
## Motion Control dedicated functions

# Axis Control

## ISaGRAF® Library

**ARTECO MOTION TECH SpA** – Via Gentili, 22 – 48018 FAENZA (RA) Italy
Tel. +39 0546 645777 – Fax +39 0546 645750 – Email: info@arteco-cnc.com - Web: www.arteco-cnc.com

**Even if a great deal of effort has been made to make sure that the information contained herein is exact and correct, Arteco Motion Tech assumes no responsibility for errors or omissions present in this document.**
Any eventual critical evaluation on the part of the user will be in any case welcome and will be taken into consideration in the preparation of future documentation.

# INDEX

**AXIS CONTROL**

# Introduction

This manual describes the function blocks that make up the interface between the ISaGRAF ® application and the axis manager.

**For any information whatsoever regarding the programming of the ISaGRAF® or the insertion of these blocks into an ISaGRAF® application it is necessary to refer to the manual ISaGRAF USER'S GUIDE.**

Before using these blocks and managing the axis found on the data card it is recommended that this entire manual be read thoroughly

This manual does not include the hardware connections necessary for axis management.
Contact the Arteco spa assistance centre.

In Appendix 2 there are the errors communicated by the target (SU data card) to the workbench, not indicated in the **ISaGRAF USER'S GUIDE**, as they were implemented by Arteco.

## Axis Numbering

The numbering of the axes for the su21x family is the following:

***PLC Axis***

0 Axis → Axis found on the PLC. Indicated in the blocks as axis X.

***Non-intelligent expansion axis "up"***

Axis 1 → PLC expansion axis without microprocessor. Indicated in the blocks as the Y axis. Available upon request.

***Intelligent "up" expansion axis***

Axis 2 → First expansion axis Exp_Ax with microprocessor. Indicated in the blocks as X expansion axis.
Axis 3 → Second expansion axis Exp_Ax with microprocessor. Indicated in the blocks as Y expansion axis.

***"Bus" expansion axis***

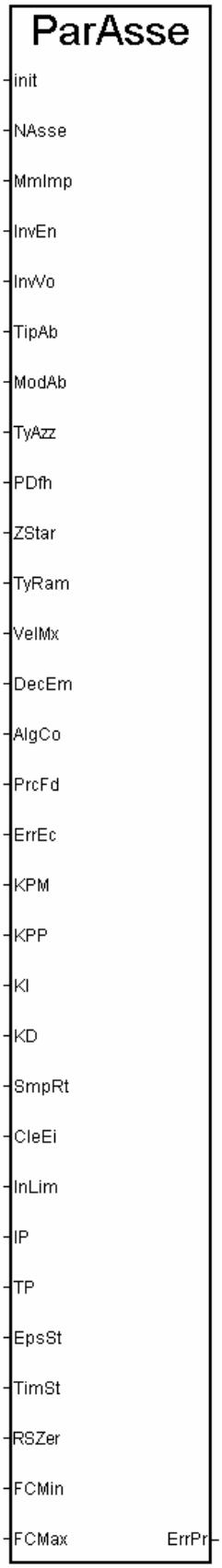
Axis 4 → First "bus" expansion axis without microprocessor. Indicated in the blocks as X expansion axis.
Axis 5 → Second "bus" expansion axis without microprocessor. Indicated in the blocks as Y expansion axis.

Non-intelligent "up" expansion axis is the alternative to the intelligent "up" expansion axis. In the event of the use of the intelligent "up" expansion axis there will therefore be three axes the numbering of which is 0, 2 and 3. In the event of the use of the non-intelligent "up" expansion axis there will only be two axes the numbering of which is 0 and 1.
In both configurations referred to above it is possible to add the two "bus" expansion axes so as to bring the total number of controllable axes respectively to 5 or 4.
The performance of the configuration based on SU210 with 5 axes is inferior compared with the performance of the CND51 family in that the latter uses a DSP. If the performance required of the system is of a high level we recommend that the CND51 system be used.

The numbering of the axes for the CND51 family is the following:
Axis 0 → Axis X
Axis 1 → Axis Y
Axis 2 → Axis Z
Axis 3 → Axis U
Axis 4 → Axis V
Axis 5 → Axis W

It is possible to connect expansion axes onto the SU21x system by way of a CAN connection. If these expansion axes are not Arteco spa products an interface through an I/O Connection will be necessary as indicated in the manual *"CanOpen Analogue Axes - Appendix 1".* If the expansion axes are Arteco products it will be possible to utilise the same management blocks used for the base axes, referring to this present manual and to the manual "*CanOpen Analogue Axes* ".
For information of any kind whatsoever please contact Arteco spa

## ParAxis


ParAsse
init
NAsse
MmImp
InvEn
InvVo
TipAb
ModAb
TyAzz
PDfh
ZStar
TyRam
VelMx
DecEm
AlgCo
PrcFd
ErrEc
KPM
KPP
KI
KD
SmpRt
CleEi
InLim
IP
TP
EpsSt
TimSt
RSZer
FCMin
FCMax
ErrPr

# Topics

| | | |
|---|---|---|
| init | BOO | On the positive edge initialise the axis. |
| Naxis | INT | Axis number to initialise. |
| MmImp | INT | Space in mm corresponding to an encoder impulse. The value is multiplied by 10 raised to the 8th power.<br>Ex. 1. To introduce the value 0.001mm (each mark is equal to 1/1000 of a mm) assign 100,000. Ex. 2. To introduce 1 mm, assign 1,000,000.<br>Values between 100 and 2,000,000 (2 mm) are accepted. |
| InvEn | BOO | Encoder inversion. Permits the direction of the encoder count to be inverted without cabling modification. |
| InvVo | BOO | Permits the direction of output voltage to be inverted. |
| TipAb | INT | Indicates the utilisation mode for axis enabling. It also determines the modality for the direction signal.<br>0 => Never enables the axis. This operation must be carried out by the application program. The axis direction signal is not provided.<br>1 => Enabling and direction are provided at 24 volts respectively on the J2 pin 16 and J3 pin 15, in the place of the user output.<br>2 => Enabling at 24 volts on J2 pin 16, while the direction is not provided leaving the 14-user output available.<br>3 => Like modality 1.<br>4 => Like modality 2.<br>5 => Like modality 1.<br>6 => Enabling provided at 24 volts on J2 pin 16 and direction provided on two outputs, respectively J3 pin 15 (positive direction) and pin 14 (negative direction).<br>Enabling management by way of user program could cause some problems.<br>It is recommended that one of the automatic modalities be used based on the actuation demands. For unforeseen requirements, contact ARTECO spa. To utilise the 5-volt outputs as user outputs if axis management is not used, contact ARTECO spa. |
| ModAb | INT | Indicates the utilisation mode for axis enabling.<br>0 => Axis always enabled. Enabling is provided at the end of the correct parameterisation of the axis. It is removed in case of serious error or if the application is stopped.<br>If the application is stopped during a movement, the enabling will be removed at the end of the arrest trajectory.<br>1=> Axis enabled only during movements. Enabling is provided at the initiation of the trajectory and is removed after 1 second from its arrival in position<br>2=> Actuation OK. Enabling is provided upon the first request of movement or at the first Axilock, after the axis has been parameterised and the “actuation OK“ input is active. If this input is lowered, the enabling also goes off. Input utilised is the special 0 input. |

| | | |
|---|---|---|
| TyAzz | INT | Type of zeroing out. Whole value that indicates the zeroing out modality of the axis.<br>0 => It zeroes out on positive edge sensor detection. The request execute machine zero out must come about with the sensor deactivated.<br>1 => Zeroes out upon release of the sensor.<br>2 => Zeroes out upon detection of the zero mark.<br>3 => Zeroes out upon detection of the zero mark without inverting the direction of movement.<br>** See ZeroAxis block.* |
| PDfh | INT | Zero position. Height at which the axis zeroing out point is to be attributed. Expressed in mm multiplied by 1,000. Ex. 100mm => 100,000.<br>** See ZeroAxis block.* |
| ZStar | INT | Determines if the axis is to be moved without having first carried out the zeroing out of the machine, after turning on the control. If equal to 0 the machine zero will have to be carried out only if the height turns out to be corrupted. If equal to 1, the machine zero out must always be performed after start up. If equal to 2 indicates that in case of a corrupted height Set Height is sufficient. If equal to 3 zero axis is never necessary. |
| TyRam | INT | Maximum time for the nearing of the axis to its destination, at the end of the trajectory. |
| VelMx | INT | Maximum velocity of the axis. This must correspond with the velocity of the axis when the control provides maximum output voltage corresponding to 10 volts.<br>Expressed in mm/sec it must be assigned multiplied by 1,000.<br>Ex. 700mm/sec => 700,000.<br>Minimum value 1; maximum value 10 m/sec. |
| DecEm | INT | Deceleration to be applied for the stops due to emergency, such as the arrival at limit stop.<br>Expressed in mm/(sec*sec) it must be assigned multiplied by 1,000.<br>Ex. 1,000mm/(sec*sec) => 1,000,000.<br>Minimum value 5mm/(sec*sec) maximum value 10 m/(sec*sec) |
| AlgCo | INT | Movement algorithm. If 0, perform the positioning utilising the PID formula, exclusively.<br>If 1, perform the positioning utilising Feedforward and proportional constant.<br>Accept only the values 0 and 1. |
| PrcFd | INT | Feedforward action percentage value. Allows the calculation of the percentage of the feedforward action in the event that the AlgCo parameter is equal to 1. The value may be from 0 to 200. |
| ErrEc | INT | Excessive error. If the movement error calculated as the difference between the real and the theoretical position is greater than what is indicated in this parameter the positioning is stopped and a tracking error is generated.<br>Expressed in mm it must be multiplied by 1,000. Ex. 100mm => 100,000.<br>Minimum value 0; maximum value 10 m. |
| KPM | INT | Proportional constant utilised by the PID formula during a movement. |
| KPP | INT | Proportional constant utilised by the PID formula during the adjustment with axis stopped, so as to maintain the position. |
| KI | INT | Integral constant applied by the PID only if AlgCo is equal to 1. |
| KD | INT | Derivative constant applied by the PID only if AlgCo is equal to 1. |
| SmpRt | INT | Divisor for the effect of the PID adjustment parameters. |
| CleEi | BOO | Activates the zeroing out of the integral error when the command changes sign. |
| InLim | INT | Integral limit. Maximum value taken on by the only integral component in the PID calculation. Utilised only if AlgCo equals 1. |

| | | |
|---|---|---|
| IP | INT | Indicates at what distance from the destination point the theoretical position equal to that of the destination is to be forced.<br>Expressed in mm it must be multiplied by 1,000. Ex. 0.5mm => 500.<br>Minimum value 0; maximum value 100mm. |
| TP | INT | Indicates the destination tolerance allowed in order to consider a movement finished without positioning error.<br>Expressed in mm it must be multiplied by 1,000. Ex. 0.5mm => 500.<br>Minimum value 0; maximum value 100mm. |
| EpsSt | INT | Maximum tolerance allowed in order to consider an axis stopped after arrival at destination. The axis must remain within this tolerance for the number of milliseconds indicated by the TimSt parameter.<br>Expressed in mm it must be multiplied by 1,000. Ex. 0.5mm => 500.<br>Minimum value 0.005mm maximum value 100mm. |
| TimSt | INT | Time expressed in msec during which the axis must remain within the tolerances indicated by the parameter EpsSt, so as to be considered stopped.<br>Ex. 100msec => 100.<br>Minimum value 0 sec; maximum value 6 sec. |
| RSZer | INT | Allows the selection, if, for the zero axis, the zero sensor is to be utilised or else the minimum or maximum sensors (according to the direction of the zeroing out). 0 indicates the utilisation of the zero sensor, while 1 indicates the utilisation of the minimum or maximum sensors. |
| FCMin | INT | Software limit stop minimum position.<br>Expressed in mm it must be multiplied by 1,000.<br>Ex. 1. 2345.67mm => 2,345,670. Ex. 2. -147.6 => -147,600.<br>PLEASE NOTE: An FC hardware not manageable by the application is available. |
| FCMax | INT | Software limit stop minimum position. Expressed in mm it must be multiplied by 1,000.<br>Ex. 1. 2345.67mm => 2,345,670. Ex. 2. -147.6 => -147,600.<br>PLEASE NOTE: An FC hardware not manageable by the application is available. |

| | |
|---|---|
| ErrPr | Code that indicates if the parameterisation is correct or if an eventual error has been verified:<br>Low byte - parameter that generates error.<br>0 indicates the axis number.<br>1 indicates MmImp.<br>etc.<br>High byte - error code<br>1 Command not executed because the axis is not stopped.<br>2 Value not admitted. Ex. the number of the axis is different from zero.<br>3 Value off scale because it is higher than the maximum value.<br>4 Value off scale because it is lower than the minimum value.<br>5 DSP in malfunction (only SU250).<br>6 Overflow after internal scale set.<br>7 Underflow after internal scale set.<br>255 Axis not foreseen in the software. Contact Arteco spa.<br>Ex. 1035 => 40B hexadecimal: Parameter 11 too small.<br>If equal to zero the parameterisation has finished correctly and the axis is available.<br>See Elaboration. |

# Description:

This block allows the axis manager to be parameterised, communicating all of the necessary data for its correct operation. It additionally allows the activation of the axis manager, as no movement is possible before having recalled this block. If there are one or more parameters that cause an error in the block (ErrPr different from zero), the axis manager remains disabled stopping any movement. This block may not be and must not be activated during movements. It is sufficient to recall it at the start up of the application. All of the data that require modification repeatedly are requested by other blocks.

## Axis parameterisation elaboration

Certain parameters of this block are described in a more elaborated manner below.

**MmImp:** this parameter should make up a certain mechanism of the machine to be checked, but it is possible to obtain it through experimentation. Set an approximate conversion ratio, and then manually move the axis and precisely measure the distance travelled and then, the distance travelled indicated by the position reader block
Apply the following formula to obtain a true and credible datum.
real_factor = (real_space_travelled * approximate_factor)/space_indicated_by_control
The greater the space utilised for the test, the greater the precision of the datum obtained.

**AlgCo**: Indicates the type of algorithm utilised to carry out the movement required. If 0, utilise the PID formula, where the error between the theoretical position and the actual position determines the movement. Said error is elaborated utilising the constants: **KPP, KPM, KI, KD**. The KP constant multiplies the error between the actual position and the theoretical position. The KD constant multiplies the difference between the error detected and that at the instant before, reducing its effect upon adhering to the real position and the theoretical one. The KI constant multiplies the summation of the accumulated errors, strengthening its effect if the error does not diminish over time. The parameter **CleEi** indicates if the summation of the errors is to be zeroed out, when it has been detected that the movement command has changed sign (direction change). The KPM constant is utilised during movement, while KPP is utilised to maintain the position of the stopped axis. In this case the derivative and integral effects are not active. The parameter **InLim** allows a limit to be set on the summation due to the integral effect. The parameter **SmpRt** allows the effect of the correction referred to above to be divided permitting a greater calibration precision. If SmpRt>=0 the classic PID formula is utilised, in which KP represents the gain of the adjustment loop and influences the remaining actions as well (integral and derivative). If SmpRt<0 a PID formula is introduced that considers the effects of the KP, KI and KD released one from the other. The absolute value of SmpRt is used as division coefficient of K to obtain values less than 1. In addition the value of the integral is reduced in percentage when the integral and the error are discordant.
If the AlgCo parameter is set at 1, the movement is obtained as the sum of the theoretical curve plus the correction of the only KP. At every instant, independent of the accumulated error, the necessary theoretical command is applied to follow the profile required for the movement. The error between the actual position (much more adherent to the theoretical one compared to the case observed previously) and the theoretical position are multiplied by KP so as to reduce the movement error to a minimum. In this case the KP constant must be inferior so that the required correction is less.
In both cases the values of the constants just cited that are too high cause an oscillation; vice versa values that are too low cause an ever-greater tracking error.

**PrcFd**: This allows the effect of the feedforward to be limited in case AlgCo equals 1. The theoretical command provided may have the percentage calculated so as to increase the proportional constant. This operation allows greater precision to be obtained in the final trajectory phase and to reduce the oscillation during movement.
**IP** and **TP**: These are utilised to increase precision in the final trajectory phase. At a distance equal to IP from the destination the theoretical position abandons the theoretical ramp profile and is set equal to the destination position. This causes an abrupt tracking error that assists the axis in attaining its destination point with greater precision. At the end of the movement if the distance from the destination point is less than at TP, the movement is considered finished without errors.
The utilisation of TP allows a positioning tolerance to be created.
The utilisation of IP to improve the positioning is particularly useful for axes with strong inertia, while it may be counter-productive if the masses at play are low.

**ZSTAR**: This allows the execution of the zero axis to be imposed before performing any movement. If equal to zero, after the parameterisation, the zero axis is requested only if the height present in memory is found to be corrupted. If equal to one, the zero axis is in any case requested even if the height is not found to be corrupted. This is useful if there is the possibility that the axis might be moved with the control turned off. In this last case performing the movement without carrying out the zero axis could cause mechanical shocks. If equal to 2 it is possible to eliminate the invalid height condition by way of a simple Set Height. If equal to 3 it is possible to perform any movement even with corrupted height without the necessity of performing a zero axis or a set height.

**FCMin FCMax**: The minimum and maximum limit stop software is not influenced by origins management. These limit stops are ignored during the zero machine search operations.

**PDfh**: The position at the termination of the zero is not influenced by origins management.

**RSZer**: This allows the indication of whether the zero machine is to be performed seeking the zero sensor or else the minimum and maximum sensors. Set at Zero it seeks the zero sensor (in the direction indicated by the ZeroAxis block), while set at 1 it seeks the minimum or maximum sensor, according to the direction set in the ZeroAxis block. If the direction indicated in the Direct parameter is FALSE (backwards) the minimum sensor is sought. If the direction indicated is TRUE (forward) the maximum sensor is sought. If the RSZer parameter is equal to zero (seek zero sensor) the minimum and maximum limit stops will be exclusively used to stop and disable the axis if they are activated, as during the normal operation of the machine. The zero axis modality is in any case determined by the TyAzz parameter in the ParAxis block.
If the zeroing out comes about on the limit stop minimum or maximum sensor an error will be generated if its persistence on the aforementioned sensor lasts for a period greater than 4 seconds.

**TyRam**: For every movement the trajectory is considered finished at the moment of the achievement of the destination point. If due to obstructions of the axis or due to a too low required velocity/acceleration this were not to come about, TyRam indicates how many milliseconds to leave for the PID to bring the real point in line with the theoretical point. Time begins from the detection of the stopped axis before the achievement of the destination (based on the relative EpsSt and TimSt parameters). If the parameter is set at zero the value will be the one established in default, equal to 500 msecs.

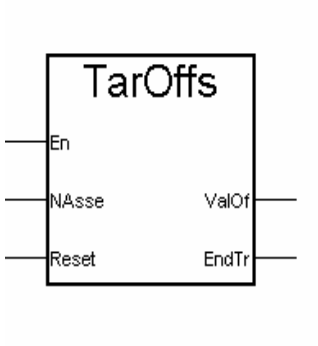
If the ErrPr output indicates error 1 (command not executed because axis not stopped), the low byte indicates with 0FFh that the axis has not finished a movement, while with 0FEh it indicates that the axis is not stopped because it is performing a zero axis.
If ErrPr indicates error 5 (DSP malfunction, only on SU250), the low byte will indicate a code that identifies the malfunction. This type of error could indicate a hardware problem: communicate the code to Arteco spa.

**ATTENTION**
At the end of parameterisation it may not be possible to move the axis. This occurs if the control detects a condition that could indicate a non-valid height. In this case an error from the ReadErr block will be signalled (See Appendix 1) and the positioning movements will again be available after having performed the ZeroAxis, or else after a Set Height (see Zstar parameter).
What was described above may occur in the event of a loss of memory data, or after a change in the firmware version. Check and verify the battery charge and if necessary contact Arteco spa.

**ARTECO MOTION TECH S.p.A.**

ISaGRAF® - axis control -

## TarOffs



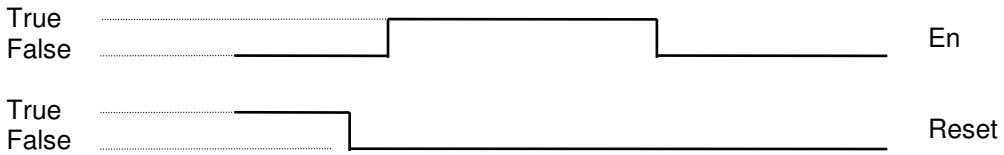

## Topics

| | | |
|---|---|---|
| En | BOO | Enabling the self/calibration function of the analogue offset. |
| NAxis | INT | Number of axes on which the self-calibration are to be performed. 0 for the X-axis. |
| Reset | BOO | Permits the offset to be zeroed out. |
| | | |
| ValOf | INT | Detected offset value in DAC unit. Zero indicates no offset |
| EndTr | BOO | It is TRUE when self-calibration is finished. |

## Description:

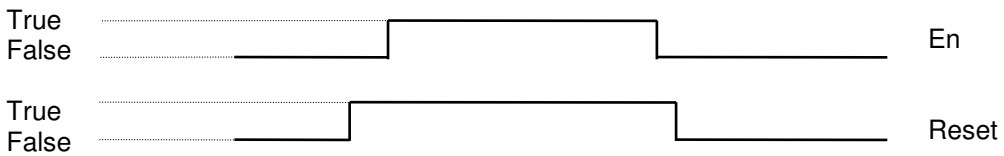
This block allows the calibration of the actuation offset in a fast and automatic manner.
To perform the self-calibration, position **Reset** on FALSE and **En** at TRUE (in this order).
Do not enable Reset before disabling En

**Self-Calibration**



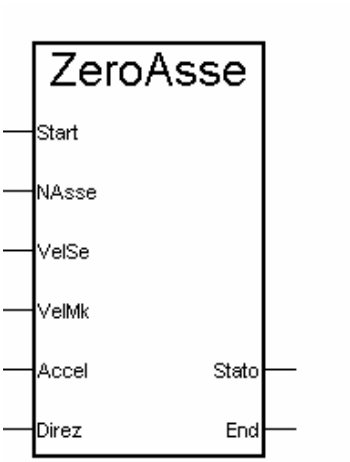

To zero out the offset put the **Reset** at TRUE and afterwards enable **En**. Do not put Reset at FALSE before having disabled En.

**Reset Offset**



This block may be activated only with the axis stopped. Vice versa a self-calibration not finished will be signalled, up until the axis has been stopped. Only at that point will the self-calibration of the axis offset be initiated. It is possible to calibrate the actuation by activating the block and acting upon the actuation trimmer itself so as to obtain **ValOf** equal to zero.
If the encoder is not connected or does not work, the offset calibration will stop immediately. If the direction encoder and/or the voltage command are not in accord (positive voltage movement corresponds to negative encoder movement) the offset calibration will signal error 11 (See Appendix 1 - Axis Manager Errors).

## ZeroAxis



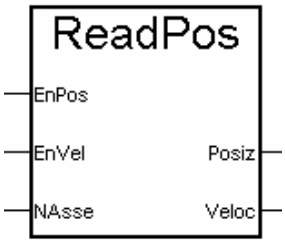

## Topics

| | | |
|---|---|---|
| Start | BOO | Starts zero axis on positive edge. |
| NAxis | INT | Axis number on which to perform zero out 0 for the X-axis. |
| VelSen | INT | Zero sensor search velocity<br>Expressed in mm/sec it must be assigned multiplied by 1,000.<br>Ex. 700mm/sec => 700,000. |
| VelMk | INT | Marker search and zero sensor release velocity.<br>Expressed in mm/sec it must be assigned multiplied by 1,000.<br>Ex. 700mm/sec => 700,000. |
| Accel | INT | Acceleration for movements at velocity as referred to above.<br>Expressed in mm/(sec*sec) it must be assigned multiplied by 1,000.<br>Ex. 1000mm/(sec*sec) => 1,000,000. |
| Direz | BOO | Indicates the zeroing out direction. TRUE indicates forward. FALSE backward. |
| End | BOO | Indicates if the zeroing out is finished. Deactivate at the start of the zeroing out, it is reactivated at the end. |
| State | INT | Indicates the internal zero execution state. Whole value that represents the movement state by way of a code:<br>0 => START UP SENSOR SEARCH<br>1 => WAIT SENSOR<br>2 => STOP AXIS<br>3 => SEARCH INVERSION FOR RELEASE<br>4 => WAIT SENSOR RELEASE<br>5 => WAIT ZERO MARK<br>6 => ZERO MARK DETECTED<br>7 => WAIT AXIS STOP<br>8 => -ZEROING OUT<br>9 => WAIT AXIS STOPPED<br>10 => PROCEDURE ENDED<br><br>PLEASE NOTE: Codes may be modified at the discretion of Arteco spa. |

# Description:

This block permits the axis zero to be performed. This procedure allows the axis to be positioned at a precise point and as necessary to assign that point a desired height.
Zeroing out modalities depend on the parameters **TyAzz** and **RSZer** of the **ParAxis** block.
If the TyAzz parameter has a value of 0, this block seeks the sensor in the direction indicated and as soon as it is detected, (within very few msec of the physical activation) zeroes out the axis eventually assigning the height required by the **PDfh** parameter of the **ParAxis** block.
If the parameter has a value of 1, this block seeks the sensor in the direction indicated and as soon as it is detected, inverts the movement direction to wait for the release of the same. Upon release of the sensor, (within very few msec of the physical activation) it zeroes out the axis eventually assigning the height required by the **PDfh** parameter of the **ParAxis** block.
If the parameter has a value of 2 the zeroing out is more precise. The block seeks the sensor in the direction indicated and as soon as it is detected inverts the movement direction to wait for the release of the same. Upon release (within very few msec of the physical deactivation) of the encoder zero mark search sensor, and as soon as it has been detected (procedure in DMA) the axis is zeroed out eventually assigning the height required by the **PDfh** parameter of the **ParAxis** block.
If the parameter has a value of 3 there is a zeroing out similar to case 2 but the axis never inverts direction.
The block seeks the sensor in the direction indicated, waits for its release, then seeks the encoder zero mark and once it has been detected, (procedure in DMA) the axis is zeroed eventually assigning the height required by the **PDfh** parameter of the **ParAxis** block.
The RSZer parameter of the ParAxis block determines if the sensor sought is the minimum or maximum zero sensor. See ZeroAxis block.
The assignment of the PDfh height introduces no error, in that the value is assigned at the zero point and not at the actual point at the moment of the end of the procedure.
If great precision is desired, avoid carrying out the same operation with the SetQuota block in that this could introduce an error.
The procedure may be ended by way of the axis stop block.
If the zero axis is performed using the minimum or maximum limit stop an error relative to the limit stop used will be generated if this latter remains active for a time greater than 5 seconds.
The zero axis procedure is carried out utilising movements at velocity, therefore with retro-action. Upon the first installation of the PLC check and verify that the encoder functions properly.

## ReadPos




## Topics

| | | |
|---|---|---|
| EnPos | BOO | Enabling input for position reading. If active in output it is provided with the position encoder expressed in mm. If FALSE it returns zero. |
| EnVel | BOO | Enabling input for velocity reading. If active in output it is provided with the axis velocity expressed in mm/sec*sec. If FALSE it returns zero. |
| NAxis | INT | Axis number of which velocity and position are to be known. 0 for the X-axis. |
| Posiz | INT | Position of the axis expressed in mm, multiplied by 1,000, read by the encoder. Ex. 1,000mm => 1,000,000. |
| Veloc | INT | Velocity of the axis expressed in mm/sec, multiplied by 1,000, measured by the encoder. Ex. 1,000mm/sec => 1,000,000. |

## Description:

This block allows the actual position of the axis and its velocity to be read. The **En** inputs enable the velocity and position outputs. If set at FALSE, the respective output becomes zero. This in no way prejudices the function of the axis but permits a reduction of the PLC cycle execution times as the calculations necessary for the transformation of the impulse encoder data to millimetres are not executed. The axis position is in any case monitored and the En activation provides correct data in real time.
The position returned is relative to the currently active origin.
It is also possible to read current velocity and position by way of the I/O nS210Axis or ExpAxis library according to the configuration utilised (See Library I/O manual Customisations paragraph).

**ARTECO MOTION TECH S.p.A.**

ISaGRAF® - axis control -

## StopAssd


## Topics

| | | |
|---|---|---|
| STOP | BOO | If active, it stops the axis. If maintained active it impedes the axis from starting. |
| NUM_A | INT | Axis number to stop. |
| DECEL | INT | Deceleration utilised to stop the axis.<br>Expressed in mm/(sec*sec) it must be assigned multiplied by 1,000.<br>Ex. 1000mm/(sec*sec) => 1,000,000. |
| STATE | INT | Indicates the internal status of the axis manager Whole Value that represents the movement status by way of a code:<br>0 => -STOPPED<br>1 => STOPPED STANDBY<br>2 =>EXPECTED STOP<br>3 => ABRUPT STOP<br>4 =>EMERGENCY STOP<br>5 =>IN EMERGENCY STOP<br>6 =>CONTROLLED STOP<br>7 =>IN CONTROLLED STOP<br>8 => RELEASED<br>9 => ZERO VOLTS<br>10 => ZERO MACHINE<br>11 => MOVEMENT AT VELOCITY<br>12 => MOVEMENT IN JOG<br>13 => ERROR<br><br>PLEASE NOTE: Codes may be modified at the discretion of Arteco spa. |

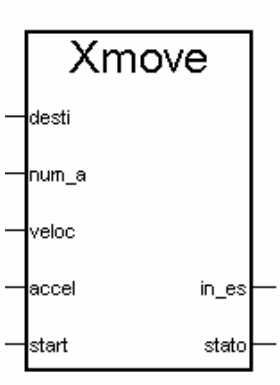
## Description:

This block allows the stop of the axis from any type of movement in progress. The stop comes about with ramp. If the block is kept active, it will not be possible to start up any movement. This block is different from StopAxis because of the possibility of assigning a value to the deceleration with which the axis is stopped.

**ARTECO MOTION TECH S.p.A.**

ISaGRAF® - axis control -

## XMove



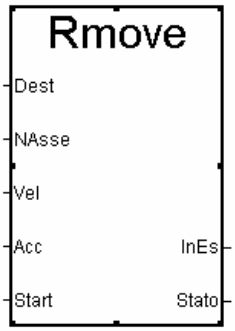

## Topics

| | | |
|---|---|---|
| desti | INT | Axis destination height. Expressed in mm multiplied by 1,000.<br>Ex. 100mm => 100,000. |
| num_a | INT | Axis number to be moved. |
| veloc | INT | Axis velocity during movement. Expressed in mm/sec it must be assigned multiplied by 1,000.<br>Ex. 700mm/sec => 700,000. |
| accel | INT | Acceleration of the ramps. Expressed in mm/(sec*sec) it must be assigned multiplied by 1,000.<br>Ex. 1000mm/(sec*sec) => 1,000,000. |
| start | BOO | Starts axis movement on positive edge. |
| in_es | BOO | Indicates if the movement is still in progress. TRUE indicates movement in execution, FALSE movement ended. |
| State | INT | Indicates the internal state of the axis manager Whole value that represents the movement state by way of a code:<br>0 => -STOPPED<br>1 => STOPPED STANDBY<br>2 =>EXPECTED STOP<br>3 => ABRUPT STOP<br>4 =>EMERGENCY STOP<br>5 =>IN EMERGENCY STOP<br>6 =>CONTROLLED STOP<br>7 =>IN CONTROLLED STOP<br>8 => RELEASED<br>9 => ZERO VOLTS<br>10 => ZERO MACHINE<br>11 => MOVEMENT AT VELOCITY<br>12 => MOVEMENT IN JOG<br>13 => ERROR<br><br>PLEASE NOTE: Codes may be modified at the discretion of Arteco spa. |

## Description:

This block allows movements at velocity to be performed toward a requested destination. Operational velocity and the acceleration to achieve it and to stop are set respectively in veloc and accel. If the destination requested goes beyond the limit stop there is the stop on the ramp upon the achievement of the limit stop software. The limit stop hardware in any case causes an abrupt stop
The final destination IS influenced by the origins management. See GstOrg block.

## Rmove




## Topics

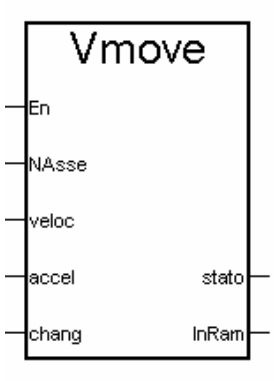
| | | |
|---|---|---|
| Dest | INT | Shift of actual axis position. Expressed in mm multiplied by 1,000.<br>Ex. 100mm => 100,000. |
| Naxis | INT | Axis number to be moved. |
| Vel | INT | Axis velocity during movement. Expressed in mm/sec it must be assigned multiplied by 1,000.<br>Ex. 700mm/sec => 700,000. |
| Acc | INT | Acceleration of the ramps. Expressed in mm/(sec*sec) it must be assigned multiplied by 1,000.<br>Ex. 1000mm/(sec*sec) => 1,000,000. |
| Start | BOO | Starts axis movement on positive edge. |
| InEs | BOO | Indicates if the movement is still in progress. TRUE indicates movement in execution, FALSE movement ended. |
| State | INT | Indicates the internal status of the axis manager Whole value that represents the movement status by way of a code:<br>0 => -STOPPED<br>1 => STOPPED STANDBY<br>2 =>EXPECTED STOP<br>3 => ABRUPT STOP<br>4 =>EMERGENCY STOP<br>5 =>IN EMERGENCY STOP<br>6 =>CONTROLLED STOP<br>7 =>IN CONTROLLED STOP<br>8 => RELEASED<br>9 => ZERO VOLTS<br>10 => ZERO MACHINE<br>11 => MOVEMENT AT VELOCITY<br>12 => MOVEMENT IN JOG<br>13 => ERROR<br><br>PLEASE NOTE: Codes may be modified at the discretion of Arteco spa. |

## Description:

This block allows movements at velocity to be performed toward a requested destination considering the current height as zero. The Dest Input indicates therefore the distance to be travelled compared to the current point and its sign indicates the direction. Operational velocity and the acceleration to achieve it and to stop are set respectively in Vel and Acc. If the destination requested goes beyond the limit stop there is the stop on the ramp upon the achievement of the limit stop software. The limit stop hardware in any case causes an abrupt stop

The final destination IS NOT influenced by the origins management. See GstOrg block.

## Vmove




## Topics

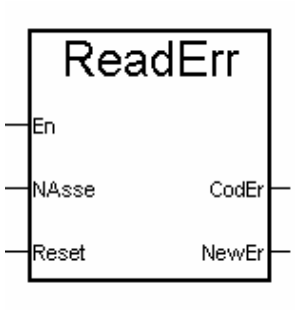
| | | |
|---|---|---|
| En | BOO | Start up the open loop movement in voltage. TRUE starts the movement, FALSE stops it. It acts on the edges. |
| Naxis | INT | Indicates the axis on which movement in voltage is to be initiated. |
| veloc | INT | Theoretical axis velocity during movement. Expressed in mm/sec it must be assigned multiplied by 1,000.<br>Ex. 700mm/sec => 700,000. |
| accel | | Acceleration of the ramps. Expressed in mm/(sec*sec) it must be assigned multiplied by 1,000.<br>Ex. 1000mm/(sec*sec) => 1,000,000. |
| chang | BOO | Also allows the variation of the velocity in movement in voltage already initiated (by way of positive edge of En). If active the axis veloc and accel values follow those indicated in the parameters referred to above. The chang activation is influential only if En is active. To avoid the execution of complex calculations it is recommended to not leave chang active unnecessarily. |

| | | |
|---|---|---|
| state | INT | Indicates the internal status of the axis manager Whole Value that represents the movement status by way of a code:<br>0 => -STOPPED<br>1 => STOPPED STANDBY<br>2 =>EXPECTED STOP<br>3 => ABRUPT STOP<br>4 =>EMERGENCY STOP<br>5 =>IN EMERGENCY STOP<br>6 =>CONTROLLED STOP<br>7 =>IN CONTROLLED STOP<br>8 => RELEASED<br>9 => ZERO VOLTS<br>10 => ZERO MACHINE<br>11 => MOVEMENT AT VELOCITY<br>12 => MOVEMENT IN JOG<br>13 => ERROR<br><br>PLEASE NOTE: Codes may be modified at the discretion of Arteco spa. |
| InRam | BOO | Indicates if the actual theoretical velocity has not achieved the requested theoretical velocity. TRUE counts if the approach ramp is in the process of achieving required voltage. FALSE counts if the theoretical velocity is equal to that required.<br>For CAN piloted axes, this output indicates if the movement is still in progress and has the same meaning as the in_es output in the Xmov block. TRUE indicates that on this axis a movement is in progress. FALSE indicates a movement ended. |

# Description:

This block allows the carrying out of movements, **in voltage**, joining the actual velocity with the theoretical velocity required. It is possible to modify said velocity (and the acceleration to achieve it) enabling the **chang** input.
In that the movement is open loop the required velocity is indicative and cannot be guaranteed.
The movement may be ended by way of the axis stop block. If the En input is active it will be necessary to deactivate it so as to be able to carry out a new movement at voltage. If a movement at voltage is stopped by way of the axis stop it is possible to perform other movements (not at voltage) even without the deactivation of the En input.
This block is indicated for movements that bring the axis to the rabbet, since it does not generate a tracking error.
If it is necessary to move the axis at a precise velocity and with control of the actual velocity utilise the VelMove block.

## ReadErr




## Topics

| | | |
|---|---|---|
| En | BOO | Input to enable error reading. |
| NAxis | INT | Axis number of which the error is to be known. |
| Reset | BOO | Activated, allows the zeroing out of the New error indication (NewEr). |
| | | |
| CodEr | INT | Code of last error detected. See table of error codes |
| NewEr | BOO | New error signalling. Active if a new error is generated. It is reset upon turn on of the control or by way of the Reset input. |

## Description:

This block allows the reading of the errors coming from the axis manager.
The **CodEr** output always shows the last error that occurred and is reset exclusively with the turning off of the control.
The NewEr output becomes TRUE each time that an error is verified and is reset exclusively with the shut down or activating the **Reset** input.
If the **En** input is put in FALSE the CodEr and NewEr outputs always show the last values read with En active.
Error codes may be found in Appendix 1.

**ARTECO MOTION TECH S.p.A.**

ISaGRAF® - axis control -

## SetHeight


## Topics

| | | |
|---|---|---|
| En | BOO | Modifies height on positive edge. |
| Naxis | INT | Axis number of which the height is to be changed. |
| Height | INT | New current height. Whole analogue. Expressed in mm multiplied by 1,000. Ex. 100mm => 100,000. |
| Ok | BOO | It is deactivated at the request of modify height. It is reactivated at the end of the change so as to indicate that the current height indicated by the system has been changed. The velocity of the system may impede the OK signal from changing. If the signal becomes false for a considerably long time, contact Arteco spa. |

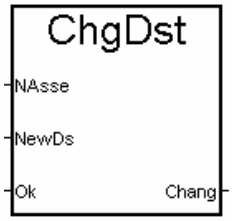
## Description:

This block allows the current axis height to be modified, having the actual position correspond to a desired height.
The height change comes about exclusively on the positive edge of the **En** signal.
If a movement into position is in progress the arrival point will not be the destination requested at the moment of the launch of the movement.
The utilisation of this block may compromise the precision of the system. Do not use it to assign a height for the zero machine point and contact Arteco in the event that an intensive use of this block is required.
Utilisation of the origins management is preferable. See GstOrg block.
The height set refers to the current origin.

**ARTECO SPA**

ISaGRAF® - axis control -

## ChgDst




## Topics

| | | |
|---|---|---|
| NAxis | INT | Axis number of which the height is to be changed. |
| NewDs | INT | New destination. Whole analogue. Expressed in mm multiplied by 1,000. Ex. 100mm => 100,000. |
| Ok | BOO | On the positive edge of this signal the height present at the NewDst input is transferred to the axis manager as a new height destination. See description. |
| Chang | BOO | Indicates the successful transfer of the NewDs height as a new destination. FALSE indicates that the transfer of the NewDs height operation is in progress at the axis manager, which interprets it as a destination position. The operation is so fast that Chang could be seen as always active by the operational program. If the signal becomes FALSE for a considerably long time, contact Arteco spa. |

## Description:

This block permits the motion of the axis to be changed in an “impromptu” manner, that is, without stopping the axis and then having to restart it toward another height. The destination change comes about on the positive edge of the OK input. The destination may be changed only during a movement and in any case in any part of the trajectory (acceleration, constant velocity or deceleration). The request of a destination already passed by the axis causes an immediate stop in ramp and the signalling of error 8 (see appendix 1), to indicate the impossibility of achieving said point.
It is necessary to underline that the deceleration of the trajectory in progress, after the change in destination, could no longer be that requested, just as the achievement of the operational velocity requested is not guaranteed.

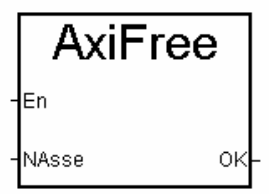
In particular:

1) If during deceleration the destination is changed, taking it away from the current one, the ramp is interrupted and it advances at a constant velocity (it does not return to operational velocity), until the achievement of the point in which the deceleration ramp starts over again in order to stop at the new point requested.
2) If during deceleration the destination is changed nearing it to the current one, the deceleration increases to guarantee the stop within the new destination programmed. In this case it is necessary to evaluate if the mechanical systems may become damaged because of a greater acceleration than that required.
3) If during the segment at constant velocity the destination is changed nearing it to the current one in a space such that the deceleration anticipated is not consented, the axis stops with a deceleration greater than that required. In this case it is necessary to evaluate if the mechanical systems may become damaged because of a greater acceleration than that required.
4) If during the acceleration phase the destination is changed nearing it to the current one in a space such that the full running of the acceleration and deceleration ramps is not permitted, the axis will immediately begin a deceleration ramp tending toward the achievement of the new destination height. In this case the deceleration will be less than that required. This case does not anticipate the achievement of the required operational velocity.

This block may be utilised together with the Capt_Int block to change the destination upon detection of an input at interrupt. For information contact Ateco spa

The final destination IS influenced by the origins management. See GstOrg block.

## AxiFree




## Topics

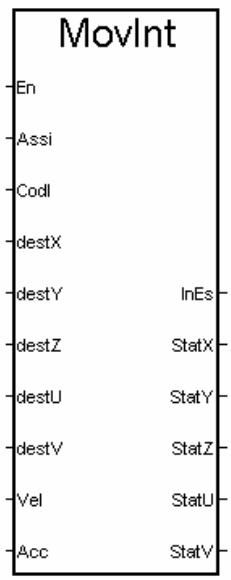
| | | |
|---|---|---|
| En | BOO | On the positive edge it disables the axis and the axis control loop. |
| NAxis | INT | Axis number to be disabled. |
| Ok | BOO | Result of the command. TRUE indicates command executed. FALSE indicates control waiting for axis release. See description. |

## Description:

This block disables the axis control loop and its enabling. The direction outputs also remain deactivated. This block disables the axis on the positive edge of the En input, only if a movement is not in execution. In this case wait for the stop of the axis and only at that point is the control loop disabled

This block allows the axis to be disabled at the end of a movement with the same modality seen in the ModAb parameter in the ParAxis parameterisation block, with the possibility of choosing when to disable it. The axis is automatically enabled again upon request of a new movement. To enable the axis again without requesting a movement, but keeping it in position by way of the control loop, it is possible to utilise the AxiLock block. If the axis is released by way of this block, it will remain so even if the emergency button is pushed and then released. If the block is recalled during the emergency, the axis will be reactivated at the end of the emergency event, if this has been anticipated by the ModAbi and TipAb parameters of the PasAxis block.

## MovInt




## Topics

| | | |
|---|---|---|
| En | BOO | On the positive edge the starts the interpolated movement requested based on the other data in the block. |
| Axes | INT | Mask containing the axes involved in the interpolation. Each bit indicates an axis to start up, where the least significant bit (zero) indicates the X axis; the one bit indicates the Y axis, up to the four bit which indicates the V axis. |
| CodI | INT | Interpolation code requested.<br>0 => Not utilised<br>1 => Launches movements in linear interpolation on two axes.<br>2 => Launches movements in linear interpolation on three axes.<br>3 => Launches movements in linear interpolation on two axes with relative heights.<br>4 => Launches movements in linear interpolation on three axes with relative heights.<br><br>For the SU210 data card with axis expansion by way of BUS and for the Motion Kit data card with 2 axis expansion, for interpolation that regards the 4 and 5 axes the following codes are used:<br>21 => Launches movements in linear interpolation on two axes.<br>22 => Launches movements in linear interpolation on three axes.<br>23 => Launches movements in linear interpolation on two axes with relative heights.<br>24 => Launches movements in linear interpolation on three axes with relative heights.<br>With the codes 21 and 23 the destination heights are in destX and destY. With the codes 22 and 24 the destination heights are in destX, destY and destZ. |

| | | |
|---|---|---|
| destX | INT | Axis X destination height (represents a coordinate of the destination point). Expressed in mm multiplied by 1,000. With Codl = 21, 22, 23 and 24 (SU210 data card with axis expansion by way of BUS or Motion Kit data card with 2 axis expansion) represents the destination height of the first axis to be interpolated. Ex. 100mm => 100,000. |
| DestY | INT | Axis Y destination height (represents a coordinate of the destination point). Expressed in mm multiplied by 1,000. With Codl = 21, 22, 23 and 24 (SU210 data card with axis expansion by way of BUS or Motion Kit data card with 2 axis expansion) represents the destination height of the second axis to be interpolated. Ex. 100mm => 100,000. |
| destZ | INT | Axis Z destination height (represents a coordinate of the destination point). Expressed in mm multiplied by 1,000. With Codl = 22 and 24 (SU210 data card with axis expansion by way of BUS or Motion Kit data card with 2 axis expansion) represents the destination height of the third axis to be interpolated. Ex. 100mm => 100,000. |
| destU | INT | Axis U destination height (represents a coordinate of the destination point). Expressed in mm multiplied by 1,000. Ex. 100mm => 100,000. |
| destV | INT | Axis V destination height (represents a coordinate of the destination point). Expressed in mm multiplied by 1,000. Ex. 100mm => 100,000. |
| Vel | INT | Velocity of the point interpolated along the trajectory during the movement. Expressed in mm/sec it must be assigned multiplied by 1,000. Ex. 700mm/sec => 700,000. |
| Acc | INT | Velocity of the point interpolated along the trajectory during the ramps. Expressed in mm/(sec*sec) it must be assigned multiplied by 1,000. Ex. 1000mm/(sec*sec) => 1,000,000. |
| InEs | BOO | Indicates if the movement is still in progress. TRUE indicates movement in execution, FALSE movement ended. |

| | | |
|---|---|---|
| StatX<br>StatY<br>StatZ<br>StatU<br>StatV | INT | Indicates the internal state of the axis manager Whole value that represents the movement state by way of a code:<br>0 => -STOPPED<br>1 => STOPPED STANDBY<br>2 =>EXPECTED STOP<br>3 => ABRUPT STOP<br>4 =>EMERGENCY STOP<br>5 =>IN EMERGENCY STOP<br>6 =>CONTROLLED STOP<br>7 =>IN CONTROLLED STOP<br>8 => RELEASED<br>9 => ZERO VOLTS<br>10 => ZERO MACHINE<br>11 => MOVEMENT AT VELOCITY<br>12 => MOVEMENT IN JOG<br>13 => ERROR<br>Each output represents the status of the respective axis.<br>With Codl = 21, 22, 23 and 24 (SU210 data card with axis expansion by way of BUS or Motion Kit data card with 2 axis expansion) the first two or three outputs indicate the status of the interpolated axes.<br>PLEASE NOTE: Codes may be modified at the discretion of Arteco spa. |

## Description:

This block allows the interpolated movements to be executed on more than one axis providing destination point coordinates, operational velocity and the acceleration to achieve it and to stop. If the requested destination goes beyond the limit stop position of one of the axes involved in the interpolation, it stops in ramp upon achievement of the limit stop software, still remaining along the trajectory programmed for the interpolation. The limit stop hardware in any case causes an abrupt stop.

The Codl code determines the type of interpolation and may assume the following values:

0 => Not utilised

1 => Launches a linear interpolation movement on two axes, the destination of which is represented by the absolute heights indicated in the *dest* parameter.

2 => Launches a linear interpolation movement on three axes, the destination of which is represented by the absolute heights indicated in the *dest* parameter.

3 => Launches a linear interpolation movement on two axes, the destination of which is represented by the relative heights (compared to the current position) indicated in the *dest* parameter.

4 => Launches a linear interpolation movement on three axes, the destination of which is represented by the relative heights (compared to the current position) indicated in the *dest* parameter.

For the SU210 data card with axis expansion by way of BUS and for the Motion Kit data card with 2-axis expansion, for interpolations that regard the 4 and 5 axes the following codes are used:

21 => Launches a linear interpolation movement on two axes, the destination of which is represented by the absolute heights indicated in the destX and the *destY* parameters.

22 => Launches a linear interpolation movement on three axes, the destination of which is represented by the absolute heights indicated in the *destX, destY* and *destZ* parameters.

23 => Launches a linear interpolation movement on two axes, the destination of which is represented by the relative heights (compared to the current position) indicated in the *destX* and *destY* parameters.

24 => Launches a linear interpolation movement on two axes, the destination of which is represented by the relative heights (compared to the current position) indicated in the *destX, destY* and *destZ* parameters.

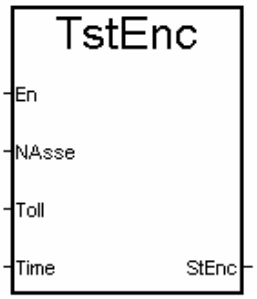
With the codes 21, 22, 23 and 24 the destination heights in destX, destY and destZ are relative to the interpolated axes beginning with the axis with the lowest number.

Similarly the StatX, StatY e StatZ outputs represent the status of the corresponding axis.

Ex: in the case of the interpolation of the 0 and the 5 axes, *destX* contains the destination height for the 0 axis (axisX) while *destY* contains the destination height for the 5 axis (axisW). The StatX output indicates the status of the X axis while StatY indicates the status of the W axis.

The final destination IS influenced by the origins management. See GstOrg block.

## TstEnc




## Topics

| | | |
|---|---|---|
| En | BOO | Enabling input for the axis status control based on the encoder height. If active in output the stopped axis indication is provided (TRUE) or in movement (FALSE), exclusively based on the encoder height. |
| NAxis | INT | Axis of which the status is to be found out. |
| Toll | INT | Tolerance inside of which the encoder height may oscillate without considering the axis in movement. Expressed in mm multiplied by 1,000.<br>Ex. 1mm => 1,000. |
| Time | INT | Time interval during which the encoder height must remain within the tolerance referred to in the Toll parameter, in order to consider the axis stopped.<br>Expressed in milliseconds.<br>**The time interval indicated by the Time parameter is to be considered purely indicative, in that the response of the block in question depends upon the sampling of the axis (datum dependent upon the control utilised) and by the PLC scanning time (time necessary for the execution of a PLC cycle). |
| StEnc | BOO | Indicates if the axis is stopped (TRUE) or else in movement (FALSE), exclusively based on the height indicated by the encoder. |

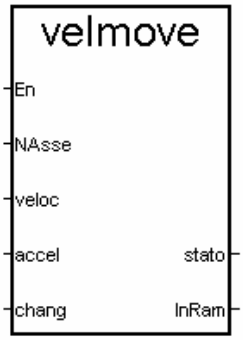
## Description:

This block lets it be known if at any time the axis is in movement or is stopped exclusively based on what is indicated by the encoder, ignoring the status of the movement and/or the function block that generated it. It is possible to know if the axis (even if disabled) is in movement due to external forces. The axis is considered stopped (StEnc = TRUE) if the encoder height is maintained within the tolerance indicated by the Toll parameter for a time indicated by the Time parameter.

**ARTECO SPA**

ISaGRAF® - axis control -

## Velmove




## Topics

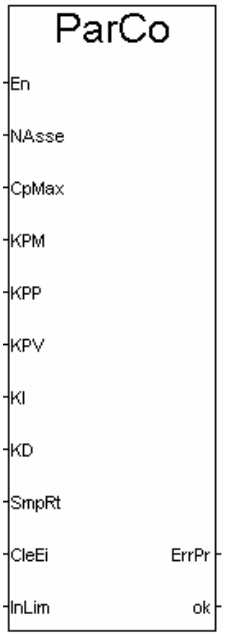
| | | |
|---|---|---|
| En | BOO | Starts movement in feedback velocity. TRUE starts the movement, FALSE stops it. It acts on the edges. |
| Naxis | INT | Indicates the axis on which movement at velocity is to be initiated. |
| veloc | INT | Axis velocity during movement. Expressed in mm/sec it must be assigned multiplied by 1,000.<br>Ex. 700mm/sec => 700,000. |
| accel | | Acceleration of the ramps. Expressed in mm/(sec*sec) it must be assigned multiplied by 1,000.<br>Ex. 1000mm/(sec*sec) => 1,000,000. |
| chang | BOO | Also allows the variation of the velocity in movement already initiated (by way of positive edge of En). If active the axis veloc and accel values follow those indicated in the parameters referred to above. The chang activation is influential only if En is active. To avoid the execution of complex calculations it is recommended to not unnecessarily leave chang active. |

| | | |
|---|---|---|
| state | INT | Indicates the internal status of the axis manager Whole value that represents the movement status by way of a code:<br>0 => -STOPPED<br>1 => STOPPED STANDBY<br>2 =>EXPECTED STOP<br>3 => ABRUPT STOP<br>4 =>EMERGENCY STOP<br>5 =>IN EMERGENCY STOP<br>6 =>CONTROLLED STOP<br>7 =>IN CONTROLLED STOP<br>8 => RELEASED<br>9 => ZERO VOLTS<br>10 => ZERO MACHINE<br>11 => MOVEMENT AT VELOCITY<br>12 => MOVEMENT IN JOG<br>13 => ERROR<br><br>PLEASE NOTE: Codes may be modified at the discretion of Arteco spa. |
| InRam | BOO | Indicates if the current theoretical velocity has not achieved the requested theoretical velocity. TRUE counts if the approach ramp is in the process of achieving required voltage. FALSE counts if the theoretical velocity is equal to that required.<br>For CAN piloted axes, this output indicates if the movement is still in progress and has the same meaning as the in_es output in the Xmov block. TRUE indicates that on this axis a movement is in progress. FALSE indicates a movement ended. |

## Description:

This block allows movements to be executed **at velocity**, connecting the current velocity with that requested. It is possible to modify said velocity (and the acceleration to achieve it) enabling the **chang** input.
The movement may be ended by way of the axis stop block. In this case if the En input is active it will be necessary to deactivate it in order to be able to carry out a new movement at voltage. If a movement at velocity is stopped by way of the axis stop it is possible to perform other movements (not at velocity) even without the deactivation of the En input.
This block is indicated for movements that bring the axis to the rabbet, since it does not generate a tracking error.
If it is necessary to move the axis with constant voltage and without control of the actual velocity utilise the VMove block.

## ParCo




## Topics

| | | |
|---|---|---|
| En | BOO | Input to activate the axis control with torque output. |
| NAxis | INT | Axis to be activated with torque control. |
| CpMax | INT | Maximum torque provided by the actuation delivered with a 10-volt command. Expressed in Nm multiplied by 1,000.<br>Ex. 2.5Nm => 2,500. |
| KPM | INT | Proportional constant utilised by the PID formula of the velocity loop, during a movement with torque control. |
| KPP | INT | Proportional constant utilised by the PID formula of the velocity loop, during adjustment with axis stopped, in the case of torque control. |
| KPV | INT | Proportional constant utilised for the velocity correction in the case of movement at velocity with torque control. |
| KI | INT | Integral constant applied by the PID for the velocity loop, in case of torque control. |
| KD | INT | Derivative constant applied by the PID for the velocity loop, in case of torque control. |
| SmpRt | INT | Divisor for the effect of the PID adjustment parameters in the velocity loop in case of torque control. |
| CleEi | BOO | Activates the zeroing out of the integral error, in the torque control velocity loop when the command changes sign (inversion of the direction of motion). |
| InLim | INT | Integral limit. Maximum value assumed by the single integral component in the velocity loop PID calculation in case of torque control. |

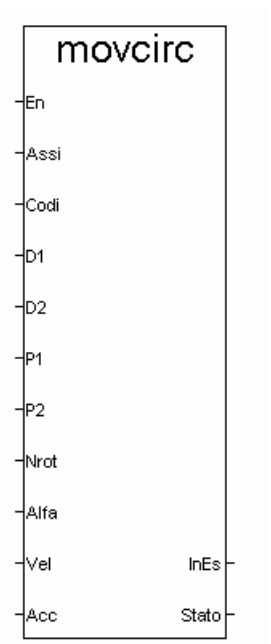
!

| | | |
|---|---|---|
| ErrPr | INT | Indicates successful torque control for the active axis if it is worth 0, while it indicates torque control not activated by error if it is worth –1. |
| Ok | BOO | The positive edge indicates the successful activation of the torque control for the specified axis. |

## Description:

This block allows the actuation to be provided with a torque reference instead of a velocity reference. This block, activated following ParAxis, introduces a velocity control loop that allows the SU data card to pilot actuations with torque input instead of with velocity input.

## MovCirc




# Topics

| | | |
|---|---|---|
| En | BOO | On the positive edge the circular interpolation movement requested is started based on the other data in the block. |
| Axes | INT | Mask containing the axes involved in the interpolation. Each bit indicates an axis to start up, where the least significant bit (zero) indicates the X-axis; the one bit indicates the Y-axis, up to the five bit which indicates the W axis. |
| CodI | INT | Interpolation code requested.<br>0 => Not utilised<br>1 => Launches movements in circular interpolation on two axes of known destination and an intermediate point provided in absolute heights.<br>2 => Launches movements in circular interpolation on two axes of known destination and an intermediate point provided in relative heights.<br>3 => Launches movements in circular interpolation on two axes with two points of the circumference known in absolute heights and the final angle to be achieved.<br>4 => Launches movements in circular interpolation on two axes with two points of the circumference known in relative heights and the final angle to be achieved.<br>5 => Launches movements in circular interpolation on two axes with coordinates of the centre known in absolute heights and the final angle to be achieved.<br>6 => Launches movements in circular interpolation on two axes with coordinates of the centre known in relative heights and the final angle to be achieved. |

| | | |
|---|---|---|
| D1 | INT | Height for the first axis to be interpolated. This parameter assumes a different meaning as a function of the Codl parameter (See the detailed description of the block). Expressed in mm multiplied by 1,000.<br>Ex. 100mm => 100,000. |
| D2 | INT | Height for the second axis to be interpolated. This parameter assumes a different meaning as a function of the Codl parameter (See the detailed description of the block). Expressed in mm multiplied by 1,000.<br>Ex. 100mm => 100,000. |
| P1 | INT | Height for the first axis to be interpolated. This parameter assumes a different meaning as a function of the Codl parameter (See the detailed description of the block). Expressed in mm multiplied by 1,000.<br>Ex. 100mm => 100,000. |
| P2 | INT | Height for the second axis to be interpolated. This parameter assumes a different meaning as a function of the Codl parameter (See the detailed description of the block). Expressed in mm multiplied by 1,000.<br>Ex. 100mm => 100,000. |
| Nrot | INT | Number of complete rotations to be carried out. (See the detailed description of the block) |
| Alfa | INT | Angle to be covered. This parameter assumes a different meaning as a function of the Codl parameter (See the detailed description of the block). Expressed in degrees multiplied by 1,000.<br>Ex. 90 degrees => 90,000. |
| Vel | INT | Velocity of the point interpolated along the trajectory during the movement. Expressed in mm/sec it must be assigned multiplied by 1,000.<br>Ex. 700mm/sec => 700,000. |
| Acc | INT | Acceleration of the point interpolated along the trajectory during the ramps. Expressed in mm/(sec*sec) it must be assigned multiplied by 1,000.<br>Ex. 1000mm/(sec*sec) => 1,000,000. |
| InEs | BOO | Indicates if the movement is still in progress. TRUE indicates movement in execution, FALSE movement ended. |
| State | INT | Indicates the internal status of the axis manager Whole value that represents the movement status by way of a code:<br>0 => -STOPPED<br>1 => STOPPED STANDBY<br>2 =>EXPECTED STOP<br>3 => ABRUPT STOP<br>4 =>EMERGENCY STOP<br>5 =>IN EMERGENCY STOP<br>6 =>CONTROLLED STOP<br>7 =>IN CONTROLLED STOP<br>8 => RELEASED<br>9 => ZERO VOLTS<br>10 => ZERO MACHINE<br>11 => MOVEMENT AT VELOCITY<br>12 => MOVEMENT IN JOG<br>13 => ERROR<br><br>PLEASE NOTE: Codes may be modified at the discretion of Arteco spa. |

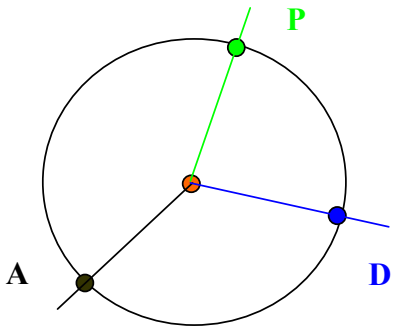
## Description:

This block allows circular interpolation movements to be executed on two axes providing the data necessary to determine the circular trajectory, the operational velocity and the acceleration to achieve it and to stop.
The D1 and P1 parameters that represent the coordinates “first axis interpolated”, the D2 and P2 parameters refer to the “second axis interpolated”.
The first axis interpolated is that with the lowest number present in the axes mask.

Ex. Axes = 3 (00000011 binary) -> (first axis = 0; second axis = 1)
Axes = 5 (00000101 binary) -> (first axis = 0; second axis = 2)
Axes = 10 (00001010 binary) -> (first axis = 1; second axis = 3)

The Codl code allows the type of data necessary to generate the trajectory to be chosen and it may assume the following values:

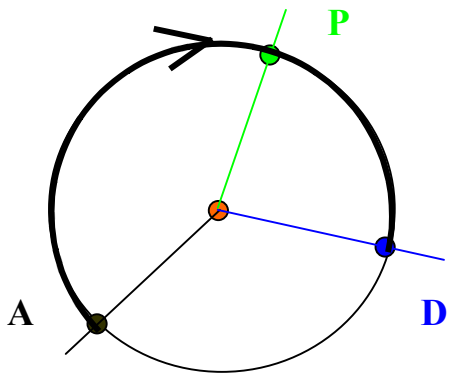
0 => Not utilised
1 => A circumference arc is followed beginning with the current position, up to the destination (of which the D1 and D2 parameters represent the coordinates) and passing through a further point P (of which the P1 and P2 parameters represent the coordinates). All of the coordinates are absolute positions. The Nrot parameter represents the number of complete circumferences that are covered before stopping at the destination point. The alfa parameter has no meaning.
If Nrot is worth 0, a circumference arc of <360 degrees is covered, which includes the waypoint P.
If Nrot > 0 i Nrot complete revolutions are covered plus an arc that includes the point P;
If Nrot is worth -1, a circumference arc of <360 degrees is covered, which does not include the point P.
If Nrot <-1 (Nrot - 1) complete revolutions are covered plus a circumference arc that does not include point P.
2 => similar modality to 1 but in this case all of the heights (D1, D2, P1, P2) are to be considered relative as compared to the current position.



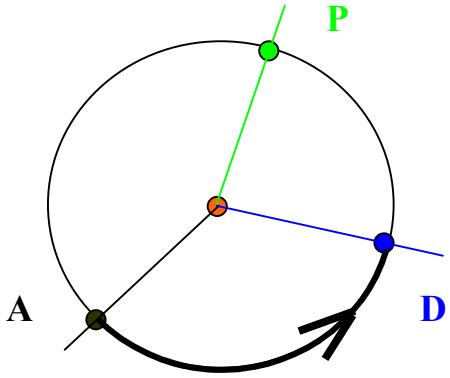

**A** = current position

**D** = destination position

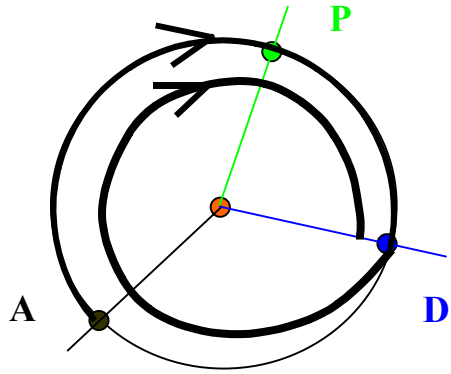
**P** = waypoint



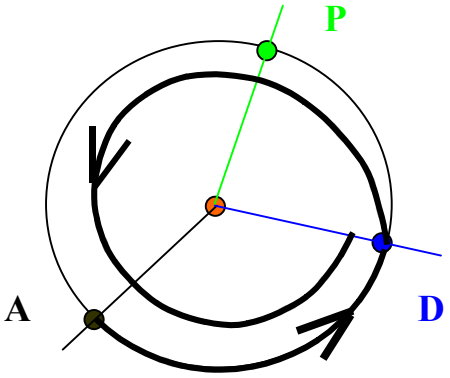

Nrot = 0




Nrot = -1




Nrot = 1




Nrot = -2

3 => The circumference is identified by its current position, from point D (of the D1 and D2 coordinates) and from point P (of the P1 and P2 coordinates). All of the coordinates are absolute positions.
The interpolated point moves on said circumference and completes a total arc equal to the number of complete rotations (Nrot parameter) plus the angle set in the alfa parameter, which can be positive or negative. If Nrot>0 the direction of travel of the circumference is such that it will first encounter point P and afterwards point D, vice versa if Nrot<0.

4 => Similar modality to 3 but in this case all of the heights (D1, D2, P1, P2) are to be considered relative as compared to the current position.

5 => The circumference is identified by its current position and by the coordinates of the centre (parameters D1 and D2). The interpolated point moves on said circumference and completes a total arc equal to the number of complete rotations (Nrot parameter) plus the angle set in the alfa parameter, which can be positive or negative. If Nrot>0 the direction of travel of the circumference is positive (that is, representing the axis in abscissa with a lower number and in ordinate that with the higher number, the point moves in a counter-clockwise direction), vice versa if Nrot<0. The P1 and P2 parameters are not utilised.

6 => similar modality to 5 but in this case all of the heights (D1, D2,) are to be considered relative as compared to the current position.

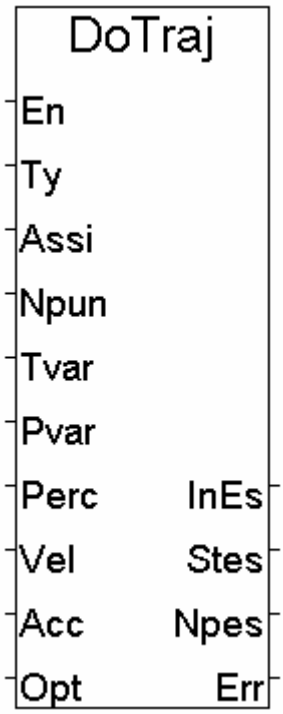
The final destination is influenced by the origins management. See GstOrg block.

In any modality if the (P1, P2, D1, D2) passed parameters do not allow a circumference to be identified, an error is generated. This may occur if the destination point coincides with the starting point or with the waypoint, if the waypoint coincides with the destination, or if the three points lie on a straight line.
In any case to improve the precision of the trajectory it is recommended that the points be set far from one another. In the case of points that are very close to one another the approximation error could generate a less precise trajectory.

**ARTECO SPA**

ISaGRAF® - axis control -

## DoTraj




## Topics

| | | |
|---|---|---|
| En | BOO | On the positive edge, starts the generation of a trajectory drawing the points from an array of variables. |
| Ty | INT | Type of interpolation required.<br>0 – Not available<br>1 – Circular spline on two axes<br>2 – NURBS type splines on 2 or more axes |
| Axes | INT | Mask containing the axes involved in the interpolation. Each bit indicates an axis to start up, where the least significant bit (zero) indicates the X-axis; the one bit indicates the Y-axis, up to the five bit which indicates the W axis. |
| Npun | INT | Number of points to interpolate. |
| Tvar | INT | Indicates the type of variable from which the interpolation data is extracted. 0 indicates an ISaGRAF variable. 1 indicates virtual variable. |
| Pvar | INT | Indicates the number of the first variable contained in the data array for the execution of the trajectory series. In the case of an ISaGRAF® variable, it indicates the Virtual Address, in the case of a Virtual Variable it indicates the number. |
| Perc | INT | Input not utilised. |

| | | |
|---|---|---|
| Vel | INT | Velocity of the point interpolated along the trajectory during the movement.<br>Expressed in mm/sec it must be assigned multiplied by 1,000.<br>Ex. 700mm/sec => 700,000. |
| Acc | INT | Acceleration of the point interpolated along the trajectory during the ramps.<br>Expressed in mm/(sec*sec) it must be assigned multiplied by 1,000.<br>Ex. 1000mm/(sec*sec) => 1,000,000. |
| Opt | INT | Additional input the meaning of which depends on the type of junction performed. |
| InEs | BOO | TRUE indicates that the trajectory is in execution, FALSE indicates that the movement is stopped. |
| Stes | INT | Movement execution status<br>0 = Ended correctly<br>1 = Data preparation in progress<br>2 = Execution of a linear segment<br>3 = Execution of a circular segment (only for Ty=1)<br>4 = Ended with error |
| Npes | INT | Number of the point in execution |
| Err | INT | Number of the error present<br>0 = No error<br>1 = Number of the axes at fault<br>2 = Type of interpolation not supported<br>3 = Error in the velocity or acceleration value<br>4 = Axes not ready to carry out motion (height not valid, request zero axis, or another movement is in progress)<br>5 = Limit stop hardware or software<br>6 = Faulty height for rotating axis requested<br>7 = Not possible to execute the trajectory with the spline radius values requested<br>8 = Radius too large, spline trajectory does not exist<br>9 = Error in calculations for the determination of the trajectory<br>10 = Error in the block input parameters<br>11 =Internal structure data transfer error<br>12 = Error in the number of variables that contain interpolation data<br>13 = Error in the dualport<br>14 = Trajectory execution error (excessive movement, obstacle encountered)<br>15 = Error in the series of points to be interpolated (ex: the line that unites the points has a change of direction of 180 degrees) |

# Description:

This block permits the execution of a trajectory beginning with a series of heights memorised in the application variables.
Npun indicates the number of points to be interpolated.
Tvar indicates the type of variables containing the data and Pvar indicates the number of the first variable.
The number of variables to be provided depends on the number of points and on the type of interpolation.
Vel and Acc represent velocity and acceleration of the material point that moves along the trajectory.
Ty indicates the type of interpolation required for the generation of the trajectory.
The InEs output indicates if the trajectory is in progress. It is TRUE during the movement and FALSE at the end.
The Stes output indicates the status of execution of the movement.
The Npes output indicates the progress status of the motion: in particular it indicates the number of the point toward which the trajectory evolves.
The Err output returns the codes of eventual errors that have occurred.

## Types of Interpolation

0) **Not available**.

1) **Circular splines on two axes**. Given a series of points, a trajectory is made up of linear segments joined by circumference arcs of which the curvature radius is provided. The trajectory begins from the current position at the moment of the enabling of the block and moves in the direction indicated by the first point provided. Before reaching this point the circular spline is executed. This permits the moveable point to be found on the line linking the first point to the second and to continue in a straight line in the direction of the second point up to the next circular spline. The trajectory evolves in this way until it ends in correspondence to the last point. The trajectory does not pass through the intermediate points but comes as close to them as the curvature radii are small. It is possible to specify a different curvature radius for each point. In this way trajectories that are more or less “soft” may be obtained. The moveable point moves on the trajectory beginning from the initial position with acceleration set by parameter until reaching the velocity required, then afterwards the segment at a constant velocity (always consider the velocity of the moveable point along the trajectory) initiates the deceleration phase until it stops at the final point.

   The variables of the application that contain data for the generation of the trajectory must be structured in this manner:

   Var 1) Total number of variables containing data
   Var 2) Height first axis first point (in mm * 1,000)
   Var 3) Height second axis first point (in mm * 1,000)
   Var 4) Length of the spline radius for the first point (in mm * 1,000)

   Var 5) Height first axis second point (in mm * 1,000)
   Var 6) Height second axis second point (in mm * 1,000)
   Var 7) Length of the spline radius for the second point (in mm * 1,000)

……..

   Var ..) Height first axis last point (in mm * 1,000)
   Var ..) Height second axis last point (in mm * 1,000)
   Var ..) Length of the spline radius for the last point

   The maximum number of points to interpolate is 21. It is possible to contemporaneously execute two Dotraj blocks (which obviously must involve two different axes).
   In the case in which one point comes out beyond the limit stop software, the trajectory is not generated and the block will return an error code. In the same manner, if it is not possible to generate a trajectory with the data provided (radius too large, inconsistent values) an error is signalled.

If two or more consecutive points lie on the same straight line, the circular segment will not be executed and the trajectory continues linearly up to the next circular spline. If two or more consecutive points coincide it passes on to consider the next point.
During the movement the StEs output indicates if a linear segment is being executed or a circular spline.
If during a trajectory an error is verified (excessive movement, activation of a limit stop) the trajectory is ended.
If during a trajectory a stop axis is requested, the moveable point continues along the trajectory generated slowing down until it stops. The deceleration used is equal to the acceleration set in the block.

The OPT input is not used.

2) **NURBS (Non Uniform Rational Bilinear Spline) type trajectory.** This type of spline allows the interpolation of up to 6 axes. Given a series of points, the trajectory identified by the points themselves is generated, by the values of the nodes (K) and the "weights" (W) that are provided as parameters. The trajectory begins from the current position at the moment of the enabling of the block and moves in the direction indicated by the first point provided. The trajectory evolves in such a manner as to pass in proximity to successive points until it ends in correspondence to the final point.
Below are found some fundamental notions that will permit an intuitive understanding of the meaning of the nodes and the weights, for a more thorough knowledge refer to texts of mathematical literature.

- The K nodes must have non-decreasing positive values.
- The W weights must have positive values.
- Ideally one might think that a fictitious coordinate exists (that we will call U), which in its initial point is worth 0 and that increases progressively, assuming the value of each node in proximity to each point, up to the final point with which the value of the final node coincides.
- Each point of the trajectory is "influenced" by 3 of the passed points as parameters and by the corresponding weights and nodes.
  Each one of these points contributes to, in a proportional manner, the distance in terms of U from the point of the trajectory. In particular, varying the value of a K, the segment of the trajectory preceding the point in question is changed. (Ex. If the value of node 5 is varied, the curve is changed in the segment that goes approximately from point 3 to point 5).
- The trajectory gets closer to a point as much as the weight of that point is greater in relation to the weight of the other points.
- Inserting 2 equal node values, the trajectory passes exactly in the point preceding the two equal values but the velocity of the axes undergoes an abrupt change and the trajectory comes out "rough edged". (Ex. if Node 3 and node 4 have the same value, the trajectory passes exactly over point 2, but to do so in its correspondence it undergoes an abrupt braking and then afterwards an abrupt acceleration).
- A weight value is not significant in and of itself but in relation to those of the other points. In practice, multiplying all of the weights by an equal value, the trajectory doesn't change.
- As a first approach it is suggested that all of the weights be set at 1 and the increasing nodes with an equal step (100, 200, 300...).

Appropriately varying the weights and the nodes, more or less "soft " trajectories may be obtained. The moveable point moves on the trajectory beginning from the initial position with acceleration set by parameter until reaching the velocity required, then after the segment at a constant velocity (the velocity of the moveable point is still considered to be along the trajectory) initiates the deceleration phase until it stops at the final point.

The variables of the application that contain data for the generation of the trajectory must be structured in this manner:

Var 1) Total number of variables containing data

Var 2) Height first axis first point (in mm * 1,000)
Var 3) Height second axis first point (in mm * 1,000)
……
Var ..) Height last axis first point (in mm * 1,000)
Var ..) Node first point
Var ..) Weight first point
Var ..) Height first axis second point (in mm * 1,000)
Var ..) Height second axis second point (in mm * 1,000)
……
Var ..) Height last axis second point (in mm * 1,000)
Var ..) Node second point
Var ..) Weight second point
……
……
Var ..) Height first axis last point (in mm * 1,000)
Var ..) Height second axis last point (in mm * 1,000)
……

Var ..) Height last axis last point (in mm * 1,000)
Var ..) Node last point
Var ..) Weight last point

The number of variables necessary depends upon the number of axes according to the equation:
NVAR = (NUM_AXES + 2) * NUMPOINTS +1.

The maximum number of points depends on the number of axes involved.

| NUM AXES | N_MAX_POINTS | NVAR = (NUM_AXES + 2) * N_POINTS +1. | NVAR(N_MAX_POINTS) |
|---|---|---|---|
| | | | |
| 2 | 31 | 4* N_POINTS +1. | 125 |
| 3 | 25 | 5* N_POINTS +1. | 126 |
| 4 | 21 | 6* N_POINTS +1. | 127 |
| 5 | 18 | 7* N_POINTS +1. | 127 |
| 6 | 15 | 8* N_POINTS +1. | 121 |

It is possible to contemporaneously execute two NURBS splines (which obviously must involve two different axes).
In the case in which one point comes out beyond the limit stop software, the trajectory is not generated and the block will return an error code.
During the movement the StEs output indicates the status of the movement.
If during a trajectory an error is verified (excessive movement, activation of a limit stop) the trajectory is ended.
If during a trajectory a stop axis is requested, the moveable point continues along the trajectory generated slowing down until it stops. The deceleration used is equal to the acceleration set in the block.

During the movement the velocity may undergo small variations. This is due to the fact that, so as to not overweight the calculations too much, the velocity value is recalculated periodically between one segment and another. It is possible to vary the number of times that it is recalculated varying the value of the OPT parameter by a minimum of 2 and a maximum of 48. The values not accepted correspond to the default value, which is 8.
If more limited velocity variations are desired it is possible to increase said parameter.
It is suggested to use even numbers to the 2nd power, which permit faster operations. (8, 16, 32).

# Appendix 1 – Algorithms available

**Algorithm N. 5: Electronic Cam.**

**Algorithm N. 6: Boolean Cam.**

**Algorithm N. 7: Generation of an optimised trajectory given a series of points.**

**Algorithm N. 8: Generation of an optimised trajectory given a series of points and realised by two axes that move two hinged mechanical arms.**

**Algorithm N. 9: self-learning of a series of points.**

**Algorithm N. 10: reproduction of a movement given a series of points.**

**Algorithm N. 11: Rotatory.**

# Appendix 2 - Axis Manager Errors

| | | |
|---|---|---|
| 0 | No Error | No errors occurred since turn on |
| 1 | Emergency Input | The activation of the emergency input causes the immediate stop of the axis by way of the disabling and the zeroing out of all of the user outputs. This error will continue to reappear until the emergency input is deactivated, which re-enables the outputs and makes the axis manager available once again. |
| 2 | Calculation Error | Trajectory calculation error. Contact Arteco spa. |
| 3 | Excessive movement error | Accumulated error during the execution of a trajectory (Like the difference between real and theoretical position) it has exceeded the value set in the ParAxis block and indicated with **ErrEc.** This is generated even if at the end of the trajectory the error compared to the destination height is greater than the TP parameter (positioning tolerance). |
| 4 | Internal Error | Contact Arteco spa. |
| 5 | Minimum Limit Stop | Indicates the exceeding of the height of the minimum limit stop set in ParAxis such as FCMin or else the activation of the apposite hardware input. |
| 6 | Maximum Limit Stop | Indicates the exceeding of the height of the maximum limit stop set in ParAxis such as FCMax or else the activation of the specific hardware input. |
| 7 | Zero sensor already active | If zeroing out is requested of the zero sensor in the ParAxis parameter (TyAzz) and of the activation of the procedure, the aforementioned sensor is already active, this error is generated. |
| 8 | Destination change fault | Linked to the ChgDst block NewDst parameter. This is generated if a destination change is requested indicating a new destination in the NewDst parameter already exceeded. The axis is stopped immediately. |
| 9 | Velocity change fault | Linked to the ChgVel block. |
| 10 | Parameterisation fault | Indicates an error during the parameterisation of the axis. It may be signalled after the call to the ParAxis block. The ErrPr output of this latter provides indications on the cause of the error. See ZeroAxis block. |
| 11 | Excessive Axis Offset | Indicates that the axis offset, detected by the TarOffs block, is greater than 1% of the maximum DAC value. It is necessary to verify the correctness of the connections and eventually adjust the actuation offset. Contact the Arteco spa Assistance centre. |
| 12 | ZeroAxis data block fault | The ZeroAxis block has been called with velocity and/or acceleration parameters equal to zero. |
| 13 | ReadPos block error | The position returned by the Posiz output of the ReadPos block has been limited, in that it is above the maximum value viewable in a whole of 32 bits. |
| 14 | Jog data block fault | The Jog block has been called with velocity and/or acceleration parameters equal to zero. |

| | | |
|---|---|---|
| 15 | Xmove data block fault | The Xmove block has been called with velocity and/or acceleration parameters equal to zero. |
| 16 | XmovOpen data block fault | The XmovOpen block has been called with velocity and/or acceleration parameters equal to zero. |
| 17 | CaptInt data block fault | The CaptInt block has been called with the Ning parameter of 17 or 18 or the Fronte parameter different by 1, 2, or 3. |
| 18 | Height corrupted. | Upon turn on, the height present in memory turns out to be corrupted. Movements are inhibited and it is necessary to perform a machine zero out. |
| 19 | Acc/Dec too high | This is generated during certain operations if the trajectory generator requests an acceleration/deceleration greater than the emergency deceleration indicated in the parameters block. |
| 20 | Calibration offset error | This is generated following an offset calibration request with the axis not stopped. |
| 21 | Rmove data block fault | The Rmove block has been called with velocity and/or acceleration parameters equal to zero. |
| 22 | GestOrg data block fault | This is generated if the axis number is not admitted, the origin number is above the maximum permitted or if the type of operation has not been anticipated. See GstOrg block. |
| 23 | ChkObst data block fault | This block is called with faulty parameters or else the movement in progress does not permit obstacle management. |
| 24 | XmovObst data block fault | The XmovObst block has been called with velocity and/or acceleration parameters equal to zero, or else the parameters relative to the management of the obstacle are faulty. |
| 25 | Zero Axis not performed | This is generated if the Zstar of the ParAxis block is equal to 1 (zero axis obligatory at turn on) and there is an attempt to perform a movement without having performed the zero out following turn on. |
| 26-31 | DSP internal error (only SU250) | The errors from 26 to 31 indicate DSP technical problems. Contact Arteco providing the error number. |
| 32 | TarOffs data block fault | This is generated if the axis number indicated is not correct. |
| 33 | Vmove data block fault | This is generated if the axis number indicated is not correct or else if the velocity or acceleration parameters are equal to zero. |
| 34 | ReadErr data block fault | This is generated if the axis number indicated is not correct. |
| 35 | SetQuota data block fault | This is generated if the axis number indicated is not correct. |
| 36 | StopAxis data block fault | This is generated if the axis number indicated is not correct. |
| 37 | AxiFree data block fault | This is generated if the axis number indicated is not correct. |
| 38 | AxiLock data block fault | This is generated if the axis number indicated is not correct. |
| 39 | PrgXMov data block fault | This is generated if the axis number indicated is not correct or else if the velocity or acceleration parameters are less than or equal to zero. |
| 40 | MoveInt data block fault | This is generated if the axis number indicated is not correct or else if the velocity or acceleration parameters are less than or equal to zero, if the type of interpolation has not been foreseen or if the number of axes does not conform to the type of interpolation required. |
| 41 | MovInt block error | This is generated if the destination requested corresponds to the current position. |
| 42 | End of trajectory error | This is generated at the end of the trajectory if the position achieved is outside of the tolerance admitted. |
| 43 | Axis_rot data block fault | This is generated if the axis number indicated is not correct or else if the type of operation is not admitted. |

| | | |
|---|---|---|
| 44 | Number of rotations on rotating axis too high | In the case of the rotating axis this error is generated if the number of rotations exceeds the maximum number anticipated. |
| 45 | AxisEl data block fault | This is generated if the axis number indicated or the parameters are not correct. This is generated even if problems are verified in the calculations necessary for the start up of the electrical axis. |
| 46 | TstEnc data block fault | This is generated if the axis number indicated is not correct. |
| 47 | GoPrgMov Block fault | This is generated if the axes of which are requested start up have not been previously programmed. |
| 48 | VelVmove data block error | This is generated if the number of the axis indicated is faulty or if the acceleration set is equal to or less than zero. |
| 49 | GstArray block error | This is generated if one of the parameters is not admitted, if the variables array contains errors (ex. number of points equal to zero) or if an error is verified during the copy of the variables array to the internal memory. |
| 50 | GstArray execution error | This is generated if an error occurs during the generation of the trajectory) in one of any of the points of which it is made up), with stop of the execution of the trajectory itself. |
| 51 | Dualport Error | This is generated if an error is verified in the communication between the PLC and the axis expansion. Contact Arteco spa. |
| 52 | Rotating axis error | This is generated if the axis is rotating and a height incompatible with the circumference of the axis must be set. |
| 53 | Paron-off block error | This is generated if the axis on-off parameterisation data are faulty: faulty axis number, mmimp conversion factor or high speed activation delay not included in the validity range, advance or slowdown height less than zero, fcmin greater than fcmax or with values such that the running of the axis bottoms out the counter that detects the heights. (See manual axes on-off). |
| 54 | Xmvon-off block error | This is generated if the on-off axis movement block parameters are faulty: axis number faulty, axis not initialised, the destination set is outside of the limit stops range, the destination is inside of the advance set. (See manual axes on-off). |
| 55 | Sqon-off block error | This is generated if the block parameters are faulty: axis number faulty, axis not initialised, the destination set is outside of the limit stops range. (See manual axes on-off). |
| 56 | Rdpnoff block error | This is generated if the axis number indicated is faulty. (See manual axes on-off). |
| 57 | Axis on-off encoder error | This is generated if an error is verified in the axis on-off encoder count. (See manual axes on-off). |
| 58 | Axis on-off overflow error | This is generated if an error is verified in the axis overflow on-off encoder count. (See manual axes on-off). |
| 59 | Gstflash block error | This is generated if the operation number is not correct. (See customisation manual) |
| 60 | Erron-off block error | This is generated if the axis number indicated is faulty. (See manual axes on-off). |
| 61 | Actuation Ok error | This occurs when the 2 enabling modality has been set (actuation ok) and there is no corresponding input. |
| 62 | Stepper motor error | This is generated when the parameterisation data of the stepper motor axis are faulty. |
| 63 | Bus encoder error | This is generated if disturbance in the digital filters of the bus encoders is detected. |

| | | |
|---|---|---|
| 64 | Bus encoder marker error | This is generated if errors in the markers of the bus encoders is detected. |
| 65 | Excessive error due to obstacle | This is generated if the axis turns out to be stopped with an excessive error during a trajectory. It can indicate an encounter with an obstacle. |
| 66 | Excessive error with axis stopped | This is generated if the axis does not remain within the TP positioning tolerance when the axis is stopped. |
| 67 | Axis not initialised error | This is generated if a block for an axis that has not been initialised has been called. |
| 68 | Torque parameters error | This is generated if the ParCo block parameters are faulty because out of range. |
| 69 | Torque enabling error | This is generated if the torque control for an axis has been enabled without having first enabled it by way of the ParAxis block. |
| 70 | SetAlgo block error | This is generated if the SetAlgo block parameters are faulty because they are out of range or because an impossible operation is requested (ex. cancellation of an algorithm that is not present). |
| 71 | GoAlgo block error | This is generated if the GoAlgo block parameters are faulty because they are out of range or because an impossible operation is requested (ex. execution of an algorithm that is not parameterised). |
| 72 | Error: algorithm cannot be parameterised | This occurs when there is the attempt to parameterise an algorithm that pilots an axis already commanded by another algorithm or else the maximum number of algorithms has already been parameterised. |
| 73 | Error in the algorithm runtime calculation | This occurs when during the execution of the algorithm, faults or anomalies occur in the trajectory calculations. |
| 74 | Error in the preliminary algorithm calculation | This occurs when anomalies are present in the variables (virtual or ISaGRAF) used for an algorithm in a preliminary calculation. (Ex: values out of range). |
| 75 | LimPos block error | Indicates the non activation of the LimPos block due to an axis not stopped or else an incongruence between the Min and Max parameters and/or between the Min and Max parameters and the minimum and maximum limit stops set in the ParAxis. |
| 76 | Error LimPos + AsseEl | Request activation of electrical axis with incongruence between the maximum or minimum limit stops induced on the master and the limitation imposed by LimPos on the master itself. |
| 77 | MovCirc block error | This is generated if the MovCirc block parameters are faulty because out of range or because two points are coincident and it is not possible to identify a circumference. |
| 78 | Aligned points in MovCirc block error | A circular interpolation with three aligned points has been requested. The resulting trajectory could be extremely imprecise. It is suggested that the linear interpolation block be used (MovInt). |
| 79 | RampaS block error | This is generated if the RampaS block parameters are faulty because they are out of range or because an inadmissible operation is requested. |
| 80 | Jerk error larger than maximum permitted jerk. | This is generated if the jerk necessary for a change in velocity or destination is greater than the maximum value permitted, set by way of the JMax parameter of the RampaS block. |
| 81 | Axis on-off power supply error | This is generated when the axis on-off is not correctly supplied with power. |
| 82 | Axis on-off fuse 1 Error | This is generated when the first axis on-off fuse protection trips. |
| 83 | Axis on-off fuse 2 Error | This is generated when the second axis on-off fuse protection trips. |
| 84 | Axis on-off driver error | This is generated when the second axis on-off driver protection trips. |

| | | |
|---|---|---|
| 85 | Inverter block error | This is generated if the inverter block parameters are faulty because out of range. |
| 86 | Inverter activation block error | This is generated when it is not possible to activate the inverter block due to the axis not being stopped. |
| 87 | DoTraj block error | This is generated if the DoTraj block parameters are faulty because out of range. |
| 88 | Axis disabling error | This is generated when the disable command parameters are faulty. |
| 89 | Gstser block error | This is generated if the Gstser block parameters are faulty. (See customisation manual). |
| 90 | Readser block error | This is generated if the Readser block parameters are faulty. (See customisation manual). |
| 91 | Writeser block error | This is generated if the Writeser block parameters are faulty. (See customisation manual). |
| 92 | SUMD actuation in malfunction error | Only for SUMD data card (See SUMD manual). |
| 93 | SUMD actuation not ready error | Only for SUMD data card (See SUMD manual). |
| 94 | SUMD actuation in alarm error | Only for SUMD data card (See SUMD manual). |
| 95 | Faulty SUMD password error | Only for SUMD data card (See SUMD manual). |
| 96 | ChgTAxis block error | Generated if the hardware version, the input range, or the emergency status are not correct. |
| 97 | Rotary algorithm error | Only for Rotary algorithm. Two successive inputs at too near a distance have been verified. |

# Appendix 3 - Target Errors

| | | |
|---|---|---|
| 101 | Application lost | Signalled if application is corrupted. It is necessary to reload it. |
| 102 | Emergency Input | Signals the moment of the emergency input deactivation. |
| 103 | Mathematical error | Signals a mathematical error in the floating-point management. |
| 104 | Last err ReadErr block | Brings up the last error signalled in the ReadErr block. The errors already signalled with errors 101, 102, 103 are excluded. This notification comes about exclusively if NewEr has been previously reset. See ReadErr block. |
| 105 | Reset hardware | This signals the nullification of the application due to the presence of the connector for the data card reset, at the turn on of the data card. |
| 106 | Application removal failed. | By way of the workbench there was an attempt to remove a compiled application, and which is therefore not removable. |
| 107 | Axis not capable of being initialised. | Attempt to initialise the axis failed because the axis was not anticipated in the firmware. Comes about only in the ISaGRAF® PLC version. Contact Arteco spa. |
| 108 | 5 DSP in malfunction (only SU250). | DSP malfunction detected.<br>Contact Arteco spa. |
| 109 | Flash card Error. | Error in the management of an external flash card has been verified. |
| 110 | Axis on-off error | An error relative to the on-off axis has been verified (see axis on-off manual). |
| 111 | Stepper Axis on-off error | An error relative to the stepper axes has been verified. (See stepper axes manual). |
| 112 | Application allocation error. | Indicates an allocation problem of the compiled application by way of an Arteco compiler. Contact Arteco spa. |
| 113 | Warning source + compiled application | Indicates the presence of a compiled application and source file present in the target simultaneously. Internal reallocation operation could have removed the source file. Contact Arteco spa. |
| 114 | Last err ReadErrMD block | Only for SUMD data card.<br>Brings up the last error signalled in the ReadErrMD block. This notification comes about exclusively if NewEr has been previously reset. See RdErrMD block. |
| 115 | Overflow/undeflow error | This is generated following a conversion of the type for one variable of the application that causes an overflow (code 1) or an underflow (code 2). |
| 116 | Unforeseen area error. | Request allocation in unforeseen memory area.<br>Contact Arteco spa. |
| 117 | ALTERA programming error | Only for Motion Kit data card<br>ALTERA malfunction detected.<br>Contact Arteco spa. |

| | | |
|---|---|---|
| 129-235 | Error driver | Internal Error. Contact Arteco spa, Indicating the precise message shown in the workbench |
| 236 | CAN network error | CAN node emergency See "*CanOpen*" block "*GestNet*" manual. |
| 237 | Internal Error | Internal Error Contact Arteco spa. |
| 301-428 | CAN axis modules Error | CAN axis module Error See "*CanOpen Analogue Axes*" manual. |

# Appendix 4 - LED Management

***WATCH-DOG LEDs (DL50)***

| | |
|---|---|
| Flashing LEDs | Indicates the correct operation of the data cards. The frequency of the flashing is faster when an application is in execution. |
| Fast flashing LED with interruption with LED lit. | This indicates the blockage of the data card due to a mathematical error in the floating-point management. It is necessary to turn off the machine and turn it back on again. Upon turning the machine back on, the error target 103 will appear.<br>Contact Arteco spa. |
| Fast flashing LED with interruption with LED not lit. | This indicates the blockage of the data card due to an internal error. It is necessary to turn off the machine and turn it back on again. Upon turning the machine back on, the application will not be present and the error target 129 to 235 will appear.<br>Contact Arteco spa. |
| LED flashing very fast, without interruptions | This indicates blockage of the data card due to a significant temporary drop in power supply voltage. The data card must be turned off and then turned back on again. |

***RUN LED (DL51)***

| | |
|---|---|
| LED off | Application not present<br>If the watch-dog LED flashes rapidly it could indicate a malfunction of the data card. Contact Arteco spa. |
| LED flashing in synchrony with the watch-dog LED | Application present |
| LED flashing with a frequency that is higher than the *watch-dog* LED | Application loading from the serial port |

# Appendix 5 – Arteco Compiler for C167

The application developed in the ISaGRAF® environment may be expressly compiled for the processor present inside of the SU210, SU212, SU250 or CND51 PLC. This permits a notable reduction in the PLC cycle time in a manner dependent upon the type of language chosen and by the percentage of function blocks present as compared to the total application.
For compiler installation, contact Arteco SpA. How to use it will be illustrated below; that is, how to compile and load the application generated.

Phase 1 – Compilation Options

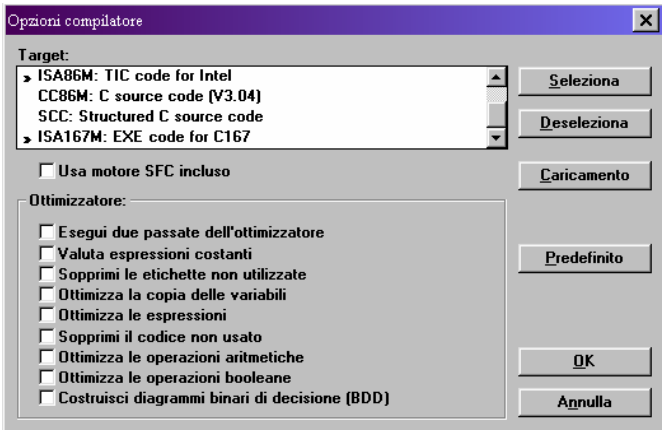



Fig. 1

In the project selected, choose “Compile” and “Compiler Options”. The window found above will appear. For utilisation of the Arteco compiler for C167 select and highlight: **ISA167M: EXE code for C167**.
At the moment of compilation, by way of the“ Compile Application” command, what is found below will be visualised, so as to indicate the actual compilation by way of the Compiler for the C167.

```
Verifica
Code generation : code for SABC167 : Release <0> Version <1> Revision <1>
Nessun errore rilevato
```

(Verification
Code generation…, etc. (in English)
No error detected)
At the end of the compilation, the application may be downloaded into the target.

Phase 2 – Download

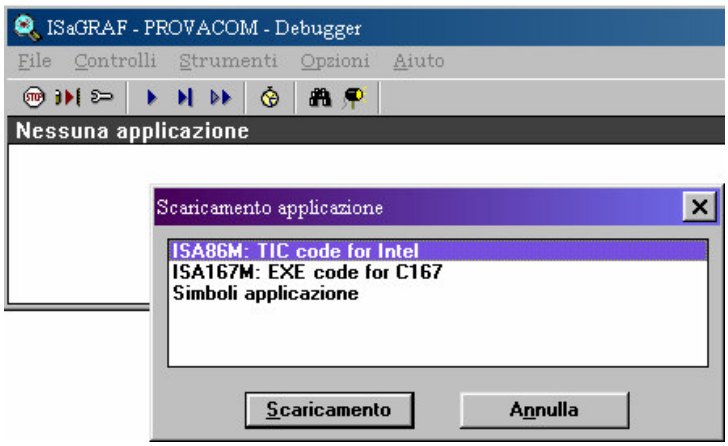



Fig. 2

After connection to the target by way of “Debug” proceed to the Download. Pushing the appropriate command button, a window will open in which the item “**ISA86M: TIC code for Intel**” and “**ISA167M: EXE code for C167**” will appear. The first sends the compiled application to the target in a traditional format (interpreted code), while the second sends the application to the target compiled for the SU data card.

All of the debug functions through Workbench continue to be usable even with the application compiled, with exception of the breakpoint in SFC environment, which is disabled.

The size of the compiled application program is significantly larger than the interpreted program (TIC Code); therefore the download time of the application through workbench turns out to be notably longer.

It is possible to save the compiled application“ EXE Code” onto the SIM and restore it with the same modalities as the “TIC Code” application. The restoring from the SIM has an unvaried duration, independent of the application and/or of the executable type, (TIC o EXE).

The format “EXE Code for C167” is admitted exclusively on data cards that are set up for them. The versions that allow this type of application are the 49 of 12 March 2001 and the versions following 59, included.
At the moment of the download of an EXE application, by way of workbench, the target version is checked and, if it does not foresee the object format, the option will not be displayed in the selection mask (See Fig. 2)
The loading of an EXE application by way of a SIM on a target that does not allow it could cause the malfunction of the SU data card.



ARTECO MOTION TECH S.p.A.
Via Gentili, 22
48018 Faenza RA
ITALY
Phone: +39 0546 645777
Fax: +39 0546 645750
Email: info@arteco-cnc.com
www.arteco-cnc.com