE_MONITOR**.

fig. 33

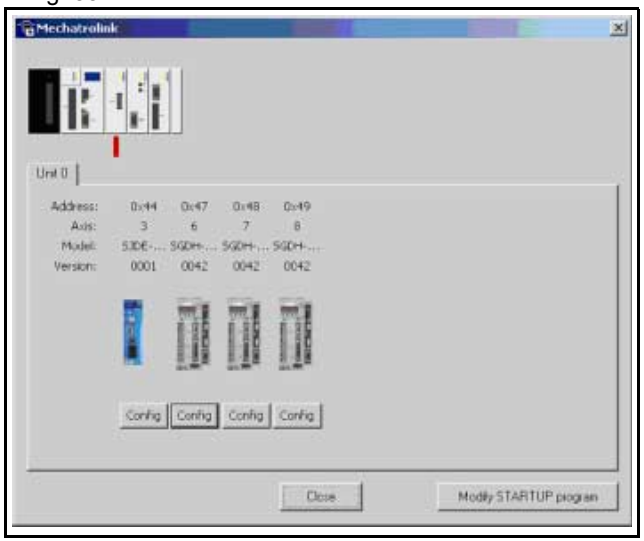
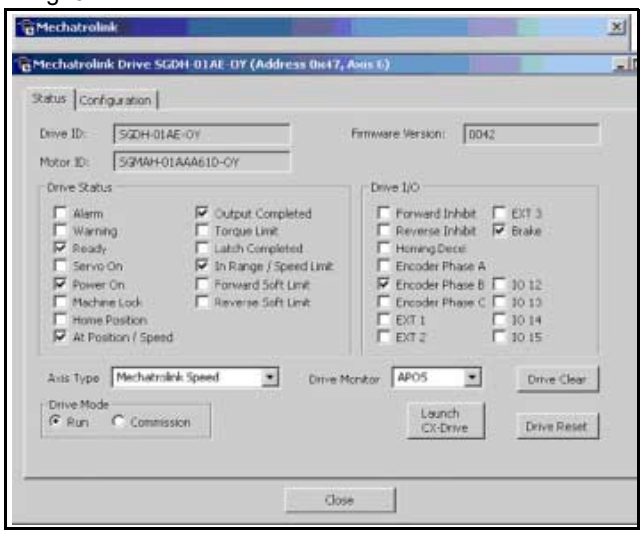


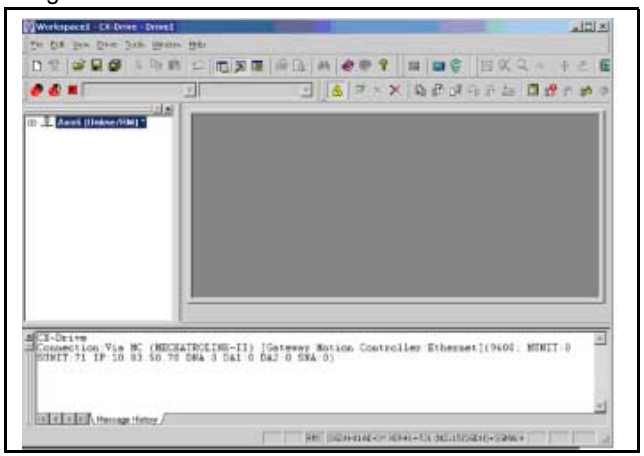
fig. 34



- **Axis Type** (Tipo de eje) selecciona el valor **ATYPE** para este eje. El valor indicado aquí se incluirá en el programa STARTUP.
- **Drive Mode** (Modos del drive): Run (Funcionamiento) o Commission (Puesta en marcha).
  - Cuando el eje se configura en modo de funcionamiento, su estado de funcionamiento y todos los movimientos se controlan por completo con los programas que se ejecutan en la TJ1-MC__.
  - Cuando el eje se configura en modo de puesta en marcha, el funcionamiento y los movimientos se controlan de forma externa mediante FINS. Este modo se ha implementado para su uso con la función Jog y la configuración desde CX-Drive. De este modo se evitan conflictos con los programas. Durante la puesta en marcha, los programas consideran al eje como virtual.
    Es posible leer y escribir parámetros desde el variador simultáneamente desde los programas o mediante FINS, independientemente del modo.
- Launch CX-Drive (Iniciar CX-Drive): desde Trajexia Tools sólo es posible leer y escribir parámetros de un servodriver. Si se necesita más funcionalidad de servodriver, por ejemplo, leer código de alarma, jog, establecer rigidez, autotuning, etc., es necesario iniciar CX-Drive. Al hacer clic en este botón se inicia CX-Drive conectado al eje actual mediante la TJ1-ML__.

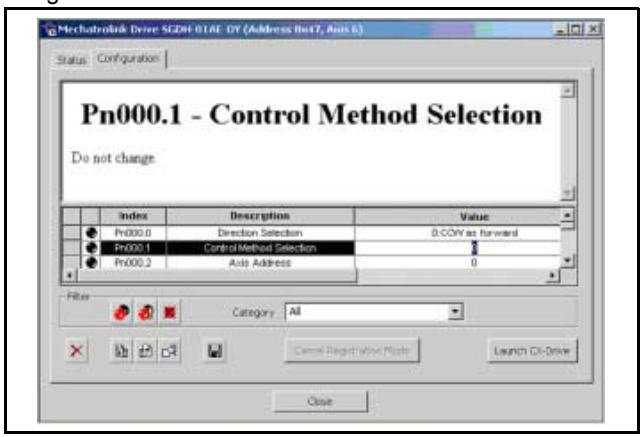
La única función de servodriver que no se admite en CX-Drive a través de MECHATROLINK-II es la de Trace (Graficado), pero se puede utilizar el osciloscopio de Trajexia Tools en su lugar.
Si cambia cualquier parámetros del servodriver mediante CX-Drive, Trajexia Tools no lo advierte automáticamente. Procure evitar que haya un parámetro distinto en el servodriver y en el proyecto.

fig. 35



Ficha **Configuration** (Configuración).
La ficha **Configuration** (Configuración) muestra una ventana de editor de parámetros idéntica a la de CX-Drive. Para obtener más detalles, compruebe la información de CX-Drive.
La nueva funcionalidad es:
Botón **Save** (Guardar): guardar los parámetros de servo actuales en el proyecto Trajexia (en el archivo ***.prj**).
Cancel Registration Mode (Cancelar modo de registro): cuando el registro está activo en el servodriver, para obtener una respuesta rápida y fiable, no es posible escribir parámetros. Es lo mismo que ejecutar **REGIST(-1)**.
Launch CX-Drive (Iniciar CX-Drive): igual que el botón explicado en la ficha Status (Estado).

fig. 36

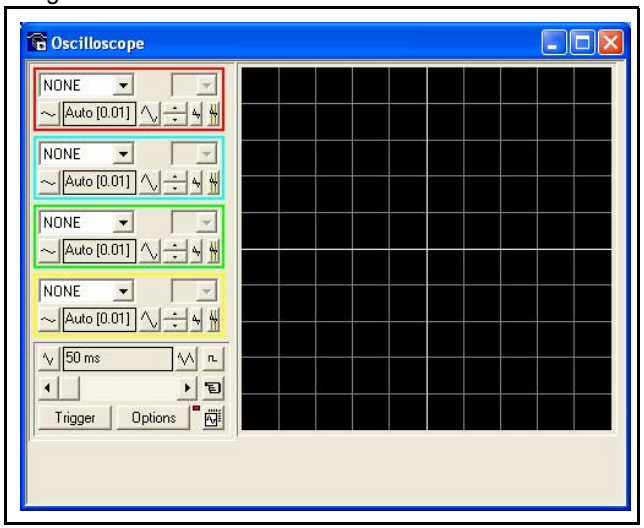


Oscilloscope (Osciloscopio)

El osciloscopio de software se puede utilizar para realizar un seguimiento de los parámetros de eje y de movimiento. Es una ayudar para el desarrollo del programa y la puesta en marcha del sistema.
Existen cuatro canales, cada uno de ellos capaz de registrar 1 muestra según el valor de **SERVO_PERIOD** con ciclo manual o activación vinculada a programa.
El controlador registra los datos con la frecuencia seleccionada y carga la información en el osciloscopio que se mostrará. Si se utiliza una base temporal mayor, los datos se recuperan en secciones y el gráfico se traza en secciones en la pantalla. El momento en el que el controlador comienza a registrar los datos necesarios depende de si el controlador está en modo manual o de activación por trigger.

- **Modo de programa**: el osciloscopio comienza a registrar datos cuando se envía una instrucción de activación desde el programa del controlador.
- **Modo manual**: el osciloscopio empieza a registrar datos inmediatamente.

fig. 37



Canales del osciloscopio

Cada canal del osciloscopio dispone de controles para los cuatro bloques de control de canal. Cada bloque de control de canal tiene un borde color para indicar el color de visualización para dicho canal.
Los canales son los siguientes:

Parámetro que se mostrará los parámetros que el osciloscopio puede registrar y mostrar se seleccionan en el menú desplegable de la esquina superior derecha de cada bloque de control de canal. El tipo de parámetro establece la siguiente etiqueta entre eje (Ax) y canal (Ch).
Para trazar los puntos almacenados en la matriz TABLE del controlador, seleccione el parámetro TABLE y elija el canal que tiene los puntos primero y último configurados mediante el cuadro de diálogo de opciones avanzadas.
Si el canal no es necesario, seleccione NONE (Ninguno) en el cuadro de lista de parámetros.
Número de eje/canal: un cuadro de lista desplegable que permite la selección del eje o canal para un parámetro Motion Control o el canal para un parámetro de entrada/salida digital o analógica.
Rango Y descendente/rango Y ascendente: la escala vertical se selecciona para cada canal y se puede configurar para modo automático o manual. En modo automático, el osciloscopio calcula la escala adecuada cuando ha completado el registro antes de que muestre el seguimiento. Cuando el osciloscopio se ejecuta en activación continua, no puede seleccionar una escala vertical adecuada. El osciloscopio se debe parar y reiniciar.
En modo manual el usuario selecciona la escala más adecuada.
Desplazamiento Y: el valor de offset vertical se utiliza para mover un seguimiento verticalmente en la pantalla. Este control resulta útil cuando dos o más seguimientos son idénticos.
Reset Y: este botón borra el valor de desplazamiento Y.
Activar/desactivar barras de cursor: cuando el osciloscopio para su ejecución y se muestra un seguimiento, se pueden activar las barras de cursor. Las barras de cursor son dos barras verticales del mismo color que el seguimiento de canal. Marcan los puntos de ubicación de seguimiento máximo y mínimo. Los valores que las barras representan se muestran debajo de la visualización de osciloscopio. Las barras de cursor se activan y desactivan pulsando el botón de cursor. Las barras de cursos se pueden seleccionar y mover con el cursor del ratón.

Controles generales

Los controles generales se encuentran en la parte inferior izquierda de la pantalla del osciloscopio. Los controles son los siguientes:

Base temporal: el valor de base temporal es el valor de tiempo de cada división horizontal del osciloscopio. La base temporal se selecciona con los botones de subir/baja escala a ambos lados del cuadro de valor de base temporal. Si la base temporal es mayor que un valor predefinido, los datos se recuperan del controlador en secciones y no como un seguimiento de datos continuo.
Las secciones de datos se trazan en la pantalla tal como se reciben. El último punto es de color blanco.
Desplazamiento X: cuando el seguimiento se completa mientras la base temporal se cambia a un valor más rápido, sólo se muestra parte del seguimiento. Utilice la barra de desplazamiento X para ver el seguimiento completo.
Si el osciloscopio está configurado para registrar parámetros Motion Control y trazar datos de tabla, el número de puntos trazados en la pantalla se puede determinar mediante el parámetro motion. Los puntos de tabla adicionales que no están visibles se pueden mostrar con la barra de desplazamiento. El seguimiento del parámetro motion no se mueve.
Activador (Trigger) individual/continuo: en modo individual, el osciloscopio funciona hasta que éste se activa (trigger) y se recupera y muestra un conjunto de datos registrado por el controlador.
En el modo de activador (Trigger) continuo, el osciloscopio funciona y recupera datos del controlador cada vez que se activa el osciloscopio y se registran nuevos datos. El osciloscopio sigue funcionando hasta que se hace clic en el botón del activador por segunda vez.
Activar/detener captura de datos: cuando se hace clic en el botón del activador (trigger), el osciloscopio se activa. Si está en modo manual, el controlador comienza a registrar datos inmediatamente. Si está en modo de programa, espera hasta que encuentra un comando de activación en un programa en ejecución.
Después de haber pulsado el botón de activación (trigger), el texto del botón cambia a “Halt” (Detener). Si el osciloscopio está en modo de un impulso, después de que los datos se hayan registrado y trazado en la pantalla, el texto del botón de activación vuelve a “Trigger” (Activar), lo que indica que la operación se ha realizado. El osciloscopio se puede detener en cualquier momento mientras esté en funcionamiento y el botón de activación muestre el texto “Halt” (Detener).

Borrar configuración: la configuración de ámbito actual (el estado de todos los controles) se guarda cuando se cierra la ventana de ámbito y se recupera cuando se vuelve a abrir dicha ventana.
El botón de reset de configuración (que se encuentra en la parte derecha inferior del panel de control) efectúa un reset de todos los controles a sus valores predeterminados.
Indicador de estado: el indicador de estado se encuentre entre los botones de opciones y de reset de configuración. Este indicador cambia de color según el estado actual del osciloscopio, tal como se indica a continuación:

- Rojo: osciloscopio parado.
- Negro: el controlador espera a que el osciloscopio finalice la grabación de los datos adquiridos.
- Amarillo: los datos se están recuperando del controlador.

Establecer opciones de captura: cuando se hace clic en este botón de opción, se muestra el cuadro de diálogo de ajustes de configuración del osciloscopio.

Opciones avanzadas del osciloscopio

Información general:

Visualización de los puntos de tabla del controlador:
Si el osciloscopio está configurado para parámetros de tabla y de motionl, el número de puntos trazados en la pantalla se determina por la base temporal (y muestras por división).
Si el número de puntos que se trazará para el parámetro de tabla es mayor que el número de puntos para el parámetro de motion, los puntos de tabla adicionales no se muestran, pero se pueden ver si se desplaza el seguimiento de tabla mediante la barra de desplazamiento horizontal. El seguimiento del parámetro motion no se mueve.

Carga de datos desde el controlador en el osciloscopio
Si la base temporal general es mayor que un valor predefinido, los datos se recuperan del controlador en bloques, por lo que la pantalla que se puede ver se actualizará en secciones. El último punto trazado en la sección actual se muestra de color blanco.
Si el osciloscopio está configurado para registrar los parámetros de motion y también para trazar datos de tabla, estos datos se leen en un bloque completo y los parámetros de motion se leen continuamente o en bloques (según la base temporal).
Incluso si el osciloscopio está en modo continuo, los datos de tabla no se vuelven a leer, sólo los parámetros de motion se leen continuamente desde el controlador.

Activación/desactivación de los controles del osciloscopio
Mientras el osciloscopio está en funcionamiento, todos los controles del osciloscopio están desactivado excepto el botón de activación. Por lo tanto, si es necesario cambiar la base temporal o la escala vertical, el osciloscopio se debe detener y reiniciar.

Precisión de visualización
El controlador registra los valores de parámetro a la frecuencia de muestreo requerida en la tabla y, a continuación, pasa la información al osciloscopio. Por lo tanto, el seguimiento mostrado es preciso en lo que respecta a la base temporal seleccionada. No obstante, hay un retraso entre el momento que el controlador registra los datos y el momento en que se muestran en el osciloscopio debido al tiempo que se tarda en cargar los datos a través del vínculo de comunicaciones.

Samples per division (Muestras por división): el valor predeterminado del osciloscopio es registrar cinco puntos por división de cuadrícula horizontal (base temporal). Este valor se puede ajustar con la barra de desplazamiento adyacente. Para lograr la frecuencia de muestreo más rápida posible, reduzca el número de muestras por división de cuadrícula a 1 y aumente la escala de base temporal a su valor más rápido (1 período de servo por división de cuadrícula).

Table range used for data capture (Rango de tabla utilizado para la captura de datos): el controlador registra los datos de valor de parámetro requeridos como datos de tabla antes de cargar dichos valores en el ventana. De forma predeterminada, el valor de tabla de osciloscopio mínimo es el cero. No obstante, si está en conflicto con programas que se ejecutan en el controlador que también puedan requerir esta sección de la tabla, el valor de tabla mínimo se puede cambiar.

El valor de tabla de osciloscopio superior se actualiza posteriormente de forma automática según el número de canales en uso y el número de muestras por división de cuadrícula. Si introduce un valor de tabla inferior que provoque que el valor de tabal superior exceda el valor permitido máximo en el controlador, el osciloscopio utilizará el valor original.

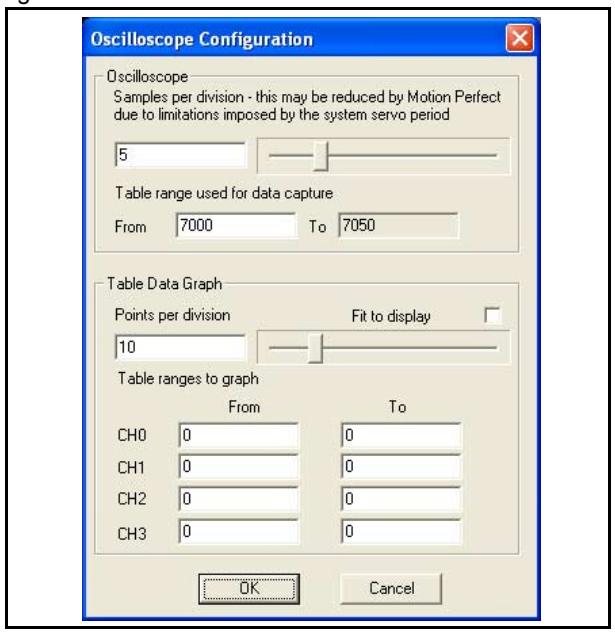
Table Data Graph (Gráfico de datos de tabla): es posible trazar los valores de tabla del controlador directamente. Los cuadros de texto de límite de tabla permiten que el usuario introduzca hasta cuatro conjuntos de índices de tabla primero/último.

Comprobaciones de parámetros

Si se están registrando las entradas analógicas, la resolución de osciloscopio más rápida (frecuencia de muestreo) es el número de canales analógicos en milisegundos (es decir, con 2 entradas analógicas se deduce que la frecuencia de muestreo más rápida es de 2 mseg.). La resolución se calcula mediante la división del valor de escala de base temporal entre el número de muestras por división de cuadrícula.

No es posible introducir valores de canal de tabla que superen el tamaño de tabla máximo del controlador ni tampoco introducir un valor de tabla de osciloscopio menor. Tampoco se permite aumentar las muestras por división de cuadrícula a un valor que provoque que el valor de tabla de osciloscopio superior exceda el valor de tabla máximo del controlador.

fig. 38



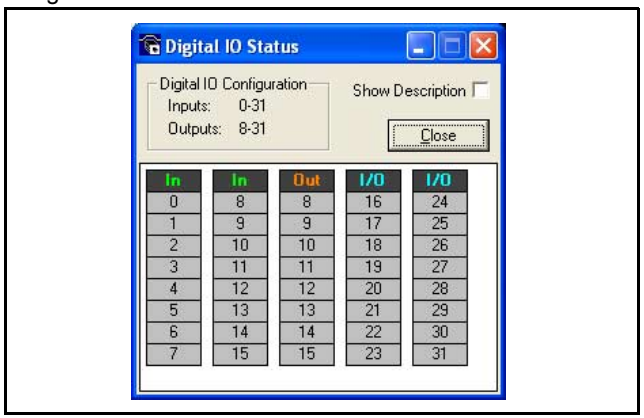
Si se aumenta el número de muestras por división de cuadrícula y posteriormente la escala de base temporal se establece a un valor más rápido que provoque una resolución que no se pueda obtener, el osciloscopio restablece automáticamente el número de muestras por división de cuadrícula.

Digital IO status (Estado de E/S digitales)

Esta ventana permite al usuario ver el estado de todos los canales de E/S y cambiar el estado de los canales de salida. Opcionalmente, permite al usuario introducir una descripción por cada línea de E/S.
Entradas digitales: muestra el número total de los canales de entrada en Trajexia.
Salidas digitales: muestra el número total de los canales de salida en Trajexia.
Visualización: la visualización está dividida en bancos que contienen 8 indicadores que representan bloques de 8 entradas o salidas:

- Banco de entradas (In)
  Representan el estado de las entradas digitales. De In(0) a In(15) son las entradas digitales incorporadas en TJ1-MC__. Las entradas digitales adicionales del sistema se asignan automáticamente a partir de In(32).
- Bancos de salidas (Out)
  Representan el estado de las salidas digitales. De OP(8) a OP(15) son las salidas digitales incorporadas en TJ1-MC__. Las salidas digitales adicionales del sistema se asignan automáticamente a partir de OP(32).
- Bancos de entradas/salidas (I/O)
  Representan las E/S virtuales que puede usar en el programa como indicadores de usuario. Al configurar una de estas salidas virtuales, también se configura la entrada virtual correspondiente. Si un indicador es de color gris, la entrada o salida correspondiente está desactivada. Si es de color (amarillo, verde, naranja, rojo, cian o magenta), la entrada o salida correspondiente está activada. Se utilizan colores distintos para representar diferentes tipos de entradas y salidas.
  Al hacer clic en un indicador que representa una salida (o entrada y salida vinculada) se obtiene como resultado el cambio de estado de dicha salida. Hacer clic en un icono que representa una entrada no tiene efecto.

fig. 39



- Algunos circuitos de salida requieren una fuente de alimentación externa. En este caso, el estado de entrada de la E/S vinculada internamente no se indica de forma correcta si la alimentación externa no está presente, incluso si una salida está activada, el estado de entrada no cambia. La misma situación se produce si una salida supera un límite actual debido a un fallo o sobrecarga.
- Show description (Mostrar descripción): al hacer clic en esta casilla de verificación se alternará entre la activación y la desactivación de las descripciones. Las descripciones se almacenan en el archivo de proyecto.

Keypad (Teclado)

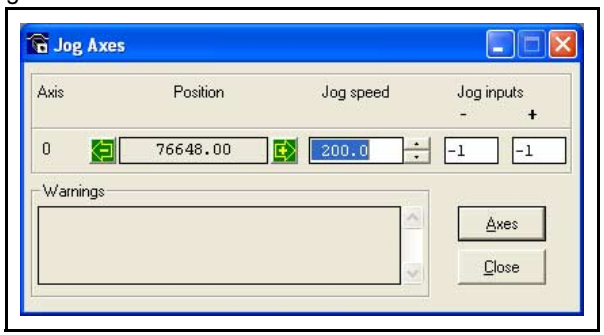
No aplicable a Trajexia.

Jog Axes (Ejes jog)

Esta ventana permite al usuario mover los ejes en el Trajexia. Esta ventana aprovecha los canales de E/S virtuales bidireccionales (de 16 a 27) en el Trajexia para establecer las entradas de marcha jog. Las entradas de marcha jog directa, inversa y rápida se identifican escribiendo en los correspondientes parámetros de eje y se espera que estén conectadas a interruptores NC. Esto significa que cuando la entrada está activada (se aplican +24 V), la función de marcha jog correspondiente está DESACTIVADA y cuando la entrada está desactivada (0 V), la función de marcha jog está ACTIVADA. Las funciones de marcha jog implementadas aquí desactivan la función de marcha jog rápida, lo que significa que la velocidad a la que se realizará la marcha jog se establece mediante el parámetro de eje JOGSPEED. Además, esta ventana limita la velocidad de marcha jog al rango de 0 a velocidad solicitada, donde la velocidad solicitada se indica mediante el parámetro de eje SPEED.
Antes de permitir que se inicie una marcha jog, la ventana de marcha jog comprueba que todos los datos establecidos en ella y en Trajexia son válidos para que se realice una marcha jog.
Jog reverse (Marcha jog inversa): Este botón iniciará una marcha jog inversa. Para ello, se debe llevar a cabo la siguiente secuencia de comprobación.

- Si es un servoeje y el servo está en OFF, se establece el mensaje de advertencia.
- Si el parámetro WatchDog está en OFF, se establece el mensaje de advertencia.

fig. 40



- Si la velocidad de marcha jog es 0, se establece el mensaje de advertencia.
- Si la velocidad de aceleración en este eje es 0, se establece el mensaje de advertencia.
- Si la velocidad de deceleración en este eje es 0, se establece el mensaje de advertencia.
- Si la entrada de marcha jog inversa está fuera del rango, se establece el mensaje de advertencia.
- Si ya se está realizando un movimiento en este eje que no sea un movimiento de marcha jog, se establece el mensaje de advertencia.

Jog forward (Marcha jog directa): Este botón inicia una marcha jog directa. Se realiza una comprobación idéntica a la marcha jog inversa. Si no hay ninguna advertencia establecida, se establece el mensaje "**Forward jog set on axis?**" (¿Marcha jog directa establecida en eje?) en la ventana de advertencias, se invalida la entrada **FAST_JOG** para este eje, la **velocidad a marcha lenta** se establece en el valor indicado en el control de velocidad jog y, finalmente, la salida **JOG_FWD** se desactiva, lo que permite la función de marcha jog directa.

Jog speed (Velocidad jog): se trata de la velocidad a la que se realizará la marcha jog. Esta ventana limita el valor al rango de cero a la velocidad solicitada para este eje, donde la velocidad solicitada se indica mediante el parámetro de eje **SPEED**. Este valor se puede cambiar escribiendo directamente en este control o utilizando el control de velocidad jog. La barra de desplazamiento sube o baja la velocidad jog en incrementos de 1 unidad por segundo.

Jog inputs (Entradas de marcha jog): son las entradas que se asociarán con las funciones de marcha jog directa/inversa. Deben estar en el rango de 8 al número total de entradas del sistema ya que los canales de entrada de 0 a 7 no son bidireccionales y, por lo tanto, el estado de la entrada no se puede establecer mediante la salida correspondiente.
Se espera que la entrada esté en ON para que la función de marcha jog esté desactivada y en OFF para que la marcha jog inversa esté activada. Para cumplir esto, cuando se establece en un número de entrada válido, la salida correspondiente se establece en ON y, a continuación, se establece el correspondiente parámetro de eje **REV_JOG**.

Axes (Ejes): muestra un cuadro de selección de eje que permite al usuario seleccionar el eje que se incluirá en la visualización de ejes de marcha jog. De forma predeterminada, se mostrarán los ejes físicos conectados al controlador.

fig. 41

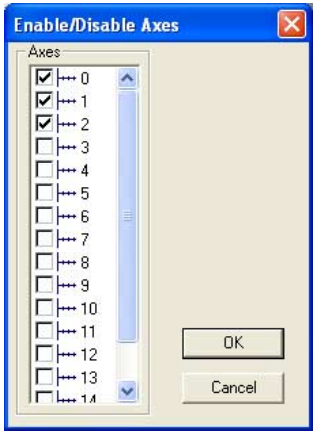
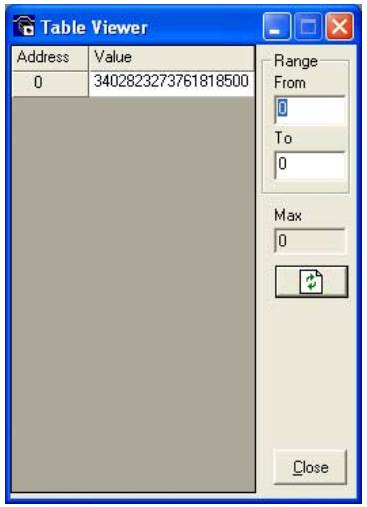


TABLE viewer (Visor de TABLE)

Las herramientas de editor de TABLE y VR son muy similares. Un rango de valores de memoria se muestra en una interfaz tipo hoja de cálculo y se puede modificar.
Para modificar un valor, haga clic en un valor existente con el ratón, escriba el nuevo valor y pulse Intro. El cambio estará activo inmediatamente y se puede efectuar mientras los programas están en ejecución.
Opciones:

- **Range** (Rango)
  Ambas herramientas disponen de la opción para establecer el inicio y el final del rango que se visualizará. En la herramienta de visualización de TABLE el valor máximo muestra el valor legible más alto (se trata del parámetro de sistema **TSIZE**).
  Si el rango de valores es mayor de lo que puede mostrar el cuadro de diálogo, la lista tendrá una barra de desplazamiento que permite ver todos los valores.
- Botón de actualización
  Esta pantalla no se actualiza automáticamente, por lo que si un valor TABLE o VR se cambia mediante el programa, no se verá el nuevo valor hasta que se actualice la pantalla.

fig. 42



Watch variables (Variables de inspección)

No implementado para Trajexia.

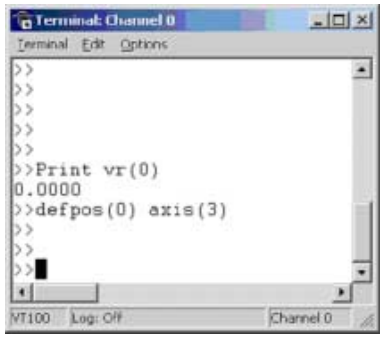
Analog inputs (Entradas analógicas)

Monitoriza el valor actual en el módulo de entradas analógicas remotas. Las entradas se añaden automáticamente al sistema a partir de AIN0, cuando se detectan uno o varios módulos AN2900.

Terminal

La ventan de terminal es un editor de textos que ofrece una conexión directa al sistema Trajexia. La mayoría de las funciones que se deben llevar a cabo durante la instalación, programación y puesta en marcha de un sistema con Trajexia se han automatizado con las opciones disponibles en los menús de Trajexia Tools.
No obstante, si se necesita comunicación directa, se puede utilizar la ventana de terminal.

fig. 43

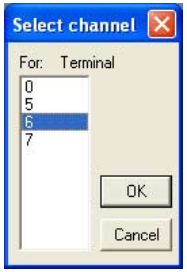


Select channel (Seleccionar canal). Cuando Trajexia Tools está conectado al controlador, la herramienta de terminal mostrará un cuadro de diálogo para seleccionar el canal de comunicaciones.

El canal 0 se utiliza para la línea de comandos de Trajexia y los canales 5, 6 y 7 se emplean para las comunicaciones con los programas que se ejecutan en Trajexia.

Seleccione el canal que desee y pulse **OK** (Aceptar) para iniciar una herramienta de terminal en el canal seleccionado. Sólo se puede conectar una herramienta de terminal (o de teclado) a un canal cada vez.

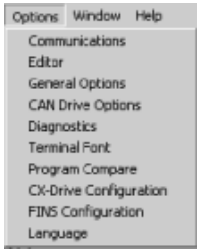
fig. 44



5.5.5 Menú Options (Opciones)

En el menú **Options** (Opciones) se establecen las opciones del sistema Trajexia:

fig. 45



Communications (Comunicaciones)

Permite establecer y ver los ajustes de comunicaciones. Los ajustes sólo se pueden cambiar offline. Las distintas opciones son:

- Serial (Serie) para otros controladores Motion Control de OMRON (C200HW-MC402-E y R88-MCW151-E).
- USB no se utiliza.
- Simulation (Simulación). Se utiliza para trabajar offline y se simula un sistema Motion Control virtual.
- Ethernet es la opción que se utiliza para Trajexia.
- PCI no se utiliza.

fig. 46

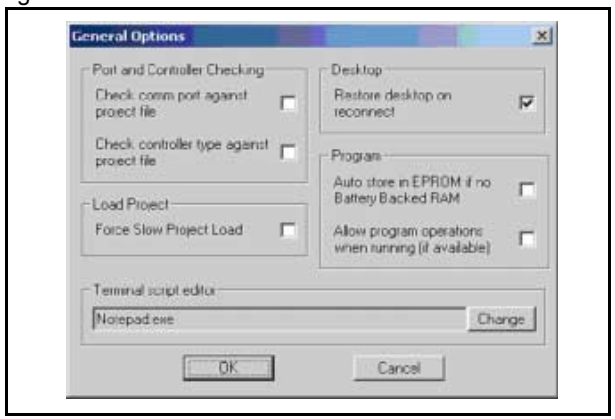
Editor

Modifica las distintas opciones del editor de textos.

General Options (Opciones generales)

Permite establecer varias opciones del sistema.

fig. 47



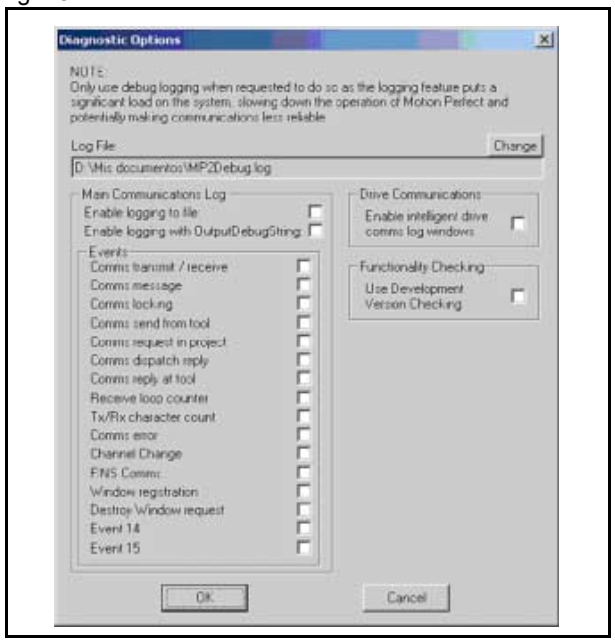
CAN Drive

No implementado en las opciones.

Diagnostics (Diagnóstico)

Permite seleccionar los eventos que se almacenarán en un archivo **.log** para su diagnóstico.

fig. 48



Terminal Font (Fuente de terminal)

Selecciona la fuente que se mostrará en la ventana de terminal. Muy útil para la puesta en marcha.

Program Compare (Comparación de programas)

Permite comparar programas.

CX-Drive Configuration (Configuración de CX-Drive)

Permite seleccionar el directorio de la base de datos de CX-Drive.

FINS Configuration (Configuración FINS)

Selecciona el puerto y el tiempo de espera de las comunicaciones FINS.

Language (Idioma)

Por el momento sólo se ha implementado el inglés.

5.5.6 Menú Window (Ventana)

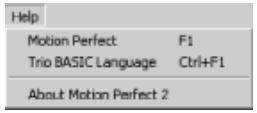
- Restore Last desktop/Restore Saved Desktop/Save Desktop/ Clear Desktop (Restaurar último escritorio/Restaurar escritorio guardado/Guardar escritorio/Borrar escritorio): son las herramientas para gestionar y configurar rápidamente el escritorio según las necesidades del usuario.
- Clear Controller Messages (Borrar mensajes de controlador): borra la ventana **Controller Messages** (Mensajes de controlador).

fig. 49

5.5.7 Menú Help (Ayuda)

- Motion Perfect: ayuda de Trajexia Tools.
- Trio BASIC Language (Lenguaje Trio BASIC): ayuda de los comandos y parámetros BASIC.
- About Motion Perfect 2 (Acerca de Motion Perfect 2): muestra la versión de Trajexia Tools.

fig. 50



6 Ejemplos y sugerencias

En este capítulo se ofrecen dos categorías de ejemplos y sugerencias:
- Procedimientos.
- Ejemplos prácticos.

6.1 Procedimientos

6.1.1 Programa de inicio

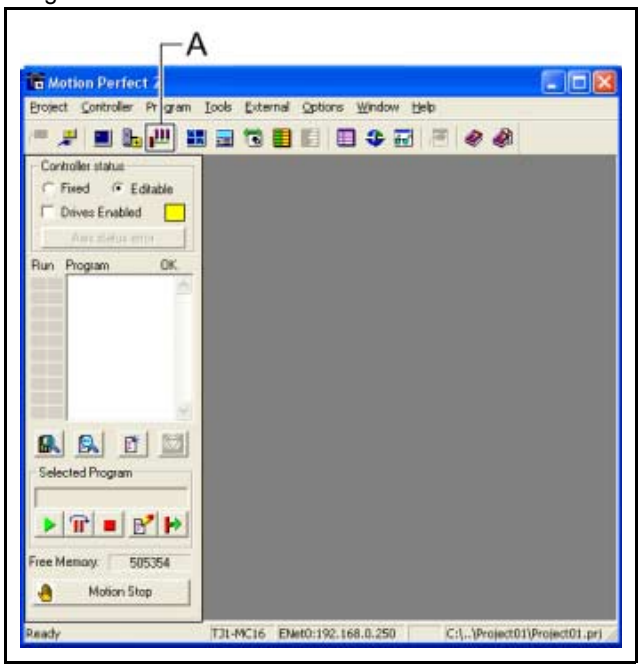
La finalidad de este programa es comparar la configuración de MECHATROLINK-II detectada con la esperada (la configuración detectada es la existente en el momento de crear el programa).
El programa STARTUP realiza estas acciones:
- Comprueba el número de nodos en el sistema.
- Comprueba que los números de nodo coincidan.
- Comprueba si todos los dispositivos están conectados y tienen alimentación.
- Si hay alguna discrepancia, el programa se para.
- Establece el valor correcto de **ATYPE** según lo seleccionado en la ventana de eje inteligente.
- Establece el modo de **marcha** o **puesta en marcha**.

Cómo usar el programa de inicio

La forma de utilizar el programa **STARTUP** se indica a continuación:

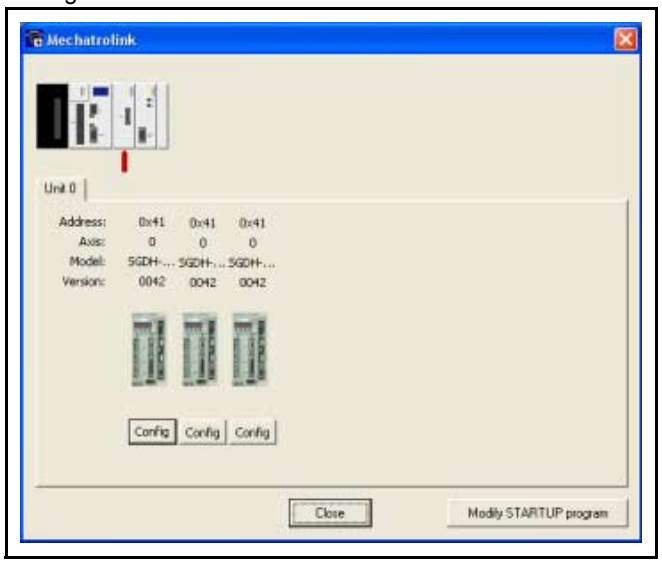
1. Haga clic en el botón **Intelligent drives** (A).

fig. 1



2. Haga clic en el botón **Modify STARTUP program** (Modificar programa STARTUP).
3. Al final de la sección creada automáticamente, coloque su propio código de aplicación. Normalmente la inicialización de variable y los parámetros de eje.
4. Al final del programa STARTUP, ejecute los programas de aplicación. Se recomienda ejecutar el programa “SHELL”, que se explica más adelante en esta sección.
5. Configure el programa STARTUP para que se ejecute en el encendido.

fig. 2



Nota
OMRON recomienda que la instrucción **RUN "programa"** se utilice al final del programa de inicio para iniciar el programa de aplicación. El programa de aplicación se inicia cuando el programa de inicio se ejecuta correctamente y sin errores.
Si configura un programa de aplicación para que se "ejecute al inicio" existe el riesgo de que la máquina arranque si hay un error en el bus MECHATROLINK-II.

Ejemplo

```
'==============================================
'LA PRIMERA PARTE DEL PROGRAMA SE GENERA
'AUTOMÁTICAMENTE POR LA VENTANA DE EJE INTELIGENTE EN
'TRAJEXIA TOOLS. CONSTA DE UNA SECUENCIA DE COMPROBACIÓN PARA
'VERIFICAR QUE LA CONFIGURACIÓN DE EJE DETECTADO ES
'LA ESPERADA.
'EN CASO AFIRMATIVO, EL PROGRAMA FINALIZA E INICIA "SHELL".
'SI NO ES ASÍ, EL PROGRAMA SE PARA Y NO SE INICIA NINGÚN OTRO
PROGRAMA.
'ESTE PROGRAMA SE DEBE CONFIGURAR PARA EJECUTAR EN EL
ENCENDIDO EN UNA
'TAREA DE PRIORIDAD BAJA (1 EN ESTE EJEMPLO)
'==============================================
'Iniciar la sección MECHATROLINK
' Comprobar dispositivos detectados
' Unidad 0
IF NOT MECHATROLINK(0,3,0) THEN
    PRINT "Error al obtener el contaje de dispositivos para
    la unidad 0"
    STOP
ELSE
    IF VR(0) <> 3 THEN
        PRINT "Contaje de dispositivos incorrecto para la
        unidad 0"
        STOP
    ENDIF
ENDIF
IF NOT MECHATROLINK(0,4,0,0) THEN
    PRINT "Error al obtener la dirección de la unidad 0,
    estación 0"
    STOP
```

```
ELSE
    IF VR(0) <> 65 THEN
       PRINT "Dirección incorrecta para unidad 0, estación 0"
       STOP
    ENDIF
ENDIF
IF NOT MECHATROLINK(0,4,1,0) THEN
    PRINT "Error al obtener la dirección de la unidad 0,
    estación 1"
    STOP
ELSE
    IF VR(0) <> 66 THEN
       PRINT "Dirección incorrecta para unidad 0, estación 1"
       STOP
    ENDIF
ENDIF
IF NOT MECHATROLINK(0,4,2,0) THEN
    PRINT "Error al obtener la dirección de la unidad 0,
    estación 2"
    STOP
ELSE
    IF VR(0) <> 67 THEN
       PRINT "Dirección incorrecta para unidad 0, estación 2"
       STOP
    ENDIF
ENDIF
' Establecer los tipos de eje
' Unidad 0
ATYPE AXIS(0)=40
ATYPE AXIS(1)=40
ATYPE AXIS(2)=40
' Configurar los servodrives en el modo de marcha
' Unidad 0
MECHATROLINK(0,20,65)
MECHATROLINK(0,20,66)
MECHATROLINK(0,20,67)
'Para la sección MECHATROLINK
'==============================================
'EL USUARIO DEB ESTABLECER ESTA SECCIÓN MANUALMENTE
'SEGÚN LA APLICACIÓN. LAS ACCIONES TÍPICAS SON
'INICIALIZACIÓN DE VARIABLES, AJUSTE DE SERVO/EJE, DENOMINACIÓN
'VARIABLES GLOBALES E INICIO DEL PROGRAMA "SHELL".
'==============================================
```

```
'Definir nombres para variables globales
GLOBAL "estado_proyecto",100
GLOBAL "estado_alarma",101
GLOBAL "accion",102
'Inicializar variables
VR(0)=0
estado_proyecto=0
estado_alarma=0
accion=0
'Iniciar programa SHELL
RUN "SHELL",2
STOP
```

6.1.2 Ajustes de ganancia

El ajuste de ganancia está relacionado con el sistema mecánico al que está conectado el motor. Existen tres conceptos principales:

- Relación de inercia
- Rigidez
- Frecuencia resonante.

Estos conceptos se describen en el manual de referencia de hardware en el capítulo Filosofía del sistema.

En esta sección se muestran valores de parámetro de ejemplo para:

- Ganancia de lazo de velocidad
- Ganancia de posición proporcional
- Ganancia de realimentación positiva de velocidad

Los valores de ejemplo para el programa y los parámetros de motion en el sistema Trajexia se ofrecen a continuación. Observe que resultan adecuados para encoders de 13 bits.

Valor de parámetro de variador	Descripción
Pn103 = 716	Relación de inercia
Pn110 = 0012	Sin autotuning

Valor de parámetro de variador	Descripción
Pn202 = 1	Numerador de relación de engranaje
Pn203 = 1	Denominador de relación de engranaje

Valores de parámetro de motion	Descripción
UNITS = 1	Trabajar en pulsos de encoder
SPEED = 200.000	Ajuste de velocidad
ACCEL = 1.000.000	Ajuste de aceleración
DECEL = 1.000.000	Ajuste de deceleración
MOVEMENT = 81.920	10 giros

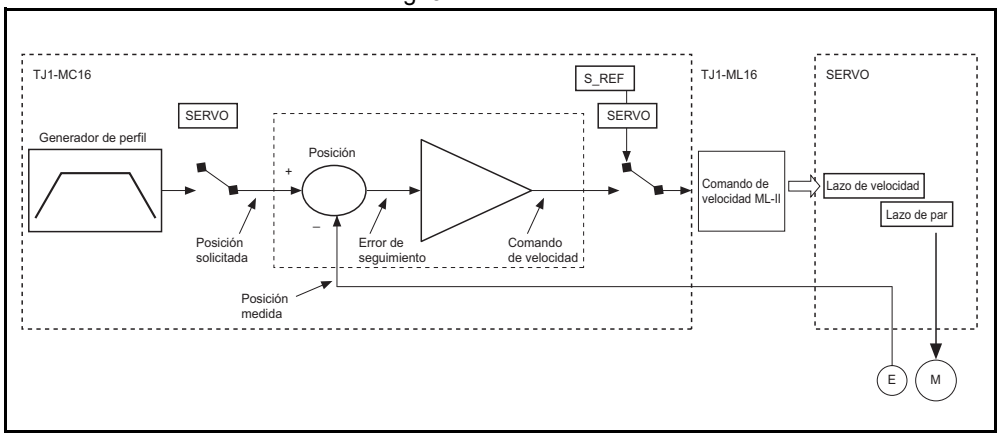
Ejemplos de modo de velocidad

En este modo, el lazo de posición es cerrado en Trajexia y el lazo de velocidad es cerrado en el servodriver. El parámetro de eje **Speed** se envía a través de la red MECHATROLINK-II al servodriver y se lee la realimentación de velocidad.

```
BASE(0)
ATYPE=41 'Modo de velocidad MECHATROLINK
SERVO=1
WDOG=1
DEFPOS(0)
lazo:
    MOVE(81920)
    WAIT IDLE
    WA(100)
    DEFPOS(0)
GOTO lazo
```

Ejemplo 1

fig. 3



Sólo la ganancia proporcional tiene un valor seleccionado, el error de seguimiento es proporcional a la velocidad. Los valores de parámetro del ejemplo son:

Valores de parámetro de Motion Control
P_Gain = 131.072
VFF_GAIN = 0
Fn001 = 4

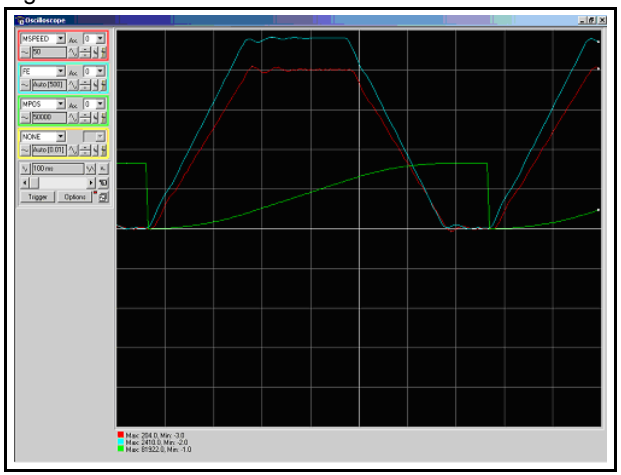
Nota:
Los colores y la escala del osciloscopio para el modo de velocidad son los siguientes:

Rojo: MSPEED (velocidad de eje medida).
Las unidades son 50 unidades/ms/división

Azul: FE (error de seguimiento).
Las unidades dependen del gráfico.

Verde: MPOS (posición de eje medida).
50.000 unidades/división

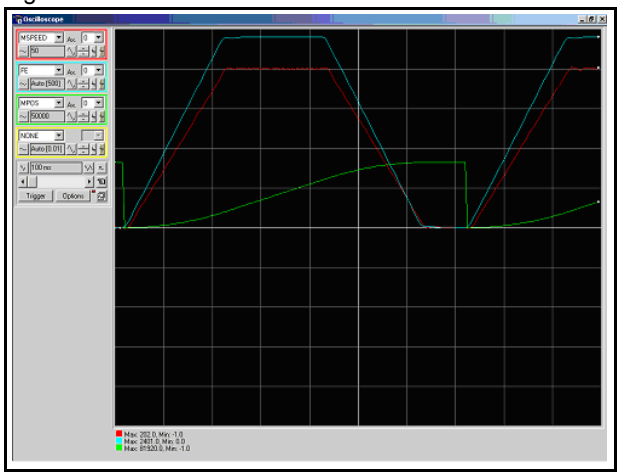
fig. 4



Ejemplo 2
El valor de rigidez se incrementa. La magnitud de error sigue siendo igual, pero la fluctuación, la estabilidad de velocidad y el sobreimpulso son mejores.
Los valores de parámetro del ejemplo son:

Valores de parámetro de Motion Control
P_Gain = 131.072
VFF_GAIN = 0
Fn001 = 6

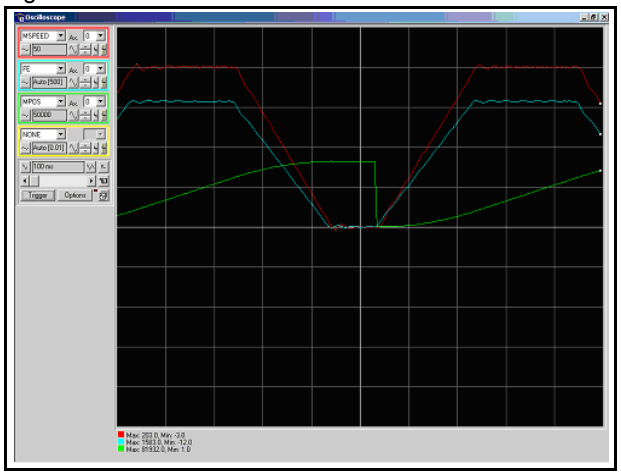
fig. 5



Ejemplo 3
El parámetro **P_GAIN** se incrementa más. El error de seguimiento se reduce proporcionalmente.
Los valores de parámetro del ejemplo son:

Valores de parámetro de Motion Control
P_Gain = 200.000
VFF_GAIN = 0
Fn001 = 6

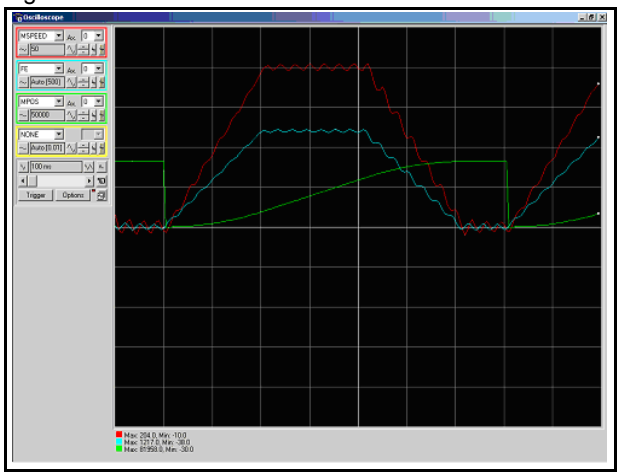
fig. 6



Ejemplo 4
El valor del parámetro **P_GAIN** duplica el valor del ejemplo 1.
El error de seguimiento es la mitad, pero hay vibraciones debido a las ganancias excesivas.
Los valores de parámetro del ejemplo son:

Valores de parámetro de Motion Control
P_Gain = 262.144
VFF_GAIN = 0
Fn001 = 6

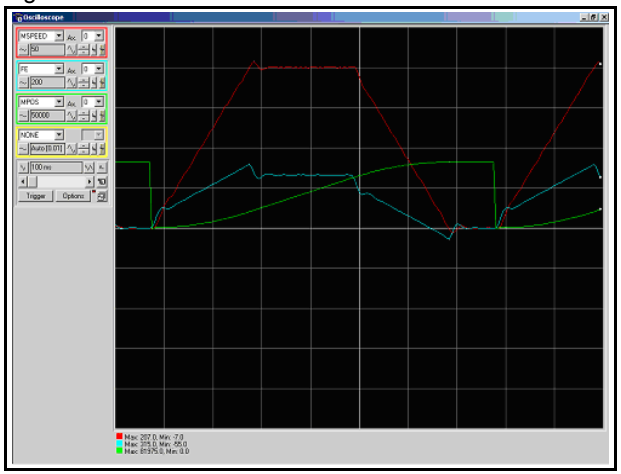
fig. 7



Ejemplo 5
El valor del parámetro **P_GAIN** se establece al valor del ejemplo 1.
Se incrementa el valor de **VFF_GAIN**. El error de seguimiento se disminuye sin reducir la estabilidad. El error de seguimiento no es proporcional a la velocidad.
Los valores de parámetro del ejemplo son:

Valores de parámetro de Motion Control
P_Gain = 131.072
VFF_GAIN = 1400000
Fn001 = 6

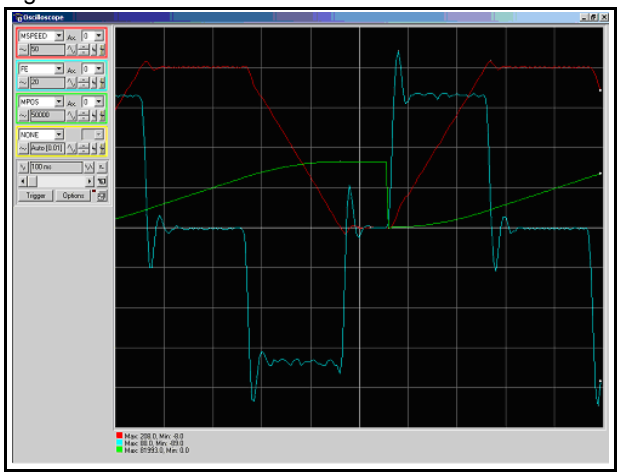
fig. 8



Ejemplo 6
Con este valor de VFF_GAIN, el error de seguimiento es proporcional a la aceleración y menor que sólo con la ganancia proporcional (la escala es 20 unidades/división). El error de seguimiento se acerca a cero durante la velocidad constante.
El efecto negativo de este conjunto de valores es el sobreimpulso y la falta de impulso cuando cambia la aceleración; esto se puede reducir, pero no eliminar, mediante el aumento de la ganancia de lazo de velocidad si el sistema mecánico puede aceptar una ganancia alta.
Los valores de parámetro del ejemplo son:

Valores de parámetro de Motion Control
P_Gain = 131.072
VFF_GAIN = 1.573.500
Fn001 = 6

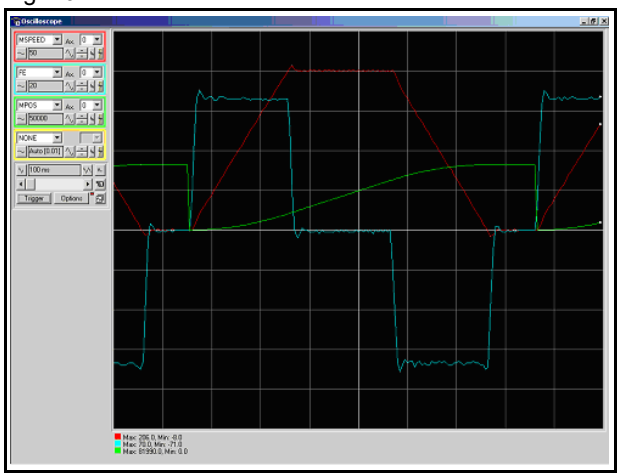
fig. 9



Ejemplo 7
El valor de la rigidez aumenta de 6 a 8. El sobreimpulso y la falta de impulso son menores, pero el motor tiene más vibraciones.
Los valores de parámetro del ejemplo son:

Valores de parámetro de Motion Control
P_Gain = 131.072
VFF_GAIN = 1573500
Fn001 = 8

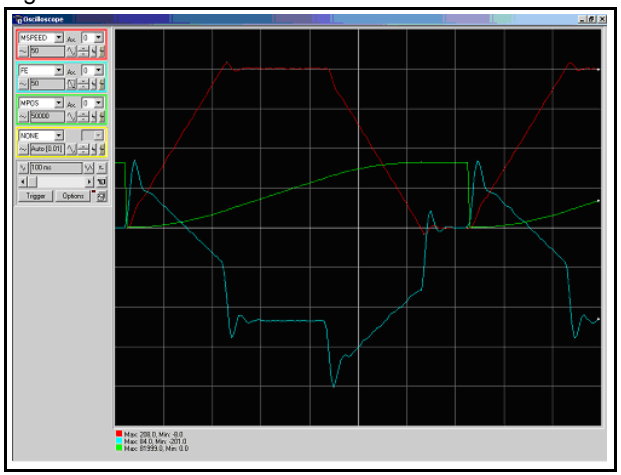
fig. 10



Ejemplo 8
A diferencia de P_GAIN, donde es mejor cuanto mayor es el valor (el límite se establece cuando el sistema mecánico comienza a vibrar), para VFF_GAIN hay un valor óptimo (el de la prueba 6). Los valores mayores que éste tienen un error proporcional a la velocidad/aceleración con signo distinto. La corrección necesaria es demasiado grande.
Los valores de parámetro del ejemplo son:

Valores de parámetro de Motion Control
P_Gain = 131.072
VFF_GAIN = 1.650.000
Fn001 = 6

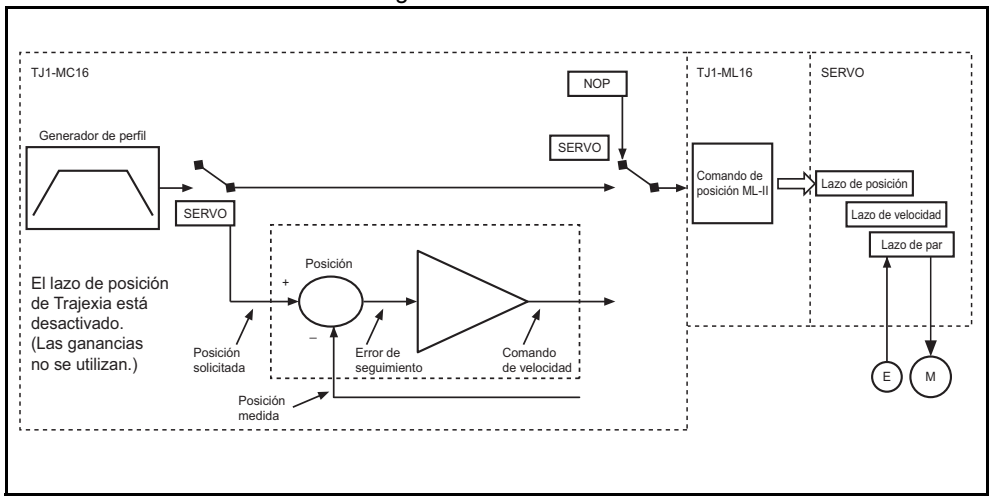
fig. 11



Ejemplos de modo de posición

En este modo, los lazos de posición y velocidad están cerrados en el servodriver. La TJ1-ML__ envía el comando de posición a través de la red MECHATROLINK-II al servodriver y lee la realimentación de posición.
Observe que este sistema no tiene retraso de muestra en comparación con el lazo de posición en el servodriver, la posición solicitada en ciclo “n” con la posición medida en ciclo “n”.
El Trajexia, para la gestión interna, sigue utilizando su propio lazo de posición, por lo que el error de seguimiento que se lee en el parámetro de eje en Trajexia no es el real del servodriver. Para leer el error de seguimiento correcto, utilice DRIVE_MONITOR.
Ajuste la rigidez del servo, la ganancia de lazo de velocidad y la ganancia de lazo de posición al mismo tiempo utilizando sólo la ganancia de posición proporcional. Los resultados son similares a los del modo de velocidad de MECHATROLINK-II con estas ventajas:

fig. 12



- El ajuste es más simple, sólo se debe establecer la rigidez (Fn001) y, si es necesario, la ganancia de realimentación directa (Pn109).
- El lazo de posición en el servo es más rápido (250µs) que en Trajexia y se activa junto con el lazo de velocidad.
- No hay retraso de tiempo de muestreo entre la “posición objetivo” y la “posición medida”.

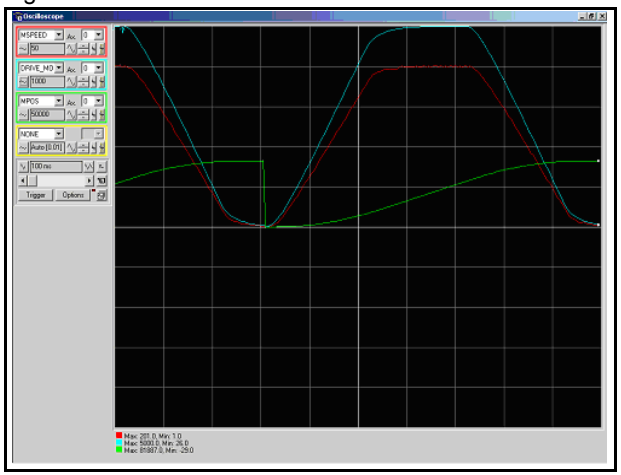
Para realizar un ajuste fino, los distintos parámetros de ganancia se pueden cambiar individualmente.

```
BASE(0)
ATYPE=41 'Modo de posición MECHATROLINK
SERVO=1
DRIVE_CONTROL=2 'Para monitorizar el error de seguimiento en
                'DRIVE_MONITOR
WDOG=1
DEFPOS(0)
lazo:
    MOVE(81920)
    WAIT IDLE
    WA(100)
    DEFPOS(0)
GOTO lazo
```

Ejemplo 1

El error de seguimiento es proporcional a la velocidad. Hay un “perfil suave” debido al ajustes de baja rigidez (ganancia baja).

fig. 13



Nota:
Los colores y la escala del osciloscopio para el modo de posición son los siguientes:

Rojo: MSPEED (velocidad de eje medida).
Las unidades son 50 unidades/ms/división

Azul: DRIVE_MONITOR (establecer como error de seguimiento en el servodriver).
Las unidades dependen del gráfico.

Verde: MPOS (posición de eje medida).
50.000 unidades/división

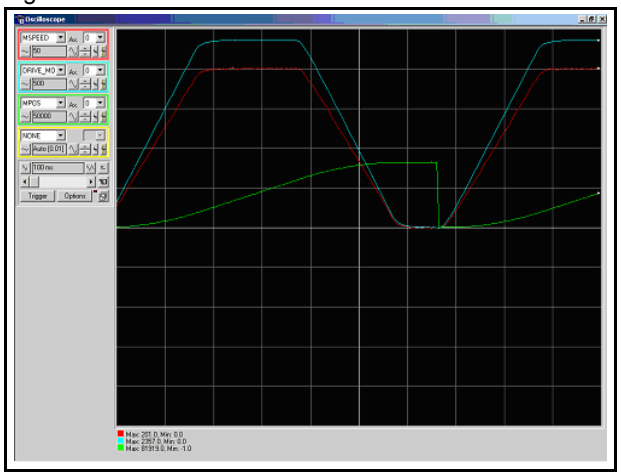
Los valores de parámetro del ejemplo son:

Valores de parámetro de Motion Control
Fn001 = 4
Pn109 = 0

Ejemplo 2
El error de seguimiento se reduce a medida que aumenta la rigidez.
Los valores de parámetro del ejemplo son:

Valores de parámetro de Motion Control
Fn001 = 6
Pn109 = 0

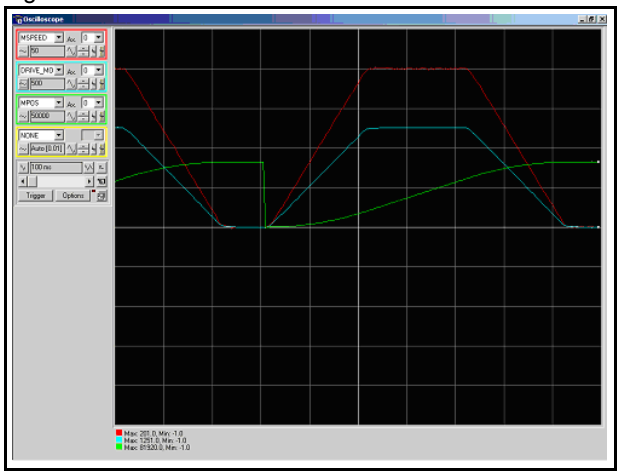
fig. 14



Ejemplo 3
Con una ganancia alta, el motor empieza a vibrar pero el perfil es más estable que en el modo de velocidad MECHATROLINK-II.
Los valores de parámetro del ejemplo son:

Valores de parámetro de Motion Control
Fn001 = 8
Pn109 = 0

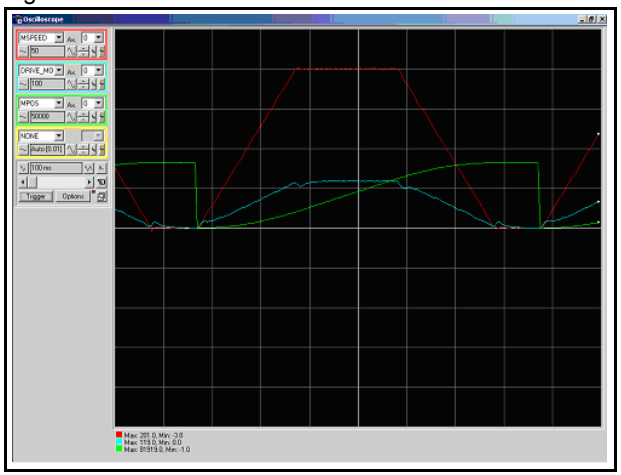
fig. 15



Ejemplo 4
El efecto de la ganancia de realimentación positiva es que el error de seguimiento se reduce y el efecto es proporcional a la aceleración.
Los valores de parámetro del ejemplo son:

Valores de parámetro de Motion Control
Fn001 = 6
Pn109 = 95

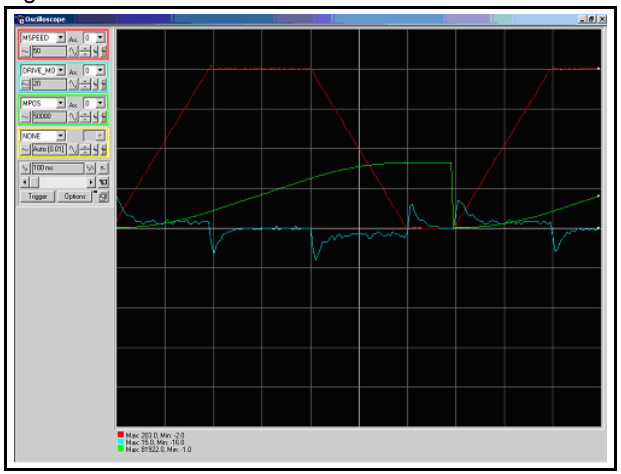
fig. 16



Ejemplo 5
Con la realimentación positiva establecida al 100%, el error de seguimiento es muy pequeño y proporcional a la aceleración.
El valor óptimo de la corrección del 100% es el valor máximo que se puede establecer. El valor de parámetro de Pn109 es más sencillo de establecer que el valor de parámetro de VFF_GAIN.
Los valores de parámetro del ejemplo son:

Valores de parámetro de Motion Control
Fn001 = 6
Pn109 = 100

fig. 17



6.1.3 Ajuste del parámetro de eje UNITS y la relación de engranaje

Al controlar el eje mecánico con la Trajexia TJ1-MC__, un servodriver y un servomotor, las únicas unidades de medida que entiende el hardware son los contajes de encoder. Todos los comandos enviados al driver para mover un eje se expresan en pulsos de encoder. Toda la información de realimentación sobre las posiciones de eje también se expresa en pulsos de encoder.
Al escribir programas en BASIC para lograr movimientos o una secuencia de movimientos, el usuario puede preferir trabajar en unidades definidas por el usuario, como milímetros, centímetros, metros, grados de ángulo, “producto”, “rotación”, “estaciones”, etc.
El parámetro de eje **UNITS** contiene el factor de conversión entre contajes de encoder y unidades definidas por el usuario. Todos los parámetros de eje relacionados con Motion Control y los argumentos de comandos de eje que determinan la cantidad de movimiento se expresan en estas unidades de usuario. Este parámetro permite que el usuario defina las unidades más adecuadas con la que desee trabajar. Por ejemplo, para una pieza móvil que realice un movimiento lineal, se puede preferir mm o una fracción de mm. Para una pieza móvil que realiza un movimiento de rotación, se puede preferir un grado de ángulo o su fracción. Para obtener más información sobre el parámetro de eje **UNITS**, consulte la sección 3.2.270.
No obstante, el usuario debe tener en cuenta que no sólo el parámetro de eje **UNITS** interviene en la conversión entre pulsos de encoder y unidades definidas por el usuario. Determinados parámetros de servodriver y algunas características del sistema mecánico también son importantes. En las siguientes secciones se describen los parámetros de servodriver que son importantes para esta conversión. También ofrecemos ejemplos de cómo establecer estos parámetros y el parámetro de eje **UNITS**, teniendo en cuenta las características del sistema mecánico.

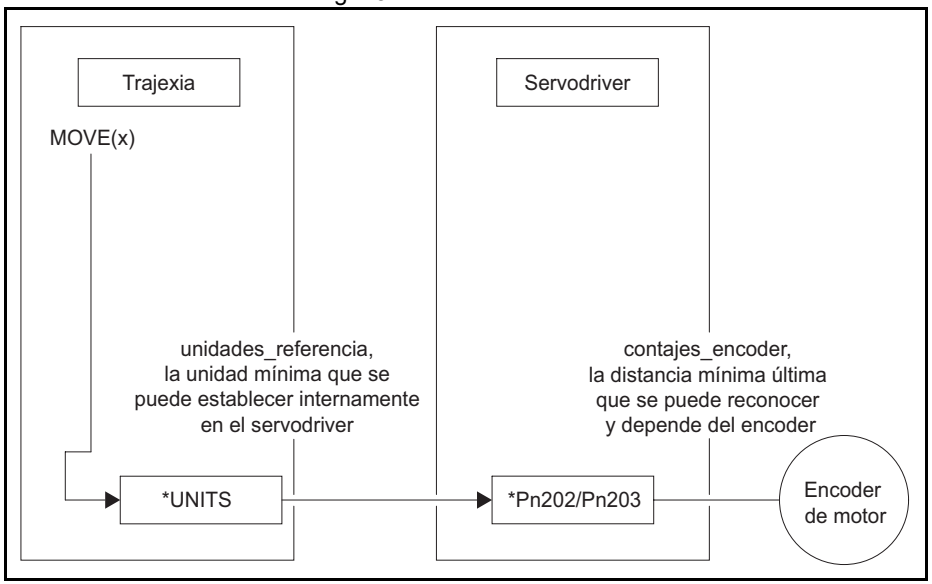
Conversión entre pulsos de encoder y unidades definidas por el usuario

Dos parámetros muy importantes del servodriver Sigma-II para la conversión de pulsos de encoder en unidades de usuario son Pn202 y Pn203. Si se utiliza un servomotor con un encoder absoluto, también es necesario ajustar el parámetro Pn205.
El parámetro Pn202 es el denominador de relación de engranaje electrónico (G1). El parámetro Pn203 es el numerador de relación de engranaje electrónico (G2). El servomotor gira utilizando el valor de la señal de comando de posición enviada por la TJ1-MC__ multiplicado por el engranaje electrónico (Pn202, Pn203). En el lado de salida (servomotor), la señal se expresa en número de pulsos de encoder. Para obtener más información sobre los parámetros Pn202 y Pn203 de servodriver, consulte el manual del servodriver Sigma-II.
El parámetro de eje UNITS expresa de forma efectiva la relación entre las unidades de usuario que el usuario desea utilizar en el programa y la posición enviada al servodriver a través del bus MECHATROLINK-II. Si se tiene en cuenta el ajuste de engranaje electrónico, la ecuación que expresa la relación entre unidades de usuario, el parámetro **UNITS**, los parámetros Pn202 y Pn203, los pulsos de encoder y la unidad de medición mecánica es:

$$\frac{Pn202}{Pn203} \cdot UNITS = \frac{y \cdot contajes_encoder}{x \cdot unidades_usuario}$$

donde *y* es el número de pulsos de encoder y *x* es la cantidad en unidades de usuario.

fig. 18



Ejemplo 1

El sistema mecánico consta de una mesa rotativa simple.
Se utiliza un servomotor con encoder incremental de 13 bits.
La relación del engranaje es 1:10.
Las unidades de usuario deseadas son grados de ángulo.
Este sistema se puede describir con las ecuaciones siguientes:

$$1 \cdot revolución_motor = 2^{13} \cdot contajes_encoder$$

$$10 \cdot revolución_motor = 1 \cdot ciclo_máquina$$

$$1 \cdot ciclo_máquina = 360^{o}$$

La combinación de estas ecuaciones da como resultado:

$$\frac{Pn202}{Pn203} \cdot UNITS = \frac{2^{13} \cdot contajes_encoder}{1 \cdot revolución_motor} \frac{10 \cdot revolución_motor}{1 \cdot revolución_máquina} \frac{1 \cdot revolución_máquina}{360^{o}} =$$

$$\frac{2^{13} \cdot 10}{360} \frac{contajes_encoder}{grado}$$

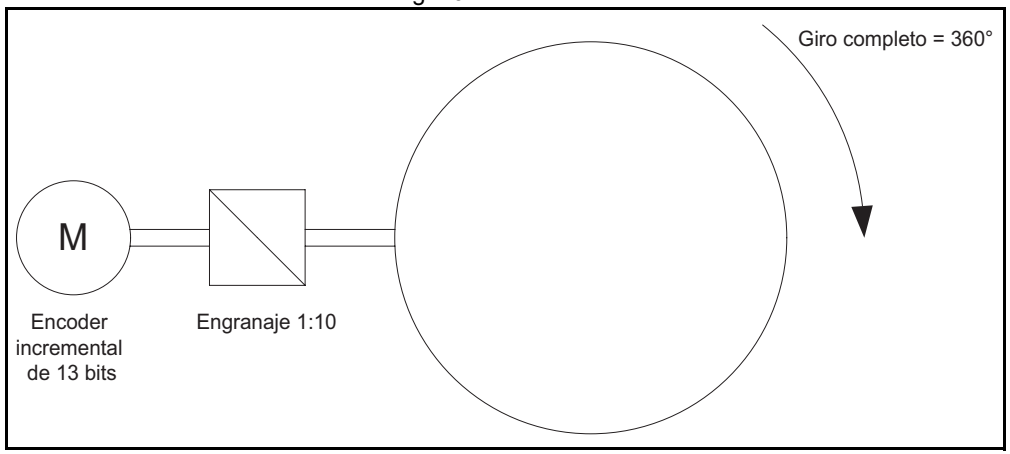
Y por lo tanto:

$$\frac{Pn202}{Pn203} \cdot UNITS = \frac{2^{13} \cdot 10}{360}$$

A partir de esta ecuación podemos derivar los valores de Pn202, Pn203 y **UNITS**, según las siguientes restricciones y recomendaciones:

1. Pn202 y Pn203 son enteros.
2. UNITS no debe tener un número infinito de decimales. Esto puede originar errores de redondeo que den lugar a pequeños errores de posición que se acumulen en grandes errores de posición acumulados.
3. Por motivos de estabilidad, es necesario evitar situaciones donde Pn202/Pn203 sea menor que 0,01 o mayor que 100.

fig. 19



La última ecuación se puede reescribir así:

$$UNITS \cdot \frac{Pn202}{Pn203} = 2^{13} \frac{10}{360}$$

Una solución a esta ecuación es:

$$UNITS = 2^{13} = 8192$$
$$Pn202 = 10$$
$$Pn203 = 360$$

Si se tiene en cuenta la tercera recomendación de la lista anterior (evitar situaciones donde Pn202/Pn203 sea menor que 0,01 o mayor que 100), se puede reescribir la última ecuación así:

$$UNITS \cdot \frac{Pn202}{Pn203} = 2^{13} \frac{10}{360} = 2^{8} \frac{2^{5}}{36} = 2^{8} \frac{32}{36}$$

Esto nos da la solución:

$$UNITS = 2^{8} = 256$$
$$Pn202 = 32$$
$$Pn203 = 36$$

Con estos valores, el comando **MOVE(28)** gira la mesa 28 grados en dirección positiva.

Ajuste de encoder absoluto

El encoder absoluto mantiene la posición de motor actual, incluso si no hay alimentación. El encoder absoluto ofrece la posición en un giro (es decir, una fracción desde 0 y excluyendo 1) y tiene un contador multigiro. Puede establecer el comportamiento multigiro del encoder absoluto con el parámetro Pn205 del servodriver Sigma-II. Este parámetro ajusta el número máximo de giros que el contador cuenta antes de que se produzca un desbordamiento.

Para obtener más información sobre el parámetro Pn205 de servodriver, consulte el manual del servodriver Sigma-II. Si se tiene en cuenta este valor de parámetro, el valor de posición máximo que el encoder puede señalizar es:

$$\text{valor_contaje_encoder_máximo} = (\text{Pn205} + 1)\ \ \text{contajes_encoder} - 1$$

lo que convierte Pn205 en giros completos, además de la posición en un turno (la fracción desde 0 y excluyendo 1). Cuando se establece la conexión MECHATROLINK con el variador, la posición de encoder absoluto se lee del variador y el valor se escribe en **MPOS** (después de la conversión: **UNITS** x Pn202/Pn203). Cuando el sistema mecánico tiene una distancia de recorrido limitada para moverse, como en un tornillo móvil, el valor del parámetro Pn205 se debe establecer lo suficientemente grande como para tener un desbordamiento del contador fuera de la posición efectiva. Esto se denomina eje limitado o eje finito. Un ejemplo típico de un eje limitado es un tornillo móvil, tal como se muestra en fig. 24. Cuando el sistema mecánico se mueve siempre en la misma dirección, alcanza el desbordamiento del contador multigiro. En este caso, el valor de Pn205 debe garantizar que el desbordamiento se produce siempre en la misma posición con respecto a la máquina. Esto se denomina eje ilimitado y un ejemplo típico es una tornamesa mostrada en fig. 20. Se puede lograr con la ecuación siguiente: el valor menor de *m* como:

$$n \cdot \text{ciclos_máquina} = m \cdot \text{revolución_motor}$$

Como *n* y *m* son enteros: Pn205 = *m* – 1. Este ajuste se explica en el siguiente ejemplo.

Ejemplo 2

El sistema mecánico consta de una mesa rotativa simple mostrada en la figura. Se utiliza un servomotor con encoder absoluto de 16 bits. La relación del engranaje es 1:10. Las unidades de usuario deseadas son grados de ángulo. La tabla rotativa está dividida en seis secciones de 60 grados cada una. Por lo tanto, el ciclo de máquina es de 60 grados.
Cuando se aplica la última ecuación a lo anterior, se obtiene:

$$10 \cdot \text{revolución_motor} = 1 \cdot \text{revolución_máquina} = 6 \cdot \text{ciclo_máquina}$$

La simplificación de esta ecuación da:

$$5 \cdot \text{revolución_motor} = 3 \cdot \text{ciclo_máquina}$$

Esto da como resultado:

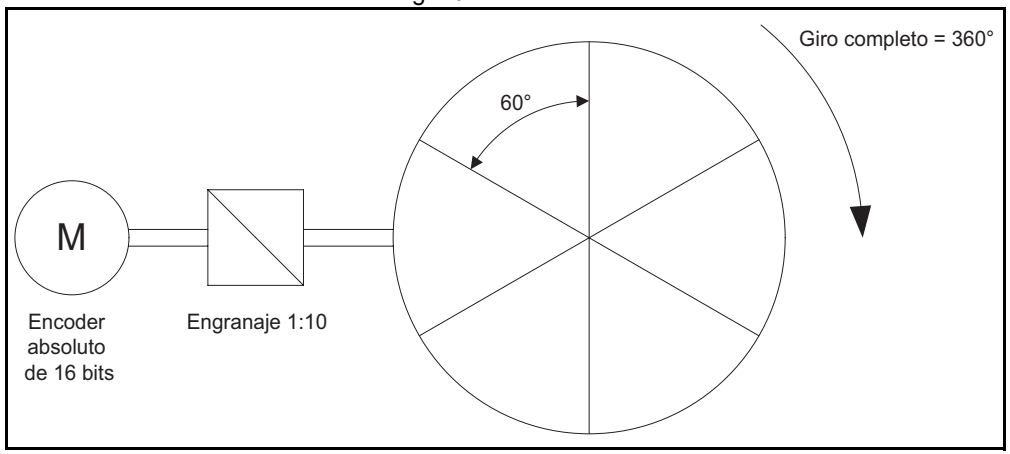
$$Pn205 = 5 - 1 = 4$$

Calculamos los parámetros tal como lo hemos hecho en el ejemplo 1.
Esto da:

$$UNITS = 2^{11} = 2048$$
$$Pn202 = 32$$
$$Pn203 = 36$$

Para garantizar el desbordamiento correcto en Trajexia y en el servodriver, debemos establecer dos parámetros de eje adicionales: **REP_DIST** = 60 y **REP_OPTION** = 1.
Con estos ajustes, el comando **MOVE(35)** gira la mesa 35 grados en dirección positiva. El rango de valores posibles para **MPOS** y **DPOS** va de 0 a 60 grados.

fig. 20



Debe inicializar el encoder absoluto antes de utilizarlo por primera vez, cuando la batería se pierde durante el apagado y cuando se cambia el ajuste de límite multigiro en el parámetro Pn205. La inicialización se puede realizar en el display del servodriver o con la herramienta de programación. Para obtener más detalles sobre la inicialización del encoder absoluto, consulte el manual del servodriver Sigma-II.

Se puede restablecer el contador multigiro, pero no la posición en un turno (la fracción desde 0 y excluyendo 1). Para ajustar el offset cero, utilice el parámetro Pn808. Para obtener más detalles, consulte el manual de la unidad de interfaz MECHATROLINK-II NS115.

En el encendido, la posición del encoder absoluto se lee desde el motor y se escribe en **MPOS** mediante la conversión siguiente:

- Para **MPOS**:

$$\text{MPOS_absoluto} = \text{encoder_posición_absoluta} \cdot \frac{1}{\text{UNITS}} \cdot \frac{\text{Pn203}}{\text{Pn202}}$$

- Esto es correcto si:

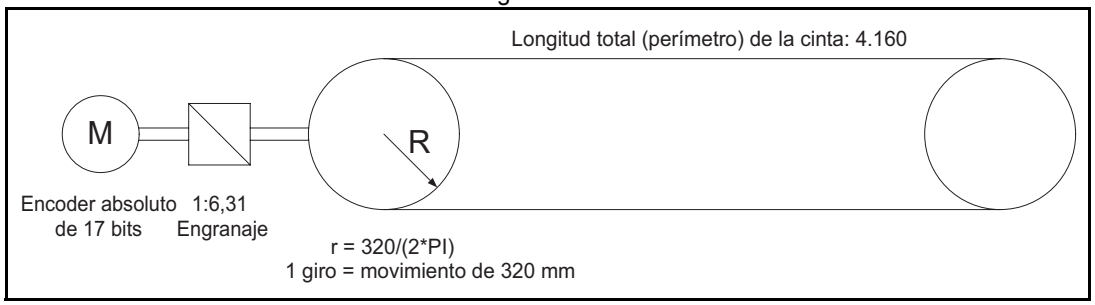
$$(\text{Pn205} + 1) \cdot \frac{\text{Pn203}}{\text{Pn202}} \cdot \text{contajes_encoder} < 5000000$$

- Si este valor es mayor que 5.000.000, **MPOS** puede tener valores incorrectos en el inicio. Para evitar este problema, añada el código de programa **DEFPOS = ENCODER/UNITS** después de todas las inicializaciones de **UNITS**.

Ejemplo 3

El sistema mecánico utiliza un servomotor con un encoder absoluto de 17 bits. La relación mecánica del engranaje es 1:6.31. Un giro de la polea mueve la pieza móvil de la correa 320 mm. La longitud total de la correa y, por lo tanto, el rango de movilidad total de la pieza de movimiento es 4.160 mm.
Las unidades de medición mecánica deben ser en mm. Esto significa que todos los parámetros y comandos de eje enviados a Trajexia se expresan en mm. Utilizando el mismo procedimiento que en el ejemplo 1, la ecuación que expresa la relación entre unidades de usuario y contajes de encoder es:

fig. 21



$$\frac{Pn202}{Pn203}\,UNITS = \frac{2^{17}\cdot contajes_encoder}{1\cdot revolución_motor}\,\frac{6{,}31\cdot revolución_motor}{1\cdot revolución_polea}\,\frac{1\cdot revolución_polea}{320\ mm} =$$

$$\frac{2^{17}\cdot 6{,}31}{320}\,\frac{contajes_encoder}{Mm}$$

Por lo tanto:

$$\frac{Pn202}{Pn203}\,UNITS = \frac{2^{17}\cdot 6{,}31}{320} = \frac{2^{17}}{2^{5}}\,\frac{631}{1000} = 2^{12}\,\frac{631}{8{,}125} = 2^{12}\,\frac{631}{2^{3}\cdot 125} = 2^{9}\,\frac{631}{125}$$

Una solución es:

$$UNITS = 2^{9} = 512$$
$$Pn202 = 631$$
$$Pn203 = 125$$

Observe que no hemos utilizado el radio de polea en el cálculo. De este modo se evita el uso de π, que no se puede expresar como un número fraccionario. En poleas dentadas, se utiliza habitualmente el número de dientes y los mm por diente.
El cálculo del ajuste de límite multigiro es:

$$m \cdot \text{revolución_motor} = n \cdot \text{ciclo_máquina}$$

$$m \cdot \text{revolución_motor} = n \cdot \text{ciclo_máquina} \frac{4.160 \cdot \text{revolución_polea}}{320 \cdot \text{ciclo_máquina}} = n \cdot 13 \cdot \text{revolución_polea}$$

$$= n \cdot 13 \frac{6 \cdot 31 \text{ revolución_motor}}{1 \text{ revolución_polea}} = n \cdot 82{,}03 \cdot \text{revolución_polea}$$

$$m = n \cdot 82{,}03$$

El entero menor *m* para el que esta ecuación es válido es 8.203. Esto da como resultado Pn205 = 8.202.
Además, para limitar el rango de unidades de movimiento al rango móvil de la pieza de movimiento, se deben establecer los siguientes parámetros de eje: **REP_DIST = 4260** y **REP_OPTION = 1**. Con estos ajustes, al ejecutar **MOVE(38)** se mueve la pieza móvil 38 mm en dirección de avance. El rango de valores posibles para **MPOS** y **DPOS** va de 0 a 4.160 mm.

Ejemplo 4

El sistema mecánico utiliza un servomotor con un encoder absoluto de 17 bits. La relación mecánica del engranaje es 1:12,24. Las unidades de medición mecánica deben ser décimas de un grado de ángulo. Por lo tanto, la distancia de repetición total para el giro completo de la pieza móvil es 3.600 décima de un grado de ángulo. Con el mismo procedimiento que en el ejemplo 1, tenemos:

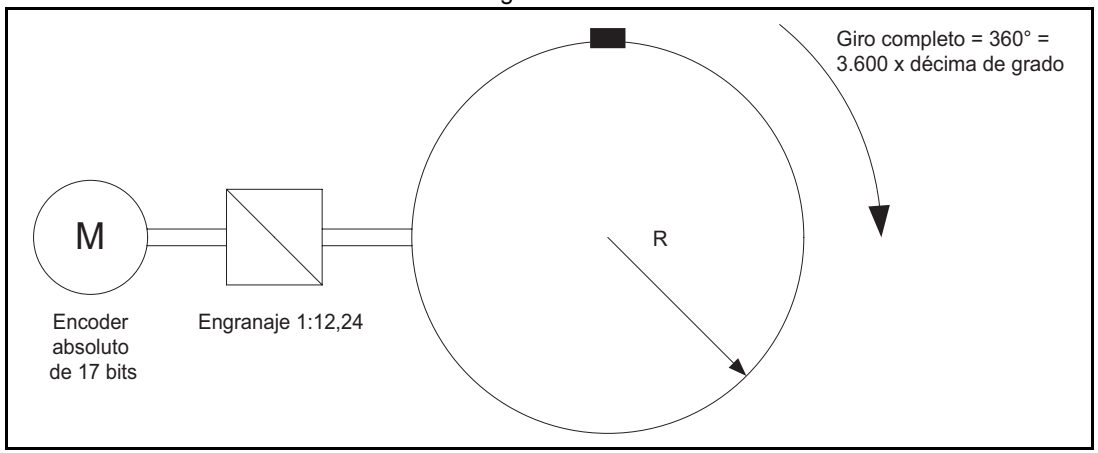
$$\frac{\text{Pn202}}{\text{Pn203}} \text{ UNITS} = \frac{2^{17} \cdot \text{contajes_encoder}}{1 \cdot \text{revolución_motor}} \frac{12{,}24 \cdot \text{revolución_motor}}{1 \cdot \text{revolución_máquina}} \frac{1 \cdot \text{revolución_polea}}{3.600 \text{ décimas de grado}} =$$

$$= \frac{2^{17} \cdot 12{,}24}{3.600} \frac{\text{contajes_encoder}}{\text{décima de grado}}$$

Por lo tanto:

$$\text{UNITS} = \frac{\text{Pn202}}{\text{Pn203}} = 2^{17} \frac{1.224}{360.000}$$

fig. 22



Una solución es:

$$UNITS = 2^{17} = 131.072$$
$$Pn202 = 1.224$$
$$Pn203 = 360.000$$

Como el máximo común divisor de Pn202 y Pn203 debe ser 1, obtenemos: Pn202 = 17 y Pn203 = 500. Por lo tanto, los parámetros son:

UNITS = 131.072
Pn202 = 17
Pn203 = 500
Pn205 = 16
REP_DIST = 3.600
REP_OPTION = 1

Para calcular el ajuste de límite multigiro Pn205, tenemos:

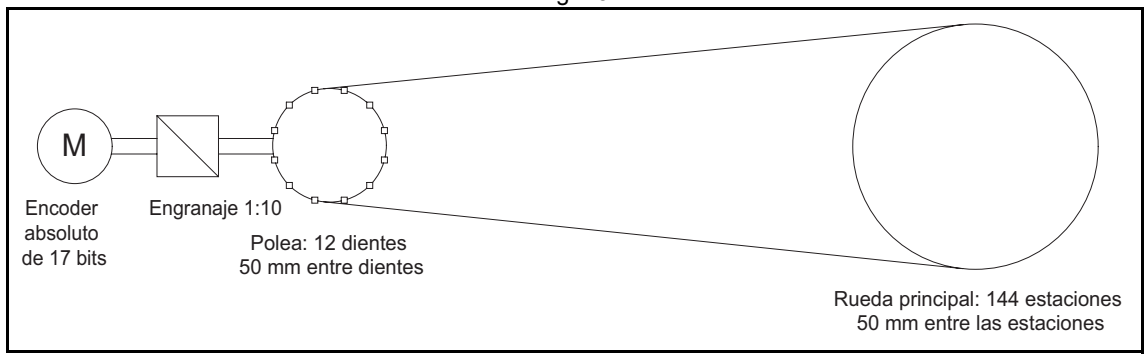
$$m \cdot revolución_motor = n \cdot ciclo_máquina = n \cdot 12{,}24 \cdot revolución_motor$$

La solución evidente es: *n* = 100 y *m* = 1.224. O, cuando simplificamos los factores: *n* = 25 y *m* = 306. Por lo tanto: Pn205 = m – 1 = 305. Con estos ajustes, al ejecutar **MOVE(180)** se mueve la pieza móvil 180 décimas de un grado de ángulo o 18 grados de ángulo en dirección de avance.

Ejemplo 5

El sistema mecánico utiliza un servomotor con un encoder absoluto de 17 bits. La relación mecánica del engranaje es 1:10. La polea tiene 12 dientes y cada dos hay una separación de 50 mm. Un giro completo de la polea equivale a 144 estaciones en la rueda principal. La distancia entre dos estaciones es de 50 mm. Las unidades de medición mecánica deben ser mm. La distancia de repetición total debe ser la distancia entre dos estaciones, 50 mm.
Con el mismo procedimiento que en el ejemplo 1, tenemos:

fig. 23



$$\frac{Pn202}{Pn203}\ \text{UNITS} =$$

$$\frac{2^{17}\cdot \text{contajes_encoder}}{1\cdot \text{revolución_motor}}\cdot\frac{10\cdot \text{revolución_motor}}{1\cdot \text{revolución_polea}}\cdot\frac{1\cdot \text{revolución_polea}}{12\cdot \text{estación}}\ \frac{1\cdot \text{estación}}{50\ \text{mm}} =$$

$$= \frac{2^{17}\cdot 10}{12\cdot 50}\ \frac{\text{contajes_encoder}}{\text{Mm}}$$

Por lo tanto, si utilizamos el sistema mecánico para establecer la relación de engranaje electrónico, tenemos:

$$\text{UNITS}\ \frac{Pn202}{Pn203} = \frac{2^{17}}{50}\ \frac{10}{12}$$

Una solución posible es:

$$\text{UNITS} = \frac{2^{17}}{50}$$

Pn202 = 5
Pn203 = 6
Pn205 = 4

Como $2^{17}/50$ es un número con decimales infinitos, podemos optar por lo siguiente:

$$\text{UNITS}\ \frac{Pn202}{Pn203} = 2^{17}\ \frac{10}{50\cdot 12} = 2^{17}\ \frac{10}{600} = 2^{17}\ \frac{1}{60} = 2^{17}\ \frac{1}{2^{2}\cdot 15} = 2^{15}\ \frac{1}{15}$$

Por lo tanto, los parámetros son:

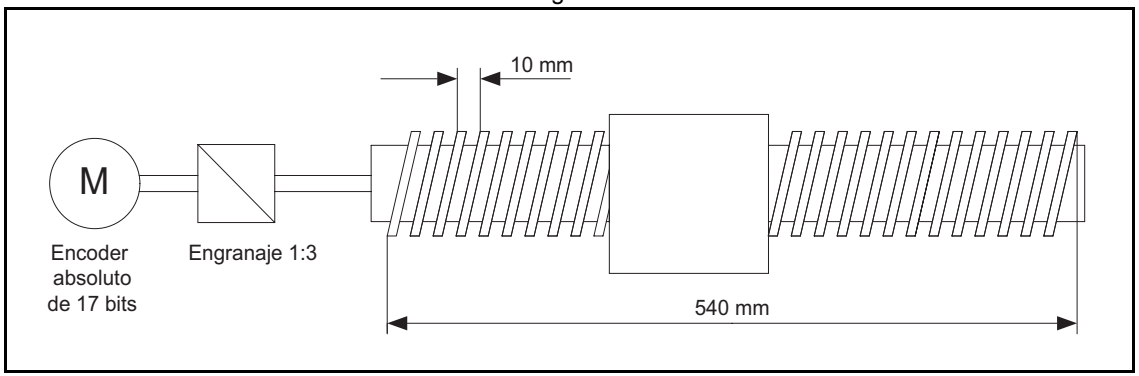
UNITS = 2^{15} = 32768
Pn202 = 1
Pn203 = 15
Pn205 = 4
REP_DIST = 50
REP_OPTION = 1

Con estos ajustes, al ejecutar **MOVE(50)** se mueve la pieza móvil 50 mm o una estación.

Ejemplo 6

El sistema mecánico consta de un tornillo móvil. Utiliza un servomotor con un encoder absoluto de 17 bits. La relación mecánica del engranaje es 1:3. La separación de rosca del tornillo móvil es de 10 mm por revolución. La distancia de recorrido total del tornillo móvil es de 540 mm. Las unidades de medición mecánica deben estar en mm.
Con el mismo procedimiento que en el ejemplo 1, tenemos:

fig. 24



$$\frac{Pn202}{Pn203}\,UNITS = \frac{2^{17}\cdot contajes_encoder}{1\cdot revolución_motor}\,\frac{3\cdot revolución_motor}{1\cdot revolución_tornillo_movil}\,\frac{1\cdot revolución_tornillo_movil}{10\ mm} =$$

$$= \frac{2^{17}\cdot 3}{10}\,\frac{contajes_encoder}{Mm}$$

Por lo tanto:

$$\frac{Pn202}{Pn203}UNITS = 2^{17}\frac{3}{10} = 2^{17}\frac{3}{2\cdot 5} = 2^{16}\frac{3}{5}$$

Una solución es:

UNITS = 2^{16} = 65536
Pn202 = 3
Pn203 = 5

En este caso no es necesario el cálculo del parámetro Pn205 de ajuste de límite multigiro porque el tornillo móvil es un sistema como un eje fijo (limitado). Basta con establecer este valor lo suficientemente grande para que el desbordamiento del contador esté fuera de la posición efectiva. Además, como el eje es finito, no es importante establecer el parámetro **REP_OPTION**, porque **REP_DIST** debe ser lo suficientemente grande para que esté fuera de la posición efectiva máxima (540 mm). Una solución es: **REP_DIST** = 1.000 y **REP_OPTION** = 0.
Con estos ajustes, al ejecutar **MOVE(17)** se mueve el tornillo móvil 17 mm en dirección de avance.

6.1.4 Asignación de entradas y salidas del servodriver

El controlador Trajexia tiene un espacio de E/S digitales que consta de 256 entradas digitales y 256 salidas digitales.
El rango de salidas digitales tiene cuatro partes:

- Salidas digitales 0 – 7.
  Estas salidas no existen físicamente en la TJ1-MC__. Si escribe en estas salidas, no sucede nada. Si lee estas salidas, devuelven 0.
- Salidas digitales 8 – 15.
  Estas salidas existen físicamente en la TJ1-MC__. Puede acceder físicamente a ellas en el conector sin tornillos de 28 pines de la parte frontal de la TJ1-MC__ (consulte el manual de referencia de hardware para obtener detalles). Si escribe en estas salidas, se activan y dan una señal de 24 Vc.c. Si lee estas salidas, devuelven su estado actual. Utilice el comando **OP** para escribir y leer estas salidas.
- Salidas digitales 16 – 31.
  Estas salidas son sólo de software. No existen físicamente en la TJ1-MC__, pero puede escribir en ellas y leer su estado correcto. Estas salidas se utilizan principalmente en programas BASIC para llevar a cabo algunas secuencias de control que requieren salidas que no sean físicas. Utilice el comando **OP** para escribir y leer estas salidas.
- Salidas digitales 32 – 255.
  Estas salidas están presentes físicamente sólo si hay unidades de E/S digitales conectadas a la TJ1-MC__ mediante el bus MECHATROLINK-II. Leer y escribir en ellas no tiene efecto si no existen físicamente (cuando las unidades de E/S no están conectadas). Utilice el comando **OP** para escribir y leer estas salidas.

Todas las salidas son únicas al controlador. No se accede a ellas por eje.

El rango de entradas digitales tiene tres partes:

- Entradas digitales 0 – 15.
  Estas entradas existen físicamente en la TJ1-MC__. Puede acceder físicamente a ellas en el conector sin tornillos de 28 pines de la parte frontal de la TJ1-MC__ (consulte el manual de referencia de hardware para obtener detalles). Estas entradas están activas (ON) cuando se les aplica una señal de 24 Vc.c. Cuando se leen, devuelven su estado actual. Utilice el comando **IN** para leer estas entradas.
- Entradas digitales 16 – 31.
  Estas entradas son sólo de software. No existen físicamente en la TJ1-MC__, pero se pueden leer. Se utilizan principalmente en programas BASIC para llevar a cabo algunas secuencias de control que requieren entradas que no sean físicas. Utilice el comando **IN** para leer estas entradas.
- Entradas digitales 32 – 255.
  Estas entrada están presentes físicamente sólo si hay unidades de E/S digitales conectadas a la TJ1-MC__ mediante el bus MECHATROLINK-II. Si se leen y no existen físicamente (las unidades de E/S no están conectadas), devuelven 0. Utilice el comando **IN** para leer estas entradas.

Todas las entradas son únicas al controlador. No se accede a ellas por eje.

Entradas de servodrivers MECHATROLINK-II en el espacio de E/S de Trajexia

Con el comando BASIC **IN**, puede acceder a las entradas presentes físicamente en un programa BASIC. Estas entradas pueden estar incorporadas en el controlador o conectadas mediante el bus MECHATROLINK-II.
Algunas entradas del servodriver están asignadas al espacio de E/S de Trajexia. Por lo tanto, puede acceder a estas entradas desde el programa BASIC. Trajexia sólo admite esto para los servodrivers conectados al sistema Trajexia mediante el bus MECHATROLINK-II.
La selección de entradas del servodriver que están asignadas al espacio de E/S de Trajexia depende del valor del parámetro Pn81E.
El valorrecomendado de Pn81E es 0x4321.
Con este valor predeterminado, la asignación es la siguiente para el servodriver Sigma-II:

Bit DRIVE_MONITOR	Señal de conector	Entrada de Trajexia
6	CN1-44	N/A
7	CN1-45	N/A
8	CN1-46	N/A
12	CN1-40	28
13	CN1-41	29
14	CN1-42	30
15	CN1-43	31

Con este valor predeterminado, la asignación es la siguiente para el servodriver Junma:

Bit DRIVE_MONITOR	Señal de conector	Entrada de Trajexia
2	CN1-2	26
6	CN1-1	27

A las entradas de servodriver asignadas al espacio de E/S de Trajexia como éste se accede desde el programa por eje y no se puede acceder de la forma habitual mediante el comando **IN**. La única forma en que se pueden utilizar estas entradas en el programa consiste en asignarlas a los parámetros de eje **DATUM_IN**, **FHOLD_IN**, **FWD_IN** y **REV_IN**. Se utilizan las entradas del servodriver de eje, según el eje del que estén establecidos los parámetros.

Ejemplo: tenemos dos drivers Sigma-II asignados a los ejes de controlador 0 y 3. Para ambos ejes deseamos utilizar la señal de entrada CN1-41 que sirva de entrada de límite de marcha inversa. Podemos hacerlo con estos comandos:

REV_IN AXIS(0) = 29
REV_IN AXIS(3) = 29

Observe que aunque los parámetros **REV_IN** para ambos ejes tienen el mismo valor 29, las entradas reales utilizadas no son las mismas. Para el eje 0, se utiliza la entrada CN1-41 del primer driver (asignado a dicho eje), pero para el eje 3 se utiliza la misma entrada CN1-41 del otro driver (el asignado al eje 3). Por lo tanto, decimos que a estas entradas se acceden por eje y no son únicas para todo el controlador. En general, estas dos entradas tienen un estado distinto al mismo tiempo. Además, observe que a ninguna de estas dos entradas se puede acceder mediante el comando **IN**. Por ejemplo, el comando **IN(29)** devuelve el estado de la entrada 29 de software de controlador (única para todos los ejes), que tiene un estado distinto a las entradas de servodriver asignadas al mismo número. Puede obtener más información sobre la asignación de entradas de servodrivers MECHATROLINK-II al espacio de E/S de Trajexia en el manual de programación de Trajexia, en los comandos BASIC **DATUM_IN**, **FHOLD_IN**, **FWD_IN** y **REV_IN**.

6.1.5 Búsqueda de origen

La función de búsqueda de origen o de retorno al inicio se suele considerar como una secuencia concreta de movimientos de un eje en la fase de inicio de la máquina. Esta secuencia se realiza automáticamente en la mayoría de los casos, sin la entrada desde el operario de la máquina. En general, un procedimiento de búsqueda de origen acopla una posición a un eje específico. Depende de los encoders utilizados (absolutos o relativos), del sistema utilizado (lineal o circular) y de la construcción mecánica de la máquina. Los encoders absolutos no necesitan un movimiento durante el procedimiento de búsqueda de origen porque las posiciones exactas se transfieren directamente al sistema. Para otros tipos de encoder es necesario un movimiento, ya que no se conoce la posición exacta dentro del sistema. Básicamente, este movimiento se realiza a baja velocidad en alguna dirección hasta que se alcanza un determinado punto de medición. Dicho punto de medición se puede escanear desde ambas direcciones para aumentar la precisión. Al inicio, las posiciones actuales de los ejes que utilizan encoders incrementales son 0. Debido a que estas posiciones no coinciden con el 0 mecánico de la máquina, es necesario ejecutar la secuencia de retorno al inicio. Si se utiliza un encoder absoluto, la posición absoluta se lee al inicio desde el encoder y no es necesario el retorno al inicio. En este caso, se debe ejecutar una secuencia de inicio una vez durante la puesta en marcha de la máquina.
En la práctica, existen diferentes secuencias de búsqueda de origen. Son distintas en estas áreas:

- Los medios utilizados para detectar las posiciones de límite de la pieza móvil (sensores, interruptores, etc.)
- Posición de origen (inicio) o referencia.
- Posiciones posibles de la pieza móvil relacionada con las posiciones de límite y posición de origen.

Trajexia incluye algunas secuencias de retorno al inicio básicas:

- **DATUM(0)**
  No se trata realmente de una búsqueda de origen. Este comando establece **DPOS = MPOS** y cancela los errores de eje.
- **DATUM(1)**
  Realiza una búsqueda de origen en dirección de marcha directa utilizando la marca Z de un encoder como interruptor de retorno al inicio.
- **DATUM(2)**
  Realiza una búsqueda de origen en dirección de marcha inversa utilizando la marca Z de un encoder como interruptor de retorno al inicio.
- **DATUM(3)**
  Realiza una búsqueda de origen en dirección de marcha directa utilizando la entrada seleccionada en **DATUM_IN** como interruptor de retorno al inicio.
- **DATUM(4)**
  Realiza una búsqueda de origen en dirección de marcha inversa utilizando la entrada seleccionada en **DATUM_IN** como interruptor de retorno al inicio.
- **DATUM(5)**
  Realiza una búsqueda de origen en dirección de marcha directa utilizando la entrada seleccionada en **DATUM_IN** como interruptor de retorno al inicio y busca la siguiente marca Z de un encoder.
- **DATUM(6)**
  Realiza una búsqueda de origen en dirección de marcha inversa utilizando la entrada seleccionada en **DATUM_IN** como interruptor de retorno al inicio y busca la siguiente marca Z de un encoder.

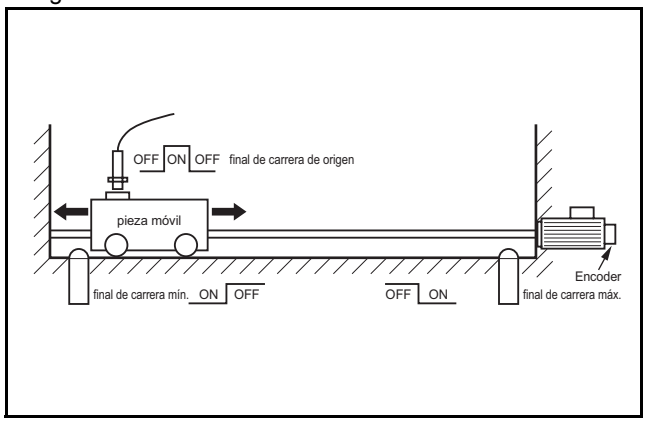
Para obtener más detalles sobre estas secuencias de retorno al inicio predefinidas, consulte la sección 3.2.68.
En algunas situaciones se requieren secuencias de retorno al inicio más complejas:

- Búsqueda de origen de interruptor absoluto además de finales de carrera.
- Búsqueda de origen según finales de carrera.
- Búsqueda de origen según piezas de hardware que bloquean el movimiento.
- Búsqueda de origen utilizando el impulso de referencia de encoder “marca cero”.
- Búsqueda de origen estático, forzando una posición desde una referencia de usuario.
- Búsqueda de origen estático, forzando una posición desde un encoder absoluto.

En la figura se muestra un caso de búsqueda de origen general. Esta secuencia de búsqueda de origen simple tiene 3 pasos:

1. Buscar una señal.
2. Buscar otra señal.
3. Mover el eje a una posición predefinida.

fig. 25



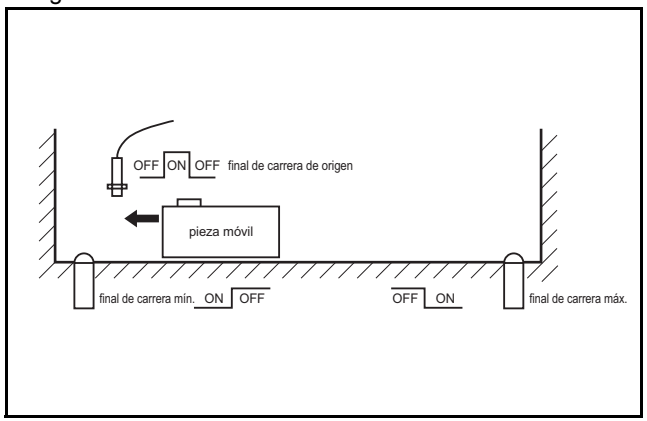
Por motivos de seguridad, los finales de carrera están normalmente cerrados. Por este motivo, en esta figura y en las posteriores de esta sección, el nivel de señal baja se indica como ON y el nivel de señal alta se indica como OFF.

Es importante observar que, antes de que se ejecute ningún procedimiento de retorno al inicio, es necesario establecer los parámetros de eje **UNITS**, **REP_DIST** y **REP_OPTION**, y los parámetros de servodriver Pn202, Pn203 y Pn205 correctamente y según el sistema mecánico y las unidades de medición deseadas que se han utilizado en programación. Estos parámetros influyen en la búsqueda de origen, especialmente si se utiliza un encoder absoluto. Para obtener más información sobre estos parámetros, consulte la sección 6.1.2.

Búsqueda de origen de interruptor absoluto además de finales de carrera

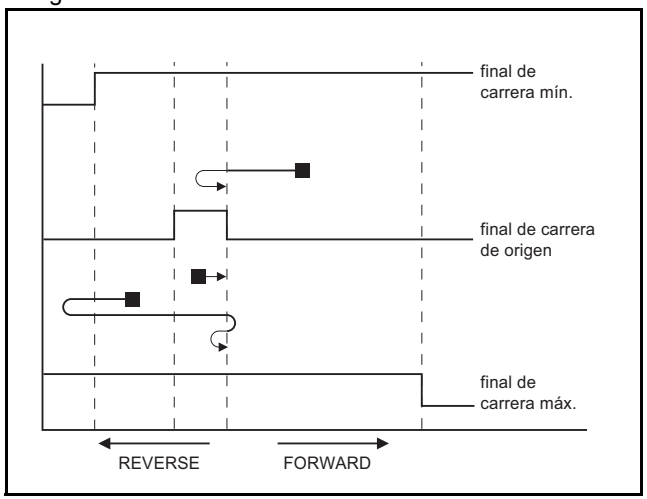
La función de búsqueda de origen se lleva a cabo buscando un final de carrera externo con una posición absoluta y cuya posición defina la de origen. En la figura se muestra el ejemplo de este procedimiento de retorno al inicio.

fig. 26



En la figura se muestran las situaciones posibles para la búsqueda de origen absoluto además de finales de carrera. Estas situaciones dependen de la posición de la pieza móvil cuando se conecta la alimentación.
El ejemplo de programa que realiza esta secuencia de búsqueda de origen se ofrece a continuación.

fig. 27



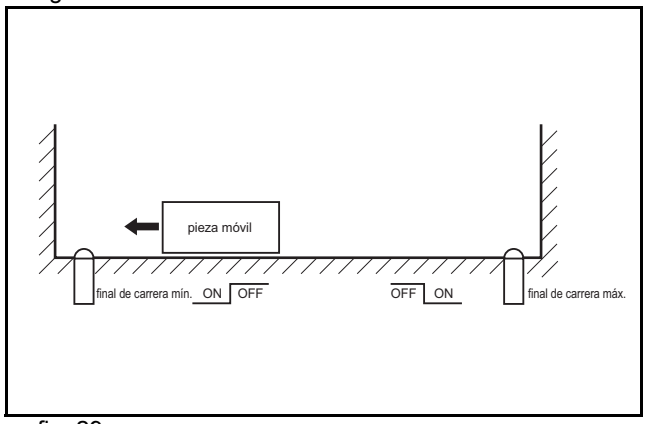

```
'Interruptor de origen absoluto: IN0
'Final de carrera izquierdo: IN1
'Final de carrera derecho: IN2
BASE(0)
DATUM_IN=0
FW_IN=2
RV_IN=1
SERVO=ON
WDOG=ON
DATUM(4)
WA(1)
WAIT UNTIL MTYPE=0 OR IN(1)=OFF
IF IN(1)=ON
    FORWARD
    WAIT UNTIL IN(0)=ON
    WAIT UNTIL IN(0)=OFF
    CANCEL
    DATUM(4)
```

```
    WA(1)
    WAIT IDLE
ENDIF
```

Búsqueda de origen según finales de carrera

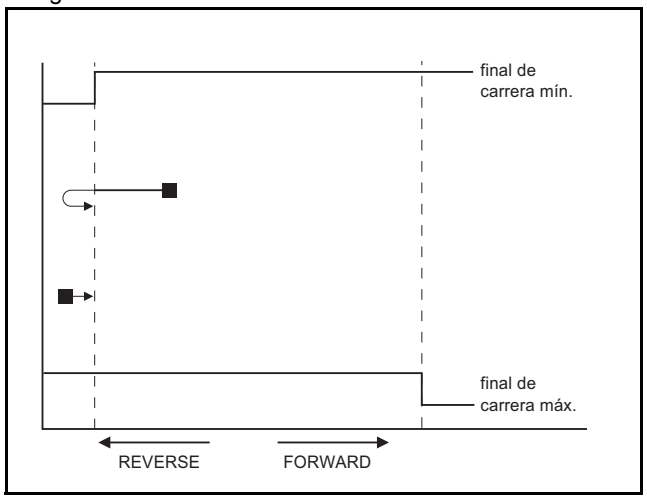
Esta función de búsqueda de origen se realiza buscando un sensor externo mediante finales de carrera únicamente. En la figura se muestra el ejemplo de este procedimiento de retorno al inicio.

fig. 28



Las situaciones posibles para la búsqueda de origen según finales de carrera, en función de la posición de la pieza móvil en el encendido, se muestran en la figura.
El ejemplo de programa que realiza esta secuencia de búsqueda de origen se ofrece a continuación.

fig. 29

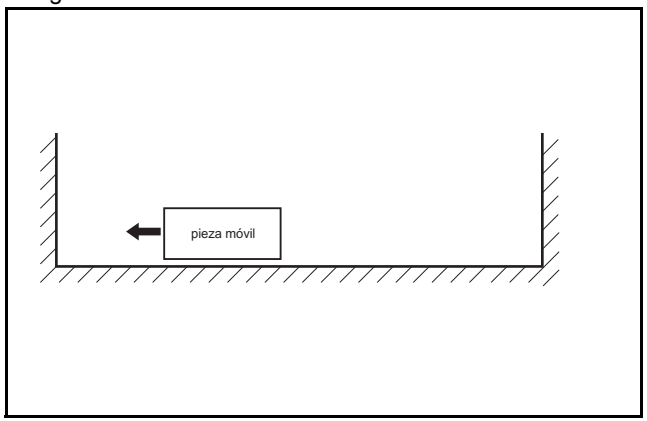



```
'Final de carrera de origen e izquierdo: IN0
'Final de carrera derecho: IN1
BASE(0)
DATUM_IN=0
SERVO=ON
WDOG=ON
DATUM(4)
WA(1)
WAIT IDLE
```

Búsqueda de origen según piezas de hardware que bloquean el movimiento

Este procedimiento de búsqueda de origen realiza la búsqueda según un objeto físico y bloquea mecánicamente el movimiento. No hay finales de carrera, interruptor de posición ni impulsos de referencia. La posición de origen se determina mediante la detección de una cantidad de par concreto en los objetos de bloqueo. Se requiere un límite de par adecuado para no dañar la parte mecánica durante el proceso de búsqueda de origen. En la figura se muestra el ejemplo de este procedimiento de retorno al inicio.
El ejemplo de programa que realiza esta secuencia de búsqueda de origen se ofrece a continuación.

fig. 30

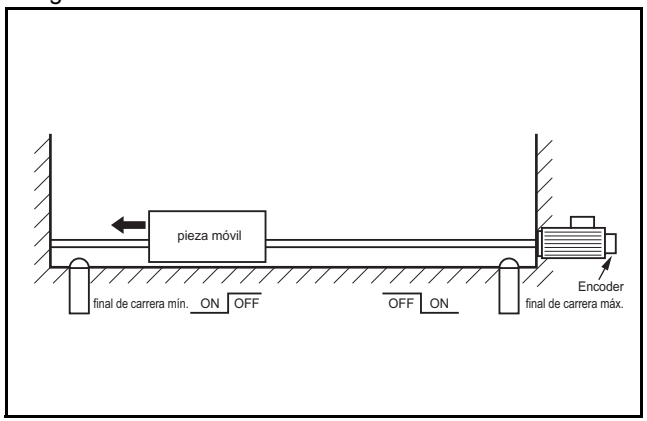



```
BASE(0)
DRIVE_CONTROL=11 'Monitorizar par con DRIVE_MONITOR
SERVO=ON
WDOG=ON
SPEED=CREEP
REVERSE
WA(1)
WAIT UNTIL DRIVE_MONITOR < -100
    'Esperar una determinada cantidad de par aplicado
CANCEL
DEFPOS(0)
MOVEABS(10)'Esto es necesario; de lo contrario, la posición
           'se mantiene empujando el límite de hardware de la
           'máquina y el motor avanza por sobrecarga
```

Búsqueda de origen utilizando el pulso de referencia de encoder “Z”

Este procedimiento de búsqueda de origen realiza la búsqueda de la señal “Z” (Zero Mark) del encoder. Esta señal también se denomina “marcador” o “impulso de referencia”. Aparece una vez por revolución de encoder completa. En la figura se muestra el ejemplo de este procedimiento de retorno al inicio.

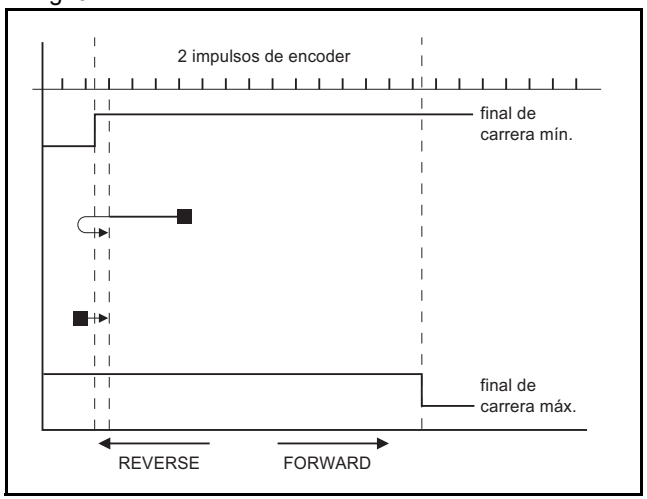
fig. 31



Las situaciones posibles para la búsqueda de origen con el pulso de referencia de encoder “Z”, en función de la posición de la pieza móvil en el encendido, se muestran en la figura.
El ejemplo de programa que realiza esta secuencia de búsqueda de origen se ofrece a continuación.

```
'Final de carrera de origen e izquierdo: IN0
'Final de carrera derecho: IN1
REV_IN=-1
BASE(0)
DATUM_IN=0
SERVO=ON
WDOG=ON
DATUM(6)
WA(1)
WAIT IDLE
```

fig. 32



Búsqueda de origen estático, forzando una posición desde una referencia de usuario

Este procedimiento de búsqueda de origen la realiza forzando directamente una posición real. No realiza ningún movimiento físico.

```
DATUM(0)
```

Búsqueda de origen estático, forzando una posición desde un encoder absoluto

Este procedimiento de búsqueda de origen establece la posición real en la posición de un encoder absoluto. No realiza ningún movimiento físico. Sólo es posible con un eje con un encoder absoluto en un lazo de control.

6.1.6 Registro

El registro, también denominado “enclavamiento o latch” o “registro de impresión”, es el almacenamiento en tiempo real de la posición de un eje cuando se activa una entrada externa. La información que se registra, es decir, se almacena, se procesa posteriormente, no en tiempo real, por parte del programa de aplicación.
El registro es distinto del procesamiento de una entrada o señal de interrupción. Con el registro, no se genera ningún evento cuando se activa la entrada de registro. Además, la ejecución normal del programa de aplicación no se ve afectada ni se interrumpe. Sólo se almacena la posición de un eje. Esta información se puede utilizar, al igual que otros parámetros o valores, en un programa. La información de registro está a disposición de un programa inmediatamente después del registro.

La ventaja del registro es que se realiza muy rápidamente. Por lo tanto, la posición de eje que se almacena es muy precisa. Para lograr esta velocidad y precisión, el registro se implementa con hardware y la entrada de registro debe estar en la misma placa que la entrada de encoder que proporciona información sobre la posición de eje. La captura y el almacenamiento de la posición de eje se efectúa en tiempo real mediante hardware. El programa de aplicación no lleva a cabo el procesamiento de esta información en tiempo real.

Comando de eje REGIST

En Trajexia, el registro se efectúa con el comando de eje **REGIST**. Este comando tiene un solo argumento. Este argumento determina la entrada externa que se registra, si el registro se ejecuta en el flanco ascendente o descendente de la señal de entrada, si se utiliza la función de intervalo y otras opciones. Para obtener más información sobre el comando **REGIST**, consulte la sección 3.2.219.
El registro varía para los distintos ejes según su conexión al sistema. Si un eje está conectado mediante el bus MECHATROLINK-II, el registro se lleva a cabo en el hardware del servodriver. Si un eje está conectado mediante la interfaz analógica del servodriver y la TJ1-FL02, el registro se efectúa en el hardware de la TJ1-FL02.
A continuación se describen los diferentes registros.

Registro en el servodriver Sigma-II

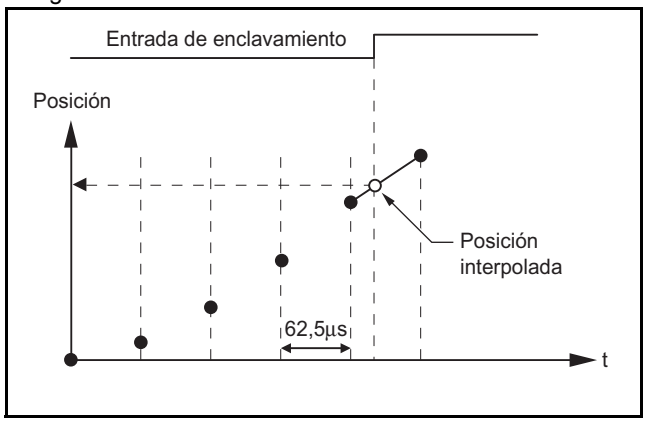
El registro en el servodriver Sigma-II se produce cuando un eje asignado al servodriver Sigma-II está conectado al sistema Trajexia mediante el bus MECHATROLINK-II. Existen tres entradas de registro en el servodriver Sigma-II, pero sólo un enclavamiento de hardware, por lo que únicamente se puede utilizar una entrada cada vez. Las entradas físicas están en los pines CN1-44, CN1-45 y CN1-46 en el conector CN1 de 50 pines, pero Trajexia utiliza las entradas lógicas EXT1, EXT2 y EXT3 para asociar las entradas físicas a las lógicas. Esta asociación se lleva a cabo estableciendo el parámetro Pn511 del servodriver. Para obtener más información sobre la configuración de esta asociación y el parámetro Pn511, consulte la sección 3.2.219, tabla 1. La entrada utilizada para el registro se determina mediante el argumento del comando **REGIST**.

El retardo en la captura en el servodriver Sigma-II es de unos 3 µs. Como la información del encoder se actualiza cada 62,5 µs, es necesario efectuar interpolación para obtener el valor de posición capturada correcto (consulte la figura). Como la velocidad del motor no puede cambiar demasiado durante 62,5 µs, la precisión resultante es muy alta.
Los retrasos en la transmisión de la información son:

- Retraso en la activación del registro: 0,625 ms a 4 ms.
- Retraso en la recepción del registro: 3,5 ms.
- Retraso en la captura del registro: 3 µs.

También es posible utilizar la marca Z del encoder para registrar una posición de eje. Esto también se realiza con el argumento del comando **REGIST**.

fig. 33



Registro en el servodriver Junma

El registro en el servodriver Junma es igual que en el Sigma-II, con una diferencia: sólo hay una entrada física y otra también de enclavamiento lógico, por lo que no se necesitan ajustes de los parámetros del servodriver. La entrada física está asociada al enclavamiento lógico EXT1 y sólo se puede utilizar el flanco ascendente para el registro.

Registro en la TJ1-FL02

La TJ1-FL02 tiene dos entradas de registro físicas y dos circuitos de enclavamiento por entrada de encoder, que se pueden utilizar independientemente. Por lo tanto, se pueden utilizar dos entradas de registros independientes al mismo tiempo. Para obtener más información sobre cómo utilizar ambas entradas de registro de la TJ1-FL02 al mismo tiempo, consulte las secciones 3.2.170, 3.2.171, 3.2.217, 3.2.218 y 3.2.219.
El retraso en la captura es de 0,5 µs. Como la posición del encoder se lee continuamente desde la entrada de encoder de line driver, no es necesaria la interpolación. El retraso de la transmisión de la información capturada es sólo un ciclo **SERVO_PERIOD**.

Uso del registro en programas de aplicación

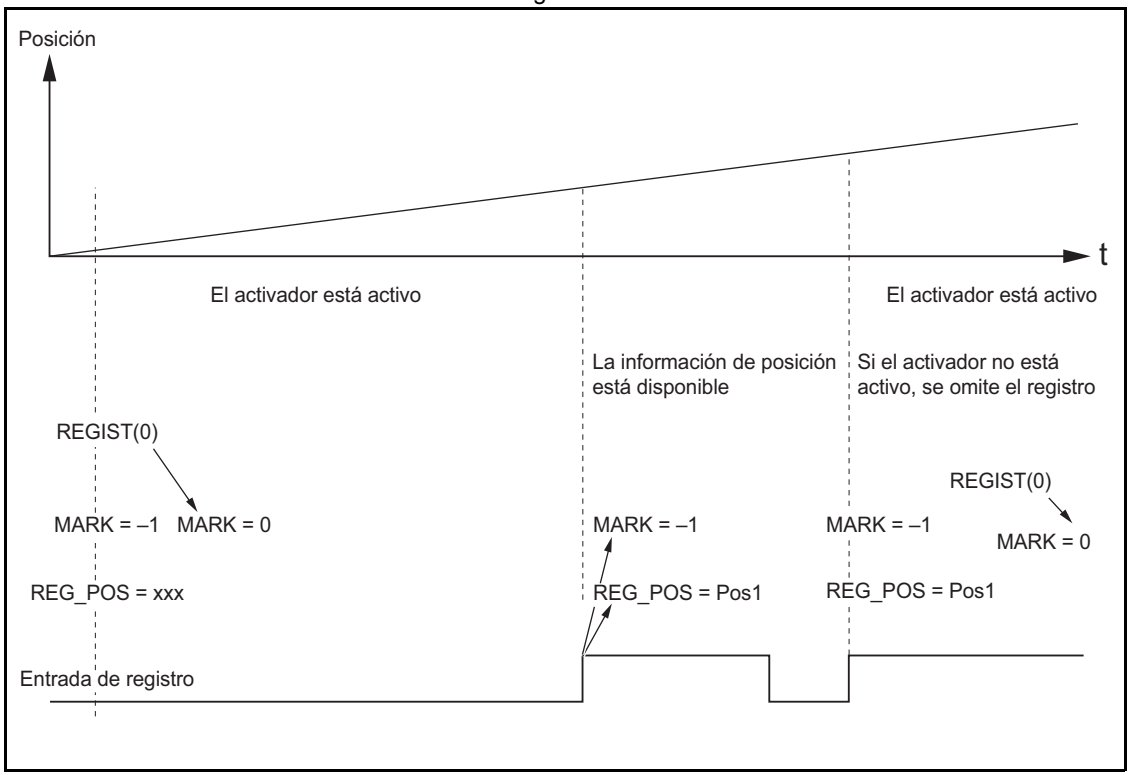
Existe un comando de eje (**REGIST**) y dos parámetros de eje (**MARK** y **REG_POS**). Con estos comandos y parámetro, se puede controlar y usar la funcionalidad de registro en los programas BASIC.

- **REGIST** captura la posición de eje cuando se detecta una señal de registro. Los ajustes disponibles dependen del tipo de eje. Consulte la sección 3.2.219.
- **MARK** es un indicador que señaliza si la posición se ha capturado o no. Para la segunda entrada de registro de la TJ1-FL02, también está disponible el parámetro **MARKB**. Para obtener más información, consulte las secciones 3.2.170 y 3.2.171.
- **REG_POS** contiene la posición de eje capturada. Sólo si el indicador **MARK** señaliza que la posición se ha capturado correctamente, el valor de **REG_POS** se puede considerar válido. Para la segunda entrada de registro de la TJ1-FL02, también está disponible el parámetro **REG_POSB**. Para obtener más información, consulte las secciones 3.2.217 y 3.2.218.

En la figura se ofrece la secuencia de la ejecución de los comandos y los registros del programa de ejemplo siguiente.

```
BASE(N)
REGIST(0)
WAIT UNTIL MARK=0
lazo:
    WAIT UNTIL MARK=-1
    PRINT "Posición capturada en: "; REG_POS
    REGIST(0)
    WAIT UNTIL MARK=0
GOTO lazo
```

fig. 34



Registro y función de intervalo

La función de intervalo permite que el registro se produzca sólo dentro de un rango especificado de posiciones de eje. Esta función se selecciona asignando el valor adecuado como un argumento del comando **REGIST**. La función de intervalo se controla mediante dos parámetros de eje, **OPEN_WIN** y **CLOSE_WIN**. Para obtener más información sobre **REGIST**, **OPEN_WIN** y **CLOSE_WIN**, consulte las secciones 3.2.49, 3.2.198 y 3.2.219.

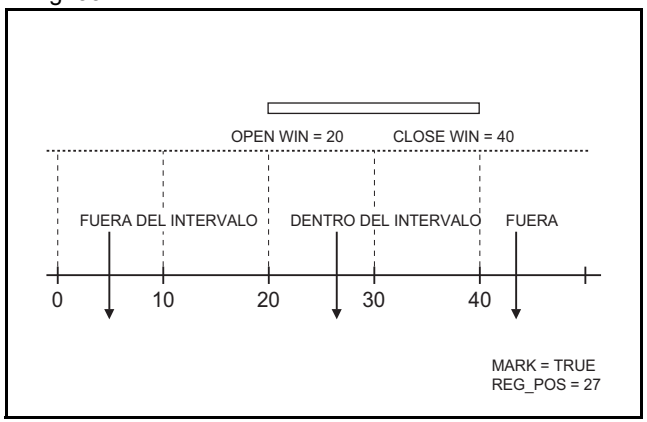
Existen dos tipos de intervalo:

- El intervalo inclusivo permite que el registro se produzca sólo dentro del intervalo especificado de posiciones de eje. Con esta función de intervalo, los eventos de registro se omiten si la posición medida del eje es menor que el parámetro de eje **OPEN_WIN** o mayor que el parámetro **CLOSE_WIN**.
- El intervalo exclusivo permite que el registro se produzca sólo fuera del intervalo especificado de posiciones de eje. Con esta función de intervalo, los eventos de registro se omiten si la posición medida del eje es mayor que el parámetro de eje **OPEN_WIN** o menor que el parámetro **CLOSE_WIN**.

Cuando se utiliza la función de intervalo, el proceso interno es el siguiente:

1. **REGIST** + intervalo se ejecuta en el programa.
2. **MARK** = 0 y se activa el enclavamiento.
3. La posición se captura y transmite al procesador de Trajexia.
4. ¿La posición capturada está dentro del intervalo inclusivo o fuera del intervalo exclusivo?
   - En caso afirmativo, **MARK** = –1 y se actualiza **REG_POS**.
   - En caso negativo, se vuelve al punto 2 (activar el enclavamiento de nuevo de forma transparente para el usuario).

fig. 35

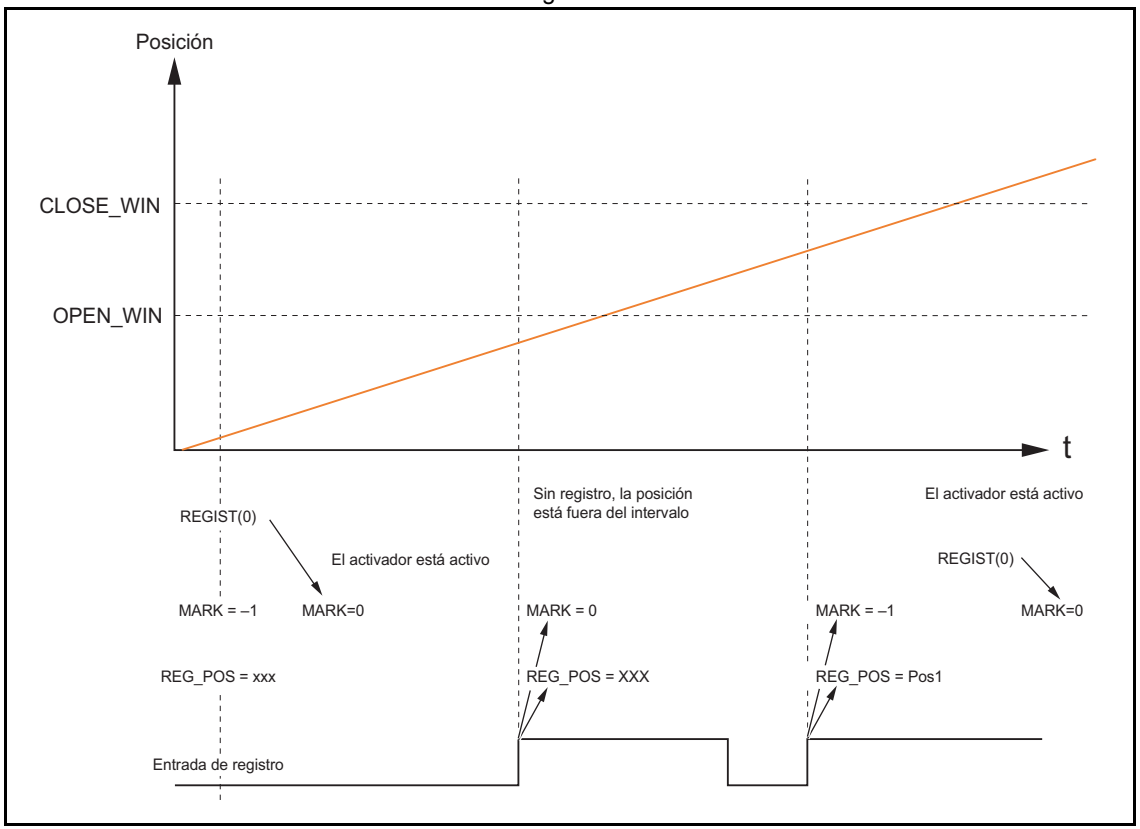


En la figura se muestra la secuencia de ejecución de los comandos anteriores y la aparición de los eventos de registro cuando se utiliza el intervalo inclusivo.
Existen retrasos entre estos eventos:

- Trajexia recibe el enclavamiento.
- Trajexia decide activar el enclavamiento de nuevo.
- Se activa el enclavamiento.

A consecuencia de estos retrasos no hay certeza en los flancos del intervalo de cuándo las marcas se pueden detectar cerca de los flancos. Esto se advierte más para los ejes conectados al sistema mediante el bus MECHATROLINK-II debido a los retrasos de bus. Para compensar estos retrasos, el usuario debe establecer los márgenes de intervalo lo suficientemente grandes.

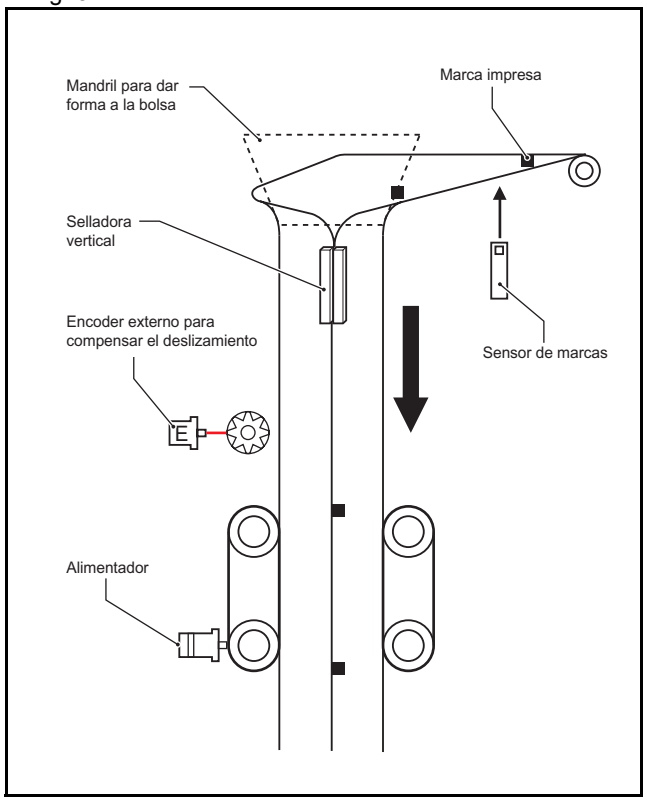
fig. 36



Ejemplo: Corrección de la posición de un eje

En la figura se muestra la máquina de llenado y sellado vertical para el embalaje de productos en bolsas. El material de las bolsas procede de una bobina de película de plástico que se va desbobinando; a continuación, se le da forma en el tubo mediante un mandril mecánico y, al mismo tiempo, el tubo se sella verticalmente. El movimiento del alimentador es intermitente y la longitud de alimentación corresponde a la longitud de la bolsa. Una vez alimentada la bolsa, el sellador horizontal cierra la bolsa para que se pueda rellenar con el producto. Tras ello, el proceso comienza de nuevo, alimentando otra bolsa.

fig. 37



El alimentador puede funcionar en dos modos: sin marca de registro y con ella. El funcionamiento sin marca de registro es un simple movimiento incremental de punto a punto. En este caso, no hay garantía alguna de que el alimentador se mueva exactamente la misma distancia que el patrón de diseño. Por ejemplo, suponga que la longitud de bolsa que se tiene que cargar es de 200 mm, pero el patrón real es de 200,1 mm. Con el simple movimiento incremental de punto a punto sin corrección, se acumula un error de 0,1 mm por bolsa. Con una cantidad pequeña de bolsas la diferencia no es apreciable, pero después de 500 bolsas el error es de 50 mm, lo que supone un 25% de la longitud de la bolsa.

Al trabajar con marcas de registro, el controlador Motion ejecuta un movimiento incremental hasta una determinada posición. Si durante el posicionamiento se detecta la marca de registro, se cambia la posición objetivo para finalizar el movimiento en una posición definida después de la marca de registro. Por lo tanto, siempre está garantiza la misma distancia con respecto a la marca de registro.

fig. 38

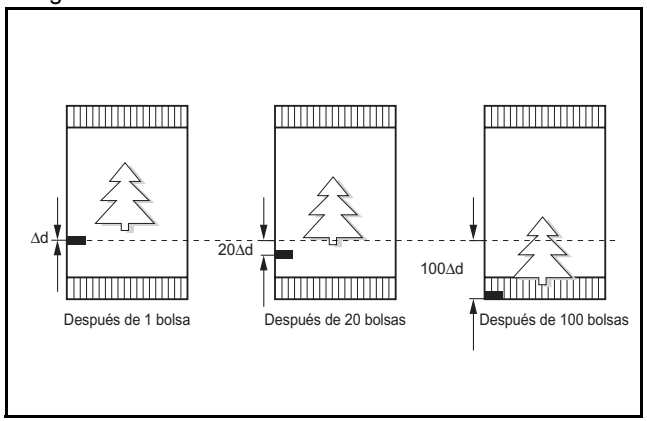
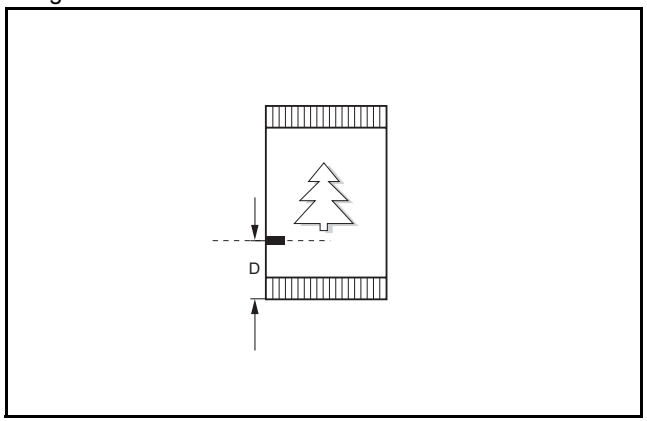


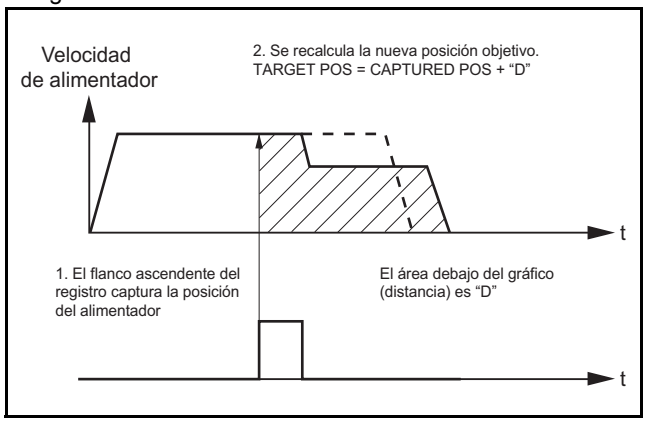
fig. 39



El perfil de movimiento y su modificación debido a la marca de registro se muestran en fig. 39.
El parámetro BASIC de este ejemplo es:

```
DEFPOS(0)
REGIST(3)            'Activar el registro de marca
MOVE(longitud_bolsa) 'Mover a la distancia teórica
WA(1)
WAIT UNTIL MARK OR MTYPE=0
IF MARK THEN
    posicion_final=REG_POS+distancia_tras_marca
    MOVEMODIFY(posicion_final)
    'Corregir la distancia según la marca
ENDIF
```

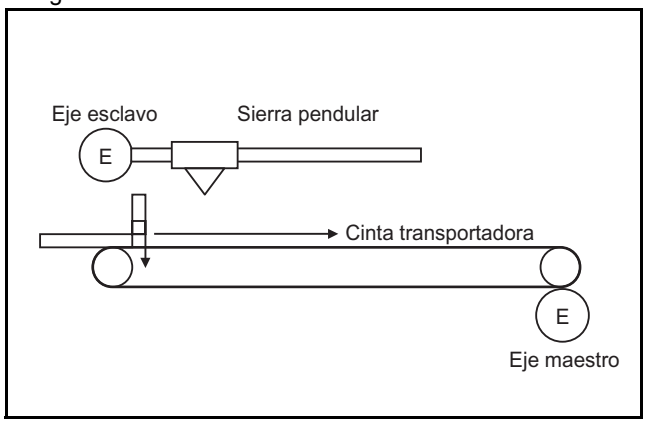
fig. 40



Ejemplo: Iniciar un eje esclavo en una posición precisa de un eje maestro

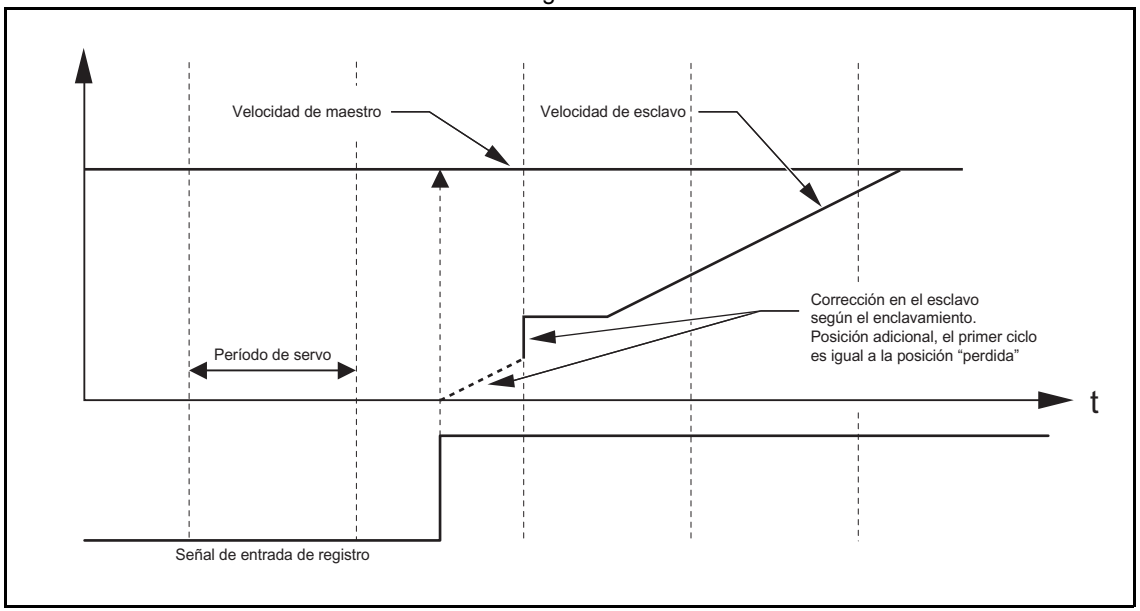
La imagen muestra una sierra pendular que corta la “cabeza” de las mesas de maderas. Cuando llega la madera, la fotocélula detecta el borde de la madera y, en el momento exacto, el movimiento de la sierra pendular empieza a sincronizarse con la posición correcta en la madera.
Si el programa inicia el movimiento, tras detectar una señal de la fotocélula, siempre hay al menos un período **SERVO_PERIOD** de tiempo de incertidumbre. En su lugar, el movimiento se inicia mediante el comando **MOVELINK** con **opcion_enlace**=1, lo que significa que el vínculo al eje maestro empieza cuando se produce el evento de registro en el eje de vínculo (maestro).
La secuencia de programa correspondiente es:
REGIST(2) AXIS(maestro)
MOVELINK(distancia,distancia_vinculo,aceleracion_vinculo, deceleracion_vinculo,maestro,1) AXIS(esclavo)
Para obtener más información sobre el comando **MOVELINK** y el argumento **opcion_vinculo**, consulte la sección 3.2.180.

fig. 41



La imagen muestra cómo se corrige la posición del eje esclavo mediante el evento de registro en el eje maestro para iniciar el movimiento del eje esclavo. La influencia de **SERVO_PERIOD** y el hecho de que el evento de registro se pueda producir en cualquier momento dentro del período **SERVO_PERIOD** se elimina por completo.

fig. 42



6.1.7 Seguimiento y monitorización

Funcionalidad de osciloscopio en Trajexia Tools

El osciloscopio de software es una parte estándar de Trajexia Tools. El osciloscopio se puede utilizar para realizar un seguimiento y representar gráficamente los parámetros de eje y sistema. Esto puede servir de ayuda con el desarrollo, puesta en marcha y solución de problemas del sistema Motion Control. Para obtener más información sobre el osciloscopio de software y sus funciones y capacidades, consulte la sección 5.5.4.
Puede activar el osciloscopio para iniciar el seguimiento según los parámetros de eje y sistema de dos formas: manualmente o por programa. La activación manual se realiza mediante la herramienta de osciloscopio. Los parámetros se almacenan en la memoria de tabla del controlador. El rango de la memoria de tabla donde los parámetros se almacenan se puede establecer desde la ventana **Oscilloscope Configuration** (Configuración de osciloscopio)

(consulte la sección 5.5.4). Con la activación manual, el usuario puede ver los cambios de los parámetros de eje y sistema en tiempo real, mientras el sistema está en funcionamiento. Un cambio en los valores de parámetro se representa gráficamente tan pronto como se produce el cambio. La limitación de la activación manual es que requiere interacción del usuario, lo que significa que el inicio del seguimiento no está sincronizado con el movimiento que se analiza. Además, con la activación manual, el rango de seguimiento está limitado a 200 muestras por canal.

Uso del osciloscopio

La alternativa, la activación mediante programa (trigger), no tiene las limitaciones de la activación manual del seguimiento. La activación mediante programa almacena los parámetros de eje y sistema en la memoria de la TJ1-MC__. Posteriormente, los parámetros se envían al osciloscopio para su representación gráfica. Los parámetros de eje y sistema se almacenan en la memoria de tabla. El rango de memoria utilizado se define mediante los parámetros del comando **SCOPE**. Cuando los parámetros están en la memoria de tabla, el osciloscopio se puede configurar para mostrar un rango de las ubicaciones de memoria de tabla en vez de los parámetros de eje y sistema. El momento exacto en que el seguimiento se inicia se puede determinar exactamente porque está controlado mediante el comando **TRIGGER**. Esto significa que el inicio del seguimiento está sincronizado con el movimiento. No hay límite de 200 muestras por canal. El osciloscopio indica tantas muestras (entradas de tabla) como lo configurado en la ventana **Oscilloscope Configuration** (Configuración de osciloscopio).

Ejemplo

En esta sección se ofrece un ejemplo práctico sobre el uso de los comandos **SCOPE** y **TRIGGER**, y cómo utilizarlos en combinación con el osciloscopio para monitorizar los parámetros de eje y solucionar problemas del sistema. Para obtener más información sobre los comandos **SCOPE** y **TRIGGER**, consulte las secciones 3.2.237 y 3.2.265.

Suponga que el sistema Motion Control consta de dos ejes: **AXIS(0)** y **AXIS(1)**. **AXIS(0)** es el eje maestro. Realiza un movimiento de avance simple. **AXIS(1)** es el eje esclavo. Debe seguir al eje maestro según la regla de coseno:

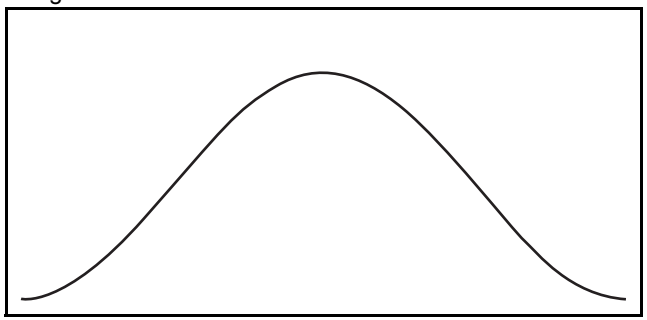
$$x_1 = end_pos \cdot \frac{1}{2}\left(1 - \cos\left(\frac{2\pi \cdot x_0}{999}\right)\right)$$

donde x_0 es la posición del maestro **AXIS(0)** y x_1 es la posición del esclavo **AXIS(1)**. Puede vincular los dos ejes con el comando **CAMBOX**. Para obtener más detalles, consulte la sección 3.2.42. Suponga además, que el parámetro **posicion_final** no es constante, sino que puede cambiar debido a las distintas condiciones del sistema Motion Control. La parte del programa que crea la tabla CAM es:

```
'Valores CAM iniciales
VR(posicion_final)=15
posicion_final_actual=VR(posicion_final)
FOR i=0 TO 999
    TABLE(i, VR(posicion_final)*(1-COS(2*PI*i/999))/2)
NEXT i
...
lazo:
IF VR(posicion_final)<>posicion_final_actual THEN
'Recalcular la tabla CAM
    FOR i=0 TO 999
        TABLE(i, VR(posicion_final)*(1-COS(2*PI*i/999))/2)
    NEXT i
    posicion_final_actual=VR(posicion_final)
ENDIF
...
GOTO lazo
```

El valor **VR(posicion_final)** se puede cambiar desde otro programa o externamente desde otro controlador mediante mensajes FINS. En este caso, se debe recalcular la tabla **CAM**.
La creación de la tabla CAM está completa. La inicialización del eje deseado y de los parámetros de sistema para el seguimiento es:

fig. 43

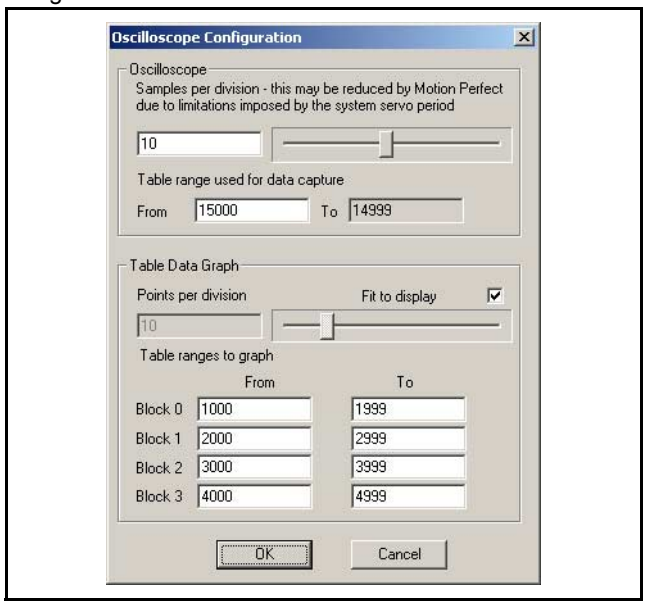


```
'Inicializaciones
FOR i=0 TO 1
    BASE(i)
    ATYPE=40
    UNITS=8192
    REP_DIST=20
    REP_OPTION=1
    FE_LIMIT=1
    DRIVE_CONTROL=11
    SPEED=8
    ACCEL=50
    DECEL=50
    DEFPOS(0)
    SERVO=ON
    CANCEL
NEXT i
WDOG=ON
BASE(1)
'Ajustes de osciloscopio:
'1 muestra cada 2 ciclos de servo
'Información almacenada en TABLE(1000) a TABLE(4999)
'Como capturamos 4 canales, tenemos 1000 muestras por canal.
'MPOS AXIS(0) se almacena en TABLE(1000) a TABLE(1999)
'DPOS AXIS(1) se almacena en TABLE(2000) a TABLE(2999)
'El par de referencia para AXIS(1) se almacena en
'TABLE(3000) a TABLE(3999)
'MSPEED AXIS(1) se almacena en TABLE(4000) a TABLE(4999)
'La captura abarca 1000 muestras * 2 ms/muestra = 2 segundos
SCOPE(ON,2,1000,4999,MPOS AXIS(0),DPOS,DRIVE_MONITOR,MSPEED)
FORWARD AXIS(0) 'Mover el eje maestro hacia delante
TRIGGER 'Iniciar seguimiento y almacenamiento de parámetros
WHILE NOT MOTION_ERROR
    'Caja de levas que se iniciará en AXIS(0) posición 1
    CAMBOX(0,999,UNITS,10,0,2,1)
    WAIT UNTIL MPOS AXIS(0)<1
    'La captura comenzará cuando el eje maestreo esté en una
    'posición entre 0 y 1. Las condiciones adicionales
    'son:
    '- La captura anterior ha finalizado
```

```
    '(SCOPE_POS=1000)
    '- Tenemos el permiso (VR(activar_activador)=ON)
    IF SCOPE_POS=1000 AND VR(activar_activador)=ON THEN
        TRIGGER
        PRINT "Activado"
    ENDIF
    WAIT IDLE
WEND
HALT
```

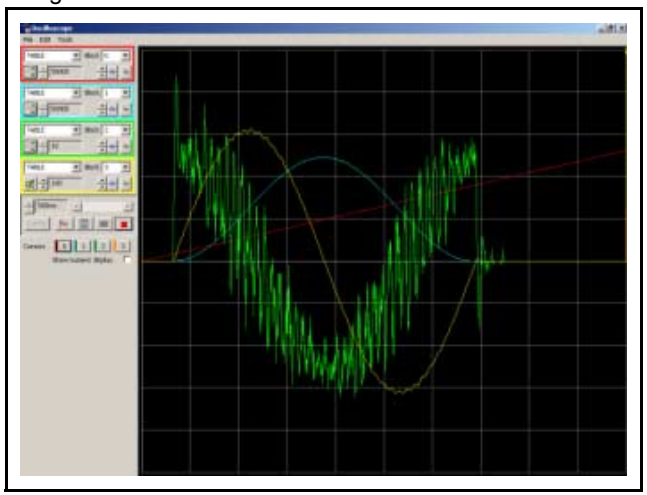
Para el resultado de la captura en el osciloscopio de Trajexia Tools, debe realizar los ajustes en la ventana **Oscilloscope Configuration** (Configuración de osciloscopio) tal como se indica en la figura. También debe desactivar la captura posterior para evitar la mezcla de resultados de dos capturas distintas en las mismas entradas de memoria de tabla.

fig. 44



El resultado de la captura se indica en la figura.

fig. 45



En el ejemplo indicado anteriormente, el valor del parámetro **UNITS** se establece en pulsos de encoder. La posición del eje maestro **MPOS AXIS(0)** se indica en rojo (bloque de tabla 0, Table(1000) a Table(1999), consulte los ajustes en la ventana **Oscilloscope Configuration** [Configuración de osciloscopio]). La posición aumenta linealmente, debido a que la velocidad del eje maestro es constante. La posición solicitada del eje esclavo **DPOS AXIS(1)** se indica en azul (bloque de tabla 1, Table(2000) a Table(2999), consulte los ajustes en la ventana **Oscilloscope Configuration** [Configuración de osciloscopio]). Este gráfico es una curva de coseno. Corresponde a la tabla CAM creada.
La velocidad medida del eje esclavo MSPEED AXIS(1) se indica en amarillo (bloque de tabla 3, Table(4000) a Table(4999), consulte los ajustes en la ventana **Oscilloscope Configuration** [Configuración de osciloscopio]). Este gráfico es una curva senoidal, porque la velocidad es una derivada de la posición y la derivada del coseno es el seno. A velocidades altas hay algunas fluctuaciones.
El gráfico verde es el par del motor para el eje esclavo configurado con **DRIVE_COMMAND = 11** como un porcentaje del par nominal. El par es proporcional a la aceleración. Debido a que la aceleración es una derivada de la velocidad y la velocidad es una curva senoidal, la aceleración (y también el par) es una curva de coseno. Hay un pico al comienzo y otro en la parada porque existe una discontinuidad en la aceleración. También hay una oscilación de alta frecuencia en la curva de par, que sugiere una frecuencia de resonancia que se puede eliminar mediante los ajustes de filtro de muesca en el servodriver Sigma-II. La alta frecuencia se refuerza, porque también se refleja en la curva de velocidad. Para obtener más información sobre los ajustes de filtro de muestra, consulte el manual del servodriver Sigma-II.

Detección y corrección de errores con el osciloscopio

Cuando los datos deseados se capturan y registran en las entrada de la memoria de tabla, puede utilizar el osciloscopio para visualizar estos datos. Esto puede servir de ayuda al poner en marcha y detectar y corregir errores del sistema. Esta sección ofrece un ejemplo de cómo un error, que es difícil de analizar, se puede explicar y solucionar claramente mediante los datos capturados y el osciloscopio.

El parámetro **posición_final**, que define los valores en la tabla CAM, depende las condiciones externas del sistema. Por lo tanto, un programa que se ejecuta en otra tarea, o incluso un dispositivo de control mediante comunicaciones FINS, puede cambiarlo mientras está en ejecución el programa que vincula dos ejes. Suponga que estos cambios en las condiciones, que dan como resultado un cambio del parámetro **posición_final**, suceden la mayor parte del tiempo cuando los ejes no están vinculados, es decir, cuando el comando **CAMBOX** no se ejecuta. Suponga, además, que en muy pocas ocasiones la condición cambia cuando los ejes están vinculados. El cambio del parámetro **posición_final** activa el recálculo de la tabla CAM mientras se ejecuta el comando **CAMBOX**. La consecuencia es que la parte de la posición solicitada del eje esclavo sigue el perfil antes del cambio y la otra parte sigue el perfil después del cambio. Al final, esto conduce a una discontinuidad del perfil, que provoca una velocidad indefinida del eje y termina con este error: WDOG se pone en OFF y todos los ejes se paran.
La situación anterior resulta difícil de analizar cuando no se sabe lo que sucede. Lo único que ve el usuario es que el eje esclavo tiene un error cada pocas horas o incluso con menos frecuencia. Pero el osciloscopio puede mostrar claramente dónde está el problema. Para poder utilizar el osciloscopio, se deben capturar todos los parámetros deseados en el momento de un error. Esto se puede llevar a cabo si se organizan los programas de aplicación de una determinada forma. La buena práctica de programación recomienda tener un programa de inicio independiente que esté configurado para ejecutarse automáticamente al inicio del sistema y comprueba la integridad del sistema, si todos los dispositivos previstos están conectados e inicializados. Para obtener un ejemplo de un programa de inicio, consulte la sección 6.1.1. Se recomienda dejar que el programa de inicio, cuando termine, inicie sólo un programa que se ocupe de la seguridad e integridad de la aplicación y ejecución de todos los demás programas de aplicación. Este programa normalmente se denomina programa SHELL. Para obtener más información sobre el diseño de un programa SHELL, consulte la sección 6.2.1.

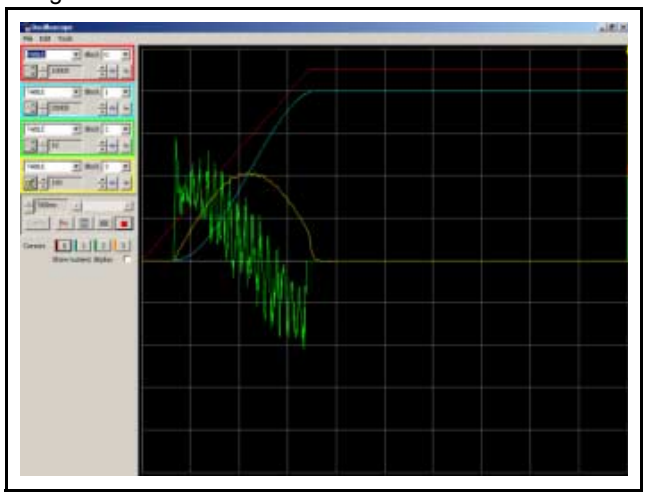
Suponga que el programa está diseñado de un modo que contiene la siguiente fracción de código:

```
'Cuando hay un error, se paran todos los programas.
'No se efectúan nuevas capturas de osciloscopio.
'Y hemos almacenado en las tablas seleccionadas
'el último seguimiento de datos en el que se ha
'producido un error. Por lo tanto, podemos recuperar
'este seguimiento y analizarlo.
lazo:
    IF MOTION_ERROR<>0 THEN HALT
GOTO lazo
```

Este código de programación provoca que todos los programas y el seguimiento se paren cuando se produce un error en un eje. Los datos ya están capturados en la memoria de tabla y podemos comenzar a utilizar el osciloscopio para ver el estado de los parámetros deseados en el momento en que se ha producido el error. Siguiendo el caso descrito anteriormente, con los ajustes de osciloscopio como se indica en fig. 44, el resultado se ofrece en fig. 46.

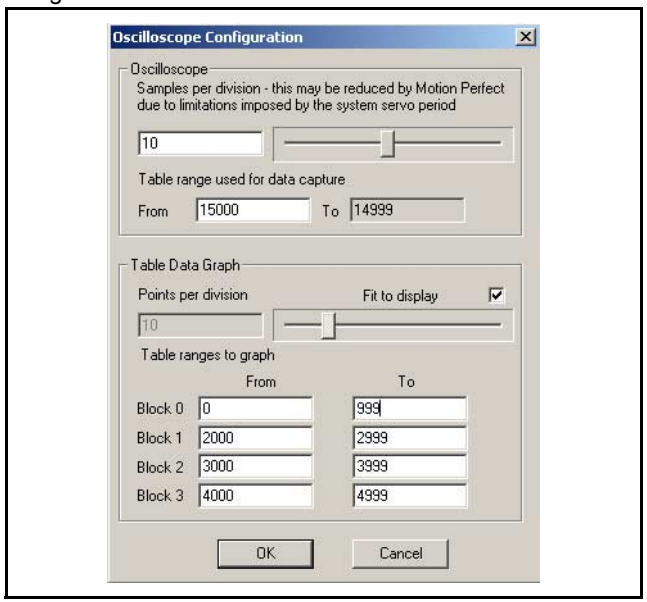
La posición medida del eje maestro, indicada en rojo, no parece ser la causa, porque no hay discontinuidad en ella. También descartamos un problema mecánico, porque el par, que se indica en verde, tiene valores bajos. En el momento del problema, la velocidad del eje esclavo, indicada en amarillo, era suave y baja, por lo tanto, tampoco es el problema.

fig. 46



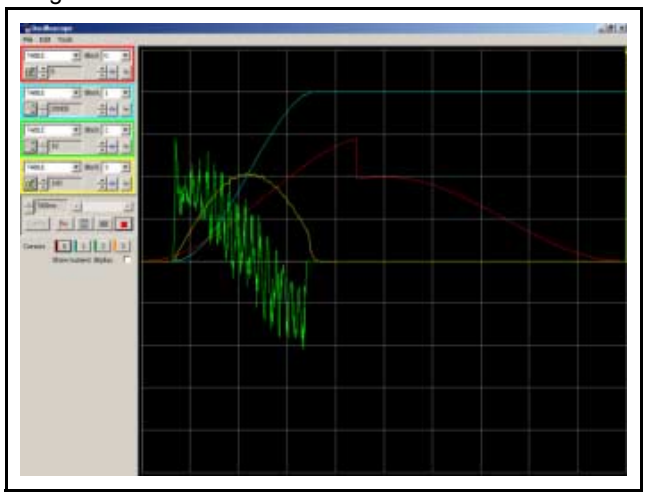
El siguiente paso consiste en analizar la tabla CAM para comprobar los valores que se utilizaron para solicitar la posición del eje esclavo. Para ello, cambiamos la configuración del osciloscopio con el fin de mostrar un bloque de valores de Table(0) a Table(999) en color rojo, porque en estas entradas de la memoria de tabla es donde se crea la tabla CAM (consulte la parte del programa que crea la tabla CAM anterior). La configuración cambiada se muestra en la figura.

fig. 47



El resultado se indica en la figura. El gráfico rojo muestra claramente una discontinuidad en los valores de posición que debe seguir el eje esclavo. Como la velocidad es una derivada de la posición, en el punto de la discontinuidad de la curva de posición la velocidad alcanza un valor alto. En teoría, este valor es igual a infinito, pero en la práctica este valor es muy grande. Esto provoca el error. El gráfico rojo muestra dónde está la raíz del problema. La amplitud de curva de coseno y, por lo tanto, el parámetro **posición_final**, ha cambiado durante la ejecución del comando **CAMBOX**. La solución es simple: se debe evitar un cambio del parámetro **posición_final** durante la ejecución de **CAMBOX**. Para ello, modifique los programas en Trajexia o en otro controlador (si el parámetro se cambia fuera del ámbito de los programas de aplicación, por ejemplo, mediante un mensaje FINS).

fig. 48



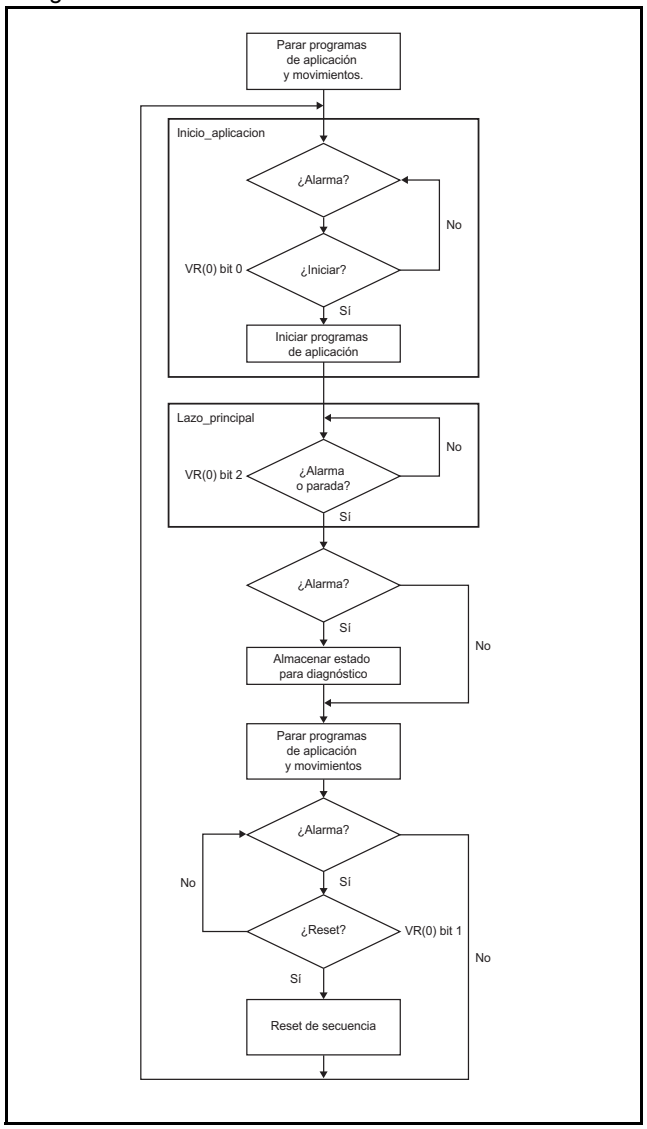
La base temporal de los puntos de la tabla CAM no es la misma que la captura de las demás señales. La discontinuidad en CAM (gráfico rojo) coincide en el tiempo con la interrupción del movimiento. Para analizarlo, compruebe los valores de posición individualmente con un programa de hoja de cálculo. Para analizar los valores de punto en detalle, puede exportar los puntos de tabla a un programa de hoja de cálculo y efectuar un análisis más complejo.

6.2 Ejemplos prácticos

6.2.1 Programa shell

La buena práctica de programación requiere un buen programa de shell. El programa de shell inicia, para y restablece los programas de aplicación. El programa de shell no es necesario, pero ofrece estructura a las aplicaciones y convierte el método de programar el controlador Motion en más efectivo.
A continuación se encuentra un ejemplo de programa de shell. Asegúrese de modificar el programa según las necesidades específicas de la aplicación. Compruebe la operación correcta antes de confiar en la operación segura del programa. Este programa normalmente está configurado para ejecutarse en el encendido a baja prioridad.

fig. 49



Ejemplo

```
'==================================================
'EJEMPLO DE PROGRAMA SHELL
'ESTA VERSIÓN ESTÁ DISEÑADA PARA SERVOS MECHATROLINK
'ADAPTE ESTE PROGRAMA SEGÚN LA APLICACIÓN
'==================================================
'EN ESTE EJEMPLO CUALQUIER ERROR PARA TODOS LOS
'PROGRAMAS Y MOVIMIENTOS
'USAMOS UNA VARIABLE GLOBAL (VR(0)) PARA LA GESTIÓN
'DEL PROGRAMA.
' VR(0) bit 0 Para iniciar la aplicación (flanco ascendente)
' VR(0) bit 1 Para efectuar un RESET de todas las alarmas
' (flanco ascendente)
' VR(0) bit 2 Para parar la aplicación (flanco ascendente)
'==================================================
'Este ejemplo es para una aplicación con tres
'servos
'Ejes 1, 2 y 3
'==================================================

'Inicialización de variables
max_ejes=2
'estado_proyecto
'=0 valor inicial
```

```
'=1 programas parados pero no hay error
'=2 programas parados y error
'=3 programas en ejecución
estado_proyecto=0

'estado_alarma
'=0 nada
'=1 alarma en eje 0
'=2 alarma en eje 1
'=3 alarma en eje 3
'=4 alarma en MECHATROLINK
'=5 correcto
estado_alarma=5

'Acción
'=0 nada
'=1 pulsar reset para reiniciar
'=2 restableciendo
'=3 OK
accion=3

GOSUB parar_todo
GOSUB inicio_aplicacion

lazo:
    'Si comando Error o Stop
    IF MOTION_ERROR<>0 OR READ_BIT(1,0) THEN GOSUB secuencia_alarma

    'Borrar la advertencia de servodriver si la hubiera
    IF (DRIVE_STATUS AXIS(0) AND 2)>0 THEN DRIVE_CLEAR AXIS(0)
    IF (DRIVE_STATUS AXIS(1) AND 2)>0 THEN DRIVE_CLEAR AXIS(1)
    IF (DRIVE_STATUS AXIS(1) AND 2)>0 THEN DRIVE_CLEAR AXIS(1)
GOTO lazo

secuencia_alarma:
    IF MOTION_ERROR<>0 THEN
        estado_proyecto=2
        accion=1
        'DIAGNÓSTICO DE ERRORES
        'Comprobar errores MECHATROLINK
```

```
        IF (AXISSTATUS AXIS(0) AND 4)<>0 THEN
            estado_alarma=4
        ELSEIF (AXISSTATUS AXIS(1) AND 4)<>0 THEN
            estado_alarma=4
        ELSEIF (AXISSTATUS AXIS(1) AND 4)<>0 THEN
            estado_alarma=4
        ELSE
            'Comprobar error de eje
            estado_alarma=ERROR_AXIS+1
        ENDIF
    ELSE
        estado_proyecto=1
    ENDIF
    GOSUB parar_todo
    GOSUB reset_todo
    GOSUB iniciar_aplicacion
RETURN

parar_todo:
    'PARAR LOS PROGRAMAS
    STOP "APPLICATION"

    'PARAR LOS EJES
    FOR i= 0 TO max_ejes
        BASE(i)
        CANCEL(1) 'Cancelar NTYPE
        WA(1)
        CANCEL(1) 'Cancelar posible búfer de programa
        CANCEL 'Cancelar MTYPE
    NEXT i
    'Liberar comando Run en los servos
    WDOG=0
    'Abrir el lazo de posición
    FOR i= 0 TO max_ejes
        BASE(i)
        WAIT IDLE
        SERVO=0
    NEXT i
RETURN
```

```
reset_todo:
    WHILE MOTION_ERROR<>0
        'Esperar flanco ascendente en entrada RESET
        WAIT UNTIL READ_BIT(2,0)=0
        WAIT UNTIL READ_BIT(2,0)=1
        accion=2
        'Reset de servos con fallo
        FOR i=0 TO max_ejes
            BASE(i)
            'En caso de error de ML-II, el reset es:
            IF (AXISSTATUS AND 4)<>0 THEN
                MECHATROLINK(0,0)
                WA(3000)
                DATUM(0)
                RUN "startup",1
                STOP
            ENDIF
            'En caso de error de servodriver
            IF (AXISSTATUS AND 8)<>0 THEN DRIVE_CLEAR
        NEXT i
        WA(100)
        'En caso de error de eje
        DATUM(0)
    WEND
    estado_proyecto=1 'Parado pero sin error
    estado_alarma=5
    accion=3
RETURN

inicio_aplicacion:
    'Esperar flanco ascendente en bit 0 de VR(0)
    WHILE READ_BIT(0,0)=0
        IF MOTION_ERROR<>0 THEN RETURN
    WEND
    WHILE READ_BIT(0,0)=1
        IF MOTION_ERROR<>0 THEN RETURN
    WEND
    RUN "APPLICATION"
    estado_proyecto=3 'Aplicación en ejecución
RETURN
```

6.2.2 Programa de inicialización

El programa de inicialización establece los parámetros de los ejes. Estos parámetros dependen de la resolución del encoder del motor y de la velocidad máxima del motor.

Nota:
Consulte la hoja de especificaciones del servodriver y de motor para obtener esta información.

```
'==================================================
'EJEMPLO DE PROGRAMA DE INICIALIZAICÓN
'ESTA VERSIÓN ESTÁ DISEÑADA PARA SERVOS MECHATROLINK
'ADAPTE ESTE PROGRAMA SEGÚN LA APLICACIÓN
'=================================================
BASE(x)
reinicio=0
relacion_inercia=establecer_relacion_inercia_carga

'----------------------------
'EJEMPLO 1
'Datos de motor SGMAH-01AAA61D-OY
'----------------------------
resolucion_enc=2^13 'encoder de 13 bits
velocidad_max=5000 'velocidad máxima de 5000 r.p.m.

'----------------------------
'EJEMPLO 2
'Datos de motor SGMAH-01A1A61D-OY
'----------------------------
resolucion_enc=2^16 'encoder de 16 bits
velocidad_max=5000 'velocidad máxima de 5000 r.p.m.

'----------------------------
'ESCRIBIR PARÁMETROS EN EL SERVO
'----------------------------
DRIVE_WRITE($103,2,relacion_inercia) 'Escribir relación de
inercia
DRIVE_READ($110,2,10)
IF VR(10)<>$0012 THEN
    DRIVE_WRITE($110,2,$0012,1)
```

```
    'Pn110=0012h (autotuning desactivado)
    reinicio=1
ENDIF
DRIVE_READ($202,2,10)
IF VR(10)<>1 THEN
    DRIVE_WRITE($202,2,1,1)
    'Pn202=1 (numerador de relación de engranaje en el variador; predeterminado = 4)
    reinicio=1
ENDIF
DRIVE_READ($511,2,10)
IF VR(10)<>$6548 THEN
    DRIVE_WRITE($511,2,$6548,1)
    'Pn511 establecer las entradas de registro en el servodriver
    reinicio=1
ENDIF
DRIVE_READ($81E,2,10)
IF VR(10)<>$4321 THEN
    DRIVE_WRITE($81E,2,$4321,1)
    'Pn81E=$4321 Hacer que las entradas digitales del servodriver estén
    'disponibles para leerlas mediante el canal DRIVE_INPUTS
    reinicio=1
ENDIF
IF reinicio=1 THEN DRIVE_RESET

'----------------------------
'Ganancias iniciales para MECHATROLINK_SPEED
'----------------------------
'Por experiencia, este ajuste e un buen punto de partida
P_GAIN=INT(214748.3648*velocidad_max/resolucion_enc)
'Es el valor óptimo. Establézcalo si es necesario
VFF_GAIN=INT(60000*1073741824/resolucion_enc/velocidad_max)

'----------------------------
'Ganancias iniciales para el modo MECHATROLINK_POSITION
'----------------------------
'Cambiar la rigidez (Fn001) según el sistema mecánico
'Cambiar ganancia de realimentación positiva Pn109 si es necesario

'----------------------------
'Parámetro inicial del eje
```

```
'----------------------------
'Si se establece en 1 (y Pn202=Pn203=1) las unidades son contajes de encoder
UNITS=1
'FE teórico, el motor estará en marcha a "velocidad_max"
'sin VFF_GAIN en MECHATROLINK SPEED
FE_LIMIT=1073741824/P_GAIN/UNITS
'SPEED establecido a un 1/3 de velocidad_max
SPEED=(velocidad_max73)*resolucion_enc/60/UNITS
'ACCEL en 200 ms de 0 a "velocidad_max"
ACCEL=SPEED/0.2
'DECEL en 200 ms de "velocidad_max" a 0
DECEL=SPEED/0.2
```

6.2.3 Programa de un solo eje

Es un programa simple que sólo pone en marcha un solo eje.

fig. 50

Ejemplo

```
'GOSUB inicio
BASE(0)
DEFPOS(0)
WA(100)
lazo:
    MOVE(1440)
    WAIT IDLE
    WA(100)
GOTO lazo
```

Las unidades son grados en este ejemplo, por lo tanto:

- Encoder de 13 bits
- Pn202 = 32
- Pn203 = 45
- **UNITS = 32**

El gráfico de la figura es típico para este movimiento punto a punto con aceleración lineal. Tenga en cuenta lo siguiente:

- Durante la aceleración lineal, el gráfico de la posición es parabólico (porque la velocidad es una derivada de la posición).
- Durante la velocidad constante, el gráfico de la posición es recto.
- Durante la deceleración lineal, el gráfico de la posición es parabólico.
- Durante la parada, el gráfico de la posición es constante.
- Cuando se produce un desbordamiento (**MPOS> = REP_DIST**), la posición salta a 0 si **REP_OPTION = 1** o a **-REP_DIST** si **REP_OPTION = 0**.

- El error de seguimiento es proporcional a la velocidad si sólo utiliza ganancia proporcional en el lazo de posición.
- El par (que se indica mediante **DRIVE_MONITOR** como un porcentaje del par nominal del motor cuando se establece **DRIVE_CONTROL = 11**) es proporcional a la aceleración según la fórmula:

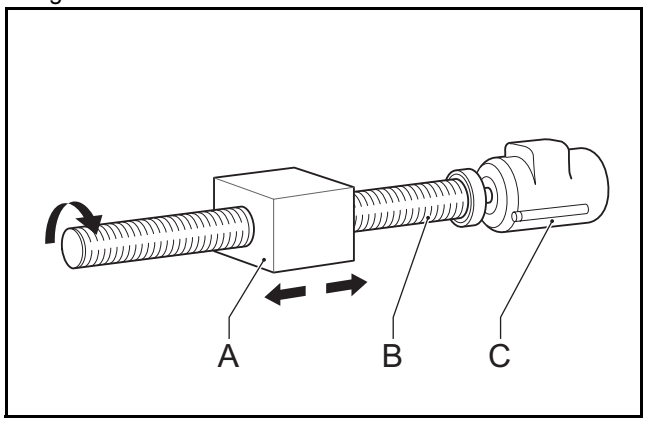
  $Par_{total} = J_{total} \times \alpha + Par_{fricción}$

  donde $Par_{fricción}$ normalmente es un valor pequeño, α es la aceleración angular y J la inercia del sistema.

6.2.4 Posición con detección de producto

Un tornillo móvil avanza a una velocidad de marcha lenta hasta que llega a un producto y se activa un microinterruptor (IN(2)). El tornillo móvil se para inmediatamente, se indica la posición en la que se ha detectado el producto y el tornillo móvil vuelve a velocidad rápida a la posición de inicio.

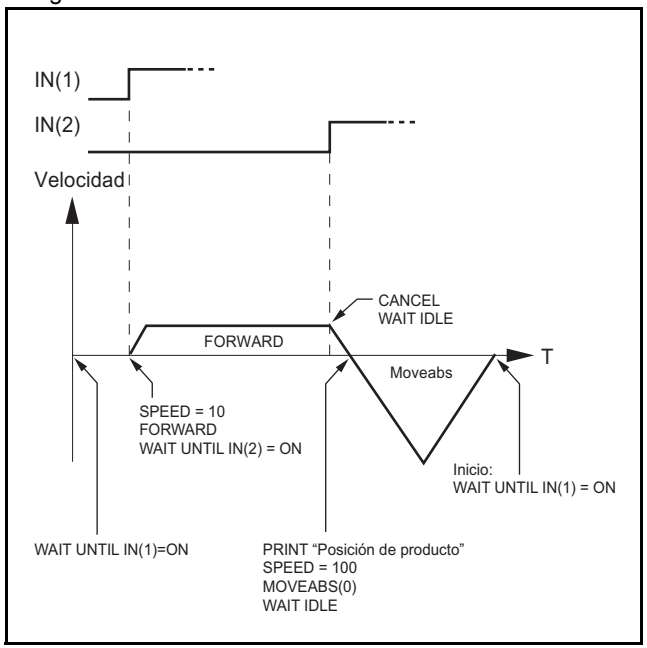
fig. 51



Ejemplo

```
inicio:
    WAIT UNTIL IN(1)=ON
    SPEED=10
    FORWARD
    WAIT UNTIL IN(2)=ON
    pos_prod=MPOS
    CANCEL
    WAIT IDLE
    PRINT "Posición de producto : "; pos_prod
    SPEED=100
    MOVEABS(0)
    WAIT IDLE
GOTO inicio
```

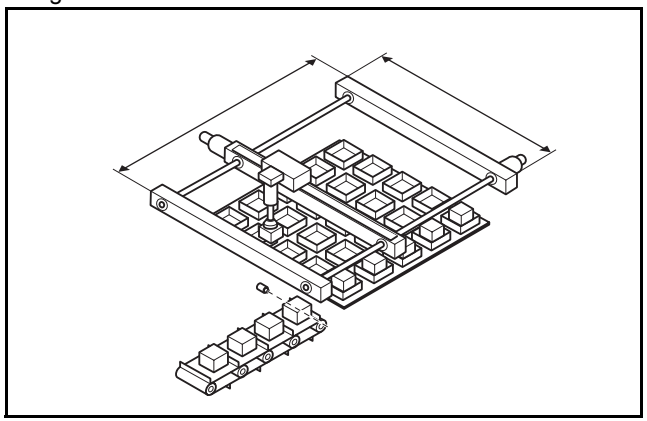
fig. 52



6.2.5 Posición en una cuadrícula

Una paleta cuadrada tiene 1 m de longitud por lado. Se divide en una cuadrícula de 5 x 5 y cada una de las posiciones de la cuadrícula tiene una caja que se tiene que rellenar utilizando el mismo patrón cuadrado de 100 x 100 mm. Una boquilla dispensadora controlada por la salida digital 8 se debe activar al rellenar la caja y desactivar en los demás momentos.

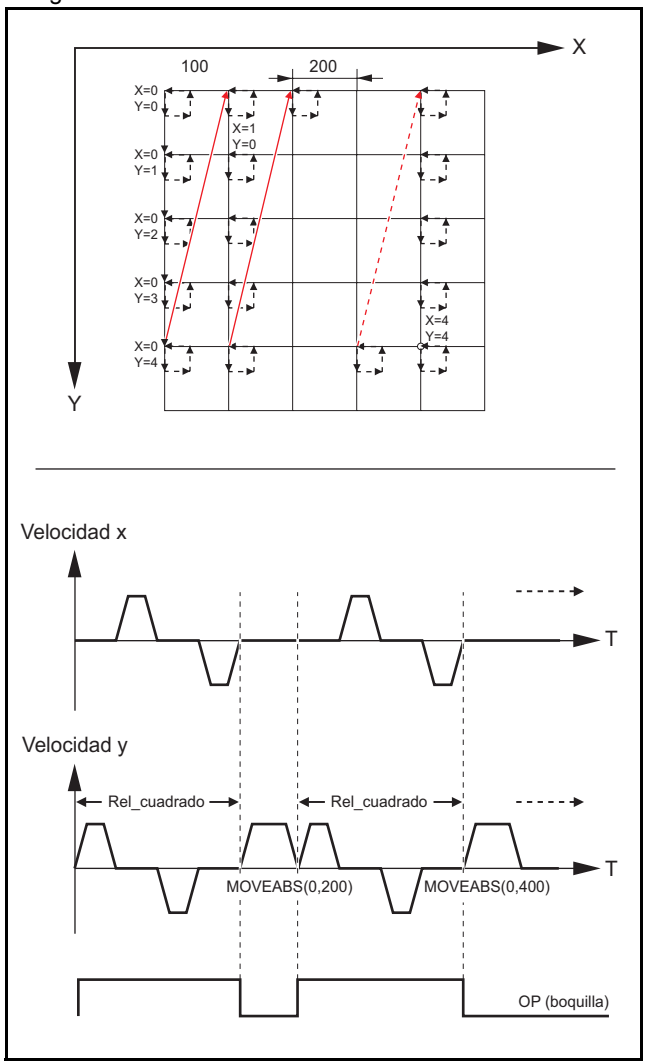
fig. 53



Ejemplo

```
boquilla = 8
inicio:
    FOR x = 0 TO 4
        FOR y = 0 TO 4
            MOVEABS(x*200, y*200)
            WAIT IDLE
            OP(boquilla, ON)
            GOSUB rel_cuadrado
            OP(boquilla,OFF)
        NEXT y
    NEXT x
GOTO inicio
rel_cuadrado:
    MOVE(0, 100)
    MOVE(100, 0)
    MOVE(0, -100)
    MOVE(-100,0)
    WAIT IDLE
    WA(1000)
RETURN
```

fig. 54



6.2.6 Programa de alimentador de bolsas

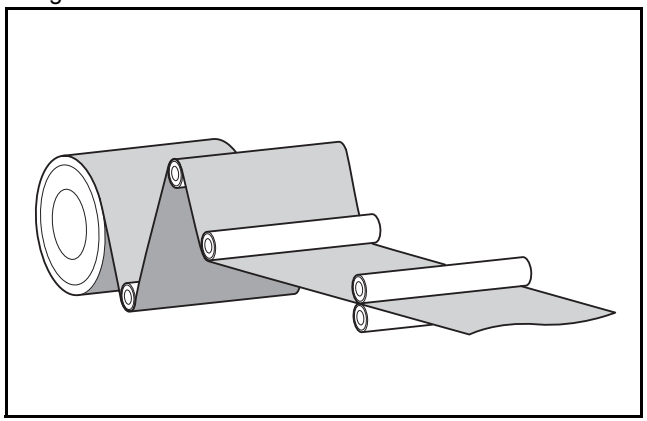
Un alimentador de bolsas alimenta una película de plástico de una longitud fija que está definida por el operario. La figura muestra un alimentador de bolsas típico que forma parte de la máquina.

Los alimentadores de bolsas tienen dos modos de trabajo.

- Sin marca: hace avanzar la película a una distancia definida, para películas de un solo color.
- Con marca: hace avanzar la película hasta una marca impresa en la película.

El programa de esta sección muestra el código típico para un alimentador de bolsas.

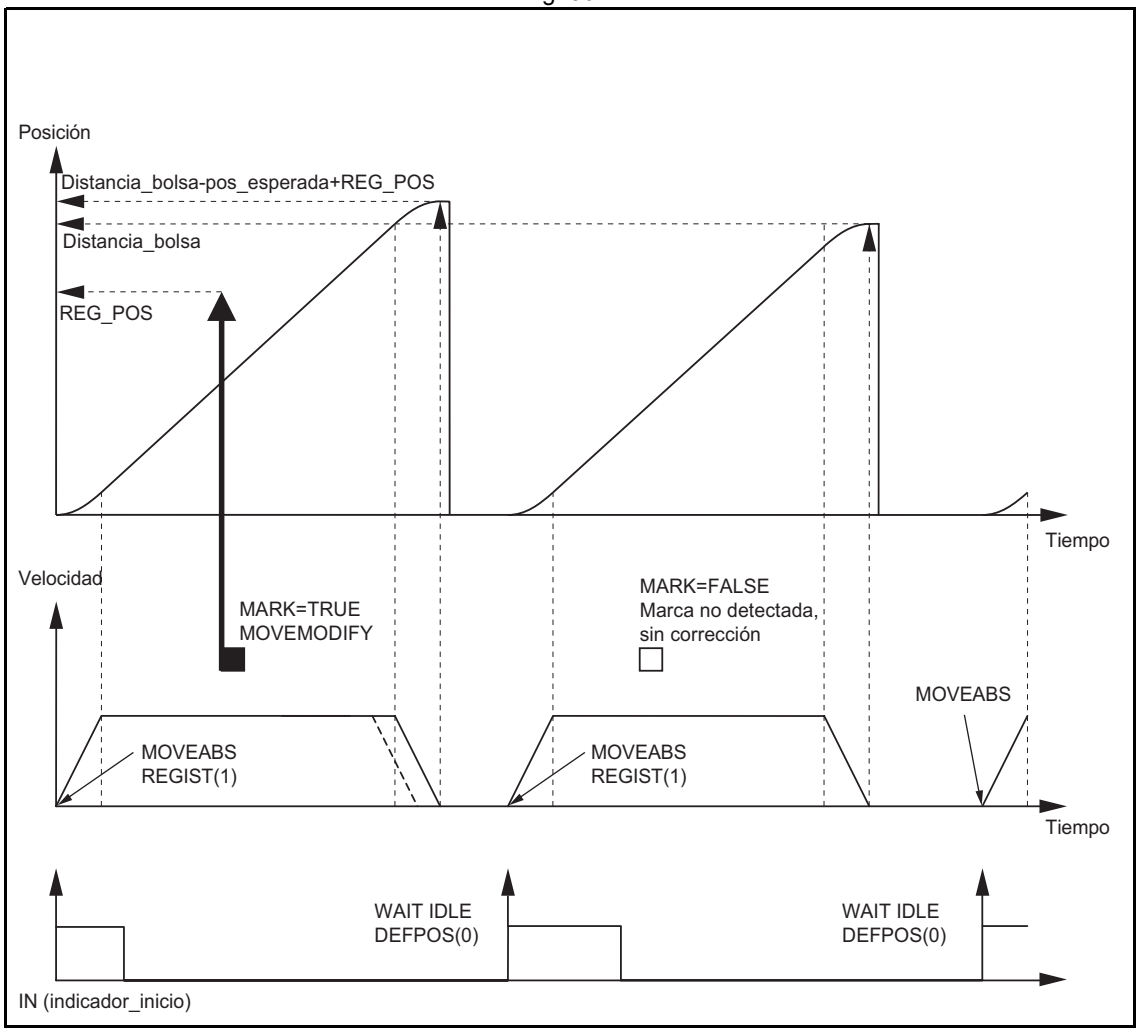
fig. 55



Ejemplo

```
'==================================================
'Programa de ALIMENTADOR DE BOLSAS
'==================================================
'Al trabajar con marcas, si falta alguna marca, cargar
'la distancia teórica. Pero si la marca falta en un número
'de bolsas consecutivas, parar la operación. Se activa
'una salida digital en un determinado tiempo para cortar
'la bolsa.
'==================================================

'Inicialización de variables
indicador_inicio=7
fallos_max=3
alarma_programa=0
fallado=0
eje_alimentador=2
BASE(eje_alimentador)
'El contador de posición (MPOS,DPOS) va de 0 a 999999
'y a 0 de nuevo
UNITS=27
SPEED=100
ACCEL=1000
DECEL=1000
REP_DIST=1000000
REP_OPTION=1
SERVO=ON
WDOG=ON

'Programa principal
lazo:
    'Definir la posición actual como cero
    DEFPOS(0)

    'Esperar flanco ascendente en entrada digital
    '"indicador_inicio"
    WAIT UNTIL IN(indicador_inicio)=0
    WAIT UNTIL IN(indicador_inicio)=1
```

fig. 56



```
    'Mover longitud de bolsa
    MOVEABS(distancia_bolsa)
    WAIT UNTIL MTYPE=2 'Para verificar que MOVEABS se está ejecutando

    'Si trabajamos con marca, activar el activador
    'MARK=FALSE si se ha activado y TRUE si no se ha activado
    IF trabajar_con_marca AND MARK THEN
        REGIST(1)
        WAIT UNTIL MARK=0
    ENDIF

    'Esperar hasta que finalice el movimiento o se detecte la marca
    WAIT UNTIL MTYPE=0 OR (MARK AND trabajar_con_marca)

    'Trabajar con marca
    IF trabajar_con_marca THEN
        IF MARK THEN 'Si se ha detectado la marca, la posición se corrige
            MOVEMODIFY(distancia_bolsa-pos_esperada+REG_POS)
            fallado=0

        ELSE 'Si la marca no se ha detectado
            PRINT "Marca no detectada"
            fallado=fallado+1
            IF fallado>fallos_max THEN 'Después de varias detecciones incorrectas consecutivas parar la aplicación
                PRINT "Marca perdida definitivamente"
                alarma_programa=3
                STOP
            ENDIF
        ENDIF
    ENDIF

    'Esperar hasta que haya terminado el movimiento de alimentación
    WAIT IDLE
GOTO lazo
```

6.2.7 Tabla CAM en un programa

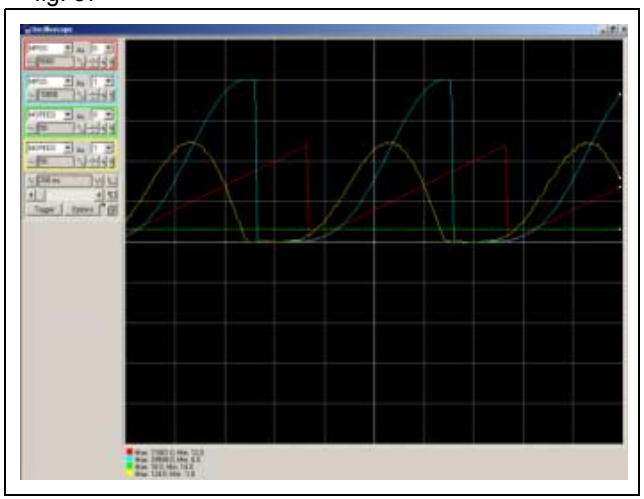
Muestra cómo crear una tabla CAM dentro de un programa y utilizar el comando Motion Control **CAMBOX**.
El perfil utilizado es el de coseno de uno al cuadrado. Se trata de un perfil bastante típico para aplicaciones de tipo alimentador como:

- El movimiento proporciona una aceleración suave sin cambios de aceleración repentinos, por lo que se reduce el deslizamiento de material.
- Ofrece una deceleración rápida por lo que se reduce el tiempo de ciclo. Durante la deceleración no hay deslizamiento de material y la fricción contribuye a pararse a cero.

Ejemplo

```
inicio:
    GOSUB rellenartabla
    WDOG=1 'Establecer servos en RUN
    BASE(1)
    SERVO=1 'Activar lazo de posición en eje 1
    BASE(0)
    SERVO=1 'Activar lazo de posición en eje 0
    'El contador de posición cuenta de 0 a 11999
    'y vuelve a 0 otra vez
    REP_OPTION=1
    REP_DIST=12000
    SPEED=200
    FORWARD

BASE(1)
lazo:
   CAMBOX(en_tbl,fin_tbl,1,dist_vinculo,maestro,salida,inicio)
   WAIT IDLE
GOTO lazo

rellenartabla:
    'La forma de CAM se almacena en TABLE(0) a
    'TABLE(360)
    npuntos=360
    en_tbl=0
    fin_tbl=en_tbl+npuntos
    'Distancia del maestro para realizar CAM
    dist_vinculo=10000
    'Eje maestro
    maestro=0
    'CAM empieza exactamente cuando el maestro llega
    'a la posición "inicio"
    salida=2
    inicio=1000
```

fig. 57



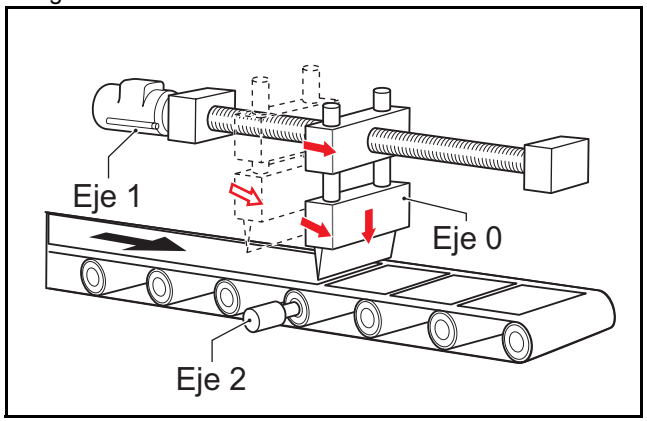
```
    k=100
    'Rellenar TABLE con la forma de onda adecuada
    FOR i= en_tbl TO fin_tbl
        TABLE(i,(k*(COS(PI*i/npuntos)-1))^2)
    NEXT i
RETURN
```

6.2.8 Programa de sierra pendular

Ejemplo del programa de sierra pendular. En esta aplicación hay tres ejes:

- Eje 0, eje_sierra, el avance de la sierra.
- Eje 1, eje_pendular, es la sierra pendular.
- Eje 2, eje_línea, transporta el material.

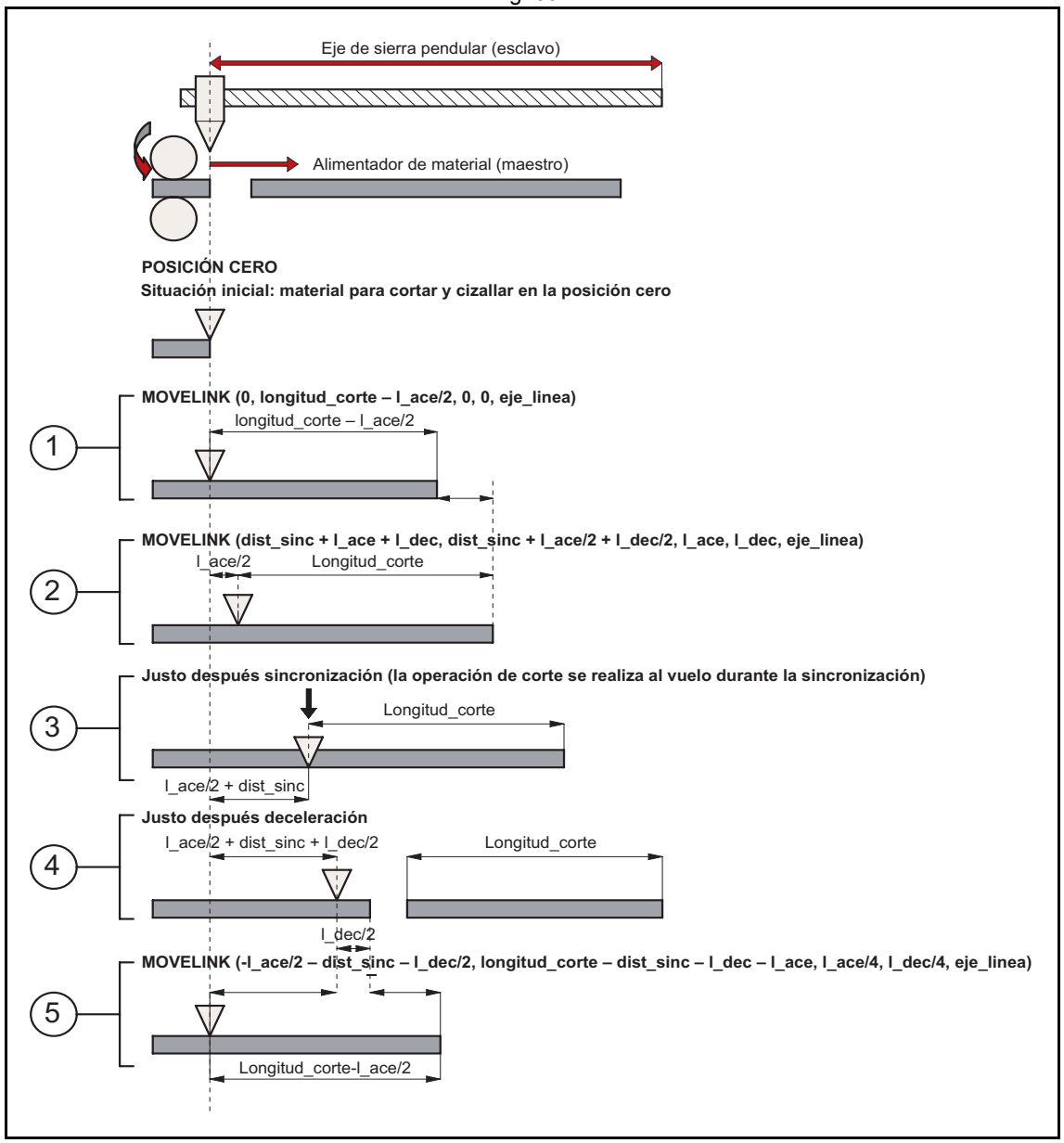
fig. 58



Ejemplo

```
'==================================================
'Programa de SIERRA PENDULAR
'==================================================
'Ejemplo típico de una aplicación de sierra pendular.
'Un eje (eje_línea) transporta el material
'El segundo eje (eje_pendular) es la sierra pendular
'El tercer eje (eje_sierra) es el avanza de corte
'La distancia de sincronización debe ser lo suficientemente
'larga para permitir el corte a la máxima velocidad.
'El retorno de la sierra pendular se realiza a una
'velocidad tal que el tiempo de espera es cero
'(optimización del movimiento).
'De nuevo se supone que todo se ha calculado para no superar
'la velocidad de motor máxima a velocidad de línea máxima
'==================================================
contador_corte=0
eje_linea=2
eje_sierra=0
eje_pendular=1

SERVO AXIS(eje_linea)=ON
SERVO AXIS(eje_sierra)=ON
SERVO AXIS(eje_pendular)=ON
WDOG=ON

'PRIMER CICLO

'Realizar un primer corte de material
MOVEABS(posicion_final) AXIS(eje_sierra)
WAIT UNTIL MTYPE AXIS(eje_sierra)=2
WAIT IDLE AXIS(eje_sierra)

'La primera vez hay que esperar un tiempo porque el
'material se acaba de cortar
distancia_espera=longitud_corte-l_ace/2
MOVELINK(0,distancia_espera,0,0,eje_linea)
AXIS(eje_pendular)
WAIT UNTIL MTYPE AXIS(eje_pendular)=22
```

fig. 59



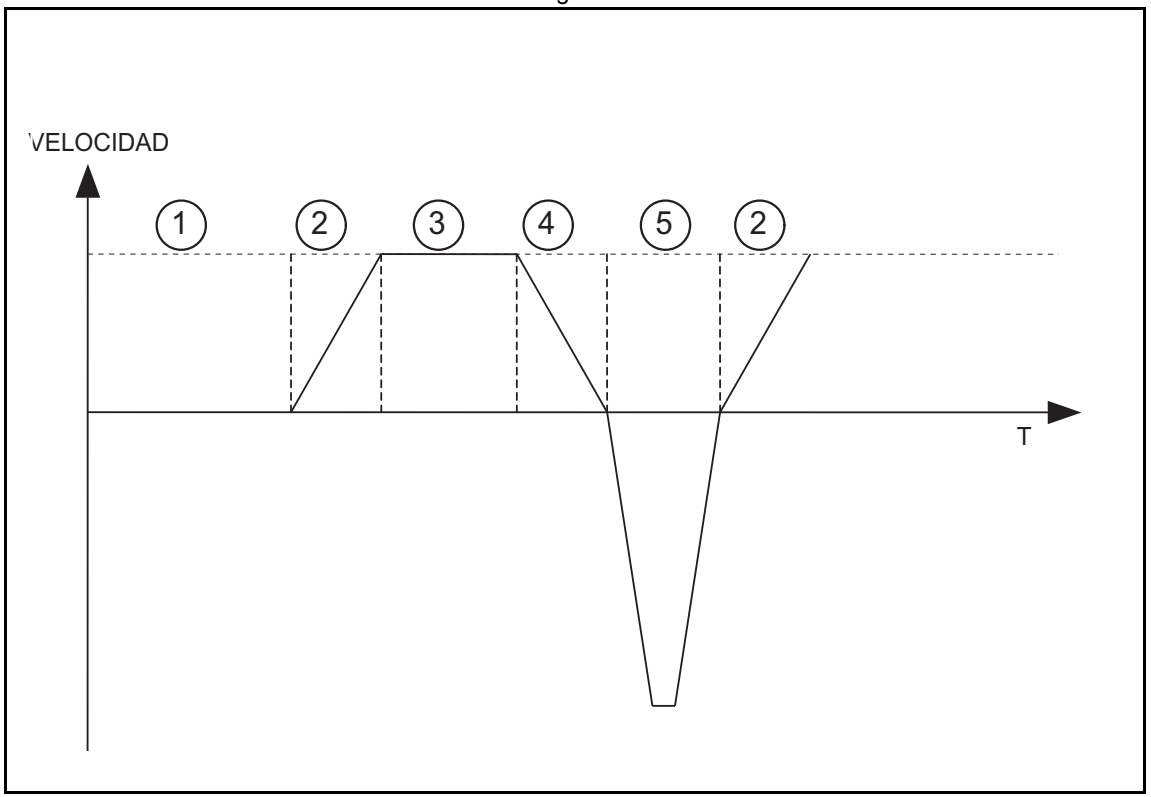
```
'Empezamos la línea
FORWARD AXIS(eje_linea)

lazo:

    'Actualizar la velocidad de línea cada ciclo
    SPEED AXIS(eje_linea)=velocidad_linea

    'Movimiento de corte a velocidad sincronizada
    corte_linea=dist_sinc+l_ace+l_dec
    corte_sierra=dist_sinc+l_ace/2+l_dec/2
    MOVELINK(corte_sierra,corte_linea,l_ace,l_dec,eje_linea) AXIS(eje_pendular)
    WAIT UNTIL MPOS AXIS(eje_pendular)>l_ace/2

    'Activar la sierra cuando está en sincronización con la línea
    'Reducir velocidad para cortar
    SPEED AXIS(eje_sierra)=velocidad_corte
    MOVEABS(posicion_final) AXIS(eje_sierra)
    MOVEABS(0) AXIS(eje_sierra)
    WAIT UNTIL NTYPE AXIS(eje_sierra)=2
    'Velocidad rápida para volver
    WAIT LOADED AXIS(eje_sierra)
    SPEED AXIS(eje_sierra)=velocidad_retorno

    contador_corte=contador_corte+1pulgada

    'Volver sincronizado con el maestro de modo que
    'no haya tiempo de espera
    atras_linea=longitud_corte-dist_sinc-l_dec-l_ace
    corte_sierra=l_ace/2+dist_sinc+l_dec/2
    MOVELINK(-corte_sierra,atras_linea,l_ace/4,l_dec/4,eje_linea) AXIS(eje_pendular)

GOTO lazo
```

El gráfico de velocidad-tiempo muestra los pasos del ejemplo anterior. Los pasos son:

1. El ciclo inicial: el esclavo espera la longitud adecuada en el producto para cortar (longitud_corte – distancia_para_acelerar/2). Es necesario dividir distancia_para_acelerar cuando utilizamos el comando **MOVELINK**, porque al realizar la sincronización el maestro mueve el doble de la distancia del esclavo.
2. El esclavo acelera para sincronizarse con el maestro. Cuando la aceleración finaliza, la distancia relativa entre el borde del producto y la sierra es longitud_corte.
3. Ésta es la parte de sincronización: la distancia relativa entre el borde del producto y la sierra permanece igual. Se realiza el corte en el material. Esto ofrece un nuevo borde de material.
4. La parte de deceleración: el material continúa y la sierra se para.
5. Volver a alta velocidad: las distancias se calculan de modo que cuando el esclavo llega a su posición original, el borde del producto está en la posición correcta para iniciar un nuevo corte.

Se inicia un nuevo movimiento (paso 2).

fig. 60



6.2.9 Programa de corrección

Esta aplicación es para una etiquetadora rotativa. Las constantes son:

- El producto llega en una cinta transportadora (eje maestro) que funciona a velocidad constante.
- Una etiquetadora rotativa que está totalmente sincronizada con la cinta transportadora pega las etiquetas.
- La distancia entre los productos es fija y está garantizada mecánicamente.

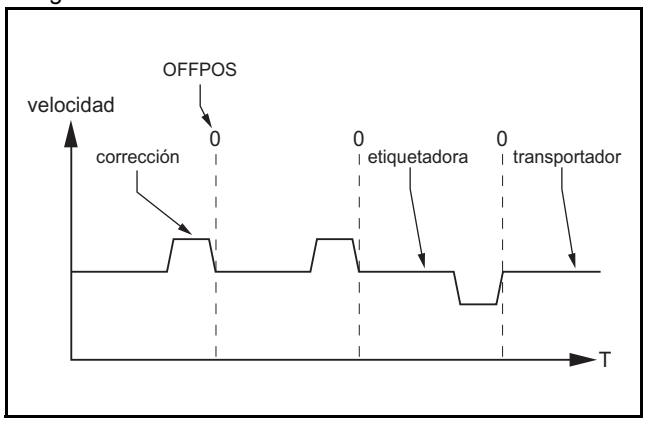
La distancia entre las etiquetas nunca es constante de forma exacta, por lo que es necesaria una corrección. Esto se lleva a cabo mediante la superposición de un eje virtual en el movimiento de la etiquetadora.

La diferencia entre la posición esperada y la posición real se mide con una fotocélula. Éste es el factor de corrección. Cada vez que se efectúa una corrección, la posición de origen se actualiza en consecuencia.

Ejemplo

```
transportador=0
etiquetadora=1
virtual=15
SERVO AXIS(transportador)=1
SERVO AXIS(etiquetadora)=1
WDOG=1

BASE(etiquetadora)
CONNECT(1,transportador)
ADDAX(virtual)
FORWARD AXIS(transportador)
REGIST(1)
WAIT UNTIL MARK=0

lazo:
    WAIT UNTIL MARK
    correccion=REG_POS+pos_esperada
    MOVE(correccion) AXIS(virtual)
    WAIT IDLE AXIS(virtual)
    OFFPOS=-longitud_etiqueta+correccion
    REGIST(1)
    WAIT UNTIL MARK=0
GOTO lazo
```

fig. 61



7 Detección y corrección de errores

7.1 Tensión y herramientas de análisis

Compruebe la tensión de los terminales de entrada de alimentación. Asegúrese de que la tensión está dentro del rango especificado. Si la tensión está fuera del rango especificado, el sistema puede operar de forma incorrecta.

Para diagnosticar errores para TJ1-MC__ y TJ1-ML__ y corregirlos en estas unidades, utilice la herramienta de programación Trajexia Tools. Para diagnosticar errores para TJ1-PRT y corregirlos en esta unidad, utilice un configurador PROFIBUS y una herramienta de monitorización (por ejemplo, **OMRON CX-PROFIBUS**).

Precaución
Desconecte todos los cables antes de comprobar si se han quemado. Incluso si ha comprobado la conducción del cableado todavía existe cierto riesgo, debido al circuito de retorno.

Precaución
Existe la posibilidad de que se descontrole la velocidad del servomotor o que se produzca un error si se pierde la señal del encoder. Asegúrese de desconectar el motor del sistema mecánico antes de comprobar la señal del encoder.

Precaución
Cuando esté detectando y corrigiendo errores, compruebe que no hay nadie dentro de las instalaciones y que éstas no se verán dañadas incluso si el servomotor sufre un exceso de velocidad. Compruebe que puede parar la máquina inmediatamente con una parada de emergencia cuando se descontrola la velocidad del servomotor.

7.2 TJ1-MC__

7.2.1 Errores de sistema

Los errores de sistema se muestran en el display de LEDs de la TJ1-MC__ como **Enn**, donde **nn** es el código de error.

Código de error	Descripción	Causa	Solución
E00	Error de la BASIC SRAM	Fallo de hardware de la TJ1-MC__.	Sustituya la TJ1-MC__.
E01	Error del canal bajo de SRAM del sistema	Fallo de hardware de la TJ1-MC__.	Sustituya la TJ1-MC__.
E02	Error del canal alto de SRAM del sistema	Fallo de hardware de la TJ1-MC__.	Sustituya la TJ1-MC__.
E03	Error de descarga de batería	La tensión de la batería es demasiado baja.	Sustituya la batería.
...	Fallo de hardware	Fallo de hardware de la TJ1-MC__.	Sustituya la TJ1-MC__.

Nota:
Consulte la sección 3.2.254 para obtener más información.

7.2.2 Errores de eje

Los errores de eje se muestran en el display de LEDs de la TJ1-MC__ como **Ann**, donde **nn** es el número del eje que ha provocado el error. Existen dos causas posibles:

- Valor incorrecto o fuera de rango del parámetro de eje configurado.
- Error o alarma en servodriver asignado al eje.

Las dos causas con sus soluciones son las siguientes:

- Valor de parámetro de eje incorrecto o fuera de rango.
- Error o alarma en servodriver asignado al eje.

Valor de parámetro de eje incorrecto o fuera de rango.

Si el valor de un parámetro de eje es incorrecto o está fuera de rango, se produce un error de eje. En el display del servodriver asignado al eje no se muestra ninguna alarma ni error.
Puede ver la causa del error con el comando **AXISSTATUS**. En la ventana de terminal de Trajexia Tools, escriba **PRINT AXISSTATUS AXIS(nn)**, donde **nn** es el número de eje. El valor de devolución del comando **AXISSTATUS** contiene el código de error de eje. Consulte el comando **AXISSTATUS**.
También puede abrir la ventana **Axis Parameter** (Parámetro de eje) en Trajexia Tools y comprobar el campo **AXISSTATUS** del eje que ha provocado el error. Los bits que indican la causa del error se muestran en letrada grandes de color rojo. Para eliminar el error, siga estos pasos:

1. Corrija el valor.
2. Efectúe un reset del controlador o haga clic en el botón **Axis status error** (Error de estado de eje).

Error o alarma en servodriver asignado al eje

Si un error o una alarma del servodriver asignado al eje provoca un error de eje, la alarma del servodrive se muestra en el display de LEDs del servodrive.
También puede abrir la ventana Axis Parameter (Parámetro de eje) en Trajexia Tools y comprobar el campo **AXISSTATUS** del eje que ha provocado el error.
El valor de devolución del comando **AXISSTATUS** tiene el segundo bit (bit **a**: error de comunicaciones de servodriver) y/o el tercer bit (bit **m**: alarma de servodriver) presentados en grandes letras rojas.
Para eliminar el error, siga estos pasos:

1. Consulte el manual del servodriver para determinar la causa del error y soluciónelo.
2. Efectúe un reset del controlador o haga clic en el botón **Axis status error** (Error de estado de eje).

7.2.3 Errores de unidad

Los errores de unidad se muestran en el display de LEDs de la TJ1-MC__ como **Unn**, donde **nn** es el número de la unidad que ha provocado el error.
Existen cuatro causas posibles:

- Unidad defectuosa.
- Unidad no conectada al bus Trajexia.
- Una unidad de E/S o un variador en una unidad MECHATROLINK-II se ha perdido o se ha desconectado.
- Sin terminación.

Unidad defectuosa

El código de error **U0n** se muestra en el display, donde **n** va de 0 a 6 y es el número de la unidad que provoca el error.
Para solucionar el problema, sustituya la unidad defectuosa.

Unidad no conectada al bus Trajexia

El código de error **U0n** se muestra en el display, donde **n** va de 0 a 6 y es el número de la unidad que provoca el error.
Para solucionar el problema, compruebe el conector de bus de la unidad.

Una unidad de E/S o un variador en una unidad MECHATROLINK-II se ha perdido o se ha desconectado

El código de error **U0n** se muestra en el display, donde **n** es el número de la TJ1-MC__ a la que está conectada la unidad MECHATROLINK-II que ha provocado el error.

Puede configurar los indicadores del sistema para activar y desactivar estos errores. Los errores están activados de forma predeterminada.
Para desactivar los errores, escriba **COORDINATOR_DATA(7,1)** en la ventana de terminal de Trajexia Tools.

Para activar los errores, escriba **COORDINATOR_DATA(7,0)** en la ventana de terminal de Trajexia Tools.

Para ver el ajuste actual, escriba **PRINT COORDINATOR_DATA(7)** en la ventana de terminal de Trajexia Tools.

Para borrar el error después de la reparación, realice estos pasos:

- Vuelva a conectar la unidad de E/S o variador MECHATROLINK-II que se ha perdido.
- Escriba MECHATROLINK(n, 5, estación, –1) en la ventana de terminal de Trajexia Tools:
  donde **n** es el número de la TJ1-ML__ a la que está conectada la unidad MECHATROLINK-II afectada y **estación** es el dispositivo MECHATROLINK-II que se ha perdido.

Si desea utilizar el sistema sin el dispositivo perdido, puede volver a conectar todos los dispositivos disponibles en la TJ1-ML__. Para ello, escriba **MECHATROLINK(n, 0)** en la ventana de terminal de Trajexia Tools, donde **n** es el número de la TJ1-ML__ que informa del error.

Sin terminación

En el display se muestra el código de error **U07**.
Para solucionar el problema, compruebe la conexión de terminación o sustituya la terminación si está defectuosa.

7.2.4 Errores de configuración

Los errores de configuración se muestran en el display de LEDs de la TJ1-MC__ como **Cnn**, donde **nn** es el número de la unidad que ha provocado el error.
Las causas de los errores de configuración son:

- El sistema tiene demasiadas unidades del mismo tipo y no cumple las reglas para añadir unidades a un sistema.
- Ha conectado numerosas estaciones MECHATROLINK-II a la TJ1-ML__.
- Existen demasiados ejes en el sistema.
- Hay demasiadas estaciones MECHATROLINK-II que no son ejes en el sistema.

Para solucionar este problema, cambie el sistema de modo que cumpla las reglas para añadir unidades a un sistema. Consulte el manual de referencia de hardware.

7.2.5 Sustituya la batería

Para sustituir la batería de respaldo, siga estos pasos:

1. Asegúrese de que la unidad de fuente de alimentación está conectada durante cinco minutos como mínimo. Si no es así, el condensador que sirve de respaldo para la memoria de la TJ1-MC__ mientras la batería no está conectada no se ha cargado por completo y puede perder los datos de la memoria.
2. Tire de la parte superior de la tapa del compartimento de la batería para abrirlo.
3. Saque los cables rojo y blanco para extraer la batería antigua.
4. Asegúrese de realizar los 2 pasos siguientes en 30 segundos para evitar la pérdida de datos de la memoria RAM.
5. Desconecte los cables de la batería antigua.
6. Conecte los cables a la batería nueva.
7. Inserte la batería nueva en el compartimento correspondiente.
8. Cierre la tapa del compartimento de la batería.

7.3 TJ1-PRT

7.3.1 Errores de sistema

Indicación	Problema	Solución
Ningún LED está encendido o parpadeando	La alimentación está desconectada.	Conectar la alimentación.
	La TJ1-PRT es defectuosa.	Sustituya la TJ1-PRT.
El LED ERH está encendido	Fallo de comunicación entre la TJ1-MC__ y la TJ1-PRT.	Efectúe un reset de la TJ1-MC__. Si esto no funciona, sustituya la TJ1-MC__.
El LED ERC está encendido	Error de unidad. La TJ1-PRT es defectuosa.	Sustituya la TJ1-PRT.

7.3.2 Problemas de comunicaciones de datos de E/S

Indicación	Problema	Solución
El LED COMM está apagado y el LED BF está encendido	La configuración PROFIBUS es incorrecta y no hay comunicación con el maestro.	• Compruebe que la TJ1-PRT tiene la misma dirección de estación que en la configuración del maestro. • Compruebe que ninguna dirección de estación se utiliza dos veces.
	El cableado de PROFIBUS no es correcto.	• Compruebe que está conectados los pines correctos del conector CN1. • Compruebe que no hay cortocircuitos ni interrupción de línea. • Compruebe que utiliza el tipo de cable correcto. • Compruebe que las líneas de conexión no sean demasiado largas.
	No ha efectuado una terminación correcta de la red PROFIBUS.	Lleve a cabo la terminación de la red PROFIBUS en los lugares adecuados.
	La unidad maestra PROFIBUS es defectuosa.	Sustituya la unidad maestra.
	La TJ1-PRT es defectuosa.	Sustituya la TJ1-PRT.

Indicación	Problema	Solución
El LED COMM está apagado y el LED BF está parpadeando	La configuración PROFIBUS es incorrecta y no hay comunicación con el maestro.	• Compruebe que utiliza archivo GSD correcto en el maestro. • Comprueba la configuración y los datos de parámetro del esclavo. • Compruebe que la red se ha configurado para comunicarse a la velocidad en baudios que admite la TJ1-PRT.
	No ha seleccionado datos de configuración para el esclavo.	Compruebe la configuración en el maestro.
	La TJ1-PRT es defectuosa.	Sustituya la TJ1-PRT.

7.4 TJ1-DRT

7.4.1 Errores de sistema

Indicación	Problema	Solución
Ningún LED está encendido o parpadeando	La alimentación está desconectada.	Conectar la alimentación.
	La TJ1-DRT es defectuosa.	Sustituya la TJ1-DRT.
El LED ERH está encendido	Fallo de comunicación entre la TJ1-MC__ y la TJ1-DRT.	Efectúe un reset de la TJ1-MC__. Si esto no funciona, sustituya la TJ1-MC__.
El LED ERC está encendido	Error de unidad. La TJ1-DRT es defectuosa.	Sustituya la TJ1-DRT.

7.4.2 Problemas de comunicaciones de datos de E/S

Indicación	Problema	Solución
NOK parpadea y el LED NF está apagado	El maestro DeviceNet no se está comunicando con la TJ1-DRT.	• Configurar e iniciar el maestro DeviceNet.
NOK está apagado y el LED NF está encendido	Error de duplicación de dirección de nodo.	• Comprobar la dirección de nodo.
	Error de cable de red.	• Compruebe los cables de red.

7.5 TJ1-ML__

7.5.1 Errores de sistema

Indicación	Problema	Solución
Todos los LEDs están apagados	La alimentación está desconectada.	Conectar la alimentación.
	La TJ1-ML__ es defectuosa.	Sustituya la TJ1-ML__.

7.5.2 Error de bus

Indicación	Problema	Solución
El LED BF está encendido	Fallo de cable en el bus MECHATROLINK-II.	Compruebe si en los cables MECHATROLINK-II entre las estaciones conectadas a la unidades hay interrupciones o irregularidades (cortocircuito entre las líneas de comunicaciones A y B, cortocircuito de cualquier línea de comunicaciones con pantalla).
	Falta la terminación de MECHATROLINK-II o está dañada.	Instale una terminación de bus MECHATROLINK-II en la última estación de la cadena o sustitúyala.
	La estación MECHATROLINK-II conectada a la unidad se ha perdido debido a un corte de alimentación o a un fallo de interfaz MECHATROLINK-II en la estación.	Compruebe la alimentación y la interfaz MECHATROLINK-II de la estación que ha provocado el problema. Sustituya la estación si fuera necesario.
	La TJ1-ML__ es defectuosa.	Sustituya la TJ1-ML__.

Nota:
Después de eliminar la causa de un error, asegúrese de reinicializar el bus MECHATROLINK-II en la unidad en la que ha aparecido el error. Escriba en la ventana de terminal de Trajexia Tools:
MECHATROLINK(n, 0)
donde **n** es el número de la unidad a que está conectada la unidad que ha provocado el error.

7.6 TJ1-FL02

7.6.1 Errores de sistema

Indicación	Problema	Solución
Todos los LEDs están apagados	La alimentación está desconectada.	Conectar la alimentación.
	La TJ1-FL02 es defectuosa.	Sustituya la TJ1-FL02.
El LED RUN está encendido. El LED A EN o B EN está apagado	El eje para el que el LED EN está apagado no está activado.	Active el eje: realice **WDOG = ON** y/o **AXIS_ENABLE** en el eje.
El LED RUN está encendido. El LED A EN o B EN parpadea	Hay un error en el eje para el que parpadea el LED EN.	La TJ1-MC__ indica el número del eje con un error de eje. Elimine la causa del error de eje y borre el error de eje o bien reinicie el sistema.

B

BASIC
Especificaciones matemáticas 27
Estructuras de datos 25
Variables 25
Batería 302

C

Comando
Comunicaciones 36
Control de programa 38
E/S 36
Eje 33
Programa 37
Sistema 38
Tarea 40
Comandos BASIC 33
Comparación de archivos de programa 192
Comparación de proyecto 192
Comparación de Trajexia 192
Conexión de red 187
Constantes 36
Creación de aplicaciones 191

D

Descripción de las interfaces 153
Descripción de protocolos 153
Descripción del hardware 23
Descripción del sistema 22
Descripciones de menús 196
DEVICENET
Configuración de las comunicaciones 173
Errores 304
Estado de las comunicaciones 178
Interfaz 173

E

Editor VR 219
Eje flexible
  Errores 305
Ejecución de Motion Control 28
Ejemplo
  Ajuste de unidades 239
  Ajustes de ganancia 229
  Búsqueda de origen 253
  Características de servodriver 251
  Modo de posición 235
  Modo de velocidad 230
  Posición con detección de producto 287
  Posición en una cuadrícula 289
  Programa de alimentador de bolsas 291
  Programa de corrección 298
  Programa de inicialización 283
  Programa de inicio 225
  Programa de sierra pendular 295
  Programa de un solo eje 286
  Programa shell 279
  Registro 259
  Retorno al inicio 253
  Seguimiento y monitorización 269
  Tabla CAM 293
Ejemplo de búsqueda de origen 253
Ejemplo de características de servodriver 251
Ejemplo de corrección 298
Ejemplo de ganancia 229
Ejemplo de inicialización 283
Ejemplo de inicio 225
Ejemplo de modo de posición 235
Ejemplo de modo de velocidad 230
Ejemplo de posición con detección de producto 287

Ejemplo de posición en una cuadrícula 289
Ejemplo de programa de alimentador de bolsas 291
Ejemplo de registro 259
Ejemplo de retorno al inicio 253
Ejemplo de seguimiento y monitorización 269
Ejemplo de shell 279
Ejemplo de sierra pendular 295
Ejemplo de tabla CAM 293
Ejemplo de un solo eje 286
Ejemplo de unidades 239
Errores
  Configuración 302
  Eje 300
  TJ1-MC__ 300
  Unidad 301
Estado de E/S 216

F

Función
  E/S 36
  Matemática 37
  Sistema 38

H

Herramienta de programación 180
Host Link
  Comandos BASIC 158
  Protocolo esclavo 163
  Protocolo maestro 158

I

Iconos 195
Instalación del software 181
Intelligent drives (Variadores inteligentes) 208

Interfaz
DEVICENET 173
Ethernet 153
MECHATROLINK 179
PROFIBUS 167
Serie 158
Interfaz de línea de comandos 30
Interfaz PROFIBUS 167
Interfaz serie 158

J

Jog 217

M

MECHATROLINK
Errores 304
Protocolo 179
Menú
Controller (Controlador) 198
Help (Ayuda) 224
Options (Opciones) 221
Program (Programa) 203
Project (Proyecto) 196
Tools (Herramientas) 205
Window (Ventana) 224
Modificador
Entrada 38
Multitarea 23

O

Operando 37
Matemático 37
Oscilloscope (Osciloscopio) 210

P

Parámetro
- Comunicaciones 36
- E/S 36
- Eje 34
- Entrada 38
- Sistema 39
- Tarea 40

PC
- Conexión 186
- Conexión directa 154
- Conexión remota 155
- Especificación 180
- Iniciar Trajexia Tools 187

Precaución, seguridad 16

PROFIBUS
- Configuración de las comunicaciones 167
- Errores 302
- Estado de las comunicaciones 172

Programa STARTUP
- Modificar 208

Programación BASIC 24

Programas BASIC 30

Protocolo
- Cliente FINS 158
- Definido por el usuario 165
- DEVICENET 173
- Esclavo FINS 156
- Esclavo Host Link 163
- Maestro Host Link 158
- MECHATROLINK 179
- PROFIBUS 167
- Trajexia Tools 156

Protocolo de esclavo FINS 156
Protocolo definido por el usuario 165
Protocolo Ethernet 153
Protocolo Trajexia Tools 156

S

Seguridad, entorno de servicio 17
Seguridad, montaje de la unidad 21

V

Ventana de aplicación 194
Visor de tabla 219

Historial de revisiones

El código de revisión manual se muestra como sufijo del número de catálogo en la portada del manual.

Código de revisión	Fecha	Contenido revisado
01	Agosto de 2006	Original
02	Octubre de 2006	Actualización de DeviceNet
03	Mayo de 2007	Actualizado con TJ1-MC04 y TJ1-ML04. Mejora de comandos BASIC, ejemplos de programación y sugerencias.