Volumen

1

AxesBrain

Lenguaje de programación por la automación

# Lenguaje de programación AxesBrain

AxesBrain

# Lenguaje de programación AxesBrain

AB&T Tecnologie Informatiche™
Via dell'About, 2/A • 10015 Ivrea
Fax 0125 234397
www.bausano.net
info@bausano.net

## AxesBrain lenguaje de programación por la automación
## Informaciones legales

Las informaciones con este documento, inclusos URL y las otras referencias sobre el situado Internet, pueden cambiar sin ninguna advertencia.
A menos que específica anotación, las referencias a compañías, organizaciones, producidas, personas y acontecimientos son ficticios y no asociáis con reales compañías, organizaciones, producidas, personas y acontecimientos.
EL AB&T Tecnologías Informaticheä puede registrar, despedir, solicitar el copyright o marcas y reivindicar la propiedad intelectual a todos los argumentos te tratada en este documento.
Sin limitar los derechos bajo copyright, ninguna parte de este documento puede ser reproducida, o modificada o transmitida bajo cada forma o medio, electrónico, meccanico,per fotocopiar, para registrar, u otro, sin la expresa autorización dell Informaticheä.
Salvo que por acuerdos escritos con el AB&T Tecnologías Informaticheä el suministro de este documento no autoriza a nadie a la grabación, a dar licencias, a solicitar el copyright o marcas y a reivindicar la propiedad intelectual a los argumentos tratada en este documento.
AxesBrainä, VisAlgoä, ScadaMERCURIOä son marcas registradas
©AxesBrain, ©VisAlgo tutelado por copyright
ActiveX, DirectX, JScript, Microsoft, Microsoft Press, MS-DOS, Visual Basic, Visual C++, Win32, Win32s, Windows, WDM, Windows NT, Windows 2000 y Windows Yo son producidos u o marcas registradas por el Microsoft Corporation en los Estados Unidos e/o en los otros Países.
Los nombres de compañías y productos emplazados en este documento pueden ser marcas registradas por sus propietarios.
.

# ÍNDICE DE LOS ARGUMENTOS

**1 INTRODUCCIÓN....1-1**

1.1 COMO UTILIZAR ESTE MANUAL....1-1
1.2 LENGUAJE DE PROGRAMACION PROPIETARIO PARA DESCRIBIR CICLOS DE MANIPULACION....1-2

**2 ARQUITECTURA....2-5**

**3 SINTAXIS DEL LENGUAJE....3-1**

3.1 DEFINICIONES DEL LENGUAJE....3-1
3.2 DIMENSIONAS PRINCIPIO DE LOS PARÁMETROS RICHIAMABILI....3-4

**4 VARIABLES GLOBALES Y LOCALES....4-1**

**5 MULTITASK Y ANTICOLLISIONE....5-3**

**6 INSTRUCCIONES LÓGICAS MATEMÁTICAS....6-1**

6.1 LET: LET VALUE ( SET )....6-2
6.2 ADD: ADD VALUE....6-3
6.3 MUL: MULTIPLICATION....6-4
6.4 DIV: DLVISION....6-5
6.5 NEG: NEGATION....6-6
6.6 LBF: LOAD BUFFER....6-7

**7 INSTRUCCIONES DE CONTROL....7-1**

7.1 JMP: JUMP....7-2
7.2 JEQ: JUMP EQUAL....7-3
7.3 JNE: JUMP NOT EQUAL....7-4
7.4 JGT: JUMP GREATER THAN....7-5
7.5 JGE: JUMP GREATER EQUAL....7-5
7.6 JLT: JUMP LESS THAN....7-5
7.7 JLE:JUMP LESS EQUAL....7-5
7.8 JRN: JUMP RANGE....7-6
7.9 JNR: JUMP NOT RANGE....7-7
7.10 JOS: JUMP OR IF BIT SET....7-8
7.11 JOC: JUMP OR IF BIT CLEAR....7-9
7.12 JAS: JUMP AND IF BIT SET....7-10
7.13 JAC: JUMP AND IF BIT CLEAR....7-11
7.14 CAL: CALL....7-12
7.15 RET: RETURN....7-13
7.16 END: END....7-14
7.17 TSK: TASK....7-15
7.18 TKM: TASK MANAGEMENT....7-21
7.19 DIM : DIMENSION....7-24

**8 INSTRUCCIONES DE MOVIMENTAZIONE....8-1**

8.1 HOM: HOMING (OMO)....8-3
8.2 MOV: MOVE....8-5

8.3 CIR: CIRCULAR INTERPOLATION RIGHT CIL: CIRCULAR INTERPOLATION LEFT ..... 8-8
8.4 CRR: CIRCULAR RADIUS RIGHT CRL: CIRCULAR RADIUS LEFT ..... 8-10
8.5 STC: START CONTINUOUS ..... 8-12
8.6 HLC: HALT CONTINUOUS ..... 8-21
8.7 ABC: ABORT CONTINUOUS ..... 8-22
8.8 CAP: CHANGE AXIS PARAMETER ..... 8-23
8.9 HMS: HANDLING MASTER SLAVE ..... 8-30
8.10 HEC: HANDLING ELECTRONIC CAM ..... 8-32
8.11 CFR: CHANGE FEED RATE ..... 8-34
8.12 CPL: CHANGE POSITION LOOP ..... 8-35
8.13 PRD: POSITION READ ..... 8-37
8.14 RAV: READ AXIS VALUE ..... 8-42
8.15 RSV: READ SPEED VALUE ..... 8-49
8.16 SFP: SET FEED PROFILE ..... 8-50
8.17 SPD: SPEED ..... 8-51
8.18 TCH: TOUCH ..... 8-52
8.19 TMT: TEST MOVEMENT TRANSDUCER ..... 8-53
8.20 TMS: TEST MOVEMENT SENSOR ..... 8-54
8.21 TPE: TOUCH PROBE ENABLE ..... 8-55
8.22 SZP: SET ZERO POINT ..... 8-56
8.23 LZP: LOAD ZERO POINTS ..... 8-57
8.24 PIN: POSITION INCREMENTAL(INQ) ..... 8-59
8.25 PAB: POSITION ABSOLUTE(ABQ) ..... 8-60
8.26 MMA: MOVE MANUAL AXIS ..... 8-61
8.27 OPT: OPEN POINT FILE ..... 8-63
8.28 MOR: MORE ..... 8-64
8.29 DCT: DEEP CONTROL TOUCH ..... 8-66
8.30 DCS: DEEP CONTROL SENSOR ..... 8-68
8.31 GRM: GROUP MASCHINE MANAGEMENT ..... 8-72

**9 INSTRUCCIONES POR LAS SEÑALES DE INPUT Y SALIDA ..... 9-73**

9.1 WDI: WAIT DIGITAL INPUT ..... 9-75
9.2 FWI: FAST WAIT DIGITAL INPUT ..... 9-76
9.3 WAI: WAIT ANALOG INPUT ..... 9-77
9.4 AIN: ATTESA INPUT ..... 9-78
9.5 TDI: TEST DIGITAL INPUT ..... 9-79
9.6 TDO: TEST DIGITAL OUTPUT ..... 9-80
9.7 IDI: IF DIGITAL INPUT ..... 9-81
9.8 IDO: IF DIGITAL OUTPUT ..... 9-82
9.9 TAI: TEST ANALOG INPUT ..... 9-83
9.10 SDO: SET DIGITAL OUTPUT ..... 9-84
9.11 SDI: SET DIGITAL INPUT ..... 9-85
9.12 EDO: ENANCED DIGITAL OUTPUT ..... 9-86
9.13 SAO: SET ANALOG OUTPUT ..... 9-87
9.14 GDI: GET DIGITAL INPUT ..... 9-88
9.15 GDO: GET DIGITAL OUTPUT ..... 9-89
9.16 GAI: GET ANALOG INPUT ..... 9-90
9.17 WBD: WRITE BUFFER DIGITAL INPUT ( BPO ) ..... 9-91
9.18 RBD: READ BUFFER DIGITAL INPUT (BPI) ..... 9-97
9.19 CPI: CHANGE PARALLEL INPUT ..... 9-99
9.20 CDI: CHANGE DIGITAL INPUT ..... 9-100
9.21 CDO: CHANGE DIGITAL OUTPUT ..... 9-101

9.22 RDI: RUN DIGITAL INPUT ....9-108
9.23 RDO: RUN DIGITAL OUTPUT ....9-109

**10 INSTRUCCIONES DE SINCRONIZACIÓN ....10-110**

10.1 EVS: EVENT SET ....10-111
10.2 EVC: EVENT CLEAR ....10-112
10.3 EVW: EVENT WAIT ....10-113
10.4 EVG: EVENT GET ....10-114

**11 INSTRUCCIONES DE SERVICIO ....11-1**

11.1 FOC: FILE OPEN CREATE(AZZ) ....11-3
11.2 FWR: FILE WRITE(SCR) ....11-4
11.3 FRD: FILE READ(LEG) ....11-7
11.4 TIM: TIME ....11-13
11.5 TMM: TIME MILLISECOND ....11-14
11.6 SWA: START WATCH ....11-15
11.7 RWA: READ WATCH ....11-16
11.8 HWA: HALT WATCH ....11-17
11.9 CWA: CONTINUE WATCH ....11-18
11.10 KYB: KEYBOARD ....11-19
11.11 DRT: DISPIAY REAL TIME ....11-20
11.12 DIS: DLSPIAY ....11-21
11.13 HLD: HOLD ....11-22
11.14 PWO: POWER ON ....11-23
11.15 EMC: EMERGENCY ....11-24
11.16 LCK: LOCK AND SET EVENT CLIENT ....11-25
11.17 ULK: UNLOCK ....11-26
11.18 RST: RESET ....11-27
11.19 SWN: SHUT DOWN ....11-28
11.20 SOR: SORT ....11-29
11.21 GTK: GET TASK INFORMATION ....11-30
11.22 MDI: EJECUTA UNA INSTRUCCIÓN ISO ....11-31
11.23 OTC: CARGO VA TIRANDO TABLERO ISO ....11-32
11.24 ROP: LEE CAMPO DEL TABLERO ISO ....11-34
11.25 ISO: EJECUTA UN PART PROGRAM ISO ....11-36
11.26 WND: WAIT NOTIFY DETECTED ....11-50
11.27 WKY: WAIT KEYBOARD ....11-52
11.28 NHL: NOHOLD ....11-53
11.29 YHL: YESHOLD ....11-54
11.30 GDT: GET DATE AND TIME ....11-55
11.31 GLN: GET LOCAL NUMBER ....11-56
11.32 GMI: GET MOTION INFORMATION ....11-57
11.33 RTC: READ TIMER AND COUNTER ....11-58
11.34 SGL: SAVE GLOBAL ....11-59
11.35 SHL: SHELL ....11-60
11.36 G80: FINO CICLO FIJO ....11-61
11.37 G81..89 ACTIVA EL CICLO FIJO PRECISADO ....11-62
11.38 ESE EXEC SEQUENZE ( SISTEMA ETEL ) ....11-79
11.39 ERR EXTERNAL READ REGISTRY ( SISTEMA ETEL ) ....11-80
11.40 EWR EXTERNAL WRITE REGISTRY ( SISTEMA ETEL ) ....11-81
11.41 ECM EXTERNAL COMMAND ( SISTEMA ETEL ) ....11-82
11.42 EWS EXTERNAL WAIT SIGNAL ( SISTEMA ETEL ) ....11-83

11.43 CLM COMANDO DA LOGICA DI MACCHINA ( SISTEMA ETEL ) ..... 11-84

**12 INSTRUCCIONES POR LA INTEGRACIÓN CON LOS DEMÁS ENTORNOS ..... 12-1**

12.1 ARI: AMBIENT REQUEST INSTRUCTION ..... 12-2
12.2 SEC: SET EVENT CLIENT ..... 12-6

**13 INSTRUCCIONES DE COMUNICACION ..... 13-1**

13.1 CSO: CONNECT SOCKET ..... 13-2
13.2 LSO: LISTENSOCKET ..... 13-3
13.3 RSO: READSOCKET ..... 13-4
13.4 TSO: WRITESOCKET ..... 13-5
13.5 DSO: DESTROYSOCKET ..... 13-6
13.6 FSO: FREESOCKET ..... 13-7
13.7 GSO: GET INFORMATION SOCKET ..... 13-8
13.8 OSL: OPEN SERIAL LINE ..... 13-10
13.9 RXL: RECEIVE SERIAL LINE ..... 13-11
13.10 TXL: TRASMIT SERIAL LINE ..... 13-12
13.11 CSL: TRASMIT SERIAL LINE ..... 13-13
13.12 FSL: FREESERIAL LINE ..... 13-14

**14 INSTRUCCIONES AWL PLC ..... 14-16**

14.1 NET: PLC NETWORK ..... 14-21
14.2 LD: PLC CARGO OPERACION EN EL STACK ..... 14-22
14.3 LDN: PLC CARGO OPERACIÓN NEGADA EN EL STACK ..... 14-23
14.4 A: PLC COMBINA EL VALOR DE BIT POR AND ..... 14-24
14.5 AN: PLC COMBINA EL VALOR DE BIT NEGADO POR AND ..... 14-25
14.6 O: PLC COMBINA EL VALOR DE BIT POR OR ..... 14-26
14.7 ON: PLC COMBINA EL VALOR DE BIT NEGADO POR OR ..... 14-27
14.8 EU: PLC NOTA FRENTE POSITIVA ..... 14-28
14.9 ED: PLC NOTA FRENTE POSITIVA ..... 14-29
14.10 EQU: PLC ASIGNADO, COPIA EN EL PARÁMETRO PRECISADO EL VALOR SUPERIOR DEL STACK ..... 14-30
14.11 S: PLC IMPUESTO A 1 EL NÚMERO DE PUNTOS PRECISADO SI EL STACK ES 1 ..... 14-31
14.12 R: PLC IMPUESTO A 0 EL NÚMERO DE PUNTOS PRECISADO SI EL STACK ES 1 ..... 14-32
14.13 LPS: PLC DUPLICACIÓN LÓGICA ..... 14-33
14.14 LPP: PLC COBRO LÓGICO ..... 14-34
14.15 LRD: PLC COPIA LÓGICA ..... 14-35
14.16 ALD: PLC COMBINADO LO PRIMERO Y EL SEGUNDO ELEMENTO POR AND ..... 14-36
14.17 OLD: PLC COMBINADO LO PRIMERO Y EL SEGUNDO ELEMENTO POR OR ..... 14-37
14.18 NOT: PLC MODIFICACIÓN DEL VALOR SUPERIOR DEL STACK ..... 14-38
14.19 LEQ: PLC CONFRONTADO DOS VALORES SI IGUAL CARGO CON EL STACK 1 SI NO 0 ..... 14-39
14.20 LGE: PLC CONFRONTADO DOS VALORES SI MAYOR IGUAL CARGO CON EL STACK 1 SI NO 0 ..... 14-40
14.21 LLE: PLC CONFRONTADO DOS VALORES SI MENOR IGUAL CARGO CON EL STACK 1 SI NO 0 ..... 14-41
14.22 AEQ: PLC CONFRONTADO DOS VALORES SI IGUALES HACE EL AND DE 1 CON EL STACK ..... 14-42
14.23 AGE: PLC CONFRONTADO DOS VALORES SI IGUALES O MAYOR HACE EL AND DE 1 CON EL STACK 14-43
14.24 ALE: PLC CONFRONTADO DOS VALORES SI IGUALES O MENOR HACE EL AND DE 1 CON EL STACK . 14-44
14.25 OEQ: PLC CONFRONTADO DOS VALORES SI IGUALES HACE EL OR DE 1 CON EL STACK ..... 14-45
14.26 OGE: PLC CONFRONTADO DOS VALORES SI IGUALES O MAYOR HACE EL OR DE 1 CON EL STACK 14-46
14.27 OLE: PLC CONFRONTADO DOS VALORES SI IGUALES O MENOR HACE EL OR DE 1 CON EL STACK 14-47
14.28 PEX: PLC EJECUTADO SOBRE 1 EN STACK LA INSTRUCCION AXESBRAIN ..... 14-48
14.29 TON: PLC TEMPORIZADOR SIN RETENCION ..... 14-49

14.30 TOR: PLC TEMPORIZADOR CON RETENCION ..... 14-49
14.31 CTU: PLC COUNTER UP ..... 14-52
14.32 CUD: PLC COUNTER UP Y DOWN ..... 14-52

**A. ASES VIRTUALES ..... 1**

**B. CONTROLO EJE CON OCTAVILLA ..... 10**

**C. ASES A PORTAL ( GANTRY) ..... 11**

**D. DIENTES DE RUEDA ELECTRONICOS ..... 12**

**E. INTEGRACIÓN CON LA VISIÓN ..... 13**

**F. DEFINICI DE GLOBALES ESPECIALES Y FUNZIONILIT ÀGENERALES ..... 15**

**G. ENUMERO ERRORES ..... 18**

# CAPÍTULO 1

## 1 Introducción

### 1.1 Como utilizar este manual

Este manual describe cómo utilizar los recursos provistos por el lenguaje de programación AxesBrain.
Su objetivo es proveerles especialmente a los programadores una rápida ayuda y soporte por función que raramente son utilizadas y proveer ofrecer una rápida guía en mérito.
El manual tiene, sin embargo, breves textos descriptivos fácilmente comprensibles.

## 1.2 Lenguaje de programación propietario para describir ciclos de manipulación

*AxesBrain es un lenguaje de programación es indicado por la definición de los ciclos de trabajo por la automación.*

Para describir el ciclo de trabajo de un manipulador o una máquina automática es necesario tener un lenguaje capaz de aprender como desenvuelven las fases. Existen numerosos lenguajes de programación, comúnmente son "propietarios" de la empresa. Las especificaciones que un lenguaje tiene que tener programación son múltiples en fases paralelas, sincronización entre las actividades programadas, alta integración con los dispositivo exteriores (visión) láser, etcétera.

Nacido por la especificación de la SIGLA, SIGMA LANGUAGE, 1976 del Olivetti como uno de los primeros lenguajes de programación de las máquinas especiales por el ensamblaje de partos, ha sido ampliado de modo que satisfacer las modernas exigencias de integración y flexibilidad. Referencia a: "Robot technology at Olivetti: the sigma system" Olivetti,Milan 1976.

Uno su prerrogativa más allá de la sencillez de sintaxis es la capacidad de tener la multiprogramación de los individuales ciclos, prerrogativa indispensable para poder efectuar operaciones de ensamblaje y manipulaciones de partos.

El presente manual quiere proveer las nociones esenciales por la redacción de programas de elaboración interpretable del sistema .

En particular ello es tan organizado:

El **Capítulo 2 Arquitectura** describe la arquitectura del fondo arregla de automación, en otras palabras de poder ejecutar los ciclos de trabajo en paralelo entre ellos sincronizándolos con de los acontecimientos.

El **Capítulo 3 Sintaxis del lenguaje** describe la sintaxis del lenguaje de programación, ella que es extremadamente simple siendo constituida por un triletterale precedido por un guión "- ", mientras uno "/" representa el delimitatore de los parámetros de la instrucción.

El **Capítulo 4 Variables globales y locales** define e introduce a los conceptos de las variables utilizados por el sistema subdividió en globales y locales.

El **Capítulo 5 Multitask y anticollisione** introduce al concepto de poder efectuar más operaciones junto, en seguridad y coordenadas entre ellos o menos, función de multitask y anticollisione de los ases.

El **Capítulo 6 Instrucciones lógicas matemáticas** afronta el problema de las instrucciones lógicas matemáticas, enumerando de ello la lista y ejemplificando cada instrucción.

El **Capítulo 7 Instrucciones de control** ilustra la lista de las instrucciones disponibles. Por cada instrucción es previsto un descriptivo de empleo y un ejemplo de programación.

El **Capítulo 8 Instrucciones de movimentazione** describe las instrucciones de programa relativo a la gestión de los ases. Por cada instrucción es previsto un descriptivo de empleo y un ejemplo de programación.

El **Capítulo 9 Instrucciones por las señales de input y salida** ilustra la lista de las instrucciones disponibles. Por cada instrucción es previsto un descriptivo de empleo y un ejemplo de programación.

El **Capítulo 10 Instrucciones de sincronización es** dedicado a la descripción del plató de struzioni necesarios a la sincronización de los acontecimientos.

El **Capítulo 11 Instrucciones de servicio** ilustra la lista de las instrucciones disponibles. Por cada instrucción es previsto un descriptivo de empleo y un ejemplo de programación.

El **Capítulo 12 Instrucciones por la integración con otros entornos** afronta el problema de la integración del sistema con otros entornos de trabajo, provee las informaciones necesarias a la gestión y una lista de las instrucciones disponibles con ejemplos de empleo..

El **Capítulo 13 Instrucción de comunicación** ilustra una lista de las instrucciones disponibles, con ejemplos de empleo, para poner en comunicación el sistema con otros recursos

El **Capítulo 14 Instrucciones AWL PLC** ilustra la lista de las instrucciones disponibles relativas al lenguaje de programación AWL, lenguaje de programación literal por PLC. Por cada instrucción es previsto un descriptivo de empleo y un ejemplo de programación.

El apéndice **A. Ases virtuales** ilustran el concepto de ases virtuales sobre sistemas a geometría SCARA y CILÍNDRICA.

El apéndice **B Control eje con octavilla** define la posibilidad de posicionamiento manual de un eje por una octavilla electrónica.

L ' **Apéndice C Ases a portal, Gantry,** ilustra el concepto de ases Gantry administrado por el sistema.

L ' **Apéndice D Dientes de rueda electrónicas** ilustra el concepto de diente de rueda electrónico y la posibilidad de gestión de parte del sistema.

L ' **Apéndice Y Movimientos en continuo** define y ejemplifica las varias posibilidades de generación tipos de perfiles con movimiento continuo.

L ' **Apéndice F Integración con la visión** ilustra el concepto de integración del sistema en un entorno de análisis de imágenes.

L ' **Apéndice G Lista errores** enumera la lista de los errores posibles que son engendrados durante una errada programación o configuración del sistema.

# CAPÍTULO 2

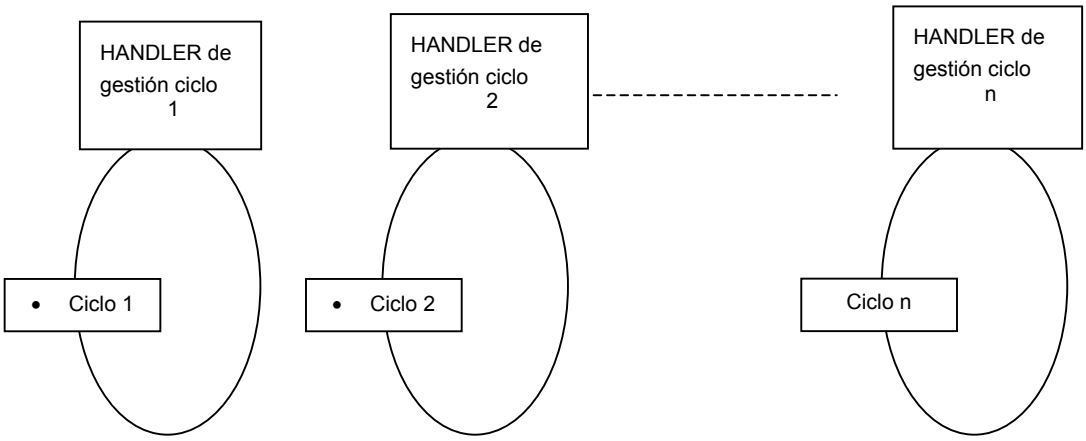
## 2 ARQUITECTURA

*La arquitectura del fondo arregla de automación es poder ejecutar los ciclos de trabajo en paralelo entre ellos sincronizándolos con de los acontecimientos.*

Cada programa tiene al propio interior un área propietaria dónde soy allocate las variables LOCALES, sobre cuyo además son depositados los parámetros de llamada.

Cuando al programa es insertado en ejecución, de un mando explícito o vuelto a llamar de un CALL o de una instrucción TSK, su código naciente es cargado en memoria y automáticamente precompilato, en memoria vosotros quedará hasta a un mando explícito de cancelación o mando al subsistema de REPOSICIÓN.

Los programas son juntados a un HANDLER de trabajo a propietario sobre que se colgarán todas las actividades del ciclo, podemos tener un número casi infinito de HANDLER, con esta modalidad se pueden tener configuración de instalación extremadamente flexible como líneas de ROBOTS, máquinas especiales multi cabeza, máquinas con cargados descargues integrados, máquinas multifuncionales, etc.


HANDLER de gestión ciclo 1
HANDLER de gestión ciclo 2
HANDLER de gestión ciclo n
Ciclo 1
Ciclo 2
Ciclo n

# CAPÍTULO 3

## 3 SINTAXIS DEL LENGUAJE

### 3.1 Definiciones del lenguaje

*La sintaxis del lenguaje es extremadamente simple un triletterale precedido por un guión "- "representa la instrucción, uno "/" representa el delimitatore de los parámetros de la instrucción que pueden ser:*

Referencias directas a variables GLOBALES o LOCALES

Referencias indirectas a variables GLOBALES

Expresiones numéricas con referencias a las variables GLOBALES o LOCALES

Expresiones matemáticas con referencias a las variables GLOBALES o LOCALES

LABEL de salto

Nombres de recursos ases, mandriles, input y salida

Los parámetros son separados por comas.

Las etiquetas o LABEL por los saltos condicionados o incondicionales son puestos antes del separador traten "- "de la instrucción.

Los comentarios son precedidos por carácter punto y coma ";"

[label]-XXX/[parametro 1],…[parametro n] [; éste es un comentario]

Se pueden comentar más rayas usando "/ *" al principio del área comentada y "* /" al final del área

**Ejemplo:**

```
; éste es un ejemplo de programación
      - LET/G1,0
qui - JNE/G1,1,qui              ; espera que el valor de la variable GLOBAL G1 sea puesto a 1
; de otro ciclo en elaboración paralela
```

Las expresiones matemáticas tienen las siguientes funciones:

| | |
|---|---|
| abs | Absoluto de un número |
| acos | Arco coseno |
| and | And entre dos números |
| asin | Arco seno |
| atan | Arco tangente |
| atanw | Arco tangente de Y,X |
| ceil | Redondeo hacia arriba de un número decimal en un número entero |
| cos | Coseno |
| cosh | Coseno hiperbólico |
| deg | Transformación en grados de un rincón expreso en radiantes |
| exp | Exponencial |
| floor | Redondeo hacia abajo de un número decimal en un número entero |
| logd | Logaritmo decimal |
| logn | Logaritmo natural |
| lshift | Shift hacia izquierda de un número |
| max | Massimo entre enne expresiones |
| min | Mínimo entre enne expresiones |
| mod | Módulo entre dos números |
| not | Negación boleana de un número |
| or | Or entre dos números |
| pe | P griego |
| rad | Transformación en radiantes de un rincón expreso en grados |
| rshift | Shift hacia derecha de un número |
| hasta | Seno |
| sinh | Seno hiperbólico |
| sqr | Raíz cuadrada |
| tan | Tangente |
| tanh | Tangente hiperbólica |
| xor | Or exclusivo entre dos números |

Los operadores en una expresión son:

+ suma
- sustracción
/ división
* multiplicación
^ elevado
( paréntesis abierto
) paréntesis cerrado

### Ejemplo de expresión matemática

```
- LET/L1, MAX(SIN(RAD(G1+12),COS(RAD(G1+12*L1/56,)
; en la variable LOCAL L1 es cargado el resultado de la expresión:
; MAX(SIN(RAD(G1+12),COS(RAD(G1+12*L1/56,)
```

## Notas

Los números en esadecimale son precedidos por 0x

### Ejemplo:

```
- LET/L1,0x10                    ; 0x10 = 16 decimal
```

El empleo del obrador esadecimale "0x" es particolarmerne útil en el mascheratura con las funciones de "and" y "or", por luego usarlos sucesivamente -

### Ejemplo:

```
- LET/L1,and(L1,0x8000,
- JEQ/L1,0x8000,Bit8000Uno
```

### 3.2 Dimensionas principio de los parámetros richiamabili

| | |
|---|---|
| GLOBALES | 32.767 |
| LOCALES | configuráis en la voz NumeroLocali = del expediente de configuración "SISTEMA.TXT", ampliables de-DIM/Numero LOCALES, |
| WATCH | 16 |
| DRT | 6 |
| Rayas DIS | 17 |
| Columnas DIS | 128 |
| Point | 32000 |
| NETwork | 128 |
| Temporizador (T) | 32 |
| Counter | 32 |

# CAPÍTULO 4

## 4 Variables globales y locales

*Para poder efectuar operaciones lógicas, leer valores numéricos son necesarias las variables.*

El fondo arregla AXESBRAIN preve dos tipos de variables:

LOCALES

GLOBALES

Cada programa al adecuado dell'entrata en función él alloca un número de variables pares a aquél configurado en el sistema, viene todo borráis y soy a disposición de las instrucciones de aquel programa, las primeras variables son programadas con los parámetros de llamada del mando de ejecución, la variable quedarán en memoria a disposición por las operaciones de interrogaciones o visualización.

El número de LOCALES puede ser cambiado por el aquel previsto en el "**SISTEMA.TXT"** a la voz "**NumeroLocali** =", utilizando la instrucción-**DIM/numero** locales, que queda sólo definido por aquella instancia de task.

Un caso particular son los programas vueltos a llamar por las instrucciones de "- CAL" y de "- TSK" con parámetros, en estos casos las primeras variables LOCALES son cargadas con los parámetros de llamadas en el mismo orden posicional, si existen de las LOCALES en los parámetros de llamadas serán cargadas con el nuevo valor a la vuelta del programa llamado.

**Ejemplo:**

- CAL/,routinemia:L9,12,23,G1+89,L7

cuando ocurre la vuelta "routinemia "L9 y L7 tendrán el valor definido en la rutina, en este caso

L9 tendrá el valor de 11

L7 tendrá el valor de 3

En efecto el código de "routinemia" es el partidario:

- LET/L1,11

- LET/L5,3

Al acto de la ejecución las variables LOCALES de "routinemia" son:

L1=valore de L9 del programa chiamante

L2 = 12

L3=23

L4=valore del variable G1 + 89

L5=valore de L7 del programa chiamante

Más allá de las variables LOCALES en el fondo arregla de automación son previstas **32767** variables **GLOBALES** que son vueltas a llamar con la carta "G" y el número de la misma.

Todo los **GLOBALES son** salvados sobre HardDisk, que por lo tanto pueden ser usáis para memorizar datos persistentes.

Las variables indirectas representadas por la carta "LOS" seguidos por el número indica que el valor que es tratado es aquel de la variable GLOBAL cuyo número es aquel de la variable LOCAL indicada por la carta "LOS."

**Ejemplo:**

- LET/L12,100

- LET/I12,120.56

En este caso el valor 120.56 es cargado en la variable GLOBAL 100 en efecto el valor 100 indicante el GLOBAL ha sido cargado en el LOCAL 12, de que se hace referencia en la instrucción "- LET/I12,120.56."

Para catalogar Globales o Locales se puede usar el paréntesis seguido por el Global o Local utilizado como convoca, el empleo del "LOS" ha quedado por compatibilidad,

**Ejemplo:**

```
; éste es un ejemplo de programación indiciada
 - LET / L1,1
 - LET/G(L1),0
aquí - JNE/G(L1)1,1,qui                ; espera que el valor de la variable GLOBAL G1 sea puesto a 1
```

Los Globales son salvados sobre HardDisk a la salida del sistema o en el procedimiento de "SHUT DOWN" es en todo caso posible efectuar la operación con la instrucción **SGL.**

LA OPERACIÓN de escritura de los Globales es efectuada con una modalidad de COMMIT para asegurar la integridad con el último rescate.

# CAPÍTULO 5

## 5 Multitask y anticollisione

*Una característica importante en la automación es el poder efectuar más operaciones junto, coordenadas entre ellos o menos, por lo tanto tenemos la necesidad de tener la función de "MULTITASK."*

Un ciclo de actividad puede ser ejecutado con un mando explícito, o bien de una instrucción "- TSK", el ciclo o programa es juntado a un HANDLER de trabajo a propietario sobre que se colgarán todas las actividades del ciclo, podemos tener un número casi infinito de HANDLER.

Un "TASK" puede ser borrado de parte de otro task o de él mismo con la instrucción "- TKM" o bien cuando es ejecutado el mando de REPOSICIÓN por el fondo arregla de automación.

El concepto de HANDLER además es útil verlo como un canal sobre que son efectuadas operaciones de principio del movimentazione en continuo, agregaciones de movimiento, espera que los movimientos sean completados.

Con esta arquitectura nosotros estamos capaz de ver el sistema de movimentazione como una instrumentación con numerosos "Bracci" que trabajan junto coordinados o menos, agregando dinámicamente grupos de ases.

Un ejemplo interesante puede ser imaginar el llenado de una bandeja de vasos, en una fase inicial tenemos dos "Bracci" que separadamente llenan las copas, el sistema administra el anticollisione y tenemos así dos grupos de ases que trabajan separadamente, cuando las copas han sido llenadas el sistema, agrupando los dos bracci como sería un camarero, entrega la bandeja en zona de descargue.

Cómo estreno evidenciado durante los llenados de las copas, los dos bracci incidiendo en un mismo eje físico X, gracias a la gestión del anticollisione es posible programar dos ciclos independientes de llenado, sincronizar los dos a cumplimiento de sus correspondientes fases, por lo tanto programar un único ciclo de descargue de la bandeja con una agrupación única de los dos bracci.

# CAPÍTULO 6

## 6 Instrucciones lógicas matemáticas

| | |
|---|---|
| .1-LET(SET, | Impuesto a una Variable el valor de la expresión (LET) |
| .2-ADD | Suma a una Variable el valor de la expresión (ADDed) |
| .3-MUL | Multiplica una Variable con el valor de la expresión (MULtiply) |
| .4-DIV | Divides una Variable con el valor de la expresión, DIVided, |
| .5-NEG | Niega el valor de una Variable (NEGation) |
| .6-LBF | Impuesto a un vector de Variables el valor de la expresión (Load Buffer) |

### 6.1 LET: LET value ( SET )

**LET(SET, LET value, SET value,**

Función: ejecutar una operación aritmética de inicialización de variable.

Parámetros: **a1,b1,a2,b2,... an,bn**

dónde:

**de a1.. an =** variable de inicializar
**de b1.. bn =** valores de introducir en las variables

## Ejemplo:

| | |
|---|---|
| - LET/L1,5.5,G7,L3 | Inicializar la variable local 1 con el valor 5.5 y la variable global 7 con el valor contenido en la variable local 3. |

**Notas:**

a, Si el parámetro **b** es una variable, éste siempre mantiene el valor inicial.

b, Por cada instrucción es posible inicializar hasta 16 variables.

c, El parámetro **b puede** ser como d ' altraparte todos los parámetros estáticos una expresión matemática.

d, Usted "LET" devuelve las instrucciones "ADD", "MUL", "DIV" y "NEG" rebosantes, pero he estado implementatate por conformidad con las especificaciones anteriores.

## Ejemplo:

- LET/L1,L1+4; corresponde - ADD/L1,4
- LET/L1,L1*4; corresponde - MUL/L1,4
- LET/L1,L1/4; corresponde - DIV/L1,4
- LET/L1,-L1; corresponde - NEG/L1

## 6.2 ADD: ADD value

**ADD** **ADD value**

Función: ejecutar una suma algebraica, a = a±b,

Parámetros: **a1,b1,a2,b2,... an,bn**

dónde:

**de a1 a an =** variables operandos en los que se conseguirá el resultado

**de b1 a bn =** valores de sumar algébraicamente

### Ejemplo:

- ADD/L1, -5,G2,L3 Sustraer 5 a la variable local 1 y sumar a la variable global 2 el valor de la variable local 3

**Notas:**

a, Si el parámetro **b** es una variable, éste siempre mantiene su valor inicial.

b, Por cada instrucción es posible obrar sobre un máximo de 16 variables.

## 6.3 MUL: MULtiplication

**MUL** **MULtiplication**

Función: ejecutar una multiplicación, a = a * b,

Parámetros: **a1,b1,a2,b2... an,bn**

dónde:

**de a1 a an =** variables operandos en los que se consigue el resultado

**de b1 a bn =** valores de multiplicar

### Ejemplo:

- MUL/G1,5.3,L2,G2 Multiplicar por 5.3 el valor de la variable global 1 y multiplicar por el valor de la variable global 2 el valor del local 2.

**Notas:**

a, Si el parámetro **b** es una variable, éste siempre mantiene su valor inicial.

b, Por cada instrucción es posible obrar sobre un máximo de 16 variables

## 6.4 DIV: DIVision

**DIV** **DIVision**

Función: ejecutar una división (a=a/b)

Parámetros: **a1,b1,a2,b2... an,bn**

dónde:

**de a1 a an =** variables operandos en los que se consigue el resultado

**de b1 a bn =** valores de dividir

### Ejemplo:

- DIV/L2,5.7,G4,L1 Dividir por 5.7 el valor de la variable local 2 y dividir por el valor de la variable local 1 el valor del global 4.

**Notas:**

a, Si el parámetro **b** es una variable, éste siempre mantiene su valor inicial.

b, Por cada instrucción es posible obrar sobre un máximo de 16 variables.

c, La división por cero engendra el mesaggio de error "error en la interpretación de la instrucción."

### 6.5 NEG: NEGation

**NEG NEGation**

Función: cambiar la señal algebraica de una variable a=-a

Parámetros: **a1,a2,... an**

dónde:

**de a1 a an =** variables de multiplicar por -1

## Ejemplo:

- NEG/L1,G7 Multiplicar por -1 los valores de las variables locales 1 y global 7

**Notas:**

a, Por cada instrucción es posible obrar sobre un máximo de 16 variables.

## 6.6 LBF: Load BuFfer

**LBF** **Load BuFfer**

Función: ejecutar una operación aritmética de inicialización de un buffer de variables.

Parámetros: **a,b,c**

dónde:

**a** = variable Local o Global inicial

**b** = variable Local o Global final

**c** = valor de introducir en las variables

### Ejemplo:

- LBF/G1,G100,0 Borrar las variables globales incluidas entre 1 y 100.

**Notas:**

Si el parámetro **c** es una variable, éste siempre mantiene el valor inicial.

Las variables indicadas en los parámetros **a, b** tienen que ser mismo tipo y progresivas.

# CAPÍTULO 7

## 7 INSTRUCCIONES DE CONTROL

| | |
|---|---|
| .1-JMP | Salta incondicionalmente a un Label (JuMP) |
| .2-JEQ | Salta a un Label si las dos expresiones son iguales, Jump if Equal, |
| .3-JNE | Salta a un Label si las dos expresiones no son iguales, Jump en Not Equal, |
| .4-JLT | Salta a un Label si el valor del primer parámetro es menor del segundo, Jump if Less Then, |
| .5-JLE | Salta a un Label si el valor del primer parámetro es menor o igual del segundo, Jump if Less then and Equal, |
| .6-JGT | Salta a un Label si el valor del primer parámetro es mayor del segundo, Jump if Great Then, |
| .7-JGE | Salta a un Label si el valor del primer parámetro es mayor o igual del segundo, Jump if Great then and Equal, |
| .8-JRN | Salta si el valor del parámetro es comprendido en el range, Jump if RaNge, |
| .9-JNR | Salta si el valor del parámetro es fuera range, Jump if Not Range, |
| .10-JOS | Salta si al menos un bit del valor del parámetro es uno, Jump Or if bit Plató, |
| .11-JOC | Salta si al menos un bit del valor del parámetro es cero, Jump Or if bit Clear, |
| .12-JAS | Salta si todos los bites del valor del parámetro son uno, Jump And if bit Plató, |
| .13-JAC | Salta si todos los bites del valor del parámetro son cero, Jump And if bit Clear, |
| .14-CAL | Llama un part program pasándole dioses parámetros (CALl) |
| .15-RET | Vuelve al chiamante del programa (RETurn) |
| .16-END | Fino proceso (END) |
| .17-TSK | Ejecuta en paralelo un ciclo de trabajo (TaSK) |
| .18-TKM | Suspende, restablece y borra un ciclo de trabajo, TasK Manegement, |
| .19-DIM | Dimensiona el número de variable L de un part pogram |

## 7.1 JMP: JuMP

**JMP JuMP**

Función: salto incondicional al label indicado, en el mismo part program.

Parámetros: **a**

dónde:

**a** = Label de salto

El label es un cordón alfanumérico sin espacios, el label puede estar singularmente sobre una raya tiene que pero existir el carácter "- "

**Ejemplo:**

```
- JMP/Pippo              El programa salta al label Pippo.
...
...
Pippo -.....
```

**Notas:**

a, Esta función sólo es válida dentro de un mismo part-program.

b, En un mismo part-program pueden coexistir más **JMP** con el mismo label.

c, Un uso particular es el el salto indiciado al valor de un LOCAL o GLOBAL
**-JMP/L #o bien-JMP/G #**
en este caso el salto ocurrirá al label numérico equivalente al valor entero contenido en la variable.

**Ejemplo:**

```
- JMP/L1        ; si el contenido de L1 es 23 salteà al label 23
...             ; si el contenido de L1 es 24 salteà al label 24
...             ; si el contenido de L1 es 35 salteà al label 35
23 -.....

24 -.....

35 -.....
```

## 7.2 JEQ: Jump EQual

**JEQ** **Jump EQual, Jump si =**

Función: ejecutar una prueba sobre n valores y en función del resultado ejecutar o no el salto condicionado.

Parámetros: **a,b1,... bn,c**

dónde:

**a** = variable o vocablo de sistema inherente la prueba

**de b1 a bn** = valores de confrontar, max. 14,

**c** = n° de salto si se averigua es decir la condición prevista si al menos un parámetro **c** es igual al parámetro **b.**

### Esempio1:

```
- JEQ/L1,6,14,13,Equal      Si el contenido de la variable local 1 es 6 o bien 14 o bien 13, el programa
                            salta al label Equal; en caso contrario procede
…..
…..
…..
Equal….
```

### Esempio2:

```
a                           - JEQ/L1,0,1,2,3,4,L1
                            - LET/L1,L1+1
                            - DIS/pippo,L1
                            - JMP/L1
                            - JMP/a
5 -
32 -
56 -
                            - TMM/1000
                            - DIS />
                            - END

Fin -
                            - TMM/1000
                            - DIS />          .
```

## 7.3 JNE: Jump Not Equal

**JNE** **Jump Not Equal, Jump si mucho,**

Función: ejecutar una prueba sobre n valores y en función del resultado ejecutar o no el salto condicionado.

Parámetros: **a,b1,... bn,c**

dónde:

**a** = variable o vocablo de sistema inherente la prueba

**de b1 a bcn** = valores de confrontar, max. 14,

**c** = n° de salto si se averigua es decir la condición prevista si todo los parámetros **c** son diferentes del parámetro **b.**

**Esempio1:**

```
- JNE/L1,10,12,-6,Noequal      Si el contenido de la variable local 1 es diferente de 10,12,-6, el programa
                               salta al label Noequal; en caso contrario procede
…….
…….
Noequal…..
```

**Esempio2:**

```
a                              - JNE/L1,0,1,2,3,4,L1
                               - LET/L1,L1+1
                               - DIS/pippo,L1
                               - JMP/L1
                               - JMP/a
5 -
32 -
56 -
                               - TMM/1000
                               - DIS />
                               - END

Fin -
                               - TMM/1000
                               - DIS />
```

## 7.4 JGT: Jump Greater Than

## 7.5 JGE: Jump Greater Equal

## 7.6 JLT: Jump Less Than

## 7.7 JLE:Jump Less Equal

**JGT Jump Greater Than , Jump si >**
**JGE Jump Greater Equal , Jump si ≥**
**JLT Jump Less Than , Jump si <**
**JLE Jump Less Equal , Jump si ≤**

Función: ejecutar una prueba sobre un valor y en función del resultado ejecutar o no el salto condicionado.

Parámetros: **a,b,c**

dónde:

**a** = variable o vocablo de sistema inherente la prueba

**b** = valor de confrontar

**c** = n° de salto si se averigua la condición prevista

**Ejemplo:**

```
- JGT/L1,4,Salt   Si el contenido de la variable local 1 es superior a 4, el programa salta al label Salt; en caso contrario procede
...
...
Salt     -
```

## 7.8 JRN: Jump RaNge

| **JRN** | **Jump RaNge** |
|---|---|
| Función: | ejecutar una prueba sobre un range, valor incluido entre 2 límites, y ejecutar el salto si Tal valor es comprendido en el range. |

Parámetros: **a,b,c,d**

dónde:

**a** = variable o vocablo de sistema inherente la prueba

**b** = valor mínimo previsto

**c** = valor máximo previsto

**d** = n° de salto si se averigua la condición prevista, es decir, si **a es** comprendido en el range **bc,** extremo incluidos,

**Ejemplo:**

```
- JRN/L1,30,50,Salt      Si 50> L1> 30, el programa salta al label Salt; en caso contrario procede
...
Salt -
```

## 7.9 JNR: Jump Not Range

**JNR** **Jump Not Range**

Función: ejecutar una prueba sobre un range, valor incluido entre dos límites, y ejecutar el salto si tal valor es externo al range.

Parámetros: **a,b,c,d**

dónde:

**a** = variable o vocablo de sistema inherente la prueba

**b** = valor mínimo previsto

**c** = valor máximo previsto

**d** = n° de salto si se averigua la condición prevista, es decir si **a** es menor o mayor del range **bc,**

**Ejemplo:**

- JNR/L1,30,50,Salt Si L1 es menor de 30 o mayor de 50, el programa salta al label Salt; en caso contrario procede

…

Salt -

**Ejemplo:**

- RAV/6:L11,Z; 6 cuota real absoluto eje Z en L11
- RAV/12:L12,Z; 12 over travel positivo absoluto eje Z en L12
- JNR/L11,L12-3,L12+3,FuoriPosizione

## 7.10 JOS: Jump Or if bit Set

**JOS** **Jump Or if bit Set**

Función: ejecutar una prueba parcial sobre el valor de una variable, máscara de bit, y en función del resultado salta si al menos uno de los bites del valor del parámetro soy uno.

Parámetros: **a,b,c**

dónde:

**a** = variable de probar

**b** = valor (máscara) de prueba

**c** = JUMP si se averigua la condición

### Ejemplo:

L1 = 11

- JOS/L1,6,Salt
  … no los bites probados no corresponden
Salt.
  … ok los bites probados corresponden

| | | | | | |
|---|---|---|---|---|---|
| máscara de L1 | 1 | 0 | 1 | 1 | (1+2+8) |
| máscara de 6 | | 1 | 1 | 0 | (2+4) |

Ya que al menos uno de los bites de prueba corresponden a los del variable L1 ocurre el salto condicionado.

### Notas:

a, En esta instrucción la prueba sólo es efectuada sobre los bites indicados por el parámetro **b**; los otros bites son ignorados.

## 7.11 JOC: Jump OR if bit Clear

**JOC** **Jump Or if bit Clear**

Función: ejecutar una prueba parcial sobre el valor de una variable, máscara de bit, y en función del resultado salta si al menos uno de los bites del valor del parámetro soy cero.

Parámetros: **a,b,c**

dónde:

**a** = variable de probar

**b** = valor (máscara) de prueba

**c** = JUMP si se averigua la condición

### Ejemplo:

L1 = 15

- JOS/L1,6,Salt
  … no los bites probados no corresponden
Salt.
  … ok los bites probados corresponden

| | | | | | |
|---|---|---|---|---|---|
| máscara de L1 | 1 | 1 | 1 | 1 | (1+2+4+8) |
| máscara de 6 | | 1 | 1 | 0 | (2+4) |

Ya que todos los bites de prueba de la máscara con respecto de aquellos del variable L1 son a uno no ocurre el salto condicionado.

### Notas:

a, En esta instrucción la prueba sólo es efectuada sobre los bites indicados por el parámetro **b**; los otros bites son ignorados.

## 7.12 JAS: Jump And if bit Set

**JAS** **Jump And if bit Set**

Función: ejecutar una prueba parcial sobre el valor de una variable, máscara de bit, y en función del resultado salta si todos los bites del valor del parámetro son uno.

Parámetros: **a,b,c**

dónde:

**a** = variable de probar

**b** = valor (máscara) de prueba

**c** = JUMP si se averigua la condición

**Ejemplo:**

L1 = 11

- JAS/L1,6,Salt
  … no los bites probados no corresponden
Salt.
  … ok los bites probados corresponden

| | | | | | |
|---|---|---|---|---|---|
| máscara de L1 | 1 | 0 | 1 | 1 | (1+2+8) |
| máscara de 6 | | 1 | 1 | 0 | (2+4) |

Ya que los bites de prueba no corresponden a los del variable L1 no ocurre el salto condicionado.

**Notas:**

a, En esta instrucción la prueba sólo es efectuada sobre los bites indicados por el parámetro **b**; los otros bites son ignorados.

## 7.13 JAC: Jump And if bit Clear

**JAC** **Jump And if bit Clear**

Función: ejecutar una prueba parcial sobre el valor de una variable, máscara de bit, y en función del resultado salta si todos los bites del valor del parámetro son cero.

Parámetros: **a,b,c**

dónde:

**a** = variable de probar

**b** = valor (máscara) de prueba

**c** = JUMP si se averigua la condición

**Ejemplo:**

L1 = 9

- JAS/L1,6,Salt
  … no los bites probados no corresponden
Salt.
  … ok los bites probados corresponden

| | | | | | |
|---|---|---|---|---|---|
| máscara de L1 | 1 | 0 | 0 | 1 | (1+8) |
| máscara de 6 | | 1 | 1 | 0 | (2+4) |

Ya que los bites de prueba de la máscara con respecto de aquellos del variable L1 son a cero ocurre el salto condicionado.

**Notas:**

a, En esta instrucción la prueba sólo es efectuada sobre los bites indicados por el parámetro **b**; los otros bites son ignorados.

## 7.14 CAL: CALl

**CAL CALI**

Función: volver a llamar el part-program con o sin parámetros de mandar y recibir

Parámetros: **a: [c1,... c32]**

dónde:

**a** = label del part program llamado**,**
nombre expediente del part program llamado

**de c1 a c32** = parámetros de transmitir en las variables locales del part program de ejecutar

**Ejemplo:**

- CAL/,c:\pippo.pp:5.4,L5,G3 Volver a llamar el part program 3 transmitiendo el valor 5.4 en la variable local 1, el valor correspondiente a la variable local 5 en la variable local 2 y el valor correspondiente a la variable global 3 en la variable local 3

El label de part program dentro de un expediente: PartProgram[label]

## 7.15 RET: RETurn

**RET RETurn**

Función: provocar la salida del part program durante la ejecución del ciclo. Si fuera una rutina vuelve al chiamante del programa.

Parámetros: **(nadie)**

**Ejemplo:**

```
- RET
...
...
- RET
...
...
- RET
```

**Notas:**

a, Sobre un mismo part-program pueden haber más **RET.**

## 7.16 END: END

**END End**

Función: provocar la salida del part program durante la ejecución del ciclo. Si fuera una rutina interrumpe en todo caso el proceso.

Parámetros: **(nadie)**

**Ejemplo:**

```
- END
...
...
- END
...
...
- END
```

**Notas:**

a, Sobre un mismo part-program pueden haber más **END.**

## 7.17 TSK: TaSK

**TSK TaSK**

Función: ejecución de un task en paralelo a otros

Parámetros: **a:[b1,... B15]:[c]**

dónde:

| | | |
|---|---|---|
| **a** | = | Label,Nome expediente del task, part program, de ejecutar |
| **de b1 a b15** | = | parámetros de transmitir en las variables locales del part program de ejecutar |
| **c** | = | modalidad de control del task<br>0 = no controla y lanza la ejecución de una instancia del TASK (estándard)<br>1 = si el task existe no el esegue,ma de ello accoda la ejecución<br>2 = señala un error de task ya operante<br>3 = si el proceso chiamante también es de tipo siempre activo el task será del mismo tipo, siempre activo, no resettabile,<br>4 = el task siempre será activo, no resettabile |

**Ejemplo:**

```
- TSK/,C:\task1.pp
- TSK/,\task2.pp:50,G4
- RET
```

**Notas:**

Los parámetros son transmitidos posizionalmente (**b1** en **L1) b2** en **L2**,...**b15** en **L15.** Y posible transmitir hasta 32 parámetros.

Los parámetros indicados como Locales serán puestos al día al final de la ejecución del TASK si el part program que solicita de ello la ejecución todavía es operante.

**Ejemplo multitask por una puesta a cero multitesta:**

```
;
;               1 = mask eje ZL
;               2 = mask eje ZR
;               4 = mask eje XL
;               8 = mask eje XR
;               16 = mask eje Y
;               32 = mask eje SL
;               64 = mask eje SR
;               128 = mask eje LU
```

```
                - DIS/0:>
                - DIS/1:>
                - DIS/2:>
                - DIS/3:>
                - DIS/4:>
                - DIS/5:>
                - DIS/6:>
                - DIS/7:>
                - DIS/8:>
                - EVC/1,2,3,4,5,6,7,8,9
                - LET/L10,1+2+4+8+16+32+64+128-1
                - LET/L11,AND(L10,1,
                - JNE/L11,1,NO_ZL
;               0 = posición eje ZL con origen hecho
                - TSK/OMOZL:0
                - JMP/DO_ZL
NO_ZL           - EVS/1
DO_ZL           - LET/L11,AND(L10,2,
                - JNE/L11,2,NO_ZR
;               0 = posición eje ZR con origen hecho
                - TSK/OMOZR:0
                - JMP/DO_ZR
NO_ZR           - EVS/2
DO_ZR           - EVW/1,2
                - LET/L11,AND(L10,12,
                - JNE/L11,12,NO_XLXR
;                500 = posición eje XR con origen hecho
;               -550 = posición eje XR por búsqueda MARKER
;               -500 = posición eje XL con origen hecho
;                600 = posición eje XL por búsqueda MARKER
                - TSK/OMOX:500,-550,-500,600
                - JMP/DO_XLXR
NO_XLXR         - EVS/9
DO_XLXR         - LET/L11,AND(L10,16,
                - JNE/L11,16,NO_Y
;                 0 = posición eje Y con origen hecho
;                250 = posición eje Y por búsqueda MARKER
                - TSK/OMOY:0,250
                - JMP/DO_Y
NO_Y            - EVS/5
DO_Y            - LET/L11,AND(L10,32,
                - JNE/L11,32,NO_SL
;               0 = posición eje SL con origen hecho
                - TSK/OMOSL:0
                - JMP/DO_SL
NO_SL           - EVS/6
DO_SL           - LET/L11,AND(L10,64,
                - JNE/L11,64,NO_SR
;               0 = posición eje SR con origen hecho
                - TSK/OMOSR:0
                - JMP/DO_SR
```

```
NO_SR           - EVS/7
DO_SR           - LET/L11,AND(L10,128,
                - JNE/L11,128,NO_LU
;               0 = posición eje LU con origen hecho
                - TSK/OMOLU:0
                - JMP/DO_LU
NO_LU           - EVS/8
DO_LU           - EVW/5,6,7,8,9
                - DIS/0: Origen HECHO SOBRE TODOS LOS ASES DESEADOS
                - RET


PartProgram[OMOZL]
;               Origen EJE ZL
;
;               L1 = Posición eje ZL con origen hecho
;
                - HOM/ZL
                - MOV/ZL,L1
                - EVS/1
                - DIS/1: Origen ZL hecho
                - RET


PartProgram[OMOZR]
;               Origen EJE ZR
;
;               L1 = Posición eje ZR con origen hecho
;
                - HOM/ZR
                - MOV/ZR,L1
                - EVS/2
                - DIS/2:Origine ZR hecho
                - RET


PartProgram[OMOX]
;               Origen VELOCES ASES XES
;               L1 = Posición eje XR con Origen hecho
;               L2 = Posición por principio Origen eje XR
;                       , con respecto de lo micro de Origen,
;               L3 = Posición eje XL con Origen hecho
;               L4 = Posición por principio Origen eje XL
;                       , con respecto de lo micro de Origen,
                - EVC/1,2,3,4
                - TSK/OMOXR:L1,L2; -550 = Pos. inicio búsqueda marker XR
                - TSK/OMOXL:L3,L4;; 600 = Pos. inicio búsqueda marker XL
                - EVW/1,2
                - EVS/9
                - DIS/3: Origen HECHO SOBRE ASES XR y XR
```

```
                - RET


PartProgram[OMOXL]
;               Origen VELOZ EJE XL
;
;               L1 = Posición eje con Origen hecho
;               L2 = Posición por principio Origen eje XL
;                       , con respecto de lo micro de Origen,
;
                - SFP/5000,1000,1000
                - HOM/3:XL
                - DIS/2:>
                - TMS/XL,-2000:5:160,0,NTCH     ; -2000 = posiz. final
                - DIS/2:NON TOCADO MICRO XL
                - JMP/FINI
NTCH            - EVS/4
                - EVW/3
                - SFP/7000,1000,1000
                - RAV/1:L10,XL
                - MOV/XL,L2+L10
                - SFP/5000,1000,1000
                - HOM/17:XL             ; Origen sobre Marker dir. positiva
                - MOV/XL,L1             ; -500
                - DIS/4: Origen XL hecho
                - EVS/1
FINOS           -
                - RET



PartProgram[OMOXR]
;               Origen VELOZ EJE XR
;
;               L1 = Posición eje con Origen hecho
;               L2 = Posición por principio Origen eje XR
;                       , con respecto de lo micro de Origen,
;
                - SFP/5000,1000,1000
                - HOM/3:XR
                - DIS/3:>
                - TMS/XR,2000:5:159,0,NTCH      ;2000 = posiz. final
                - DIS/3:NON TOCADO MICRO XR
                - JMP/FINI
NTCH    - EVW/4
                - SFP/7000,1000,1000
                - RAV/1:L10,XR
                - MOV/XR,L2+L10
                - SFP/5000,1000,1000
                - HOM/17:XR             ; Origen sobre Marker dir. positiva
```

```
                - MOV/XR,L1             ;500
                - EVS/3
                - DIS/3: Origen XR hecho
                - EVS/2
FINOS           -
                - RET


PartProgram[OMOY]
;               Origen VELOZ EJE Y
;
;               L1 = Posición eje con Origen hecho
;               L2 = Posición por principio Origen eje Y
;                       , con respecto de lo micro de Origen,
;
                - SFP/5000,1000,1000
                - HOM/3:Y
                - DIS/1:>
                - TMS/Y,-2000:5:158,0,NTCH      ; -2000 = posiz. final
                - DIS/1:NON TOCADO MICRO
                - JMP/FINI
NTCH            - SFP/7000,1000,1000
                - RAV/1:L10,Y
                - MOV/Y,L2+L10
                - SFP/5000,1000,1000
                - HOM/49:Y              ; Origen sobre Marker dir. rechazo
                - MOV/Y,L1
                - EVS/5
                - DIS/5: Origen Y hecho
FINOS           -
                - RET


PartProgram[OMOSL]
;               Origen EJE SL
;
;               L1 = Posición eje SL con Origen hecho
;
                - OMO/SL
                - MOV/SL,L1
                - EVS/6
                - DIS/6:OMO SL hecho
                - RET
```

```
PartProgram[OMOSR]
;               Origen EJE SR
;
;               L1 = Posición eje SR con Origen hecho
;
                - HOM/SR
                - MOV/SR,L1
                - EVS/7
                - DIS/7: Origen SR hecho
                - RET


PartProgram[OMOLU]
;               Origen EJE LU
;
;               L1 = Posición eje LU con Origen hecho
;
                - HOM/LU
                - MOV/LU,L1
                - EVS/8
                - DIS/8: Origen LU hecho
                - RET
```

**Notas:**

Los parámetros son transmitidos posizionalmente (c1 en L1) b2 en L2,... c15 en L15. Y posible transmitir hasta 32 parámetros.

Si los parámetros transmitidos son variables locales, regresan en input al par program chiamante con los valores asumidos en el part program llamado. En el ejemplo sobre indicado, después de la ejecución de la instrucción CAL la variable local L5 asumirá el valor corriente del variable L2 del part program pippo.pp.

## 7.18 TKM: TasK Management

**TKM TasK Management**

Función: Provoca el cierre del part program indicado

Parámetros: **a:b[:c:d] [, e,f][,g]**
dónde:

**a** = Tipo de operación (0-7)
0; ABORT DEL TASK Kill part program
1; SUSPENSIÓN TASK CON DESACELERACIÓN INMEDIATA
2; RICONTINUAZIONE DEL TASK SUSPENDIDO
3; SUSPENSIÓN TASK A. FIN MOVIMIENTOS AGREGADO
4; RESTART DEL TASK SUSPENDIDO
5; PLATÓ FEED PROFILE
6; CHANGE FEED PROFILE en porcentaje 0-100 en el parámetro **d**
7; CHANGE SPEED en porcentaje 0-100 en el parámetro **d**
**b** = Label,Nome expediente del task, part program, de cerrar
**c** = Flag parado del mandril y su ripartenza ves mando
caso tipo operación 0-4:
**d** = Local o Global de estado de vuelta 0 si no encontrara el TASK 1 si encontrara
caso tipo operación 5,6,7:
**d** = Velocidad verdadera
caso tipo operación 5:
**y** = Aceleración
**f** = Desaceleración
**g** = Local o Global de estado de vuelta 0 si no encontrara el TASK 1 si encontrara

TKM/(0-7):[nomecall] [, nomefile.pp]: [flagMandrino]:[[Vel],[Acc],[Dec]]: Local estado
TKM/(0-7):[nomefileISO.prg]:[flagMandrino][:[Vel],[Acc],[Dec]:] Local estado
TKM/(0-5):[numerotesta ISO]:[flagMandrino][:[Vel],[Acc],[Dec]]: Local estrado

**Ejemplo:**

```
-       - TKM/1:,A_TKMSet.pp::L1
        - TKM/1:,A_TKMWait.pp::L2
        - DIS/10:Trovato,L1
        - DIS/11:Trovato,L2
        - TMM/2000
        - TKM/2:,A_TKMSet.pp::L3
        - TKM/2:,A_TKMWait.pp::L4
```

**Notas:**

**a** = Tipo de operación (0-7)

Puede ser asociado en OR con la elección del tipo operación (0-7) los siguientes flag
En esadecimale:

10H: ; Gestión del mandril como MASCHIATURA
20H: ; Gestión del mandril como FRESATURA

En la gestión del mandril como maschiatura, ello es parado
o hecho repartir al mismo tiempo al movimiento; excepto con
la suspensión del task a fin movimientos ya' agregado, dónde
el mandril es parado a movimientos terminados
En la gestión del mandril como fresatura, ello es parado
después de que el movimiento y' parado; hecho repartir primera que el
movimiento sea reenvidado.

- TKM/5:,PP1.PP::c1[,d1[,e1]]
  c1 = Vel régimen sobre el trayecto
  d1 = Aceleración sobre el trayecto
  e1 = Desaceleración sobre el trayecto

  Cambia la velocidad' y las aceleraciones en ejecución, a partir
  todavía das movimientos que agregar.
  Los valores quedan sobre el task hasta a un próximo cambio con TKM
  o con SFP, instrucción que tiene que estar sobre el task llamado,
  El task tiene que ser reconocido por AxesBrain; o sea antes de
  usar un TKM el task llamado tiene que haber utilizado al menos
  una instrucción del tipo: SFP/0 o movimentazioni.

- TKM/6:,PP1.PP::c1[,d1[,e1]]
  c1 = Vel% sobre el trayecto
  d1 = Acc% sobre el trayecto
  e1 = Dec% sobre el trayecto

  Cambia en porcentaje la velocidad' y las aceleraciones
  del movimiento en ejecución en luego.
  Los valores quedan sobre el task hasta a un próximo cambio con TKM
  El task tiene que ser reconocido por AxesBrain; o sea antes de
  usar un TKM el task llamado tiene que haber utilizado al menos
  una instrucción del tipo: SFP/0 o movimentazioni.

- TKM/7:,PP1.PP::c1
  c1 = rotación% del mandril

  Cambia en porcentaje de rotación del mandril en ejecución
  en luego. Si el task pone en rotación más' de un mandril, viene
  mandado sólo él último que fue puesto en rotación por el
  task.

Los valores quedan sobre el task hasta a un próximo cambio con TKM
Si son presentes cambios en porcentaje de rotación,
, instrucción CAP/56:S1,10, ellas sólo actúan sobre el mandril
mandado de modo directo, pero a un siguiente mando SPD presiente
sobre el task llamado, viene riutillizzato el valor cargado con TKM
El task tiene que ser reconocido por AxesBrain; o sea antes de
usar un TKM el task llamado tiene que haber utilizado al menos
una instrucción del tipo: SFP/0 o movimentazioni.

## 7.19 DIM : DIMension

**DIM DIMension**

Función: Dimensiona el número de variable L de un part pogram

Parámetros: **a**

dónde:

**a** = Numero variables Locales del part program dimensionados

**Ejemplo:**

```
- DIM/100        ; Dimensiono a 100 el número de las variables Locales por el contexto del programa
```

**Notas:**

a,

- Los parámetros a la creación de un part program es configurado con el valor definido en la voz "**NumeroLocali"** del expediente de configuración "**sistema.txt"**, sección "**[ParametriGenerali]."**

# CAPÍTULO 8

## 8 Instrucciones de movimentazione

*En este capítulo son descritas las instrucciones de programa relativo a la gestión de los ases.*

Las instrucciones relativas a la gestión de los ases, descritas de modo detallado en las páginas siguientes, soy:

| | |
|---|---|
| .1-HOM(OMO, | Origen de un eje (HOMing) |
| .2-MOV | Movimiento interpolado linealmente de un grupo de ases (MOVe) |
| .3-CIR | Movimiento interpolado circular o elíptico en sentido horario de un grupo de ases (CIrcular Right) |
| .4-CIL | Movimiento interpolado circular o elíptico en sentido a izquierdas de un grupo de ases (CIrcular Left) |
| .5-CRR | Movimiento interpolado circular o elíptico en sentido horario de un grupo de ases con rayo conocido (Circular Radius Right) |
| .6-CRL | Movimiento interpolado circular o elíptico en sentido a izquierdas de un grupo de ases con rayo conocido (Circular Radius Left) |
| .7-STC | Principio de un movimentazione en continuo con definición de recorrido (STart Continuous) |
| .8-HLC | Espera del cumplimiento del movimentazione en continuo (HaLtContinuous) |
| .9-ABC | Cancelación del movimiento en continuo (Abort Continuous) |
| .10-CAP | Cambia los parámetros eje, Change Axis Parameter, |
| .11-HMS | Gestión Máster eslavas (Handling Máster eslavas) |
| .12-HEC | Gestión de los dientes de rueda (Handling Electronic Cam) |
| .13-CFR | Cambia los parámetros dinámicos de un eje (Change Feed Plazos) |
| .14-CPL | Cambia el loop de posición (Change Position Loop) |
| .15-PRD | Lee las posiciones del eje (Position ReaD) |
| .16-RAV | Lee los parámetros del eje (Read Axis Value) |
| .17-RSV | Lee la velocidad de un mandril (Read Speed Value) |
| .18-SFP | Impone la velocidad del perfil de movimentazione, Plató Feed Profile, |
| .19-SPD | Impone la velocidad de rotación de un mandril (SPeeD) |
| .20-TCH | Animo con tostación (TouCH) |
| .21-TMT | Animo con búsqueda valor de señal analógica, Prueba Movement Trasducer, |
| .22-TMS | Animo con búsqueda valor de sensor digital, Prueba Movement Sensor, |

| | |
|---|---|
| .23-TPE | Habilita el tastatore (Touch Probos Enable) |
| .24-SZP | Defines la posición de un plató ceros maquina, Plató Cero Point, |
| .25-LZP | Activa un plató de ceros maquina (Load Cero Point) |
| .26-PIN | (INQ) Flag de incremental sobre un eje (Position INcremental) |
| .27-PAB ABS, | Flag de absoluto sobre un eje (Position ABsolute) |
| .28-MMA | Mueve un eje con un movimiento manual, Move Manual Axis, |
| .29-OPT | Abre un expediente de puntos, Open PoinT expediente, |
| .30-MOR | Movimiento interpolado linealmente de un grupo de ases con antelación (MOv Rey) |
| .31-DCT | Animo con profundidad controlada por tastatore (Deep Control Touch) |
| .32-DCS | Animo con profundidad controlada por entrada digital (Deep Control Sensor) |
| .33-GRM | Mandas a los ases y a mandriles agrupados (GRoup Management) |

## 8.1 HOM: HOMing (OMO)

**HOM OMO, Origen de un eje (HOMing)**

Función: puesta a cero eje

Parámetros: **[b]:a**

dónde:

**a** = número o nombre del eje borrar
**b** = modalidad de puesta a cero Estándard 0

**modalidad de puesta a cero**

0 = origen según caracterización (estándard)
1 = origen sin micro sólo con marker
2 = origen sin marker solo con micro
3 = origen con puesta a cero eje incondicional sin movimiento
4 = origen a "GOLPE"
8 = prueba origen sobre marker por verificación pérdida steps encoder
16 = espacio origen caracterizado en dirección positiva por la búsqueda micro
32 = espacio origen caracterizado con dirección invertida con respecto de ella caracterización por la búsqueda micro
48 = espacio origen caracterizado en dirección negativa por la búsqueda micro

Luego:

- HOM/17:A1 sólo significa ejecución homing sobre eje A1 con búsqueda sobre marker en dirección negativa. Notar que la dirección y' rechazo porque' no es investigado lo micro, que y' virtualmente rebuscado en dirección positiva.
- HOM/18:A1 sólo significa ejecución homing sobre eje A1 con búsqueda sobre micro en dirección positiva.

## Ejemplo:

```
HOM/X1          Puesta a cero eje X1
```

**Notas:**

a, Esta instrucción tiene que ser ejecutada por todos los ases reales incrementales (función caracterizada, que se quiere mover después de cada encendido del sistema) ya que el control asume como posición cero aquella corriente al encendido.

b, Para borrar los ases en tiempos siguientes hace falta prever n instrucciones **HOM.**

c, El origen de los ases reales, transductor encoder o resolver, son ejecutados como sigue:

1 Búsqueda veloz de lo micro de cero, en la dirección, caracterizada o prevista por una instrucción CAP anterior, y con la velocidad caracterizada; si lo micro de cero corresponsal, no es caracterizado, interviene el mensaje de error.

2 Reducción de la velocidad a aquél caracterizado y a liberación de lo micro de cero en dirección opuesta.

3 Búsqueda del "marker", que se vuelve el cero eléctrico del control..

4 Si a este punto se desea tener un cero eje mecánico, hace falta averiguar el delta entre el cero eléctrico y la posición de golpe e introducirlo como compensación en caracterización.

y, Antes de borrar un eje hace falta averiguar que ello esté en el campo idóneo, positivo si la puesta a cero está en dirección negativa y viceversa,; de otro modo el eje iría en colisión.

f, Si se solicita la función **HOM** sobre de un eje virtual es señalado el mensaje de error.

g, Si se solicita la función HOM sobre de un eje, el origen es efectuado sobre el 1° marker encoder encontrado;.

## 8.2 MOV: MOVe

**MOV MOVe**

Función: mover los ases de la máquina "en rápido", a la velocidad vectorial indicada, el vector es definido por los ases Precisados, si indicara una salida lo pone en la condición requerida.

**Sintaxis 1**

Parámetros: **[a:] a1,b1,... bn:[c,d]**

dónde:

| | | |
|---|---|---|
| **a** | = | velocidad vectorial |
| **de a1 a an** | = | ases interesados, ven conocido b, |
| **de b1 a b** | = | coordenadas |
| **c** | = | nombre o número de una salida digital |
| **d** | = | valor a 1 u O de la salida digital |

**Sintaxis 2**

Parámetros: **[a:] Pb:[c,d]**

dónde:

| | | |
|---|---|---|
| **a** | = | velocidad vectorial |
| **Pb** | = | número del punto interesado |
| **c** | = | nombre o número de una salida digital |
| **d** | = | valor a 1 u O de la salida digital |

**Ejemplo:**

MOV/5000:X1,50.5,Y1 ,75 Ejecutar un movimiento con velocidad 5 m/min sobre el vector X,Y a las cuotas indicadas.

MOV/5000:X1,50.5,Y1 ,75:12,1 Ejecutar un movimiento con velocidad 5 m/min sobre el vector X,Y a las cuotas indicadas, al final del movimiento es puesto al valor 1 la salida 12

MOV/5000:P10 ejecutar un movimiento relativo a los ases, y a las coordenadas, indicadas en el punto P10 con velocidad de 5 m/min.

**Notas:**

a, La sintaxis 1 permite de ejecutar movimientos sobre indicación directa de ases y coordenadas. La sintaxis 2 permite de ejecutar movimientos utilizando los puntos de las matrices.

b, ll vector puede ser formado por un máximo de 6 ases en el caso de empleo de puntos y 7 ases por indicación específica de ases y coordenadas.

c, Si el parámetro **a es** omitido, el movimiento es ejecutado a la máxima velocidad posible, ella que es utilizada además en el caso el parámetro **a sea** precisado, con un valor demasiado elevado o con valor cero. Tal velocidad es constante a lo largo de todo el trayecto si el movimiento se realiza sobre ases reales, cartesianos o no, es en cambio variable en el caso de movimientos sobre ases virtuales porque durante el movimiento no es constante la variación angular sobre los dos ases reales interesado.

d, Si sobre la misma instrucción es repetido el eje, en ejecución comparece el mensaje de error.

e, La solicitud de **MOV** sobre un eje borrado, si caracterizara, provoca la visualización del mensaje de error.

f, Con la instrucción **MOV** es posible obrar sea sobre ases reales que sobre ases virtuales, conque no hay abbinamento, real virtual, entre los mismos; en caso contrario interviene la señal de error, en ejecución.

h, Si la instrucción **MOV** preve un movimiento sobre más ases, este se mueven en el siguiente modo:
si se trata de solos ases reales no cartesianos, por ejemplo, SCARA, este se mueven en sincronismo pero no en interpolación. Prácticamente, los varios movimientos inician y acaban junto pero el trayecto conseguido no es una línea recta pero depende de la geometría de los ases interesados;
si se trata de ases cartesianos, reales o virtuales, este se mueven en interpolación lineal.

Los, En el caso de movimiento sobre ases virtuales, debe ser recordado que el eventual eje rotatorio "**C**" puesto en la extremidad del brazo, en el caso no sea mencionado en la instrucción, tendrá o pego una compensación automática a segunda que tal situación haya sido prevista o menos en caracterización, ves caracterización ases virtuales.

Mueve los ases
Generalmente el movimiento y' interpolado a menos que no él muevan ases virtuales. En este caso son interpolados los ases físicos y no el movimiento virtual

- MOV/A1,b1,C1,d1,....

A1 C1.... nombres de los ases
b1 d1.... posición final del movimiento

MOVe
Mueve los ases con velocidad' eplicita siempre interpolando.
- MOV/v1:A1,b1,C1,d1,....

v1 = velocidad' vectorial del movimiento, 0 y' el valor
de estándard; en este caso el movimiento y como MOV
sin switch:)
A1 C1.... nombres de los ases
b1 d1.... posición final del movimiento

## 8.3 CIR: Circular Interpolation Right CIL: Circular Interpolation Left

**CIR** **Circular Interpolation Right**
**CIL** **Circular Interpolation Left**

Función: movimiento interpolado circular o elíptico de un grupo de ases en sentido a izquierdas y a izquierdas

Parámetros: **[a]:b,c,b,d,e,f,e,g,[h,i]**

dónde:

| | | |
|---|---|---|
| **a** | = | velocidad vectorial |
| **b** | = | nombre o número primer eje |
| **c** | = | posición final del movimiento primer eje |
| **d** | = | centro del círculo primer eje |
| **y** | = | nombre o número segundo eje |
| **f** | = | posición final del movimiento segundo eje |
| **g** | = | centro del círculo segundo eje |
| **y** | = | nombre o número tercer eje (hélice) |
| **f** | = | posición final del movimiento tercer eje (hélice) |

CIrcle Right
Movimiento circular horario (G02)
- CIR/z1:A1,b1,A1,cb1,C1,d1,C1,cd1,

z1 = velocidad' vectorial del círculo
A1 C1.... nombres de los ases
b1 d1.... posición final del movimiento
cb1 cd1 centro círculo

CIrcle Left
Movimiento circular a izquierdas (G03)
- CIL/z1:A1,b1,A1,cb1,C1,d1,C1,cd1,

z1 = velocidad' vectorial del círculo
A1 C1.... nombres de los ases
b1 d1.... posición final del movimiento
cb1 cd1 centro círculo

### Ejemplo movimiento rotatorio continuo

```
        - MOV/X1,100,Y1,100
        - STC/0:10
YY      - CIR/X1,100,X1,50,Y1,100,Y1,50
        - JMP/YY

* * * * * * * * * * * * * * HÉLICE SOBRE X1 Y1 C1 * * * * * * * * * * * * * *
        - DIS/1:-----------------
; Espera sirvo on
INI - LET/L1,0
        - SFP/10000,100,100
        - MOV/X2,100,Y2,0,Z1,0
        - CAP/0:Z1,128
        - STC/0:10
PIPPO   - CIR/X2,100,X2,50,Y2,0,Y2,0,Z1,10
        - ADD/L1,1
        - JLE/L1,10,PIPPO
        - HLC
        - CAP/0:Z1,-128
        - JMP/INI
```

## 8.4 CRR: Circular Radius Right CRL: Circular Radius Left

**CIR** **Circular Radius Right**
**CIL** **Circular Radius Left**

Función: movimiento interpolado circular o elíptico de un grupo de ases en sentido a izquierdas y a izquierdas con rayo conocido

Parámetros: **[a]:b,c,b,d,e,f,e,g,[h,i]**

dónde:

| | | |
|---|---|---|
| **a** | = | velocidad vectorial |
| **b** | = | nombre o número primer eje |
| **c** | = | posición final del movimiento primer eje |
| **d** | = | rayo del círculo |
| **y** | = | nombre o número segundo eje |
| **f** | = | posición final del movimiento segundo eje |
| **g** | = | no usado |
| **y** | = | nombre o número tercer eje (hélice) |
| **f** | = | posición final del movimiento tercer eje (hélice) |

CIrcle with Radius Right
Movimiento circular horario con rayo explícito
- CRR/z1:A1,b1,A1,rb1,C1,d1,C1,rb2

z1 = velocidad' vectorial del círculo
A1 C1.... nombres de los ases
b1 d1.... posición final del movimiento
rb1 reluzco círculo
rb2 unused

CIrcle with Radius Left

Movimiento circular a izquierdas con rayo explícito

- CRL/z1:A1,b1,A1,rb1,C1,d1,C1,rb2

z1 = velocidad' vectorial del círculo
A1 C1.... nombres de los ases
b1 d1.... posición final del movimiento
rb1 reluzco círculo
rb2 unused

## 8.5 STC: STart Continuous

**STC** **STart Continuous**

Función: habilitar el sistema a ejecutar las varias instrucciones del part program, siguientes a STC, de modo continuo en vez de individual.

Sintaxis 1, continuo normal o ejecución de spline,

Parámetros: **a:b,c,d,e**

dónde:

**a** = tipo sin parar, switch:)
- 0 = continuo estándar (valor de estándard)
- 1 = spline de tipo 1
- 2 = spline de tipo 2
- 3 = spline de tipo 3
- 5 = carga valores especiales
- 10 = movimientos paralelos
- 11 = doble interpolar, fly motion,
- 21 = número de entes en el buffer sin parar
- 22 = porcentual smooth en continuo
- 23 = SPLINE carga ciclos iterativos máximos
- 24 = SPLINE carga porcentual esmero velocidad'
- 25 = ESPACIO MÍNIMO de los movimientos en continuo

**b** = parámetro que depende de a
- a = 0 = parámetro por gestión overshoot * *>
- a = 1 = parámetro por gestión overshoot * *>
- a = 2 = parámetro por gestión overshoot * *>
- a = 3 = parámetro por gestión overshoot * *>
- a = 5 = HLC wait (>1), nowait, =0 o unused (<0,
- a = 10 = parámetro por gestión overshoot * *>
- a = 11 = parámetro por gestión overshoot * *>
- a = 21 = número de entes en el buffer sin parar (>3)
- a = 22 = Porcentual smooth en continuo
- a = 23 = ciclos iterativos máximos en spline, 1 - 100,
- a = 24 = porcentual accur. velocidad' en spline, >0 = >10,
- a = 25 = ESPACIO MÍNIMO del movimiento en contionuo

* *> parámetro por gestión overshoot
0 = continuo con velocidad' de esquina igual a cero
y salida inmediata.
>0 = continuo con velocidad' de esquina mayor de cero
y salida inmediata.
<0 = continuo con velocidad' de esquina mayor de cero
y salida diferida.
La salida de lo continuo ocurre o por fin continuo
(HLC) o por cambio parámetro B1 con valor >= 0 o por
finos entes de movimentazione admitidos.

**c** = parámetro que depende de a
a = 0 no usado
a = 1 modificador 1 de combadura, estándard = 1,
a = 2 modificador 1 de combadura, estándard = 1,
a = 3 modificador 1 de combadura, estándard = 1,
a = 5 = ciclos iterativos spline
a = 10 cuánto son los movimientos paralelos (<= 16,
a = 11 espacio de anticipación del segundo interpolatore
Con valor <0 el segundo interpolatore parte
cuando lo primero inicia a decelerar
a = 21 no usado
a = 22 no usado
a = 23 no usado
a = 24 no usado
a = 25 no usado

**d** = parámetro que depende de a
a = 0 no usado
a = 1 modificador 2 de combadura, estándard = 1,
a = 2 modificador 2 de combadura, estándard = 1,
a = 3 modificador 2 de combadura, estándard = 1,
za = 5 = Número entes del buffer sin parar
a = 10 no usado
a = 11 no usado
a = 21 no usado
a = 22 no usado
a = 23 no usado
a = 24 no usado
a = 25 no usado

a = parámetro que depende de a
a = 0 no usado
a = 1 modificador 3 de combadura, estándard = 1,
a = 2 modificador 3 de combadura, estándard = 1,
a = 3 modificador 3 de combadura, estándard = 1,
a = 5 = Porcentual smooth en continuo * * *>
a = 6 no usado
a = 10 no usado
a = 11 no usado
a = 21 no usado
a = 22 no usado
a = 23 no usado
a = 24 no usado
a = 25 no usado

* * *> Porcentual smooth
<=0 el parámetro no actúa
>0 Porcentaje de renta de tiempo en lo continuo por rasgos pequeños donde no se acelera si el ahorro de tiempo no supera este porcentaje , lleva sin cúspide entre vel.iniziale y vel.finale.

SPLINE

Los ases de los movimientos spline tienen que siempre ser los mismos, no importa el orden, pero no se pueden omitir aunque ella coordinada no cambia.

Spline de tipo 1:

La curva y' cuota a la cuerda subtendida al principio y al final a la cuerda siguiente

Spline de tipo 2

La curva tiene de las cuotas que son calculadas utilizando tres puntos

Spline de tipo 3

La curva y' parecido a la anterior, pero las cuotas son artificiosas utilizando 5 puntos

## Ejemplo 1:

- STC/10
- MOV/X1,100,Y1,100 ; Ejecutar los dos movimientos indicados de modo continuo con salida imme - MOV/X1,200 ; diata de los movimientos.
- TIM/0.5
...
...
- HLC

## Ejemplo 2:

- STC/2:-10,0.1 ; Ejecutar los 3 movimientos indicados con spline de tipo 2 sobre el perfil con salida- MOV/X1,100,Y1,100
**;** del primer movimiento en cuanto viene schedulata la instrucción HLC.
- MOV/X1,150,Y1,150
- MOV/X1,200,Y1,150
..
- HLC

**Notas** relativas a la sintaxis 1:

a, Si el parámetro **a es** omitido, por estándard asume valor 0, continuo normal,; en tal caso no ha significado el eventual parámetro **c,** indicado o menos.

b, Si el parámetro **c es** omitido, en tal caso tiene que ser necesariamente estar omitido también **d,** por estándard asume valor **1.**

c, La función continúo, por todos los tipos, queda válida hasta a que no es anulada por una instrucción HLC, Haft Continuous, o de una instrucción ABC (Abort Continuous).

d, Las instrucciones siguientes a **STC** e inherentes los movimientos (**MOV) ACC.** etcétera, son ejecutadas en la progresión indicada, mientras las instrucciones de otro tipo (**LET,JMP)** etcétera, incluso respetando la progresión con que son indicadas son ejecutadas enseguida; en el ejemplo sopracitato la excitación de la salida 1, la pausa de 0,5 segundos y el diseccitazione de la salida ocurren en cuanto parte el primero movimiento de X1.

y, El empleo del parámetro **d**, que puede ser omitido y que sólo es aceptado con parámetro "a" igual a cero, ha significado cuando en los movimientos de ejecutar en continuo hay de los TMS, TMT o TCH, movimientos que pueden ser interrumpidos en función de situaciones externas.
En estos casos, para utilizar correctamente este parámetro, observar cuánto sigue:

- el parámetro **c no** tiene ningún sentido pero no es omettibile;
- son tomadas en consideración 5 words consecutivos a partir de aquella relativa al bit indicado en el parámetro; tales words son borrados automáticamente a cada principio ejecución del STC;
- en el primer word son memorizadas dos cosas y precisamente:

- sobre el primer bit es memorizado el estado del último movimiento ejecutado; el bit asume valor 0 si el movimiento es acabado regularmente mientras asume valor 1 cuando el movimiento ha sido interrumpido porque ha **sido tocada la pared";**
- sobre el segundo bit es memorizada la diferencia entre movimientos "agregado" a lo continuo y movimientos "ejecutados"; el bit asume valor 0 cuando la cantidad de movimientos ejecutados es inferior a la de movimientos agregado; asume valor 1 cuando estas dos cantidades son iguales;

- en la segunda y terceros word, es memorizada la cantidad de movimientos agregado a lo continuo;
- en ella cuarta y quintos word, son memorizados la cantidad de movimientos ejecutados;
- ya que la memorización sobre los words ocurre a nivel de bits, es necesario en ciertos casos ejecutar la instrucción BPI para trasladar el valor de word a una variable del GP, local, global, indirecta,; por ejemplo BPI sobre 16 bites para conocer el valor memorizado en la segunda más terceros word o en la cuarta más quintos word;
- es facultad del programador probar estos words primeros e/o después de las varias instrucciones de movimiento;

f, Aunque no hay límites a la cantidad de task con continuos en ejecución que pueden intervenir en un dato instante, el número de entes "anima" que pueden ser agregados, precalcolati, es:

- 30 en la interpolación circular (CIR,CIL);
- 60 en los otros casos.

Si el número de entes de uno o más continuos es superior a estos límites, los excesos vienen accodati y artificiosa mano a mano que se libran los entes ya ejecutados.

g, Si el cálculo de los entes resulta más lento que los movimientos, como en el caso de los movimientos muy breves, es ejecutada una desaceleración final, al límite con detención de los ases. Los movimientos son retomados no en cuanto el cálculo agrega nuevos entes.

h, Por todos los movimientos ejecutados dentro de un continuo, entre instrucción **STC** e instrucción **HLC** o **ABC, no** son activados el control de posición mientras siempre quedan activos sea el control de colisión que el control paro cuenta; después de cada overshoot los ases son vueltos a llamar en todo caso en posición "tirada en sumisión" es decir vuelta a llamar por la sumisión a que ha sido dada la máxima tensión de referencia.

los, En todos los casos sin parar, si al parámetro **b es** asignado valor 0, se consigue un perfil siguiente:

- por cada punto del perfil os es parada a fin de cada movimiento; prácticamente los varios movimientos son efectuados como si fuera no activo lo continuo con la única excepción que no son activados los tiempos de **position time** y **wait cycle;**
- la conjunción entre las botaduras puntos del perfil siempre es una línea recta en el caso sin parar normal
  , parámetro **a** = 0,; es un arco más o chato en el caso de spline, parámetro **a**> de 0, en función del valor del parámetro **c.**

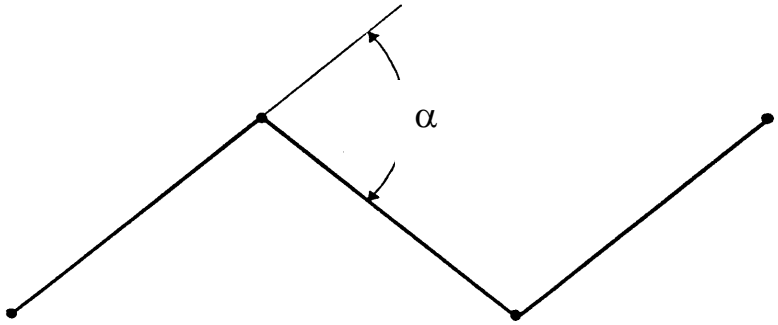
Los, En todos los casos sin parar, si al parámetro **b es** asignado un valor diferente por 0, se consigue un perfil siguiente:

- el paso sobre cada apunto perfil, sin parada, siempre está garantizado;
- no hay nunca detención sobre cada apunto perfil, pero sólo una disminución de velocidad de la velocidad; tal disminución de velocidad es más o menos marcado en función del tipo sin parar y del valor asignado al coeficiente STC.

- la conjunción entre las botaduras puntos del perfil puede ser una línea recta o un arco más o chato en función del valor asignada al parámetro **c.**
- después de cada apunto perfil puede haber o menos overshoot en función del tipo sin parar y del valor asignado al parámetro **c.**

Las notas siguientes **m,n,o,** describen el perfil ejecutado sin parar con las botaduras tipos.

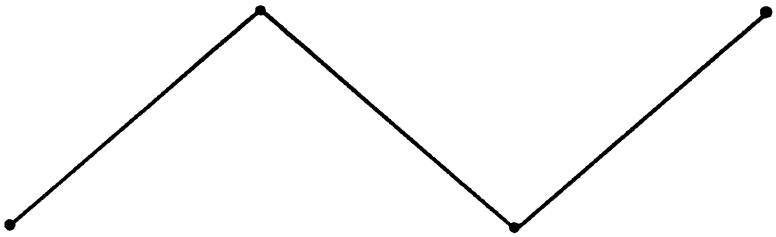
Los varios tipos sin parar, se refieren al perfil a teórico partidario:



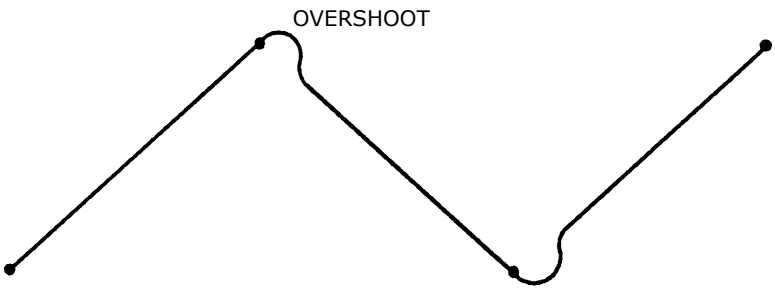

m, En el caso sin parar normal, parámetro **a** = 0, siempre hay overshoot después de cada apunto perfil. La velocidad de overshoot después de cada movimiento es directamente proporcional a la aceleración y al coeficiente STC y mayor, con incremento, o menor, con disminución, de modo no proporcional con respecto del rincón $\alpha$ (cambio de dirección entre 2 movimientos siguientes). Ya que la máxima velocidad de overshoot no puede superar en todo caso la máxima velocidad del eje, un valor grande de STC sólo es significativo con un valor pequeño de aceleración; viceversa con una gran aceleración ha significado un pequeño valor de STC, si este valor es grande, es forzado en todo caso a lo sumo admisible.

**Ejemplos de perfil con continuo normal**

Perfil práctico con coeficiente STC, parámetro **b,** = 0



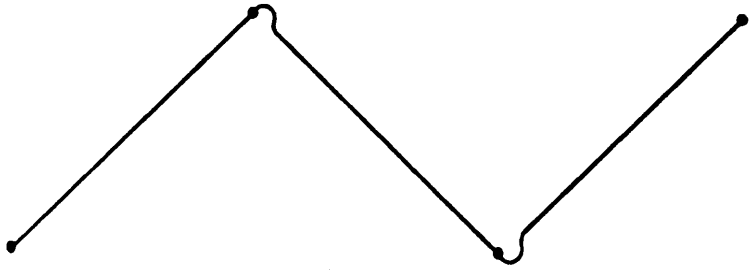
Perfil práctico con coeficiente STC diferente de 0



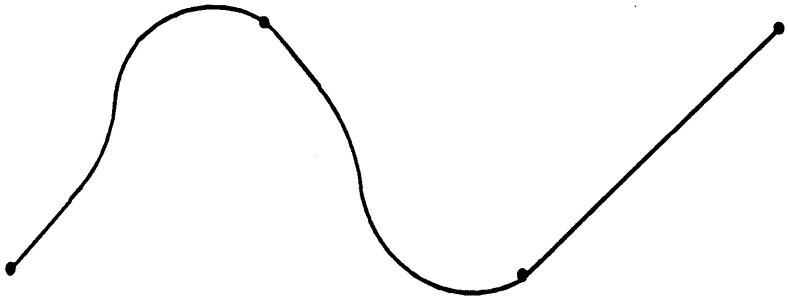
n, En el caso de spline de tipo 1, parámetro **a** = 1, el overshoot después de cada apunto perfil es claramente inferior a aquel relativo a lo continuo normal; prácticamente es nulo si se asigna valor 1 al parámetro **c**; varia de 0 a un valor creciente, en todo caso siempre bajo, más el valor del parámetro **c se** acerca al 0.
También para este tipo sin parar, se tiene una variación de velocidad a lo largo del perfil, por ella es menos muy evidente con respecto de lo continuo normal; naturalmente la velocidad sobre el perfil será tan más uniforme mucho menor es el rincón a, tan más el valor del parámetro o se acerca a 1 y a mucho mayor, aunque poco influyente, es el valor del coeficiente STC, parámetro **b.**

**Ejemplos de perfil con spline de tipo 1**

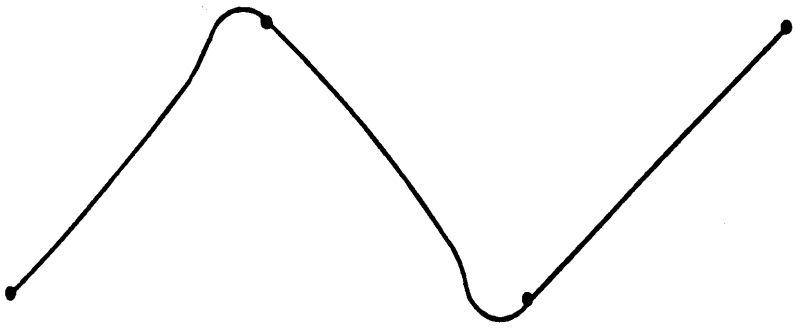
OVERSHOOT PICCOLO



Perfil conseguido con parámetro 0=0 y coeficiente STC diferente de 0; si el coeficiente STC es 0, no hay naturalmente los overshoot.



Perfil conseguido con parámetro **c** = los, con coeficiente STC diferente de 0,

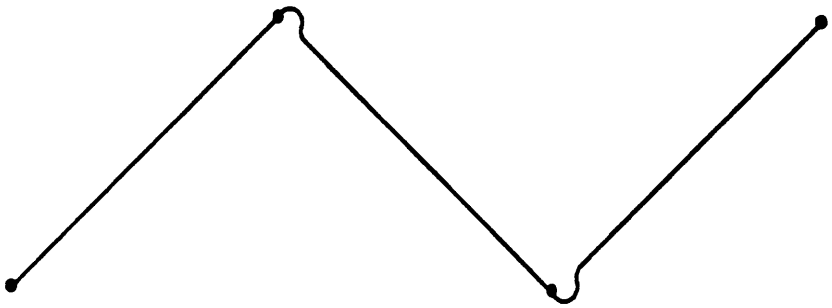


Perfil conseguido con parámetro **c** comprendido entre 0 y 1, coeficiente STC diferente de 0,

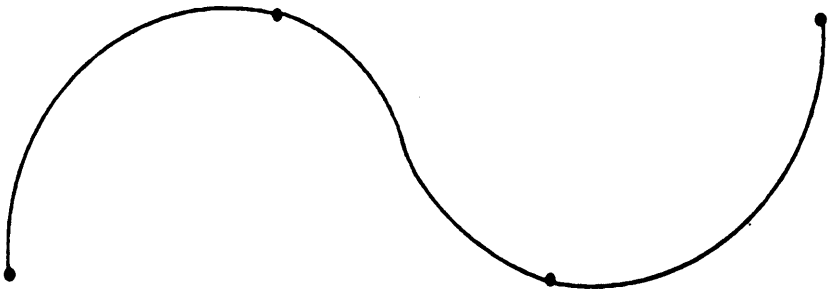
o, En el caso de spline de tipo 2, parámetro **a** = 2, y de tipo 3, parámetro **a** = 3, se tiene un comportamiento parecido al spline de tipo 1 pero con la diferencia que por el spline de tipo 2 el precalcolo sobre el perfil viene tacto tomando en consideración 3 puntos del perfil mientras por el spline de tipo 3 los puntos tomados en consideración son 5.

**Ejemplos de perfil con spline de tipo 2 y 3**

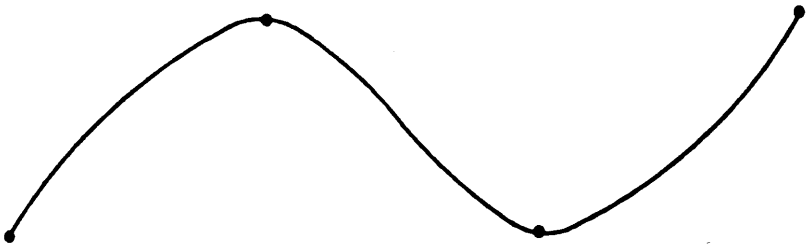
OVERSHOOT PEQUEÑO



Perfil conseguido con parámetro **c** = 0 y coeficiente STC diferente de 0, si el coeficiente STC es 0, no hay naturalmente los overshoot.



Perfil conseguido con parámetro **c** =1, coeficiente STC diferente de 0,



Perfil conseguido con parámetro **c** comprendido entre 0 y 1, coeficiente STC diferente de 0.

p, En el caso de los spline, cuando un movimiento concierne más ases, hacen falta también repetir todos los ases por los movimientos siguientes; si por ejemplo un eje no es interesado en el movimiento, hace falta repetirlo remachando la coordenada anterior; en caso contrario sucede la señal de error.

## Ejemplo

- STC/2: 10,1
- MOV/X1,100,Y1,100
- MOV/X1,150,Y1,100 ; eje Y1 remachado con coordinado precedente.

q, Con base en las consideraciones vistas, el empleo de un tipo o el otro sin parar es atado a muchos factores y precisamente:

- uniformidad del perfil práctico con respecto del teórico
- uniformidad de la velocidad a lo largo del perfil
- tiempo de ejecución del perfil.

Las consideraciones que él puede hacer son las partidarias:

- lo continuo normal casi es cada vez más veloz como tiempo de ejecución pero comporta una mayor variación de velocidad a lo largo del perfil con un consiguiente mayor apremio de los motores de los ases.
- El continuo tipo spline comporta una mayor uniformidad como velocidad a lo largo del perfil y por lo tanto un menor apremio sobre los motores a pérdida pero del trayecto práctico con respecto de la teórica y a pérdida muchas veces del tiempo de ejecución.

## 8.6 HLC: HaLt Continuous

**HLC** **HaLt Continuous**

Función: inhabilitar el modo continuo, de forma tal que los movimientos siguientes sean actuados singularmente.

Parámetros: **(nadie)**

**Ejemplo:**

```
- STC/1
- MOV/X1,100,Y1,50
- MOV/X1,125,Y1,200
- HLC
- MOV/X1,150          ; este movimiento será efectuado solamente una vez concluido el ultimomovimento
                      anterior;
```

**Notas:**

Con la instrucción **HLC** todos los movimientos (entes) pertenecientes a lo continuo son acabados; el programa retoma con la primera instrucción siguiente a **HLC.**

## 8.7 ABC: ABort Continuous

**ABC** **ABort Continuous**

Función: abortar lo continuo

Parámetros: (**nadie)**

**Ejemplo:**

```
        - STC/10
        - MOV/100:Z1,400
L1      - JLT/T1,150,L1
        - ABC                   ; El movimiento de Z1 ha fracasado no en cuanto T1 alcanza el valor 150.
```

**Notas:**

a, Con la instrucción **ABC** lo continuo es fracasado: los ases en movimiento son parados con desaceleración forzada. El programa continúa con la instrucción siguiente a **ABC**

b, En el caso de dos bloques de programa consecutivo que preven movimientos, relativos a más ases, en continuo (STC), hace falta insertar, entre el primer bloque y lo siguiente, una instrucción de movimiento relativo a los ases interesados al primer bloque; si viceversa el eje interesado es un solista, esta inserción no es necesaria.

**Ejemplo:**

```
                - STC
                ...
                ...
                ...
1° BLOQUE       ...
                - MOV/X1,100
                - MOV/Y1,150
                - ABC
                - MOV/X1,n,Y1,n

                - STC
                ...
2° BLOQUE
                ...
                ...
                - ABC
```

## 8.8 CAP: Change Axis Parameter

**CAP** **Change Axis Parameter**

Función: modificar los parámetros de caracterización eje.

Sintaxis

Parámetros: **a:b,c**

dónde:

**a** = numero que localiza el parámetro de caracterización eje sobre que obrar
**b** = eje interesado

**c** = valor de reemplazar a aquél caracterizado

Los valores de **a soy:**

0 = informaciones sobre eje
1 = unused
2 = unused
3 = unused
4 = velocidad' eje
5 = unused
6 = unused
7 = unused
8 = unused
9 = compensación relativa
10 = Over travel positivo
11 = Over travel negativo
12 = Over travel positivo absoluto
13 = Over travel negativo absoluto
14 = unused
15 = KP , proportional gain,
16 = VFF , velocity feed forward,
17 = voltio a DAC
18 = voltio a DAC absoluto
19 = KD , derivativos gain,
20 = KI , complementarios gain,
21 = Max complementarios errores

22 = voltio compensación
23 = KD2 , derivativos 2 gain,
24 = AFF , acceleration feed forward,
25 = Collision Error
26 = sirvo Error
27 = Position Error
28 = prescripción Máxima (Volts)
29 = tiempo de latencia por ALTO COUNTER, Segundo,
30 = tensión máxima por ALTO COUNTER (Volts)
31 = velocidad' máxima por ALTO COUNTER
32 = KS (sordina)
33 = tramos no lineal INCLINACIÓN ACC, variación acc al sec,
34 = tramos no lineal INCLINACIÓN DEC, variación dec al sec,
35 = tramos no lineal TIEMPO INCLINACIÓN ACC
36 = tramos no lineal TIEMPO INCLINACIÓN DEC

38 = KT (predictive)
39 = velocidad' límite del eje
40 = aceleración límite del eje
41 = desaceleración límite del eje
43 = cuota de marker sin correcciones
46 = reducción velocidad' por supero error

50 = activo BUFFER CIRCULAR
51 = desactiva o Resetta BUFFER CIRCULAR
52 = Trigger BUFFER CIRCULAR

55 = CFR, Change FeedRate%,
56 = CSP, Change Speed%,

b1 c1.... valores leídos por los ases
A1 B1.... nombres de los ases
b1 c1.... valores cambiados a los ases

Con 35 36 el TIEMPO INCLINACIÓN tiene el siguiente sentido:
> 0 Scurve (tiempo adicional al tramo lineal)
= 0 lineal
<0 sinusoidal
Si la aceleración y' sinusoidal la desaceleración puede'
ser lineal o sinusoidal pero no Scurve; en éste
caso es forzado la desaceleración sinusoidal.

INFO

CAP/0:X1,4 Fuerza el estado de OMO hecho sobre eje X1 sin borrar la cuota
CAP/0:X1,-4 Sacado el estado de OMO hecho sobre eje X1
CAP/0:X1,128 Activo lo incremental sobre eje X1
CAP/0:X1,-128 Sacado lo incremental de eje X1
CAP/0:X1,256 Activo lo complementario con movimiento sobre eje X1
CAP/0:X1,-256 Sacado lo complementario con movimiento sobre eje X1
CAP/0:X1,512 Fuerza el estado de eje atascado sobre eje X1
CAP/0:X1,-512 Sacado el estado de eje atascado sobre eje X1
CAP/0:X1,4096 Activo la corrección sobre eje X1, si el tablero y' cargada,
CAP/0:X1,-4096 Sacado la corrección sobre eje X1
CAP/0:X1,8192 Activo la gestión sobre eje X1 de "COLLISION ERROR"
CAP/0:X1,-8192 Sacado la gestión sobre eje X1 de "COLLISION ERROR"
CAP/0:X1,16384 Activo la gestión sobre eje X1 de "SERVO ERROR"
CAP/0:X1,-16384 Sacado la gestión sobre eje X1 de "SERVO ERROR"
CAP/0:X1,32768 Activo la gestión sobre eje X1 de "POSITION ERROR"
CAP/0:X1,-32768 Sacado la gestión sobre eje X1 de "POSITION ERROR"
CAP/0:X1,65536 Activo la gestión sobre eje X1 de "ALTO COUNTER ERROR"
CAP/0:X1,-65536 Sacado la gestión sobre eje X1 de "ALTO COUNTER ERROR"
CAP/0:X1,131072 Activo la gestión sobre eje X1 de "DRIVER_OK ERROR"
CAP/0:X1,-131072 Sacado la gestión sobre eje X1 de "DRIVER_OK ERROR"
CAP/0:YM,262144 Intercambiado un máster GANTRY caracterizado con eslavas
CAP/0:YM,-262144 Restablecido como de caracterización el GANTRY

INFO suplementario (100)

CAP/100:X1,1 Activo la gestión del KP de modo SQUARE del eje X1
CAP/100:X1,-1 Puesto la gestión del KP de modo LINEAL del eje X1
CAP/100:X1,2 Activo la gestión del KD de modo SQUARE del eje X1
CAP/100:X1,-2 Puesto la gestión del KD de modo LINEAL del eje X1
CAP/100:X1,4 Activo la gestión del KD2 de modo SQUARE del eje X1
CAP/100:X1,-4 Puesto la gestión del KD2 de modo LINEAL del eje X1
CAP/100:X1,8 Activo la gestión del KI de modo SQUARE del eje X1
CAP/100:X1,-8 Puesto la gestión del KI de modo LINEAL del eje X1
CAP/100:X1,16 Inhabilitado el anticollisione del eje X1
CAP/100:X1,-16 Habilitado el anticollisione caracterizado eje X1
CAP/100:X1,32 Si el eje y' ROLLOVER UNSIGNED se mueve a menor espacio
CAP/100:X1,-32 Si el eje y' ROLLOVER UNSIGNED se mueve a espacio mandado
CAP/100:X1,64 Puesto la aceleración de tipo S-CURVA
CAP/100:X1,-64 Sacado la aceleración de tipo S-CURVA
CAP/100:X1,128 Puesto la desaceleración de tipo S-CURVA
CAP/100:X1,-128 Sacado la desaceleración de tipo S-CURVA
CAP/100:X1,256 Excluido el eje, EJE FUERA DE SERVICIO,
CAP/100:X1,-256 Reinclude el eje (EJE EN SERVICIO)

CAP/100:X1,512 Definido el eje como NORMALMENTE INHABILITADO
CAP/100:X1,-512 Sacado la función NORMALMENTE INHABILITADA
CAP/100:YM,1024 Definido el gantry como gantry rígido
CAP/100:YM,-1024 Repuesto el gantry como "no rígido" si caracterizara
CAP/100:YM,2048 Inhabilitado el sistema gantry, sólo sobre eje máster,
CAP/100:YM,-2048 Rehabilitado el sistema gantry, sólo sobre eje máster,
CAP/100:YM,4096 Permitido el movimiento también sin OMO
CAP/100:YM,-4096 Rehabilitado la prueba sobre OMO hecho, si y' caracterizado,
CAP/100:YM,65536 Definido el eje en pase de plottatura
CAP/100:YM,-65536 Sacado el estado de plottatura
CAP/100:YM,131072 Fuerza el estado marker verdadero, si omo hecho, por debug
CAP/100:YM,-131072 Sacado el estado marker verdadero, si omo hecho, por debug
CAP/100:X1,262444 Inhabilitado la gestión MOV_SWITCH
CAP/100:X1,-262444 Rehabilitado la gestión MOV_SWITCH, si caracterizara,
CAP/100:X1,0x80000 Otro modo para ejecutar HOM/3: , MARKER_VERO saca,
CAP/100:X1,-0x80000 Reconducido él cero sobre el MARKER, MARKER_VERO pone,

NOTA si el eje que es puesto o sacado por la fase de plottatura y'
el máster de un sistema GANTRY, también sobre el eje eslavas viene
ejecutada la función (CAP/100:YM,0x10000)

Por algunos valores de caracterización dando como CÓDIGO
el valor correspondiente en OR con 80H también es modificado
el valor sobre estática; o sea el valor también queda después de REPOSICIÓN
Los códigos son:

| | |
|---|---|
| (15) (143, | KP |
| (19) (147, | KD |
| (23) (151, | KD2 |
| (20) (148, | KI |
| (16) (144, | VFF |
| (24) (152, | AFF |
| (21) (149, | MAX_ERR_I |
| (25) (153, | Collision Error |
| (26) (154, | Sirvo Error |
| (27) (155, | Position Error |
| (28) (156, | Prescripción Máxima (Volts) |

| | |
|---|---|
| (0) (128, | INFO para los siguientes sottocodici |
| (256) | MSK_INTEGRATIVO_MOV |
| (512) | MSK_DYNAMIC_OFFSET |
| (8192) | COLLISION ERROR |
| (26384) | SIRVO ERROR |
| (32768) | POSITION ERROR |

(65536) ALTO COUNTER ERROR
(131072) DRIVER_OK_ERROR

(100) (228, INFO suplementario para los siguientes sottocodici
(1) MSK_KP_SQUARE
(2) MSK_KD_SQUARE
(4) MSK_KD2_SQUARE
(8) MSK_KI_SQUARE
(64) ACELERACIÓN S-CURVA
(128) DESACELERACIÓN S-CURVA
(512) NORMALMENTE INHABILITADO

CAP/9:X1,50 Puesto la compensación relativa sobre eje X1: es desplazado
él cero con respecto de la compensación absoluta de 50 en positivo.
, Si el eje fuera a cero después de esta instrucción se convierte en -50,
Puesto que lo es desplazado cero con respecto de la compensación absoluta,
y no con respecto del cero relativo, cada cambio no tiene
cuenta de aquel precedente.
Luego para volver a cero absoluto basta con dar un cambio
con valor 0.

CAP/50:X1,14,X1,17 Habilita el buffer a memorizar EPSI por X1
y VOLTIO por X1.
En este caso el valor que sigue el eje y' el
código RAV correspondiente. Se pueden tamponar
de 1 a 8 ases. Son válidos los siguientes códigos:
1 = cuotas ases reales
2 = cuotas ases teóricas
6 = cuotas ases reales absolutos
7 = cuotas ases teóricas absolutas
14 = error de interpolación
17 = voltio a DAC
18 = voltio a DAC absoluto
48 = tiempo de muestreo
49 = tiempo usado en muestreo
90 = Speed hardware, P.E. digital pasiva,
91 = Current hardware, P.E. digital pasiva,

CAP/51:X1,0 Inhabilita la memorización sobre buffer

El eje no y' usado.
Si el valor y' >= 0 la memorización del buffer es parado, éste ocurre naturalmente con el buffer lleno,
Si el valor y' <0 el área del buffer viene borrada

CAP/52:X1,1,X1,100 Activa el trigger sobre buffer circular

El eje X1 hace partir la memorización cuando la cuota real y' >= a 100.
CAP/52: tiene que preceder CAP/50:, y viene anulada por CAP/51:
La condición de trigger es hecha sobre un solista eje con los siguientes códigos:

1 = cuotas ases reales
2 = cuotas ases teóricas
6 = cuotas ases reales absolutos
7 = cuotas ases teóricas absolutas
14 = error de interpolación
17 = voltio a DAC
18 = voltio a DAC absoluto
90 = Speed hardware, P.E. digital pasiva,
91 = Current hardware, P.E. digital pasiva,

Si el código y' positivo la condición y' >= al valor; con código negativo la condición y' <= al valor.

## Ejemplo:

- CAP/16:X1,50 Definir por el eje X1 el valor de VFF dioses 50%.

## Ejemplos de CAP y RAV

Controlo abierto a un eje por el calibrado de la compensación dell accionamiento

```
;       ASIGNA UNA TENSIÓN A. EL EJE Z
;       Y VISUALIZA DE ELLO EL VELCITA'
;
        - TSK/VEL_Z
        - LET/L1,0
AA      - KYB/DAMMI VOLTIO Z,L1
        - CPL/-1:Z
        - CAP/17:Z,L1
        - JMP/AA
        - RET


PartProgram[VEL_Z]
AA      -RAV/4:L1,Z
        -DIS/1:VELOCITA' Z,L1
;       -TIM/10
        -JMP/AA
        -RET
```

## 8.9 HMS: Handling Master Slave

**HMS** **Handling Master Slave**

Función: modificar los parámetros de caracterización eje acerca de los parámetros de máster eslavas.

Sintaxis 1

Parámetros: **a,b,c**

dónde:

**a** = eje Máster interesado

**b** = eje eslavas interesadas

**c** = modalidad (0) ± valor, de funcionamiento:
0 significa disassociare el eje eslavas de su Máster
± valor significa asociar el eje eslavas con el Máster definiendo por el eje a eslavas el valor de following plazos.

**Ejemplo 1:**

- HMS/X1,0 Disociar el eje eslavas, caracterizado, del eje Máster X1.

**Ejemplo 2:**

- HMS/X1,Y1,-50 Asociar con el eje Máster X1, el eje eslavas Y1 que seguirá el Máster en dirección opuesta con un valor de Following plazos del 50%.

**Notas**

a, Esta sintaxis, siempre relativa a ases máster-eslavas, permiten sea la modificación de parámetros caracterizados, sea de formar nuevas parejas de ases máster-eslavas.

b, Tienen que ser indicados todos los parámetros; en caso contrario sucede el mensaje de error

c, Si se actúa sobre una pareja de ases caracterizada, los nuevos datos se sustituyen a los datos de caracterización.

d, Si es indicado +2, significa que el eje eslavas incluso siendo asociado con su Máster, no tiene que necesariamente seguirlo a tiempo real según los parámetros definidos; la posición del eje eslavas no es subordinada por lo tanto a la posición del eje máster.
En este caso si el eje eslavas no logra seguir el máster, el eje eslavas llegará cuando pueda, no interviene alguna señal de error.
Si viceversa es indicado -2, significa que el eje eslavas tiene que necesariamente seguir su máster a tiempo real; en caso contrario interviene la emergencia COLLISION.

y, El eje Máster puede ser o **Real** o **R.Q.** mientras el eje eslavas sólo puede ser Real; señal de error en todos los otros casos.

f, Las instrucciones de movimiento relativo a los ases Máster y eslavas pueden hacer parte de un continuo, instrucción STC, pero no puede hacer parte de ello la instrucción HMS ya que pudiera provocar anomalías, en función del momento en que es ejecutada con respecto de las instrucciones de movimiento.

c, Si es solicitada en aquel entonces el disassociazione de un eje eslavas y este no es asociado, interviene el mensaje de ser.

## 8.10 HEC: Handling Electronic Cam

**HEC** **Handling Electronic Cam**

Función: modificar los parámetros de caracterización eje acerca de los parámetros gestión de los dientes de rueda.

Creación del tablero con:

- HEC/0:a,b[,c]:X,d,Y,d,Z,d...
  dónde 0: significa creación tablero
  a y' el número del tablero
  b y' el estado a bit del tablero
  c son el máximo número de rayas del tablero
  d y' el estado absoluto (0) incremental (1) etcétera
  de cada eje
  El primer eje, X, y' el MÁSTER; los otros las ESLAVAS

Carga del tablero con:
- HEC/1:a:X,b,Y,b,Z,b
  dónde 1: significa carga tablero
  a y' el número del tablero
  b y' el valor de asignar a cada eje
  Los ases sin variación pueden no sernos
  El orden entre los ases no y' importante
  El eje máster tiene que siempre ser nosotros porque' la suya
  posición no puede ser igual a la anterior;
  además la dirección tiene que ser concorde en todo
  los puntos de tablero, dirección positiva o rechazo,

Anulación fecha en tablero con:
- HEC/2:a:X,b
  dónde 2: DELETE significa tablero ECM
  a y' el número del tablero
  b usado
  Después de la anulación el tablero tiene que ser cargada
  para poderla ejecutar
  Ya' El tablero no ha dado pero el espacio el suyo
  espacio y' todavía usado.

Start a la elaboración del tablero con:
- HEC/3:a:X,b

dónde 3: EXEC significa tablero ECM
a y' el número del tablero
b con ases máster rollover y' la anticipada
de hacer al ataque del tablero

Notar que para prescindir del tipo de eje, el primero punto del tablero y' siempre absoluto

El diente de rueda cerrado tiene que tener lo primero y el último punto del tablero igual, y el tablero todo cargado.

## 8.11 CFR: Change Feed Rate

**CFR** **Change Feed Rate**

Función: variar el feedrate de un handle en movimiento

Parámetros: **a,b,c**

dónde:

**a** = porcentaje de feedrate, de 0 a 100,
**b** = porcentaje de aceleración, de 0 a 100,
**c** = porcentaje de decellerazione, de 0 a 100,

## 8.12 CPL: Change Position Loop

**CPL** **Change Position Loop**

Función: variar (abrir) cerrar, el loop de posición sobre los ases

Parámetros: **[a:] b1,b2,... bn**

dónde:

**a** = tipo de "loop", switch:)
-2 = sacado sumisión + Enable
-1 = sacado sumisión (DAC)
0 = sacado sumisión (SAC) (Estándard,
1 = repuesto sumisión + Enable
2 = Phase axis

**b1.. n** = nombre del eje
Activando el "loop" -1, DAC, se puede mandar el eje a loop abierto con la instrucción:
- CAP/4:.. , - CAP/17:.. Y - CAP/18:.. .
Activando el "loop" 0, SAC, se puede mandar el eje a loop abierto con la instrucción - SAC /..

**Ejemplo:**

```
- CPL/1:X1        abrir el loop de posición sobre el eje X1
```

**Ejemplo:**

```
                - DRT/1,PR,X1
A               - CPL/-2:X1
                - TIM/100
                - CPL/1:X1
                - TIM/100
                - JMP/A
```

**Nota:**
Si el parámetro **a es** omitido, por estándard asume valor 0.

Cuando es abierto el loop de posición, viene automáticamente forzado 0 como valor de Kv sobre el eje y vienen disabiIitati los varios controles eje, sirvo error, collision, posición y bloque cuenta,; cuando el loop de posición es cerrado, son restablecidos los valores caracterizados.

Cuando es cerrado el loop de posición, es indispensable que la solicitud de movimiento eje ocurra con un retraso de al menos un muestreo con respecto del richiusura loop.

**Ejemplo:**

- CPL/1:X1 richiusura loop de posición
- TIM/2 retraso
- MOV/X1, animo eje

Si interviene una emergencia después de que ha sido abierto un loop de posición, eliminando la condición de emergencia tal loop es cerrado automáticamente.

Las funciones de Reposición cierran automáticamente eventuales loop de posición abiertos anteriormente.

Las solicitudes de abertura loop de posición eje sobre de un eje en movimiento o bien la solicitud de movimiento eje con loop de posición al aire libre, no son aceptadas y engendran el mensaje de anomalía.

## 8.13 PRD: Position ReaD

**PRD** **Position ReaD**

Función: memorizar en las variables las cuotas correspondientes a la posición de los ases.

Parámetros: **[a:] b1,c1,... bn,cn**

dónde:

**a** = switch que determina el tipo de memorización y precisamente:

0 = informaciones sobre eje o sistema
1 = cuotas ases reales
2 = cuotas ases teóricas
3 = servicio por marker, después de OMO,
4 = velocidad' eje
5 = cuotas ases staticizzate, de TCH TPE,
6 = cuotas ases reales absolutos
7 = cuotas ases teóricas absolutas
8 = cuotas ases staticizzate absolutos
9 = compensación relativa
10 = Over travel positivo
11 = Over travel negativo
12 = Over travel positivo absoluto
13 = Over travel negativo absoluto
14 = error de interpolación
15 = KP , proportional gain,
16 = VFF , velocity feed forward,
17 = voltio a DAC
18 = voltio a DAC absoluto
19 = KD , derivativos gain,
20 = KI , complementarios gain,
21 = Max complementarios errores
22 = voltio compensación
23 = KD2 , derivativos 2 gain,
24 = AFF , acceleration feed forward,
25 = Collision Error
26 = sirvo Error
27 = Position Error

28 = prescripción Máxima (Volts)
29 = tiempo de latencia por ALTO COUNTER
30 = tensión máxima por ALTO COUNTER
31 = velocidad' máxima por ALTO COUNTER
32 = KS (sordina)
33 = tramos no lineal INCLINACIÓN ACC, variación acc al sec,
34 = tramos no lineal INCLINACIÓN DEC, variación dec al sec,
35 = tramos no lineal TIEMPO INCLINACIÓN ACC
36 = tramos no lineal TIEMPO INCLINACIÓN DEC
37 = valor de la corrección EJE
Cotiza real = Cuota Encoder + Valor Corrección
38 = KT (predictive)
39 = velocidad' límite del eje
40 = aceleración límite del eje
41 = desaceleración límite del eje
42 = cuota de marker con correcciones
43 = cuota de marker sin correcciones
44 = distancia real de la meta
45 = distancia teórico cálculo, de la meta

50 = lectura nivela hechos en el BUFFER CIRCULAR * *1
51 = lectura nivela libre del BUFFER CIRCULAR * *2

55 = CFR, Change FeedRate%,

100 = informaciones suplementarias sobre eje
101 = informaciones hardware, PRIMEROS ELECTRINICS Status_R,
102 = informaciones hardware, PRIMEROS ELECTRINICS Device_R,
Eje STEPPER valor PULSE, generador de frecuencia,
103 = informaciones hardware, PRIMEROS ELECTRINICS ErrorCode_R,
Eje STEPPER valor FRECUENCIA conseguida con "PULSE"

* *1 el valor vuelto y' 0 o el número de niveles memorizados
* *2 el valor vuelto y' 0 o el número de niveles libres,
si no y' activa la gestión buffer el valor y' -1

b1 c1.... valores leídos por los ases
A1 B1.... nombres de los ases

INFO
valor leído, a bits,:
1H; EJE DE SOLA LECTURA
2H; EJE CON LOOP DE POSICIÓN ABIERTO (DAC)

4H; EJE CON OMO RALEA
8H; EJE EN SERVO ERROR (EJE FIRME) * * Y
10H; EJE EN COLLISION (EJE EN MOVIMIENTO) * * Y
20H; EJE NO EN POSICIÓN * * Y
40H; EJE EN BLOQUE CUENTA * * Y
80H; EJE INCREMENTAL
100H; EJE CON PID COMPLEMENTARIO SOBRE ANIMO
200H; EJE QUE no se TIENE QUE MOVER, UNMOVED,
400H; EJE CON SUMISIÓN ACTIVA
800H; EJE EN DRIVER KO * * Y
1000H; EJE CON CORRECCIÓN ACTIVA
2000H; EJE CON COLLISION ERROR PROBADO
4000H; EJE CON SIRVO ERROR PROBADO
8000H; EJE CON POSITION ERROR PROBADO
10000H; EJE CON ALTO COUNTER ERROR PROBADO
20000H; EJE CON DRIVER VALE ERROR PROBADO
40000H; EJE GANTRY INTERCAMBIADO EN ESLAVAS (MÁSTER)
80000H; EJE CON MOVIMIENTO AGREGADO
100000H; EJE CON ANTICOLLISION PERIÓDICO
200000H; EJE CON MOVIM. PARADO POR ANTIC. O CFR/0
400000H; EJE EN ANTICOLLISION
800000H; EJE EN DEADLOCK

Si el nombre eje y' SYSTEM
valor leído, a bits,:
1H; LECTURA ASES ACTIVA
2H; SUMISIONES ACTIVAS
4H; RUN DE PART PROGRAMA
8H; CYCLE START
10H; CYCLE ALTO
20H; EMERGENCIA
40H; SISTEMA EN RIPARTENZA
80H; SISTEMA EN REPOSICIÓN
100H; SISTEMA CON AL MENOS UN EJE CON DRIVER KO
200H; SISTEMA CON ASES EN ANTICOLLISIONE O DEADLOCK

INFO suplementario (100)
valor leído, a bits,:
1H; EJE CON KP TIPO SQUARE
2H; EJE CON KD TIPO SQUARE
4H; EJE CON KD2 TIPO SQUARE
8H; EJE CON KI TIPO SQUARE
10H; EJE CON ANTICOLLISIONE INHABILITADA

20H; EJE ROLLOVER UNSIGNED CON ANIMO A. MENOR ESPACIO
40H; EJE CON ACELERACIÓN S-CURVA
80H; EJE CON DESACELERACIÓN S-CURVA
100H; EJE FUERA DE SERVICIO
200H; EJE NORMALMENTE INHABILITADO
400H; GANTRY QUE NO SIGUE (GANTRY RÍGIDO)
800H; GANTRY INHABILITADO
1000H; MOVIMIENTO PERMITIDO TAMBIÉN SIN OMO

INFO hardware (101)
valor leído: Como de lámina inteligente

3 = servicio por marker
El valor extraído tiene los siguientes sentidos:
1, después de un OMO con micro y marker, la distancia de lo micro del marker.
2, después de un OMO sin micros e/o sin marker, la posición dónde el eje fue borrado
3, después de un OMO/8: o un TCH/8: la posición del marker; tal valor y' típicamente cero exacto, a menos que no él sea hecha anteriormente un OMO sobre marker.
El no hallar cero significa pérdida de cuenta.

**de b1 a bn** = variables, solista locales o globales, en los que se quiere memorizar las cuotas

**de c1 a cn** = ases interesados

**Ejemplo:**

- PRD/1:L1,X1,G3,Y1 Memorizar en la variable local L1 la cuota correspondiente al eje X1 y en la variable global 3 la cuota correspondiente al eje Y1.

**Notas:**

a, Si el parámetro **a es** omitido, por estándard asume valor 0.

b, Por cada instrucción es posible memorizar un máximo de 8 cuotas.

c, Las cuotas memorizadas siempre son **absolutas** pero referidas a la situación anterior la instrucción **PRD,** referidas a la cero máquina o a eventuales platós point.

d, Si el eje interesado está en movimiento, es memorizada la cuota de paso, es decir aquel existente al momento de la ejecución de **PRD.**

y, La memorización de las cuotas relativas a la lámina transductores, parámetro **a** = 5, tiene que ser efectuada es decir después de una ejecución de la instrucción **TCH** o **TPE después de que ha** sido ejecutada un strobbatura de las cuotas de parte del tastatore; en caso contrario serían memorizadas es decir cuotas falsas las últimas cuotas staticizzate por el tastatore de la lámina.

f, La memorización de las cuotas teóricas de los ases, parámetro **a** = 2, puede ser utilizada para ejecutar sobre el vídeo un plottatura del recorrido teórico efectuado por los ases con *LOS* sin los ases que se mueven realmente; está claro que si los ases no se mueven, tienen que ser inhabilitados los siguientes controles: SIRVO ERROR, COLLISION, ALTO COUNTER, K.O. POSITION.

## 8.14 RAV: Read Axis Value

**RAV** **Read Axis Value**

Función: memorizar en las variables las cuotas correspondientes los parámetros de los ases.

Parámetros: **a:b1,c1,... bn,cn**

dónde:

**a** = switch que determina el tipo de memorización y precisamente:

0 = informaciones sobre eje o sistema
1 = cuotas ases reales
2 = cuotas ases teóricas
3 = servicio por marker, después de OMO,
4 = velocidad' eje
5 = cuotas ases staticizzate, de TCH TPE,
6 = cuotas ases reales absolutos
7 = cuotas ases teóricas absolutas
8 = cuotas ases staticizzate absolutos
9 = compensación relativa
10 = Over travel positivo
11 = Over travel negativo
12 = Over travel positivo absoluto
13 = Over travel negativo absoluto
14 = error de interpolación
15 = KP , proportional gain,
16 = VFF , velocity feed forward,
17 = voltio a DAC
18 = voltio a DAC absoluto
19 = KD , derivativos gain,
20 = KI , complementarios gain,
21 = Max complementarios errores
22 = voltio compensación
23 = KD2 , derivativos 2 gain,
24 = AFF , acceleration feed forward,
25 = Collision Error
26 = sirvo Error
27 = Position Error
28 = prescripción Máxima (Volts)

29 = tiempo de latencia por ALTO COUNTER
30 = tensión máxima por ALTO COUNTER
31 = velocidad' máxima por ALTO COUNTER
32 = KS (sordina)
33 = tramos no lineal INCLINACIÓN ACC, variación acc al sec,
34 = tramos no lineal INCLINACIÓN DEC, variación dec al sec,
35 = tramos no lineal TIEMPO INCLINACIÓN ACC
36 = tramos no lineal TIEMPO INCLINACIÓN DEC
37 = valor de la corrección EJE
Cotiza real = Cuota Encoder + Valor Corrección
38 = KT (predictive)
39 = velocidad' límite del eje
40 = aceleración límite del eje
41 = desaceleración límite del eje
42 = cuota de marker con correcciones
43 = cuota de marker sin correcciones
44 = distancia real de la meta
45 = distancia teórico cálculo de la meta
46 = reducción velocidad' por supero error
47 = pasos perdidos por el último Homing si activara el control sobre el encoder
50 = lectura nivela hechos en el BUFFER CIRCULAR * *1
51 = lectura nivela libre del BUFFER CIRCULAR * *2

55 = CFR, Change FeedRate%,
56 = CSP, Change Spindle%,

60 = paso Encoder Caracterizado

100 = informaciones suplementarias sobre eje

101 = informaciones hardware
PRIMEROS ELECTRINICS = Status_R
NEUTONIO = ENCODER_NEWTON
PCI2000 = ENCODER_SIX
SERCOS = ENCODER_SERCOS
YASKAWA = ENCODER_YASKAWA
ETEL = ENCODER_ETEL

102 = informaciones hardware
PRIMEROS ELECTRINICS = Device_R
NEUTONIO = ENCODER_MARKER
PCI2000 = ENCODER_MARKER
SERCOS = ERROR_SERCOS

YASKAWA = ERROR_YASKAWA
ETEL = ERROR_ETEL
Eje STEPPER = Valor PULSE, generador de frecuencia,

103 = informaciones hardware
PRIMEROS ELECTRINICS = ErrorCode_R
NEUTONIO = QUANTI_MARKER
PCI2000 = QUANTI_MARKER
SERCOS = DAC_SERCOS
YASKAWA = DAC_YASKAWA
ETEL = DAC_ETEL
Eje STEPPER = Valor FRECUENCIA conseguida con "PULSE"

* *1 el valor vuelto y' 0 o el número de niveles memorizados
* *2 el valor vuelto y' 0 o el número de niveles libres,
si no y' activa la gestión buffer el valor y' -1

b1 c1.... valores leídos por los ases
A1 B1.... nombres de los ases

INFO
valor leído, a bits,:
1H; EJE DE SOLA LECTURA
2H; EJE CON LOOP DE POSICIÓN ABIERTO (DAC)
4H; EJE CON OMO RALEA
8H; EJE EN SERVO ERROR (EJE FIRME) * * Y
10H; EJE EN COLLISION (EJE EN MOVIMIENTO) * * Y
20H; EJE NO EN POSICIÓN * * Y
40H; EJE EN BLOQUE CUENTA * * Y
80H; EJE INCREMENTAL
100H; EJE CON PID COMPLEMENTARIO SOBRE ANIMO
200H; EJE QUE no se TIENE QUE MOVER, UNMOVED,
400H; EJE CON SUMISIÓN ACTIVA
800H; EJE EN DRIVER KO * * Y
1000H; EJE CON CORRECCIÓN ACTIVA
2000H; EJE CON COLLISION ERROR PROBADO
4000H; EJE CON SIRVO ERROR PROBADO
8000H; EJE CON POSITION ERROR PROBADO
10000H; EJE CON ALTO COUNTER ERROR PROBADO
20000H; EJE CON DRIVER VALE ERROR PROBADO
40000H; EJE GANTRY INTERCAMBIADO EN ESLAVAS (MÁSTER)
80000H; EJE CON MOVIMIENTO AGREGADO
100000H; EJE CON ANTICOLLISION PERIÓDICO
200000H; EJE CON MOVIM. PARADO POR ANTIC. O CFR/0
400000H; EJE EN ANTICOLLISION
800000H; EJE EN DEADLOCK
1000000H; EJE CON SUMISIÓN EN CORRIENTE
2000000H; EJE EN DRIVER FAULT
4000000H; EJE EN OVER TRAVEL POSITIVO, SWITCH FIN CARRERA +)
8000000H; EJE EN OVER TRAVEL NEGATIVO, SWITCH FIN CARRERA -)

Si el nombre eje y' SYSTEM
valor leído, a bits,:
1H; LECTURA ASES ACTIVA
2H; SUMISIONES ACTIVAS
4H; RUN DE PART PROGRAMA
8H; CYCLE START
10H; CYCLE ALTO
20H; EMERGENCIA
40H; SISTEMA EN RIPARTENZA
80H; SISTEMA EN REPOSICIÓN
100H; SISTEMA CON AL MENOS UN EJE CON DRIVER KO
200H; SISTEMA CON ASES EN ANTICOLLISIONE O DEADLOCK

INFO suplementario (100)
valor leído, a bits,:
1H; EJE CON KP TIPO SQUARE
2H; EJE CON KD TIPO SQUARE
4H; EJE CON KD2 TIPO SQUARE
8H; EJE CON KI TIPO SQUARE
10H; EJE CON ANTICOLLISIONE INHABILITADA
20H; EJE ROLLOVER UNSIGNED CON ANIMO A. MENOR ESPACIO
40H; EJE CON ACELERACIÓN S-CURVA
80H; EJE CON DESACELERACIÓN S-CURVA
100H; EJE FUERA DE SERVICIO
200H; EJE NORMALMENTE INHABILITADO
400H; GANTRY QUE NO SIGUE (GANTRY RÍGIDO)
800H; GANTRY INHABILITADO
1000H; MOVIMIENTO PERMITIDO TAMBIÉN SIN OMO
2000H; EJE MOVIDO EN ACELERACIÓN
4000H; EJE MOVIDO EN DESACELERACIÓN
8000H; EJE MOVIDO A. VELOCIDAD' CONSTANTE
10000H; EJE EN PLOTTATURA
20000H; EJE CON MARKER AUTÉNTICO
40000H; EJE CON GESTIÓN MOV_SWITCH INHABILITADA

INFO hardware (101)
valor leído: Como de lámina inteligente

3 = servicio por marker
El valor extraído tiene los siguientes sentidos:
1, después de un OMO con micro y marker, la distancia de lo micro del marker.
2, después de un OMO sin micros e/o sin marker, la posición dónde el eje fue borrado
3, después de un OMO/8: o un TCH/8: la posición del marker; tal valor y' típicamente cero exacto, a menos que no él sea hecha anteriormente un OMO sobre marker.
El no hallar cero significa pérdida de cuenta.

**de b1 a bn** = variables, solista locales o globales, en los que se quiere memorizar las cuotas

**de c1 a cn** = ases interesados

**Ejemplo:**

- RAV/1:L1,X1,G3,Y1 Memorizar en la variable local L1 la cuota correspondiente al eje X1 y en la variable global 3 la cuota correspondiente al eje Y1.

NOTAS SOBRE LA GESTIÓN PÉRDIDA STEP ENCODER

Con los ases NEWTON y PCI 2000 se pueden recobrar las pérdidas de steps
del encoder utilizando los siguientes campos de caracterización
StepsEncoder=4000
empujados por vuelta encoder. usado para arreglar la posición
si en la vuelta se perdonan steps
MaxStepsLost=100
empujados por vuelta encoder que pueden ser perdidos
MarkerSteps=2
Ancho en impulsos del MARKER
Con 0x200 en Kind_axis_0 (reparación en una sola dirección) el
valor >=0 significa en dirección positiva; <0 en dirección negativa.
Si la pérdida paso en la vuelta supera el valor de MaxStepsLost =
es marcado el error y sacada sobre el eje la condición de cero hecho.
Con la sumisión activo la señal se pone efectiva y queda
hasta a la ejecución de la primera reposición. Una nueva señal de pérdida
paso excesivo será diagnosticado después de la puesta a cero del eje que
tiene que ser rehecho.
Se pueden leer el número acumulativo de steps perdidos con la instrucción
RAV/47:L1,X011
En L1 es cargada la distancia persa/aggiunta sobre la posición encoder
en Unida' De Medida
Cargando una variable con RAV/60:G1,Z011 y' posible tener el número
de steps perdido dividendo el valor conseguido con RAV/47: con éste
variable
Este valor viene resettato de una puesta a cero eje
Utilizando la instrucción RAV/102:L1,X011 se puede visualizarlo última
posición encoder del MARKER, en steps encoder,
Utilizando la instrucción RAV/103:L1,X011 se puede visualizar cuanto
vueltas el MARKER y' estado encontrado. La cuenta y' resettato de
una puesta a cero eje.
También con la función pérdida paso no activa él pueden utilizar
la instrucción RAV/102 RAV/103 que activan la fase de carga
marker. Con reposición esta fase es eliminada.
Y' por lo tanto posible hacer PART-PROGRAM que averiguan si el MARKER
y' presente y el control del número de steps por vuelta encoder
además de' el ancho del marker.

E.G.

```
II          - RAV/103:L3,A          ; Leyes Num. MARKER
AA          - RAV/103:L1,A          ; Leyes Num. MARKER
            - JEQ/L1,L3,AA
            - DIS/4:MARKER ENCONTRADO, L1
            - RAV/102:L2,A          ; Leyes Cuota MARKER
BB          - RAV/103:L3,A          ; Leyes Num. MARKER
            - JEQ/L3,L1,BB
            - DIS/4:MARKER ENCONTRADO, L3
```

```
- RAV/102:L4,A                          ; Leyes Cuota MARKER
- DIS/5:Numero STEPS, abs(L4-L2,
- JMP/II
- END
```

Este part program puede dar el número de steps por vuelta si el encoder rueda, no muy velozmente, siempre de la misma parte o el ancho del marker si el encoder oscila a caballo del mismo.

## 8.15 RSV: Read Speed Value

**RSV** **Read Speed Value**

Función: lee las vueltas de los mandriles, de 1 a 8 mandriles,

Parámetros: **a:b1,c1…. bn,cn**

dónde:

**a** = tipo de parámetro de leer, switch:)
0 = informaciones sobre mandril
1 = vueltas mandril reales
2 = vueltas mandril teóricos
3 = Volts fecha al mandril

INFO
valor leído, a bits,:
1H; MANDRIL EN ROTACIÓN
2H; MANDRIL A. RÉGIMEN

**b1.. n** = variables, solista locales o globales, en los que se quiere memorizar el valor de rotación

**c1.. n** = mandriles interesados

**Ejemplo:**

- RSV/L1,M1 Memorizar en la variable local L1 la cuota correspondiente al mandril M1

## 8.16 SFP: Set Feed Profile

**SFP** **Set Feed Profile**

Función: define un valor genérico de feed, aceleración y desaceleración que utilizar sobre las instrucciones de movimiento que siguen.

Parámetros: **a,b,c**

dónde:

**a** = valor máximo de velocidad' sobre el perfil
**b** = valor máximo de aceleración sobre el perfil
**c** = valor máximo de desaceleración sobre el perfil

**Ejemplo:**

```
- SFP/500
- MOV/X1,100,Y1,100     Todas este instrucciones, a excepción del MOV/200:X1 300, viene
- MOV/200:X1,300        ejecutáis con un feed de 500.
- MOV/X1,400
- MOV/1000:X1 ,500
…
…
```

**Notas:**

a, Cada función **SFP** anula la anterior y queda válida hasta la prójima.

b, Para anular la función **SFP** hace falta definirla con parámetro **a** = cero; las funciones de Reposición anulan eventuales **SFP** activos.

c, Los movimientos que siguen la instrucción **SFP**, son ejecutados con el valor de feed menor entre: aquel indicado en la instrucción SFP; aquel eventualmente indicado en las instrucciones de movimiento, en el ejemplo sobre citado el MOV/200:X1,300,; y aquel posible en función de los ases de mover, vector formado de más ases con datos diferentes entre ellos; espacio de recorrer diferente entre las botaduras ases, etcétera.

## 8.17 SPD: SPeeD

**SPD Speed**

Función: impone la velocidad de rotación de un mandril (SPeeD).

Parámetros: **a,b,[c,d]**

dónde:

**a** = número o nombre mandril
**b** = valor en vueltas al minuto de asignar al mandril
**c** = aceleración en vueltas al sec2, 0 = max,
**d** = desaceleración en vueltas al sec2, 0 = max,

**Ejemplo:**

```
- SPD/M1,11000
```

**Notas:**

La rotación del mandril y' lanzada con SPD/M1,giri
dónde M1 y' el nombre simbólico del mandril y vueltas son giri/min.
Hay algunos I/O que administran la rotación.

Number_ENABLE_switch_port = "Enable Spindle ZB" / / Salida por abilit. mandril "OUT11"
FalseTrue_logic_ENABLE_switch=1 / / Salida por capacitación mandril

Number_DRVOK_switch_port = "/ / Input por mandril VALE
FalseTrue_logic_DRVOK_switch=2 / / Input por mandril VALE

Number_STOP_switch_port = "/ / Input por mandril firme
FalseTrue_logic_STOP_switch=2 / / Input por mandril firme

Number_LOCK_switch_port = "/ / Salida por mandril ATASCADO
FalseTrue_logic_LOCK_switch=2 / / Salida por mandril ATASCADO

Number_RUN_switch_port = "/ / Input por mandril a RÉGIMEN
FalseTrue_logic_RUN_switch=2 / / Input por mandril a RÉGIMEN

## 8.18 TCH: TouCH

**TCH TouCH**

Función: ejecutar una prueba sobre el tastatore de las láminas transductores durante un movimiento

Parámetros: **a,b:c:d,e:f**

dónde:

**a** = Eje movido
**b** = Posición final por NO TOUCH
**c** = Espacio max de parada
**d** = Global / Local por staticizzazione cotiza touch
**y** = Eje staticizzato
**f** = Label de salto por touch hecho

### Ejemplo de TCH

```
        - DIS/1:>
        - DIS/2:>
PRINCIPIO       -
        - MOV/Y,0
        - TCH/Y,100:10:L1,Y:TOCCATO
        - DIS/1:NON TOCADO EN IDO
        - TMM/2000
        - JMP/RITORNO
TOCADO -
        - DIS/1:VALORE, L1
        - TMM/2000
VUELTA -
        - TCH/Y,0:1:L1,Y:SENTITO
        - DIS/2:NON OÍDO EN VUELVO
        - TMM/2000
        - JMP/INIZIO
OÍDO -
        - DIS/2:VALORE, L1
        - TMM/2000
        - JMP/INIZIO
```

## 8.19 TMT: Test Movement Transducer

**TMT** **Test Movement Transducer**

Función: ejecutar por fuerza una prueba sobre los transductores, input analógicos, durante un movimiento.

Parámetros: **[a]:a1.. n,b1.. n:c:d,e:f:g**

dónde:
**a** = Velocidad vectorial
**a1.. n** = Número o nombre eje
**b1.. n** = Posición final si no toca
**c** = Espacio de parada
**d** = Número / Nombre transductor
**y** = Valor transductor
**f** = 1 (>= al valor, 0 (<= al valor,
**g** = Label de salto por tocada ralea

; TMT a Transductor. TMT/A1,b1:SpaceStop:C1,Valore:d1:LABEL
A1 = Eje movido
b1 = Posición final si no toca
SpaceStop = Espacio de parada
C1 = Número / Nombre transductor
Valor = Valor transductor
d1 = 1 (>= al valor, 0 (<= al valor,
LABEL = Label de salto por tocada ralea

; TMT a Golpe TMT/A1,b1:Time:A1,Volt:d1:LABEL
A1 = Eje movido
b1 = Posición final si no toca
SpaceStop = Tiempo de golpe en segundo
C1 = Eje probado, igual a eje movido
Valor = Tensión en Voltio
d1 = no usado y considerado 1
LABEL = Label de salto por tocada ralea

## 8.20 TMS: Test Movement Sensor

**TMS** **Test Movement Sensor**

Función: ejecutar una prueba sobre los sensores, input digitales, durante un movimiento

Sintaxis 1

Parámetros: **[a]:a1.. n, b1.. n:c:d,e:f**

dónde:

**a** = Velocidad vectorial
**a1.. n** = Número o nombre eje
**b1.. n** = Posición final si no encontrara switch
**c** = Espacio de parada
**d** = Número / Nombre del digital input / Nombre del digital salida
**y** = 1, digital input on, 0, digital input off,
**f** = Label de salto por micro encontrado

; TMS/A1,b1:SpaceStop:Switch,Tipo:LABEL
A1 = Eje movido
b1 = Posición final si no encontrara switch
SpaceStop = Espacio max de parada
Switch = Número / Nombre del digital input
Tipo = 1, digital input on, 0, digital input off,
LABEL = Label de salto por micro encontrado

**Ejemplo:**

```
- HOM/2:REV041

- TMS/REV041,-720:0.01:I406,0:TrPTO
- LET/L1,-97
- JMP/ErroreBloccoRev

TrPTO -
```

## 8.21 TPE: Touch Probe Enable

**TPE** **Touch Probe Enable**

Función: ejecutar una prueba sobre los tastatori de las láminas transductores sin efectuar ningún movimiento

Parámetros: **a1.. n,b1.. n**

dónde:

**a1.. n** = Global / Local por staticizzazione cotiza touch
**b1.. n** = Número o nombre eje

**Ejemplo:**

; TPE/G1,X1,G2,Y1,....

## 8.22 SZP: Set Zero Point

**SZP** **Set Zero Point**

Función: permite definir los orígenes de los sistemas de referencia de los ases.

Parámetros: **a:b1,c1,.. bn,cn**

dónde:

**a** = n° de origen, de 1 a 32,

**de b1 a bn** = ases interesados

**de c1 a cn** = coordenadas, es decir distancia entre la cero máquina de cada eje y el origen.

**Ejemplo:**

```
SZP/1:X1 ,50,Y1 ,75.43,Z, -4.3
```

**Notas:**

a, Los ases interesados también pueden pertenecer a más bracci físicos, ven el conocido g de la instrucción **LZP.**

b, Si los parámetros son inmediatos y no variables, la instrucción **SZP se** convierte en después ejecutivo immedia-tamente [Enter] sea en TEACH que en EDIT, y no hace falta ejecutar el part program,; en caso contrario es necesario ejecutar el part program, una sola vez, después de TEACH o después de EDIT.

c, Cada **SZP** anula la anterior conmigo mismo número, aunque los ases interesados son diferentes.

d, Y conveniente programar todos las **SZP** relativos al robot agrupados sobre un mismo part-program.

e, Con cada instrucción **SZP** es posible obrar sobre un máximo de 7 ases.

## 8.23 LZP: Load Zero Points

**LZP** **Load Zero Points**

Función: permite programar con base en los varios orígenes definidos por la instrucción SZP

Parámetros: **a**

dónde:

**a** = n° del origen interesado, de 1 a 32,

**Ejemplo:**

```
LZP/1
MOV/X1,-10.5
```

El coordinado X1 resultante será 39,5 mm, equivalente a 50 mm de la instrucción SZP pego 10,5 mm del movimiento.

**Notas:**

a, Cada función **LZP** anula la anterior y queda válida hasta la prójima.

b, Las cuotas resultantes después de la instrucción **LZP** serán iguales a la suma algebraica de las coordenadas del movimiento y a aquellos del origen.

c, **LZP puede** ser anulada por el mando de REPOSICIÓN

d, Si no se preve alguno **LZP** todas las coordenadas se refieren a los ceros maquina (ver el conocido g).

y, Si **LZP es** introducido sin el relativo **SZP** comparece el mensaje de error **CERO POINT NOT PLATÓ.**

f, Si dopo un **LZP/n es** movido un eje no le indicado en el relativo **SZP/n**, este eje le será referido al **LZP** anterior.

g, Si en una misma instrucción **SZP es** indicado ases no riguardanti el mismo brazo físico, pueden averiguarse inconvenientes durante los trabajos multitask, task paralelos, es por lo tanto cuanto importante tener presente sigue:

- la primera instrucción **LZP** ejecutado sobre uno cualquiera de los task hace ejecutivos todos los deltas relativos a los ases indicados en el **SZP** correspondiente.

- por consiguiente el **LZP** activa sobre los otros task no es aquella prevista; son previsibles las consecuencias en caso de cambio **LZP.**

Por eso se aconseja indicar ases relativos a más bracci físicos sobre el mismo SZP sólo cuando éste sean animados por el mismo task.

## 8.24 PIN: Position Incremental(INQ)

**PIN (INQ)** **Position Incremental (INcremental Cuotas)**

Función: habilitar el sistema a efectuar los desplazamientos con cuotas incrementales.

Parámetros: **a1,a2,... an**

dónde:

**de a1 a an** = ases que se quieren programar de modo incremental, máximo 16,

**Ejemplo:**

- PIN/X1,Z2 habilitar el sistema a recibir cuotas incrementales sobre los ases X1 y Z2

**Notas:**

a, Los restos de las operaciones de conversión métrica no son perdidos pero considerados en los movimientos siguientes.

La instrucción **PIN** es válida hasta que no es anulada por **PAB,** relativa a los mismos ases o a mando de REPOSICIÓN

## 8.25 PAB: Position Absolute(ABQ)

**PAB(ABQ)** **Position Absolute (ABsolute Cuotas)**

Función: inhabilitar lo incremental de modo que los movimientos siguientes vengan de nuevo referidos al cero de salida.

Parámetros: **a1, a2,... an**

dónde:

**de a1 a an** = ases que se desea reprogramar en absoluto, máximo 16,

**Ejemplo:**

```
- PAB/X1      ; Las coordenadas relativas al eje X1 que siguen esta instrucción serán de nuevo en absoluto.
```

## 8.26 MMA: Move Manual Axis

**MMA** **Move Manual Axis**

Función: Mueve un eje con un movimiento manual, Move Manual Axis,

Parámetros: **a,b,c**

dónde:

| | | |
|---|---|---|
| **a** | = | 0 punta el movimiento, 1 él activo |
| **b** | = | eje interesado |
| **a** | = | porcentaje de velocidad del movimiento |

**Ejemplo:**

```
;
; Gestión JOG
;
        - RAI/L30,AD1; potenciómetro ases
        - JGE/L30,0,ADAOK
        - LET/L30,-L30
ADAOK -
        - LET/L30,L30/16300*30
        - JLE/L30,30,ADA_1OK
        - LET/L30,30
ADA_1OK -

MM_Z011 -
        - IDI/IN6,1,MuoviZ11Piu
        - IDI/IN7,1,MuoviZ11Meno
FermoAsseZ11 -
        - JEQ/L31,0,MM_Z021
        - MMA/0:Z011
        - SDO/M1.4,0
        - LET/L31,0
        - JMP/MM_Z021
MuoviZ11Piu -
        - JNE/and(G14011,2),2,MM_Z021
        - JEQ/L31,1,MM_Z021
        - SDO/M1.4,0
        - MMA/1:Z011,L30
        - LET/L31,1
        - JMP/MM_Z021
MuoviZ11Meno -
        - JNE/and(G14011,2),2,MM_Z021
        - JEQ/L31,-1,MM_Z021
```

```
        - SDO/M1.4,0
        - MMA/1:Z011,-L30
        - LET/L31,-1
        - JMP/MM_Z021
MM_Z021 -
        - IDI/IN4,1,MuoviZ21Piu
        - IDI/IN5,1,MuoviZ21Meno
FermoAsseZ21 -
        - JEQ/L32,0,MM_Z
        - SDO/M1.4,0
        - MMA/0:Z021
        - SDO/M1.4,0
        - LET/L32,0
        - JMP/MM_Z
MuoviZ21Piu -
        - JNE/and(G14021,2),2,MM_Z
        - JEQ/L32,1,MM_Z
        - SDO/M1.4,0
        - MMA/1:Z021,L30
        - LET/L32,1
        - JMP/MM_Z
MuoviZ21Meno -
        - JNE/and(G14021,2),2,MM_Z
        - JEQ/L32,-1,MM_Z
        - SDO/M1.4,0
        - MMA/1:Z021,-L30
        - LET/L32,-1
        - JMP/MM_Z
MM_Z -
MM1 -
        - JMP/a
;
; No en Manual Parar los ases si en movimiento
;
NoManuale -

NO_Z011 -
        - JEQ/L31,0,NO_Z021
        - MMA/0:Z011
        - SDO/M1.4,0
        - LET/L31,0
        - JMP/MM_Z021
NO_Z021 -
        - JEQ/L32,0,NO_Z
        - SDO/M1.4,0
        - MMA/0:Z021
        - SDO/M1.4,0
        - LET/L32,0
        - JMP/NO_Z
NO_Z -
```

## 8.27 OPT: Open PoinT file

**OPT** **Open PoinT file**

Función: Abre un expediente de puntos, Open PoinT expediente,

Parámetros: **a**

dónde:

**a** = Nombre expediente de puntos

**Ejemplo:**

```
- OPT/Point.pt
```

## 8.28 MOR: MORe

**MOR MORe**

Función: mover los ases de la máquina "en rápido", a la velocidad vectorial indicada, el vector es definido por los ases Precisados, si indicara una salida lo pone en la condición requerida, la señal de cumplimiento del movimentazione es adelantada al logro de la distancia por el punto final.

Sintaxis 1

Parámetros: **[a:] a1,b1,... bn:c:[d,e]**

dónde:

| | | |
|---|---|---|
| **a** | = | velocidad vectorial |
| **de a1 a an** | = | ases interesados, ven conocido b, |
| **de b1 a b** | = | coordenadas |
| **c** | = | distancia del punto final |
| **d** | = | nombre o número de una salida digital |
| **y** | = | valor a 1 u O de la salida digital |

Sintaxis 2

Parámetros: **[a:] Pb:c:[d,e]**

dónde:

| | | |
|---|---|---|
| **a** | = | velocidad vectorial |
| **Pb** | = | número del punto interesado |
| **c** | = | distancia del punto final |
| **d** | = | nombre o número de una salida digital |
| **y** | = | valor a 1 u O de la salida digital |

**Ejemplo:**

| | |
|---|---|
| - MOR/5000:X1,50.5,Y1 ,75:1 | Ejecutar un movimiento con velocidad 5 m/min sobre el vector X,Y a las cuotas indicadas, con antelación de señal a 1 mm del fin. |
| - MOR/5000:X1,50.5,Y1 ,75:1:12,1 | Ejecutar un movimiento con velocidad 5 m/min sobre el vector X,Y a las cuotas indicadas, al final del movimiento es puesto al valor 1 la salida 12, con antelación de señal a 1 mm del fin. |

| | |
|---|---|
| - MOR/5000:P10:1 | Ejecutar un movimiento relativo a los ases, y a las coordenadas, indicadas en el punto P10 con velocidad de 5 m/min, con antelación de señal a 1 mm del fin. |

**Notas:**

Para esperar el cumplimiento del movimiento utilizar la instrucción **HLC.**

**Ejemplo:**

```
- MOR/5000:X1,50.5,Y1 ,75:25
- SDO/output21,1        ; viene settato a 1il señal "output21" a 25mm del fin del movimiento
- HLC                   ; espera el fin del movimiento
```

## 8.29 DCT: Deep Control Touch

**DCT** **DeepControlTouch**

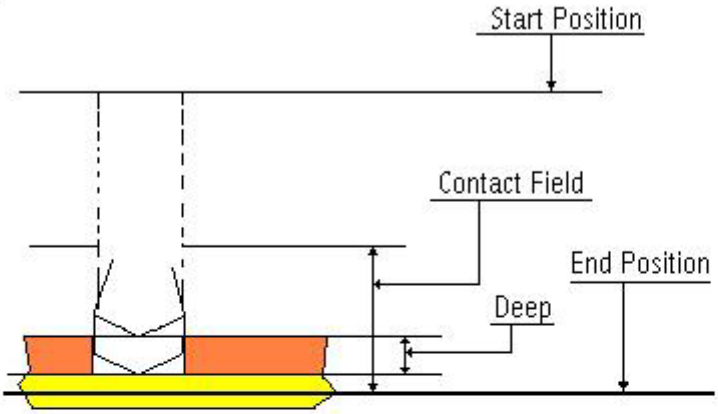
Función: mover los ases de la máquina, a la velocidad vectorial indicada, el vector es definido por los ases Precisados, el contacto de tastatura, cambia la velocidad del movimiento y la posición final que será definida por el valor que tuvo al contacto más un delta programado.

Sintaxis 1
Parámetros: **[a:] a1,b1.an,bn:Deep,Field,Feed:G1:LABEL**

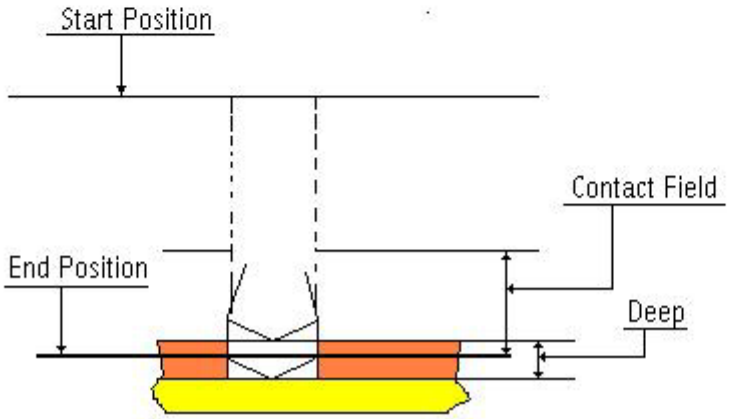
dónde:

| | | |
|---|---|---|
| **a** | = | Velocidad vectorial |
| **de a1 a an** | = | Ases interesados |
| **de b1 a bn** | = | Posición final por NO TOUCH |
| **Deep** | = | Profundidad' de pinchazo de la cuota palpada, si en campo, |
| **Field** | = | Campo de tastatura, referida a b1, |
| **Feed** | = | Feed de tastatura (aquel de pinchazo y aquel estándar) |
| **G1** | = | Global / Local por estado tastatura<br>-2 = tastatura después del campo de sondeo (SURFACE CONTROL)<br>-1 = tastatura antes del campo de sondeo , DEEP CONTROL Y SURFACE CONTROL,<br>0 = tastatura correcto y pinchazo ejecutado<br>1 = NO TOUCH |
| **LABEL** | = | Label de salto por contacto encontrado |

Mode A




Mode B




**Ejemplo:**

```
        - SFP/0,0,0
        - MOV/Z1,0
        - SFP/2500,0,0
        - DCT/Z1,0.5:0.8,0.2,500:G1:FRKO
        - SFP/0,0,0
        - MOV/Z1,0
        - JMP/EXIT
FRKO    - JLT/G1,0,TRUC
        - DIS/NO TOUCH
        - JMP/EXIT
TRUC - DIS/TRUCIOLO
EXIT - RET
```

## 8.30 DCS: Deep Control Sensor

**DCS** **DeepControlSensor**

Función: mover los ases de la máquina, a la velocidad vectorial indicada, el vector es definido por los ases Precisados, el contacto de una señal digital, cambia la velocidad del movimiento y la posición final que será definida por el valor que tuvo al contacto más un delta programado.

Sintaxis 1
Parámetros: [a:] a1,b1.an,bn:Deep,Field,Feed:Input Digitale,Valore:G1:LABEL
dónde:

**a** = Velocidad vectorial
**de a1 a an** = Ases interesados
**de b1 a bn** = Posición final si no encontrara switch
**Deep** = Profundidad' de pinchazo de la cuota palpada, si en campo,
**Field** = Campo de tastatura, referida a b1,
**Feed** = Feed de tastatura (aquel de pinchazo y aquel estándar)
**Input digital** = Número / Nombre del digital input
**Valor** = 1, digital input on, 0, digital input off,
**G1** = Global / Local por estado tastatura
-2 = tastatura después del campo de sondeo (SURFACE CONTROL)
-1 = tastatura antes del campo de sondeo , DEEP CONTROL Y SURFACE CONTROL,
0 = tastatura correcto y pinchazo ejecutado
1 = NO TOUCH
**LABEL** = Label de salto por contacto encontrado

## Casos particulares:

Field = 0 Funcionamiento tipo TCH TMS con cuota palpadura
La instrucción se paran después con desaceleración
haber sentido la pared. La velocidad de tastatura y' aquél
de Feed si Deep <= 0, de otro modo la menor entre Feed y aquél
necesaria a pararse con un espacio igual a Deep.

E.G.

```
                - SFP/0,0,0
                - MOV/Z1,0
                - SFP/2500,0,0
                - DCS/Z1,0.5:0.8,0,500:INP1,1:G1:FRKO
                - RAV/5:L1,Z1
                - DIS/1:POSIZIONE PALPADURA, L1
                - SFP/0,0,0
                - MOV/Z1,0
                - JMP/EXIT
FRKO            -
                - DIS/NO TOUCH
EXIT            - RET
```

Field> 0 Funcionamiento DEEP CONTROL
La posición final por NO TOUCH y' el fin del range
de tastatura. Si la pared no y' encontrada el quadrotto
sólo puede venir mellado. El movimiento de trabajo y'
casi interrumpido en el range de sondeo parecéis.
Valores de Deep <= 0 funciona como un tipo de RETRACT.

E.G.

```
;               DEEP CONTROL
                - SFP/0,0,0
                - MOV/Z1,0
                - SFP/2500,0,0
                - DCS/Z1,0.5:1.2,0.5,0:INP1,1:G1:FRKO
                - RAV/5:L1,Z1
                - DIS/1:POSIZIONE TOCATA, L1
                - SFP/0,0,0
                - MOV/Z1,0
                - JMP/EXIT
FRKO            - JLT/G1,0,TRUC
                - DIS/NO TOUCH
                - JMP/EXIT
TRUC            - DIS/TRUCIOLO
EXIT            - RET
;               RETRACT
                - SFP/0,0,0
                - MOV/Z1,0
                - DCT/Z1,50:-0.2,50,200:G1:FRKO
                - RAV/5:L1,Z1
                - DIS/1:POSIZIONE PALPADURA, L1
                - JMP/EXIT
```

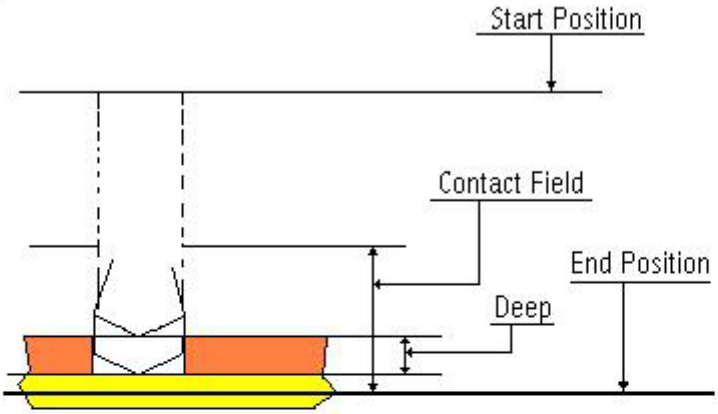
```
FRKO        - JLT/G1,0,TRUC
            - DIS/NO TOUCH
            - JMP/EXIT
TRUC        - DIS/TASTATURA ANTES DEL RANGE
EXIT        - RET
```

Field <0 Funcionamiento SURFACE CONTROL
La posición final por NO TOUCH y' el pinchazo más'
profunda prevista (quadrotto desfondado).
Si la pared no y' encontrada el quadrotto
viene por lo tanto completamente pinchado. El movimiento de trabajo y'
continuo. En este caso el range y' incluido entre Deep+Field
y Deep, con respecto de la posición final.
No y' aprobado Deep <= 0.
En este caso y' previsto un nuevo estado de tastatura en G1
G1 = Global / Local por estado tastatura
-2 = tastatura después del campo de sondeo

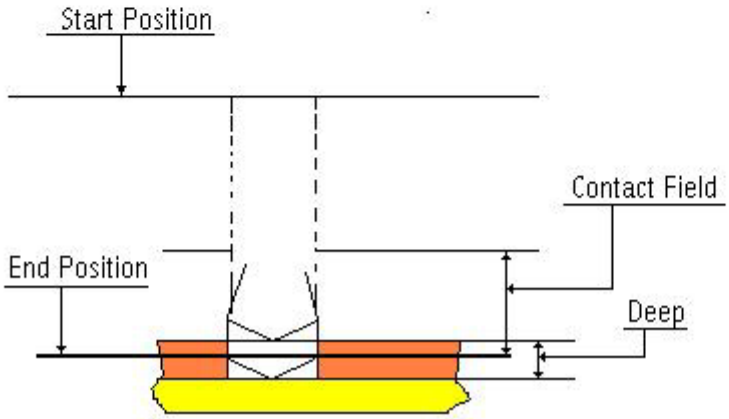
E.G.

```
;           SURFACE CONTROL
            - SFP/0,0,0
            - MOV/Z1,0
            - SFP/2500,0,0
            - DCS/Z1,-2:1.2,-0.5,0:INP1,1:G1:FRKO
            - RAV/5:L1,Z1
            - DIS/1:POSIZIONE TOCATA, L1
            - SFP/0,0,0
            - MOV/Z1,0
            - JMP/EXIT
FRKO        - JEQ/G1,-1,TRUC
            - JEQ/G1,-2,SCHEG
            - DIS/NO TOUCH - PICA RUTA
            - JMP/EXIT
TRUC        - DIS/TRUCIOLO
            - JMP/EXIT
SCHEG       - DIS/PUNTA MELLADO
EXIT        - RET
```

Mode A




Mode B




**Ejemplo:**

```
        - SFP/0,0,0
        - MOV/Z1,0
        - SFP/2500,0,0
        - DCS/Z1,0.5:0.8,0.2,500:Bit,1,G1:FRKO
        - SFP/0,0,0
        - MOV/Z1,0
        - JMP/EXIT
FRKO    - JLT/G1,0,TRUC
        - DIS/NO TOUCH
        - JMP/EXIT
TRUC - DIS/TRUCIOLO
EXIT - RET
```

## 8.31 GRM: GRoup maschine Management

**GRM GRoup maschine Mnagement**

Función: Gestión de los ases y mandriles asociados en grupos maquina

Parámetros: **a:b1,,,,bn**
dónde:

**a** = Tipo de operación (0-7)

0: ; SIRVO OFF
1: ; / / Suspensión grupo con desaceleración inmediata
2: ; / / Ricontinuazione del grupo suspendido
3: ; SIRVO ON
4: ; REPOSICIÓN
5: ; / / SFP de un task chiamante
6: ; / / CFR de un task chiamante
5: ; / / CSP de un task chiamante
10H: ; / / Gestión del mandril como MASCHIATURA
20H: ; / / Gestión del mandril como FRESATURA

**b1,,bn** = Número identificativo Grupo Maquina interesado a la operación

Nota:

En la gestión del mandril como maschiatura, ello es parado o hecho repartir al mismo tiempo al movimiento; excepto con la suspensión del task a fin movimientos ya' agregado, dónde el mandril es parado a movimientos terminados

En la gestión del mandril como fresatura, ello es parado después de que el movimiento y' parado; hecho repartir primera que el movimiento sea reenvidado.

**Ejemplo:**

- - GRM/4:1,2,3,4

Efectúa la operación de REPOSICIÓN ASES y MANDRILES de los GRUPOS MAQUINA 1,2,3 y 4

# CAPÍTULO 9

## 9 Instrucciones por las señales de input y salida

| | |
|---|---|
| 1-WDI (WIN) | Espera que una señal de entrada digital se ofrecida a un determinado estado (Wait Digital Input) |
| .2-WAI | Espera que una señal de entrada analógica se ofrecida a un determinado estado (Wait Analog Input) |
| .3-AIN | Te acampas por las casas uno de los input poniendo el número sobre el G o L, Esperado Input, |
| .4-TDI(TES) | Efectúa una operación de prueba sobre una señal de entrada digital , Prueba Digital Input, |
| .5-TDO | Efectúa una operación de prueba sobre una señal de salida digital , Prueba Digital Salida, |
| .6-IDI | Efectúa una operación de prueba sobre una señal de entrada digital (If Digital Input) |
| .7-IDO | Efectúa una operación de prueba sobre una señal de salida digital (If Digital Salida) |
| .8-TAI | Efectúa una operación de prueba sobre una señal de entrada analógica , Prueba Analog Input, |
| .9-SDO | (SAXO) Secta o resetta de las señales de salida digital, Plató Digital Salida, |
| .10-EDO | Secta o resetta de las señales de salida digital en base a una prueba (Enanced Digital Salida) |
| .11-SAO | (SAC) Escribe el valor de una señal analógica de salida, Plató Analog Salida, |
| .12-GDI | (RBI) Lee el valor de una señal digital de entrada (Get Digital Input) |
| .13-GDO | Lee el valor de una señal digital de salida (Get Digital Salida) |
| .14-GAI(RAI) | Lee el valor de una señal analógica de entrada (Get Analog Input) |
| .15-WBD(BPO) | Escribe un bloque de señales digitales en salida, Write Buffer Digital input, |
| .16-RBD(BPI) | Lee un bloque de señales digitales en entrada, Read Buffer Digital input, |
| .17-CPI | Espera que una de las señales de entrada digital padezca una variación |
| .18-CDI | Al cambio de la señal de entrada digital activa un task o proceso |
| .19-CDO | Al cambio de la señal de entrada digital activa un task o proceso |
| .20-RDI | Efectúa una operación de prueba sobre una señal de entrada digital activando un Task si la prueba es positiva, Run Digital Input, |

.21-RDO Efectúa una operación de prueba sobre una señal de salida digital activando un Task si la prueba es positiva, Run Digital Salida,

## 9.1 WDI: Wait Digital Input

**WDI (Wait Digital Input)**

Función: espera que una señal de entrada digital se ofrecida a un determinado estado.

Parámetros: **a,b:[,c,d]**

dónde:

**a** = número o nombre del input digital

**b** = valor del input digital

**c** = tiempo máximo de espera

**d** = label de salto si el time out es transcurrido

**Ejemplo:**

- WDI/11,1 espera que la señal digital número 11 se convierte en 1
- WDI/11,1:2000,NonArrivaSegnale

**Notas:**

a, LOS parámetros c y d son opcionales

## 9.2 FWI: Fast Wait Digital Input

**FWI (Fast Wait Digital Input)**

Función: espera que un señal de entrada digital si ofrecida a un determinado stato.Il loop de control es efectuada con un frequena mucho más alto del instrucción WDI

Parámetros: **a,b[,c,d]**

dónde:

**a** = número o nombre del input digital

**b** = valor del input digital

**c** = tiempo máximo de espera

**d** = label de salto él el time out es transcurrido

**Ejemplo:**

- WDI/11,1 espera que el señal digital número 11 si convierte en 1

**Notas:**

a, LOS parámetros c y d son opcionales

## 9.3 WAI: Wait Analog Input

**WAI (Wait Analog Input)**

Función: espera que una señal de entrada digital se ofrecida a un determinado estado.

Parámetros: **a:b,c,d[,e,f]**

dónde:

**a** = modalidad de esperado valor
1 = range interior espera que el valor entra en el range
2 = range externo espera que el valor salga del range
**b** = número o nombre del input analógico

**c** = valor del input analógico suela inferior

**d** = valor del input analógico suela superior

**y** = tiempo máximo de espera

**f** = label de salto si el time out es transcurrido

**Ejemplo:**

- WAI/1:Pressure,75,255 espera que la señal analógico Pressure entra en el umbral entre 75 y 255

**Notas:**

a, LOS parámetros y y f son opcionales

## 9.4 AIN: Attesa INput

**AIN (Attesa INput)**

Función: Te acampas por las casas uno de los input poniendo el número sobre el G o L.

Parámetros: **a:b1-bn:c[:d,e]**

dónde:

**a** = 1 o 0 por el valor de los input

**de b1 a bn** = número o nombre de los input digitales

**c** = Global o Local en cuyo es insertado el número de la entrada requerida

**d** = time out

**y** = label de salto si la señal de input digital es igual al valor indicado en **b**

En la Local o Global es cargado 0=se ningunos input consigue el valor requerido en el tiempo deseado, de otro modo inicia de 1 si lo primero de la lista, 2 si el segundo etcétera hasta n

**Ejemplo:**

- AIN/1:inp1,inp2,inp3:g1,1000,labelsalto

## 9.5 TDI: Test Digital Input

**TDI (Test Digital Input)**

Función: efectúa una operación de prueba sobre una señal de entrada digital.

Parámetros: **a,b,c[:d]**

dónde:

**a** = número o nombre del input digital

**b** = valor del input digital

**c** = label de salto si la señal de input digital es diferente al valor indicado en **b**

**d** = Valor en milisegundos en el que la señal tiene que quedar estable (antirimbalzo)

**Ejemplo:**

| | |
|---|---|
| - TDI/11,1, SignalZero | Efectúa una operación de prueba sobre la señal digital número 11 y si 0 efectúa un salto al label SignalZero, de otro modo continúa a la instrucción siguiente |
| - TDI/11,1, SignalZero:1000 | Efectúa una operación de prueba sobre la señal digital número 11 y si 0 efectúa un salto al label SignalZero, de otro modo continúa a la instrucción siguiente si la señal es estable para 1 según, si no estable le salta a SignalZero, |

## 9.6 TDO: Test Digital Output

**TDO (Test Digital Output)**

Función: efectúa una operación de prueba sobre una señal de salida digital.

Parámetros: **a,b,c[:d]**

dónde:

**a** = número o nombre del input digital

**b** = valor de la salida digital

**c** = label de salto si la señal de input digital es diferente al valor indicado en **b**

**d** = Valor en milisegundos en el que la señal tiene que quedar estable (antirimbalzo)

**Ejemplo:**

- TDO/11,1, SignalZero Efectúa una operación de prueba sobre la señal digital número 11 y si 0 efectúa un salto al label SignalZero, de otro modo continúa a la instrucción siguiente

- TDO/11,1, SignalZero:1000 Efectúa una operación de prueba sobre la señal digital número 11 y si 0 efectúa un salto al label SignalZero, de otro modo continúa a la instrucción siguiente si la señal es estable para 1 según, si no estable le salta a SignalZero,

## 9.7 IDI: If Digital Input

**IDI (If Digital Input)**

Función: efectúa una operación de prueba sobre una señal de entrada digital.

Parámetros: **a,b,c[:d]**

dónde:

**a** = número o nombre del input digital

**b** = valor del input digital

**c** = label de salto si la señal de input digital es igual al valor indicado en **b**

**d** = Valor en milisegundos en el que la señal tiene que quedar estable (antirimbalzo)

**Ejemplo:**

- IDI/11,1, SignalOne Efectúa una operación de prueba sobre la señal digital número 11 y si 1 efectúa un salto al label SignalOne

- IDI/11,1, SignalOne:1000 Efectúa una operación de prueba sobre la señal digital número 11 y si 1 efectúa un salto al label SignalOne si queda a 1 por 1 según

## 9.8 IDO: If Digital Output

**IDO (If Digital Output)**

Función: efectúa una operación de prueba sobre una señal de salida digital.

Parámetros: **a,b,c[:d]**

dónde:

**a** = número o nombre del input digital

**b** = valor de la salida digital

**c** = label de salto si la señal de input digital es igual al valor indicado en **b**

**d** = Valor en milisegundos en el que la señal tiene que quedar estable (antirimbalzo)

**Ejemplo:**

- IDO/11,1, SignalOne Efectúa una operación de prueba sobre la señal digital número 11 y si 1 efectúa un salto al label SignalOne
- IDO/11,1, SignalOne:1000 Efectúa una operación de prueba sobre la señal digital número 11 y si 1 efectúa un salto al label SignalOne si queda a 1 por 1 según

## 9.9 TAI: Test Analog Input

**TAI (Test Analog Input)**

Función: efectúa una operación de prueba sobre una señal de entrada analógica.

Parámetros: **a,b,c,d**

dónde:

**a** = número o nombre del input analógico

**b** = valor del input analógico suela inferior

**c** = valor del input analógico suela superior

**d** = label de salto si la señal analógica entra en el range de valores expresado por b y c

**Ejemplo:**

| | |
|---|---|
| - TAI / Pressure,75,255, SignalOnRange | ; Efectúa una operación de prueba sobre la señal analógica Pressure y si entra en el range de 75 a 255 efectúa un salto al label SignalOnRange |

## 9.10 SDO: Set Digital Output

### SDO (Set Digital Output)

Función: efectúa una operación de escritura sobre de las señales de salida digital.

Parámetros: **[n:] a1,b1…an,bn**

dónde:

**[n:]** = caso particular de gestión de las señales ves nota 1

**de a1 a an** = número o nombre de la salida digital

**de b1 a bn** = Valor de la salida digital 1 y 0 es los dos valores permitidos

**Ejemplo:**

- SDO/Elevator,1,Driver,0 — Efectúa una operación de escritura a 1 sobre la señal Elevator y 0 sobre la señal Driver

Nota 1:
Es efectuado la salida consecutiva de bit a partir del parámetro indicado por a1 con los valores de la Local o Global o numero directo indicado en el parámetro b1.
Parecido al WBO es pero por complementario con la instrucción GDO

**Ejemplo:**

- SDO/16:M2.1,L1 — Efectúa una operación de escritura del valor contenido en el local L1 sobre el merker WORD M2

**Ejemplo:**

```
        - SDO/16:M3.1,L1
AS -
        - GDO/16:L1,M3.1
        - LET/L1,L1+1
        - SDO/16:M3.1,L1
        - TMM/1
        - JMP/AS
        - SDO/16:IM1.1,L2
AZ -
        - GDI/16:L2,IM1.1
        - LET/L2,L2+1
        - SDI/16:IM1.1,L2
        - TMM/1
        - JMP/AZ
```

## 9.11 SDI: Set Digital Input

**SDI (Set Digital Input)**

Función: efectúa una operación de escritura sobre de las señales de salida digital.

Parámetros: **[n:] a1,b1…an,bn**

dónde:

**[n:]** = caso particular de gestión de las señales ves nota 1
**de a1 a an** = número o nombre del input digital

**de b1 a bn** = Valor del input digital 1 y 0 es los dos valores permitidos

**Ejemplo:**

- SDI/M1.3,1,M1.4,0 Efectúa una operación de escritura a 1 sobre la señal M1.3 y 0 sobre la señal M1.4

Nota 1:
Es efectuado la salida consecutiva de bit a partir del parámetro indicado por a1 con los valores de la Local o Global o numero directo indicado en el parámetro b1.
Parecido al WBO, actúa pero sobre los input, además es por complementario con la instrucción GDI

**Ejemplo:**

- SDI/16:M1.1,L1 Efectúa una operación de escritura del valor contenido en el local L1 sobre el merker WORD M1

**Ejemplo:**

```
        - SDO/16:M3.1,L1
AS -
        - GDO/16:L1,M3.1
        - LET/L1,L1+1
        - SDO/16:M3.1,L1
        - TMM/1
        - JMP/AS
        - SDO/16:IM1.1,L2
AZ -
        - GDI/16:L2,IM1.1
        - LET/L2,L2+1
        - SDI/16:IM1.1,L2
        - TMM/1
        - JMP/AZ
```

## 9.12 EDO: Enanced Digital Output

**EDO (Enanced Digital Output)**

Función: Secta o resetta de las señales de salida digital en base a una prueba (Enanced Digital Salida).

Parámetros: **a,b:c1,d1…cn,dn[:e]**

dónde:

| | |
|---|---|
| **a** | = variable Local o Global, o bien el nombre o el número de una señal de entrada o bien el nombre de una señal de salida. |
| **b** | = valor de la prueba por que si igual a **a** las salidas **c1 a cn** de los parameri son puestos al valor indicado por **d1 a dn** |
| **de c1 a cn** | = número o nombre de la salida digital |
| **de d1 a dn** | = valor de la salida digital 1 y 0 es los dos valores permitidos |
| **y** | = valor si cero indica en caso no igual de **a** a **b** las señales de salida **c1 a cn** no vendrán settati al modo contrario de aquel indicado por **d1 a dn** , Estándard 1, |

**Ejemplo:**

```
- EDO/input,1:out1,1,out2,1:1

- EDO/ISA8255_OUT9,1:ISA8255_OUT10,1,ISA8255_OUT11,1,ISA8255_OUT12,1,ISA8255_OUT13,1:0
```

## 9.13 SAO: Set Analog Output

**SAO (Set Analog Output)**

Función: efectúa una operación de escritura sobre una señal de salida analógica.

Parámetros: **a1,b1…an,bn**

dónde:

**de a1 a an** = número o nombre de la salida analógica

**de b1 a bn** = Valor de la salida analógica

**Ejemplo:**

| | |
|---|---|
| - SAO/Climp,200,Rout,0 | Efectúa una operación de escritura a 200 sobre la señal Climp y 0 sobre la señal Rout |

## 9.14 GDI: Get Digital Input

### GDI Get Digital Input

Función: ejecutar una operación de lectura de un valor de una señal digital de entrada y lo pone en una variable.

Parámetros: **[n:] a1,b1,a2,b2,... an,bn**

dónde:

**[n:]** = caso particular de gestión de las señales ves nota 1
**de a1 a an =** variable de escribir
**de b1 a bn =** valores dell señal de entrada digital de introducir en las variables

**Ejemplo:**

- GDI/L1,Pulse1,G7,Pulse2 Escribo la variable local 1 con el valor de Pulse1 y la variable global 7 con el valor de Pulse2.

Nota 1:
Es efectuado la lectura consecutiva de bit a partir del parámetro indicado por **b1** y el valor es puesto en el Global o Local indicado en el parámetro **a1.**
Parecido al RBI es pero por complementario con la instrucción SDI

**Ejemplo:**

- GDI/16:L1, M2.1 Efectúa una operación de lectura del merker WORD M2 lo escribe sobre L1

**Ejemplo:**

```
        - SDO/16:M3.1,L1
AS -
        - GDO/16:L1,M3.1
        - LET/L1,L1+1
        - SDO/16:M3.1,L1
        - TMM/1
        - JMP/AS
        - SDO/16:IM1.1,L2
AZ -
        - GDI/16:L2,IM1.1
        - LET/L2,L2+1
        - SDI/16:IM1.1,L2
        - TMM/1
        - JMP/AZ
```

## 9.15 GDO: Get Digital Output

### GDO Get Digital Output

Función: ejecutar una operación de lectura de un valor de una señal digital de salida y lo pone en una variable.

Parámetros: **[n:] a1,b1,a2,b2,... an,bn**

dónde:
**[n:]** = caso particular de gestión de las señales ves nota 1
**de a1 a an =** variable de escribir
**de b1 a bn =** valores dell señal de salida digital de introducir en las variables

**Ejemplo:**

- GDO/L1,Pulse1,G7,Pulse2 Escribo la variable local 1 con el valor de Pulse1 y la variable global 7 con el valor de Pulse2.

Nota 1:

Es efectuado la lectura consecutiva de bit a partir del parámetro indicado por **b1** y el valor es puesto en el Global o Local indicado en el parámetro **a1.**
Parecido al RBI, pero actúa sobre las salidas y además es complementaria con la instrucción SDO

**Ejemplo:**

- GDO/16:L1, M1.1 Efectúa una operación de lectura del merker WORD M1 lo escribe sobre L1

**Ejemplo:**

```
        - SDO/16:M3.1,L1
AS -
        - GDO/16:L1,M3.1
        - LET/L1,L1+1
        - SDO/16:M3.1,L1
        - TMM/1
        - JMP/AS


        - SDO/16:IM1.1,L2
AZ -
        - GDI/16:L2,IM1.1
        - LET/L2,L2+1
        - SDI/16:IM1.1,L2
        - TMM/1
        - JMP/AZ
```

## 9.16 GAI: Get Analog Input

### GAI Get Analog Input

Función: ejecutar una operación de lectura de un valor de una señal analógica de entrada y lo pone en una variable.

Parámetros: **a1,b1,a2,b2,... an,bn**

dónde:

**de a1 a an =** variable de escribir
**de b1 a bn =** valores dell señal de entrada analógica de introducir en las variables

**Ejemplo:**

| | |
|---|---|
| - GAI/L1,Pressure,G7,Temp | Escribo la variable local 1 con el valor de Pressure y la variable global 7 con el valor de Temp. |

## 9.17 WBD: Write Buffer Digital input ( BPO )

**WBD (BPO) Write Buffer Digital output (Bit Parallel Output )**

Función: ejecutar una operación de escritura de un bloque de señales digitales en salida.

Parámetros: **a,b,c**

dónde:

**a** = número de bit sobre que actuar (salida de settare) 1-16

**b** = valor decimal que traducido en máscara binaria corresponde al estado de las salidas de settare
, la modalidad de escritura en esadecimale 0X.. puede ser muy útil,

**c** = n° de la salida de salida

**Ejemplo:**

- WBD/6, 11,8 Actuar sobre las salidas, físicas normales, 8:13, settandoli en ON/OFF

**Notas:**

a, Las salidas tienen que ser progresivos a partir de aquél precisado en el parámetro **c.**

b, Si no existen las salidas, físicos del paquete A., correspondientes a la suma de los parámetros **b+d es** visualizado el mensaje de error.

c, El valor binario de la salida excitada es 1 y diseccitato es 0.

d, En el ejemplo sobre indicado, son excitados las salidas 8,9 y 11 y diseccitati las salidas 10,12,13, porque al valor decimal 11 corresponde la máscara binaria 001011,:

| | | | | | | |
|---|---|---|---|---|---|---|
| salida | 13 | 12 | 11 | 10 | 9 | 8 |
| máscara binaria | 0 | 0 | 1 | 0 | 1 | 1 |
| | 25 | 24 | 23 | 22 | 21 | 20 |
| excitados | | | X | | X | X |

valor decimal 11=1+2+8

e, La máscara binaria de un valor negativo le corresponde al complemento a 2 del valor positivo, que se consigue sumando 1 al complemento de la máscara:

**Ejemplo:**

| | | | | | | |
|---|---|---|---|---|---|---|
| +5 | máscara | 0 | 1 | 0 | 1 | |
| | complemento | | 1 | 0 | 1 | 0 |
| | | + | | | | 1 |
| | | --- | --- | --- | --- | --- |
| disfraza rechazo | | | 1 | 0 | 1 | 1 |

**Ejemplos de valores positivos y negativos con la misma máscara:**

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| obrando su4bit | 0 = -16 | máscara | 0 | 0 | 0 | 0 | | | |
| (val.max.15) | 1 = -15 | | 0 | 0 | 0 | 1 | | | |
| | 2 = -14 | | 0 | 0 | 1 | 0 | | | |
| | . | | | | | | | | |
| | . | | | | | | | | |
| | . | | | | | | | | |
| | 15 = -1 | | 1 | 1 | 1 | 1 | | | |
| operandosu8bit | 0 = -256 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| (val.max.255) | 1 = -255 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| | 2 = -254 | | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| . | | | | | | | | | | |
| . | | | | | | | | | | |
| . | | | | | | | | | | |
| . | | | | | | | | | | |
| | 255 = -1 | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| | | |
|---|---|---|
| obrando sobre 16 bites | 0 = -65536 | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 |
| , val. max. 65535, | 1 = -65535 | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 |
| | 2 = -65534 | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 |
| | . | |
| | . | |
| | . | |
| | . | |
| | 65535 = -1 | 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 |

; L1 = número pasos
; L2 = motor 1-2-3

**Ejemplo:**

```
; Impostación velocidad
        - JEQ/L2,1,mot1
        - JEQ/L2,2,mot2
        - JEQ/L2,3,mot3
        - JEQ/L2,4,mot4
        - JEQ/L2,5,mot5
        - JEQ/L2,6,mot6
        - JEQ/L2,7,mot7
        - JEQ/L2,8,mot8
        - JEQ/L2,9,mot9
        - JEQ/L2,10,mot10
        - JEQ/L2,11,mot11
        - JEQ/L2,12,mot12
        - JMP/errore

mot1 -
        - LET/L17,201,L18,L17+15            ; Motor 1
        - JMP/aa
mot2 -
        - LET/L17,217,L18,L17+15            ; Motor 2
        - JMP/aa
mot3 -
        - LET/L17,233,L18,L17+15            ; Motor 3
        - JMP/aa
mot4 -
        - LET/L17,249,L18,L17+15            ; Motor 4
        - JMP/aa
mot5 -
        - LET/L17,265,L18,L17+15            ; Motor 5
        - JMP/aa
mot6 -
        - LET/L17,281,L18,L17+15            ; Motor 6
        - JMP/aa
mot7 -
        - LET/L17,297,L18,L17+15            ; Motor 7
        - JMP/aa
mot8 -
        - LET/L17,313,L18,L17+15            ; Motor 8
        - JMP/aa
mot9 -
        - LET/L17,329,L18,L17+15            ; Motor 9
        - JMP/aa
mot10 -
        - LET/L17,345,L18,L17+15            ; Motor 10
```

```
        - JMP/aa
mot11 -
        - LET/L17,361,L18,L17+15                ; Motor 11
        - JMP/aa
mot12 -
        - LET/L17,377,L18,L17+15                ; Motor 12
        - JMP/aa



aa -
        - BPO/16,0x0,L17                        ; requerida ejecución mando
        - WDI/L18,0:1000,Errore0
        - BPO/16,0x0701,L17             ; enable motor
        - WDI/L18,1:1000,Errore1
        - BPO/16,0x0,L17                        ; requerida ejecución mando
        - WDI/L18,0:1000,Errore0
        - BPO/16,0x0D00,L17             ; puesta a cero encoder
        - WDI/L18,1:1000,Errore1

        - BPO/16,0x0,L17                        ; requerida ejecución mando
        - WDI/L18,0:10000,Errore0
        - JGE/L1,0,dir_pos
        - BPO/16,0x0901,L17             ; dirección motor negativo
        - JMP/con_dir
dir_pos  -
        - BPO/16,0x0900,L17             ; dirección motor positivo
con_dir  -
        - WDI/L18,1:1000,Errore1

b -
        - BPO/16,0x0,L17                        ; requerida ejecución mando
        - WDI/L18,0:1000,Errore0
        - BPO/16,0x0A00,L17             ; ejecución 1 step
        - WDI/L18,1:1000,Errore1
        - BPI/15,L19,L17                ; ejecución 1 step
        - DIS/L2:Stato, L19,4,-1
        - BPO/16,0x0,L17                        ; requerida ejecución mando
        - WDI/L18,0:1000,Errore0
        - BPO/16,0x0C00,L17             ; mando Leyes Encoder
        - WDI/L18,1:1000,Errore1
        - BPI/14,L20,L17                ; ejecución 1 step
        - LET/L21,L2+5
        - DIS/L21:Encoder, L20,4,-1
        - LET/L19,and(L19,1,
        - JEQ/L19,0,b
/ / / / encontrado micro

/ / invierto dirección
        - BPO/16,0x0,L17                        ; requerida ejecución mando
```

```
        - WDI/L18,0:10000,Errore0
        - JGE/L1,0,dir_posc
        - BPO/16,0x0900,L17             ; dirección motor positivo
        - JMP/con_dirc
dir_posc -
        - BPO/16,0x0901,L17             ; dirección motor negativo
con_dirc -

        - WDI/L18,1:1000,Errore1

/ / reposición marker
        - BPO/16,0x0,L17                        ; requerida ejecución mando
        - WDI/L18,0:1000,Errore0
        - BPO/16,0x0E00,L17             ; reposición marker
        - WDI/L18,1:1000,Errore1

/ / micro a cero
z -
        - BPO/16,0x0,L17                        ; requerida ejecución mando
        - WDI/L18,0:1000,Errore0
        - BPO/16,0x0A00,L17             ; ejecución 1 step
        - WDI/L18,1:1000,Errore1
        - BPI/15,L19,L17                ; ejecución 1 step
        - DIS/L2:Stato, L19,4,-1
        - BPO/16,0x0,L17                        ; requerida ejecución mando
        - WDI/L18,0:1000,Errore0
        - BPO/16,0x0C00,L17             ; mando Leyes Encoder
        - WDI/L18,1:1000,Errore1
        - BPI/14,L20,L17                ; ejecución 1 step
        - LET/L21,L2+5
        - DIS/L21:Encoder, L20,4,-1
        - LET/L19,and(L19,1,
        - JEQ/L19,1,z

/ / búsqueda marker
j -
        - BPO/16,0x0,L17                        ; requerida ejecución mando
        - WDI/L18,0:1000,Errore0
        - BPO/16,0x0A00,L17             ; ejecución 1 step
        - WDI/L18,1:1000,Errore1
        - BPI/15,L19,L17                ;
        - DIS/L2:Stato, L19,4,-1
        - BPO/16,0x0,L17                        ; requerida ejecución mando
        - WDI/L18,0:1000,Errore0
        - BPO/16,0x0C00,L17             ; mando Leyes Encoder
        - WDI/L18,1:1000,Errore1
        - BPI/14,L20,L17                ; ejecución 1 step
        - LET/L21,L2+5
        - DIS/L21:Encoder, L20,4,-1
        - LET/L19,and(L19,0x10,
        - JNE/L19,0x10,j
```

```
/ /

        - BPO/16,0x0,L17                        ; requerida ejecución mando
        - WDI/L18,0:1000,Errore0
        - BPO/16,0x0700,L17             ; saco enable
        - WDI/L18,1:1000,Errore1
        - BPI/15,L19,L17          ;
        - DIS/L2:Stato, L19,4,-1

        - BPO/16,0x0,L17                        ; requerida ejecución mando
        - WDI/L18,0:1000,Errore0
        - BPO/16,0x0C00,L17             ; mando Leyes Encoder
        - WDI/L18,1:1000,Errore1
        - BPI/14,L20,L17                ; ejecución 1 step
        - LET/L21,L2+5
        - DIS/L21:Encoder, L20,4,-1


        - DIS/finito
        - RET


Errore0 -
        - DIS/Errore 0
        - END
Errore1  -
        - DIS/Errore 1
        - END
```

## 9.18 RBD: Read Buffer Digital input (BPI)

**RBD(BPI) Read Buffer Digital input (Bit Parallel Input)**

Función: recibir sobre una variable el estado, ON/OFF, de un buffer de inputs digitales

Parámetros: **a,b,c**

dónde:

**a** = número de input (bit) de que se recibe la situación, de 1 a 16,

**b** = variable en el que es memorizado el valor decimal correspondiente a la máscara binaria función del estado de los input

**c** = n° delI'input de salida

**Ejemplo:**

- RBDI/5,L1,7 memorizar en la variable local 1 el valor correspondiente al estado de los input 7-11, input físicos normales.

**Notas:**

a, Los input interesados tienen que ser progresivos a partir de aquel definido del parámetro **c.**

b, Si los input, físicos del paquete A., correspondientes a la suma de los parámetros **a** y **c no** existen, en ejecución comparece el mensaje de error.

c, Al input excitado corresponde el bit 1; al input diseccitato corresponde el bit 0.

d, En el ejemplo sobre indicado, suponemos que el estado de los input es el partidario:

7,9,10 eccitati8,11

diseccitatiinput

| | 11 | 10 | 9 | 8 | 7eccitato | | |
|---|---|---|---|---|---|---|---|
| | | X | X | | X | | |
| diseccitato | | X | | | X | | |
| | | 24 | 23 | 22 | 21 | 20 | |
| máscara | | 0 | 1 | 1 | 0 | 1 | (1+4+8) = 13 |

El variable L1 asume el valor 13.

## 9.19 CPI: Change Parallel Input

**CPI** **Change Parallel Input**

Función: espera que una de las señales de entrada digital padezca una variación

Parámetros: **a,b,c:[d,e]**

dónde:

| | | |
|---|---|---|
| **a** | = | número de bit sobre que actuar (salida de settare) 1:16 |
| **b** | = | valor decimal que traducido en máscara binaria corresponde al estado de las salidas de settare |
| **c** | = | n° del inpput de salida |
| **y** | = | timeout en milisegundos |
| **f** | = | label de salto |

**Ejemplo:**

```
- CPI/8,L1,Input:1000,labeltimeout
```

## 9.20 CDI: Change Digital Input

**CDI** **Change Digital Input**

Función: al cambio de la señal de entrada digital activa un task o proceso

Parámetros: **a,b,c:d:e:f1.. fn:g**

dónde:

| | | |
|---|---|---|
| **a** | = | variable Local o Global en es registrado el número de la señal de entrada |
| **b** | = | variable Local o Global en es registrado la señal 0 o 1 de la señal de entrada |
| **c** | = | número o nombre señal de entrada |
| **d** | = | nomefile, nomecall del TASK activado cuando la señal conmuta a valor lógico 1 |
| **y** | = | nomefile.pp, nomecall del TASK activado cuando la señal conmuta a valor lógico 0 |
| **f 1.. n** | = | parámetros trasladados al TASK a los su Locales de 1.. n |
| **g** | = | modalidad de control sobre la existencia del TASK:<br>0=viene siempre lanzado creando una nueva instancia<br>1=non es lanzado, pero espera la salida de aquel activo para lanzarlo<br>2=segnala el error |

**Ejemplo:**

```
- CDI/L1,L2,INPUT: nomefile1.pp, nomecall1: nomefile0.pp, nomecall0:I2,56,I1:0
```

## 9.21 CDO: Change Digital Output

**CDO** **Change Digital Output**

Función: al cambio de la señal de salida digital activa un task o proceso

Parámetros: **a,b,c:d:e:f1.. fn:g**

dónde:

| | | |
|---|---|---|
| **a** | = | variable Local o Global en es registrado el número de la señal de salida |
| **b** | = | variable Local o Global en es registrado la señal 0 o 1 de la señal de salida |
| **c** | = | número o nombre señal de salida |
| **d** | = | nomefile, nomecall del TASK activado cuando la señal conmuta a valor lógico 1 |
| **e** | = | nomefile.pp, nomecall del TASK activado cuando la señal conmuta a valor lógico 0 |
| **f 1.. n** | = | parámetros trasladados al TASK a los su Locales de 1.. n |
| **g** | = | modalidad de control sobre la existencia del TASK:<br>0=viene siempre lanzado creando una nueva instancia<br>1=non es lanzado, pero espera la salida de aquel activo para lanzarlo<br>2=segnala el error |

**Ejemplo:**

```
-CDO/L1,L2,OUTPUT: nomefile1.pp, nomecall1: nomefile0.pp, nomecall0: l2,56,l1:1
```

## Ejemplo gestión movimiento manual con I/O

```
                - LET/G1,0,G2,0
pp              - CDO/L1,L2,1:avanti:zero:L1,L2:1
                - CDO/L1,L2,2:indietro:zero:L1,L2:1
                - JMP/pp

PartProgram[indietro]
                - DIS/1:Muovi manual atrás
                - MMA/1:X,-10
                - RET

PartProgram[avanti]
                - DIS/1:Muovi manual adelante
                - MMA/1:X,10
                - RET

PartProgram[zero]
                - DIS/1:Ferma manual
                - MMA/0:X
                - RET



;////////////////////////////////////////////////////////
; NUEVA GESTIÓN ALTO MAQUINA
;////////////////////////////////////////////////////////
;--------------------------------------------------------
;               VARIABLES GLOBALES USADAS
;--------------------------------------------------------
;               G50 = código de error
;               G50-bit1=1              emergencia
;               G50-bit2=1              falta aire general
;               G50-bit3=1              falta aire mandril izquierdo
;               G50-bit4=1              falta aire pinza izquierda
;               G50-bit5=1              falta aire mandril derecho
;               G50-bit6=1              falta aire pinza derecha
;               G50-bit7=1              failure centralita enfriamiento
;--------------------------------------------------------
;               G51 =           estado programa p.p.
;               G51=0           idle
;               G51=1           programa en ejecución
;--------------------------------------------------------
;                       G58 = FLAG DE EMERGENCIA
;               G58 = 1 emergencia en acto
;--------------------------------------------------------
;                       G350-bits
;               0x100 = HOME.PP running, plató reposición de HOME.PP,
;               0x002   falta aire general
```

```
;                 0x004   falta aire mandril izquierdo
;                 0x008   falta aire pinza izquierda
;                 0x010   falta aire mandril derecho
;                 0x020   falta aire pinza derecha
;                 0x040   failure centralita enfriamiento
;-----------------------------------------------------
;                         G500 = CAPACITACIÓN A. DADO POTENCIA
;                 G500 = 0 disabilitazione a dar potencia
;                 G500 = 1 capacitación a dar potencia
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;                    INICIO PROGRAMA
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;
                - JEQ/G500,0,AVANT                              ; disabilitazione a dar potencia
                - SDO/40,1                                      ; capacitación SW al power-on
                - SDO/51,0                                      ; apago lámpara emergencia
AVANT           - RST/1
                - TMM/10                                        ; mando una reposición a la ficha ases
                - LET/G50,0,G350,0                              ; reposiciones variables globales
PRINCIPIO               - LET/G51,1                                     ; señalo programa en ejecución
;               - TMM/10
                - LET/L11,L11+1
                - JGE/L11,0,StartNeg
                - DIS/10:PLC Running.. , L11
                - JLT/L11,0,StartCon
                - LET/L11,10
                - JMP/StartCon
StartNeg                - DIS/10:  , L11
                - JLT/L11,10,StartCon
                - LET/L11,-10
StartCon -
;               - TSK/,gestpa~1.PP,L32:1                        ; llamo gestión panel
;               - TSK/,lampcontr.PP,L32:1                       ; llamo gestión lámparas
;               - JEQ/G500,0,CONT                       ; disabilitazione a dar potencia
                - SDO/40,1                              ; capacitación SW alpower-on
CONT            - CDI/L1,L2,1:RESET1:SEGNAL1::1         ; control estado emergencia
                - CDI/L1,L2,2:RESET2:SEGNAL2::1         ; control aire general
                - CDI/L1,L2,3:RESET3:SEGNAL3::1         ; control aire mandril izquierdo
                - CDI/L1,L2,4:RESET4:SEGNAL4::1         ; control aire mandril derecho
                - CDI/L1,L2,5:RESET5:SEGNAL5::1         ; control aire pinza izquierda
                - CDI/L1,L2,6:RESET6:SEGNAL6::1         ; control aire pinza derecha
                - CDI/L1,L2,7:RESET7:SEGNAL7::1         ; control centralita
                - JMP/INIZIO
;-----------------------------------------------
;            LANZO LOS VARIOS TASK
;-----------------------------------------------
PartProgram[SEGNAL1]
                - DIS/9:0 :\ \ \ \ \ -
                - TDI/24,1,SALT1                        ; control aire general
                - LET/G50,OR(G50,0x1,                   ; señalo estado de emergencia sobre el bit 1
                - LET/G58,1                             ; señalo emergencia en acto
```

```
                - LET/G250,1                    ; tabique flag emergencia
                - LET/G500,1                    ; capacitación a dar potencia * * *
                - SDO/34,0,35,0                 ; apago nacientes
                - TDI/18,0,SALT                 ; control si el gripper izquierdo es sù
                - SDO/21,1                      ; abro el gripper
                - TMM/100                       ; espera 1 sec
                - SDO/22,0                      ; abajo el gripper
                - TMM/1000                      ; espera 1 sec
                - SDO/21,0                      ; cierro el gripper
SALT    - TDI/20,0,SALT1                        ; control si gripper derecho es sù
                - SDO/23,1                      ; abro el gripper
                - TMM/100                       ; espera 1 sec
                - SDO/24,0                      ; abajo el gripper
                - TMM/100                       ; espera 1 sec
                - SDO/23,0                      ; cierro el gripper
SALT1   - DIS/9:0:EmerG
                - END

;-------------------------------------------------------------------------
PartProgram[SEGNAL2]
SEGNAL2         - DIS/9:6 :\ \ \ \ \ -
                - TDI/24,0,FINE2                ; controlo aire general
                - TMM/3000                      ; espera
                - TDI/24,0,SEGNAL2              ; controlo aire general
                - LET/G50,OR(G50,0x2,           ; señalo falta aire general sobre el bit 2
                - LET/G58,1                     ; señalo emergencia en acto
                - SDO/65,0,66,0,67,0,68,0               ; firmes carretillas
                - SDO/34,0,35,0                 ; apago los nacientes RX
                - RST/1                         ; resetto los movimientos en curso
                - SPD/M1,0                      ; firme mandril SX
                - SPD/M2,0                      ; firme mandril DX
                - CAL/ZLUP::1                   ; ZL UP
                - CAL/ZRUP::1                   ; ZR UP
                - SDO/40,0                      ; disabilitazione SW al power-on
                - LET/G50,AND(G50,NOT(0x6D,)
                - LET/G500,0                    ; disabilitazione a dar potencia * * *
                - SDO/51,1,49,0,50,0            ; enciendo lámpara emergencia y apago las otras
FINE2           - DIS/9:6:AriaG
                - END

;-------------------------------------------------------------------------
PartProgram[SEGNAL3]
SEGNAL3         - DIS/9:12 :\ \ \ \ \ -
                - TDI/1,0,FINE3                 ; controlo aire mandril izquierdo
                - TMM/500
                - TDI/1,0,SEGNAL3               ; controlo aire mandril izquierdo
                - LET/G50,OR(G50,0x4,           ; señalo falta aire mandril izquierdo sobre el bit 3
                - LET/G58,1                     ; señalo emergencia en acto
                - SPD/M1,0                      ; firme mandril SX
                - SDO/51,1,49,0,50,0            ; enciendo lámpara emergencia y apago las otras
FINE3           - DIS/9:12:Ar.MS -
```

```
                - END
;-------------------------------------------------------------------------
PartProgram[SEGNAL4]
SEGNAL4         - DIS/9:18 :\ \ \ \ \ -
                - TDI/4,0,FINE4                         ; controlo aire mandril derecho
                - TMM/500
                - TDI/4,0,SEGNAL4                       ; controlo aire mandril derecho
                - LET/G50,OR(G50,0x8,                   ; señalo falta aire mandril derecho sobre el bit 4
                - LET/G58,1                             ; señalo emergencia en acto
                - SDO/51,1,49,0,50,0                    ; enciendo lámpara emergencia y apago las otras
FINE4           - DIS/9:18:Ar.MD -
                - END
;-------------------------------------------------------------------------
PartProgram[SEGNAL5]
SEGNAL5         - DIS/9:24 :\ \ \ \ \ -
                - TDI/2,0,FINE5                 ; controlo aire pinza izquierda
                - TMM/500
                - TDI/2,0,SEGNAL5                       ; controlo aire pinza izquierda
                - LET/G50,OR(G50,0x10,          ; señalo falta aire pinza izquierda sobre el bit 5
                - LET/G58,1                             ; señalo emergencia en acto
                - SPD/M1,0                              ; firme mandril SX
                - SDO/51,1,49,0,50,0                    ; enciendo lámpara emergencia y apago las otras
FINE5           - DIS/9:24:Ar.PS -
                - END
;-------------------------------------------------------------------------
PartProgram[SEGNAL6]
SEGNAL6         - DIS/9:30 :\ \ \ \ \ -
                - TDI/5,0,FINE6                 ; controlo aire pinza derecha
                - TMM/500
                - TDI/5,0,SEGNAL6                       ; controlo aire pinza derecha
                - LET/G50,OR(G50,0x20,          ; señalo falta aire pinza derecha sobre el bit 6
                - LET/G58,1                             ; señalo emergencia en acto
                - SDO/51,1,49,0,50,0                    ; enciendo lámpara emergencia y apago las otras
FINE6           - DIS/9:30:Ar.PD -
                - END
;-------------------------------------------------------------------------
PartProgram[SEGNAL7]
SEGNAL7         - DIS/9:36 :\ \ \ \ \ -
                - TDI/39,0,FINE7        ; controlo input centralita
                - TMM/300
                - TDI/39,0,SEGNAL7              ; controlo input centralita
                - LET/G50,OR(G50,0x40,          ; señalo anomalía centralita enfriamiento sobre el bit 7
                - LET/G58,1                     ; señalo emergencia en acto
                - SPD/M1,0                      ; firme mandril SX
                - SDO/51,1,49,0,50,0            ; enciendo lámpara emergencia y apago las otras
FINE7           - DIS/9:36:Centr -
                - END

;-------------------------------------------------
;               TASK DE REPOSICIÓN
;-------------------------------------------------
```

```
; Reposición señal falta potencia
PartProgram[RESET1]
                - LET/G50,AND(G50,NOT(0x1,)     ; resetto bit 1
                - JEQ/G500,0,AVANT1             ; disabilitazione a dar potencia
                - SDO/40,1                      ; capacitación SW alpower-on
                - LET/G58,0                     ; resetto flag emergencia
                - CAL/,SpinPreset.PP,I32        ; llamo preset mandriles
AVANT1 - DIS/9:0:------
                - END

;--------------------------------------------------
; Reposición señal falta aire general
PartProgram[RESET2]
                - LET/G50,AND(G50,NOT(0x2,)             ; resetto bit 2
                - LET/G350,AND(G350,NOT(0x2,)
                - DIS/9:6:------
                - END
;--------------------------------------------------
; Reposición señal falta aire mandril izquierdo
PartProgram[RESET3]
                - LET/G50,AND(G50,NOT(0x4,)             ; resetto bit 3
                - LET/G350,AND(G350,NOT(0x4,)
                - DIS/9:12:------
                - END
;--------------------------------------------------
; Reposición señal falta aire mandril derecho
PartProgram[RESET4]
                - LET/G50,AND(G50,NOT(0x8,)             ; resetto bit 4
                - LET/G350,AND(G350,NOT(0x8,)
                - DIS/9:18:------
                - END
;--------------------------------------------------
; Reposición señal falta aire pinza izquierda
PartProgram[RESET5]
                - LET/G50,AND(G50,NOT(0x10,)            ; resetto bit 5
                - LET/G350,AND(G350,NOT(0x10,)
                - DIS/9:24:------
                - END
;--------------------------------------------------
; Reposición señal falta aire pinza derecha
PartProgram[RESET6]
                - LET/G50,AND(G50,NOT(0x20,)            ; resetto bit 6
                - LET/G350,AND(G350,NOT(0x20,)
                - DIS/9:30:-----
                - END
--------------------------------------------------
; Reposición señal anomalía centralita enfriamiento
PartProgram[RESET7]
                - LET/G50,AND(G50,NOT(0x40,)            ; resetto bit 7
                - LET/G350,AND(G350,NOT(0x40,)
                - DIS/9:36:------
```

```
                  - END

;*************************************************
;*************************************************
;                         SUBROUTINE
;*************************************************
; PartProgram[EMERGENZA]
;                 Emergencia ases
;                 - SDO/40,0                          ; disabilitazione SW al power-on
;                 - RET
;--------------------------------------------------
PartProgram[ZLUP]
;                 Eje ZL alto
;                 L1 = Valor ZL alto
                  - RAV/0:L2,SYSTEM
                  - LET/L2,AND(L2,2,
                  - JEQ/L2,0,VIA
RIP               - RAV/0:L2,ZL
                  - LET/L3,AND(L2,0x4,
                  - JEQ/L3,0,VIA
                  - LET/L2,AND(L2,0x80000,
                  - JNE/L2,0,RIP
                  - MOV/ZL,L1
CALLE             - RET
;---------------------------------------------------
PartProgram[ZRUP]
;                 Eje ZR alto
;                 L1 = Valor ZR alto
                  - RAV/0:L2,SYSTEM
                  - LET/L2,AND(L2,2,
                  - JEQ/L2,0,VIA
RIP               - RAV/0:L2,ZR
                  - LET/L3,AND(L2,0x4,
                  - JEQ/L3,0,VIA
                  - LET/L2,AND(L2,0x80000,
                  - JNE/L2,0,RIP
                  - MOV/ZR,L1
CALLE             - RET
;----------------------------------------------------
;                 - EMC/11:Servo OFF
;                 - PWO/10:Power ON
;                 - CYC/10:Cycle start
```

## 9.22 RDI: Run Digital Input

**RDI (RunDigital Input)**

Función: efectúa una operación de prueba sobre una señal de entrada digital y si igual activa un task o proceso.

Parámetros: **a,b,c,d:e1.. n:f**

dónde:

**a** = número o nombre del input digital
**b** = valor del input digital
**c** = nomefile, nomecall del TASK activado cuando la señal conmuta a valor lógico 1
**d** = nomefile.pp, nomecall del TASK activado cuando la señal conmuta a valor lógico 0
**y 1.. n** = parámetros trasladados al TASK a los su Locales de 1.. n
**f** = modalidad de control sobre la existencia del TASK:
0=viene siempre lanzado creando una nueva instancia
1=non es lanzado, pero espera la salida de aquel activo para lanzarlo
2=segnala el error

**Ejemplo:**

- RDI/11,1, SignalOne Efectúa una operación de prueba sobre la señal digital de entrada número 11 y si 1 activa el task al proceso SignalOne

..

PartProgram[SignalOne]
..

## 9.23 RDO: Run Digital Output

**RDO (Run Digital Output)**

Función: efectúa una operación de prueba sobre una señal de salida digital.

Parámetros: **a,b,c,d:e1.. n:f**

dónde:

**a** = número o nombre del input digital
**b** = valor de la salida digital
**c** = nomefile, nomecall del TASK activado cuando la señal conmuta a valor lógico 1
**d** = nomefile.pp, nomecall del TASK activado cuando la señal conmuta a valor lógico 0
**y 1.. n** = parámetros trasladados al TASK a los su Locales de 1.. n
**f** = modalidad de control sobre la existencia del TASK:
0=viene siempre lanzado creando una nueva instancia
1=non es lanzado, pero espera la salida de aquel activo para lanzarlo
2=segnala el error

**Ejemplo:**

```
- RDO/11,1, SignalOne   Efectúa una operación de prueba sobre la señal digital de salida número 11 y si 1
                        activa el task al proceso SignalOne
..


PartProgram[SignalOne]
..
```

# CAPÍTULO 10

## 10 Instrucciones de sincronización

| | |
|---|---|
| .1-EVS | Secta de los acontecimientos de sincronización (EVent Plató) |
| .2-EVC | Resetta de los acontecimientos de sincronización (EVent Clear) |
| .3-EVW | Espera algunos acontecimientos de sincronización (EVent Wait) |
| .4-EVG | Lee el estado de los acontecimientos (EVent Get) |

## 10.1 EVS: EVent Set

**EVS** **EVent Set**

Función: de calle libre al semáforo, es considerado ocurrido un acontecimiento,

Parámetros: **a1.. an**

dónde:

**de a1 a an =** = número del semáforo interesado, 1 ÷ 64, (65÷ 128,

### Ejemplo:

- EVS/1,3 Al semáforo 1 y 3.

**Notas:**

a, La instrucción **EVS** presiente en un part program, relativo a un task en elaboración, obra paralelamente a la instrucción **EVW** presiente sobre otro part program, relativo a otro task en elaboración.
Prácticamente la instrucción **EVS**, que es ejecutada siempre enseguida, de calle libre al semáforo mientras la instrucción **EVW** sólo es ejecutado si el semáforo está libre; en este último caso neutraliza el efecto de la instrucción **EVS** ribloccando el semáforo.

b, Los acontecimientos son 128 de cuyo los primeros 64 nacen settati 0 es decir borrado, mientras de 65-128 nacen es decir a 1 settati

### 10.2 EVC: EVent Clear

**EVC** **EVent Clear**

Función: del bloque al semáforo, es considerado ocurrido un acontecimiento,

Parámetros: **a1.. an**

dónde:

**de a1 a an =** = número del semáforo interesado, 1 ÷ 64, (65÷ 128,

## Ejemplo:

- EVC/1,3 Al semáforo 1 y 3.

**Notas:**

a, La instrucción **EVC** presiente en un part program, relativo a un task en elaboración, obra paralelamente a la instrucción **EVW** presiente sobre otro part program, relativo a otro task en elaboración.
Prácticamente la instrucción **EVC**, que es ejecutada siempre enseguida, de un alto al semáforo mientras la instrucción **EVW** sólo es ejecutado si el semáforo está libre; en este último caso neutraliza el efecto de la instrucción **EVC** ribloccando el semáforo.

### 10.3 EVW: EVent Wait

**EVW** **EVent Wait**

Función: ejecutar una prueba sobre los semáforos y continuar o menos a segunda que éste estén libres o atascados

Parámetros: **a1, an:[,b,c]**

dónde:

**de a1 a an** = número del semáforo interesado, 1 ÷ 64, nace atascados, 65÷ 128)nascono libre

**b** = tiempo máximo de espera en milisegundos

**c** = label de salto si el time out es transcurrido

## Ejemplo:

- EVW/1,5,7 ll programa sólo pasa a la instrucción siguiente si los semáforos 1,5,7 es todo libres.

- EVW/5:2000, NonArrivaEvento ll programa sólo pasa a la instrucción siguiente si el semáforo 5 es verde, libre, si dopo 2 sec este no ocurre es ejecutado un salto a lla label "NonArrivaEvento"

**Notas:**

a, La instrucción **EVW** obra paralelamente a la instrucción **EVS,** ves esta última instrucción.

b, Cada vez que cada es ejecutado una instrucción **EVW**, si al menos uno de los semáforos interesados es no libero, el programa se para sobre esta instrucción hasta a que todos los semáforos no se hacen libres; a tal punto el programa continúa con la instrucción siguiente y es borrada la condición de semáforo libre definida por la instrucción **EVS.**

c, Cuando un task está en espera sobre de una instrucción **EVW**, este task no viene schedulato más hasta a que en interrupt no interviene la condición de semáforo libre.
El empleo de las instrucciones **EVS** y **EVW** es por lo tanto muy conveniente cada cada vez que una instrucción sobre de un task tiene que ser ejecutada en función de una condición que se averigua sobre de otro task.
Es obvio que si un task está en espera sobre de una instrucción **EVW** y no interviene una instrucción **EVS** a desbloquearlo, queda atascado éste al infinito,a menos que el time out si declarara.

## 10.4 EVG: EVent Get

**EVG** **EVent Get**

Función: Ligeras el estado de los acontecimientos

Parámetros: **a1,b1… an,bn**

dónde:

**de a1 a an** = variables, solista locales o globales, en los que se quiere memorizar el estado del acontecimiento o semáforo
**de b1 a bn** = número del acontecimiento (semáforo) interesado, 1 ÷ 64, (65÷ 128,

### Ejemplo:

- EVG/L1,1,L2,5,L3,7 Le carga en el Local L1 el estado, 1o 0, del acontecimiento 1, cargo en el Local L2 el estado, 1o 0, del acontecimiento 5, cargo en el Local L3 el estado, 1o 0, del acontecimiento 7

**Notas:**

La instrucción **EVG no** interfiere sobre el estado de un acontecimiento, por lo tanto es que utilizar exclusivamente como lectura, no **utilices como función de sincronización**

# CAPÍTULO 11

## 11 Instrucciones de servicio

| | |
|---|---|
| .1-FOC(AZZ, | Borra el contenido un expediente o la crea si no existe, Expediente Open Creadas, |
| .2-FWR | (SCR) Escribe un récord sobre expediente, Expediente WRite, |
| .3-FRD | (LEG) Lee un récord de expediente, Expediente ReaD, |
| .4-TIM | Temporizador en segundo (Time) |
| .5-TMM | Temporizador en milésimo de según (Time Millisecond) |
| .6-SWA | Inicializa un reloj (Start WAtch) |
| .7-RWA | Lee un reloj (Read WAtch) |
| .8-HWA | Para un reloj (Halt WAtch) |
| .9-CWA | Continua un reloj, Continuos WAtch, |
| .10-KYB | Espera una operación de teclado, KeYBoard, |
| .11-DRT | Visualiza continuamente los valores de ases, globales, señales |
| .12-DIS | Visualiza una raya de mensaje (DISplay) |
| .13-HLD | Manda el sistema en el estado de Cycle Alto (HoLD) |
| .14-PWO | Manda el sistema en el estado de Power ON (PoWerOn) |
| .15-EMC | Reposición del sistema (Reposición) |
| .16-LCK | Lock el task y eventualmente señala con un SEC |
| .17-ULK | Concede todos los task en estado de LOCK |
| .18-RST | Reposición del sistema (Reposición) |
| .19-SDW | Shut down del sistema (ShutDoWn) |
| .20-SOR | Orden de una secuencia de valores (SORt) |
| .21-GTK | Nota las informaciones inherentes a un TASK o proceso, Get TasK information, |
| .22-MDI | Ejecuta un programa ISO - GCODE (ISO) |
| .23-OTC | Ejecuta un programa ISO - GCODE (ISO) |
| .24-ISO | Ejecuta un programa ISO - GCODE (ISO) |
| .25-WND | Espera una señal de estado de error de los recursos ases o mandril (Wait Notify Detected) |
| .26-WKY | Espera la comprimida de una tecla, Wait KeYboard, |
| .27-NHL | No Hold |
| .28-YHL | Yes Hold |
| .29-GDT | Get Fechas and Time |
| .30-GLN | Get Número locales |
| .31-GMI | Get Motion Information |
| .32-RTC | Read Temporizador o Counter |
| .33-SGL | Save Globales |

| | |
|---|---|
| .34-SHL | Shell de aplicados os o procedimientos |
| .35-G80 | Fino ciclo fijo (G80) |
| .36-G81 ..89 | G81-G89 Activo el ciclo fijo precisado |
| .37-ESE | Ejecuta secuencias externas (Exec Sequence) **sistema ETEL** |
| .38-ERR | Lees registros externos (External Read Registry) **sistema ETEL** |
| .39-EWR | Escribes registros externos (External Write Registry) **sistema ETEL** |
| .40-ECM | Ejecuta un mando externo (External CoMmand) **sistema ETEL** |
| .41-EWS | Espera una señal (External Wait Signal) **sistema ETEL** |
| .42-CLM | Mando de Lógica de Máquina **sistema ETEL** |

## 11.1 FOC: File Open Create(AZZ)

**FOC(AZZ) File Open Create**

Función: borra el contenido un expediente o lo crea si no existe

Parámetros: **[t:] a**

dónde:

**t** = Tipo si uno (1) el expediente viene si existe borrado y no más creado vacío , Estándard 0,
**a** = Nombre del expediente

Atención si en el nombre del expediente hay el carácter '#con número d una variable LOCAL o GLOBAL, viene comosto con el número entero indicado.
**Ejemplo:**

- GDT/L1,L3
- LET/L11,L1-2000,L11,L11+L2*100,L11,L11+L3*10000
- FOC/c:\pippo#L11.txt

- GDT/L1,L3
en L1 = 2003
  L2 = 12
  L3 = 20

- LET/L11,L1-2000,L11,L11+L2*100,L11,L11+L3*10000
en L11=201203

Por cuyo
c :\ pippo#L11.txt el nuevo tamaño: c :\ pippo201203.txt

**Ejemplo:**

- FOC/c:\check\date.dat el expediente si existe es borrado y ceato vacío

**Ejemplo:**

- FOC/1:c:\check\date.dat el expediente si existe es borrado

## 11.2 FWR: File Write(SCR)

**FWR(SCR) File WRite**

Función: escribe un récord sobre

Parámetros: **a,[b],[c],[d1,dn] [, e][,f,g,h]**

dónde:

**a** = Nombre del expediente
Atención si en el nombre del expediente hay el carácter '#con número d una variable LOCAL o GLOBAL, viene comosto con el número entero indicado.
**Ejemplo:**

- GDT/L1,L3
- LET/L11,L1-2000,L11,L11+L2*100,L11,L11+L3*10000
- FOC/c:\pippo#L11.txt

- GDT/L1,L3
en L1 = 2003
  L2 = 12
  L3 = 20

- LET/L11,L1-2000,L11,L11+L2*100,L11,L11+L3*10000
en L11=201203

Por cuyo
c :\ pippo#L11.txt el nuevo tamaño: c :\ pippo201203.txt

**b** = TIME escribe la hora y el día
**c** = cordón de escribir
**d1,dn** = Primera Variable de escribir, Última variable de Escribir
**y** = Número progresivo en el récord
**f** = variable cuyo contenido, valor numérico, se quiere añadir después del cordón **c**
**g** = número de cifras enteras de visualizar (0:10)
**h** = número de cifras decimales de visualizar (0:5)

**Notas:**

a,

- Si es puesto valor de –1 en el parámetro **y es** visualizado el valor **c** en esadecimale

**Ejemplos:**

- FWR/c:\pippo,TIME, cosa, G1, G12, L1,G5,5,-1
- FWR/c:\pippo,TIME, cosa, G1, G12, L1
- FWR/c:\pippo, cosa, G1, G12
- FWR/c:\pippo,TIME, G1, G12
- FWR/c:\pippo, G1, G12
- FWR/c:\pippo,cosa
- FWR/c:\pippo,TIME,cosa

**Ejemplo:**

```
                - DIS/1:Inizio
                - DIS/2:>
                - AZZ/\DINX1.TXT
                - SCR/\DINX1.TXT,N Pos_X1          Voltio
                - LET/L10,0
                - CAP/27:X1,0.1
                - CAP/9:X1,-300
INIZ            - LET/L1,0
                - CAP/51:X1,0                      ; buffer off
                - CAP/50:X1,1,X1,17,X1,14          ; buffer on
POSIZ           - MOV/X1,L1
                - TMM/2
                - LET/L1,L1-2.54*2
                - JGE/L1,-200,POSIZ
                - CAP/51:X1,0                      ; buffer off
                - MOV/X1,0
                - CAP/9:X1,0
LOOPS           - GAV/L11,L12,L1,L4
                - JLE/L11,0,FINE
                - JEQ/L12,0,DOPOS
                - SCR/\DINX1.TXT,Sbordamento,L12,L12
                - DIS/2:Overflow,L12
                - JMP/FINE
DOPOS           - ADD/L10,L11
                - ADD/L9,1
                - SCR/\DINX1.TXT,L1,L3,L9
                - JMP/LOOPS
FIN             - DIS/1:Finito,L10
```

**Ejemplo:**

```
                - SFP/10000
                - DIS/1:Inizio
                - DIS/2:>
                - AZZ/\DINX1.TXT
                - SCR/\DINX1.TXT,N Pos_YEncoder Pos_YScala
                - RAV/1:L1,Y,L2,YS
                - MOV/Y,-100
                - TMM/500
                - RAV/1:L1,Y,L2,YS
                - CAP/9:Y,L1,YS,L2          ; Puesta a cero de los 2 ases
                - CAP/51:Y,0; buffer off
                - CAP/50:Y,1,YS,1; buffer on
                - STC/0
                - MOV/Y,-400
                - TMM/2000
                - CAP/51:Y,0; buffer off
                - MOV/Y,0
                - CAP/9:Y,0,YS,0 ; Anulación compensación de los 2 ases
                - SET/L9,0,L10,0
LOOPS           - GAV/L11,L12,L1,L2
                - JLE/L11,0,FINE
                - JEQ/L12,0,DOPOS
                - SCR/\DINX1.TXT,Sbordamento,L12,L12
                - DIS/2:Overflow,L12
                - JMP/FINE
DOPOS           - ADD/L10,L11
                - ADD/L9,1
                - SCR/\DINX1.TXT,L1,L2,L9
                - JMP/LOOPS
FIN             - DIS/1:Finito,L10
                -END
```

## 11.3 FRD: File ReaD(LEG)

**FRD(LEG) File ReaD**

Función: lee un récord de expediente

/ / - FRD/c:\pippo, Posición De Leer,
/ /, Última Posición Leída, -1 finos expedientes,
/ /, Número Globales Hojeadas
/ /, G1, G12

Parámetros: **a,b,c,d,e1,en**

dónde:

**a** = Nombre del expediente
Atención si en el nombre del expediente hay el carácter '#con número d una variable LOCAL o GLOBAL, viene comosto con el número entero indicado.
**Ejemplo:**

- GDT/L1,L3
- LET/L11,L1-2000,L11,L11+L2*100,L11,L11+L3*10000
- FOC/c:\pippo#L11.txt

- GDT/L1,L3
en L1 = 2003
L2 = 12
L3 = 20

- LET/L11,L1-2000,L11,L11+L2*100,L11,L11+L3*10000
en L11=201203

Por cuyo
c :\ pippo#L11.txt el nuevo tamaño: c :\ pippo201203.txt

**b** = Posición de leer
**c** = variable sobre cuyo es cargado la última posición leída, si –1 finos expedientes,
**d** = variable sobre cuyo es cargado el número variables leídas
**e1,en** = Primera variable de cargar.. Última variable de cargar

### Ejemplo:

- FRD/c:\check\date.dat,12,L1,L2,G100,G2001

### Ejemplo:

```
        - DIS/Inizio

        - LET/L3,1; posición de leer
aa      - FRD/c:\dante.txt,L3,L1,L2,L11,L12

        - DIS/1:Valore pos. de leer =, L3
        - DIS/2:Valore pos. leída =, L1
        - DIS/3:Valore variables hojeadas =, L2
        - DIS/4:Valore 1 =, L11
        - DIS/5:Valore 2 =, L12
        - DIS/6:Valore 3 =, L13

        - JEQ/L1,-1,Finito
        - LET/L3,L1+1
        - TMM/100
        - JMP/aa
Terminado -
        - DIS/Finito
```

**El expediente dante.txt contiene los siguientes datos:**

```
 1 X 0 Y 0
 2 X -170 Y -260
 3 X -110 Y -190
 4 X -50 Y -60
 5 X 50 Y 60
 6 X 110 Y 190
 7 X 170 Y 260
 8 X -170 Y 260
 9 X -110 Y 190
10 X -50 Y 60
11 X 50 Y -60
12 X 110 Y -190
13 X 170 Y -260
14 X -170 Y -190
15 X -110 Y -120
16 X -50 Y 0
17 X 0 Y 60
18 X 50 Y 190
19 X 110 Y 260
20 X 170 Y 190
21 X 110 Y 120
22 X 50 Y 0
23 X 0 Y -60
24 X -50 Y -190
25 X -110 Y -260
26 X 110 Y -260
27 X 50 Y -190
28 X -110 Y 120
29 X -170 Y 190
30 X -110 Y 260
31 X -50 Y 190
32 X 110 Y -120
33 X 170 Y -190
34 X 50 Y -260
35 X -170 Y 120
36 X -50 Y 260
37 X 170 Y -120
38 X -50 Y -260
39 X -170 Y -120
40 X 50 Y 260
41 X 170 Y 120
42 X -140 Y -225
43 X -80 Y -155
44 X -25 Y -30
45 X 25 Y 30
46 X 80 Y 155
47 X 140 Y 225
48 X -140 Y 225
49 X -80 Y 155
50 X -25 Y 30
51 X 25 Y -30
52 X 80 Y -155
53 X 140 Y -225
54 X -140 Y -155
55 X 140 Y 155
56 X 80 Y 225
57 X -80 Y 225
58 X -140 Y 155
59 X 140 Y –155
```

## LEG Lees Expediente

Función: lee un récord de expediente

/ / - LEG/c:\pippo, Posición De Leer,
/ /, Última Posición Leída, -1 finos expedientes,
/ /, Número Globales Hojeadas
/ /, G1, G12
/ / - LEG/1:c:\pippo, Posición De Leer
/ /, Última Posición Leída, -1 finos expedientes,
/ /, Número Global CARGADO
/ /
/ /, G1 Tipo 1 = número
/ / 2 = carta
/ / G2 Valor del número o de la carta
/ /...
/ / Gn-1 Tipo 1 = número
/ / 2 = carta
/ / Gn Valor del número o de la carta

Parámetros: [**t:] a,b,c,d,e1,en**

dónde:

**t** = Tipo si uno, 1, es cargado de modo diferente los parámetros el,en
**a** = Nombre del expediente
Atención si en el nombre del expediente hay el carácter '#con número d una variable LOCAL o GLOBAL, viene comosto con el número entero indicado.
**Ejemplo:**

```
- GDT/L1,L3
- LET/L11,L1-2000,L11,L11+L2*100,L11,L11+L3*10000
- FOC/c:\pippo#L11.txt

- GDT/L1,L3
en L1 = 2003
  L2 = 12
  L3 = 20

- LET/L11,L1-2000,L11,L11+L2*100,L11,L11+L3*10000
en L11=201203
```

Por cuyo
c :\ pippo#L11.txt el nuevo tamaño: c :\ pippo201203.txt
**b** = Posición de leer
**c** = variable sobre cuyo es cargado la última posición leída, si –1 finos expedientes,

**d** = variable sobre cuyo es cargado el número variables leídas
**e1,en** = Primera variable de cargar.. Última variable de cargar

**si**
**t** = Tipo si uno (1) entonces
el = 1 número
2 carta
el+1 = Valor del número o de la carta
……..
en = 1 número
2 carta
en+1 = Valor del número o de la carta

**Ejemplo:**

```
        - DIS/Inizio

        - LET/L31,1
        - SWA/1


        - LET/L3,1; posición de leer

aa      - LBF/L11,L22,0
        - LEG/1:c:\temp\Prova.txt,L3,L1,L2,L11,L22
        - JEQ/L1,-1,Finito

        - DIS/1:Valore pos. de leer =, L3
        - DIS/2:Valore pos. leída =, L1
        - DIS/3:Valore variables hojeadas =, L2
        - DIS/4,0:Valore 1 =, L11
        - DIS/4,25:Valore 2 =, L12
        - DIS/5,0:Valore 3 =, L13
        - DIS/5,25:Valore 4 =, L14
        - DIS/6,0:Valore 5 =, L15
        - DIS/6,25:Valore 6 =, L16
        - DIS/7,0:Valore 7 =, L17
        - DIS/7,25:Valore 8 =, L18
        - DIS/8,0:Valore 9 =, L19
        - DIS/8,25:Valore10 =, L20
        - DIS/9,0:Valore11 =, L21
        - DIS/9,25:Valore12 =, L22


        - DIS/10:Record letti,L31
        - TMM/2000
        - LET/L31,L31+1
        - RWA/L32,1
```

- DIS/11:Tempo,L32
- JLE/L32,0.1,li
- DIS/12:Quanti sec,L31/L32

ellos -

- JEQ/L1,-1,Finito
- LET/L3,L1+1
- JMP/aa

Terminado -

- HWA/1
- RWA/L32,1
- DIS/11:Quanti sec,L31/L32
- DIS/10:Tempo,L32
- DIS/9:Record letti,L31
- DIS/Finito

**El expediente c:\temp\Prova.txt contiene los siguientes datos:**

```
N01 O1
N02 T1 M8
N03 M3 S1000
N04 G4 1000
N05 G00 Z17.5
N06 G01 Z15 F30
N07 G01 Z7 F75
N08 M3 S500
N09 G01 Z6.5 F50
N10 G00 Z93.4
;N07 G80
N11 M28
N12 M5
```

## 11.4 TIM: TIMe

**TIM TIMe**

Función: efectúa un retraso definido en segundo

Parámetros: **a**

dónde:

**a** = retardo en céntesimos de según

**Ejemplo:**

```
- TIM/100                ; Retraso de 1 según
```

## 11.5 TMM: TiMe Millisecond

**TMM TiMe Millisecond**

Función: efectúa un retraso definido en milisegundos

Parámetros: **a**

dónde:

**a** = retardo en milisegundos

**Ejemplo:**

- TMM/10 Retraso de 10 milisegundos

## 11.6 SWA: Start WAtch

**SWA** **Start WAtch**

Función: Inicializa un reloj

Parámetros: **a**

dónde:

**a** = reloj 1-16

**Ejemplo:**

- SWA/1

## 11.7 RWA: Read WAtch

**RWA** **Read WAtch**

Función: lee la cuenta de un reloj en una variable

Parámetros: **a,b**

dónde:

**a** = variable sobre cuyo es leído el valor del reloj
**b** = reloj 1-16

**Ejemplo:**

```
- RWA/L1,1
```

## 11.8 HWA: Halt WAtch

**HWA** **Halt WAtch**

Función: para la cuenta de un reloj

Parámetros: **a**

dónde:

**a** = reloj 1-16

**Ejemplo:**

- HWA/1

## 11.9 CWA: Continue WAtch

**CWA** **Continue WAtch**

Función: continúa la cuenta de un reloj anteriormente interrumpido

Parámetros: **a**

dónde:

**a** = reloj 1-16

**Ejemplo:**

- CWA/1

## 11.10 KYB: KeYBoard

**KYB KeYBoard**

Función: Espera un input de teclado

Parámetros: **a,b[:c,d,e]**

dónde:
**a** = Cordón de visualizar
**b** = Global o Local de adquirir
**c** = variable cuyo contenido, valor numérico, se quiere visualizar después del cordón **a**
**d** = número de cifras enteras de visualizar (0:10)
**y** = número de cifras decimales de visualizar (0:5)

**Notas:**

a,
f) Si es puesto valor de –1 en el parámetro **y es** visualizado el valor **c** en esadecimale

**Ejemplo:**

- KYB/Input Key,G1

**Ejemplo:**

- LET/L1,123
- KYB/Input Key,G1:L1,7,0 Visualizado: Input Key 123

**Notas:**

- Es mandado un acontecimiento a los Client EVENTO_ATTESA_DA_TASTIERA

## 11.11 DRT: DispIay Real Time

**DRT** **Display Real Time**

Función: visualizar un mensaje con o sin el contenido de una variable

Parámetros: **a,b,c,[a2,b2,c2.. an,bn,cn]**

dónde:

**a** = posición 1-6
**b** = Tipo elemento
**RELACIONES PÚBLICAS** posición real eje
**PT** posición teórico eje
**ER** error eje
**G** Global
**c** = Identificativo elemento

**Ejemplo:**

- DRT/1,PR,A1,2,G,100

## 11.12 DIS: DlSpIay

**DIS DlSplay**

Función: visualizar un mensaje con o sin el contenido de una variable

Parámetros: **a,[b]:c,[d,e,f]**

dónde:

| | | |
|---|---|---|
| **a** | = | raya de visualización 0 -17 |
| **b** | = | columna de principio visualización 0-127 |
| **c** | = | mensaje de visualizar |
| **d** | = | variable cuyo contenido, valor numérico, se quiere visualizar |
| **y** | = | número de cifras enteras de visualizar (0:10) |
| **f** | = | número de cifras decimales de visualizar (0:5) |

**Ejemplo:**

- DIS/2,34;PEZZI ESEGUITI,G2 visualizar sobre la raya 2, partiendo de la columna 34, el mensaje TROZOS EJECUTADOS = seguido por el contenido del variable G2.

**Notas:**

Si es indicada la raya -1nel parámetro **a**, es mandado un acontecimiento a los Client EVENTO_DISPLAY

Si es indicada la raya –2 en el parámetro **a**, es mandado un acontecimiento a los Client EVENTO_TRACE

Si es puesto valor de –1 en el parámetro **f es** visualizado el valor **d** en esadecimale

**Ejemplo:**

```
- DIS/commento=0x,L1,4,-1
                es visualizado en la raya 0:
                commento=0x0fc1|->4 cifras esadecimali valor en L1, local 1,
```

**Ejemplo:**

```
a       - LET/L11,L11-1
        - JLE/L11,0,StartNeg
        - DIS/0,0:PlcRunning...
        - JMP/StartCon
StartNeg        -
        - DIS/0,0:
        - JGT/L11,-10,StartCon
        - LET/L11,10
StartCon        -

        - JMP/a
```

## 11.13 HLD: HoLD

**HLD HoLD**

Función: manda el sistema en el estado de Cycle Alto

Parámetros: **[a],[b]:[c],[d,e,f]**

dónde:

| | | |
|---|---|---|
| **a** | = | raya de visualización 0 -17 |
| **b** | = | columna de principio visualización 0-127 |
| **c** | = | mensaje de visualizar |
| **d** | = | variable cuyo contenido, valor numérico, se quiere visualizar |
| **e** | = | número de cifras enteras de visualizar (0:10) |
| **f** | = | número de cifras decimales de visualizar (0:5) |

**Ejemplo:**

```
- HLD                    manda el subsistema en estado de Cycle Alto
```

## 11.14 PWO: PoWer On

**PWO** **PoWer ON**

Función: Manda el sistema en el estado de Power ON

Parámetros: **[a],[b]:[c],[d,e,f]**

dónde:

| | | |
|---|---|---|
| **a** | = | raya de visualización 0 -17 |
| **b** | = | columna de principio visualización 0-127 |
| **c** | = | mensaje de visualizar |
| **d** | = | variable cuyo contenido, valor numérico, se quiere visualizar |
| **y** | = | número de cifras enteras de visualizar (0:10) |
| **f** | = | número de cifras decimales de visualizar (0:5) |

**Ejemplo:**

```
- PWO
```

## 11.15 EMC: EMergenCy

**EMC** **Emergency**

Función: Manda el sistema en el estado de Emergencia (EMergenCy)

Parámetros: **[a],[b]:[c],[d,e,f]**

dónde:

| | | |
|---|---|---|
| **a** | = | raya de visualización 0 -17 |
| **b** | = | columna de principio visualización 0-127 |
| **c** | = | mensaje de visualizar |
| **d** | = | variable cuyo contenido, valor numérico, se quiere visualizar |
| **y** | = | número de cifras enteras de visualizar (0:10) |
| **f** | = | número de cifras decimales de visualizar (0:5) |

**Ejemplo:**

```
- EMC
```

## 11.16 LCK: LOCK and set event client

### LCK Lock task

Función: Lock el task and plató event client

Parámetros: **[[a],b,c]**

dónde:

**a** = iCookie, estándard –1,
**b** = Cordón de comunicación
**c** = valor de comunicación

Ejemplo:

```
/ / - LCK/iCookie:MACHINE_LOGIC,1; Info
/ / - LCK/iCookie:MACHINE_LOGIC,2; Warning
/ / - LCK/iCookie:MACHINE_LOGIC,3; Allarm
/ / - LCK/iCookie:MACHINE_LOGIC,4; Dinamic Área
/ / - LCK

        - LCK/MACHINE_LOGIC,3
        - RET

        - WDI/ingresso,1,10000,errore
        ..
        ..
error   - LCK
```

## 11.17 ULK: UnLOCK

### ULK Lock task

Función: Unlock libera los task en LCK

Parámetros:

Ejemplo:

- ULK

## 11.18 RST: ReSeT

**RST** **Reset**

Función: Reposición del sistema (Reposición)

Parámetros: **a**

dónde:
**a** = Modalidad de reposición
0 reposiciones ases y mandriles, reposición PartProgram, reposición me Nivela, reposición Socket, reposiciones Seriales
1 reposiciones ases y mandriles
2 reposiciones ases y mandriles, reposición me Nivela, reposición Socket, reposiciones Seriales

**Ejemplo:**

```
- RST/1          ; reposición ases y mandriles
```

## 11.19 SWN: Shut down

**SWN** **Shut down**

Función: Shut down del sistema (ShutDoWn)

Parámetros:

dónde:

**Ejemplo:**

```
- SDW
```

## 11.20 SOR: SOrt

**SOR SORt**

Función: Orden de una secuencia de valores (SORt)

Parámetros: **a:b,c**

dónde:
**a** = Modalidad de orden
1 ordena del más grande
–1 ordena del más pequeño
**b** = Dirección Principio variable
**c** = Dirección Fina variable

**Ejemplo:**

```
- LET/L1,93,L2,48,L3,4,L4,7,L5,8
- LET/L6,12,L7,9,L8,76,L9,200,L10,45,L11,95,L12,58
- SOR/1:L1,L12                                    ; ordena del más grande
- DIS/Valore L1, L1
- DIS/1:Valore L2, L2
- DIS/2:Valore L3, L3
- DIS/3:Valore L4, L4
- DIS/4:Valore L5, L5
- DIS/5:Valore L6, L6
- DIS/6:Valore L7, L7
- DIS/7:Valore L8, L8
- DIS/8:Valore L9, L9
- DIS/9:Valore L10,L10
- DIS/10:Valore L11,L11
- DIS/11:Valore L12,L12
```

## 11.21 GTK: Get TasK information

**GTK** **Get TasK information**

Función: Nota las informaciones inherentes a un TASK o proceso, Get TasK information,

Parámetros: **a:b,[c,d,e]**

dónde:
**a** = nomefile.pp, nomecall:
**b** = Variable dónde es cargado: 1 tsk activos, 0 tsk no activo
**c** = Variable dónde es cargado: el número locales
**d** = Primera variable dónde es cargado el valor del primer local
**y** = Última variable dónde es cargado el valor de la enésima local

**Ejemplo:**

- GTK/TaskName:L1,L2,L10,L32

L1 Variable dónde es cargado: 1 tsk activos, 0 tsk no activo
L2 Variable dónde es cargado: el número locales
L10 Primera variable dónde es cargado el valor del estreno local L1 del part program TaskName
L32 Última variable dónde es cargado el valor del local L22 del part program TaskName

## 11.22 MDI: Ejecuta una instrucción ISO

**MDI** **Ejecuta una instrucción ISO**

Función: Ejecuta una instrucción ISO - GCODE (ISO)

Parámetros: **a:btesto itruzione iso**

**a** = ID (Identificativo numérico) del proceso ISO
o bien nombre del proceso ISO
**b** = texto de la instrucción

**Ejemplo:**

- MDI / Scara:X123Y45F1000

## 11.23 OTC: Cargo va tirando tablero ISO

### OTC Cargo va tirando tablero Origen - Tool-Corrector - Paramétrico ISO

Función: Carga los campos del tablero Origen-Tool - Corrector-Paramétrico ISO - GCODE (ISO)

Parámetros: **a:b:c:d1,e1……dn, en**

**a** = ID (Identificativo numérico) del proceso ISO
o bien nombre del proceso ISO
**b** = U o bien T o bien C o bien P
**c** = Número del tablero

**d1.. n** = Identificativo voy tirando del tablero o Valor del campo del Tablero P
**el.. n** = Valor del campo del Tablero O, T,C

Le vos identificados de los campos de los tableros soy:

Tablero Origina:
XES compensaciones eje X
Y compensación eje Y
Z compensación eje Z
A compensación eje A.
B compensación eje B
C compensación eje C
U compensación eje U
V compensación eje V
W compensación eje W

Tablero Tool, útil,:
LC Largo Tool
RT Rayo Tool
Free1 Campo 1
Free2 Campo 2
Free3 Campo 3
Free4 Campo 4
Size
Spec
NM

PM
PD
CP
SC
Diámetro
Paso
Inch

Tablero Corretori:
LC Largo Tool
RT Rayo Tool
Free1 Campo 1
Free2 Campo 2
Free3 Campo 3
Free4 Campo 4

**Ejemplo:**

```
- OTC / Scara:O ,1:X,234
- OTC / Scara:T ,2:LC,11
- OTC / Scara:C ,9:LC,23.8
- OTC / Scara:P ,10:23.8
```

## 11.24 ROP: Lee campo del tablero ISO

**ROP** **Lee campo del tablero Origen - Tool - Corrector - Paramétrico ISO**

Función: Lee un campo del tablero Origen-Tool-Corrector - Paramétrico ISO - GCODE (ISO)

Parámetros: **a:b:c:d1,e1……dn, en**

**a** = ID (Identificativo numérico) del proceso ISO
o bien nombre del proceso ISO
**b** = U o bien T o bien C o bien P
**c** = Número del tablero

**d1.. n** = Local o Global sobre cuyo cargar el valor del campo
**el.. n** = Identificativo voy tirando del tablero o nada por del Tablero P

Le vos identificados de los campos de los tableros soy:

Tablero Origina:
XES compensaciones eje X
Y compensación eje Y
Z compensación eje Z
A compensación eje A.
B compensación eje B
C compensación eje C
U compensación eje U
V compensación eje V
W compensación eje W

Tablero Tool, útil,:
LC Largo Tool
RT Rayo Tool
Free1 Campo 1
Free2 Campo 2
Free3 Campo 3
Free4 Campo 4
Size
Spec
NM
PM

PD
CP
SC
Diámetro
Paso
Inch

Tablero Corretori:
LC Largo Tool
RT Rayo Tool
Free1 Campo 1
Free2 Campo 2
Free3 Campo 3
Free4 Campo 4

**Ejemplo:**

- ROP / Scara:O,1:L1,X
- ROP / Scara:T,2:L1,LC
- ROP / Scara:C ,9:L1,LC
- ROP / Scara:P ,10:L1

## 11.25 ISO: Ejecuta un part program ISO

**ISO** **Ejecuta un part program ISO**

Función: Ejecuta un programa ISO - GCODE (ISO)

Parámetros: **a,b:[c:d,e,f:g,h:i,l:m,n]**

dónde:
**a** = nomefile.prg o bien palabra clave **ISOCNC,** Ve Nota 1,
**b** = ID (Identificativo numérico) del proceso ISO que ejecutará el programa que él tiene que encontrar nell directorio de sistema CNC;
o bien nombre del proceso ISO en este caso el directorio del expediente. prg es el directorio del proceso ISO le definido en el "sistema.txt"
**c** = Time de esperado cumplimiento elaboración: 0 no espera Estándard 0
-1 infinito
> 0 tiempo de espera
**d,e,f** = compensación x,y,z
**g,h** = parámetros de ir a casa de P1-Pn
**i,l** = parámetros vuelvo de P1-Pn
**m** = nomefile, nomecall del TASK activado cuando el proceso ISO es interrumpido de una anomalía ases
**n** = modalidad de control sobre la existencia del TASK:
0=viene siempre lanzado creando una nueva instancia
1=non es lanzado
2=segnala el error

Nota 1: si al sitio del nombre de un programa ISO, es indicado la palabra clave **ISOCNC** el sistema ejecutará el programa selecto en el módulo CNC del entorno CNC, correspettivo al proceso ISO identificado por el ID.

**Ejemplo:**

```
modo 1  , explícito,                  - ISO/A.PRG,Scara:-1:0,0,0:L141,L145:L146,L149:,SistemaInErrore

modo 2, directorio CNC,               - ISO/CAVALLO.PRG,13:-1:0,0,0:L141,L145:L146,L149:,SistemaInErrore

modo 3, último selezionanato, - ISO/ISOCNC,3:-1:0,0,0:L141,L145:L146,L149:,SistemaInErrore

        en caso de anomalía sobre los ases es ejecutado el Task "SistemaInErrore"

 los parámetros de entrada salidos pueden ser omitidos       - ISO/FresaB.PRG,2:-1:0,0,0::,0_ErroreISO
```

## Ejemplo llamado de un part program ISO con orientación de Visión

```
                - DIM/150
PRINCIPIO       -
                - SAX/101,0
                - MOV/X,3,Y,0
                - LET/L1,0              / / Left
                - LET/L2,0
                - LET/L3,1              / / Número integraciones
                - LET/L4,10             / / Size blob
                - LET/L5,0              / / Shape


                - DIS/1:  Inicio Measure
                - DIS/2:
                - DIS/3:
                - DIS/4:
                - DIS/5:
                - DIS/6:
;
                - EVW/71
                - SWA/1

                - JEQ/L1,0,Cam2
/ /
/ / Número telecámaras
/ /
Cam1            - LET/L11,1       / / iNumberCameraActive
                - JMP/ViaPr

Cam2            - LET/L11,2       / / iNumberCameraActive
ViaPr           - ARI/1,12,VisAlgo,SetActiveCamera,L11,L11,L31,L31


/ /
/ / Leyes imagen
/ /
                - LET/L11,0             / / iTipoPalette,
                - LET/L12,L3            / / Número integraciones
;               - ARI/1,12,VisionBase,ReadMultiImages,L11,L12,L31,L31
                - ARI/1,12,VisionBase,ReadImage,L11,L12,L31,L31
/ / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / /
/ /       Tomas 1 blob              / /
/ / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / /
                - LET/L11,-215          / / iLeft,
                - LET/L12,200           / / iTop,
                - LET/L13,55            / / iWidth,
                - LET/L14,55            / / iHeight,
                - LET/L15,77            / / 77 GetTreshold
```

```
                - JEQ/L2,0,DopoNegativo1
                - JEQ/L2,1,DopoPositivo1
                - JMP/fine
DopoNegativo1   - LET/L16,1             / / iModeLowParam,
                - JMP/Continua1
DopoPositivo1   - LET/L16,0             / / iModeLowParam,
                - JMP/Continua1
Continua1       - LET/L17,2             / / iModeHighParam Tipo de Sogliatore Funny
                - LET/L18,0             / / iTypeBlob 0 vacío 1 lleno
                - LET/L19,200           / / iLowArea
                - LET/L20,0             / / iTypeFilter
                - LET/L21,0             / / iFilterParam
                - LET/L22,1             / / Distancia mínima del centro
                - LET/L23,30
                - ARI/1,12,VisAlgo,GetAllBlobs,L11,L23,L31,L31
                - LET/L48,L31
                - LET/L49,1
LoopB1          - JGT/L49,L48,NOBLOB
/ /
/ / Valor blob
/ /
                - LET/L11,L49    / / iNumberBlobBySize
                - ARI/1,12,VisAlgo,GetValueBlob,L11,L11,L31,L51
                - LET/L50,L31 / /
                - LET/L51,L32 / / XES Baricentro geométricos
                - LET/L52,L33 / / Y Baricentro geométrico
                - LET/L53,L34 / / Área
                - LET/L54,L35 / / Perímetro
                - JMP/contin1
                - JLT/L34,20000,contin
                - LET/L49,L49+1
                - JMP/LoopB1



/ /
/ / Círculo blob
/ /
contin1  - LET/L11,L49          / / iNumberBlobBySize,
                - LET/L12,L141 / / dRaggioNoto,
                - ARI/1,12,VisAlgo,GetCircleBlob,L11,L12,L30,L38
                - LET/L55,L31 / / Rayo
                - LET/L56,L32 / / X Centro
                - LET/L57,L33 / / Y Centro
                - LET/L58,L34    / / Redondez

/ /
/ / Dibuja blob
/ /
                - LET/L11,L49           / / iNumberBlobBySize,
```

```
                - LET/L12,0             / / iFlagBlobPicture,
                - LET/L13,80            / / uRGBPictureColor,
                - LET/L14,1             / / iFlagBlobEdge,
                - LET/L15,008           / / uRGBEdgeColor,
                - LET/L16,0             / / iFlagValue,
                - LET/L17,0             / / iFlagDimension,
                - LET/L18,17            / / iFlagCircle,
                - LET/L19,0             / / iFlagXAxesRect,
                - LET/L20,0             / / iFlagYAxesRect,
                - LET/L21,1             / / iFlagBordoROI,
                - LET/L22,0             / / iFlagBordoInternoROI,
                - LET/L23,0             / / iFlagCoordinateROI,
                - LET/L24,0             / / iFlagEllisse,
                - ARI/1,12,VisAlgo,DrawBlob,L11,L24,L30,L30
/ /
/ / Texto de las cuotas sobre vídeo
/ /
                - LET/L11,L51-70/ / IX,
                - LET/L12,L52           / / iY,
                - LET/L13,0             / / iFlagStringa
                - LET/L14,0             / / pStringa,
                - LET/L15,0             / / iFlagAng,
                - LET/L16,0             / / iFlagAngMilliToSessa,
                - LET/L17,L55           / / Rayo,
                - LET/L18,1             / / iFlagX,
                - LET/L19,L56           / / dX,
                - LET/L20,1             / / iFlagY,
                - LET/L21,L57           / / dY,
                - LET/L22,2             / / Número font,
                - LET/L23,888           / / color,
                - ARI/1,12,VisAlgo,DrawTextOnVideo,L11,L23,L30,L30
/ / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / /
/ /       Tomas 2 blob              / /
/ / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / /
                - LET/L11,145           / / iLeft,
                - LET/L12,200           / / iTop,
                - LET/L13,55            / / iWidth,
                - LET/L14,55            / / iHeight,
                - LET/L15,77            / / 77 GetTreshold
                - JEQ/L2,0,DopoNegativo2
                - JEQ/L2,1,DopoPositivo2
                - JMP/fine
DopoNegativo2   - LET/L16,1             / / iModeLowParam,
                - JMP/Continua2
DopoPositivo2   - LET/L16,0             / / iModeLowParam,
Continua2       - LET/L17,2             / / iModeHighParam Tipo de Sogliatore Funny
                - LET/L18,0             / / iTypeBlob 0 vacío 1 lleno
                - LET/L19,200           / / iLowArea
                - LET/L20,0             / / iTypeFilter
                - LET/L21,0             / / iFilterParam
                - LET/L22,1             / / Distancia mínima del centro
```

```
                - LET/L23,30
                - ARI/1,12,VisAlgo,GetAllBlobs,L11,L23,L31,L31

                - LET/L48,L31
                - LET/L49,1
LoopB2          - JGT/L49,L48,NOBLOB
/ /
/ / Valor blob
/ /
                - LET/L11,L49     / / iNumberBlobBySize
                - ARI/1,12,VisAlgo,GetValueBlob,L11,L11,L31,L51
                - LET/L60,L31 / /
                - LET/L61,L32 / / XES Baricentro geométricos
                - LET/L62,L33 / / Y Baricentro geométrico
                - LET/L63,L34 / / Área
                - LET/L64,L35 / / Perímetro
                - JMP/contin2
                - JLT/L34,20000,contin
                - LET/L49,L49+1
                - JMP/LoopB2
/ /
/ / Círculo blob
/ /
contin2         - LET/L11,L49     / / iNumberBlobBySize,
                - LET/L12,L141 / / dRaggioNoto,
                - ARI/1,12,VisAlgo,GetCircleBlob,L11,L12,L30,L38
                - LET/L65,L31 / / Rayo
                - LET/L66,L32 / / X Centro
                - LET/L67,L33 / / Y Centro
                - LET/L68,L34     / / Redondez
/ /
/ / Dibuja blob
/ /
                - LET/L11,L49             / / iNumberBlobBySize,
                - LET/L12,0               / / iFlagBlobPicture,
                - LET/L13,80              / / uRGBPictureColor,
                - LET/L14,1               / / iFlagBlobEdge,
                - LET/L15,008     / / uRGBEdgeColor,
                - LET/L16,0               / / iFlagValue,
                - LET/L17,0               / / iFlagDimension,
                - LET/L18,17              / / iFlagCircle,
                - LET/L19,0               / / iFlagXAxesRect,
                - LET/L20,0               / / iFlagYAxesRect,
                - LET/L21,1               / / iFlagBordoROI,
                - LET/L22,0               / / iFlagBordoInternoROI,
                - LET/L23,0               / / iFlagCoordinateROI,
                - LET/L24,0               / / iFlagEllisse,
                - ARI/1,12,VisAlgo,DrawBlob,L11,L24,L30,L30
/ /
/ / Texto de las cuotas sobre vídeo
/ /
```

```
                - LET/L11,L61-70/ / IX,
                - LET/L12,L62              / / iY,
                - LET/L13,0                / / iFlagStringa
                - LET/L14,0                / / pStringa,
                - LET/L15,0                / / iFlagAng,
                - LET/L16,0                / / iFlagAngMilliToSessa,
                - LET/L17,L65              / / Rayo,
                - LET/L18,1                / / iFlagX,
                - LET/L19,L66              / / dX,
                - LET/L20,1                / / iFlagY,
                - LET/L21,L67              / / dY,
                - LET/L22,2                / / Número font,
                - LET/L23,888              / / color,
                - ARI/1,12,VisAlgo,DrawTextOnVideo,L11,L23,L30,L30


/ / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / /
/          Tomas 3 blob                / /
/ / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / / /
                - LET/L11,-215             / / iLeft,
                - LET/L12,-145             / / iTop,
                - LET/L13,55               / / iWidth,
                - LET/L14,55               / / iHeight,
                - LET/L15,77               / / 77 GetTreshold
                - JEQ/L2,0,DopoNegativo3
                - JEQ/L2,1,DopoPositivo3
                - JMP/fine
DopoNegativo3   - LET/L16,1                / / iModeLowParam,
                - JMP/Continua3
DopoPositivo3   - LET/L16,0                / / iModeLowParam,
                - JMP/Continua3
Continua3 -
                - LET/L17,2                / / iModeHighParam Tipo de Sogliatore Funny
                - LET/L18,0                / / iTypeBlob 0 vacío 1 lleno
                - LET/L19,200              / / iLowArea
                - LET/L20,0                / / iTypeFilter
                - LET/L21,0                / / iFilterParam
                - LET/L22,1                / / Distancia mínima del centro
                - LET/L23,30
                - ARI/1,12,VisAlgo,GetAllBlobs,L11,L23,L31,L31
                - LET/L48,L31
                - LET/L49,1
LoopB3          - JGT/L49,L48,NOBLOB
/ /
/ / Valor blob
/ /
                - LET/L11,L49              / / iNumberBlobBySize
                - ARI/1,12,VisAlgo,GetValueBlob,L11,L11,L31,L51
                - LET/L70,L31              / /
                - LET/L71,L32      / / XES Baricentro geométricos
                - LET/L72,L33              / / Y Baricentro geométrico
```

```
                - LET/L73,L34     / / Área
                - LET/L74,L35     / / Perímetro
                - JMP/contin3
                - JLT/L34,20000,contin
                - LET/L49,L49+1
                - JMP/LoopB3
/ /
/ / Círculo blob
/ /
contin3         - LET/L11,L49                   / / iNumberBlobBySize,
                - LET/L12,L141    / / dRaggioNoto,
                - ARI/1,12,VisAlgo,GetCircleBlob,L11,L12,L30,L38
                - LET/L75,L31               / / Rayo
                - LET/L76,L32     / / X Centro
                - LET/L77,L33               / / Y Centro
                - LET/L78,L34               / / Redondez
/ /
/ / Dibuja blob
/ /
                - LET/L11,L49               / / iNumberBlobBySize,
                - LET/L12,0                 / / iFlagBlobPicture,
                - LET/L13,80                / / uRGBPictureColor,
                - LET/L14,1                 / / iFlagBlobEdge,
                - LET/L15,008               / / uRGBEdgeColor,
                - LET/L16,0                 / / iFlagValue,
                - LET/L17,0                 / / iFlagDimension,
                - LET/L18,17                / / iFlagCircle,
                - LET/L19,0                 / / iFlagXAxesRect,
                - LET/L20,0                 / / iFlagYAxesRect,
                - LET/L21,1                 / / iFlagBordoROI,
                - LET/L22,0                 / / iFlagBordoInternoROI,
                - LET/L23,0                 / / iFlagCoordinateROI,
                - LET/L24,0                 / / iFlagEllisse,
                - ARI/1,12,VisAlgo,DrawBlob,L11,L24,L30,L30

/ /
/ / Texto de las cuotas sobre vídeo
/ /
                - LET/L11,L71-70/ / IX,
                - LET/L12,L72               / / iY,
                - LET/L13,0                 / / iFlagStringa
                - LET/L14,0                 / / pStringa,
                - LET/L15,0                 / / iFlagAng,
                - LET/L16,0                 / / iFlagAngMilliToSessa,
                - LET/L17,L75               / / Rayo,
                - LET/L18,1                 / / iFlagX,
                - LET/L19,L76               / / dX,
                - LET/L20,1                 / / iFlagY,
                - LET/L21,L77               / / dY,
                - LET/L22,2                 / / Número font,
                - LET/L23,888               / / color,
```

```
                  - ARI/1,12,VisAlgo,DrawTextOnVideo,L11,L23,L30,L30
//////////////////////////////////////////
//      Tomas 4 blob              //
//////////////////////////////////////////
                  - LET/L11,145           // iLeft,
                  - LET/L12,-145          // iTop,
                  - LET/L13,55            // iWidth,
                  - LET/L14,55            // iHeight,
                  - LET/L15,77            // 77 GetTreshold
                  - JEQ/L2,0,DopoNegativo4
                  - JEQ/L2,1,DopoPositivo4
                  - JMP/fine
DopoNegativo4     - LET/L16,1             // iModeLowParam,
                  - JMP/Continua4
DopoPositivo4     - LET/L16,0             // iModeLowParam,
                  - JMP/Continua4
Continua4 -
                  - LET/L17,2              // iModeHighParam Tipo de Sogliatore Funny
                  - LET/L18,0             // iTypeBlob 0 vacío 1 lleno
                  - LET/L19,200           // iLowArea
                  - LET/L20,0             // iTypeFilter
                  - LET/L21,0             // iFilterParam
                  - LET/L22,1             // Distancia mínima del centro
                  - LET/L23,30
                  - ARI/1,12,VisAlgo,GetAllBlobs,L11,L23,L31,L31
                  - LET/L48,L31
                  - LET/L49,1
LoopB4  - JGT/L49,L48,NOBLOB
//
// Valor blob
//
                  - LET/L11,L49            // iNumberBlobBySize
                  - ARI/1,12,VisAlgo,GetValueBlob,L11,L11,L31,L51
                  - LET/L80,L31            //
                  - LET/L81,L32     // XES Baricentro geométricos
                  - LET/L82,L33     // Y Baricentro geométrico
                  - LET/L83,L34     // Área
                  - LET/L84,L35     // Perímetro
                  - JMP/contin4
                  - JLT/L34,20000,contin
                  - LET/L49,L49+1
                  - JMP/LoopB4
//
// Círculo blob
//
contin4  - LET/L11,L49     // iNumberBlobBySize,
                  - LET/L12,L141    // dRaggioNoto,
                  - ARI/1,12,VisAlgo,GetCircleBlob,L11,L12,L30,L38
                  - LET/L85,L31            // Rayo
                  - LET/L86,L32     // X Centro
                  - LET/L87,L33     // Y Centro
```

```
                - LET/L88,L34           / / Redondez
/ /
/ / Dibuja blob
/ /
                - LET/L11,L49           / / iNumberBlobBySize,
                - LET/L12,0             / / iFlagBlobPicture,
                - LET/L13,80            / / uRGBPictureColor,
                - LET/L14,1             / / iFlagBlobEdge,
                - LET/L15,008           / / uRGBEdgeColor,
                - LET/L16,0             / / iFlagValue,
                - LET/L17,0             / / iFlagDimension,
                - LET/L18,17            / / iFlagCircle,
                - LET/L19,0             / / iFlagXAxesRect,
                - LET/L20,0             / / iFlagYAxesRect,
                - LET/L21,1             / / iFlagBordoROI,
                - LET/L22,0             / / iFlagBordoInternoROI,
                - LET/L23,0             / / iFlagCoordinateROI,
                - LET/L24,0             / / iFlagEllisse,
                - ARI/1,12,VisAlgo,DrawBlob,L11,L24,L30,L30
/ /
/ / Texto de las cuotas sobre vídeo
/ /
                - LET/L11,L81-70/ / IX,
                - LET/L12,L82           / / iY,
                - LET/L13,0             / / iFlagStringa
                - LET/L14,0             / / pStringa,
                - LET/L15,0             / / iFlagAng,
                - LET/L16,0             / / iFlagAngMilliToSessa,
                - LET/L17,L85           / / Rayo,
                - LET/L18,1             / / iFlagX,
                - LET/L19,L86           / / dX,
                - LET/L20,1             / / iFlagY,
                - LET/L21,L87           / / dY,
                - LET/L22,2             / / Número font,
                - LET/L23,888           / / color,
                - ARI/1,12,VisAlgo,DrawTextOnVideo,L11,L23,L30,L30
/ /
/ / línea recta entre dos centros
/ /
                - LET/L11,(L86-L56)*(L86-L56,
                - LET/L12,(L87-L57)*(L87-L57,
                - LET/L13, sqr(L11+L12, / / Rayo
                - JLE/L13,0.001,noele1
                - LET/L22,(L87-L57)/L13
                - LET/L23,(L86-L56)/L13
                - LET/L24,-1
                - LET/L21,(L22*L24*L56)+(L23*L57,
;               - ARI/1,12,VisAlgo,DrawElementLine,L21,L23,L20,L20
noele1  -
                - LET/L11,(L76-L66)*(L76-L66,
                - LET/L12,(L77-L67)*(L77-L67,
```

```
                - LET/L13, sqr(L11+L12, / / Rayo
                - JLE/L13,0.001,noele2
                - LET/L22,(L77-L67)/L13
                - LET/L23,(L76-L66)/L13
                - LET/L24,-1
                - LET/L21,(L22*L24*L66)+(L23*L67,
;               - ARI/1,12,VisAlgo,DrawElementLine,L21,L23,L20,L20
noele2  -
                - LET/L31,200 / / MM X
                - LET/L32,200 / / MM Y
                - LET/L41,L66-L56 / / DELTA Xsup
                - LET/L42,L86-L76 / / DELTA Xsup
                - LET/L43,abs(L41+L42)/2
                - LET/G501,1
                - JLE/L43,0.001,nomisx
                - LET/G501,L31/L43
nomisx  -
                - LET/L41,L77-L57 / / DELTA Ysin
                - LET/L42,L87-L67 / / DELTA Ydes
                - LET/L43,abs(L41+L42)/2
                - LET/G502,1
                - JLE/L43,0.001,nomisy
                - LET/G502,L32/L43
nomisy  -
                - DIS/10:Valore pixel X,G501
                - DIS/11:Valore pixel Y,G502

/ /
/ / Tomas blob
/ /
                - LET/L11,-215          / / iLeft,
                - LET/L12,200           / / iTop,
                - LET/L13,415           / / iWidth,
                - LET/L14,400           / / iHeight,
                - LET/L15,77            / / 77 GetTreshold
                - JEQ/L2,0,DopoNegativo
                - JEQ/L2,1,DopoPositivo
                - JMP/fine
DopoNegativo    - LET/L16,1             / / iModeLowParam,
                - JMP/Continua
DopoPositivo    - LET/L16,0             / / iModeLowParam,
                - JMP/Continua
Continua        - LET/L17,2             / / iModeHighParam Tipo de Sogliatore Funny
                - LET/L18,0             / / iTypeBlob 0 vacío 1 lleno
                - LET/L19,200           / / iLowArea
                - LET/L20,0             / / iTypeFilter
                - LET/L21,0             / / iFilterParam
                - LET/L22,1             / / Distancia mínima del centro
                - LET/L23,30
                - ARI/1,12,VisAlgo,GetAllBlobs,L11,L23,L31,L31
                - LET/L48,L31
```

```
                - LET/L49,1
LoopB           - JGT/L49,L48,NOBLOB



/ /
/ / Valor blob
/ /
                - LET/L11,L49           / / iNumberBlobBySize
                - ARI/1,12,VisAlgo,GetValueBlob,L11,L11,L31,L51
                - LET/L50,L31           / /
                - LET/L51,L32   / / XES Baricentro geométricos
                - LET/L52,L33           / / Y Baricentro geométrico
                - LET/L53,L34   / / Área
                - LET/L54,L35   / / Perímetro
                - LET/L55,L36           / / X Baricentro gravita
                - LET/L56,L37   / / Y Baricentro gravita
                - LET/L57,L38   / / Área de gravita
                - JMP/contin
                - JLT/L34,20000,contin
                - LET/L49,L49+1
                - JMP/LoopB
/ /
/ / Círculo blob
/ /
contin          - LET/L11,L49           / / iNumberBlobBySize,
                - LET/L12,0.1           / / dPixelGap,
                - ARI/1,12,VisAlgo,GetCircleBlob,L11,L12,L30,L38
                - LET/L61,L31   / / Rayo
                - LET/L62,L32   / / X Centro
                - LET/L63,L33   / / Y Centro
                - LET/L64,L34           / / Redondez

                - LET/L11,2             / / iTypeFirstElement, / / 1=line 2=radius
                - LET/L12,2             / / dLineDistance dCircleRadius,
                - LET/L13,2             / / dLineSin dCircleXCenter
                - LET/L14,2             / / dLineCos dCircleYCenter
                - LET/L15,2              / / iTypeSecondElement, / / 1=line 2=radius
                - LET/L16,2             / / dLineDistance dCircleRadius,
                - LET/L17,2             / / dLineSin dCircleXCenter
                - LET/L18,2             / / dLineCos dCircleYCenter
                - ARI/1,12,VisAlgo,PointsAcross2Elements,L11,L18,L30,L36
                - LET/L111,L31  / / piNumberPoint,
                - LET/L112,L32          / / pPAX,
                - LET/L113,L33  / / pPAY,
                - LET/L114,L34  / / pPBX,
                - LET/L115,L35  / / pPBY,
                - LET/L116,L36  / / pDist
                - LET/L11,L49   / / iNumberBlobBySize,
                - ARI/1,12,VisAlgo,GetEllipseBlob,L11,L11,L30,L44
```

```
                - LET/L101,L31    / / pdXc
                - LET/L102,L32    / / pdYc
                - LET/L103,L33    / / pdAngA
                - LET/L104,L34    / / pdRgA
                - LET/L105,L35    / / pdAngB
                - LET/L106,L36 / / pdRgB
                - LET/L107,L37              / / pdX1
                - LET/L108,L38              / / pdY1
                - LET/L109,L39              / / pdX2
                - LET/L110,L40              / / pdX2
                - LET/L111,L41              / / pdX3
                - LET/L112,L42              / / pdY3
                - LET/L113,L43              / / pdX4
                - LET/L114,L44              / / pdX4
/ /
/ / Dimensión blob
/ /
                - LET/L11,L49               / / iNumberBlobBySize,
                - ARI/1,12,VisAlgo,GetDimensionBlob,L11,L11,L30,L46
                - LET/L71,L31               / / XHigh
                - LET/L72,L32               / / YHigh
                - LET/L73,L33               / / XLow
                - LET/L74,L34               / / YLow
                - LET/L75,L35     / / XRigh
                - LET/L76,L36     / / YRight
                - LET/L77,L37     / / XLeft
                - LET/L78,L38     / / YLeft
                - LET/L79,L39     / / XHighAlongBaricenter
                - LET/L80,L40     / / YHighAlongBaricenter
                - LET/L81,L41     / / XLowAlongBaricenter
                - LET/L82,L42     / / YLowAlongBaricenter
                - LET/L83,L43     / / XRightAlongBaricenter
                - LET/L84,L44     / / YRightAlongBaricenter
                - LET/L85,L45     / / XLeftAlongBaricenter
                - LET/L86,L46     / / YLeftAlongBaricenter
/ /
/ / Dibuja blob
/ /
                - LET/L11,L49               / / iNumberBlobBySize,
                - LET/L12,0                 / / iFlagBlobPicture,
                - LET/L13,80                / / uRGBPictureColor,
                - LET/L14,1                 / / iFlagBlobEdge,
                - LET/L15,008               / / uRGBEdgeColor,
                - LET/L16,1                 / / iFlagValue,
                - LET/L17,0                 / / iFlagDimension,
                - LET/L18,0                 / / iFlagCircle,
                - LET/L19,0                 / / iFlagXAxesRect,
                - LET/L20,0                 / / iFlagYAxesRect,
                - LET/L21,1                 / / iFlagBordoROI,
                - LET/L22,1                 / / iFlagBordoInternoROI,
                - LET/L23,0                 / / iFlagCoordinateROI,
```

```
                - LET/L24,0x61          / / iFlagEllisse,
                - ARI/1,12,VisAlgo,DrawBlob,L11,L24,L30,L30
/ /
/ / Texto de las cuotas sobre vídeo
/ /
                - LET/L11,L51-70        / / IX,
                - LET/L12,L52           / / iY,
                - LET/L13,0             / / iFlagStringa
                - LET/L14,0             / / pStringa,
                - LET/L15,1             / / iFlagAng,
                - LET/L16,0             / / iFlagAngMilliToSessa,
                - LET/L17,L103          / / dAng,
                - LET/L18,0             / / iFlagX,
                - LET/L19,L51           / / dX,
                - LET/L20,0             / / iFlagY,
                - LET/L21,L52           / / dY,
                - LET/L22,2             / / Número font,
                - LET/L23,000           / / color,
                - ARI/1,12,VisAlgo,DrawTextOnVideo,L11,L23,L30,L30
;
; Espectáculo la imagen
;
                - ARI/1,12,VisionBase,ShowImage,L11,L11,L30,L31
                - HWA/1
                - RWA/L1,1
                - DIS/Orologio,L1
                - LET/G71,0
                - MOV/X,L51*G501,Y,L52*G502
                - DIS/1:X pixel,L101
                - DIS/2:Y pixel,L102
                - DIS/3:Ang notado, L103
                - LET/L141,L101*G501    ; Delta X
                - LET/L142,L102*G502    ; Delta Y

                - LET/L143,L103-166.2   ; Rincón
                - LET/L144,24.5*G501            ; Delta X Nominal
                - LET/L145,-7.5*G502            ; Delta Y Nominal

                - DIS/4:X Real mm, L141
                - DIS/5:Y Real mm,L142
                - DIS/6:Ang Rotación, L143


                - EVS/71
                - ISO/CAVALLOECS.PRG,11:-1:0,0,0:L141,L145:L146,L149
                - MOV/X,0,Y,0

                - JMP/INIZIO

NOBLOB          - LET / L62,0     / / X Centro
                - LET / L63,0     / / Y Centro
```

- LET / L53,0 / / X Baricentro gravita
- LET / L54,0 / / Y Baricentro gravita
- LET / L61,0 / / Rayo
- LET / L61,0 / / Rayo
- LET / L77,0 / / XLeft
- LET / L72,0 / / YHigh
- LET / L75,0 / / XRigh
- LET / L74,0 / / YLow

- LET/G71,0
- LET/L141,0,L142,0,L143,0,L144,0,L145,0

- EVS/71
**- ISO/CAVALLOECS.PRG,11:-1:0,0,0:L141,L145:L146,L149**

- JMP/INIZIO

## 11.26 WND: Wait Notify Detected

**WND** **Wait Notify Detected**

Función: Espera una señal de estado de error de los recursos ases
(Wait Notify Detected)

Parámetros: **a,b,c:[d,e]**

dónde:

| | | |
|---|---|---|
| **a** | = | máscara de los errores de que se quiere la señal<br>o bien 0 por todas las señales de anomalía |
| **b** | = | Variable local dónde es cargado el número eje o mandril<br>o grupo maquina si disfraza errores = 0x80000000 |
| **c** | = | Variable local dónde es cargado el tipo error o<br>máscara de los grupos maquina si disfraza errores = 0x80000000 |
| **y** | = | Timeout en milisegundos |
| **f** | = | Label de salto por timeout caducado |

Cargando la máscara error o señal es activado la solicitud de señal si eje cualquiera va en error con una de aquellas modalidades

**máscara de los errores de que se quiere la señal**

| | |
|---|---|
| 0 | Todo 7 las posibles anomalías |
| 0x8 | EJE EN SERVO ERROR (EJE FIRME) |
| 0x10 | EJE EN COLLISION (EJE EN MOVIMIENTO) |
| 0x20 | EJE NO EN POSICIÓN |
| 0x40 | EJE EN BLOQUE CUENTA |
| 0x2000000 | EJE DRIVER FAULT |
| 0x4000000 | EJE EN OVER TRAVEL POSITIVO |
| 0x8000000 | EJE EN OVER TRAVEL NEGATIVO |

**caso especial:**

| | |
|---|---|
| 0x80000000 | ANOMALÍA GRUPOS MAQUINA |

El tipo de anomalía eje es devuelto en la variable local **b** con los siguientes valores:

| **Tipo anomalía** | |
|---|---|
| 1 | EJE EN SERVO ERROR (EJE FIRME) |
| 2 | EJE EN COLLISION (EJE EN MOVIMIENTO) |
| 3 | EJE NO EN POSICIÓN |
| 4 | EJE EN BLOQUE CUENTA |
| 5 | EJE DRIVER FAULT |
| 6 | EJE EN OVER TRAVEL POSITIVO |
| 7 | EJE EN OVER TRAVEL NEGATIVO |

**Ejemplo 1:**

- WND/0x8,L2,L3:1000,LabelTimeout

0x8 EJE EN SERVO ERROR (EJE FIRME)
L2 = número eje o mandril
L3 = tipo error

**Ejemplo 2:**

```
A       - LET/L1,0
        - WND/L1,L2,L3
```

**Nota:** si la máscara de los errores requeridos, parámetro **a,** es:

| | | |
|---|---|---|
| 0x80000000 | | ANOMALÍA GRUPOS MAQUINA en los parámetros **b** y **c** |
| **b** | = | El número del grupo maquina en anomalía eje |
| **c** | = | La máscara de todos los grupos maquina en anomalía |

**Ejemplo 3:**

- WND/0x80000000,L2,L3:1000,LabelTimeout

L2 = número grupo maquina en anomalía
L3 = máscara grupos maquina en anomalía

## 11.27 WKY: Wait KeYboard

**WKY Wait KeYboard**

Función: Espera la comprimida de una tecla, Wait KeYboard,

Parámetros: **a**

dónde:

**a** = SCAN colas del dasto de esperar

**Ejemplo:**

- WKY/1 ; Espera la tecla ESC

- WKY/16 la Tecla Q
- WKY/17 la Tecla W
- WKY/18 la Tecla ES
- WKY/1 la Tecla ESC
- WKY/29 la Tecla Ctrl
- WKY/56 la Tecla Alto
- WKY/28 la Tecla Enter
- WKY/73 la Tecla PGup
- WKY/81 la Tecla Pgdown

Tablero de los Códigos Ascii Extensos correspondientes a los System Scan Colas., por Consola ECS,

| | |
|---|---|
| REPOSICIÓN | 0x6B |
| AXISH | 0x71 |
| START | 0x6E |
| REL | 0x6D |
| HOLD | 0x70 |

Tablero de los System Scan Colas, en salida por el Keyboard Controller,

| | |
|---|---|
| SP1 | 0x72 |
| SP2 | 0x73 |
| SP3 | 0x74 |
| SP4 | 0x75 |
| SP5 | 0x77 |
| SP6 | 0x78 |
| SP7 | 0x79 |
| SP8 | 0x55 |
| SGOUP | 0x58 |

## 11.28 NHL: NoHoLd

**NHL** **NoHoLd command**

Función: Inhabilita la realización de instrucciones TKM/1: de otros part program

Parámetros: **nadie**

La instrucción sirve para inhabilitar porciones de part program la das posibilidad de ser puestas en el estado HOD de parte de otros part program por la instrucción TKM/1:…..
Al fin de la parte de código crítico, por la instrucción YHL se puede volver en las condiciones normales si durante el esuzione de la parte entre NHL.. YHL, ha sido solicitado la operación de Hold tal función será activada enseguida después de YHL.

Ejemplo:

- NHL
..
. Zona no sospendibile de TKM/1:….
…
…
- YHL

## 11.29 YHL: YesHoLd

**YHL** **YesHoLd command**

Función: Habilita la realización de instrucciones TKM/1: de otros part program inhabilitado dal'istruzione NHL

Parámetros: **nadie**

La instrucción de YHL es el complementario del NHL, es utilizada para restablecer el estado normal de sospensionabilità por la instrucción TKM/1: … activada por otros part program que obran en paralelo.

Ejemplo:

```
- NHL
..
.          Zona no sospendibile de TKM/1:….
…
…
- YHL
```

## 11.30 GDT: Get Date and Time

**GDT** **Get Date and Time**

Función: Devuelve en los varaibili L o G la fecha y la hora actual

Parámetros: **a,b**

dónde:

**a** = Antes Variable Local o Global
**b** = Último Variable Local o Global

En ella a = año
a+1 = mes
a+2 = día
a+3 = ahora
a+4 = minutos
a+5 = segundo
a+6 = milisegundos

**Ejemplo:**

```
- GDT/L1,L7
- DIS/1:Anno          =, L1,4,0
- DIS/2:Mese          =, L2,4,0
- DIS/3:Giorno        =, L3,4,0
- DIS/4:Ora           =, L4,4,0
- DIS/5:Minuti        =, L5,4,0
- DIS/6:Secondi       =, L6,4,0
- DIS/7:Millisecondi  =, L7,4,0
```

## 11.31 GLN: Get Local Number

**GLN** **Get Local Number**

Función: Devuelve en el varaibile L o G el número de variables Locales del part program

Parámetros: **a**

dónde:

**a** = Variable Local o Global en cuyo es devuelto el número de locales definido en el part program

**Ejemplo:**

- GLN/L1
- DIS/1:Numero locales definís =, L1,4,0

## 11.32 GMI: Get Motion Information

**GMI** **Get Motion Information**

Función: Devuelve en el varaibili L o G el número de entes o movimientos ejecutados y de hacer de un part program de automación o una unidad ISO

Parámetros: **a:b,c[,d,e,f]**

dónde:

**a** = Label,Nome expediente del task, part program, de cerrar
[nomecall] [, nomefile.pp]
[nomefileISO.prg]
[numerotesta ISO]
**b** = Variable Local o Globales movimientos cargados
**c** = Variable Local o Globales movimientos cargados movimientos ejecutados
**d** = Variable Local o Global Feed vectorial
**y** = Variable Local o Global máscara de los movimientos
**f** = Variable Local o Global estado de la operación 1 si encontrara el part program o 0

**Ejemplo 1:**
Tool -
- TMM/10
- GMI/,Cos_Crash:L1,L2,L3,L4; Part program Cos_Crash.PP
- DIS/1:Movimenti Fatti,L2
- DIS/2:Movimenti Indrodottti,L1
- DIS/3:Feed,L3
- DIS/4:Delta, L1-L2
- JMP/Tool

**Ejemplo 2:**
Tool -
- TMM/10
- GMI/11:L1,L2,L3,L4; 11 Número cabeza ISO
- DIS/1:Movimenti Fatti,L1
- DIS/2:Movimenti Indrodottti,L2
- DIS/3:Feed,L3

- JMP/Tool

## 11.33 RTC: Read Timer and Counter

**RTC** **Read Timer e Conter**

Función: Devuelve en el varaibili L o G el valor de un Temporizador, TON y TOR, o de un Counter(CTU y CUD,

Parámetros: **a,b[,c,d]**

dónde:

**a** = Variable Local o Global contenedor el valor del temporizador o el contador
**b** = T o C con el relativo número
**c** = Valor máximo programado
**d** = Estado 1 o 0 del temporizador o contador

**Ejemplo:**

- RTC/L1,T1,L2,L3
- DIS/1:Valore del temporizador actual =, L1,7,0
- DIS/2:Valore del temporizador final =, L2,7,0
- DIS/1:Valore del estado del temporizador =, L3,7,0

## 11.34 SGL: Save GLobal

**SGL** **Save GLobal**

Función: Salva sobre disco con "commit" las variables Globales, los que son persistentes sobre "hard disk"

Parámetros: nadie

**Ejemplo:**

- SGL

Son salvadas sobre el HardDisk todas las 32768 variables Globales con una técnica que asegura la escritura en modlità "Commit "tal de garantizar el cumplimiento de la escritura manteniendo en todo caso el sistema anterior en caso de anomalía del sistema.

## 11.35 SHL: SHeLl

**SHL SHeLl**

Función: Un "Shell" de un applicativo ejecuta (. EXE, o de un procedimiento (. BAT, del sistema operativo Windows.

Parámetros: [a]:b[,c,d]

dónde:
**a** = Espera el cumplimiento del SHELL si 1 o bien 0 no espera
, Estándard = 0 no espera,
**b** = Nombre del procedimiento (Bat) o del eseguibile/applicativo (. EXE,
**c** = Visualización si 1 del shell o bien 0 si no se desea la visualización.
, Estándard = 1 Visualiza,
**d** = Modalidad de visualización:
0=SW_SHOW, Estándard 0 = SW_SHOW,
1 = SW_SHOWMAXIMIZED
2 = SW_SHOWMINIMIZED
3 = SW_RESTORE
4 = SW_HIDE

- SHL/Wait:Calc.exe,Show,ModeShow

**Ejemplo:**

- SHL/Calc.exe

- SHL/Esplorer.exe

- SHL/C:AxesBrain\Save.bat

## 11.36 G80: Fino ciclo fijo

**G80** **Fino ciclo fijo**

Función: Fino ciclo fijo (G80)

Parámetros:

dónde:

**Ejemplo:**

- G80

## 11.37 G81..89 Activa el ciclo fijo precisado

### G81 ..89 Activa el ciclo fijo precisado

Función: G81-G89 Activo el ciclo fijo precisado

Parámetros:

dónde:

**Ejemplo:**

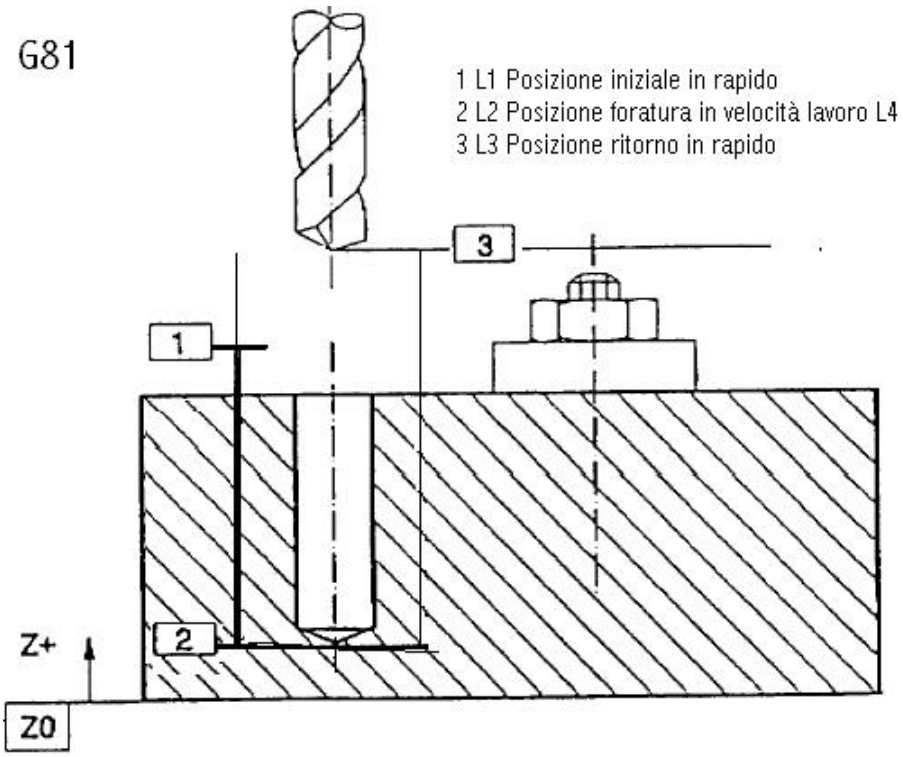
- G81 ..89

- G81/Z:L1,L2,L3,L4 , L1=PosZ_Inizio L2=PosZ_Foratura L3=PosZ_Ritorno L4=VelZ_Foratura,

- G82/Z:L1,L2,L3,L4,L5 , L1=PosZ_Inizio L2=PosZ_Foratura L3=PosZ_Ritorno L4=VelZ_Foratura
  L5 = Tiempo en Millisec,
- G83/Z: L1 = PosZ_Inizio
  L2 = PosZ_Foratura
  L3 = PosZ_Ritorno
  L4 = VelZ_Foratura
  L5 = Tiempo en Millisec
  L6 = Primo incremento de PosZ_Inizio
  L7 = Incremento Siguiente
  L8 = Incremento Seguridad

;
- G84/Z,M1:L2,L3,L4,L5,L6

L1 = PosZ_Inizio
L2 = PosZ_Foratura
L3 = PosZ_Ritorno
L4 = Paso
L5 = Perc. reducción velocidad por compensador o cero por maschiatura rígido
L6 = Speed mandril
- G84/Z,Sp:120,200,80,1.5,12,300; Maschiatura con compensador
- G84/Z,Sp:120,140,120,1.5,0,300; Maschiatura rígido
- MOV/X12,Y12
- MOV/X12,Y24
- MOV/X12,Y12
- G80




## G81 Pinchazo

- G81/Z:L1,L2,L3,L4

L1=PosZ_Inizio
L2=PosZ_Foratura
L3=PosZ_Ritorno
L4=VelZ_Foratura

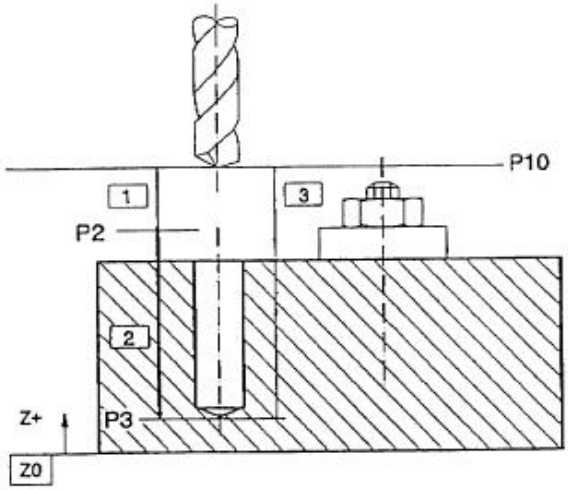


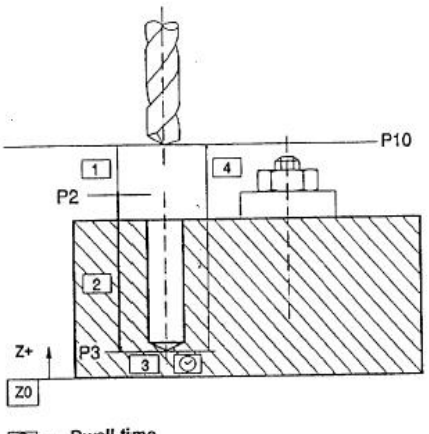

Fig. 7.1-2: Drilling cycle G81

Sequence of the drilling cycle G81:
1. Rapid traverse in the Z direction to the reference plane (P2).
2. Drill to the final depth required (P3) using the current feed rate.
3. Pull out in rapid traverse to the retract plane (P10).

## G82 Pinchazo con parada

- G82/Z:L1,L2,L3,L4,L5

L1=PosZ_Inizio
L2=PosZ_Foratura
L3=PosZ_Ritorno
L4=VelZ_Foratura
L5 = Tiempo en Millisec



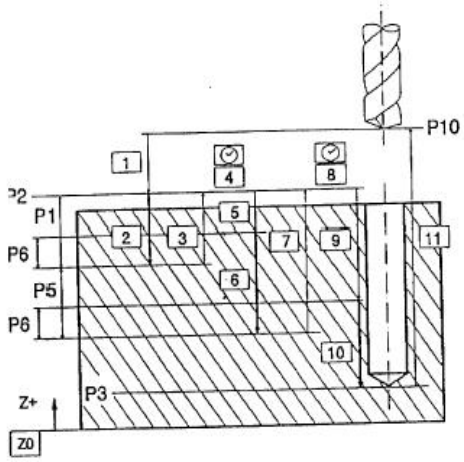

– **Dwell time**

Fig. 7.1-3: Drilling cycle G82

Sequence of the drilling cycle G82:
1. Rapid traverse in the Z direction to the reference plane (P2).
2. Drill to the final hole depth (P3) using the current feed rate.
3. Wait for the dwell time (P4) to elapse before breaking contact with workpiece.
4. Pull out in rapid traverse to the retract plane (P10).

## G83 Pinchazo con rotura trucciolo

- G83/Z: L1 = PosZ_Inizio
  L2 = PosZ_Foratura
  L3 = PosZ_Ritorno
  L4 = VelZ_Foratura
  L5 = Tiempo en Millisec
  L6 = Primo incremento de PosZ_Inizio
  L7 = Incremento Siguiente
  L8 = Incremento Seguridad



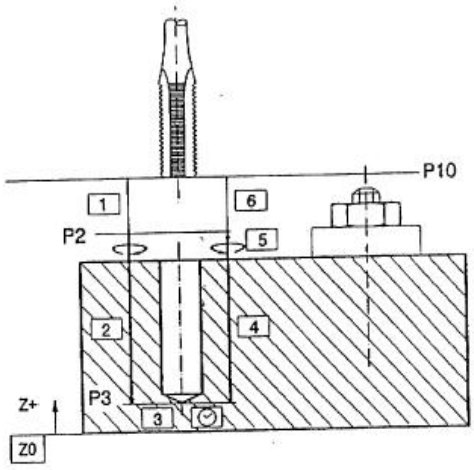

= Dwell time

Fig. 7.1-4: Drilling cycle G83

Sequence of the drilling cycle G83:
1. Rapid traverse in the Z direction to the reference plane (P2).
2. Drill using the current feed rate with the first feed-in value (P1) to the depth 1.
3. Pull out in rapid traverse to the reference plane (P2).
4. To allow the drill bit to cool, the Z axis remains on the reference plane (P2) during the dwell time (P4).
5. Move in rapid traverse to P1-P6 (first feed-in minus safety clearance) in the hole.
6. Drill to the depth 2: P6+P5 (safety clearance plus feed-in) using the current feed rate.

## G84 Maschiatura

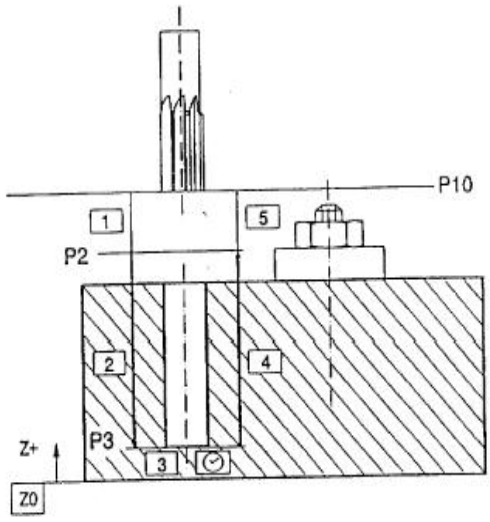
- G84/Z,Mandrino:
  L1 = cuota rápida entrada 1
  L2 = cuota trabajo final 3
  L3 = cuota rápida salida 6
  L4 = Paso
  L5=Perc.Stiramentoo 0 Rígido
  L6 = Speed Mandril




= Dwell time

Fig. 7.1-5: Drilling cycle G84

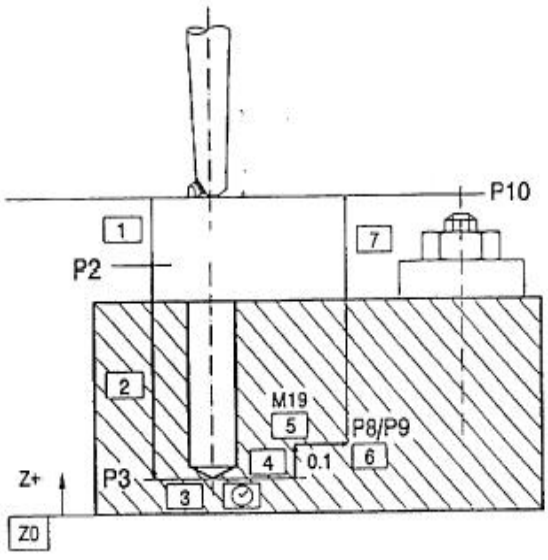
Sequence of the drilling cycle G84:
1. Rapid traverse in the Z direction to the reference plane (P2).
2. Drill using the current feed rate and clockwise rotating spindle (M03) to the final hole depth (P3).
3. Reverse spindle, i.e. the direction of rotation changes; a pause is made for the dwell time (P4).
4. Pull out using the current feed rate to the reference plane (P2).
5. Reverse spindle, i.e. spindle's direction of rotation is again clockwise.
6. Move in rapid traverse to the retract plane (P10).




= Dwell time

Fig. 7.1-6: Drilling cycle G85

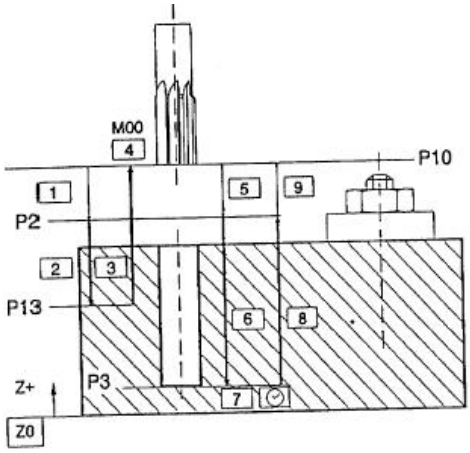
Sequence of the drilling cycle G85:
1. Rapid traverse in the Z direction on the reference plane ( P2).
2. Drill using the current feed rate to the final hole depth (P3).
3. Wait for the dwell time (P4) to elapse.
4. Pull out using the current feed rate to the reference plane (P2).
5. Move in rapid traverse to the retract plane (P10).




= Dwell time

Fig. 7.1-7: Drilling cycle G86

Sequence of the drilling cycle G86:
1. Rapid traverse in the Z direction to the reference plane (P2).
2. Bore to the final hole depth (P3) using the current feed rate.
3. Wait for the dwell time (P4) to elapse.
4. Move away 0.1 mm using the current feed rate.
5. Spindle is orientated to 0 degrees (M19).
6. Spindle is moved in the X or Y axis by the lift distance (P8 or P9).
7. Pull out to the retract plane (P10) in rapid traverse.




= **Dwell time**

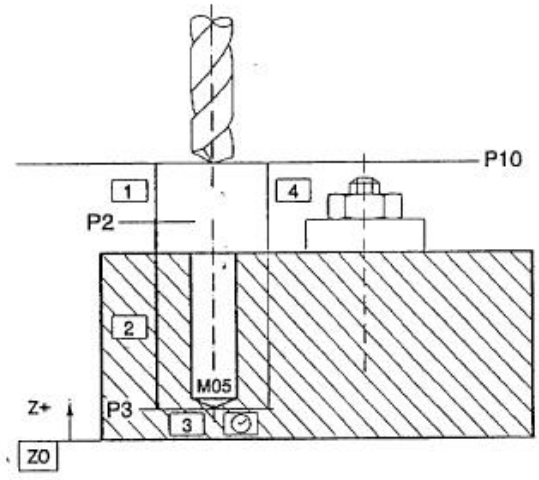
Fig. 7.1-8: Drilling cycle G87

Sequence of the drilling cycle G87:

1. Rapid traverse in the Z direction to the reference plane (P2).
2. Ream with the processing feed rate (P11) to the first reamed depth (P13).
3. Pull out to the retract plane (P10) with the retract feed rate (P12).
4. Halt feed rate to allow measuring of the hole, press START to continue with the processing.
5. Rapid traverse to the reference plane (P2).
6. Ream with the processing feed rate (P11) to the final hole depth (P3).
7. Wait for the dwell time (P4) to elapse.
8. Pull out with the retract feed rate (P12) to the reference plane (P2).
9. Move to the retract plane (P10) in rapid traverse.

Attention:

After leaving the drilling cycle G87 the retract feed rate is active!

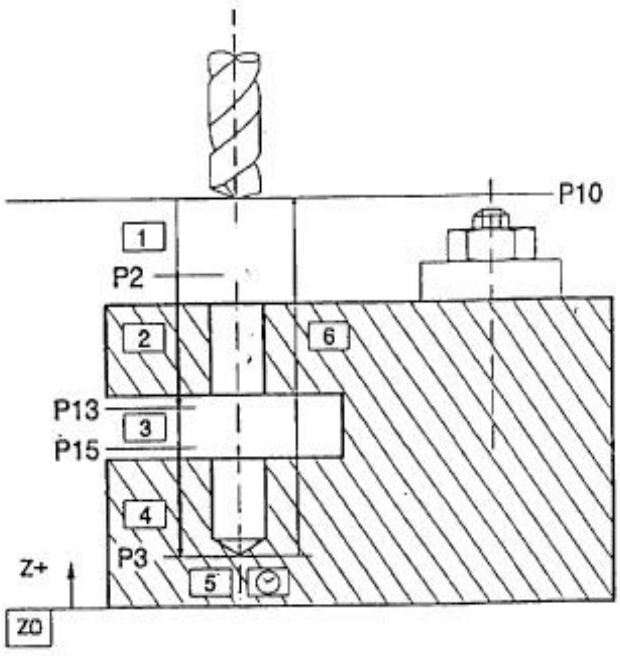



= **Dwell time**

Fig. 7.1-9: Drilling cycle G88

Sequence of the drilling cycle G88:

1. Rapid traverse in the Z direction to the reference plane (P2).
2. Bore to the final hole depth (P3) using the current feed rate.
3. Wait for the dwell time (P4) to elapse, after that the spindle stops.
4. Pull out to the retract plane (P10) in rapid traverse with stopped spindle.




= **Dwell time**

Fig. 7.1-10: Drilling cycle G89

Sequence of the drilling cycle G89:

1. Rapid traverse in the Z direction to the reference plane (P2).
2. Bore to the first drilling depth (P13) using the current feed rate.
3. Rapid traverse in the Z direction to the second drilling plane (P15).
4. Bore to the final hole depth (P3) using the current feed rate.
5. Wait for the dwell time (P4) to elapse.
6. Pull out in rapid traverse to the retract plane (P10).

## TABLEROS de conversión por instrucciones por el sistema ETEL

| | | |
|---|---|---|
| DMD_TYP_NONE | 0 | no type |
| DMD_TYP_IMMEDIATE | 0 | disabled or inmediatos value |
| DMD_TYP_USER | 1 | user registers |
| DMD_TYP_PPK | 2 | unidad de disco parameters |
| DMD_TYP_MONITOR | 3 | monitoring registers |
| DMD_TYP_SEQUENCE | 5 | sequence buffer |
| DMD_TYP_TRACE | 6 | trace buffer |
| DMD_TYP_ADDRESS | 7 | address value |
| DMD_TYP_LKT | 8 | movement lookup tables |
| DMD_TYP_TRIGGER | 9 | triggers buffer |
| DMD_TYP_REALTIME | 10 | realtime buffer |
| DMD_TYP_HALL_LKT | 11 | hall lookup tables (deprecated) |
| DMD_TYP_Y | 11 | Y type variables |
| DMD_TYP_FLOAT | 12 | float register |

**convertion constants**

| | | |
|---|---|---|
| DMD_CONV_DWORD | 0 | double word value without conversion |
| DMD_CONV_BOOL | 1 | boolean value |
| DMD_CONV_INT | 2 | integer value without conversion |
| DMD_CONV_LONG | 3 | long integer value without conversion |
| DMD_CONV_K8 | 3 | |
| DMD_CONV_K14 | 3 | |
| DMD_CONV_STRING | 4 | packed string value |
| DMD_CONV_FLOAT | 5 | float value |
| DMD_CONV_UPI | 10 | user position increment |
| DMD_CONV_USI | 11 | user speed increment |
| DMD_CONV_UAI | 12 | acceleration, user acceleration increment |
| DMD_CONV_DPI | 15 | unidad de disco position increment |
| DMD_CONV_DSI | 16 | unidad de disco speed increment |
| DMD_CONV_DAI | 17 | unidad de disco acceleration increment |
| DMD_CONV_C13 | 20 | current 13bit range |
| DMD_CONV_C14 | 21 | current 14bit range |
| DMD_CONV_C29 | 22 | current 29bit range |
| DMD_CONV_C15 | 23 | current 15bit range |
| DMD_CONV_CUR2 | 25 | los <SUP>2 </ SUP>, dissipation value |
| DMD_CONV_CUR2T | 26 | los <SUP>2 </ SUP>t, integration value |
| DMD_CONV_M82 | 28 | current limit en 10 MA unit, 100 <=> 1.0A |
| DMD_CONV_STI | 30 | slow time increment (500us-2ms) |
| DMD_CONV_FTI | 31 | fast time increment (125us-166us) |

| | | |
|---|---|---|
| DMD_CONV_CTI | 32 | current loop time increment (41us) |
| DMD_CONV_EXP10 | 33 | ten power factor |
| DMD_CONV_HSTI | 34 | half slow time increment |
| DMD_CONV_M242 | 35 | quartz frequency en Hz |
| DMD_CONV_QZTIME | 36 | interrupt time en sec = inc / m242 |
| DMD_CONV_SPEC2F | 37 | filter time, T = [fti] *, 2 <SUP>n </ SUP>-1, |
| DMD_CONV_TEMP | 38 | 2 <SUP>0 </ SUP> = 1 <=> 1.0 |
| DMD_CONV_UFTI | 39 | user friendly time increment |
| DMD_CONV_AVI | 40 | analog voltage increment -8192 <=> 10V 8192 <=> -10V |
| DMD_CONV_VOLT | 41 | 2 <SUP>0 </ SUP> = 1 <=> 1.0 |
| DMD_CONV_ENCOFF | 42 | 11bit with 2048 compensaciones |
| DMD_CONV_VOLT100 | 43 | , 2 <SUP>0 </ SUP>) /100 = 1 <=> 1.0 |
| DMD_CONV_PH11 | 44 | 2 <SUP>11 </ SUP> = 2048 <=> 360° |
| DMD_CONV_PH12 | 45 | 2 <SUP>12 </ SUP> = 4096 <=> 360° |
| DMD_CONV_PH28 | 46 | 2 <SUP>28 </ SUP> = 65536*4096 <=> 360° |
| DMD_CONV_AVI12BIT | 47 | analog voltage increment 2048 <= >10V 2048 <=> -10V |
| DMD_CONV_AVI16BIT | 48 | analog voltage increment 32767 <= >10V-32768 <=> -10V |
| DMD_CONV_BIT0 | 50 | 2 <SUP>0 </ SUP> = 1 <=> 1.0 |
| DMD_CONV_BIT5 | 55 | 2 <SUP>5 </ SUP> = 32 <=> 1.0 |
| DMD_CONV_BIT8 | 58 | 2 <SUP>8 </ SUP> = 256 <=> 1.0 |
| DMD_CONV_BIT9 | 59 | 2 <SUP>9 </ SUP> = 512 <=> 1.0 |
| DMD_CONV_BIT10 | 60 | 2 <SUP>10 </ SUP> = 1024 <=> 1.0 |
| DMD_CONV_BIT11 | 61 | 2 <SUP>11 </ SUP> = 2048 <=> 1.0 |
| DMD_CONV_BIT15 | 65 | 2 <SUP>15 </ SUP> = 32768 <=> 1.0 |
| DMD_CONV_BIT24 | 74 | 2 <SUP>24 </ SUP> = 256*65536 <=> 1.0 |
| DMD_CONV_BIT31 | 81 | 2 <SUP>31 </ SUP> = 32768*65536 <=> 1.0 |
| DMD_CONV_BIT11P2 | 82 | |
| DMD_CONV_UFPI | 85 | user friendly position increment |
| DMD_CONV_UFSI | 86 | user friendly speed increment |
| DMD_CONV_UFAI | 87 | user friendly acceleration increment |
| DMD_CONV_MSEC | 88 | milliseconds |
| DMD_CONV_K1 | 90 | pl prop gain, k(A/m, = k1 * Iref * dpi_factor / 2 <SUP>29 </ SUP> |
| DMD_CONV_K2 | 92 | pl speed retroalimentación gain, k(A/(m/s,) = k2 * Iref * dsi_factor <SUP>29 </ SUP> |
| DMD_CONV_K4 | 94 | pl integrator gain, k(A/(m*s,) = k1 * Iref * dpi_factor / 2 <SUP>29 </ SUP> / pl_time |
| DMD_CONV_K5 | 96 | anti-windup K[m/A]=K5*4096/(dpi_factor * Iref, |
| DMD_CONV_K9 | 98 | 1st order filter en pl |

| | | |
|---|---|---|
| DMD_CONV_K20 | 102 | speed retroalimentación, sec unit (m/(m/s)), F = k20 / 2 <SUP>16-k50 </ SUP> * dsi_factor / dpi_factor |
| DMD_CONV_K21 | 104 | speed retroalimentación, sec unit, m/(m/s <SUP>2 </ SUP>)), F = k20 / 2 <SUP>24-k50 </ SUP> * dai_factor / dpi_factor |
| DMD_CONV_K23 | 106 | commutation phase advance period/(m/s, |
| DMD_CONV_K75 | 108 | encoder múltiples index distance, 1/1024 * encoder perion unit |
| MD_CONV_K80 | 110 | cl prop gain delta[1/A] |
| MD_CONV_K81 | 112 | cl prop integrator delta[1/(A*s,] |
| MD_CONV_K82 | 114 | filter time, T = [cti] *, 2 <SUP>n </ SUP>-1, |
| MD_CONV_K94 | 116 | time en 2x current loop increment |
| MD_CONV_K95 | 118 | current plazos for k95 |
| MD_CONV_K96 | 120 | phase plazos for k96 |
| MD_CONV_K205 | 122 | came value, 100 <=> 1.0 |
| MD_CONV_PER_1000 | 123 | por thousand unit |
| MD_CONV_K239 | 124 | motor Kt factor en mN(m)/A, 1000 <=> 1.0mN(m)/A |

## ETEL arregla fast:

En la configuración DSTEB3 los mandos implementados son ligeramente diferentes de los mandos por el DSMAX y en particular:

**1, función ejecutas mandos ETEL, ves tablero mandas Etel,**

- ESE/NomeAsse:Numero comando,L#,L#+1

ejecuta la función ETEL:

param1 = L #
param2 = L#+1

execute_command_s (NomeAsse) cmd, param1, param2, FAST,

Ejemplo: para ejecutar la ejecución de una secuencia cuyo número es 1 sobre el eje X2

- LET/L1,1
- LET/L2,0
- ESE/X2:26:L1,L2

dónde:

| | |
|---|---|
| 26 = JMP | command |
| 1 = número secuencia llamada | param1 |
| 0 = no significativo mando 26 | param2 |

**2, función lees registros ETEL, ves tablero registras Etel,**

- ERR/NomeAsse:Variabile sobre cuyo cargado el valor letto:L#,L#+2:

Parámetros: **a:b: c,d**
dónde:

**a** = Número o nombre eje
**b** = Variable en cuyo es cargado el valor del registro externo
**c** = Variable de salida dónde soy declararte los parámetros del registro externo
**d** = Variable final dónde soy declararte los parámetros del registro externo

typ = Tipo registro L #
idx = Número registro L#+1
sidx = Número bajo registro L#+2

ejecuta la función ETEL:

get_register(int axis, int typ, int idx, int sidx, long * value,

**3, función escribes registros ETEL, ves tablero registras Etel,**

- EWR/NomeAsse:L#,L#+2:valore

Parámetros: **a:b,c:d**

dónde:

**a** = Número o nombre eje
**b** = Variable de salida dónde soy declararte los parámetros del registro externo
**c** = Variable final dónde soy declararte los parámetros del registro externo
**d** = Valor con que es cargado el valor del registro externo

typ = Tipo registro L #
idx = Número registro L#+1
sidx = Número bajo registro L#+2

ejecuta la función ETEL:

set_register(int axis, int typ, int idx, int sidx, long value,

**Los tipos de registros ETEL son:**

| | |
|---|---|
| TYP_NONE | 0 / * no type * / |
| TYP_IMMEDIATE | 0 / * disabled or inmediatos value * / |
| TYP_USER | 1 / * user registers * / |
| TYP_PPK | 2 / * unidad de disco parameters * / K |
| TYP_MONITOR | 3 / * monitoring registers * / M |
| TYP_SEQUENCE | 5 / * sequence buffer * / |
| TYP_TRACE | 6 / * trace buffer * / |
| TYP_ADDRESS | 7 / * address value * / |
| TYP_LKT | 8 / * movement lookup tables * / |
| TYP_TRIGGER | 9 / * triggers buffer * / |
| TYP_REALTIME | 10 / * realtime buffer * / |
| TYP_HALL_LKT | 11 / * hall lookup tables (deprecated) * / |
| TYP_Y | 11 / * Y type variables * / |

**4, función lees status ETEL**

- EWS/NomeAsse:3:L#,L #

Parámetros: **a:b:c,d**

dónde:

**a** = Número o nombre eje
**b** = NO significativo
**c** = Variable de salida dónde es cargado el estado
**d** = Valor final dónde es cargado el estado

| Bit0-31 | Valor | Significado |
|---|---|---|
| 0 | 0x00000001 | = User bit 0, could be modified by trigger fuctions or by K177 |
| 1 | 0x00000002 | = User bit 1, could be modified by trigger fuctions or by K177 |
| 2 | 0x00000004 | = User bit 2, could be modified by trigger fuctions or by K177 |
| 3 | 0x00000008 | = User bit 3, could be modified by trigger fuctions or by K177 |
| 4 | 0x00000010 | = User bit 4, could be modified by trigger fuctions or by K177 |
| 5 | 0x00000020 | = User bit 5, could be modified by trigger fuctions or by K177 |
| 6 | 0x00000040 | = User bit 6, could be modified by trigger fuctions or by K177 |
| 7 | 0x00000080 | = User bit 7, could be modified by trigger fuctions or by K177 |
| 8 | 0x00000100 | = User bit 8, could be modified by trigger fuctions or by K177 |
| 9 | 0x00000200 | = User bit 9, could be modified by trigger fuctions or by K177 |
| 10 | 0x00000400 | = User bit 10, could be modified by trigger fuctions or by K177 |
| 11 | 0x00000800 | = User bit 11, could be modified by trigger fuctions or by K177 |
| 12 | 0x00001000 | = User bit 12, could be modified by trigger fuctions or by K177 |
| 13 | 0x00002000 | = User bit 13, could be modified by trigger fuctions or by K177 |
| 14 | 0x00004000 | = User bit 14, could be modified by trigger fuctions or by K177 |
| 15 | 0x00008000 | = User bit 15, could be modified by trigger fuctions or by K177 |
| 16 | 0x00010000 | = The unidad de disco is en power on |
| 17 | 0x00020000 | = |
| 18 | 0x00040000 | = |
| 19 | 0x00080000 | = Motor axis present. Bit normaly plató to 1 |
| 20 | 0x00100000 | = Motor is executing a trajectory (moving) |
| 21 | 0x00200000 | = Motor en the position windows |
| 22 | 0x00400000 | = |
| 23 | 0x00800000 | = Unidad de disco global warning |
| 24 | 0x01000000 | = **Unidad de disco executing an internal sequence** |
| 25 | 0x02000000 | = |
| 26 | 0x04000000 | = Unidad de disco en error modas |
| 27 | 0x08000000 | = Trace busy flag is plató during a register trace acquisition |

28 0x10000000 =
29 0x20000000 =
30 0x40000000 =
31 0x80000000 =

**Ejemplo:**

- EWS/Y:0:L9,L9
- DIS/3:Stato eje Y,L9,8,-1
- JEQ/and(L9,0x01000000, 0x01000000,DriveExeInternalSequence

**Ejemplos instrucciones ETEL DSteb3:**

```
;
; Ejemplo lectura Regitro M7 position actual
;
        - LET/L1,3; 3=M 2=K 1=X
        - LET/L2,7
        - LET/L3,0
        - ERR/Y:L10:L1,L3
        - DIS/-1:M7(Y),L10
        - DIS/1:Valore M7 eje Y,L10
;
; Ejemplo contrata Registro User X3
;
        - LET/L1,1; 3=M 2=K 1=X
        - LET/L2,3
        - LET/L3,0
        - EWR/Y:L1,L3:12345
;
; Ejemplo lectura Registro User X3
;
        - LET/L1,1; 3=M 2=K 1=X
        - LET/L2,3
        - LET/L3,0
        - ERR/Y:L10:L1,L3
        - DIS/2:Valore X3 eje Y,L10


;
; Ejemplo mando power off 124 param 1 0=off
;
        - LET/L1,0; param 1 0=off
        - LET/L2,0; param 2
                                        ; 124 power
        - ESE/Y:124:L1,L2

        - TMM/1000
```

```
;
; Ejemplo mando power on 124 param 1 1=on
;
        - LET/L1,1; param 1 1=on
        - LET/L2,0; param 2
                                        ; 124 power
        - ESE/Y:124:L1,L2

;
; En el power on hace falta también dar el mando WTM
;
        - LET/L1,0; param 1
        - LET/L2,0; param 2
                                        ; 8 WTM
        - ESE/Y:8:L1,L2

;
        - TMM/1000

;
; Ejemplo lectura
;
        - EWS/Y:0:L9,L9
        - DIS/3:Stato eje Y,L9,8,-1
        - JEQ/and(L9,0x01000000, 0x01000000,DriveExeInternalSequence
```

## 11.38 ESE Exec Sequenze ( Sistema ETEL )

**ESE** **Exec Sequence**

Función: Ejecuta secuencias externas (Exec Sequence)

Parámetros: **a:b:[c,d]**

dónde:

**a** = Número o nombre eje
**b** = Número secuencia
**c** = Variable de salida dónde soy declararte los parámetros de la secuencia
**d** = Variable final dónde soy declararte los parámetros de la secuencia

external sequence - ESE / Eje:
Numero secuencia:
Da,A(Da+5)*n
De 1=Tipo registro
2=Numero registro
3=Numero bajo registro
4=Valore
5=Costante conversión

**Ejemplo:**

```
- ESE/X1:100:L101,L110
```

## 11.39 ERR External Read Registry ( Sistema ETEL )

**ERR External Read Registry**

Función: Lees registros externos (External Read Registry)

Parámetros: **a:b:c,d**

dónde:

**a** = Número o nombre eje
**b** = Variable en cuyo es cargado el valor del registro externo
**c** = Variable de salida dónde soy declararte los parámetros del registro externo
**d** = Variable final dónde soy declararte los parámetros del registro externo

external read register - ERR / Eje:
Local o Global:
Da,A(Da+5,
De 1=Tipo registro
2=Numero registro
3=Numero bajo registro
4=Valore
5=Costante conversión

**Ejemplo:**

```
ERR/X1:G11:L101,L105
```

## 11.40 EWR External Write Registry ( Sistema ETEL )

**EWR External Write Registry**

Función: Escribes registros externos (External Write Registry)

Parámetros: **a:b,c:d**

dónde:

**a** = Número o nombre eje
**b** = Variable de salida dónde soy declararte los parámetros del registro externo
**c** = Variable final dónde soy declararte los parámetros del registro externo
**d** = Valor con que es cargado el valor del registro externo

external write register - EWR / Eje:
Da,A(Da+5,:
Valor o Local o Global
De 1=Tipo registro
2=Numero registro
3=Numero bajo registro
4=Valore
5=Costante conversión

**Ejemplo:**

```
- EWR/X1:L101,L105:123
```

## 11.41 ECM External CoMmand ( Sistema ETEL )

**ECM External CoMmand**

Función: Ejecuta un mando externo (External CoMmand)

Parámetros: **a:b:c,d:e:f,g**

dónde:

**a** = Número o nombre eje
**b** = Número comand
**c** = Falsas o True Fast command
**d** = Falsas o True réport unidad de disco errors
**y** = Type parameters
**f** = Variable de salida dónde soy declararte los parámetros del mando
**g** = Valor final dónde soy declararte los parámetros del mando

external command - ECM/Asse:
Numero comand
Falsas o True Fast command,
Falsas o True réport unidad de disco errors:
Type parameters
Type parameter 0 = no param
1 = d 1 long
2 = los 1 double
3 = dd 2 long
4 = id 1 double 2 long
5 = de 1 long 2 double
6 = ii 2 double
7 = Number parameter free
Da,A(Da+3)*n
De 1=Type
2=Valore
3=Costante conversión

**Ejemplo:**

```
- ECM/X1:109:0,0:3:L11,L19
```

## 11.42 EWS External Wait Signal ( Sistema ETEL )

**EWS External Wait Signal**

Función: Espera una señal externa (External Wait Signal)

Parámetros: **a:b:c,d**

dónde:

**a** = Número o nombre eje
**b** = FlagEqualOrNotEqua
**c** = Variable de salida dónde son declarados los parámetros de la señal
**d** = Valor final dónde son declarados los parámetros de la señal

external sequence - EWS / Eje:
FlagEqualOrNotEqual:
Da,A
Valor 1 = mask.drive.present
2 = mask.drive.moving
3 = mask.drive.in_window
4 = mask.drive.sequence
5 = mask.drive.error
6 = mask.drive.trace
7 = mask.drive.warning
8 = mask.drive.breakpoint
9 = mask.drive.user

**Ejemplo:**

```
- EWS/X1:1:L101,L110
```

## 11.43 CLM Comando da Logica di Macchina ( Sistema ETEL )

**CLM** **Mando de lógica de máquina**

Función: Mando de Lógica de Máquina, Sistema ETEL,

Parámetros: **a:b:c,d**
dónde:
**a** = Número o nombre eje
**b** = Valor con que es cargado el valor del registro externo
**c** = Variable de salida dónde soy declararte los parámetros del registro externo
**d** = Variable final dónde soy declararte los parámetros del registro externo

Mando de lógica de machina - CLM / Eje:
Numero mando:
Da,A(Da+5,
De 1=Tipo registro
2=Numero registro
3=Numero bajo registro
4=Valore
5=Costante conversión

Nota: La señal '- 'delante del eje, indica que el mando es dirigido a los registros del DSMAX y no a las UNIDADES DE DISCO.
Los mandos son:

1 ida en pinchazo - CLM/ZA:1:Pos.finale pinchazo,
Pos.inizio pinchazo,
Velocidad Trabajo

2 vuelta veloz - CLM/ZA:2:Pos.finale vuelvo,
Pos.Sgancio X y Y,
Pos.FuoriPezzo,
Vel.Salita

7 Impone la distancia animo completado
-CLM/7:ZA,Distanza

8 ZOVR le aumento en porcentual Z y S
- CLM/ZA:8,
Incremento en porcentaje

# CAPÍTULO 12

## 12 Instrucciones por la integración con los demás entornos

.1-ARI Solicitud de ejecución de una instrucción por el entorno precisado en el primero parámetro y espera de la respuesta (vuelta efectuada con la función de "WriteServiceParametersAndContinue" al subsistema de automación AXESBRAIN)
(Ambient Request Instruction)

.2-SEC Plató event client

## 12.1 ARI: Ambient Request Instruction

**ARI** **Ambient Request Instruction**

Función: solicitud de ejecución de una instrucción por el entorno precisado en el primero parámetro y espera de la respuesta (vuelta efectuada con la función de "WriteServiceParametersAndContinue" al subsistema de automación AXESBRAIN)

Parámetros: **a:b,c,d,e1…en:f1.. fn:[g,h]**

dónde:

**a** = entorno interesado
- 1 Entorno "CLIENT"
- 2 Entorno o bajo sistema CNC GMSISO
- 3 Entorno o bajo sistema PLC GMSAWL
- 4 Entorno de análisis de la imagen GVS
- 101 Entorno o bajo sistema 1 OEM
- 102 Entorno o bajo sistema 2 OEM
- 10n Entorno o bajo sistema n OEM

**b** = llave de respuesta
**c** = mensaje
**d** = función
de **e1 a en** = variables Locales o Globales de trasladar a la solicitud
de **f1 a fn** = variables Locales o Globales de recibir en la respuesta a la solicitud
**g** = Timeout en caso de no contestada 0=infinito (Estándard)
**h** = Label de salto por Timeout vencido

**Ejemplo:**

- ARI/1:23,Messaggio,Funzione,G12,G45:G100,G200
- ARI/1,12,VisAlgo,GetValueBlob,L11,L11,L31,L51:1000,NonRisponde

Solicita un servicio del applicativo usuario en términos de datos, que cuando hallados serán provistos por la oportuna función de escritura u a otros subsistemas.
La aplicación del usuario predispuesto a ejecutar el richiesta,una se vuelve que ha acabado la tarea tiene que señalar por la función
"WriteServiceParametersAndContinue" que la operación ha sido ejecutada.

**Notas:**

Manda a las aplicaciones conectaron el siguiente acontecimiento:

```
HRESULT OnRequestServiceEvent (     BSTR    sMsg,
                                    int lSubSystem,
                                    int lKey,
                                    int iNumberParameter.
                                    VARIANT * pVar
                               )
```

**Parámetros**

| | |
|---|---|
| BSTR sMsg | Mensaje dependiente del servicio requerido |
| int lSubSystem | Subsistema que ha producido el mensaje<br>0 Arregla central<br>**1 AXESBRAIN**<br>2 GMSISO<br>3 GMSAWL<br>101 Primero subsistema OEMCapability<br>102 Secondo subsistema OEMCapability<br>...<br>...<br>109 noveno subsistema OEMCapability |
| int lKey | Llave por didtinguere las solicitudes<br>101 Solicitud de un servicio genérico<br>102 solicitud de un input de teclado<br>103 Solicitud de una función GVS<br>104 Solicitud de un procedimiento GVS |
| int iNumberParameter | Número de parámetros |
| VARIANT * pVar | Parámetros de la solicitud |

El proceso del fondo arregla se pone en espera mandando una solicitud él servicio, será nota el el ejecutor del servicio a llamar el método: "WriteServiceParametersAndContinue"

```
STDMETHODIMP WriteServiceParametersAndContinue (
                                        int         iNumberSubSystem,
                                        BSTR        sMsg,
                                        int         lParam1,
                                        int         lKey,
                                        int         iNumberParameter,
                                        VARIANT *   pVarParameter,
                                        int *       piError
                                        )
```

**Valor de vuelta**

| | |
|---|---|
| S_OK | vuelvo ok |
| S_FALSE | Error en los parámetros |

**Parámetros**

| | |
|---|---|
| int iNumberSubSystem | Tipo de identificación del subsistema<br>**1 AXESBRAIN**<br>2 GMSISO<br>3 GMSAWL<br>101 Primero subsistema OEMCapability<br>102 Secondo subsistema OEMCapability<br>...<br>...<br>109 noveno subsistema OEMCapability |
| BSTR sMsg | Mensaje |
| int lParam1 | Parámetro |
| int lKey | Llave de reconocimiento |
| int iNumberParameter | Número de parámetros |
| VARIANT * pVarParameter | Variant con el valor de cada parámetro |
| int * piError | Vuelve el tipo de error en el caso que stdmethodimp sea S_FALSE y en particular:<br>1 iNumberSubSystem errado<br>2 sMsg errado<br>3 lParam1 errado<br>4 lKey errado<br>5 iNumberParameter errado |

## 12.2 SEC: Set Event Client

**SEC** **Plató Event Client**

Función: Plató event client

Parámetros: **[a],b,c**

dónde:

**a** = iCookie, estándard –1,
**b** = Cordón de comunicación
**c** = valor de comunicación

**Ejemplo:**

```
/ / plató acontecimiento client - SEC/iCookie:MACHINE_LOGIC,1; Info
/ / plató acontecimiento client - SEC/iCookie:MACHINE_LOGIC,2; Warning
/ / plató acontecimiento client - SEC/iCookie:MACHINE_LOGIC,3; Allarm
/ / plató acontecimiento client - SEC/iCookie:MACHINE_LOGIC,4; Dinamic Área
/ / plató acontecimiento client - SEC/iCookie: prueba de comunicazione,g1

PartProgram[InOne]
        - DIS/11:AlarmInOne Input =, L1,3,0
        - DIS/12:AlarmInOne Status =, L2,1,0
        - RET

PartProgram[InZero]
        - DIS/11:AlarmInZero Input =, L1,3,0
        - DIS/12:AlarmInZero Status =, L2,1,0
        - RET


PartProgram[OutOne]
        - DIS/8:AlarmOutOne Salida =, L1,3,0
        - DIS/9:AlarmOutOne Status =, L2,1,0
        - SEC/MACHINE_LOGIC,3
        - RET

PartProgram[OutZero]
        - DIS/8:AlarmOutZero Salida =, L1,3,0
        - DIS/9:AlarmOutZero Status =, L2,1,0
        - SEC/MACHINE_LOGIC,3
        - RET
```

# CAPÍTULO 13

## 13 Instrucciones de comunicación

| | |
|---|---|
| .1-CSO | Conexión a un socket TCP/IP (Connect SOcket) |
| .2-LSO | Listen a un socket TCP/IP (Listen SOcket) |
| .3-RSO | Lees los datos que transitan sobre un socket TCP/IP, Read SOcket, |
| .4-TSO | Escribe sobre un socket TCP/IP, Write SOcket, |
| .5-DSO | Borra una conexión a un socket TCP/IP, Destroy SOcket, |
| .6-FSO | Borra los datos eventualmente recibidos sobre un socket TCP/IP, Free SOcket, |
| .7-GSO | Adquiere las informaciones de un socket TCP/IP, Get information SOcket, |
| .8-OSL | Abre una línea serial (Open Serial Line) |
| .9-RXL | Recibe dados por una línea serial (Receive Serial Line) |
| .10-TXL | Transmite datos sobre una línea serial, Trasmit Serial Line, |
| .11-CSL | Cierra la línea serial (Close Serial Line) |
| .12-FSL | Borra los datos eventualmente recibidos por una línea seriale(Free Serial Line, |

## 13.1 CSO: Connect SOcket

**CSO** **Connect SOcket**

Función: Conexión a un socket TCP/IP (Connect SOcket)

Parámetros: **a,b:c:d,e,f,g:[h]**

dónde:

| | | |
|---|---|---|
| **a** | = Número socket | 1-10 |
| **b** | = Número Lleva | 1025 -> |
| **c** | = Loc o Global vuelta: | 0=Ok 1=NON conexo |
| **d** | = IP connectA 000 | Ejemplo 127 |
| **y** | = IP connectB 000 | Ejemplo 17 |
| **f** | = IP connectC 000 | Ejemplo 2 |
| **g** | = IP connectD 000 | Ejemplo 7 |
| **h** | = Carácteres Buffer Recepción | Default1024 |

**Ejemplo:**

```
CSO/2,1027:L1:192,103,5,7
```

**Notas:**

El largo del buffer circular que es 1024 por estándard, es utilizado para adquirir los carácteres que llegan puesto che es stat abierto la conexión, con la instrucción RSO es adquirido por el part program por laloro elaboración. Con la instrucción FSO es borrado todos el byte recibido.

## 13.2 LSO: ListenSOcket

**LSO** **Listen SOcket**

Función: Listen a un socket TCP/IP (Listen SOcket)

Parámetros: **a,b:[c,e]**

dónde:

| | | |
|---|---|---|
| **a** | = Número socket | 1-10 |
| **b** | = Número Lleva | 1025 -> |
| **c** | = Loc o Global vuelta: | 0=Ok 1=Busy |
| **y** | = Carácteres Buffer Recepción 1024 | Estándard 1024 |

**Ejemplo:**

```
- LSO/1,1027:L1          ; Listen, se mete en escucha sobre la puerta 1027,
```

## 13.3 RSO: ReadSOcket

**RSO** **Read SOcket**

Función: Lees los datos que transitan sobre un socket TCP/IP, Read SOcket,

Parámetros: **a:b,c:[d,e,f]**

dónde:

**a** = Número socket
**b** = Dirección Principio variable
**c** = Dirección Fina variable
**d** = Dirección variable de Estado 0=OK 1=Linea no abierto 2=timeout caducado
**y** = Timeout, sec):0=infinito Estándard 0
**f** = Dirección variable número carácteres recibidos

**Ejemplo:**

- RSO/1:L11,L19:L2:L3

L11: primer carácter de recibir
L19: último carácter de recibir
L2: estado 0=OK 1=Linea no abierto 2=timeout caducado
L3: numero carácteres recibidos

## 13.4 TSO: WriteSOcket

**TSO** **Write SOcket**

Función: Escribe sobre un socket TCP/IP, Write SOcket,

Parámetros: **a:b,c:[e,f]**

dónde:

**a** = Número socket
**b** = Dirección Principio variable
**c** = Dirección Fina variable
**y** = Dirección variable de Estado 0=OK 1=Linea no abierto
**f** = Dirección variable número carácteres transmitidos L22

**Ejemplo:**

- TSO/2:L11,L19:L2:L3

L11: primer carácter de transmitir
L19: último carácter de transmitir

L2: estado socket 0=OK 1=Linea no abierto
L3: Dirijo variable número carácteres transmitidos L22

## 13.5 DSO: DestroySOcket

**DSO** **Destroy SOcket**

Función: Borra una conexión a un socket TCP/IP, Destroy SOcket,

Parámetros: **a**

dónde:

**a** = Número socket

**Ejemplo:**

- DSO/1

## 13.6 FSO: FreeSOcket

**FSO** **Free SOcket**

Función: Borra los datos eventualmente recibidos sobre un socket TCP/IP, Free SOcket,

Parámetros: **a**

dónde:

**a** = Número socket

**Ejemplo:**

- FSO/1

## 13.7 GSO: Get information SOcket

**GSO** **Get information SOcket**

Función: Adquiere las informaciones de un socket TCP/IP, Get information SOcket,

Parámetros: **a,b**

dónde:

**a** = Número socket
**b** = Variable de estado dónde: 0 = nada 1=WaitForListen 2=ListenWithConnect 3=Connect

**Ejemplo:**

```
        - GSO/1,L1
```

**Ejemplo sobre la transmisión sobre socket TCP/IP:**

```
; Connect
        - CSO/2,1027:L1:192,103,5,7
        - LET/L1,1
        - LET/L11,0x41,L12,0x42,L13,0x43
aa      - TSO/2:L11,L19:L2:L3
        - RSO/2:L11,L19:L2:L3
        - DIS/1:locale,L11
        - TMM/1000
        - LET/L1,L1+1
        - LET/L11,L11+1,L12,L12+1,L13,L13+1
        - JLE/L1,10,aa
        - DSO/2:

; Listen
        - LSO/1,1027:L1
aa      -
        - RSO/1:L11,L19:L2:L3
        - TSO/1:L11,L19:L2:L3
        - JLE/L11,0x41+10,aa
        - DSO/1
```

## Ejemplo doble enlace socket

```
; Programa de prueba de la transmisión sobre red (Client)

        - LET/L11,0                     ; Inicialización valor
        - DIS/1:Connessione en curso...  ; Prensa "Conexión en curso"...
        - DIS/2:                        ; Limpia la raya
        - DIS/3:                        ; Limpia la raya
        - CSO/1,1027:L1:192,103,5,1     ; Connect, se conecta all' dirijo 192.103.5.1 y sobre la puerta 1027,
        - DIS/1:Connesso                ; Prensa "Conectada"
LOOP    -
        - TSO/1:L11,L19:L2:L3           ; Transmite el valor
        - DIS/2:Trasmesso, L11          ; Prensa el valor transmitido
        - ADD/L11,1                     ; Incrementa el valor
        - TMM/300                       ; Pausa de 300 ms
        - JLT/L11,256,LOOP              ; Si el valor es mayor de 255 acaba el comuncazione     -
DSO/1                                   ; Cierra la comunicación
        - DIS/1:                        ; Limpia la raya
        - DIS/3:Connessione acabado!    ; Prensa "Conexión acabada"!


; Programa de prueba de la transmisión sobre red (Servidor)

        - DIS/1:Attesa conexión...      ; Prensa "Esperada conexión"...
        - DIS/2:                        ; Limpia la raya
        - DIS/3:                        ; Limpia la raya
        - LSO/1,1027:L1                 ; Listen, se mete en escucha sobre la puerta 1027,
LOOP    -
        - RSO/1:L11,L19:L2:L3           ; Se pone en espera de recibir algo
        - DIS/1:Connesso                ; Prensa "Conectada"
        - DIS/2:Ricevuto, L11           ; Prensa el valor rivevuto
        - JLE/L11,254,LOOP              ; Si el valor es mayor de 254 acaba el comuncazione
        - DSO/1                         ; Cierra la comunicación

        - DIS/1:                        ; Limpia la raya
        - DIS/3:Connessione acabado!    ; Prensa "Conexión acabada"!
```

## 13.8 OSL: Open Serial Line

**OSL** **Open Serial Line**

Función: Abre una línea serial (Open Serial Line)

Parámetros: **a:b:[c:d,e,f,g,h,i]**

dónde:

| | | | |
|---|---|---|---|
| **a** | = Número serial 1-n | | |
| **b** | = Tipo Protocolllo: | 0=free,1=C.U. | |
| **c** | = Loc o Global vuelta0 = Ok 1=Busy | | |
| **d** | = BaudRate | 1200-115000 | Estándard 9600 |
| **e** | = Igualdad | 0=No parity 1=Dispari 2 = Igual | Estándard 0 |
| **f** | = NumeroBit | 7-8 | Estándard 8 |
| **g** | = BitStop | 1-2 | Estándard 1 |
| **h** | = Carácteres Buffer Recepción | 1024 | Estándard 1024 |
| **los** | = Flag RS485 | 0=no 1=si | Estándard 0 |

**Ejemplo:**

```
- OSL/1:0:L1:9600,0,8,1
```

**Notas:**

El largo del buffer circular que es 1024 por estándard, es utilizado para adquirir los carácteres que llegan puesto che es stat abierto la línea serial, con la instrucción RXL es adquirido por el part program por laloro elaboración. Con la instrucción FSL es borrado todos el byte recibido.

Los seriales son aquéllos administrados por el sistema operativo de la máquina.

El flag RS485 si 1 activa la gestión según la modalidad RS485

### 13.9 RXL: Receive Serial Line

**RXL** **Receive Serial Line**

Función: Recibe dados por una línea serial (Receive Serial Line)

Parámetros: **a:b,c:[d,e,f]**

dónde:
**a** = Número serial
**b** = Dirección Principio variable
**c** = Dirección Fina variable
**d** = Dirección variable de Estado 0=OK 1=Linea no abierto 2=timeout caducado
**e** = Timeout(sec, 0=infinito -1 = le vuelve en **f** el número de carácteres en el buffer de recepción sin cargar nada en **b** y **c**
**f** = Dirección variable número carácteres recibidos

Nota si **y** Timeout(sec, = -1 no es cargado carácteres, pero en **f** viene puesto el número de carácteres recibidos en el buffer interior de recepción

## Ejemplo:

```
- RXL/1:L1,L1:L21:0:L3
```

L1: primer carácter de recibir
L20: último carácter de recibir
L21: 0=OK 1=Linea no abierto 2=timeout caducado
L3: numero carácteres recibidos

### 13.10 TXL: Trasmit Serial Line

**TXL** **Trasmit Serial Line**

Función: Transmite datos sobre una línea serial, Trasmit Serial Line,

Parámetros: **a:b,c:[d:e]**

dónde:
**a** = Número serial
**b** = Dirección Principio variable
**c** = Dirección Fina variable
**d** = Dirección variable de Estado 0=OK 1=Linea no abierto
**y** = Dirección variable número carácteres transmitidos

## Ejemplo:

- TXL/1:L1, L20:L2:L3

L1: primer carácter de transmitir
L20: último carácter de transmitir
L2: 0=OK 1=Linea no abierto
L3: Dirijo variable número carácteres transmitidos

## 13.11 CSL: Trasmit Serial Line

**CSL** **Close Serial Line**

Función: Cierra la línea serial (Close Serial Line)

Parámetros: **a**

dónde:
**a** = Número serial

**Ejemplo:**

- CSL/1

## 13.12 FSL: FreeSerial Line

**FSL** **FreeSerial Line**

Función: Borra los datos eventualmente recibidos por una línea seriale(Free Serial Line,

Parámetros: **a**

dónde:
**a** = Número serial

**Ejemplo:**

```
-FSL/1
```

**Ejemplo gestión de la línea serial 1 sobre un dispositivo cambio utensilios:**

```
                - CAL/AzzTamburo:L1
                - DIS/1:Azzeramento Eseguito=,L1
                - CAL/ToolTamburo:L1,449
                - DIS/2:Comando Eseguito,L1


PartProgram[AzzTamburo]
                - LET/L11,8,L12,0x44,L13,3
                - LET/L14,3,L15,0,L16,0,L17,0,L18,0

                - LET/L19,L11+L12+L13+L14+L15+L16+L17+L18
                - LET/L19,and(-L19,0xFF,

                - OSL/1:0:L1:9600,0,8,1
                - FSL/1
                - TXL/1:L11,L19:L2:L3
                - RXL/1:L1,L1:L2:0:L3
                - JEQ/L1,6,O_OK
                - CSL/1
                - LET/L1,-1
                - RET
O_OK            - RXL/1:L1,l5
                - CSL/1
                - LET/L1,L4
                - RET

PartProgram[ToolTamburo]
; L2 Número tool
                - LET/L11,8,L12,0x44,L13,3
                - LET/L14,1,L15,0,L16,0
                - LET/L17,and(L2/256,0xFF),L18,and(L2,0xFF,

                - LET/L19,L11+L12+L13+L14+L15+L16+L17+L18
                - LET/L19,and(-L19,0xFF,

                - OSL/1:0:L1:9600,0,8,1
                - FSL/1
                - TXL/1:L11,L19:L2:L3
                - RXL/1:L1,L1:L2:0:L3
                - JEQ/L1,6,O_OK
                - CSL/1
                - LET/L1,-1
                - RET
O_OK            - RXL/1:L1,l5
                - CSL/1
                - LET/L1,L4
                - RET
```

# CAPÍTULO 14

## 14 Instrucciones AWL PLC

| | |
|---|---|
| .1-NET | PLC Network |
| .2-LD | PLC Cargo operación |
| .3-LDN | PLC Cargo operación negada |
| .4-A. | PLC Combinado el valor de bit por AND |
| .5-AN | PLC Combinado el valor de bit negado por AND |
| .6-O | PLC Combinado el valor de bit por OR |
| .7-ON | PLC Combinado el valor de bit negado por OR |
| .8-EU | PLC Encuesta de frente positiva |
| .9-ED | PLC Encuesta de frente negativa |
| .10-EQU | PLC Asignado, copia en el parámetro precisado el valor superior del stack |
| .11-S | PLC Impuesto a 1 el número de puntos precisado si el stack es 1 |
| .12-R | PLC Impuesto a 0 el número de puntos precisado si el stack es 1 |
| .13-LPP | PLC Cobro lógico |
| .14-LPS | PLC Duplicación lógica |
| .15-LRD | PLC Copia lógica |
| .16-ALD | PLC Combinado lo primero y el segundo elemento por AND |
| .17-OLD | PLC Combinado lo primero y el segundo elemento por OR |
| .18-NOT | PLC Modificación del valor superior |
| .19-LEQ | PLC Confrontado dos valores si igual cargo con el stack 1 si no 0 |
| .20-LGE | PLC Confrontado dos valores si igual o mayor cargo con el stack 1 si no 0 |
| .21-LLE | PLC Confrontado dos valores si igual o menor cargo con el stack 1 si no 0 |
| .22-AEQ | PLC Confrontado dos valores si iguales hace el AND de 1 con el stack |
| .23-AGE | PLC Confrontado dos valores si iguales o mayor hace el AND de 1 con el stack |
| .24-ALE | PLC Confrontado dos valores si iguales o menor hace el AND de 1 con el stack |
| .25-OEQ | PLC Confrontado dos valores si iguales hace el OR de 1 con el stack |
| .26-OGE | PLC Confrontado dos valores si iguales o mayor hace el OR de 1 con el stack |
| .27-OLE | PLC Confrontado dos valores si iguales o menor hace el OR de 1 con el stack |
| .28-PEX | PLC Ejecutado sobre 1 en stack la instrucción AXESBRAIN |
| .29-TON | PLC Temporizador sin retención |
| .30-TOR | PLC Temporizador con retención |
| .31-CTU | PLC Counter Up |
| .32-CUD | PLC Counter Up y Down |

**Operaciones especiales a contactos**

El **Contacto normalmente abierto es** cerrado (on) si el valor del bit interrogado dirección **n es** 1.
En AWL el contacto normalmente abierto es representado por las operaciones del tipo:
**Cargada operación, Combina el valor de bit por And, Combina el valor de bit por OR.**
Estas operaciones, respectivamente, cargan el valor de bit de la dirección n en el valor superior del stack lógico, o combinan por And u OR el valor bit de la dirección n con el valor superior del stack lógico.

**Normalmente contacto cerrado es** cerrado (on) si el valor del bit interrogado dirección n es 0
En AWL el contacto normalmente cerrado es representado por las operaciones del tipo:
**Carga el valor de bit negado, Combina el valor de bit negado por And, Combina el valor de bit negado por OR.**
Estas operaciones cargan el valor de bit de la dirección n en el valor superior del stack lógico, o combinan por And u Or el valor bit de la dirección n con el valor superior del stack lógico.

Le descritas operaciones consiguen el valor dirigido por el registro de las imágenes de proceso, si tal valor es puesto al día al principio de cada ciclo UCP.

**Contactas directos**
El **Contacto directo normalmente abierto es** cerrado (on) si el valor del bit de la entrada físico dirigido n es **1.**
En AWL el contacto directo normalmente abierto es representado por las operaciones del tipo
**Carga directamente el valor de bit, Combina directamente bit por And, y Combina directamente bit por OR.**
Estas operaciones, respectivamente, cargan directamente el valor de bit de la dirección n al valor superior del stack lógico, o combinan directamente por And u OR el valor bit de la entrada físico dirigido n con el valor superior del stack lógico.

**El Contacto directo normalmente cerrado es** cerrado (on) si el valor del bit de la entrada físico dirigido n es 0.
En AWL el contacto directo normalmente cerrado es representado por las operaciones del tipo **Carga directamente el valor de bit negado, Combina directamente** el **valor de bit negado por And, y Combina directamente el valor de bit negado por OR.** Estas operaciones, respectivamente, cargan directamente el valor de bit negado por la dirección n en el valor superior del stack lógico, o combinan directamente por And u OR el valor bit negado de la dirección n con el valor superior del stack lógico.
Le descritas operaciones consiguen el valor dirigido por la entrada física cuando son ejecutadas, pero el registro de las imágenes de proceso no es puesto al día.

**Contacto Not**
El contacto NOT modifica el estado de las señales. Si el flujo de corriente alcanza el contacto Not, éste es parado. Si el flujo no alcanza el contacto Not, éste engendra flujo de corriente.
En AWL la operación **Negación del valor superior** modifica el valor superior del stack de 0 a 1, o de 1 a O.

**Normativas semántica**

El carácter **'#**indica que el valor de la expresión numérica es interpretado como valor 0 o bien 1.

**Ejemplo 1:**

- NET/1
- LD/#1 ; Es cargado el valor 1 en el STACK

**Ejemplo 2:**

; Ejemplo del dgestione de los Driver VALE para dar y sacar las capacitaciones a los ases

- NET/1
- LD/DriveOKZ011
- A/DriveOKZ021
- A/DriveOKS011
- A/DriveOKS021
- PEX/-LET/L1,1
- NOT
- PEX/-LET/L1,0

- NET/2
- LD/#L1
- EU
- PEX/-PWO; Habilita potencia ases
- PEX/-RST/5; Reposición Arregla sin condicionar las frentes de los I/O

- NET/3
- LD/#L1
- ED
- PEX/-EMC; Inhabilita potencia ases
- PEX/-RST/5; Reposición Arregla sin condicionar las frentes de los I/O

**Ejemplo 3: Ejemplo de accenzione Maquina**

| | | |
|---|---|---|
| **entrada** | **"IN1"** | **Pulsante Inserción Manda** |
| **salida** | **"OUT12"** | **Podridos Mandos** |
| **Merker** | **"M1.1"** | **Merker Marcha Mandos integrado** |
| **Merker** | **"M1.2"** | **Merker Merker Capacitación Ases VALE** |

```
Loop   - TMM/100

       - NET/1
       - LD/M1.1
       - TON/1,5000

       - IDO/M1.1,1,MacchinaInserita

       - NET/2
       - LD/IN1
       - EU
       - EQU/M1.1          ; Inserción manda
       - S/OUT12           ; TLR podridos mandos ON
       - R / M1.2          ; Merker Capacitación Ases KO

       - IDO/M1.1,0, Loop

MacchinaInserita -

       - NET/3
       - LD/DriveOKZ011
       - A/DriveOKZ021
       - A/DriveOKS011
       - A/DriveOKS021
       - PEX/-LET/L1,1
       - NOT
       - PEX/-LET/L1,0

       - NET/4
       - LD/#L1
       - EU
       - PEX/-PWO          ; Habilita potencia ases
       - PEX/-RST/5        ; Reposición Arregla sin condicionar las frentes de los I/O
       - S/M1.2            ; Merker Capacitación Ases VALE
       - PEX/-SEC/CLR_MESSAGE,1
```

```
        - NET/5
        - LD/#L1
        - ED
        - PEX/-EMC          ; Inhabilita potencia ases
        - PEX/-RST/5        ; Reposición Arregla sin condicionar las frentes de los I/O
        - R/M1. 2           ; Merker Capacitación Ases KO

        - IDO/M1.2,1, Loop

        - NET/4
        - LD/T1
        - PEX/-JMP/AzionamentiNonSalgono
        - JMP / Loop

AzionamentiNonSalgono -
        - SEC/SET_MESSAGE,1
        - JMP/Loop
```

## 14.1 NET: PLC Network

**NET** **PLC Network**

Función: Inicializa una rama NETWORK lógico

Parámetros: **a**

dónde:
**a** = Nombre de la rama de 1 a 128

### Ejemplo:

- NET/1

## 14.2 LD: PLC Cargo operación en el stack

**LD** **PLC Cargo operación en el stack**

Función: El operador carga en el stack

Parámetros: **a**

dónde:
**a** = Nombre del operador:
Input
Salida
Temporizador 1-32
Contador 1-32

### Ejemplo:

- NET/1
- LD/T1

## 14.3 LDN: PLC Cargo operación negada en el stack

**LDN** **PLC Cargo operación negada en el stack**

Función: El operador negado carga en el stack

Parámetros: **a**

dónde:
**a** = Nombre del operador:
Input
Salida
Temporizador 1-32
Contador 1-32

### Ejemplo:

- NET/1
- LDN/T1

## 14.4 A: PLC combina el valor de bit por AND

**A PLC combina el valor de bit por AND**

Función: Combina el valor del bit con el stack por AND

Parámetros: **a,…,an**

dónde:
**a,…,an** = Nombre del operador:
Input
Salida
Temporizador 1-32
Contador 1-32

### Ejemplo:

```
- NET/1
- LD/I_1
- A/I_12
```

## 14.5 AN: PLC combina el valor de bit negado por AND

**A PLC combina el valor de bit negado por AND**

Función: Combina el valor del bit negado con el stack por AND

Parámetros: **a,…,an**

dónde:
**a,…,an** = Nombre del operador:
Input
Salida
Temporizador 1-32
Contador 1-32

### Ejemplo:

```
- NET/1
- LD/I_1
- AN/I_12
```

## 14.6 O: PLC combina el valor de bit por OR

**O** **PLC combina el valor de bit por OR**

Función: Combina el valor del bit con el stack por OR

Parámetros: **a,…,an**

dónde:
**a,…,an** = Nombre del operador:
Input
Salida
Temporizador 1-32
Contador 1-32

### Ejemplo:

- NET/1
- LD/I_1
- O/I_12

## 14.7 ON: PLC combina el valor de bit negado por OR

**ON** **PLC combina el valor de bit negado por OR**

Función: Combina el valor del bit negado con el stack por OR

Parámetros: **a,…,an**

dónde:
**a,…,an** = Nombre del operador:
Input
Salida
Temporizador 1-32
Contador 1-32

### Ejemplo:

- NET/1
- LD/I_1
- ON/I_12

## 14.8 EU: PLC nota frente positiva

**EU** **PLC nota frente positiva**

Función: Nota la frente positiva en el valor superior del stack

**Transición positiva** permite a la corriente de circular por un ciclo de barrido, por cada transición de 0 (off) a 1 (on).
En AWL esta operación es representada como Encuesta de **frente positiva.** Si ella nota de un ciclo de barrido al otro una transición de **0** a **1** en el valor superior del stack, tal valor es **programado a 1**; de otro modo a 0.

Parámetros: **nadie**

### Ejemplo:

- NET/1
- LD/I_1
- EU

## 14.9 ED: PLC nota frente positiva

**ED** **PLC nota frente positiva**

Función: Nota la frente positiva en el valor superior del stack

**Transición negativa permite a la corriente de circular por** un ciclo de barrido, por cada transición de 1 (on) a 0 (off).
En AWL esta operación es representada como **Encuesta de frente negativa.** Si ella nota de un ciclo de barrido al otro una **transición de 1 a 0** en el valor superior del stack, tal valor es **programado a 1**; de otro modo a 0.

Parámetros: **nadie**

### Ejemplo:

```
- NET/1
- LD/I_1
- ED
```

## 14.10 EQU: PLC Asignado, copia en el parámetro precisado el valor superior del stack

**EQU PLC Asignado, copia en el parámetro precisado el valor superior del stack**

Función: Asigna, copia en el parámetro precisado el valor superior del stack

**Asigna**

Cuando es ejecutada la operación Asigna, es activado el parámetro precisado (n).

En AWL la operación Asigna copia en el parámetro precisado (n) el valor superior del stack.

Parámetros: **a,…,an**

dónde:
**a,…,an** = Nombre del operador:
Salida
Input

## Ejemplo:

- NET/1
- LD/I_1
- EQU/O_12

## 14.11 S: PLC Impuesto a 1 el número de puntos precisado si el stack es 1

**S PLC Impuesto a 1 el número de puntos precisado si el stack es 1**

Función: Impuesto a 1 el número de puntos precisado si el stack es 1

**Impuesto**
Cuando son ejecutadas las operaciones **Impuestas es** programado, activado, el número de puntos precisado por el operando

Parámetros: **a,…,an**

dónde:
**a,…,an** = Nombre del operador:
Salida
Input

## Ejemplo:

- NET/1
- LD/I_1
- S/O_12

### 14.12 R: PLC Impuesto a 0 el número de puntos precisado si el stack es 1

**R PLC Impuesto a 0 el número de puntos precisado si el stack es 1**

Función: Impuesto a 0 el número de puntos precisado si el stack es 1

**Resetta**

Cuando son ejecutadas las operaciones **Resetta,** desactivado, el número de puntos precisado por el operando
Utilizando la operación Resetta, si el operando es indicado un bit como T o C, son borrados el bit de temporizzazione/conteggio o el valor corriente de temporizzazione/conteggio.

Parámetros: **a,…,an**

dónde:
**a,…,an** = Nombre del operador:
Salida
Input

## Ejemplo:

```
- NET/1
- LD/I_1
- R/O_12
```

## 14.13 LPS: PLC Duplicación lógica

### LPS PLS Duplicación lógica

Función: Duplicación lógica

La operación **Duplicación lógica** duplica el valor superior del stack y coloca este valor en el stack.
El valor más bajo al fondo del stack es trasladado fuera y debe ser perdido.

Parámetros: **nadie**

Esta función es importante siendo el principio de abertura de una rama en paralelo

_……._____LPS____ **Nuevo Rama en desarrollo**
**|**
**| _____Vecchio Rama congelado**

## Ejemplo:

- NET/1
- LD/I_1
- A/I_2
- LPS
- A/I_3

## 14.14 LPP: PLC Cobro lógico

### LPP LPP Cobro lógico

Función: Cobro lógico

La operación **Cobro lógico** traslada fuera el valor a la cumbre del stack.
El segundo valor del stack se convierte en el nuevo valor a la cumbre del stack.
Después de la ejecución de LPP, la profundidad del stack viene decrementata de uno.

Parámetros: **nadie**

_……..__LPS____ **Nuevo Rama en desarrollo….. …, Fin NUEVA Rama,** LPP **_ Reanudación Vieja Rama**
**|** **|**
**| _____Vecchio Rama congelato_____ _____ |**

## Ejemplo:

```
- NET/1
- LD/I_1
- LD/I_1
- A/I_2
- LPS     ; Nueva rama con rescate del viejo
- A/I_3
- PEX/-DIS/Valore Rama Nuevo a 1
- NOT
- PEX/-DIS/Valore Rama Nuevo a 0
- LPP     ; Abandono nueva rama y reanudación del viejo
- A/I_2
-……
```

## 14.15 LRD: PLC Copia lógica

### LRD PLC Copia lógica

Función: Copia lógica

La operación **Copia lógico** cargo el segundo valor del stack en la cumbre del mismo.
El valor no viene ni situado ni retirado por el stack pero el valor que estuvo a su cumbre viene sovrascritto de aquel nuevo.

Parámetros: **nadie**

_……__LPS____ **Nuevo Rama en desarrollo..., Fin NUEVA Rama,** LPP **_ Reanudación Vieja Rama**
**| |**
**| _____Vecchio Rama congelato_____ _____|____ Viejo Rama congelado**

## Ejemplo:

- NET/1
- LD/I_1
- LD/I_1
- A/I_2
- LPS ; Nueva rama con rescate del viejo
- A/I_3
- PEX/-DIS/Valore Rama Nuevo a 1
- NOT
- PEX/-DIS/Valore Rama Nuevo a 0
- LRD ; Abandono nueva rama y reanudación del viejo con su continuación de la rama congelada
- A/I_2
-……

## 14.16 ALD: PLC Combinado lo primero y el segundo elemento por AND

### ALD PLC Combinado lo primero y el segundo elemento por AND

Función: Combina lo primero y el segundo elemento por AND

La operación **Combina primera y según nivel por AND** combina los valores de lo primero y el segundo nivel del stack usando una operación lógico combinatoria AND.
El resultado es cargado en la cumbre del stack.
Después de la ejecución de ALD, la profundidad del stack viene decrementata de uno.

Parámetros: **nadie**

```
_.........__LPS____ Nuevo Rama en desarrollo... _________ALD, Continua Nuevo en AND con el Viejo,
          |                                                |
          | ______Vecchio Rama congelato____________   _ |
```

## Ejemplo:

```
a         -
          - SDO/M1.1,1; a = (M1.1 And M1.2)
          - SDO/M1.2,1
          - SDO/M1.3,1; b = (M1.3 Or M1.4)
          - SDO/M1.4,0
          - SDO/M1.5,1; r = M1.5 And a And b

          - NET/1
          - LD/M1.1; a = (M1.1 And M1.2)
          - A/M1.2
          - LPS
          - LD/M1. 1; b = (M1.3 Or M1.4))
          - O/M1.4
          - LPS
          - LD/M1.5; r = M1.5 And a And b
          - ALD
          - ALD

          - PEX/-DIS/Stack valor 1
          - NOT
          - PEX/-DIS/Stack valor 0
          - JMP/a
```

El valor visualizado por el DIS es igual a **Stack valor 1**

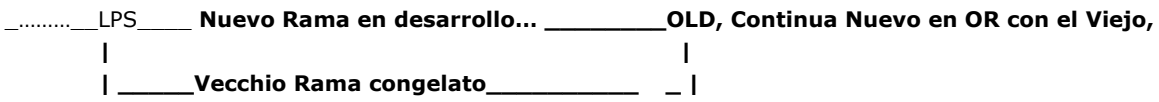
### 14.17 OLD: PLC Combinado lo primero y el segundo elemento por OR

**OLD PLC Combinado lo primero y el segundo elemento por OR**

Función: Combina lo primero y el segundo elemento por OR

La operación Combina primera y según nivel por OR ejecuta una combinación lógico OR de los valores de bit en lo primero (cumbre) y según nivel del stack. allí resultado es cargado en la cumbre del stack.
Después de la ejecución de OLD, la profundidad del stack se reduce de una.

Parámetros: **nadie**




## Ejemplo:

```
a       -
        - SDO/M1.1,1; a = (M1.1 And M1.2)
        - SDO/M1.2,0
        - SDO/M1.3,1; b = (M1.3 And M1.4)
        - SDO/M1.4,0
        - SDO/M1.5,0; r = M1.5 Or a Or b


        - NET/1
        - LD/M1.1; a = (M1.1 And M1.2)
        - A/M1.2
        - LPS
        - LD/M1.3; b = (M1.3 And M1.4)
        - A/M1.4
        - LPS
        - LD/M1.5; r = M1.5 Or a Or b
        - OLD
        - OLD

        - PEX/-DIS/Stack valor 1
        - NOT
        - PEX/-DIS/Stack valor 0
        - JMP/a
```

El valor visualizado por el DIS es igual a **Stack valor 0**

## 14.18 NOT: PLC Modificación del valor superior del stack

**NOT PLC Modificación del valor superior del stack**

Función: Modificación del valor superior del stack

Parámetros: **nadie**

## Ejemplo:

- NET/1
- LD/I_1
- NOT

## 14.19 LEQ: PLC Confrontado dos valores si igual cargo con el stack 1 si no 0

### LEQ PLC Confrontado dos valores si igual cargo con el stack 1 si no 0

Función: PLC Confrontado dos valores si igual cargo con el stack 1 si no 0.

La operación **Confronto de números reales es** utilizada para confrontar dos valores: n1 y n2. Usted podrá ejecutar una comparación n1 = n2

En AWL, si la comparación es verdadera, esta operación respectivamente cargan un 1 en el valor superior del stack lógico

Las comparaciones de números reales están con señal.

**Advertencia:** se podrá crear una comparación **<>,** <o> utilizando la operación Not con la operación de comparación =. La secuencia aquí indicada es equivalente a una comparación <> de L1 con 50:

- LEQ L1, 50.0
- NOT

Parámetros: **a,b**
dónde:

**a** = variable o vocablo de sistema inherente la prueba

**b** = valores de confrontar

**Ejemplo:**

- NET/1
- LEQ/L1,6

## 14.20 LGE: PLC Confrontado dos valores si mayor igual cargo con el stack 1 si no 0

### LEQ PLC Confrontado dos valores si mayor igual cargo con el stack 1 si no 0

Función: PLC Confrontado dos valores si igual o mayor cargo con el stack 1 si no 0.

La operación **Confronto de números reales es** utilizada para confrontar dos valores: n1 y n2. Usted podrá ejecutar una comparación n1 >= n2

En AWL, si la comparación es verdadera, esta operación respectivamente cargan un 1 en el valor superior del stack lógico

Las comparaciones de números reales están con señal.

**Advertencia:** se podrá crear una comparación **<,** utilizando la operación Not con la operación de comparación> =. La secuencia aquí indicada es equivalente a una comparación <de L1 con 50:

- LGE L1, 50.0
- NOT

Parámetros: **a,b**
dónde:

**a** = variable o vocablo de sistema inherente la prueba

**b** = valores de confrontar

**Ejemplo:**

- NET/1
- LGE/L1,6

## 14.21 LLE: PLC Confrontado dos valores si menor igual cargo con el stack 1 si no 0

## LEQ PLC Confrontado dos valores si menor igual cargo con el stack 1 si no 0

Función: PLC Confrontado dos valores si igual o menor cargo con el stack 1 si no 0.

La operación **Confronto de números reales es** utilizada para confrontar dos valores: n1 y n2. Usted podrá ejecutar una comparación n1 <= n2

En AWL, si la comparación es verdadera, esta operación respectivamente cargan un 1 en el valor superior del stack lógico

Las comparaciones de números reales están con señal.

**Advertencia:** se podrá crear una comparación**>,** utilizando la operación Not con la operación de comparación <=. La secuencia aquí indicada es equivalente a una comparación> de L1 con 50:

- LLE / L1, 50.0
- NOT

Parámetros: **a,b**
dónde:

**a** = variable o vocablo de sistema inherente la prueba

**b** = valores de confrontar

**Ejemplo:**

```
- NET/1
- LLE/L1,6
```

## 14.22 AEQ: PLC Confrontado dos valores si iguales hace el AND de 1 con el stack

### AEQ PLC Confrontado dos valores si iguales hace el AND de 1 con el stack

Función: PLC Confrontado dos valores si iguales hace el AND de 1 con el stack

La operación **Confronto de números reales es** utilizada para confrontar dos valores: n1 y n2. Usted podrá ejecutar una comparación n1 = n2

En AWL si la comparación es verdadera combina por AND un 1 con el valor superior del stack lógico.

Las comparaciones de números reales están con señal.

Parámetros: **a,b**
dónde:

**a** = variable o vocablo de sistema inherente la prueba

**b** = valores de confrontar

**Ejemplo:**

```
- NET/1
- AEQ/L1,6.
```

## 14.23 AGE: PLC Confrontado dos valores si iguales o mayor hace el AND de 1 con el stack

### AEQ PLC Confrontado dos valores si iguales o mayor hace el AND de 1 con el stack

Función: PLC Confrontado dos valores si iguales o mayor hace el AND de 1 con el stack

La operación **Confronto de números reales es** utilizada para confrontar dos valores: n1 y n2. Usted podrá ejecutar una comparación n1> = n2

En AWL si la comparación es verdadera combina por AND un 1 con el valor superior del stack lógico.

Las comparaciones de números reales están con señal.

Parámetros: **a,b**
dónde:

**a** = variable o vocablo de sistema inherente la prueba

**b** = valores de confrontar

**Ejemplo:**

```
- NET/1
- AGE/L1,6.
```

## 14.24 ALE: PLC Confrontado dos valores si iguales o menor hace el AND de 1 con el stack

### AEQ PLC Confrontado dos valores si iguales o menor hace el AND de 1 con el stack

Función: PLC Confrontado dos valores si iguales o menor hace el AND de 1 con el stack

La operación **Confronto de números reales es** utilizada para confrontar dos valores: n1 y n2. Usted podrá ejecutar una comparación n1 <= n2

En AWL si la comparación es verdadera combina por AND un 1 con el valor superior del stack lógico.

Las comparaciones de números reales están con señal.

Parámetros: **a,b**
dónde:

**a** = variable o vocablo de sistema inherente la prueba

**b** = valores de confrontar

**Ejemplo:**

```
- NET/1
- ALE/L1,6.
```

## 14.25 OEQ: PLC Confrontado dos valores si iguales hace el OR de 1 con el stack

### AEQ PLC Confrontado dos valores si iguales hace el OR de 1 con el stack

Función: PLC Confrontado dos valores si iguales hace el OR de 1 con el stack

La operación **Confronto de números reales es** utilizada para confrontar dos valores: n1 y n2. Usted podrá ejecutar una comparación n1 = n2

En AWL si la comparación es verdadera combina por OR un 1 con el valor superior del stack lógico.

Las comparaciones de números reales están con señal.

Parámetros: **a,b**
dónde:

**a** = variable o vocablo de sistema inherente la prueba

**b** = valores de confrontar

**Ejemplo:**

```
- NET/1
- OEQ/L1,6.
```

## 14.26 OGE: PLC Confrontado dos valores si iguales o mayor hace el OR de 1 con el stack

### AEQ PLC Confrontado dos valores si iguales o mayor hace el OR de 1 con el stack

Función: PLC Confrontado dos valores si iguales o mayor hace l OR de 1 con el stack

La operación **Confronto de números reales es** utilizada para confrontar dos valores: n1 y n2. Usted podrá ejecutar una comparación n1> = n2

En AWL si la comparación es verdadera combina por OR un 1 con el valor superior del stack lógico.

Las comparaciones de números reales están con señal.

Parámetros: **a,b**
dónde:

**a** = variable o vocablo de sistema inherente la prueba

**b** = valores de confrontar

**Ejemplo:**

```
- NET/1
- OGE/L1,6.
```

## 14.27 OLE: PLC Confrontado dos valores si iguales o menor hace el OR de 1 con el stack

### AEQ PLC Confrontado dos valores si iguales o menor hace el OR de 1 con el stack

Función: PLC Confrontado dos valores si iguales o menor hace el OR de 1 con el stack

La operación **Confronto de números reales es** utilizada para confrontar dos valores: n1 y n2. Usted podrá ejecutar una comparación n1 <= n2

En AWL si la comparación es verdadera combina por OR un 1 con el valor superior del stack lógico.

Las comparaciones de números reales están con señal.

Parámetros: **a,b**
dónde:

**a** = variable o vocablo de sistema inherente la prueba

**b** = valores de confrontar

**Ejemplo:**

```
- NET/1
- OLE/L1,6.
```

## 14.28 PEX: PLC Ejecutado sobre 1 en stack la instrucción AXESBRAIN

**PEX PLC Ejecutado sobre 1 en stack la instrucción AXESBRAIN**

Función: Ejecuta sobre 1 en stack la instrucción AXESBRAIN

Parámetros: **a**

dónde:

**a** = instrucción AXESBRAIN

### Ejemplo:

```
- NET/1
- LD/I_12
- PEX/-DIS/1: prueba
```

## 14.29 TON: PLC Temporizador sin retención

## 14.30 TOR: PLC Temporizador con retención

**TON TOR PLC Temporizador sin retención y Temporizador con retención**

Función: PLC Temporizador sin retención y Temporizador con retención
Encamina como temporizzazione retardo a la inserción, con memoria,

Si habilitáis, las operaciones Encamina como temporizzazione retardo a la inserción y Encamina como temporizzazione retardo a la inserción con memoria miden el tiempo hasta al valor máximo. Cuando el valor corriente, Txxx, es >= al tiempo de estándard, PIADOSO, el bit de temporizzazione es activado.

Si desactivaran, el temporizador como retardo a la inserción él resetta, mientras que el temporizador como retardo sencillamente a la inserción con memoria se para. Ambos los temporizadores se paran al logro del valor máximo.
Operandos: Txxx: TON TOR

Los temporizadores también son utilizados para implementar las funciones de cuenta mandadas a tiempo. El sistema provee dos muchas operaciones de temporizzazione: encamina como temporizzazione retardo a la inserción (TON) y encamina como temporizzazione retardo a la inserción con memoria, TOR. Los temporizadores difieren en el modo en que reaccionan al estado de la entrada de capacitación. Sea TON que TOR cuentan hacia adelante mientras on es activo, la entrada de capacitación:
ambos no cuentan hacia adelante mientras es desactivado, 0ff, la entrada de capacitación, pero mientras el temporizador TON él resetta automáticamente, el temporizador TOR no él resetta y mantiene su último valor. Pues TON se utilizará de la mejor manera por el temporizzazione de un individual intervalo. TOR será más apropiado para acumular un número de intervalos de tiempo.

Los temporizadores tienen las siguientes características.

- Los temporizadores son controlados con una individual entrada de capacitación, y tienen un valor corriente que conserva el tiempo transcurrido de cuando el temporizador ha sido habilitado, LOS temporizadores también tienen un valor de estándard (PT), que es

confrontado con el valor corriente cada vez que este último es puesto al día, y si es ejecutada la operación de temporizzazione.

- Un bit de temporizzazione es programado o resettato con base en el resultado de la comparación del valor corriente con el valor de estándard.

- Si el valor corriente es mayor o igual al valor de estándard, el bit de temporizzazione, bit T, es activado.

**Advertencia**
Algunos valores corrientes de temporizzazione pueden ser a retención, LOS bites de temporizzazione no son a retención, y sólo son programados como resultado de una comparación entre valor corriente y valor de estándard.

Si él resetta un temporizador, es borrado su valor corriente y desactivado el bit T. Cualquier temporizador puede ser resettato con la operación Resetta pero un temporizador TOR únicamente puede ser resettato de esta operación. El bit de temporizzazione de un temporizador no viene resettato escribiendo un cero como valor corriente del temporizador. El mismo vale al reverso: escribiendo un cero no en el bit T del temporizzazione él resetta su valor corriente.

Puesta al día de temporizadores con resolución a 1 ms

La UCP provee temporizadores que se ponen al día una vez a milisegundo, temporizadores de los ms, a través de la rutina de sistema que memoriza la base de tiempo de sistema. Se trata día temporizadores que garantizan el exacto control de las operaciones.

La puesta al día es automática en cuanto el valor corriente de un temporizador activo de los ms es puesto al día nota por una rutina de sistema. Una vez habilitada, la ejecución de una operación TON/TOR que controla un temporizador de los ms sólo es solicitada para controlar el estado attivato/disattivato del temporizador.

Ya que el valor corriente y el bit T de los temporizadores de los ms es puesto al día por una rutina de sistema, independiente del ciclo del revisor programable y el programa usuario, estos valores pueden ser puestos al día en cualquier punto del ciclo y eso awerrà más de una vez por ciclo si el tiempo de ciclo supera un milisegundo. No es garantizado por tanto que tales valores queden constantes por toda una determinada ejecución del programa usuario principal.
La reposición de un temporizador de 1 ms habilitados lo desactiva, resetta a cero su valor corriente y borra su bit.

Parámetros: **a,b**

dónde:

**a** = número temporizador, máximo 32,
**b** = valor en milisegundos

**Ejemplo:**

```
- NET/6
- LD / Q_InMac.15
- TON/1,3000

- NET/7
- LD / T1
- EQU/Q_InMac.16
```

**Ejemplo:**

```
a       - NET/1
        - LDN/T1
        - TON/2,1000

        - NET/2
        - LD/T2
        - TON/1,1000

        - NET/3
        - LD/T2
        - PEX/-DIS/pippo
        - NOT
        - PEX/-DIS /

        - JMP/a
```

## 14.31 CTU: PLC Counter Up

## 14.32 CUD: PLC Counter Up y Down

**CTU CUD PLC Counter Up e Counter Up y Down**

Función: PLC Contador hacia adelante y Contador adelante y atrás

Detalles de las operaciones de cuenta

La operación Cuenta hacia adelante (CTU) cuenta hacia adelante a partir del valor corriente del contador cada vez que la entrada de cuenta hacia adelante da de off a on. allí contador viene resettato cuando se activa la entrada de reposición o cuando es ejecutada la operación Reposición. Ello se para al logro del valor máximo (32.767).

La operación Cuenta en avanti/indietro (CUD) cuenta hacia adelante cada vez que la entrada de cuenta hacia adelante da atrás atrás de off a on y a cuenta al cada vez que la entrada de cuenta al da de off a on. allí contador viene resettato cuando se activa la entrada de reposición o cuando es ejecutada la operación Reposición. Al logro del valor máximo (32.767), hacia adelante la frente de subida siguiente de la entrada de cuenta hará sí que el valor corriente se recoja alrededor del valor mínimo, -32.768. Análogamente, al logro del valor mínimo (-32.768) atrás la siguiente frente de subida de la entrada de cuenta al hará sí que el valor corriente se recoja alrededor del valor máximo, 32.767.

Si el usuario resetta un contador con la operación Resetta, sea el bit de cuenta sea el valor corriente de cuenta serán resettati.

Los contadores hacia adelante y en avanti/all'indietro tienen un valor corriente que memoriza la cuenta corriente. Ellos también tienen un valor de estándard (PV) que es confrontado con el valor corriente cada vez que es ejecutada la operación de cuenta. Si el valor corriente es mayor o igual al valor de estándard, se activa el bit de cuenta, bit C. De otro modo el bit C se desactiva.

Se utilizas el número del contador para hacer referencia sea al valor corriente que al bit C del contador mismo.

**Advertencia**
Sólo hay un valor corriente por cada contador. No él cheques por tanto el mismo numero a más que un contador, los contadores hacia adelante y en avanti/indietro acceden al mismo valor corriente.

Defines modo por contador veloz, Activo contador veloz

Parámetros: **a,b**

dónde:

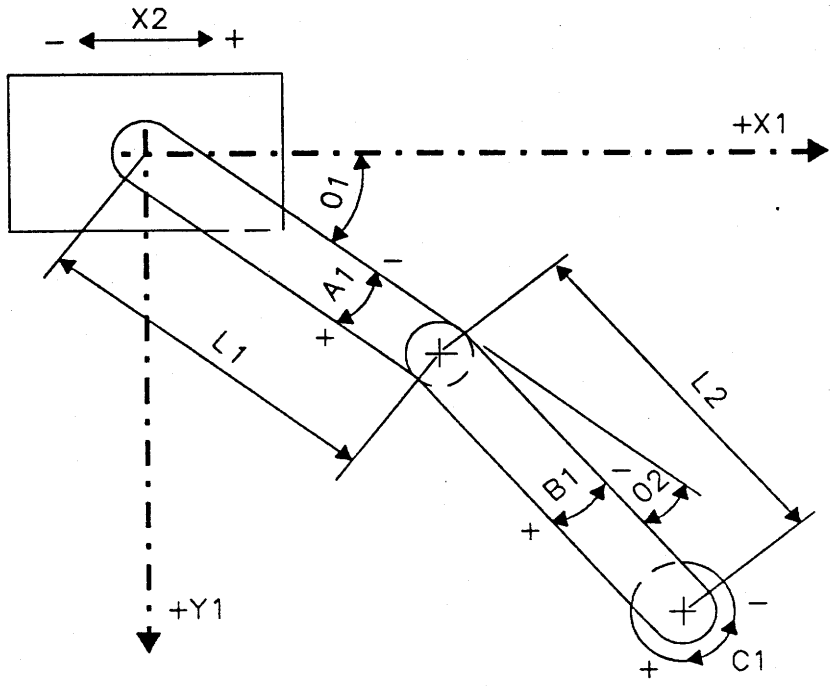
**a** = número contador, máximo 32,
**b** = valor final

# APÉNDICE A

## A. Ases Virtuales

*Los ases virtuales, como dice la palabra misma, no son ases físicos pero ases que por comodidad usuario son caracterizados sobre sistemas a geometría SCARA o CILÍNDRICA de modo que además de efectuar movimientos sobre los ases físicos se pueda efectuar también movimientos según un sistema cartesiano.*
*Con tal sistema, además se pueden efectuar interpolaciones lineales y circulares.*

SCARA arregla




**Número de ases SCARA:** indica el número de ases que tomar en consideración; mínimo 2, los primeros 2,; tres, 1,2,3 o bien 4 (todo)

En el ejemplo indicado, son tomados en consideración todo y 4, las 2 articulaciones A1 y B1,il 3° eje rotatorio C1 puesto a la extremidad de la segunda articulación y al 4° eje traslante X2,

**Nombre del primer eje:** indica el primero eje real del sistema, 1° articulación,; en el ejemplo el eje A1

**Largo del primer eje:** indica el largo del primer eje (distancia entre lo primero y según articulación); en el ejemplo L1

**Nombre del segundo eje:** indica el segundo eje real del sistema, 2° articulación,; en el ejemplo B1

**Largo del segundo eje:** indica el largo del segundo eje, distancia entre articulación central y eje pinza,; en el ejemplo L2.
En el caso esté presente el tercer eje, en el ejemplo C1, el eje pinza tiene que ser concéntrico con el eje de rotación del tercer eje, en este caso el largo del segundo eje está fijo. Si en cambio hay excentricidad, el largo del segundo eje, con respecto del eje pinza, es variable, siendo dependiente de la posición angular de la pinza. Por consiguiente, ya que se impuesta un solo largo, el sistema cartesiano conseguido sólo será válido en aquellas condiciones.

Con base en estas consideraciones, en caso de excentricidad, hace falta cuanto tener presente sigue:

- no es posible conseguir trayectos rectilíneos y circulares y el error de trayecto será tanto mayor como mayor es la excentricidad.

**Nombre del tercer eje:** indica el eventual eje rotatorio sitio a la extremidad del segundo eje; en el ejemplo C1.
La indicación o menos que este eje en tablero, influyen sólo en la geometría del sistema a los efectos de la compensación automática de dicho rotatorio y precisamente:

- Eje no indicado en tablero: en caso de requerido movimiento ases, reales o virtuales, tal eje rotatorio es ignorado, naturalmente si no solicitara en el movimiento,

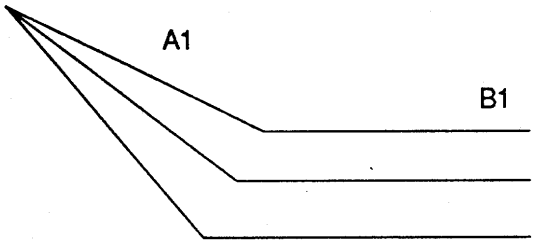
- Eje indicado en tablero: en caso de requerido movimiento ases, reales o virtuales, tal eje rotatorio, naturalmente si no solicitara en el movimiento, viene automáticamente aglomerado; la pinza apuesta sobre tal eje queda por lo tanto siempre paralela a él mismo.
  Y alguno que si este eje es solicitado en el movimiento, la indicación o menos que el eje en el tablero no tiene ningún sentido más siendo el movimiento requerido determinante.

**Nombre del cuarto eje:** indica el eventual eje traslante, traslado de todo Yo SCARA,; en el ejemplo X2

**Compensación mecánica:** este fecho sirve para localizar el tipo de robot según la convención siguiente:

0 = robot de tipo SCARA tradicional
± 1 = robot de tipo SCARA pero con sistema a pantógrafo

dónde sólo moviendo uno eje, en el ejemplo de abajo A1, el segundo eje, en el ejemplo B1, se mantiene paralelo a él mismo.




La señal algebraica debe ser entendida en el modo siguiente:
Con la señal positiva se entiende la compensación automática sobre el eje B1 sumando el rincón efectuado por el eje A1.

Con la señal negativa se entiende la compensación automática sobre el eje B1 sustrayendo el rincón efectuado por el eje A1.

**Compensación absoluta:** este fecho es utilizado cada vez se desea conseguir un cero eje práctico diferente de aquel teórico.

**Máxima velocidad:** corresponde a la máxima velocidad querida sobre el eje virtual; si esta velocidad fuera imposible que efectuar, hace falta tener presente que los dos ases reales juntados al eje virtual tienen cada uno una máxima velocidad, por estándard es utilizada el principio posible.

**Máxima aceleración y máxima desaceleración:** corresponden a la máxima aceleración y desaceleración querida sobre el eje virtual; también por este fechas si los valores indicados fueran superiores a lo posible, por estándard son utilizados los máximos posible.

**Notas:**

a, Ya que todos los cálculos para los ases virtuales son ejecutados según la geometría clásica obrando sobre senos y cosenos de rincones expresos en grados, hace falta tener presente cuanto sigue:

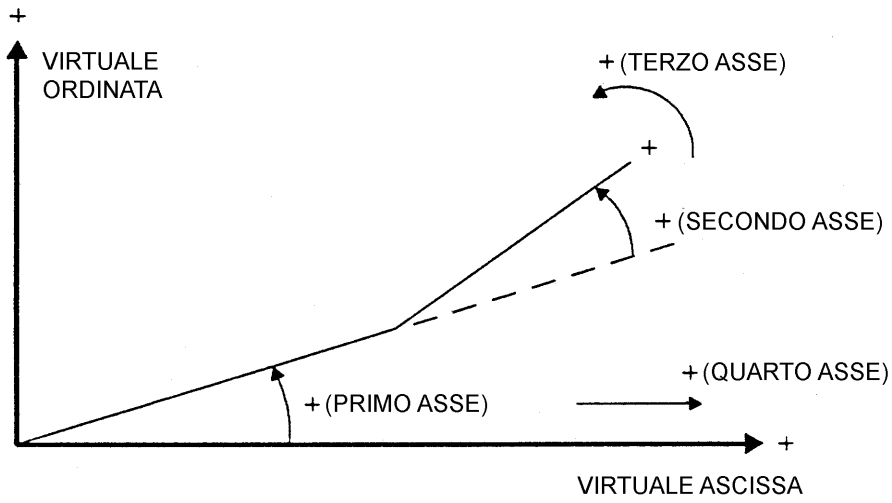
1 - La unidad de medida de los primeros 3 ases reales tiene que ser el grado.

2 - La unidad de medida del cuarto eje, puede ser el mm o pulgares a segunda que a los largo primero y según eje sean dadas respectivamente en mm o inc.

3 - La compensación absoluta del primer eje real tiene que ser dado, si está presente el 4° eje traslante, de modo que el eje virtual abscisa sea paralelo al cuarto eje SCARA.

4 - La compensación absoluta del segundo eje real tiene que ser dado de modo que los primeros dos ases resulten alineados, brazo todo tenso.

5 - Las direcciones de movimiento, positivo, rechazo, de los ases del sistema tienen que ser todo concordes: por los primeros 3 ases rotatoria dirección positiva significa ir de abscisa positiva a ordenada positiva; el eventual cuarto eje tiene que tener la misma dirección de la abscisa. La figura siguiente provee un ejemplo de dichas direcciones.

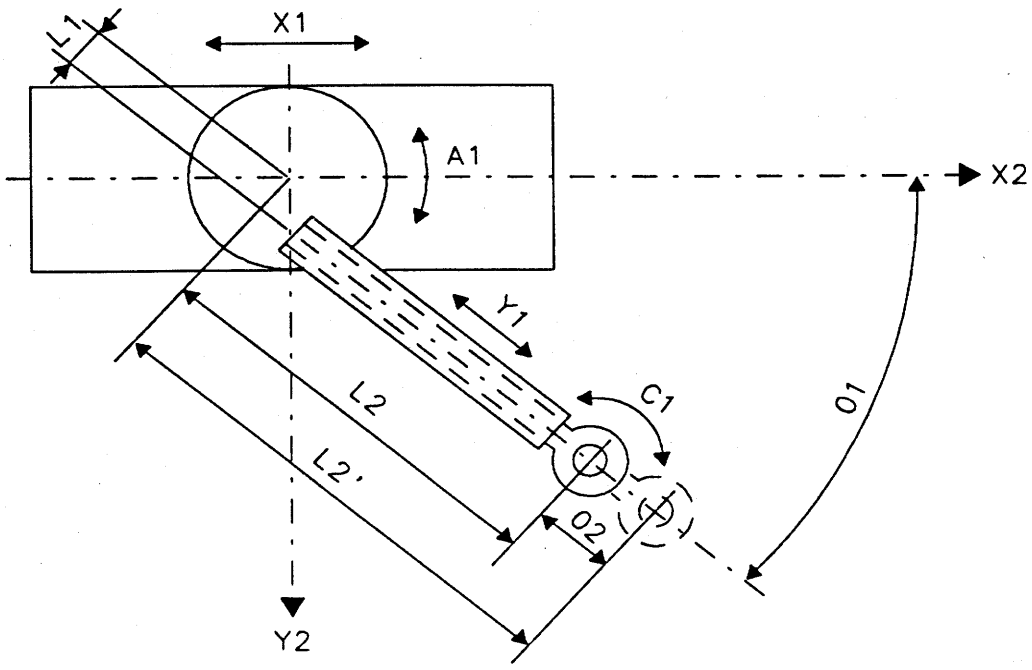



6 - Con base en las consideraciones vistas en los puntos 1-5 precedentes, se tendrá que después de un movimiento a cero de todo y a 4 los ases reales SCARA, el eje virtual abscisa se encontrará a una coordenada equivalente a la suma de los largo de lo primero y según eje; el eje virtual ordenado a coordenada cero.

b, Con un sistema SCARA, no es posible solicitar por una misma instrucción de movimiento, ases virtuales juntamente a los primeros dos ases reales del sistema, uno de los dos o ambos.

Y viceversa posible solicitar juntamente ases virtuales al 3° e/o 4° eje real del sistema.

Sistema CILINDRICO




**Número de ases del sistema CILÍNDRICO:** indica el número de ases que tomar en consideración; 1°+2°,1°+2°+ 3°,1°+2°+3°+ 4°. Se aconseja siempre indicar 4 y si algún eje es no presiente (2°) 3°, 4°, no lo indiques en las rayas adecuadas.
En el ejemplo indicado, son tomados en consideración todo y 4 los ases.

**Nombre del primer eje:** indica el primero eje real del sistema; en el ejemplo A1. El primer eje siempre es el eje rotatorio general y tiene que siempre estar presente

**Largo del primer eje:** indica la distancia, también puede ser 0, entre el eje de rotación del primer eje y el eje pinza; tal distancia debe ser dada perpendicularmente al eje cannotto, si cannotto faltante, al brazo lleva pinza,; en el ejemplo L1.

**Nombre del segundo eje:** indica el segundo eje real del sistema; en el ejemplo Y1.
Tal eje siempre es el eje cannotto y podría ser no presiente también; en tal caso tiene que pero necesariamente estar presente el cuarto eje

**Largo del segundo eje:** indica la distancia entre el eje de rotación del primer eje y el eje pinza; tal distancia debe ser dada paralelamente al eje cannotto, si cannotto faltante, al brazo lleva pinza,; en el ejemplo, L2 si el segundo eje, cannotto, no tiene compensación absoluta; 12, si este eje tiene un compensaciones suyo, 02.
En el caso esté presente el tercer eje, en el ejemplo C1, el eje pinza tiene que ser concéntrico con el eje de rotación del tercer eje, en este caso el largo del segundo eje está fijo. Si en cambio hay excentricidad, el largo del segundo eje, con respecto del eje pinza, es variable, siendo independiente de la posición angular de la pinza. Por consiguiente, ya que se impuesta un solo largo, el sistema cartesiano conseguido sólo será válido en aquellas condiciones.
Con base en estas consideraciones, en caso de excentricidad, hace falta cuanto tener presente sigue:

- no es posible conseguir trayectos rectilíneos y circulares y el error de trayecto será tanto mayor como mayor es la excentricidad.

**Nombre del tercer eje:** indica el eventual eje rotatorio sitio a la extremidad del cannotto, o brazo lleva pinza si cannotto faltante,; en el ejemplo C1.
La indicación o menos que este eje en tablero, influyen sólo en la geometría del sistema a los efectos de la compensación automática de dicho rotatorio y precisamente:

- Eje no indicado en tablero: en caso de requerido movimiento ases, reales o virtuales, tal eje rotatorio es ignorado, naturalmente si no solicitara en el movimiento,

- Eje indicado en tablero: en caso de requerido movimiento ases, reales o virtuales, tal eje rotatorio, naturalmente si no solicitara en el movimiento, viene automáticamente aglomerado; la pinza apuesta sobre tal eje queda por lo tanto siempre paralela a él mismo.
  Y alguno que si este eje es solicitado en el movimiento, la indicación o menos que el eje en el tablero no tiene ningún sentido más siendo el movimiento requerido determinante.

**Nombre del cuarto eje:** indica el eventual eje traslante, traslado de todo lo CILÍNDRICO,; en el ejemplo X1. Tal eje podría ser no presiente también conque esté presente el segundo eje;

**Compensación mecánica:** tal dato siempre va dato con valor 0.

**Compensación absoluta:** este fecho es utilizado cada vez que cada se desea conseguir un cero eje práctico diferente de aquel teórico.

**Máxima velocidad:** corresponde a la máxima velocidad querida sobre el eje virtual; si esta velocidad fuera imposible que efectuar, (cada eje real miembro al sistema tiene un mismo principio velocidad, por estándard es utilizada el principio posible.

**Máxima aceleración y desaceleración:** corresponden a la máxima aceleración y desaceleración querida sobre el eje virtual; también por este fechas si los valores indicados fueran superiores a lo posible, por estándard son utilizados los máximos posible.

**Notas:**

a, Ya que todos los cálculos para los ases virtuales son ejecutados según la geometría clásica, obrando sobre senos y cosenos de rincones expresos en grados, hace falta tener presente cuanto sigue:

1 - La unidad de medida de los ases rotatorios, primero y tercer eje, reales tiene que ser el grado.

2 - La unidad de medida del cuarto eje real puede ser el mm o el inc. a segunda que el largo según eje sea dada respectivamente en mm o inc.

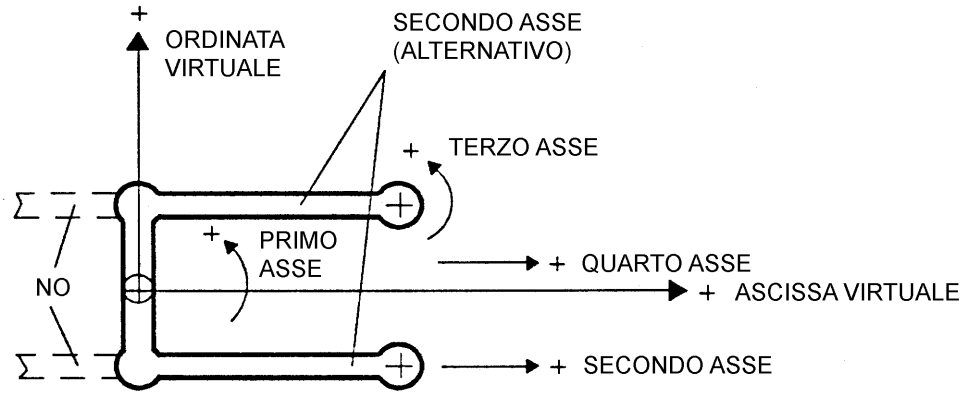
3 - La compensación absoluta del primer eje real tiene que ser dado, si está presente el 4 eje traslante, de modo que el eje virtual abscisa sea paralelo al cuarto eje CILÍNDRICO.

4 - ll según eje, cannotto, puede o menos tener compensación; tal compensación implica claramente un diferente largo del segundo eje.

5 - Las direcciones de movimiento, positivo, rechazos, de los ases del sistema tienen que ser todo concordes: para los ases rotatorios, primero y tercer eje, dirección positiva significa ir de abscisa positiva a ordenada positiva; los ases lineales, según y cuarto, tiene que tener la misma dirección de las abscisas.

6 - ll según eje, cannotto, no puede encontrarse en abscisa negativa en caso contrario girar el primer eje de modo que se encuentra en abscisa positiva.

La figura siguiente provee como un ejemplo de tienen que ser puestos los ases sobre de un sistema cilíndrico



7 - Con base en las consideraciones vistas en los puntos 1-6 precedentes, se tendrá que después de un movimiento a cero de todo y a 4 los ases reales CILÍNDRICOS, el eje virtual abscisa se encontrará a una coordenada igual al largo del segundo brazo y el eje virtual ordenados a una coordenada igual al largo del primer brazo. , Por este último eje la coordenada será positiva o rechazo,

b, Con un sistema CILÍNDRICO, no es posible solicitar por una misma instrucción de movimiento, ases virtuales juntamente a los dos ases reales, uno de los dos o ambos, constituyentes el sistema cilíndrico y precisamente: 1° eje del sistema, Rotatorio general, 2° eje del sistema, cannotto o 4° eje del sistema (traslante) en el caso de ausencia del eje cannotto

Y en cambio posible solicitar juntamente ases virtuales al 3° e/o 4° eje, si está presente el 2° eje,
Recapitulación de los ases reales no abbinabili a los virtuales en una misma instrucción de movimiento:

- Sistema cilíndrico compuesto por el 1° y 2° eje, rotatorio general + cannotto, no es admitido virtuales +1° e/o 2° eje real

- Sistema cilíndrico compuesto por el 1° y 4° eje, rotatorio general + traslante, no es admitido virtuales + 1° e/o 4° eje real

- Sistema cilíndrico traje de 1°, 2° y 4° eje
-

No son admitidos virtuales + 1° e/o 2° eje, solo o con el 4°.

# APÉNDICE B

## B. Controlo eje con octavilla

*El posicionamiento de un eje en manual puede ser juntado a un aparato llamada octavilla que es visto como por el sistema un eje de sola lectura.*

El valor de posición leído por la octavilla modifica la posición del eje juntado, es tan posible dar incrementos micrométricos all eje mismo.

La octavilla es vista cómo un eje de sola lectura, que es enganchado a un eje por oportuno mando que quedará controlado.

En el lenguaje de automación AXESBRAIN, la instrucción "- HMS" permite la inserción y el disinserimento de un eje octavilla "máster" a un eje de posicionamiento "eslavas."

Para administrar el abbinamento eje octavilla con eje de posicionamiento en entorno "DCOM", tiene que ser utilizado el servicio "WriteAxesRegister" por ambos los ases.

# APÉNDICE C

## C. Ases a portal ( Gantry)

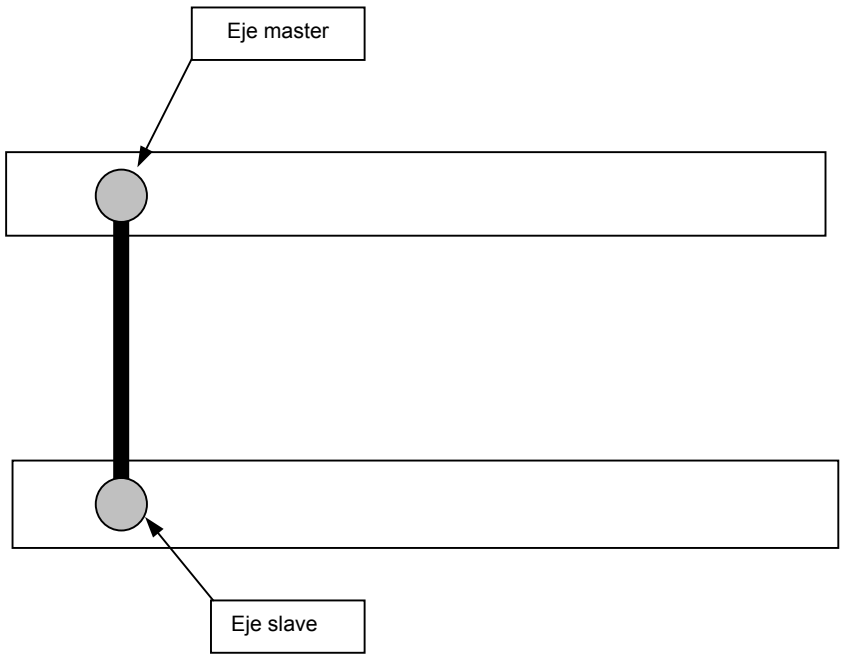
*El eje a portal, Gantry)è una estructura mecánicamente rígida, es normalmente una estructura a portal, y corresponde por tanto a un eje único, pero es tratada por el control como si fuera constituida por una pareja de ases, eje máster y eje eslavas, cada uno con los mismos sistemas de cuenta y el propio accionamiento. Una de las funciones del control es mantener la posición del eje "eslavas" más próximo posible a la del eje "máster."*

Los movimientos requeridos sobre los ases máster-eslavas, son ejecutadas en el modo siguiente:

La solicitud de movimiento puede ser efectuada por todas las instrucciones de movimiento.

En el caso de movimiento relativo al solo eje Máster, el eje eslavas lo sigue o queda respectivamente firme en caso de asociación o disassociazione.

Durante el movimiento, el eje eslavas sigue su Máster a tiempo real



# APÉNDICE D

## D. Dientes de rueda electrónicos

*El diente de rueda electrónico permite de juntar la posición de un grupo de ases a un eje "máster" y un tablero de posiciones múltiples.*

Y tan posible aparentar electrónicamente el comportamiento de los dientes de rueda, reemplazar de ello el funcionamiento mecánico con un sistema análogo formado por un grupo de ases sometido a un eje "máster" que puede ser de sola lectura.

Las leyes del movimiento de los cedentes es definida como un tablero de vectores, que de ello defisse las posiciones con respecto del movimiento del diente de rueda.

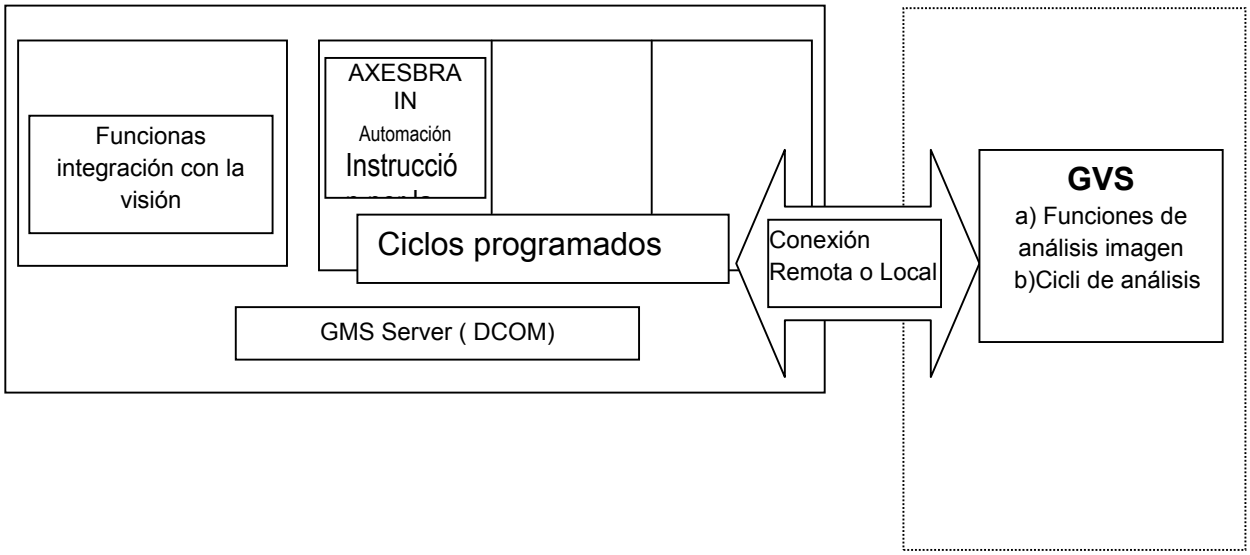
En el lenguaje de automación AXESBRAIN, la instrucción "- HEC" permite la gestión de los dientes de rueda electrónicos.

Para administrar los dientes de rueda eléctricos en entorno "DCOM", tiene que ser utilizado el servicio "WriteAxesRegister" por todos los ases.

# APÉNDICE E

## E. Integración con la visión

*La integración con el entorno de análisis de imagen GVS puede ser obrado en dos modalidades o por de las específicas solicitudes del GMS Server que efectúan llamadas a las funciones base o solicitudes de ejecutar ciclos de análisis, una segunda calle es utilizar las instrucciones de interfacciamento visión del lenguaje AXESBRAIN.*




Esquema integración análisis de imagen

Como se puede notar del esquema la máquina predispuesta por el análisis de imagen puede ser remotata sobre otro PC, naturalmente unida en red local o red internet.

**GVS**

El paquete GVS trabaja por solicitudes de funciones de análisis de la imagen que efectúan operaciones de:

- Filtro y manipulación del imagina
- Análisis luz y extracciones automáticas de umbrales
- Blobs sobre rectángulos de interés
- Extracciones de contornos sobre intersección de figuras sobre rectángulos de interés
- Extracciones de bordos
- Medición de contornos
- Extracciones de entes sobre los contornos
- Medición sobre los entes

Además es previsto un lenguaje GVSL capaz de programar ciclos de elaboración de imágenes, de modo de conseguir de los resultados dirigidos a las situaciones reales.

# APÉNDICE G

## F. Definici de Globales especiales y funzionilit àgenerales

En el expediente de caratterizzazione/configurazione del sistema "sistema.txt", ve manual de caracterización, ha sido prevista **si declaráis** de los Globales especiales que se completan con el entorno de programación y algunas funciones generales.

Las declaraciones se colocan en el expediente "sistema.txt" bajo la sesión "**[ParametriGenerali]"** y soy:

a - GlobaleNumeroAllarmi
b - GlobaleBloccoMovimentazione
c - NumeroLocali
d - NumeriDecimaliDRT
y - NumeriCifreDRT
f - NumeriDecimaliFILE
g - NumeriCifreFILE
h - NumeroIstruzioniMax
los - PartProgramG89
l - FlagFastWaitIO
m - TempoLatenzaErrori
n - WaitSEC
o - TempoScansione
p - SeGlobaleErroreNoDISM1
q - FlagHoldG8183

**GlobaleNumeroAllarmi**

El número indicado si mucho de cero indica el Global en el que el sistema carga el número de alarmas presentes.

Estándard = 0

**GlobaleBloccoMovimentazione**

El número indicado si mucho de cero indica el Global en el que el sistema averigua la inhibición de los movimientos manuales, no programados, en el caso caso que el valor sea diferente de cero 0.

Estándard = 0

**NumeroLocali**

El valor indica el número de variables Locales que vengno definidos por cada part program e inicializáis a 0.

La instrucción "DIM" reorganiza este valor en la sola instancia del part program

Estándard = 32

**NumeriDecimaliDRT**

El valor indica el número de decimales que son representados en la visualización de los campos de la instrucción "DRT"

Estándard = 3

**NumeriCifreDRT**

El valor indica el número de cifras que son representadas en la visualización de los campos de la instrucción "DRT"

Estándard = 7

**NumeriDecimaliFILE**

El valor indica el número de decimales que son representados en la visualización de los campos de la instrucción "FWR"

Estándard = 3

**NumeriCifreFILE**

El valor indica el número de cifras que son representadas en la visualización de los campos de la instrucción "FWR"

Estándard = 7

**NumeroIstruzioniMax**

Para evitar que una secuencia de instrucciones lógicas y control condicionen el tiempo dedicado por la UCP, es indicado el número de instrucciones en el que si el task no es suspendido, el sistema fuerza una pausa de **TempoScansione** milisegundos por el part program en examen.

Estándard = 64

**PartProgramG89**

Nombre del part program que es ejecutado con las modalidades de los ciclos fijos.

Estándard = G89.pp

**FlagFastWaitIO**

Valor que activa la modalidad fast por las instrucciones "WDI" activos.

Estándard = 0

**TempoLatenzaErrori**

Tiempo en segundo que tiene que transcurrir entre dos mensajes con el mismo número de error engendrado por "- SEC/SET_MESSAGE, #", para actualizarlo sobre la zona de vídeo dedicada.

Estándard = 1

**WaitSEC**

Tiempo él suspensión corchete automático que es introducido a cada instrucción de "SEC" y "DIS/-1:"

Estándard = 0.03

**TempoScansione**

El tiempo de barrido por cada part program si no es suspendido por actividad no lógicas o matemáticas

Estándard = 0.010

**SeGlobaleErroreNoDISM1**

Para evitar el sovrascrittura en la parte de señal con la instrucción-DIS/-1: …. si son presentes señales, el **GlobaleNumeroAllarmi** averiguando, el sistema no manda el mensaje.

Estándard = 1

**FlagHoldG8183**

Flag que define**, si** a 1 que a los ciclos fijos de G81,G82 y G83 pueden irle en HOLD

Estándard = 0

# APÉNDICE H

## G. Enumero Errores

| | | |
|---|---|---|
| Error | 1 | Función todavía activa |
| Error | 2 | Nombre asse/mandrino/IO no correcto |
| Error | 3 | OVERFLOW Tabelal nombres ases |
| Error | 4 | OVERFLOW Tablero ases |
| Error | 5 | Tipo eje incongruente |
| Error | 6 | FALTA EJE FÍSICO POR SISTEMA VIRTUAL |
| Error | 7 | EJE HELP NO UTILIZABLE (CYL_TRASLANTE) |
| Error | 8 | OVERFLOW Tablero JOB de movimentazione |
| Error | 9 | JOB de movimentazione DEFINIDO |
| Error | 10 | OVERFLOW ÁREA movimentazione |
| Error | 11 | PARÁMETROS Start continúo ERRADOS |
| Error | 16 | PARÁMETROS Movimentazione ERRADOS |
| Error | 17 | DEMASIADOS TPE CONTEMPORÁNEAS |
| Error | 18 | EJE YA' USADO EN OTROS JOBS |
| Error | 19 | EJE FUERA DE CAMPO DE TRABAJO |
| Error | 20 | RELUZCO FINAL <> RAYO INICIAL |
| Error | 21 | JOB NO DEFINIBLE |
| Error | 22 | DEMASIADOS ASES, MAX 8, |
| Error | 23 | NO HAY MÁS' TEMPORIZADORES |
| Error | 24 | ASES EN MOVIMIENTO |
| Error | 25 | ESTADO INCORRECTO |
| Error | 26 | ERROR DE SISTEMA EN EL EMPLEO DEL JOB |
| Error | 27 | EJE FRENADO |
| Error | 28 | CLOSE U OPEN POSITION LOOP INÚTIL |
| Error | 29 | FALTA ÁREA ESLAVAS |
| Error | 30 | MUESTREO NO ACTIVA |
| Error | 31 | DEMASIADOS PARÁMETROS |
| Error | 32 | FALTA SUMISIÓN ACTIVA |
| Error | 33 | DEMASIADAS LÁMINAS I/O CONFIGURADAS |
| Error | 34 | LÁMINA I/O EQUIVOCADO |
| Error | 35 | BIT NO SOBRE LÁMINA |
| Error | 36 | OVERFLOW DEL BIT LÓGICO |
| Error | 37 | BIT I/O EQUIVOCADO |
| Error | 38 | DEMASIADAS LÁMINAS ANALÓGICAS CONFIGURÁIS |
| Error | 39 | LÁMINA ANALÓGICA EQUIVOCADA |
| Error | 40 | CANAL ANALÓGICO NO SOBRE LÁMINA |

| | | |
|---|---|---|
| Error | 41 | OVERFLOW DEL CANAL ANALÓGICO, LÓGICO, |
| Error | 42 | CANAL ANALÓGICO EQUIVOCADO |
| Error | 43 | SELECTA DESCONOCIDA |
| Error | 44 | TIEMPO CADUCADO (Time OUT) |
| Error | 45 | INTERPOLACIÓN SPLINE ERRADO |
| Error | 47 | PUESTA A CERO EJE HECHO, MANCA HOMING, |
| Error | 48 | OVERFLOW TABLE SPINDLE |
| Error | 49 | MANDRIL EN ACTIVIDAD' SAMPLE |
| Error | 50 | MANDRIL EN ROTACIÓN |
| Error | 51 | SUMISIÓN INHABILITADA |
| Error | 52 | COTIZA CORRECCIÓN ERRADA, IGUAL, |
| Error | 53 | OVERFLOW ÁREA CORRECCIÓN |
| Error | 54 | BUFFER CIRCULAR NO ACTIVO |
| Error | 55 | OVERFLOW COUNTER EN EL EJE STEPPER |
| Error | 56 | COUNTER NO DEFINIDO EN EL EJE STEPPER |
| Error | 58 | LE ANIMO EN DEADLOCK |
| Error | 59 | TASK EN ABORT (TKM) |
| Error | 60 | OVERFLOW TKM |
| Error | 61 | EJE SOBRE MICRO DE MÁS ALLÁ DE CARRERA POSITIVA |
| Error | 62 | EJE SOBRE MICRO DE MÁS ALLÁ DE CARRERA NEGATIVA |
| Error | 63 | FASE ACABADA |
| Error | 1001 | HANDLE INCORRECTO |
| Error | 1002 | DRIVER OCUPADO |
| Error | 1003 | INICIALIZACIÓN ERRADA |
| Error | 1004 | ANOMALÍA EN FASE DE PUESTA A CERO |
| Error | 1006 | DEMASIADOS HANDLER DE MOVIMENTAZIONE ACTIVA |

| | | |
|---|---|---|
| Error Driver Fault | 0x5 | Configuration error - Decision Table |
| | 0x7 | Calibration error |
| | 0x10E | Fiber not connected to digital driver |
| | 0x111 | Digital driver assh't hardware enable |
| | 0x113 | Fiber rx Frame error |
| | 0x114 | Fiber rx Parity error |
| | 0x116 | Cero command refused for movement |
| | 0x117 | Lost cero axis |