Tpaedi32

Version 1.7.8

Program editor

Tecnologie e Prodotti per l'Automazione

Table of Contents

1 Updating

1.1 Release 1.7.8

New functionalities

- The technological assignment by default, set for the setup workings in the page Customize->Technology, are not applied to the blade setup any more.
- Dialog box revision for interface functioning with Cyrillic messages

Solved problems

- In the <r> variables setting window in the Subroutine call codes the programming of spaces in the string variables was managed incorrectly.
- In the setting window of the <r> variables of a program, capital letters were not managed in the string variables.
- In the installation procedure an error of the Langs.exe appeared. Sometimes, as a consequence of this error it could not be possible to change the interface language.

1.2 Release 1.7.7

New functionalities

- Now the sequence list includes also the construct workings
- Importing a DXF file: an import option has been added to assign the workpiece thickness by default
- Parametric programming: arguments of *subang0, submir0, substr0 variables* has been added,
- *STOOL* Code: now it also applied a complex code, which cannot be exploded (previous features: it was not considered)

Solved problems

- Choices in list in *Custom sections*: string management in case of use of special characters has been regulated
- Application of *technological filters*: in case of assigned filter for a setup code, no profile elements assigned to the setup were displayed.
- Cases of *xz or xyz mirror* arc applied in a macro or subroutine development has been regulated

1.3 Release 1.7.6

New functionalities

- Program saving: if the program seems to be modified with respect to the read version, a message appears with request of confirmation for saving execution.
- Parametric programming: added prototype into the function *pown.*

Solved problems

- Some cases of fictive and automatic faces.
- Multiple setups: sequence assignments was not propagated

1.4 Release 1.7.5

New functionalities

- A new program can be initialised also with assigned workings in the (default or selected) prototype file.
- In the items of Customisation of the application program the option ***Manage a model file*** in ***Creating a program*** has been removed. At any rate this option is managed as enabled.

Solved problems

- Custom Sections: some cases of value truncation assigned to the file selection field in case of

existing fields
- In the configuration window of the DXF file import the child windows (concerning e.g. working or parameter selection) are now displayed.

1.5 Release 1.7.4

Solved problems
- In ***Emptying*** (both tool and working) some unnecessary rises to the coordinate in air have been removed
- Some cases of automatic and fictive faces.

1.6 Release 1.7.3

New functionalities
- Setting the r variables and string parameters, the range of the accepted characters (now also including e.g. literal accented characters) has been extended.

1.7 Release 1.7.2

Solved problems
- Some cases of automatic and fictive faces.

1.8 Release 1.7.1

Solved problems
- Inch configuration: dimensions of arrows and edge points in the graphic display.

1.9 Release 1.7.0

New functionalities
- Spanish manual: translation into the other languages has been aligned
- Import of DXF files: new options added

Solved problems
- In the section ***Execution sequences*** some cases of missing scrolling of the current working blocks during the graphic simulation of the sequence order.
- In the ***Custom sections*** some cases of missing update of the language messages for le selection in the list.
- After the activation of the graphic option ***Working coordinates*** some cases of partial view of the working coordinates.
- Some cases of failed working copy and paste command sequences. They are extreme cases of complex workings with many parameters (e.g. over 100 parameters deriving from a considerable number of reassignable variables).
- Installation process did not copy the CERCHIO.TMCR macro.

1.10 Release 1.6.9

New functionalities
- The start-up of the application program in Demo mode has an unlimited duration .
- In the window ***Open File,*** if a DXF - file is selected, the layers can be customized again.
- In the ***Execution Sequences*** section the column concerning the working description has been added.
- An option in the dialog box ***Customize ->Editor*** enabling or disabling the closure of the technology window after direct selection of the technology (during the insertion and/or the change of a working) has been added.
- In the dialog box **Customize->Editor** it has been added an option that enables or disables the propagation of the **Snap on Entity** mode (see **Insertion of Geometric Entities from Drawing**

Menu).

1.11 Release 1.6.8

New functionalities

- External management of the images on the menus
- Revision of the dialog box Information in Tpaedi32
- The organisation of the working area has been changed, by positioning the graphic areas in a more efficient way.
- In the **File** menu the command ***Open the prototype program*** has been added.
- In the ***Open File*** dialog box a DXF file is converted for the preview and the possibility to customize the layers has been eliminated.
- Opening a program created with Edicad custom sections are automatically assigned, using the settings defined into the prototype file.
- The *XYZ coordinates* of three-dimensional view of the face in the absolute system now can be represented.
- The toolbar of face view selection has been modified.
- In the *Special Settings* section now a field to exclude the graphic representation of elements such as arrows, edge points, 3d overall dimensions can be assigned.
- In setup workings the parameter ***Geometric Profile*** has been added, which if selected, excludes from the profile itself the graphic display of elements such as arrows, extreme points, overall dimension in 3D.
- In *Parametric Programming*
  - in the function *hypot* up to 3 argumentss can be defined
  - Fields belonging to the *Special sections* can be read
  - A *sysfeed* variable argument has been added in order to read the unit of measurement in which the speed is expressed.
- In the status bar you can see the indication of multiple setup (icon showing the tool movement in a profile).
- In the window **General program assignments** (dimensions, variables) the editing mode is always active.
- The tools of Horizontal mirror or Vertical mirror and of Profile inversion invert the tool compensation settings, as well and the selection of entry/exit segments, in case of right or left arc settings.
- During the execution of induced calls the propagation of J variables is applied.
- During the parametric programming in the window showing the list of the functions, the syntax of the selected function appears, as well. By clicking the [F1] button the application file guide is opened to the page matching the description of the parametric function required.
- During the insertion of a working, the last working can be inserted again through a shortcut menu, that is opened by clicking the right mouse button within the face visualization area.

Solved problems

- In case of interface in english language, the help was not opened, if the default language set was not English.

1.12 Release 1.6.7

New functionalities

- In *Tool Correction* the correction side can be changed
- New items in Parametric Programming:
  - logic function not
  - functions to read the additional parameters of fictive or automatic face. They are the geo[pr1;...], geo[pr2;...];geo[pr3;...] functions
  - the remarkable parameter items of complex code (subi -y/z-, subxn -y/z-,subxp -y/z-) in geo [param;...]
  - variable arguments to read the additional parameters in execution (prun1,...prun5)
- In the tool *Apply Feed in Z*, the option to apply the inversion of correction, if the function has been

enabled
- In Parametric Programming the window for immediate help has been restructured
- In the window Open the Piece the button **[Open]** is always enabled.
- On the right side of the Menu Bar the vertical allocation area of the default images has been reduced and the custom logo is displayed, only if it is present.
- In the window Information about Tpaedi32 a bitmap, describing its functioning context (Office or Office-Machine) in accordance to the available hardware key. In addition: the indication of the access level.
- The default language to choose the help file. If the help file of the selected language is not found, the help of the default language is loaded. Up to now the help in English was loaded in both these cases.
- In the table of the tools, that control the tool-holders, it is now possible to choose the point of the tool-holder, even if it is not an electrospindle
- The tool ***Repeat*** can be applied also to only a part of the profile
- In the graphic "match" of the working the next correspondence can be searched.
- The importation of a DWG - file has been removed.
- The menu bar *Selection Face* has been added.

Solved problems

- In Tools and text development codes the use of some particular fonts caused an error in Tpaedi32. Now these fonts cannot be used anymore.
- In the tool ***Lengthen a segment*** the elongation was not correctly performed, if the segment was a 3D arc.
- In Parametric Programming the function geo[param;..] did not read the parameters from a sub-program or macro code.
- In the segments of profile of an automatic face Construct and Level properties were not propagated.

1.13 Release 1.6.4

New functionalities

- In Tool Correction it is possible to correct the value of the tool radius.
- Inverted system for the slewing axis
- Management of a reduced menu in Help menu in parametric programming

1.14 Release 1.6.3

New Functionalities

- The tool Dimensioning
- Parametric Programming parameter "name" in the multi-purpose functions of the geometric library.
- Parametric Programming sysxz function showing the arc typology on a xz plane.
- The hardware keys of the same type can also be used for non - Tpa applications.
- Following tools ***Rotate, Mirrors***, ***Scale*** can be applied to a part of the profile.
- In the tool ***Profile Building*** intersection of arcs not assigned on a plane different than xy is recognized.
- In the local drawing bar the snap mode on the edge of a face has been added.

1.15 Release 1.6.1

New Functionalities

- Profile working codes with tangent solution in the space
- Option of linear segment in the space in the ***Apply entry/exit to profile*** tools
- Arc in the space in the ***Apply entry/exit to profile*** tools
- Possibility to fragment the arcs by means of the application of the chordal error in the ***Fragmentation to profile***
- Cut management of the arcs on no-xy planes in the tool **Profiles cut**
- ***Repeat profile*** tool
- In ***Compensation Tool*** : compensation management of the arcs assigned on a plane different than xy
- Some commands were moved from the Tool menu to the Construction menu.

- In Parametric Programming: a new function of geometric library to read a working parameter
- A command to activate and/or deactivate the representation of the overall dimensions into the three-dimensional view of the piece.
- Setup of added parameters into the variable geometries definition.
- Tangent and perpendicular snap mode to a segment in the local drawing menu.

Solved Problems

- The tool ***Apply exit to profile*** did not apply the programming of the Z-quote.

1.16 Release 1.6.0

New Functionalities

- Product working certification with the operative system Windows Vista
- New USB key codes, working with Vista O.S. have been assigned. Previous USB key codes are also valid with Windows Operative Systems, that are not Vista
- Custom files are saved in custom folders, created from the cadcfg folder. (Example: cadcfg\custom)
- Zoom +/- can be executed directly through the mouse scroll wheel
- Searching file options can be assigned in the ***Custom Sections*** file
- The tool ***Connection between profiles*** checks the connection between profiles, excluding the depth
- On the Status Bar also the tool correction is displayed
- In the technology table display of the only tools enabled for the current face
- In ***Customize Technology*** settings in parametric form can be assigned

Solved Problems

- In Parametric Programming a problem on the geometric library functions of correction of a linear and circular segment
- Management of assigned folders on a network resource, where one unity is not defined. For instance: "\\Server\\Machine\Program".
- In ***Compensation Tool*** were corrected cases in which the reduction of profile was not applied properly.
- The tool ***Connection between consecutive profiles*** did not apply the value of distance connection set, but always used the default epsilon.

1.17 Release 1.5.8

New Functionalities

- Added the tool ***Apply feed in Z***
- Added the tool ***Change edge into arc***
- In the tool ***Apply entry to profile*** it was added the possibility to assign an enter and/or exit coverage to a closed profile.
- In the tool ***Apply exit to profile*** it was added the possibility to assign an exit coverage to a closed profile.
- Added the possibility to execute the Zoom +/- directly through the mouse.
- Added the rotation on threedimensional view directly through the mouse.
- Added the rotation on threedimensional view on the plane on a generic face.
- Added in parametric programming the functions of the geometric library correction of a linear and circular line.
- Added in parametric programming the functions of geometric library, that use the working names.

1.18 Release 1.5.7

New Functionalities

- The tool ***Lengthen*** can also be applied to arcs laying on a different plane from XY.
- Advanced settings were added for the configuration of Settings from DXF/DWG formats.

Solved problems

- Solved cases of incorrect solution for XYZ arcs
- The Setup in "Per machine" mode was made functioning and now it is available for all the users.

1.19 Release 1.5.6

New functionalities

- It is possible to assign the folder and/or the file name during the storage of a file in an external format.
- During the importation of a file from an external format ist is possible to use the program according a default prototype file (variables 'o' and 'V, custom sections and names of the faces).
- A fictive face can be assigned according to the orientation of a selected line.
- Added the tool ***Measure***
- To the tool ***Circular series*** it was added the possibility to assign the center position in interactive mode. .

Solved problems

- During the functioning with S*tandard Key* the message of *Professional Key* failed testing was displayed.

1.20 Release 1.5.5

New functionalities

- During the storage of a program it is possible to assign a name without the extension.
- In Parametric Programming were added variable arguments (subvl, subvb,subvm,subvk).
- The piece graphics may be performed in transparency.
- You may exclude the color application to the current working
- In the piece graphics it is possible to set a pattern of background.
- You can set the types of files default, while reading a piece.

Solved problems

- Change languages did not update the title of the application.
- While setting the size of the piece was sometimes lost the decimal part
- The filters activation in Special Views could exclude the display of all the workings.
- If you open a program directly from Windows Explorer (double-click on the name of the program), if the name of the folder or of the program contained spaces, they were not open in Tpaedi32
- The table technology sometimes had problems in the presence of a tool collectors (carousel) or a toolholder.
- The points grid was not enabled.
- In the graphic representation of the piece in the window of fictive face assignment, the face of reference was not coloured.

2 Introduction

2.1 Overview

Tpaedi32 release 1.7.8

Tpaedi32 is an editor built in a graphic environment which allows to create, modify and import working programs as well as develop custom Subroutines and Macros.

It is available in two operating modes:

- **Standard mode**
- **Professional mode**

The **Standard mode** corresponds to the basic working level.
The **Professional mode** corresponds to the advanced working level. Compared to the basic level, the added commands are

- text generation
- emptying
- profile cut
- profile building
- spline curve generation
- extention of the tool compensation function
- assignment of automatic faces
- assignment of fictive faces with indication of the reference face
- extention of the parametric programming with addition of custom Functions
- automatic convertion into other file formats during program storage
- introduction of workings which apply geometric transforms.

The commands enabled in **Professional** mode only are marked in the manual by the symbol

PROFESSIONAL

The working program is represented both in graphical and text format with immediate interaction between the two representations.
The representation in graphical format can be made in 2D or 3D: 2D graphics allows the display of workings on the plane of every single face; while 3D graphics enables the visualization of all piece workings. In this view the representation can be: rotated (on three rotation planes, independent the one of the other), enlarged or reduced (multiple level zoom) or centered according to the needs.

The **graphical** representation enables the interactive identification of a working or a set of workings, with possibility to assign multiple representation filters and activate the visualization of only those workings which verify all assigned filters (types of workings, assignment levels, technology,..).
The graphical representation is based on multiple tools:

- cross cursor
- constant step grid, with possibility to customize grid elements (mesh, vertices,..)
- scattered elements grid, with possibility to customize grid elements (graticule, vertices,..)

The representation in **text** format allows to have the structured face program vision. It includes in fact all programmed blocks for a given face, together with those blocks which do not dispose of an associated graphical representation:

- blocks of conditional statements (IF.. ELSE .. ENDIF)
- blocks corresponding to programmed errors
- assignments of local variables to the face (j variables)
- commented blocks

In particular, the program text appears indented to underline logical conditions.

The text format is really an ASCII representation of the program and enables:

- direct modifications
- single or multiple selections
- in case of subroutines and macros, the visualization of every single working which corresponds to its development

A working program consists of a sequence of workings introduced by selection from a graphic palette and/or by insertion of geometric elements and/or application of CAD tools such as for example text writing and emptying of closed areas.
Workings can be modified as follows:

- by acting directly on every single working
- by applying modifications common to a set of workings
- by applying geometric transforms to a set of workings (translation, mirrors, repetitions)
- by applying profile handling tools (scale, inversion, disconnections, connections, tool radius compensation)

Moreover, a lot of tools are available for a targeted handling of the concerned working program:

- general tools: translation, rotation, mirrors, serial repetitions on pre-established paths, exploded view of subroutines or macros
- profile tools: inversion, scale, application of technology, profiles joint (with translations, connection segments, inversions), vertices editing (with chamfering or fillet solution), application of opening or closing setup, fragmentation and minimization, disconnection, extension of profiles
- CAD tools: text generation, emptying process of closed areas, profiles cut, generation of spline curves from polylines, use of workings which apply geometric transforms

An important aspect in the assignment of workings is the parametric programming, which allows the use of:

- piece variables
- mathematical functions
- geometric functions

The parametric programming can be used to assign program variables, variable geometries and working parameters.
It is important to underline the possibility to assign custom functions. It is about functions, whose assignment logic is defined according to custom needs, and which can be used at every programming level (variable, geometry, working assignment).

Functions and variable arguments, available in parametric programming, make it possible to exercise an effective and complete control on the context in which Tpaedi32 and the individual working program are operating:

- technology
- configuration settings
- execution mode
- geometric characterizations of the piece
- application mode of a subroutine or macro

The multi-purpose geometric library function is particularly useful, since it provides an immediate solution to problems of geometric nature, even very complex ones.
The high number of functions and variable arguments, available in parametric programming, has suggested to design contextual aids to help programming:

- it is possible to select the function (or variable argument) from a sorted list
- during assignment of a function it is possible to require the display of a help concerning the accepted call syntax

General piece assignments enable a great number of logical conditions which characterize the **executive composition** of a working program. Therefore, the same program can generate machine working programs with different geometric data (size, assignment of working faces), execution mode (normal, mirrors), exclusions, reassignment of program variables (offsets, cycle variables).

2.2 How to import an Edicad program

A program written with Edicad is recognized as a Tpaedi32 compatible format and can be directly loaded and processed. The contrary is not true: in fact, a program generated with Tpaedi32 cannot be read by Edicad.
To read a macro written with Edicad it is necessary to save it in ASCII format.
The custom sections are automatically assigned, by using the settings defined into the prototype file PIECE.TCN (stored in the cadcfg\custom folder). The custom sections include the special Settings, the joined Infos, the section of Constrains, the optimizing Settings.

Information retrieved during importation:

- The three offsets are retrieved in the first three "o" variables
- The cycle variables are retrieved in "v" variables
- The shaped piece assignment is retrieved in the Special Settings section, in the WRAP=ni string identifier
- Fictive face information: the similar face assignment is retrieved in the direction set for z axis
- In parametric programming every use of ,(comma) is replaced by ; (semicolon)
- During reading of program workings: the parameter corresponding to the comment is retrieved as working comment (e.g. IF, FOR,..)
- During reading of program workings, a few working codes are reassigned with other equivalent codes
- Programs assigned in Edicad as sub-cycles are retrieved with macro typology

Information lost during importation:

- Sequence field assignment in single face

Information not retrieved during importation:

- Technological parametric programming functions concerning multi-drill heads
- The sub-cycle call syntax with the "*" character to address the subroutine call in cadcfg\sub directory is no longer supported.

2.3 New Functionalities

Tpaedi32 offers new implementations and functions compared to Edicad:

Programming Environment

- The window layout for displaying piece, piece faces and workings is completely renewed
- the representation of workings is managed directly on the piece in three-dimensional view
- the application of tools such as zoom and rotation, already included in Piece Overall View, is managed
- the program representation and modification both in graphical and text format is always contextual
- it is possible to develop new programs from a predefined model
- a commands visualization area and an errors visualization area are added
- the graphic preview is added in the program opening window
- the possibility to assign visualization filters and/or modify workings is added
- conversion into remarkable formats (DXF, CNC90, custom) during program loading, is automated
- program generation in external, Edicad format and in other custom formats is directly managed during storage
- the application of the program execution mode to the active program as well as the exclusion of a set of workings from being executed are implemented

Environment Configuration

- customization of the meaning of "o" (offset in Edicad) variables and "v' (cycle variables in Edicad) variables
- possibility to custom up to 4 sets of general information on custom use
- free face numbering
- setup of reference origins for each face
- application of tools to macros as well
- workings palette customization
- possibility to operate without workings palette

Profiles

- insertion of open profiles: it is no longer necessary to introduce a setup working as first work on a profile
- arc programming on general xyz plane
- new performances in tool compensation such as profile reduction in correction, suspension and possible resumption of correction

Face Program

- display and/or modification of the face program in ASCII format with possibility of:
  - nested text
  - direct modification of the ASCII text
  - display of the list of workings corresponding to the application of a complex working
- 2D and 3D face program view with possibility of rotation, zoom, pan, multiple selections

Workings

- introduction of new workings: ELSE, Ternary assignment code of J variables, geometric figures such as rectangle, polygon, portion of ellipse, portion of oval
- introductions of workings, which apply general geometric transforms (translation, mirrors, rotation, scale, repetitions), emptying, spline curve generation
- optimization and reduction of the number of profile codes by maintaining the same Edicad performances
- addition of new working properties such as Field C , Field K and Field N. Other properties have been expanded
- management of a descriptive field for each working

Subroutines

- possibility of nesting for no more than 4 subroutine calls
- a subroutine or a macro can recall subroutines or macros
- addition of functionalities to the mechanism of induced calls
- addition of transform parameters such as stretch, emptying, text generation

Parametric Programming

- the maximum dimension of a parametric expression has been increased
- recall of custom functions to solve, for example, specific repetitive calculations of a single application
- possibility to recall immediate Help for functions and variable arguments available for parametric programming
- assignment and modification of J variables during program modification
- possibility to verify the value taken by the J variable in every point of the program
- addition of o0-o7, v0-v7 variables
- redefinition of technology functions
- new geometric library functions
- new functions for string management

Fictive Faces

- new data entry window
- simplified creation of fictive faces, on the basis of geometric entities programmed in a face of the piece
- possibility to create a face in relative format compared to another already assigned
- possibility to configure the z axis direction and the face thickness

New Tools

- disconnect profile: it allows to delete part of the current profile or split a segment of profile into two separate segments
- exploded view of complex profiles: it converts the complex working (e.g. ellipse) into arc and line elementary workings
- emptying of closed areas
- spline generation from polylines
- profile scale: it applies a scale factor to one or more profiles, with possibility to operate in 2-dimensional plane (xy) or in 3-dimensional space (xyz)
- profiles cut: it cuts a profile at an edge defined by one or more profiles
- profile building: it builds a profile at an edge defined from one or more profiles
- Setup application to open profiles (without opening setup) or to geometric setups
- conjunction with translation of multiple profiles
- profiles joint
- extension of segments of profile: it extends a segment of profile up to intersect a selected boundary item
- text generation: it inserts text in the program directly as profile development
- exit/entrance application to complex profile
- closure of open profile
- repetitions on rectangular path
- repetitions on circular path

2.4 Authorization

The Tpaedi32 program can be executed in Professional or Standard mode
If the test fails, it is possible to enable the Demo mode (see Chapter ***Authorization->Demo***)

2.4.1 Hardware

The use of the application is authorized by the presence of a properly programmed USB hardware key. The hardware key can be moved by a computer to another, allowing to operate in *CAD or Standard Mode* on different *Tpaedi32* installations: obviously it is not possible to work in both operating modes at the same time. The presence of the key, in fact, is tested each time it is required to execute specific commands.

2.4.2 Demo

The Demo mode has un unlimited duration and it is enabled when *Tpaedi32* does not find the the hardware key. In this case, when the application is run, a window appears to warn that the selected installation does not allow to use all program functions.
If this window is displayed although a properly programmed hardware key exists, it means that something is malfunctioning:

- the hardware key is not properly read or is not typed in the appropriate port.

In this case perform all the necessary tests.
In this mode not all program functions are available, in particular:

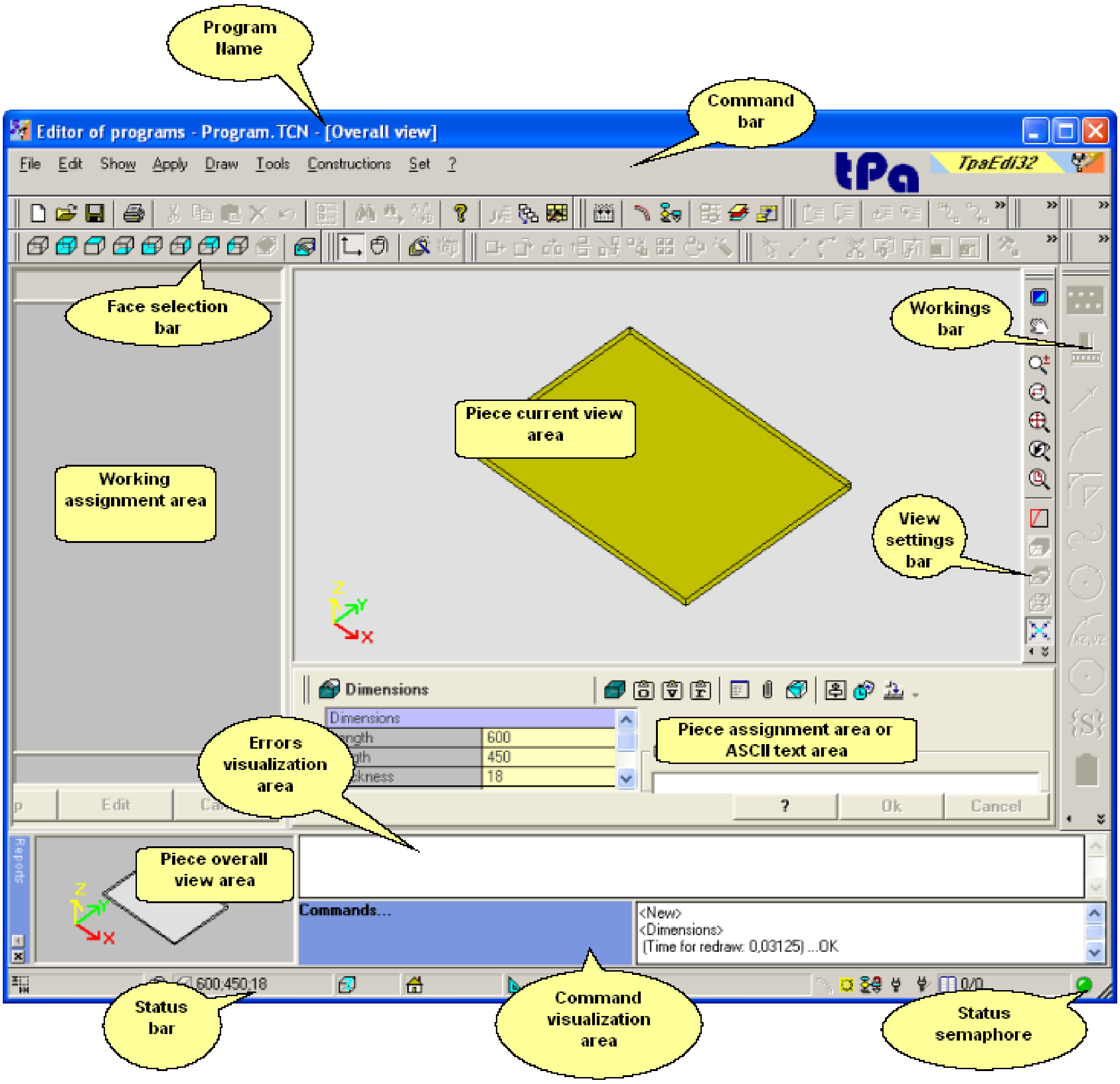
- the minimum access level is always active;
- it is not possible to save programs;
- it is not possible to optimize programs;
- it is not possible to create, modify or delete user workings.

If the setup is carried out on a computer, whose access is configured to more users, the start of Demo modes may be carried out only by the user with access modes as Administrator.

3 Graphical Interface

3.1 How It Appears

At start-up of a new program the work area appears as follows:



Dimensions	
ngth	600
th	450
kness	18

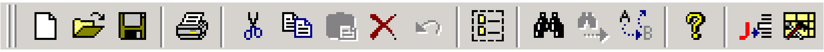
Commands...

```
<New>
<Dimensions>
(Time for redraw: 0,03125) ...OK
```

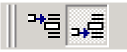
3.2 Command Bars

In Tpaedi32 multiple button bars are displayed. Each button corresponds to a Command or Setting or Activation. Generally speaking, each button corresponds to a menu item.
Each individual bar can be repositioned inside the main window, as it usually happens in a Windows application.

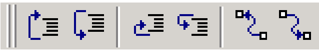
Main Bar



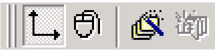
Insertion Bar



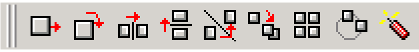
Placement Bar



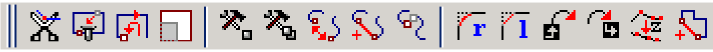
Tool Option Bar



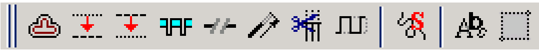
General Tool Bar



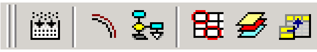
Profile Tool Bar



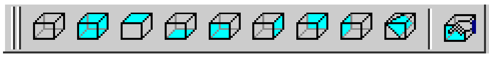
Advanced Tool and Path Bar



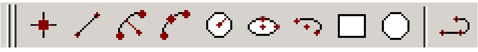
Special Views Bar



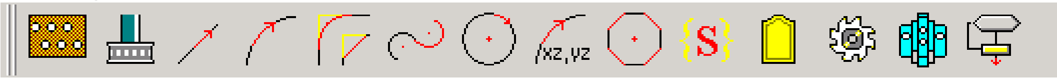
Face Selection Bar



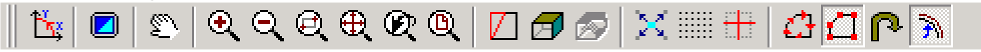
Drawing Bar



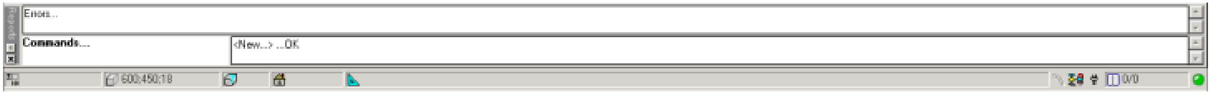
Workings Bar



View Settings Bar



Reports Bar

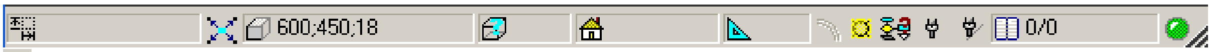


Errors...

Commands... <New...> ...OK

600;450;18 0/0

Status Bar



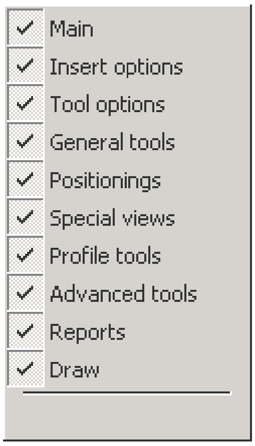
600;450;18 0/0

● The green traffic light indicates that Tpaedi32 is waiting for commands.
The red traffic light indicates that Tpaedi32 is performing

- a process (e.g. graphic is being updated)
- an edit (e.g. piece dimensions and <r> variables are being edited)

Display, hide and position command bars

To modify the command bar status right click the mouse on the **Command Bars** area and then select the item corresponding to the bar to be displayed or hidden. The displayed bars are marked with the following symbol ✓



☑ Main
☑ Insert options
☑ Tool options
☑ General tools
☑ Positionings
☑ Special views
☑ Profile tools
☑ Advanced tools
☑ Reports
☑ Draw

Can't disable the visualization of Workings Bar, Face Selection Bar and View Settings Bar. The View Settings Bar can be displaced only within the area of the graphic representation.

As a result of a monitor change or graphic resolution the Face Selection Bar could not be displayed when starting Tpaedi32. To replace the bar within the graphic area of the application, the procedure is the following:

- open the menu changing the status of the command bars (see the figure before);
- hide some bars, until the Face Selection Bar is displayed;
- position the Face Selection Bar onto the point required;
- show again those bars, that were hidden.

3.2.1 Piece View Areas

Piece Overall View Area
It provides the 3-dimensional graphical representation of the piece. The piece is represented in scale to display all assigned faces, including fictive faces. The area manages interactions only for the rotation of the piece.
The area cannot be seen, if the Report Bar view has been disabled. In this case check the piece current view area.

Piece Current View Area
It provides the graphical representation of the current view.
<u>Overall View</u>
It is possible to choose between two options:

- inactive graphical representation: the area appears empty
- 3-dimensional graphical representation of the piece: the area displays the piece in spatial view (xyz), with the relevant face workings applied

<u>Face View</u>
A single face graphical representation is managed, in the face plane, with possibility to assign the xy axes orientation to adopt for the face plane.
Furthermore, it is possible to shift from the face plane view to a 3-dimensional view, with:

- the piece displayed in its overall dimensions and in the faces which compose it;
- current face workings displayed in 3D, together with their overall dimensions.

Assignment Area in Current view
It consists of several pages which can be activated by selection from tool bar. Each page displays and allows to configure a homogeneous set of piece assignments, generally arranged in tables.

Working Assignment Area
This area is managed only in face view.

4 How To Configure the Graphical Representation

4.1 Activation and Deactivation Of Visual Items

It is recalled from ***Show->Cursor*** menu. It enables or disables the cross-cursor display to help identifying the active working. The cursor is centered on the working application point. In piece overall view the cursor is displayed in execution sequences assignment.

It is recalled from ***Show->Working reference*** menu. It activates or deactivates the display of the graphic item which indicates the reference set for the active working. The activation depends on the assignment of an interpretation in Field O as reference (face edge or side). In piece overall view the reference is displayed in sequences assignment.

It is recalled by ***Show->Working coordinates*** menu. It activates or deactivates graphic display of coordinates for current working. For example, if current working is an arc, coordinates are displayed for arc extreme ends, centre and initial radius, like a straight line between arc initial point and centre. This command is only enabled in front view.

It is recalled from ***Show->Grid*** menu. It enables or disables the grid display. The grid activation is interpreted only in face view. It is disabled in 3d face view.

It is recalled from ***Show->Special Grid*** menu. It enables or disables the special grid display. It is about a grid directly assigned for individual points and defined, during configuration, from the machine manufacturer. The grid activation is interpreted only in face view. It is disabled in the 3d face view.

It is recalled from ***Show->Snap to Grid*** menu. If active, it limits the cursor movement at default active grid vertices. Snap to grid affects:

- mouse position display on the status bar
- acquisition of coordinates in some tools
- direct application of geometric items

It is recalled from ***Show->Profile direction*** menu. It enables or disables the profile direction arrows display. Compensated profiles as well as construct and emptying profiles are excluded. The activation is interpreted also in piece overall view.

It is recalled from ***Show->Points on profiles*** menu. It enables or disables the display of profile edge points (small circles). Compensated profiles as well as construct and emptying profiles are excluded. The activation is interpreted also in piece overall view.

It is recalled from ***Show->Face Building*** menu. In the case of fictive faces, where the programming of the three remarkable points of the face does not correspond to a system of three orthogonal axes, this option enables or disables the display of the face building between the programmed y axis (not perpendicular to the face x axis) and the calculated y axis (perpendicular to the face x axis). The activation is interpreted:

- in Overall View Area, for all concerned faces
- in Current View Area, only in case of face view and limited to the current face: this to avoid to load down and complicate the face representation.

The activation is not interpreted in Current View Area, in case of piece overall view.

It is recalled from ***Show->3D-graphics overall dimensions.*** It activates or deactivates the flag application of the overall dimension view into the three-dimensional graphics. Flags are sets into the page *Customize->Views*. If the option is disabled, any kind of overall dimension is displayed. This option is important only within a three-dimensional representation.

4.2 View Check

Zoom and Pan commands allow to enlarge, reduce, redimension what is displayed in the active view area. Zoom and Pan do not modify the face size, but only the dimension of the area represented within the view window. Zoom and Pan are active also in piece overall view. Commands work on the area of current view.

It is recalled from ***Show->Zoom->Zoom In Out***. It increases or decreases the representation scale (zooming in or zooming out). To enlarge, hold down the left mouse button and drag the mouse cursor to the top. To reduce, hold down the left mouse button and drag the mouse cursor to the bottom. To exit press ESC or the right mouse button. The command is disabled when it reaches the maximum zoom-in or Zoom-out level.

Zoom In Out using the mouse. This command is always active and increases or decreases the current representation scale (zoom in or zoom out) in the Overall View, if the window is active and placed in the Current View area. To zoom in, rotate the mouse scroll wheel up, to zoom out rotate the mouse scroll wheel down.

It is recalled from ***Show->Zoom->Zoom Window*** menu. It zooms in on a rectangular area by specifying the edges which define it. The shape of the specified zoom window does not necessarily match the new view, which is automatically adjusted to maintain the proportional view on the plane.

It is recalled from ***Show->Zoom->Zoom Extents*** menu. It displays the face and the relevant workings in their overall dimensions, in the maximum allowed representation scale.

It is recalled from ***Show->Zoom->Previous*** menu. It restores the previous view settings with memory up to 10 levels.

It is recalled from ***Show->Zoom->Zoom All*** menu. It displays the face and the relevant workings in their overall dimensions, in the scale representation allowed by view dimensions assigned.

It is recalled from ***Show->Pan*** menu. It enables to generate an interactive view overview: click on the view area and *drag* in the required direction without releasing the mouse key, thus displaying a different part of the view. At the end, release the key.

4.3 Three-Dimensional Representation

It is recalled from ***Show->3D View*** menu. The command activates the three-dimensional representation, with the possibility to rotate the piece in space, to highlight the concerned part. In face view (or in overall view during the sequence assignment):

- when it is selected: the three-dimensional representation is activated
- when it is deselected: the representation sets the view on the face plane.

When the three-dimensional representation is active it is possible to rotate the piece on three planes, assigned according to the graphical representation selected:

- xy plane: the piece rotates counterclockwise or clockwise on the view plane, with rotation axis perpendicular to the view plane itself (cartesian z axis)
- yz plane: the piece rotates upward or downward, with horizontal rotation axis (cartesian x axis)
- xz plane: the piece rotates leftward or rightward, with vertical rotation axis (cartesian y axis).

The piece can be rotated using the mouse. You need to move the mouse pointer within the area of representation of the piece in the direction towards which you will turn the piece itself, while keeping down the left mouse button. The rotation with the mouse is disabled when you are using other interactive commands as Zoom In, Zoom Out, Zoom Window and Pan.

It is recalled from ***Show->Specials in 3d view->Upward rotation step*** menu. The piece rotates upward, with horizontal rotation axis. The rotation increment, assigned in the view customization page, is applied. Rotation can be activated by holding pressed **[Shift + UP arrow]** keyboard buttons; rotation ends when keys are released.

It is recalled from ***Show->Specials in 3d view->Downward rotation step*** menu. The piece rotates downward with horizontal rotation axis. The rotation increment, assigned in the view customization page, is applied. Rotation can be activated by holding pressed **[Shift + DOWN arrow]** keyboard buttons; rotation ends when keys are released.

It is recalled from ***Show->Specials in 3d view->Left rotation step*** menu. The piece rotates leftward, with vertical rotation axis. The rotation increment, assigned in the view customization page, is applied. Rotation can be activated by holding pressed **[Shift + LEFT arrow]** keyboard buttons; rotation ends when keys are released.

It is recalled from ***Show->Specials in 3d view->Right rotation step*** menu. The piece rotates rightward, with vertical rotation axis. The rotation increment, assigned in the view customization page, is applied. Rotation can be activated by holding pressed **[Shift + RIGHT arrow]** keyboard buttons; rotation ends when keys are released.

It is recalled from ***Show->Specials in 3d view->Counterclockwise rotation step*** menu. The piece rotates counterclockwise on the view plane, with rotation axis perpendicular to the view. The rotation increment, assigned in the view customization page, is applied. Rotation can be activated by holding pressed **[Shift + "+"]** keyboard buttons; rotation ends when keys are released.

Rotation using the mouse. This command is always active. To rotate the piece you need, keep the left mouse button pressed and move the cursor in the direction toward which the piece should be turned. The rotation with the mouse is deactivated, if other interactive graphic commands are activated at the same time, such as ***Zoom In-Out, Zoom window, Pan.***

It is recalled from ***Show->Specials in 3d view->Clockwise rotation step*** menu. The piece rotates clockwise on the view plane, with rotation axis perpendicular to the view. The rotation increment, assigned in the view customization page, is applied. Rotation can be activated by holding pressed **[Shift + "-"]** keyboard buttons; rotation ends when keys are released.

It is recalled from ***Show->Specials in 3d view->Default view*** menu. The piece is positioned according to default angles, as assigned in the view customization page.

It is recalled from ***Show->Specials in 3d view->Top view*** menu. The piece is viewed from the top face (face 1).

It is recalled from ***Show->Specials in 3d view->Button view*** menu. The piece is viewed from the button face (face 2)

It is recalled from ***Show->Specials in 3d view->Front view*** menu. The piece is viewed from the front-side face (face 3).

It is recalled from ***Show->Specials in 3d view->Back view*** menu. The piece is viewed from the back-side face (face 5).

It is recalled from ***Show->Specials in 3d view->Right view*** menu. The piece is viewed from the right-side face (face 4).

It is recalled from ***Show->Specials in 3d view->Left view*** menu. The piece is viewed from the left-side face (face 6).

It is recalled from ***Show->Specials in 3d view->Show all fictive faces*** menu. It enables or disables the overall display of the variable geometries of the piece. The activation is interpreted only in face view. When it is activated: also the overall dimensions of all fictive faces are represented in face view. When it is deactivated: the overall dimension of the parallelepiped piece, the automatic faces and the active face are represented in face view.

It is recalled from menu ***Show->Specials in 3d view->Face plane.*** The piece is placed with view of the current face.

4.4 Special Views and View Filters

The activation of special views and view filters allows to modify the content of the view.

It is recalled from ***Show->Selections*** menu. It activates the display of the only selected workings. This view is activated together with the other special views (Tool Radius Compensation, Logical Conditions) and the view filters (Levels, Special filters).

It is recalled from ***Show->Tool radius compensation*** menu. It enables or disables the display of Tool radius compensation. This view is activated together with other special views and view filters and it is interpreted also in piece overall view. In case of error detection by the application procedure tool radius compensation:

- the special view is not activated
- in Errors area it is possible to see error conditions.

It is recalled from ***Show->Compensated profiles extents*** menu. If active, compensated profiles and those which do not apply any correction are represented with thickness equal to the tool diameter. For these profiles edge points as well as direction arrows are not displayed. In any case, the following items are represented with the unit thickness:

- the construct profiles
- the segments of profile built in air, unless the option ***Grafic profiles extents in the air*** in the dialog box ***Customize->Graphic*** is activated.

It is recalled from ***Show->Compensated original profiles*** menu. If enabled, the view displays also original profiles (incorrect profiles). If disabled, the view shows only compensated profiles and those which do not apply any correction (with application of direction arrows on represented segments, if necessary). The selection is active in tool compensation view.

It is recalled from ***Show->Logic conditions*** menu. It enables or disables the application and the consequent display of logical conditions. With active selection, only the workings which are verified according to logical conditions are shown. This view is activated together with other special views and view filters. The view activation is interpreted also in piece overall view. If some exclusions are assigned, these last are applied and evaluated in the same way as logical conditions. In case of error detection by the application procedure of logical conditions:

- the special view is not activated
- in Errors area it is possible to see error conditions.

It is recalled from ***Show->Levels*** menu. It assigns the visibility status of those workings which are defined on a given level. This view is activated together with the other special views and the view filters and it is interpreted also in piece overall view. The command is not available if the Level management is not enabled.

It is recalled from ***Show->Special views*** menu. It assigns the visibility status of workings according to their association with one or more remarkable assignments (property, technology,..). This view is activated together with the other special views and the view filters and it is interpreted also in piece overall view. This is an optional command.

4.5 General Graphics Commands

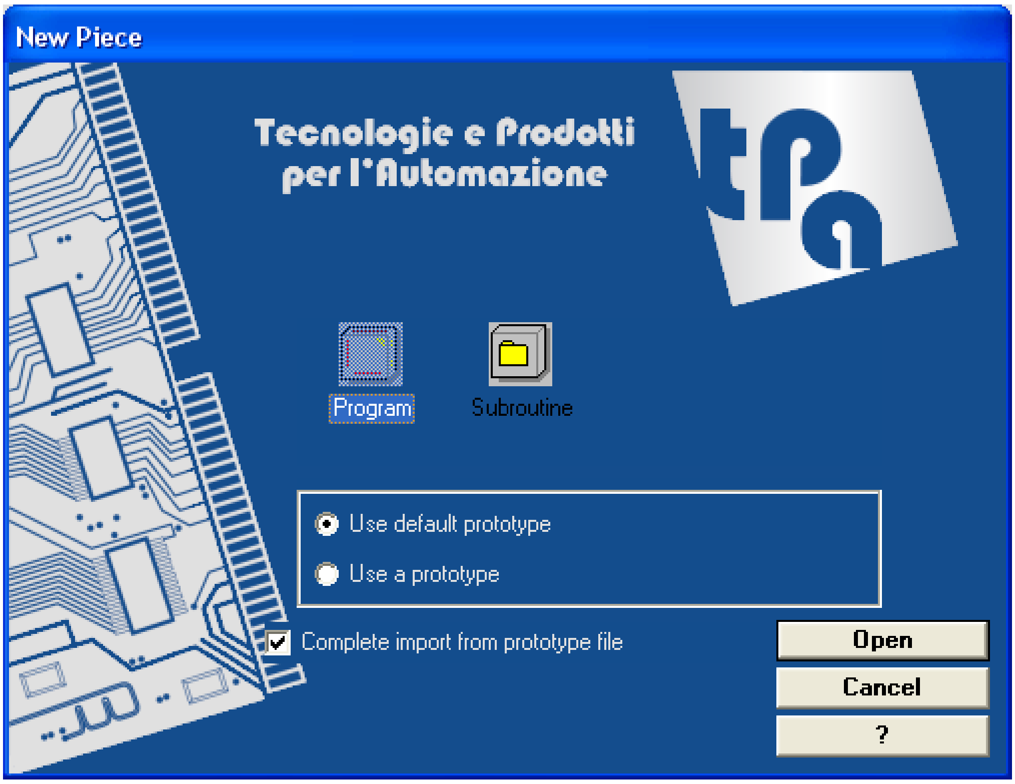
It is recalled from ***Show->Redraw*** menu. It regenerates the overall view display, with application of all currently assigned graphic settings (visual items, zoom, pan, special views and view filters).

Command only from ***Show->Show*** menu. If disabled, it resets and stops the representation of the current view.

5 Working With Programs

5.1 Creating a Program

Tpaedi32 allows to create programs, subroutines and macros by the ***File->New*** command or icon 🗋.



● Use default prototype

○ Use a prototype

☑ Complete import from prototype file

It is usually chosen to develop a **ProgramFile.** When it is necessary to define once for all a set of workings to be used repetitively in a ProgramFile, it is chosen to create a **SubprogramFile** The possibility to create a **MacroFile** is active only if the access level is equal or higher than Constructor. On starting a new program or subroutine or macro, following two options are available:

- **Use default prototype:** if selected, an initialisation process of the program, according to a default model program is activated. (PIECE.TCN, in the cadcfg\custom folder). In case this default program is not installed, the data concerning units of measurement, dimensions and face names are retrieved from the last program opened. When you use the model file the program is initialised using the data of measure and dimension, variables and variable geometries, distinctive custom sections if the selected program. If the option **Complete import from prototype file** is selected, into the new piece the programs of the face are initialised according to the programs of the model file; if the options not selected the programs of the face are initialised empty
- **Use a prototype:** if selected, after confirming the creation of a new piece File Open window opens, to allow the file-piece selection to be used as a model. The research is preset in the standard storage folder of the programs to be used as models (folder: Product\Models). Dialog box File Open can manage the preview of the file-piece. The program is initialised as above unit of measure and dimensions, variables and variables geometries, distinctive custom sections of the selected program.

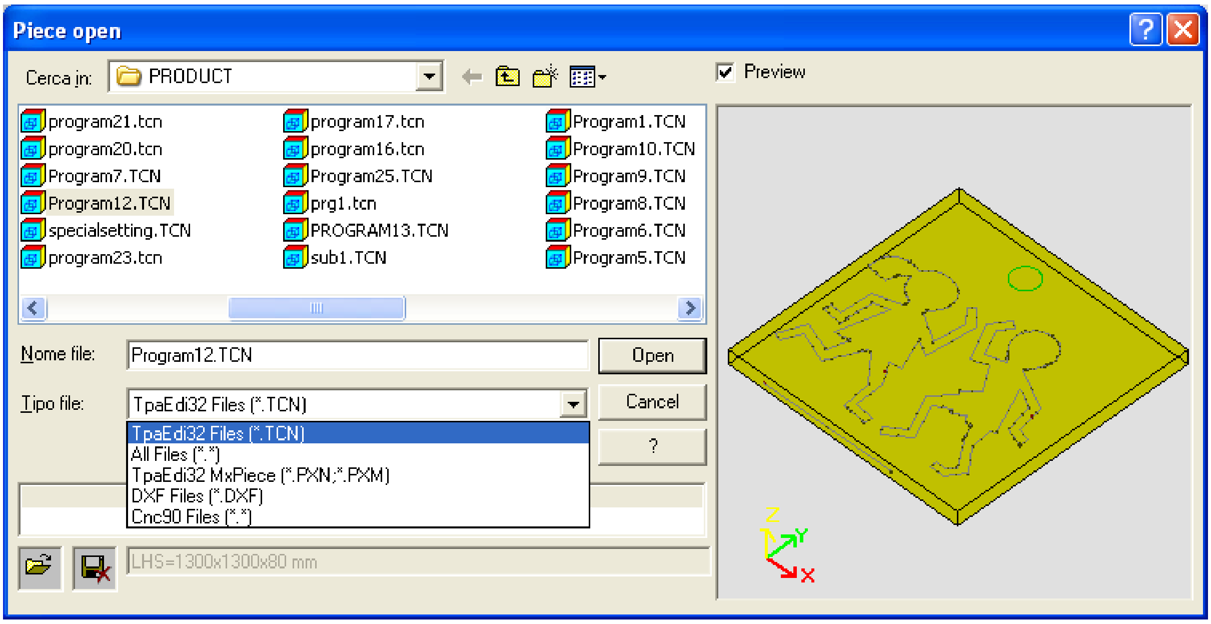
How to modify the file PIECE.TCN
In the menu ***File*** the command ***Open prototype program*** is available**.** It opens directly the model program to modify. It the program model cannot be found, it is created automatically

5.2 Opening and Importing a Program

Tpaedi32 allows to open programs, subroutines and macros by the ***File->Open*** command or icon 📂.
Tpaedi32 recognizes two typologies of programs:

- TCN extension: default extension for programs and subroutines (file type: **Tpaedi32 Working Program**)
- TMCR extension: default extension for macros (file type: **Tpaedi32 Macro File**).

The two typologies are included in **File Type** list. That of macro-program only if the access level allows its opening. In any case, it is not compulsory to assign for a piece-program the extension matching the selected typology. However, it may help you recognize immediately piece-programs.
If the selected program is recognized as piece-program, it is possible to enable a graphic preview **(Preview)**, which can be aborted by pressing the **[ESC]** key. Comment and dimensions are also displayed in the relevant window.
If you try to upload a program created using professional level tools in a system working with standard level keys, the program is not opened and
the system shows that it is not possible both to open and to record the program.
On the bottom left hand side two images are displayed: they signal if the selected program is read and/or write protected. In the example given, the selected user-level program is only write-protected.



Cerca in: PRODUCT

☑ Preview

program21.tcn
program20.tcn
Program7.TCN
Program12.TCN
specialsetting.TCN
program23.tcn
program17.tcn
program16.tcn
Program25.TCN
prg1.tcn
PROGRAM13.TCN
sub1.TCN
Program1.TCN
Program10.TCN
Program9.TCN
Program8.TCN
Program6.TCN
Program5.TCN

Nome file: Program12.TCN

Tipo file: TpaEdi32 Files (*.TCN)

TpaEdi32 Files (*.TCN)
All Files (*.*)
TpaEdi32 MxPiece (*.PXN;*.PXM)
DXF Files (*.DXF)
Cnc90 Files (*.*)

LHS=1300x1300x80 mm

In the open window of an existing program it is possible to select programs of formats differing from the one specified in Tpaedi32.
In particular:

- DXF Files (*.DXF);
- CNC90 Files (*.*);
- Custom format files.

In case of recognized valid format for DXF files the graphic preview (**Preview**) with file convertion is managed.
In case of recognized valid format for CNC90 files comment and dimensions are extracted and displayed.

Some custom types of files can also added, for each of which the program cannot interpret any considerable format, therefore it is assigned a format converter.

In those cases where a file is open, where it is necessary to convert the format, global instruments on the program can be activated, according to the *Customization of Tpaedi32*. In particular, the activation may:

- be automatically operational, without any request for information,
- dipend on a proposal requested to the operator with a message "*Do you want to apply the selected automatic assignments?",*
- be disabled.

If the program to open is in DXF format, it is possible to indicate which layers need to be converted and which need to be excluded. After confirming the opening and if the program has assigned layers, it is required whether to assign the layers. If the answer is **[No]** the conversion rules are applied just as they have been assigned in ***Customization of Tpaedi32,*** ifthe answer is **[Yes]** it is opened the list of the layers which can be included. To exclude a layer the visible check next to the name should be removed. If a layer assigning the panel dimension is required, place the cursor on the name of the layer itself and select the button

Automatic assignment may involve the application of specific instruments on the whole program:

- general settings with reading from the prototype file PIECE.TCN (stored in the folder cadcfg);
- application of the technology to the open profiles or to the profiles assigned with geometric setup;
- application of the technology to the point workings assigned with geometric point code;
- profiles reduction with reduction angle reset;
- fragmentation of the arcs profiles with following linearization;
- automatic profiles connection verifying their geometric continuity;

While opening a file that needs a custom conversion, you can request to change the arguments used for the conversion. In the opening window
and only in this case, a flag **Arguments inquiry** is displayed. This one, if selected, enables an opening window, where the arguments carried to the conversion can be changed. In any case, this option is active only if determined while configuring Tpaedi32. It is important that the user, who set up the conversion parameters, knows its meaning.

5.3 Saving a Program

Tpaedi32 saves the current program by assigning the file name and the current location by ***File->Save*** command or icon 💾. In case the program being edited is new or if ***File->Save as*** command is selected from menu, a window is displayed where the user can assign the file name and its storage location. The extension to be assigned to the file can be chosen among those, that has been suggested or can be defined by the user. The extension by default are as follows: TNC for programs and subroutines, TMCR for macros. In case of storage of a macroprogram, the only TMCR extention is suggested.
It is advisable not to assign the TMCR extension (default extension for macro) to programs or subroutines.

The storage procedure can be followed by other procedures configurable by the machine manufacturer. In particular and in execution order:

- program conversion into an external format. It is useful, for example, when a program is created with Tpaedi32, but it is wished to be imported into Tpaedi32
- program optimization, with optional storage of the piece matrix
- in alternative to optimization, the original program text can generate directly the piece matrix without applying any optimization procedure.

These procedures are not activated in case of macro-programs.
Sometimes, enabling these procedures may be time demanding; when clicking inside the graphic area, the user is warned that program filing is not finished yet, by a message window opening.

5.4 Exporting a Program

Tpaedi32 exports an active program in the format configured by machine manufacturer. The command is enabled by ***File->Export*** menu. If more export programs are configured, a menu is displayed to select the conversion type. Marking the option **Save as** for the program that need to be converted, it is possible to select his storage folder and a file name. Export parameters are the same as set up in the program until the command is selected: running mode, exclusions, dimensions, variables, etc.

5.5 Printing a Program

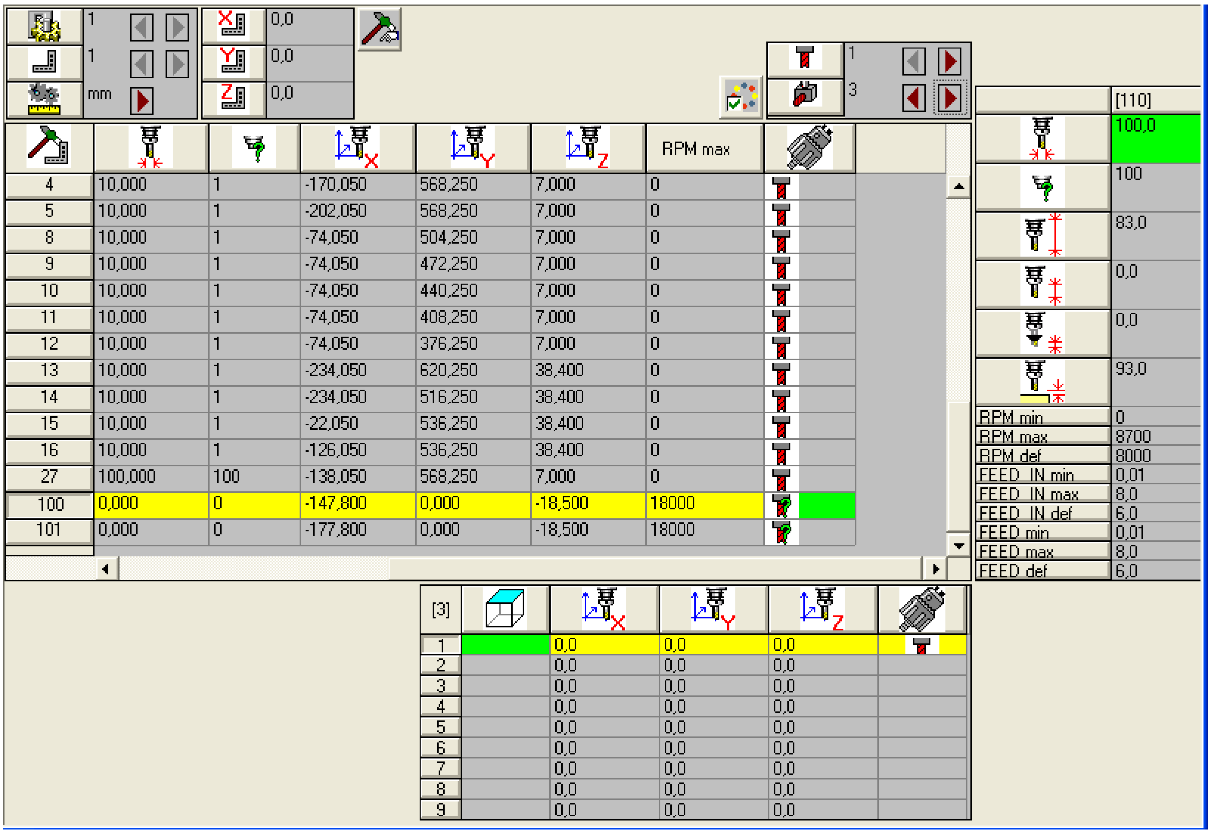
<%NOME_ESEGUIBILE%> prints the active program both in text format, by ***File->Print ISO*** menu item or by icon 🖨 and in graphical format by ***File->Print Graphics*** menu item or by icon 🖨***.***
In Overall View graphic print-out corresponds to the current graphical representation of the piece. In Face View it corresponds to the current graphical representation of the face.
In particular:

- it respects the active zoom and pan settings
- cursor, grid, end points and direction of profiles are printed, if displayed
- it respects view filters and special views

The text print-out of the active program prints the entire program in Overall View; while in Face View it prints only the face program. Print does not match exactly the ASCII text which is stored for the program. Each section is headed by a language message and spacing lines are introduced among sections.

5.6 Tools Table

Tpaedi32 usually operates in a technological context: that is directly interfaced to one or more machines, of which it knows the assignments concerning working tools.
The technological assignment of tools is of primary interest for workings which can be executed in a program. For this reason it is usually possible to display the table of tools available for workings



	1
	1
	mm

X	0,0
Y	0,0
Z	0,0

	1
	3

			X	Y	Z	RPM max	
4	10,000	1	-170,050	568,250	7,000	0	
5	10,000	1	-202,050	568,250	7,000	0	
8	10,000	1	-74,050	504,250	7,000	0	
9	10,000	1	-74,050	472,250	7,000	0	
10	10,000	1	-74,050	440,250	7,000	0	
11	10,000	1	-74,050	408,250	7,000	0	
12	10,000	1	-74,050	376,250	7,000	0	
13	10,000	1	-234,050	620,250	38,400	0	
14	10,000	1	-234,050	516,250	38,400	0	
15	10,000	1	-22,050	536,250	38,400	0	
16	10,000	1	-126,050	536,250	38,400	0	
27	100,000	100	-138,050	568,250	7,000	0	
100	0,000	0	-147,800	0,000	-18,500	18000	
101	0,000	0	-177,800	0,000	-18,500	18000	

	[110]
	100,0
	100
	83,0
	0,0
	0,0
	93,0
RPM min	0
RPM max	8700
RPM def	8000
FEED IN min	0,01
FEED IN max	8,0
FEED IN def	6,0
FEED min	0,01
FEED max	8,0
FEED def	6,0

[3]		X	Y	Z	
1		0,0	0,0	0,0	
2		0,0	0,0	0,0	
3		0,0	0,0	0,0	
4		0,0	0,0	0,0	
5		0,0	0,0	0,0	
6		0,0	0,0	0,0	
7		0,0	0,0	0,0	
8		0,0	0,0	0,0	
9		0,0	0,0	0,0	

The figure shows a possible example of tools layout. In any case, the actual display window can change according to the configuration assigned to Tpaedi32.

The default window is divided into three areas: table of spindles, list of the fitted out spindles and list of the non fitted out spindles.
In the area **Table of the spindles** following data are described:

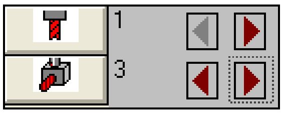
- **Fixture:** fixture number.
- **Machine:** machine number. A number of configured machines not exceeding 8 can be displayed.
- **Head -Group:** group number. The groups that can be selected are those concerning the configured **Machine**. For each machine no more than 10 groups can be displayed. The X,Y,Z correctors of the head group are displayed beside.
- **Display unit of measurement :** Measurement unit of the parametric data [mm] or [inch]. The measurement unit of the active program is suggested. If [mm] is chosen, the coordinates are displayed in in mm and the speed parameter in [mm/min] or [m/min] according to the configuration of Tpaedi32. If [inch] is chosen, the coordinates are displayed in inch and the speed parameters [inch/sec] or [inch/min] according to the configuration of Tpaedi32.

The dimension of the column table can be changed. The new positioning is saved and shown in every following opening.

The area **Fitted out spindles** is displayed when the spindles are assigned by the tooling system. To a spindle position can be related a tool, a tool-holder, an electrospindle position, that must be fitted out manually or automatically according to the plant specifications.

Tooling is indicated in the column . The technological spindle features are partly displayed in the table of the lower side of the window, where we see correctors, maximum rotation speed of the spindles (RPM) and partly in the table on the right, where diameter, lengths and working speed are displayed. When the spindle is fitted out with a tool-holder, all the information concerning the tooling are displayed in the lower part of the window in the table of no more of 9 single tools, that can be placed in a tool-holder.

In case of electrospindle the catalogues available for the tools and the tool-holders can be displayed.



The two-rows table allows to scroll the two catalogues. The numbers correspond with the positions effectively configured for the tools and the tool-holders. When an electrospindle is associated with a tool magazine (carousel, array), the selection of the button restricts the table to the tools and the tool-holder only, that are effectively assigned in the magazine.

Non fitted out Spindle
When in technological parametric the spindle are directly assigned, the column of the tooling does not appear in the window displaying the tools and all the information concerning a spindle are displayed in the corresponding row of the only table displayed.
The tables of the catalogues are never displayed.
The window of Technology can also be displayed during the insertion of a working clicking the button at the item Tool

By button the parameters referring to the selected tool are brought back in the data entry of the working that is being edited.

5.7 Plant

A plant consists of one or more machines. Machines (or modules) consist of assembly, divided into sub-assemblies or devices. Usually the plant is single, thus no change is provided for. Sometimes, more configurations are to be installed for different plants (independent or dependent between them).
Plant selection window can be recalled by ***Set->Plant*** menu or selection can be required when *Tpaedi32* is started. This is an optional item to be configured. It is only active when program is closed.
If Tpaedi32 is not installed in *Auxiliary Cad* mode*,* in the window of the plant choice the option **Update the plant selection** is proposed. If selected, the information about the plant are memorized in the tpa.ini file, so that at a following restart of Tpaedi32 the technological data concerning the last chosen plant are proposed once again.

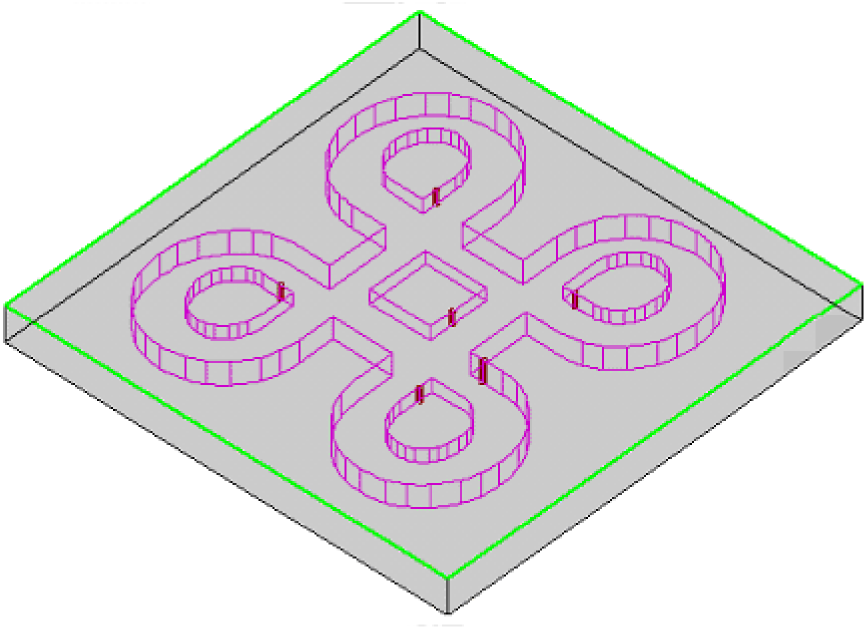
6 Piece

6.1 Graphical Representation of the Overall View

Piece Overall View

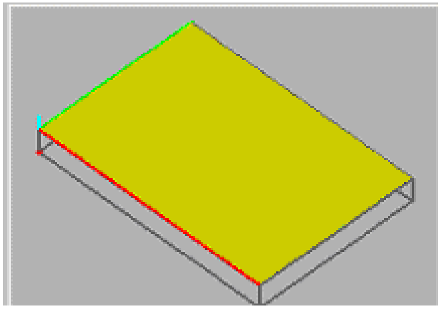
The three-dimensional piece is represented with any fictive faces assigned also externally to the basic parallelepiped: the representation is in the piece current view area.

The piece can be rotated, moved and enlarged or reduced. Workings are represented in the space to make visible their actual overall dimensions. Here is an example:

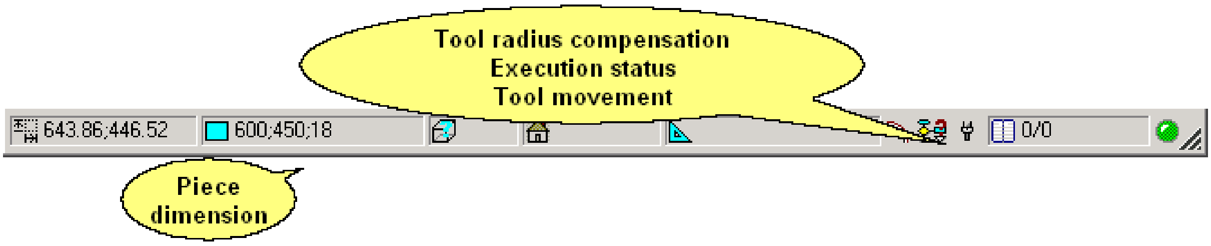


Piece Overall View

The three-dimensional piece is represented with all faces that the piece itself has assigned. The active face is highlighted, with indication of the reference system origin and the face axes.



Status Bar



Commands Visualization Area and Errors Visualization Area

The commands area displays the command exit status: for example, Creating a new program or Opening an existing program.

The errors area provides the complete list of the errors which have been detected during program processing.

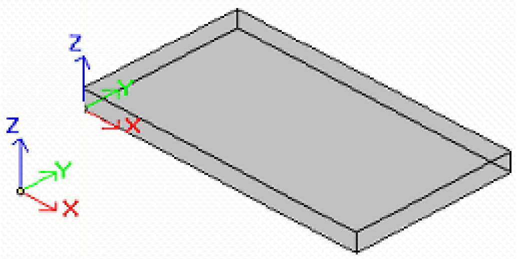
By **left-clicking** the mouse on a line in the list and pressing **[SHIFT]** or **[CTRL]** keys the contextual help is displayed with reference to the selected error.

6.2 Piece geometry

This piece is a parallelepiped object, featured by:

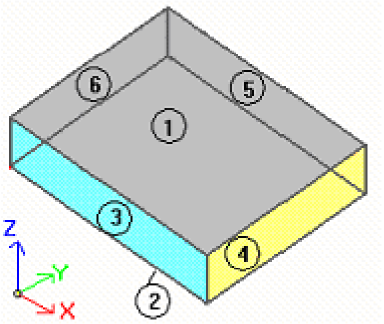
- three dimensions: length, height and thickness
- six faces.

Tpaedi32 uses a 3D system with fixed Cartesian coordinates, named **Reference Absolute System of the piece**, common to all pieces.
The Absolute Reference System is fixed and cannot be modified; it is assigned as shown in figure:



- axes are indicated like X, Y and Z
- system origin is located on piece left bottom edge
- X axis is associated to piece length (indicated by l) and has a positive direction to right
- Y axis is associated to piece height (indicated by h) and has a positive direction to piece inside part
- Z axis is associated to piece thickness (indicated by s) and has a positive direction upward.

Six parallelepiped faces are indicated like **real faces** and numbered 1 to 6. Figure below shows face **automatic** numbering. *Tpaedi32* can be configured to operate with a numbering different from the automatic one: in this case, a **custom** numbering is assigned, anyway using face numbers 1 to 6.
Automatic numbering is as follows:

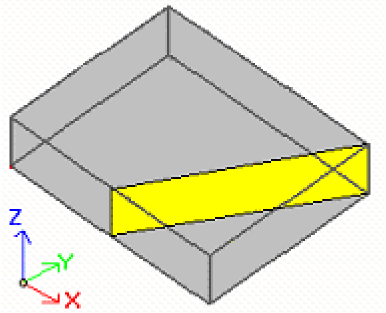


- top face is number 1
- bottom face is number 2

- front view face is number 3
- side view face is number 4
- face opposite to front view is number 5
- face opposite to side view is number 6.

Tpaedi32 can be configured not to manage one or more real faces.

In addition to six real faces, other faces can be assigned, usually located in the piece, and named **fictive**.



Fictive faces are numbered 7 to 99 and can be:

- internal, partially or totally external to the piece
- be inclined from three absolute points of piece.

Piece programming always refers to a face and uses 3D system with face Cartesian coordinates. Also in this case, three XYZ axes are used, where:

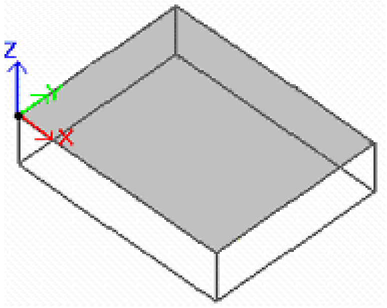
- the face plane assigns X and Y axes
- direction perpendicular to face plane assigns Z axis, which is indicated as depth axis.

In face reference system:

- X axis is associated to face length dimension (indicated by **lf**)
- Y axis is associated to face height dimension (indicated by **hf**)
- X axis is associated to face thickness dimension (indicated by **sf**).

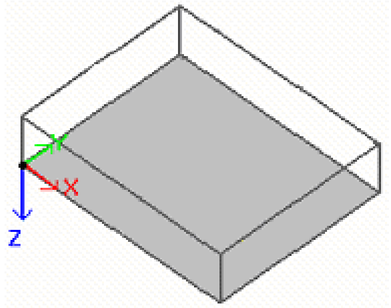
Reference systems for real faces are examined, as automatically assigned:

<u>Faces 1 and 2</u>:



Face 1:

face dimensions:
lf=l
hf=h
sf=s



Face 2:

face dimensions:
lf=l
hf=h
sf=s

Local systems of faces 1 and 2 are similar:

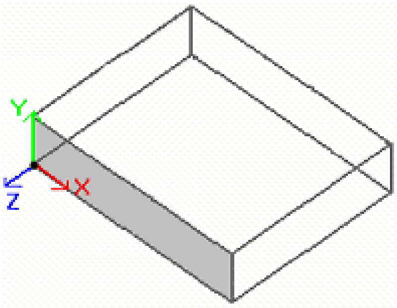
- X axis has the same orientation and direction as X axis in the Reference Absolute System of the piece

- Y axis has the same orientation and direction as Y axis in the Reference Absolute System of the piece
- Z axis has the same orientation and direction as Z axis in the Reference Absolute System of the piece, but opposite direction in face 2

Compared with Reference Absolute System of the piece, face point of origin:

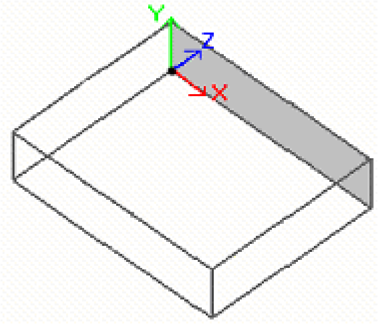
- is (0; 0; s) in face 1;
- is (0; 0; 0) in face 2.

Faces 3 and 5:



Face 3:

face dimensions:
lf=l
hf=s
sf=h



Face 5:

face dimensions:
lf=l
hf=s
sf=h

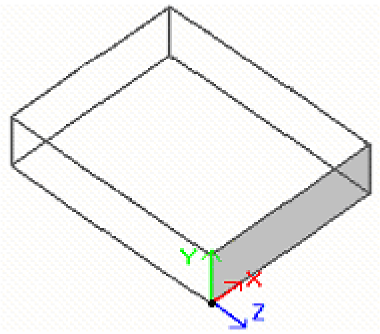
Local systems of faces are similar:

- X axis has the same orientation and direction as X axis in the Reference Absolute System of the piece
- Y axis has the same orientation and direction as Z axis in the Reference Absolute System of the piece
- Z axis has the same orientation and direction as Y axis in the Reference Absolute System of the piece, but opposite direction in face 3

Compared with Reference Absolute System of the piece, face point of origin:

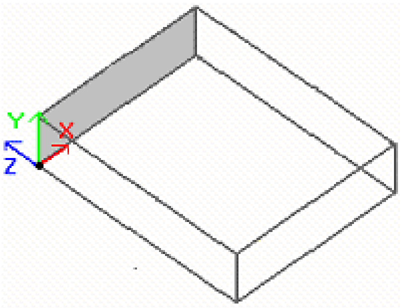
- is (0; 0; 0) in face 3
- is (0; h; 0) in face 5.

Faces 4 and 6:



Face 4:

face dimensions
lf=h
hf=s
sf=l



Face 6:

face dimensions
lf=h
hf=s
sf=l

Local systems of faces are similar:

- X axis has the same orientation and direction as Y axis in the Reference Absolute System of the piece
- Y axis has the same orientation and direction as Z axis in the Reference Absolute System of the piece
- Z axis has the same orientation and direction as X axis in the Reference Absolute System of the piece, but opposite direction in face 5

Compared with Reference Absolute System of the piece, face point of origin:

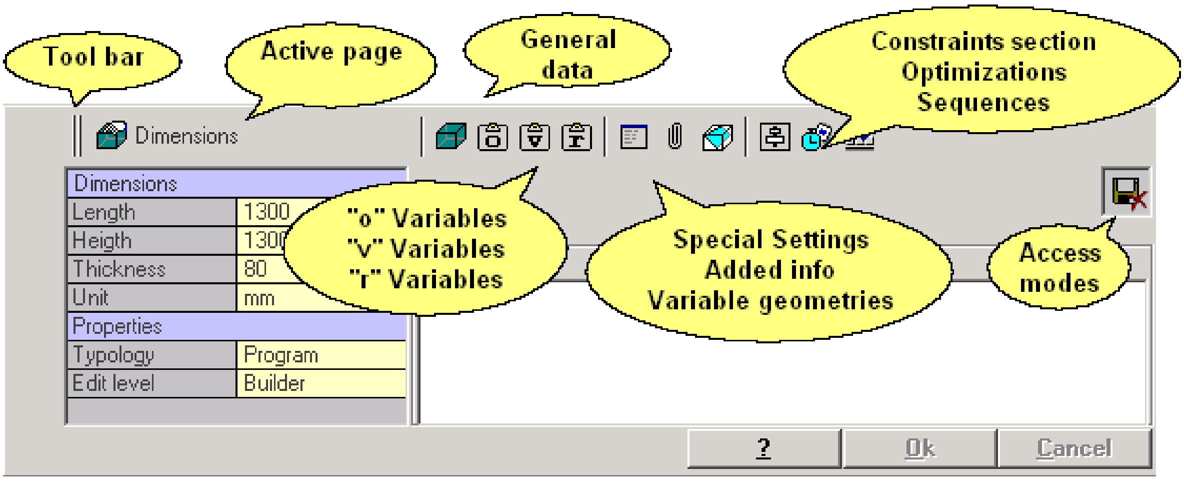
- is (l; 0; 0) in face 4
- is (0; 0; 0) in face 6.

Local systems of piece real faces can be locally assigned to Tpaedi32 configuration in a different way, by moving front XY plan origin on a different edge and/or rotating orientation of X and Y axes.

6.3 Assignments

6.3.1 Assignment Area

The assignment area in piece overall view consists of several pages, which can be activated by selection of the corresponding icon from **Tool bar**. Each page displays and allows to configure a set of piece assignments, usually arranged in tables.


Tool bar
Active page
General data
Constraints section Optimizations Sequences
Dimensions
"o" Variables "v" Variables "r" Variables
Special Settings Added info Variable geometries
Access modes
?
Ok
Cancel

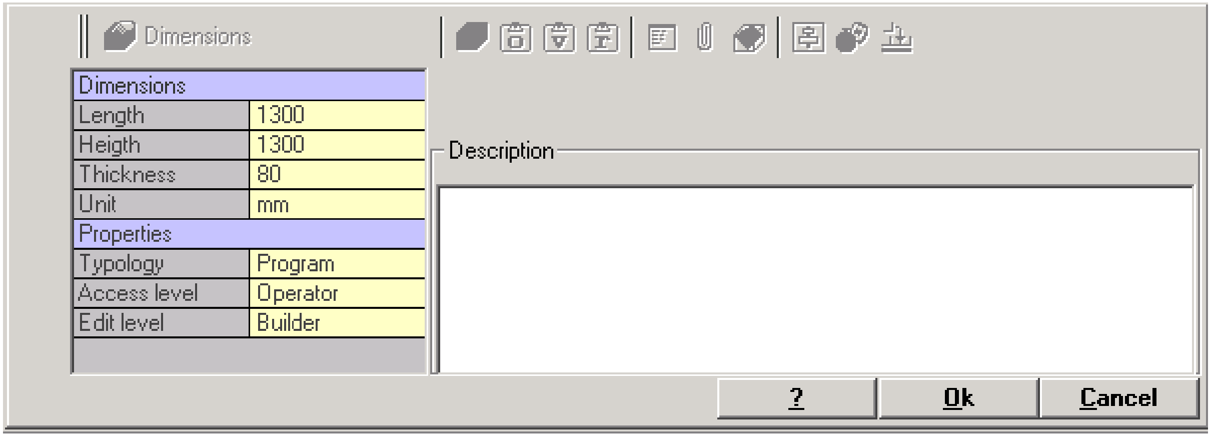
Dimensions	
Length	1300
Heigth	1300
Thickness	80
Unit	mm
Properties	
Typology	Program
Edit level	Builder

Access modes are displayed only if the password level for reading and/or writing the program is lower than that requested. In the example the program cannot be stored by the user. Also the **Edit level**, which is Constructor, is displayed. This means that only a user with Constructor password will be able to edit and save program modifications and, if necessary, modify the **Edit level** field.

Editing mode is always active. The buttons are activated by any editing. **[OK]** to confirm the editing and **[Cancel]** to cancel.
If a command from menu or toolbar is displayed, that asks the operator whether the inserted amendments should be saved.
The **[?]** button recalls the page of the help on line corresponding to the active page.

6.3.2 Dimensions

General information is provided such as dimensions and units of measurement, piece typology, access levels and comment.

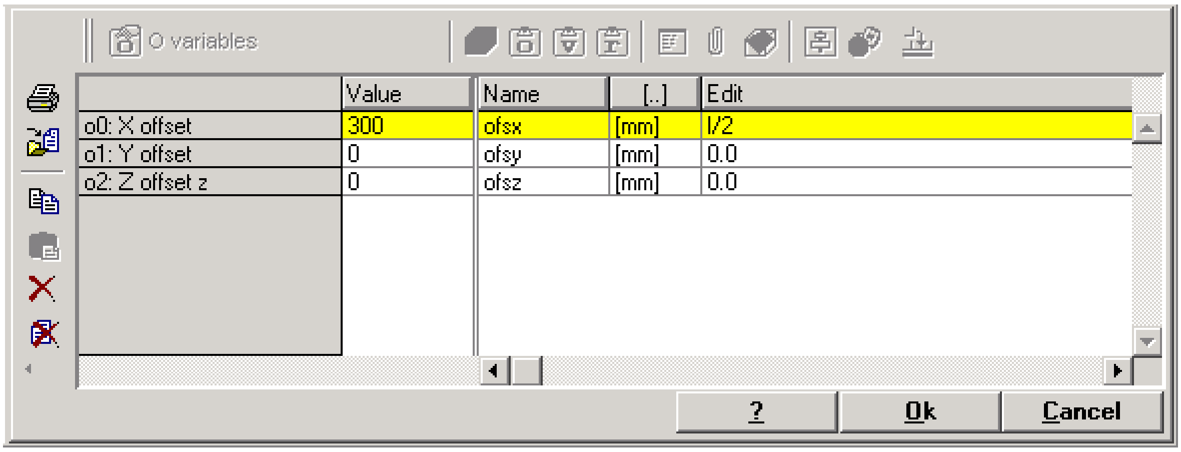


Dimensions	
Length	1300
Heigth	1300
Thickness	80
Unit	mm
Properties	
Typology	Program
Access level	Operator
Edit level	Builder

- **Length, Height, Thickness**: piece dimensions.
  - the three boxes accept positive numeric settings (>=0.0)
  - piece dimensions can be used, in the form of “l” (length), “h” (height), “s” (thickness) symbols for assigning variables or working parameters. (**See Chapter** *Parametric Programming*)
- **Unit**: unit of measurement of the piece ([mm] or [inch])
- **Typology**: working program, subroutine and macro, with possibility of modification. The macro typology is available only if the access level is equal or higher than Constructor. The sub-program typology can only be proposed, if during the configuration of Tpaedi32 its creation is allowed only from the not minimum access level.
- **Access and Edit level**: they assign respectively the minimum access level to open, modify and store a program. Can't set levels higher than the current access user level. If the current access level matches the Operator level these items are disabled
- **Description**: it is a comment to the program.

6.3.3 "o" Variables

It is an optional page
The table below lists the "o" variables which must not exceed the maximum number of 8.
In case of macro-program "o" variables are always no more than 8.



	Value	Name	[..]	Edit
o0: X offset	300	ofsx	[mm]	l/2
o1: Y offset	0	ofsy	[mm]	0.0
o2: Z offset z	0	ofsz	[mm]	0.0

- **Header** (Example: "o0: x Offset") it is composed of the name (o0) and the customized variable extended name ("x Offset")
- **Value**: it displays the value resulting from the solution of the expression defined in the **Edit** column. The field cannot be modified
- **Name**: it displays the symbolic name associated to the variable which can be used in Parametric Programming. The field cannot be modified and it is assigned by the machine manufacturer in configuring Tpaedi32. The column is not displayed if to any "o" variable has not been assigned a symbolic name in words.
- **[..]:** it displays the unit of measurement of the variable :
  - if the variable defines a dimension, the measurement unit is expressed in [mm] or [inch]
  - if the variable defines a speed, the measurement unit can be expressed in [m/min] o [mm/ min] o [inch/sec] o [inch/min] according to the configuration of Tpaedi32
  - if the variable is dimensionless, the field unit of measurement cannot be assigned. The field cannot be modified. The column is not displayed if none of the "o" variables has an assigned variable dimension.
- **Edit**: it is the field where the variable value is assigned. This field can be modified and it may contain a number or a numeric expression. The maximum length of the field is 100 characters. In Figure you can see an example of assignment of parametric expression: the o0 variable is set ="l/2", where l indicates the length of the piece. The value resulting from the calculation of the expression is 300, as given in the **Value** column. An example of numeric expression is ="600/2", which produces the following result Value = 300.
- **Description:** it displays a description of the variable. The description is defined from the machine producer during its configuration. The column is not displayed when no description exists.

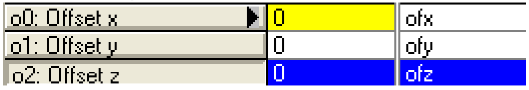
The "o" variable setting can be parameterized only using piece dimensions (l, h, s).

The buttons bar which appears on the left-hand side of the table shows the commands available for editing variables:

Print variables with the relevant assignments

Import variable assignments from a selected program

Copy the selected variable settings to Clipboard. The copied variables are available to perform a following Paste command into the same program or another.
To select one or more variables, click on the corresponding line by pressing the **[Ctrl]** key.
Selecting the line, the head button is depressed with white writing on blue background. The figure is an example of 02 selected variable.



o0: Offset x	0	ofx
o1: Offset y	0	ofy
o2: Offset z	0	ofz

To deselect one or more variables, click on the corresponding line while pressing the **[Ctrl]** key.

Where the line is now deselected, the selection button is up and the colours are restored. To delete the selection from the whole list, click on any position of the table. In this way the line on which you have clicked becomes the selected line.

Paste the variable settings previously copied to Clipboard, by respecting variable names as follows: 'o0' assigns the 'o0' variable. It is enabled only if a copy of one or more "o" variables is available in Clipboard.

Reset the setting of the selected variables.

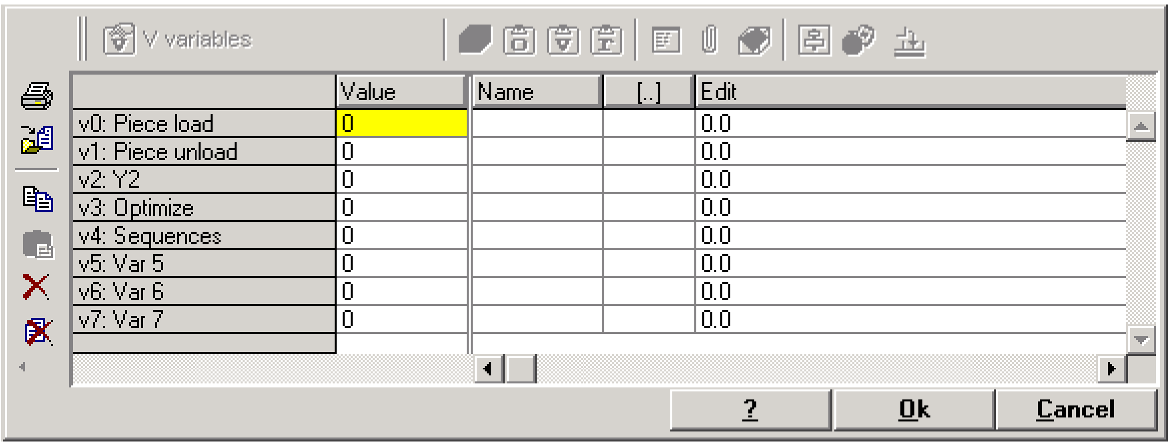
Reset the setting assigned to all variables

The errors area displays the list of the only errors detected during "o" variables assignment.

During programming, it is possible to recall an immediate help for those functions and variable arguments which are available for parametric programming.

6.3.4 "v" Variables

It is an optional page
The table below lists the "V' variables which must not exceed the maximum number of 8.
In case of macro-program "V' variables are always no more than 8.



	Value	Name	[..]	Edit
v0: Piece load	0			0.0
v1: Piece unload	0			0.0
v2: Y2	0			0.0
v3: Optimize	0			0.0
v4: Sequences	0			0.0
v5: Var 5	0			0.0
v6: Var 6	0			0.0
v7: Var 7	0			0.0

- **header** (Example: "v0: Piece load") it is composed of the name (v0) and the customized variable extended name ("Piece load ");
- **Value**: it displays the value resulting from the solution of the expression defined in the **Edit** column. The field cannot be modified
- **Name**: it displays the symbolic name associated to the variable which can be used in Parametric Programming. The field cannot be modified and it is assigned by the machine manufacturer in configuring Tpaedi32. The column is not displayed if to none of the "V' variables has been assigned a symbolic name in words.
- **[..]:**it displays the unit of measurement of the variable:
  - if the variable defines a dimension, the measurement unit is expressed in [mm] o [inch]
  - if the variable defines a speed, the measurement unit can be expressed in [m/min] o [mm/min] o [inch/sec] o [inch/min] according the configuration of Tpaedi32 ,
  - if the variable is dimensionless, the field unit of measurement cannot be assigned.

  The field cannot be modified. The column is not displayed, if none of the "V' variable has an assigned variable dimension.
- **Edit**: it is the field where the variable value is assigned. This field can be modified and it may contain a number or a numeric expression. The maximum length of the field is 100 characters.
- **Description:** it displays a description of the variable. The description is defined from the machine

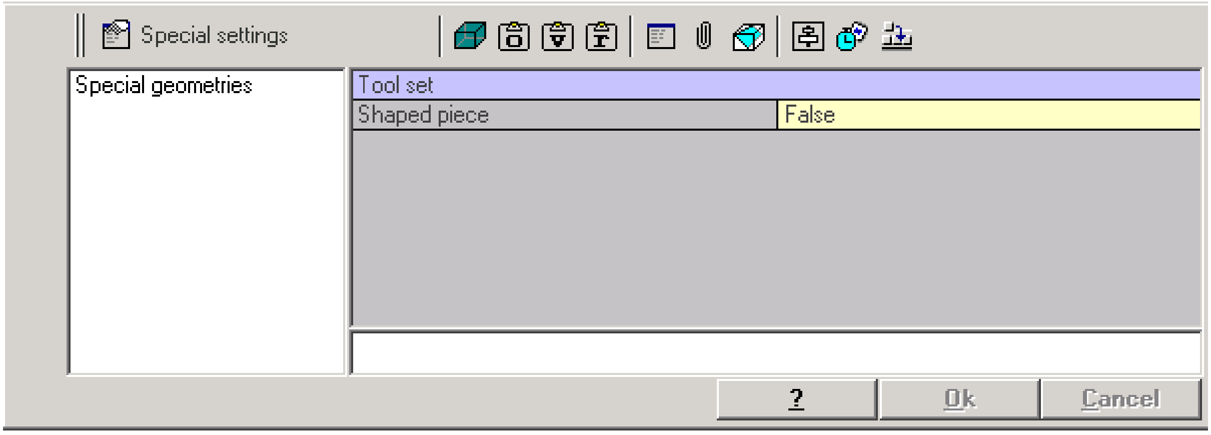
producer during its configuration. The column is not displayed when no description exists.
The "v" variable setting can be parameterized only using piece dimensions (l, h, s).

The buttons bar which appears on the left-hand side of the table shows the commands available for editing variables. (See Paragraph ***"o" Variables***).
The errors area displays the list of the only errors detected during "v" variables assignment.

During programming, it is possible to recall an immediate help for those functions and variable arguments which are available for parametric programming.

6.3.5 Special Settings

It is an optional page.
They are included in the remarkable information section for which Tpaedi32 must activate particular authentication measures and procedures, together with information of exclusively custom type.
Custom assignments have a meaning which is unknown to the application and are configured by the machine manufacturer in tailoring the program.
Even the section title, here for example is Special Settings, may be different, since it can be reassigned at custom level.



Special geometries

Tool set	
Shaped piece	False

In the left-hand part of the figure the list of pages containing information (no more than 9 pages) is displayed.
In the right-hand part the list of information on the active page is shown.
A single item (information) can be displayed in form of:

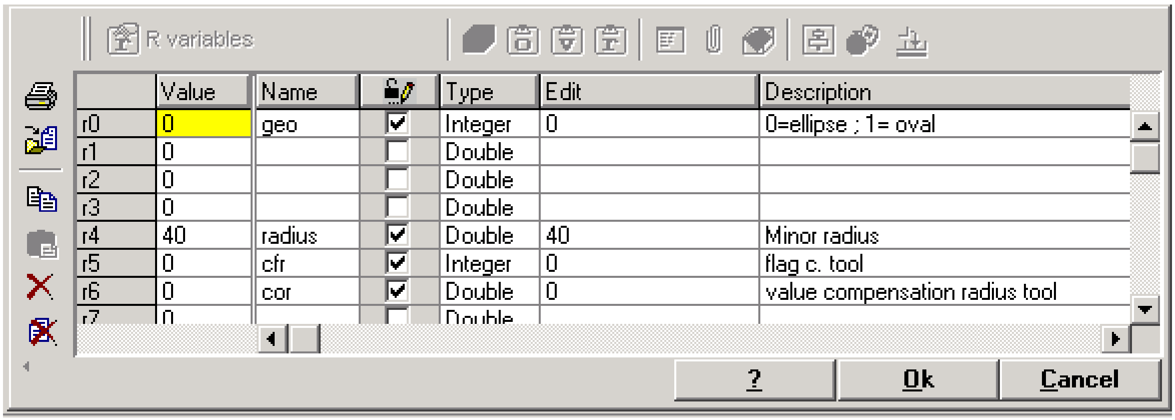
- direct edit field of numeric value of double type (example: "100.5")
- direct edit field of numeric value of integer type (example: "12")
- direct edit field of numeric value which can be modified in parametric form (example: "l-100")
- check box on a list
- box of list of values to be sorted
- selected color from a palette of colors
- searching file field. The opening file window is displayed.

Moreover, for each single item a Help text can be assigned: it is displayed in the white frame above buttons. For any more information about the meaning of each individual item provided in the window please contact the machine Manufacturer.

The buttons bar on the left is associated with the available commands for section modifying. In particular, the bitmap : initializes to the default settings section. The bitmap initializes to the default settings the only selected page of the sections.
Default settings are read by the program PIECE.TCN, used also as a prototype to create new programs.

"r" Variables

The table below lists the "r" variables always in number of 300:



	Value	Name	🔒	Type	Edit	Description
r0	0	geo	☑	Integer	0	0=ellipse ; 1= oval
r1	0		□	Double		
r2	0		□	Double		
r3	0		□	Double		
r4	40	radius	☑	Double	40	Minor radius
r5	0	cfr	☑	Integer	0	flag c. tool
r6	0	cor	☑	Double	0	value compensation radius tool
r7	0		□	Double		

- **header** (Example: "r0 ") it shows the variable name
- **Value**: it displays the value resulting from the solution of the expression defined in the **Edit** column. The field cannot be modified
- **Name**: it displays the symbolic name associated to the variable which can be used in Parametric Programming. The syntax is valid if the maximum length is 16 alphanumeric characters with lower-case characters. If a name has already been assigned to another "r" variable, it is not accepted.
- 🔒: it enables or disables the possibility to reassign a variable. This operation can be carried out during program execution and when the program is used as subroutine. In the first case, for example, it is assumed that r0 assigns a variable position for the positioning of a drilling working. When the program execution is recalled, it will be possible to change the r0 value. In the second case, by using the same example, when the subroutine is recalled into another program, it is possible to change the r0 value.
  A variable which cannot be reassigned is used for program definition auxiliary settings. The variables which cannot be reassigned usually use those which can be reassigned (for tests, assignments). It can be stated that a reassignable variable is global, while a variable which cannot be reassigned is local.
- **Type**: it assigns the type of variable. Two numeric types (Double, Integer) are available and one non-numeric type. (String). See Chapter ***Parametric Programming->Variables and Parameters of String Type***.
- **Edit**: it is the field where the variable value is assigned. The field can be modified and it may contain a number or a numeric expression or an alphanumeric expression. The maximum length of the field is 100 characters.
- **Description**: it is the comment to the variable.

The "r" variable setting can be parameterized by:

- piece dimensions (l, h, s),
- "o" and "v" (o0 – o7, v0 –v7) variables,
- "r" variables in the table above (example: r15 can use "r" variables from r0 to r14).

For a more detailed knowledge of the possibilities of variable parametric expression refer to the chapter concerning the Parametric Programming.
An unset variable (Edit box empty) has Value = 0.0 and Type = Double and cannot be reassigned.
The "r" variable type is not fixed, but it can be:

- Double: numeric type variable; the calculated Value keeps the decimal part. Examples of use are working positions, movement speed
- Integer: numeric type variable similar to the previous case, but the calculated Value resets the decimal part. Examples of use are counters, working selection, rotation speed
- String: non-numeric type variable used, for example, to assign the subroutine name. Also the

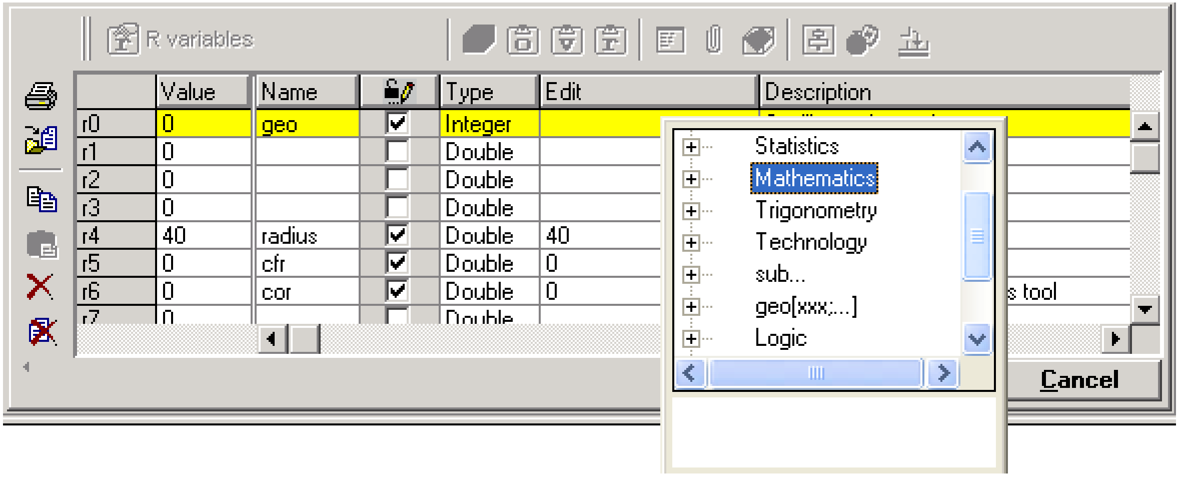
calculated Value is of String type.

The buttons bar that appears on the left-hand side of the table shows the commands available for editing variables.
The errors area displays the list of the only errors detected during "r" variables assignment.

Guided Edit

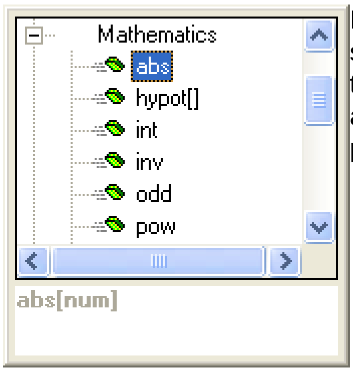
During programming, it is possible to recall an immediate help for functions, variable arguments and "r" variables which are available for parametric programming.

Let us see as example the "r" variable assignment.
The Figure below shows the edit case of the r0 variable.



	Value	Name		Type	Edit	Description
r0	0	geo	☑	Integer		
r1	0		□	Double		
r2	0		□	Double		
r3	0		□	Double		
r4	40	radius	☑	Double	40	
r5	0	cfr	☑	Double	0	
r6	0	cor	☑	Double	0	s tool
r7	0		□	Double		

To open the immediate help window press the **[CTRL + spacebar]** key combination. The available functions and arguments are grouped into nodes. Select the node you wish, open the node and scroll the items in the list and confirm by pressing the **[ENTER] key** or **double-clicking** the item. The string matching the selected item will be automatically entered into the edit box, at the cursor position.



In the figure the Mathematic node is open and the function abs is selected. In the area at the end of the list of functions the syntax related to the selected function is displayed. By clicking the [F1] button the application file guide is opened to the page matching the description of the parametric function required.

Once the data have been confirmed, the corresponding string, which corresponds to the selected item, will be automatically entered in the edit field, at the cursor position
The list items are of two types, distinguished by different icons:

geo[alfa;]: example of display of a function

cnq example of display of a variable argument

If the cursor is positioned on a function name, the **[CTRL+I]** key combination shows the function syntax.

In case the cursor is positioned on a function name, the **[CTRL+J]** key combination starts the application guide file, on the relative section
It is also possible to ask for help on "r" variables assigned in the program, by the **[CTRL+R]** key combination.

By the key combination **[CTRL+F1]** in a edit field a local menu is opened including the above mentioned commands.

Recovering of "r"Variables from an existing program

In buttons bar selecting the bitmap the entire list of "r" variables list can be imported from another program.

If it should be necessary to recover only some variables of an existing program, commands of Copy and Paste must be used according to the procedure, as follows :

- close the running program, after memorizing it, if necessary
- open the program, which the variables must be copied from
- open the page of the"r" variables and select the variables to import. For instance, from r5 to r9.
- select the bitmap to copy the selected variables into the Local Notes
- open the program which the variables must be imported into.
- open the "r" variables page.
- select the bitmap . Variables previously copied are pasted. In particular, the variables from r5 to r9 are overwritten, according to the example.

6.3.6 Added Info

It is an optional page .
It is included in the custom information section. The configuration of assignments is supported by a kind of dedicated function and their meaning is unknown to the program.
What previously described about custom information in Special Settings is always valid

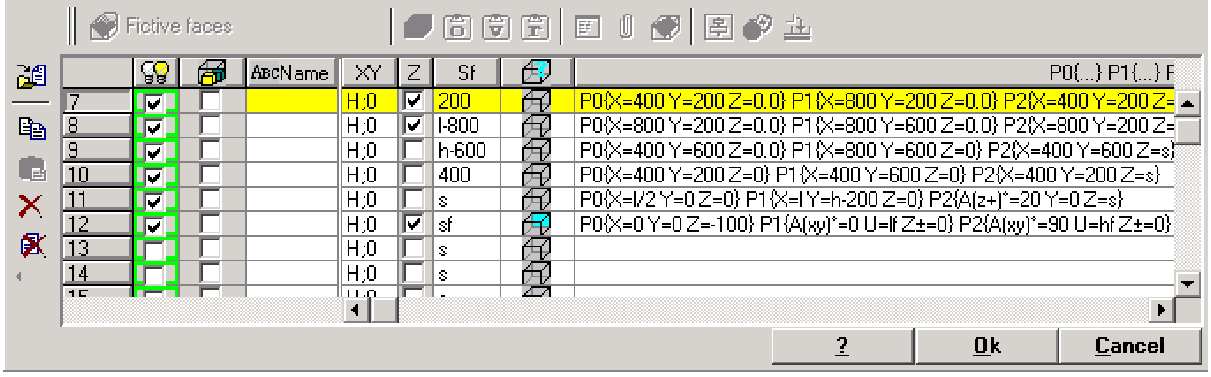
6.3.7 Fictives faces

It is an optional page.
A fictive face is a face that does not belong to a parallelepiped piece. For pieces with complex shapes, such as, for instance, the cavity for the glass in a door, the definition of new faces must be set. The fictive faces are progressively numbered from 7 to 99.
A fictive face can be defined as similar to another face, when the triad of faces by

- translation in every direction and/or
- rotation only on the plane of a face.

The planes of the two faces must be parallel and the Z semi-axis must overlap.
Workings and technology, that are applied to the real face, can also be applied on a fictive face similar to one of the 6 real faces of the piece.

The assignment of a face is fully independent of the assignment of a reference face. In fact, the similarity sizes up the geometric conditions of the face after her defining, while the reference face is a construction face that cannot be programmed and considered into the geometry of the piece.



Fictive faces

			AbcName	XY	Z	Sf		P0{...} P1{...} F
7	☑	□		H;0	☑	200		P0{X=400 Y=200 Z=0.0} P1{X=800 Y=200 Z=0.0} P2{X=400 Y=200 Z=
8	☑	□		H;0	☑	l-800		P0{X=800 Y=200 Z=0.0} P1{X=800 Y=600 Z=0.0} P2{X=800 Y=200 Z=
9	☑	□		H;0	□	h-600		P0{X=400 Y=600 Z=0.0} P1{X=800 Y=600 Z=0} P2{X=400 Y=600 Z=s}
10	☑	□		H;0	□	400		P0{X=400 Y=200 Z=0} P1{X=400 Y=600 Z=0} P2{X=400 Y=200 Z=s}
11	☑	□		H;0	□	s		P0{X=l/2 Y=0 Z=0} P1{X=l Y=h-200 Z=0} P2{A(z+)°=20 Y=0 Z=s}
12	☑	□		H;0	☑	sf		P0{X=0 Y=0 Z=-100} P1{A(xy)°=0 U=lf Z±=0} P2{A(xy)°=90 U=hf Z±=0}
13	□	□		H;0	□	s		
14	□	□		H;0	□	s		

The buttons bar which is located on the left-hand side of the table displays the commands which are available for editing fictive faces:

Import assignments for variable geometries from another program. The *File Open* window is activated and the preview for the only variable geometries is available in order to *see* the faces as assigned

Copy the settings of the current face to Clipboard

Paste the settings of the current face previously copied in Clipboard

Clear the setting of the current face

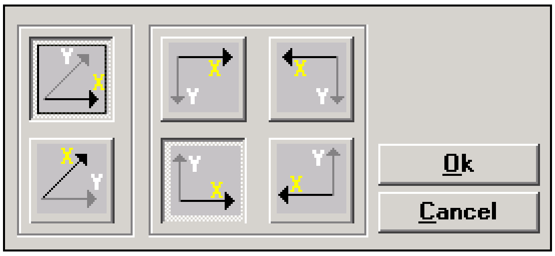
Clear the setting of all fictive faces which do not have programmed workings

The errors area displays the list of the only errors detected during assignment of fictive face geometries.

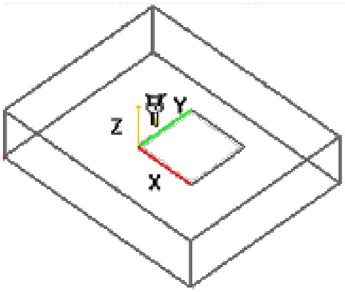
The graphic representation of the fictive face selected in the table is performed in the piece overall view area. If this one is not visible, the representation is performed in the overall view area, after deleting the graphic view of the programmed workings. Exiting the fictive face setting window, the graphic representation of the program is updated.

The Fictive Faces page displays the complete list of faces which can be assigned (face 7 to face 99), since it is not necessary to define them sequentially.
The table above lists the faces which can be assigned:

- **header** (Example: "7") it provides the face number
- : if selected, it enables face assignment
- :if selected, it enables the use of the face only as auxiliary face for building one or more faces. A building auxiliary face cannot be programmed, nor be considered in the piece geometry
- ABC **Name**: it shows the face name
- **XY**: it opens a window for assigning the horizontal or vertical representation of the X axis. The column appears only if enabled.



- **Z**: it sets the direction of the z axis in air with respect to the xy plane of the face. **Z** is the depth axis and it is perpendicular to the xy plane assigned to the face concerned. If selected, it assigns a left-handed or a right-handed coordinate system (xyz). The here assigned direction indicates how the tool works.



- With the positive Z axis (Z+) upwards, the tool enters the piece from above: in this case we are talking about a right-handed coordinate system (it follows the right-hand rule, with: x axis on the thumb, y axis on the index finger, z axis on the middle finger);
- With the Z axis orientation opposite to that shown in Figure (Z+ downwards) the tool enters the piece from below: in this case we are talking about a left-handed coordinate system (it follows the left-hand rule, with: x axis on the thumb, y axis on the index finger, z axis on the middle finger);

Using the parametric programming, the axis orientation is returned by the Multi-Purpose Geometry Library Function geo[zface; nside]

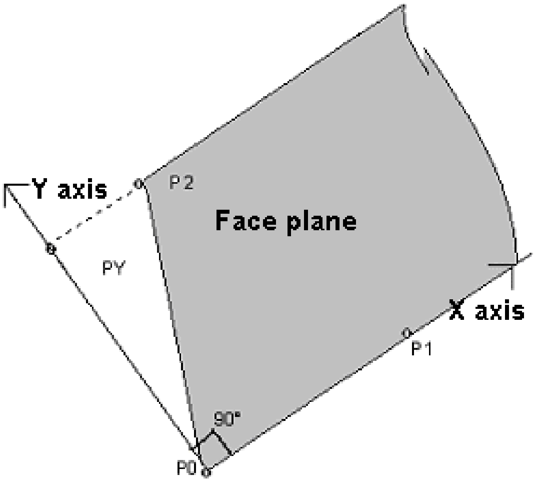
The column appears only if enabled.

- **Sf**: it defines the thickness of the face. For unassigned faces, the field is preset to the default value = "s", corresponding to the thickness of the piece.

  Thickness setting can be parameterized with:
  - piece dimensions
  - "o" and "v' Variables
  - "r" Variables

  The column appears only if enabled. Using the parametric programming the thickness of the face is returned is returned by the Multi-Purpose Geometry Library Function geo[sface;(nside)].
- ▣**:** it displays the icon of the reference face in assigning the fictive face matching the line. It cannot be modified here. The column appears only if enabled.
- **Pr1,Pr2,Pr3** sets 3 additional face parameters. Values assignment can be parameterized according to the rules valid to the assignment of the **Sf** thickness.The values are interpreted like coordinates. The column is optional. Using the parametric programming the 3 parameters are returned by the Multi-Purpose Geometry Library Function geo[pr1; nside], geo[pr2; nside], geo[pr3; nside].
- **P0{...} P1{...} P2{...}** assigns the coordinates of the three face points in a dedicated window.

  A fictive face is always identified by three different and non-aligned points:



- P0 is the origin of the xy plane of the face
- P1 is the point which defines the direction of the x+ axis;
- P2 is the third point on the xy plane:
  - if the line passing through P2-P0 is perpendicular to the line passing through P0-P1: P2 is the point which defines the direction of the y+ axis;
  - otherwise: the point which defines the direction of the y+ axis in PY is identified.

The P0-P1 distance assigns the face length.
The P0-PY distance assigns the face height.

From the operational point of view, the points shown in Figure are not always known. The exact values of the coordinates of the three points are not always known, but it is known for example:

- how inclined the face is to another, or
- how high is the face, or
- where the face plane ends.

The assignment modes adopted to fix the three face points aim to satisfy all these options. In the following paragraphs several examples for defining fictive faces are given.

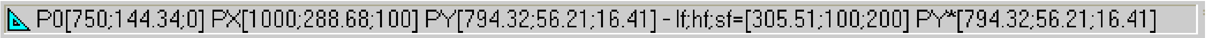
- : set the face which is being edited in such a way that it is similar to the selected reference face. This option is disabled in case the reference face is Overall View.
- ⚠: the **Errors** area displays any possible error indications detected in defining the fictive face
- **Command Bar**:

Multiply z axis: if active, for the representation of the z axis of the piece it applies a multiplying factor, as set in program configuration; if inactive, the z axis of the piece is represented in 1:1 scale. The application of an amplifying scale for the z axis is justified by the fact that the thickness of a piece is often small, compared to the length and height dimensions; therefore, the graphical representation in 1:1 scale would make it little visible. In any case it is necessary to keep into account also the distortion effect which amplification introduces, mostly in angle evaluation.

For all the other commands of the Command bar refer to Chapters How To Configure the Graphical Representation->General Graphics Commands, How To Configure the Graphical Representation->View Check and How To Configure the Graphical Representation->Three-Dimensional Representation

Information concerning the fictive faces in the status bar

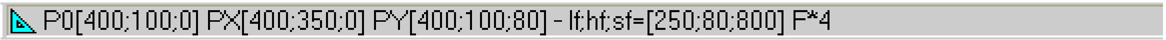
Below the status bar showing a selected face line:



P0[750;144.34;0] PX[1000;288.68;100] PY[794.32;56.21;16.41] - lf;hf;sf=[305.51;100;200] PY*[794.32;56.21;16.41]

P0[...]	face's point of origin
PX[...]	edge point along x+ axis
PY[...]	edge point along y+ axis (calculated)
lf,hf,sf=[...]	dimensions of the face
PY*	programmed point on y-face appearing only if it does not coincides with the edge point calculated along the y+ axis.

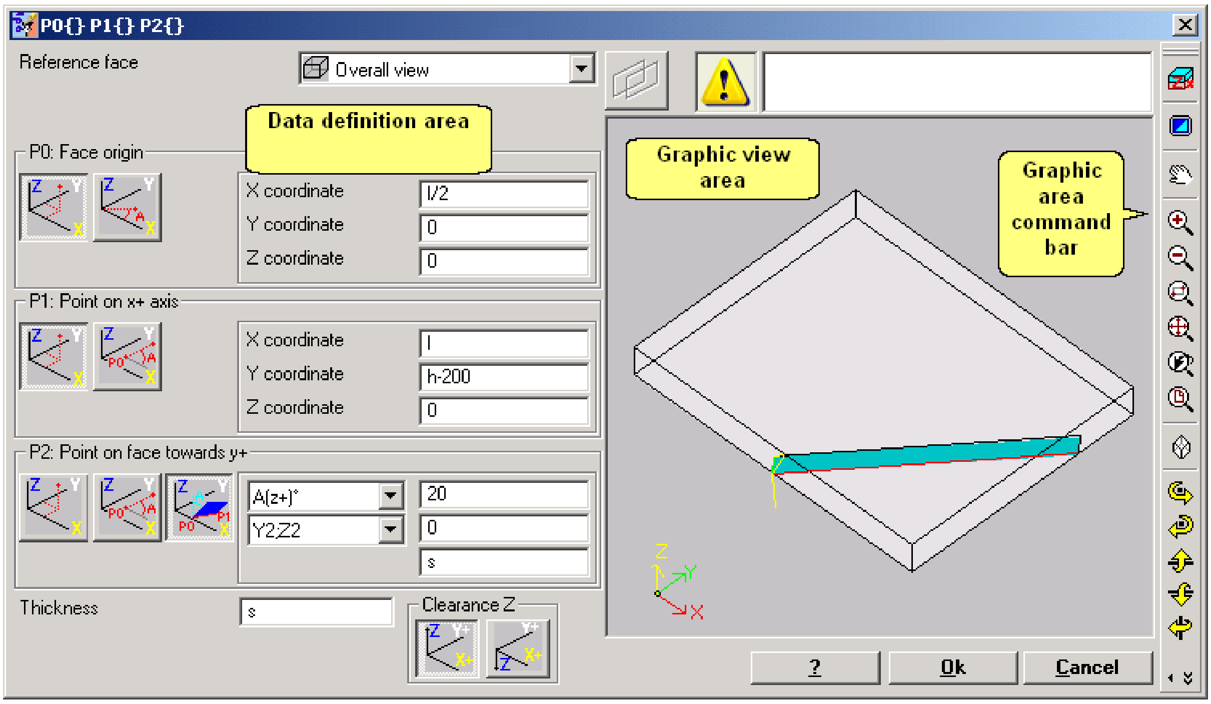
A second example shows the parameters of a fictive face similar to the face 4 (F*4):



P0[400;100;0] PX[400;350;0] PY[400;100;80] - lf;hf;sf=[250;80;800] F*4

The indication of similar face only appears, if the similarity identification of the geometric variables during the Configuration of Tpaedi32 is enabled by the machine constructor.

Example 1



P0{} P1{} P2{}

Reference face: Overall view

P0: Face origin

X coordinate	l/2
Y coordinate	0
Z coordinate	0

P1: Point on x+ axis

X coordinate	l
Y coordinate	h-200
Z coordinate	0

P2: Point on face towards y+

A(z+)°	20
Y2;Z2	0
	s

Thickness: s

Clearance Z

- **Reference face**: it selects the xyz system to assign to the fictive face to be defined. It may be the absolute cartesian system of the piece or the xyz system of another face.
- **P0: Face origin**: the x, y, z coordinates of the origin of the fictive face (P0 point) are assigned in cartesian (first selected bitmap) or polar (second selected bitmap) coordinates
- **P1: Point on x+ axis**: the coordinates of P1 are assigned in cartesian (first selected bitmap) or polar (second selected bitmap)
- **P2: Point on face towards y+**: the coordinates of P2 are assigned in cartesian (first selected bitmap) or polar (second selected bitmap) coordinates or the rotation of the p0-p1 segment with respect to an axis is assigned. If this last option is chosen, data in the left-hand drop-down box allows to select one of the 6 coordinated semi-axes of the piece:
  - A(z+)° , A(z-)°
  - A(x+)° , (x-)°
  - A(y+)° , A(y-)°.

  The selected semi-axis assigns the reference Y axis of the face (with origin in P0 and y+ direction on the selected semi-axis). The value sets the rotation angle (in degrees) of the y+ axis of the face around its own x axis: the axis rotates in positive direction towards the z+ axis of the face (selection on **Clearance Z**)

  In the example:
  - the reference y+ axis of the face is assigned as Z+ of the piece;
  - the rotation angle is 20°:
    - with *Z in air* in the right-handed coordinate system: the face plane rotates outwards of the Figure
    - with *Z in air* in the left-handed coordinate system: the face plane rotates inwards of the Figure.

  Up to now the face plane has been assigned, but the P2 point is still to be positioned: the y+ semi-axis is fixed, but not the position of P2.

The other drop-down box allows to select among different modes to complete the P2 assignment:

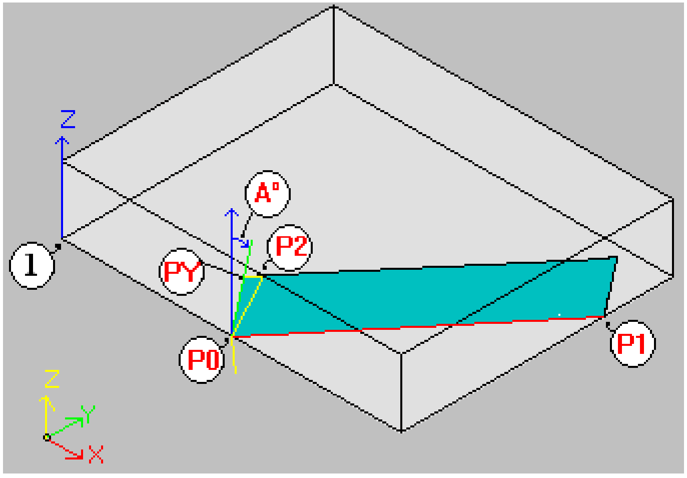
- hf : it assigns the face height: the P2 point falls on the y+ axis, at the assigned distance (field on the right-hand of the drop-down box). The assigned value is taken in absolute value:
- *X2;Y2*: it assigns the X and Y coordinates of P2, while the z coordinate is calculated provided that the point belongs to the face plane

- *X2;Z2*: it assigns the X and Z coordinates of P2, while the y coordinate is calculated provided that the point belongs to the face plane
- *Y2;Z2*: it assigns the Y and Z coordinates of P2, while the x coordinate is calculated provided that the point belongs to the face plane

By assigning two of the coordinates, the P2 point usually falls out of the y+ axis. In Figure the selection made is *Y2;Z2:*

- Y2 is set in the field on the right of the box ("0");
- Z2 is set in the field on the right-hand side, but below the box ("s").

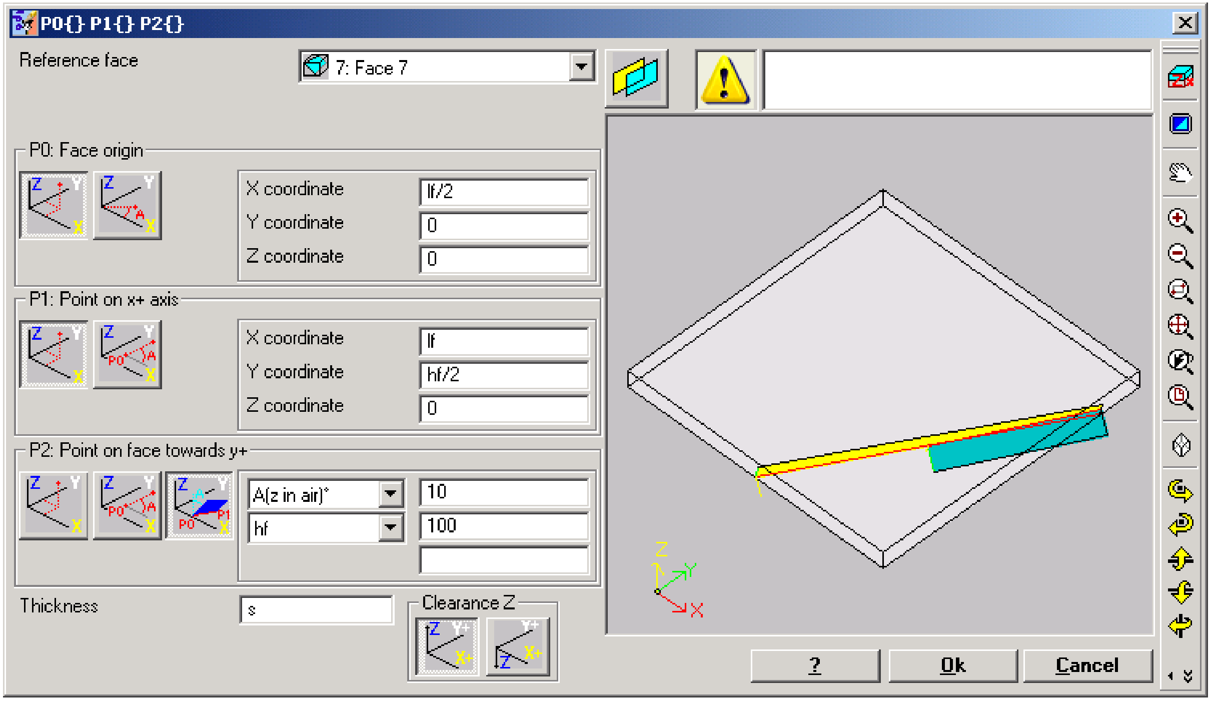
The Figure shows how to operate to build the face with the assigned parameter settings:



- The piece is displayed with the axes oriented as in the representation of the cartesian coordinate system shown in the lower left part of the Graphic View Area (the 1 point is the origin of the axes)
- the face origin is P0 and the x+ axis is between P0 and P1
- a semi-axis oriented as Z+ of the piece is drawn from P0: it is the y+ axis of the face, with null rotation
- The selection of the right-handed cartesian coordinate system orients the z+ axis of the face outwards of the Figure
- the y+ axis as identified is rotated of A° (20°-> positive value-> it rotates towards the z+ axis of the face)
- the y and z coordinates fix the P2 point on the left-hand side face of the piece: P2 forms with the x axis of the face (P0 to P1) an angle smaller than 90°, therefore the P2 projection on the y axis of the face (PY) is recalculated. The linear segments which join the origin of the face to P2 and P2 to PY indicate exactly that the P2 and PY points do not coincide.

Example 2

In the following example a fictive face already defined as reference face is configured:



P0{} P1{} P2{}

Reference face: 7: Face 7

P0: Face origin
X coordinate: lf/2
Y coordinate: 0
Z coordinate: 0

P1: Point on x+ axis
X coordinate: lf
Y coordinate: hf/2
Z coordinate: 0

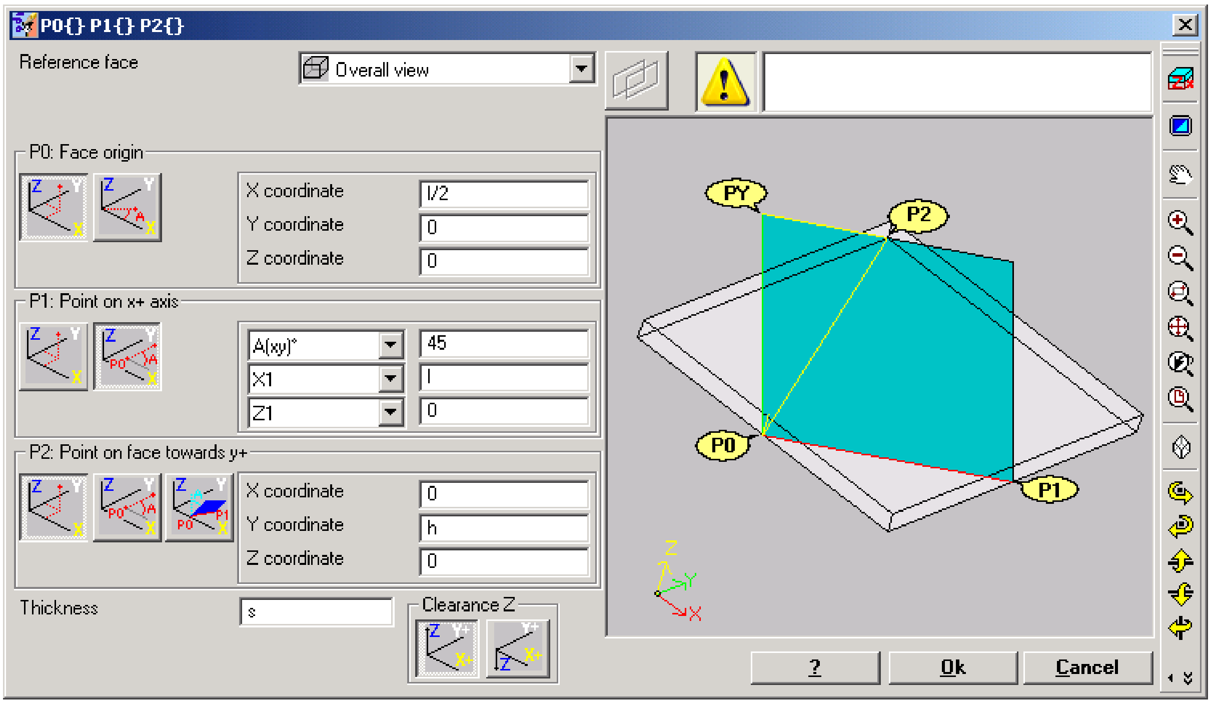
P2: Point on face towards y+
A(z in air)°: 10
hf: 100

Thickness: s

Clearance Z

- **Reference face**: Face 7 is configured. The selection means that the fictive face is to be assigned by using the xyz system of another fictive face: face 7. We assume that face 7 is that assigned in the previously example (Example 1). The drop-down box of the reference face provides a list of faces:
  - all real faces assigned on the piece
  - the fictive faces assigned on the piece with lower number than that of the fictive face to be defined. If, for example, face 8 is being assigned it is possible to select a real reference face or, if fictive, only face 7
- **P0: Face origin**: the first bitmap (from left) is selected in the frame. The selection means that the three coordinates of the point are known, but now it is about coordinates assigned on face 7. The coordinate fields position the face origin on the half of the x axis of face 7 (lf/2; 0; 0):
  - the adoption of *lf, hf, sf* (face length, height and thickness) variable arguments in parametric programming leads to the use of the dimension values of face 7
  - attention: the *Z Coordinate* value, if different from 0, uses the same conventional signs which are valid for faces (negative or positive in face working)
- **P1: Point on x+ axis**: also in this case, the first bitmap (from left) is selected in the frame. The selection means that the three coordinates of the point are known and, also for this option, they are assigned on face 7. The coordinate files position the P1 point to (lf; hf/2; 0);
- **P2: Point on face towards y+**: the third bitmap (from left) is now selected in the frame: the selection means that the inclination of the face to one of the coordinated axes of the reference face (face 7) is known. Assignments are similar to those of the previous example (Example 1):
  - **A (z in air)°**: it matches the already described A(z+)° selection, but in this case the message means that the choice falls on the z in air semi-axis (in the same way: *A(z piece)°* corresponds to the already explained A(z-)° selection; but now the message indicates that the choice falls on the z semi-axis in piece working);
  - **hf**: the P2 point is now designated by setting the face height.

Example 3



P0{} P1{} P2{}

Reference face: Overall view

P0: Face origin

X coordinate: l/2
Y coordinate: 0
Z coordinate: 0

P1: Point on x+ axis

A(xy)°: 45
X1: l
Z1: 0

P2: Point on face towards y+

X coordinate: 0
Y coordinate: h
Z coordinate: 0

Thickness: s

Clearance Z

- **Reference face**: Overall View is set. The choice means that the fictive face is to be assigned by using the absolute xyz system taken for each piece
- **P0: Face origin**: the first bitmap (from left) is selected in the frame. The choice means that the three coordinates of the point are known. The (l/2; 0; 0) coordinate fields place the face origin on the x half-axis of the piece;
- **P1: Point on x+ axis**: the left-hand bitmap is selected in the frame. The choice means that the polar coordinates of the point on one of the three cartesian planes of the piece are known (if a reference face were set, we could say: "…on *one of the three cartesian planes of the reference face*"). The options available in the relevant drop-down boxes are different from the previous cases:
  - **A(xy)°45**: the left-hand drop-down box allows to select one of the 3 cartesian planes:
    - *A(xy)°*: it assigns the xy rotation plane
    - *A(xz)°*: it assigns the xz rotation plane
    - *A(yz)°*: it assigns the yz rotation plane

  The value sets the rotation angle in degrees on the plane, the pole (center) of the polar coordinate system is the P0 point: the axis which comes out of P0 on the plane together with the assigned angle defines the face x+ axis.

  On the three planes, the angle rotates in positive direction:
    - with the x+ axis which closes towards y+, in case of xy rotation plane
    - with the x+ axis which closes towards z+, in case of xz rotation plane
    - with the y+ axis which closes towards z+, in case of yz rotation plane

  The x semi-axis is fixed, but not the position of P1 on this last.
  - **X1l**: the left-hand drop-down box allows to choose among 3 different modes to complete the P1 assignment on the plane defined by a polar coordinate system:

    U: it assigns the module of the polar coordinate system (distance of P1 from P0, on the rotation plane). The assigned value is taken in absolute value

    X1: it assigns the x coordinate of P1, while the y coordinate is calculated provided that the P1 point belongs to x+ axis of the face

    Y1: it assigns the y coordinates of P1, while the y coordinate is calculated provided that the P1 point belongs to x+ axis of the face

  The coordinates selected in the drop-down box match the rotation plane chosen:
  - X1 and Y1, in case of xy rotation plane
  - X1 and Z1, in case of xz rotation plane

- Y1 and Z1, in case of yz rotation plane

For X1, the option selected in Figure is: X1=l.

Once the position of P1 on the selected plane of the polar coordinate system has been assigned, the position of P1 on the axis perpendicular to the plane is still to be defined.

- **Z10**: the left-hand drop-down box allows to choose among 3 different modes to complete the P1 assignment on the third axis (in the example: Z axis):
  - Z1: it assigns directly the position;
  - Z±: it assigns the variation of position with respect to the assigned value in P0 point;
  - AZ°: it assigns the angular variation with respect to the value assigned to P0 point. The set value must range between -90° and +90°: the value is considered valid if it is included in the above range, extremes excluded (less than epsilon°=0.001°). Positive values of the angle determine height increment, negative values determine height reduction.

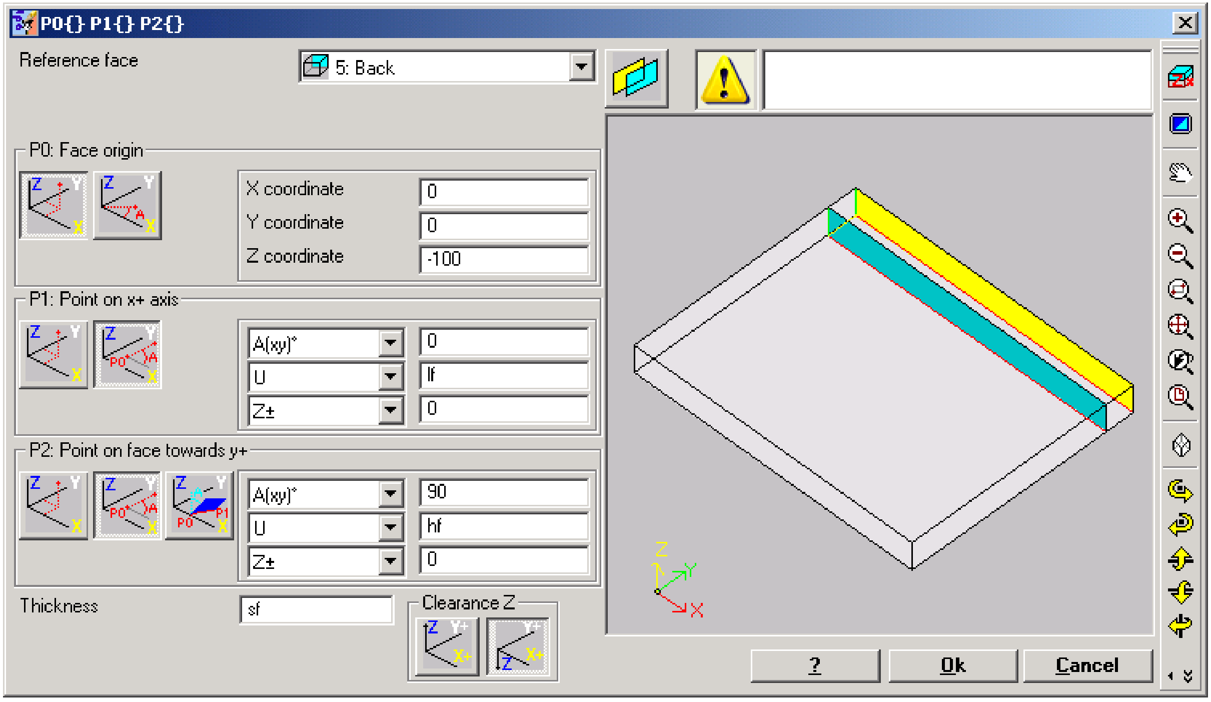
The here assigned coordinate matches the axis perpendicular to the rotation plane:

- z, in case of xy rotation plane
- y, in case of xz rotation plane
- x, in case of yz rotation plane

- **P2: Point on face towards y+**: the first bitmap (from left) is selected in the frame. The selection means that the three coordinates of the point are known. The coordinate fields position the P2 point in (0; h; 0). P2 forms an angle smaller than 90° with the x axis of the face (P0 to P1); therefore, the P2 projection on the y axis of the face (PY) is recalculated. The linear segments which join the face origin to P2 and P2 to PY in graphical representation demonstrate that really P2 and PY do not coincide.

Example 4

Assign a parallel face to a pre-existing one:



P0{} P1{} P2{}

Reference face: 5: Back

P0: Face origin

X coordinate	0
Y coordinate	0
Z coordinate	-100

P1: Point on x+ axis

A(xy)°	0
U	lf
Z±	0

P2: Point on face towards y+

A(xy)°	90
U	hf
Z±	0

Thickness: sf

Clearance Z

- **Reference face: 5: Back**: press then the following button . Set the Z coordinate of the P0 point to the –100 height (default value: 0) to translate the face along the Z- axis of the reference face so as to obtain the face shown in Figure. If it is necessary, move the P0 point on the x and/or y coordinates: assign values different from 0; if it is necessary assign different length and/or height dimensions: replace lf and/or hf values.

6.3.8 Constraints section

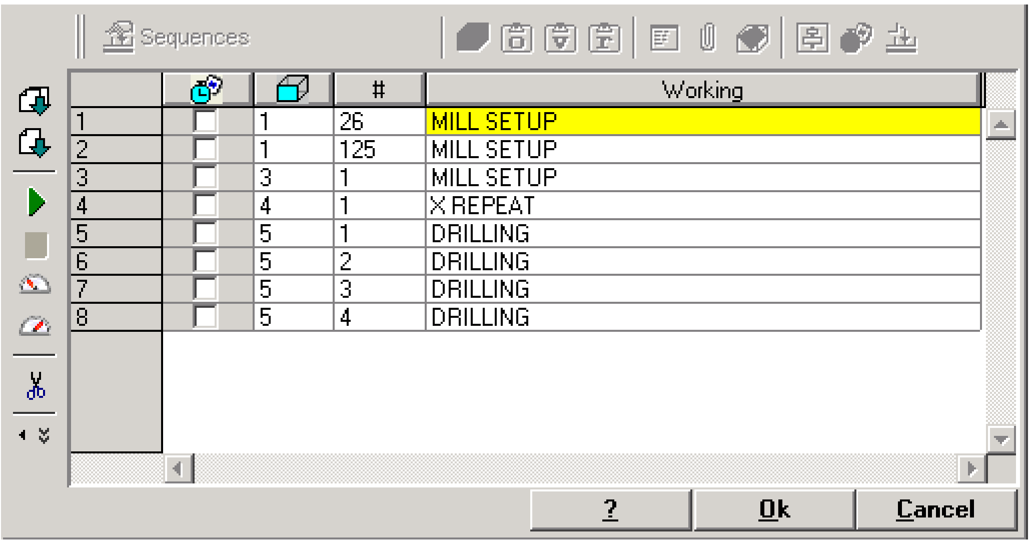
It is an optional page .
It is included in the custom information section. The configuration of assignments is supported by a kind of dedicated function and their meaning is unknown to the program..
What previously described about custom information in Special Settingss is always valid

6.3.9 Optimizations

It is an optional page .
It is included in the custom information section. The configuration of assignments is supported by a kind of dedicated function and their meaning is unknown to the program..
What previously described about custom information in Special Settings is always valid

6.3.10 Sequences

It is an optional page
It allows to establish a specific execution order of all workings assigned to the piece concerned.



			#	Working
1	□	1	26	MILL SETUP
2	□	1	125	MILL SETUP
3	□	3	1	MILL SETUP
4	□	4	1	X REPEAT
5	□	5	1	DRILLING
6	□	5	2	DRILLING
7	□	5	3	DRILLING
8	□	5	4	DRILLING

- **header**: it provides the consecutive sequence number
- **:** working optimization flag. The interpretation of the flag depends on each single application. The column is not displayed, if the management of the optimization flag is not configured.
- **:** number of the face on which the working is programmed
- : provides a selected box for workings, which assign a non-null "B" property field. The column is shown in the table, if almost one of the listed workings has an assign construct.
- #: sequential working number in the program face
- **Working**: working name
- **G..X..Y..Z:** it displays the ASCII working name (e.g. HOLE), the application point and the technology assignments (machine, group, head, spindle. diameter).

If almost one of the listed workings has an assigned description, a column to display the description is added.
The Buttons bar which is located on the left-hand side of the table displays the following commands:

It initializes the list of execution sequences according to the programming order of each face.

This command cancels every modification made manually from the opening of the session (Cut, Paste, Assignment of optimization flag).

It enters the list of face workings the selected program line belongs to. For example if you choose the line 4 which in figure corresponds to a working on face 4 and select this command all face 4 workings are entered starting from line 4 by following the execution order set inside the face itself.

Commands for the startup of a graphic simulation of the sequence order given in the list.

▶: it starts the graphic simulation, The current working is moved from the first row to the last one of the list, maintaining a constant period. The simulation can be interrupted, by clicking the button ❚❚ and restarted, by clicking the button ▶.

After selecting the button, the following buttons are displayed ▶**:**

■: it ends the simulation

: it decreases the display period of the simulation

: it increases the display period of the simulation

It cuts selected rows from the table and pastes them into Clipboard. This command is available only if Clipboard is empty. To select one or more lines, click on the corresponding line, keeping the **[Ctrl]**key pressed . Selecting the line, the head button is depressed with white writing on blue background. To deselect one or more lines, click on the line of the variable, while pressing the **[Ctrl]** key . Where the line is now deselected, the selection button is up and the colours are restored. To delete the selection from the whole list, click on any position of the table. In this way the line on which you have clicked has become the selected line.

It pastes the content of Clipboard at the current line and clears Clipboard. This command is available only if one or more rows have been copied into Clipboard

It enables to paste before the current row the content temporarily copied into Clipboard

It enables to paste after the current line the content temporarily copied into Clipboard

It activates the optimization flag of selected rows. The icon is displayed, if the management of the optimization flag is enabled.

It deactivates the optimization flag of selected lines. The icon is displayed, if the management of the optimization flag is enabled.

In the workings view, workings programmed in Piece-Face are not displayed.(see Chapter Piece-Face->Generation Of Execution Sequences)
According to the functions assigned in customizing Tpaedi32, the graphical representation may display all piece programmed workings or the only workings for which it is possible to assign the sequence. In this case the program does not show the workings which match the following items:

- open profiles
- subroutine or macro induced calls
- workings for which the sequence management is disabled

In the graphic representation the working matching the selected line in the table is highlighted. The representation may be:

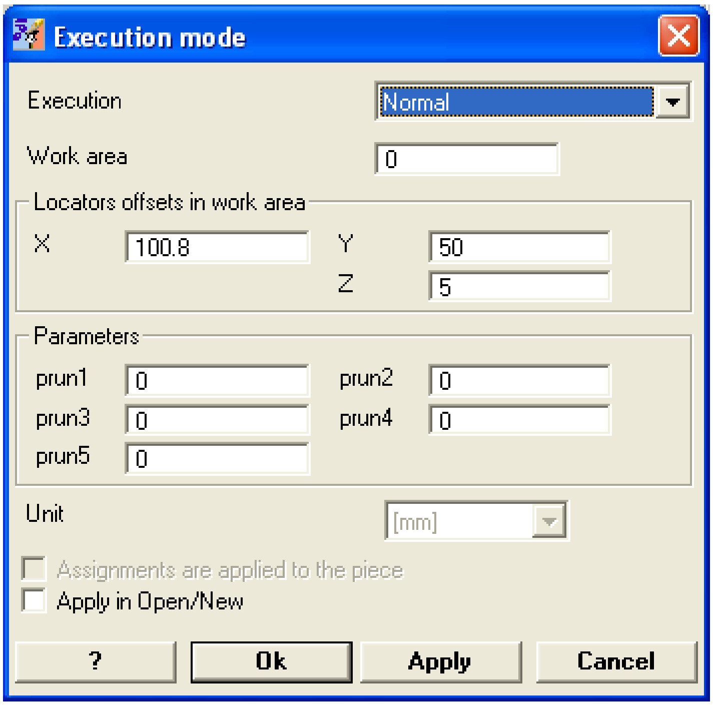
- three-dimensional of the piece. The area displays the piece in spatial view (xyz), with the relevant face workings applied;
- of the single face, in the face plane (with its xy axis orientation of the face).

If you quit’the page of the Sequences, a comprehensive update of the graphic representation of the program is made.

6.4 Advanced Assignments

6.4.1 Execution mode

This command is optional. This mode allows to set program execution data, with possibility of immediate application to the active program.The execution mode does not remain programmed and the machine is defined by the program managing the lists.
The setting window is recalled from ***Set->Execution mode*** menu



Execution mode

Execution: Normal

Work area: 0

Locators offsets in work area

X: 100.8 Y: 50
Z: 5

Parameters

prun1: 0 prun2: 0
prun3: 0 prun4: 0
prun5: 0

Unit: [mm]

□ Assignments are applied to the piece
□ Apply in Open/New

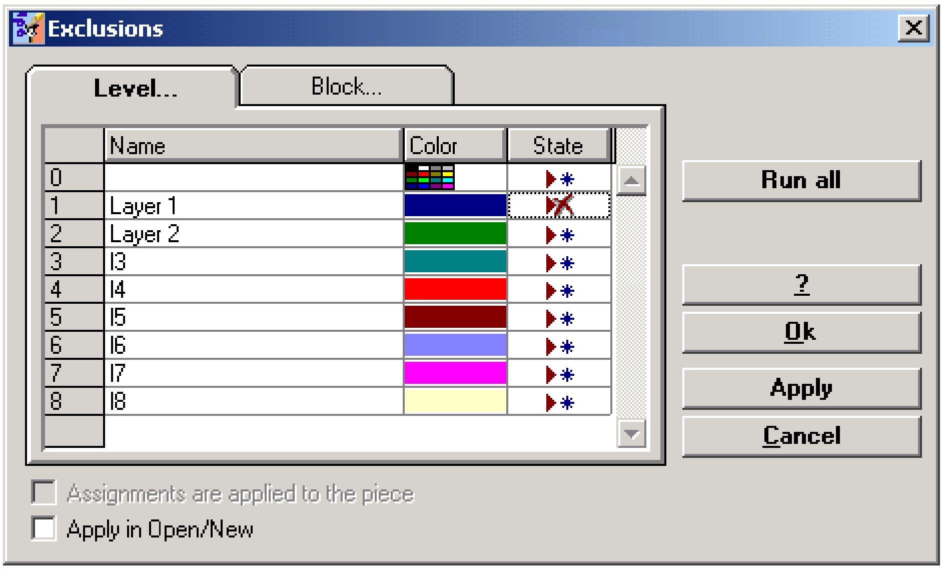
- **Execution**: program execution mode. Available items:
  - **Normal**
  - **Mirror X**
  - **Mirror Y**
  - **Mirror XY**

  Using the parametric programming, the parameters correspond to the prgn, prgx, prgy, prgxy functions of the Variable Arguments group.
- **Work area**: it assigns an identification number for each work area. It is a custom parameter, then it acquires a specific meaning for each application. Using the parametric programming, the parameter corresponds to the prarea function of the Variable Arguments group.
- **Unit**: it indicates the unit of measurement in which step offsets are expressed.
- **Locators offsets in work area**: they assign the step position of the selected work area with respect to the machine zero. Using the parametric programming, the parameters correspond to the prqx, prqy, prqz functions of the Variable Arguments group.
- **Parameters:** they assign the additional parameters in execution. They are numerical parameters of integer type. Using the parametric programming, the parameters correspond to the prun1,…,prun5 of the Variable Arguments group.
- **Assignments are applied to the piece** (grayed option): if selected, it indicates that Execution modes, as assigned in the window, are applied to the active program. This item is deselected at the first modification of assignments
- **Apply in Open/New**: select this option in case the execution Modes, which have been configured, are assigned to all programs to be created or recalled.

Click the **[Apply]** button to assign Execution modes to the active program in the same way as they are assigned.

6.4.2 Exclusions

This command is optional.
Tpaedi32 allows to exclude a set of workings from being executed. Such workings are identified by a common property, for example the L (Level) property and the K (Block) property.
An exclusion is equivalent to a logical added conditioning, with the essential difference that it does not remain programmed. The exclusion in a machine is defined by the program managing the lists. Referring to the figure, the program chart is tested with the level 1 workings excluded from the execution.
The setting window is recalled from ***Set->Exclusions*** menu and enables to choose between two similar configuration pages:



	Name	Color	State
0		▦	▸✱
1	Layer 1	🟦	▸✗
2	Layer 2	🟩	▸✱
3	l3	🟦	▸✱
4	l4	🟥	▸✱
5	l5	🟫	▸✱
6	l6	🟪	▸✱
7	l7	🟪	▸✱
8	l8	🟨	▸✱

☐ Assignments are applied to the piece
☐ Apply in Open/New

- **Level** for the assignment of exclusions to "L" property values
- **Block** for the assignment of exclusions to "K" property values

In a page of the Exclusions window a table shows the items adopted to manage levels or blocks:

- **Name**: name assigned to the level or block. (Modifiable from ***Set->Customize*** menu)
- **Color**: color assigned to the level or block. (Editable from ***Set->Customize*** menu)
- **State**: level or block status:
  - ▸✗ excluded (in the above example level 1)
  - ▸✱ not excluded (in the above example all levels except for level 1);

The first line matches level or block 0 (unassigned level or block). The multi-color bitmap means that the workings of this level use colors which differ each other according to the type of working.

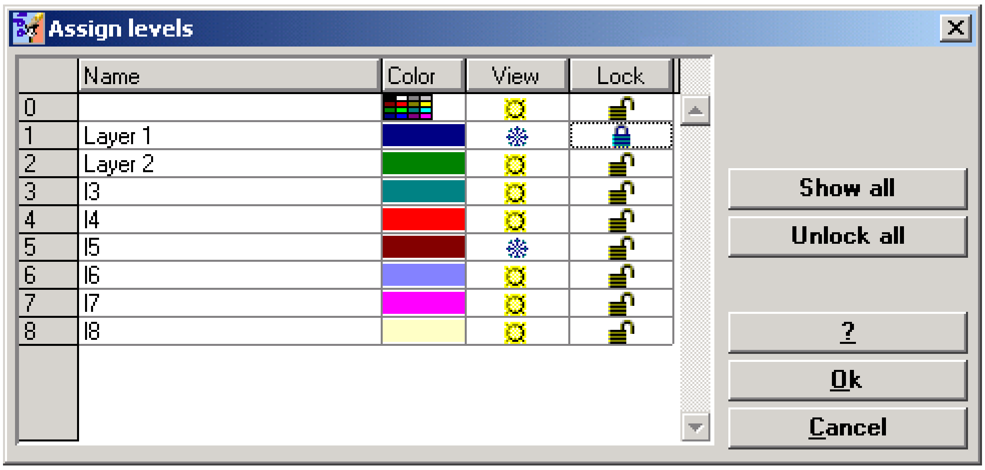
- **Assignments are applied to the piece** (grayed option): if selected, it indicates that Exclusions, as assigned in the window, are applied to the active program. This item is deselected at the first modification of assignments
- **Apply in Open/New**: if selected, the Exclusions, which have been configured, are assigned to all programs to be created or recalled

The **[Run all]** button resets all the exclusions set on the page. Click the **[Apply]** button to assign Exclusions to the active program in the same way as they are assigned.

6.4.3 Levels

All levels managed in configuring the program are listed in the above page (in the example: to level 8).
The setting window is displayed by selecting ***Set->Assign levels*** menu and includes the following

options:



	Name	Color	View	Lock
0		▦	☀	🔓
1	Layer 1	🟦	❄	🔒
2	Layer 2	🟩	☀	🔓
3	l3	🟦	☀	🔓
4	l4	🟥	☀	🔓
5	l5	🟥	❄	🔓
6	l6	🟪	☀	🔓
7	l7	🟪	☀	🔓
8	l8	🟨	☀	🔓

- **Name**: name assigned to the level (Modifiable from ***Set->Customize*** menu)
- **Color**: color assigned to the level (Modifiable from ***Set->Customize*** menu)
- **View**: level display status:
  - displayed (it shows a sun: in the example levels 0, 2, 3, 4, 6, 7, 8)
  - not displayed (it shows a snowflake: in the example levels 1, 5);
- **Lock**: free or locked level status:
  - free (it shows an open padlock: in the example all levels except for level 1)
  - locked (it shows a closed padlock: in the example level 1).

The first line matches level 0 (unassigned level): means that the workings of this level use colors which differ each other according to the type of working.

The **[Show all]** button activates the Displayed status for all access levels.
The **[Unlock all]** button activates the Free status for all access levels.

6.4.4 Special Filters

This command is optional. In this section, the O Field, Construct and Technology values are provided. The setting window appears by selecting ***Set->Assign special filters*** menu and includes the following options:

O field

- **Name**: name assigned to the O field. (Modifiable from ***Set->Customize***)
- **Reference Origin**: reference origin assigned to the O Field. (Modifiable from ***Set->Customize***)

Alternatively, it may display the following columns:

- **Color**: color assigned to the O field. (Editable from ***Set->Customize*** menu)
- **View**: O field display status:
  - displayed (it shows a sun)
  - not displayed (it shows a snowflake);
- **Lock**: free or locked 0 field status:
  - free (it shows an open padlock)
  - locked (it shows a closed padlock)

The **[Show all]** and **[Unlock all]** buttons deactivate special filters set for the O Field
The page Field O is not available, if the management of the "O" Field is not enabled or if the assignment of the "O" Field on the single segments of profile is not enabled.

Construct

Construct values configured in Tpaedi32 are provided

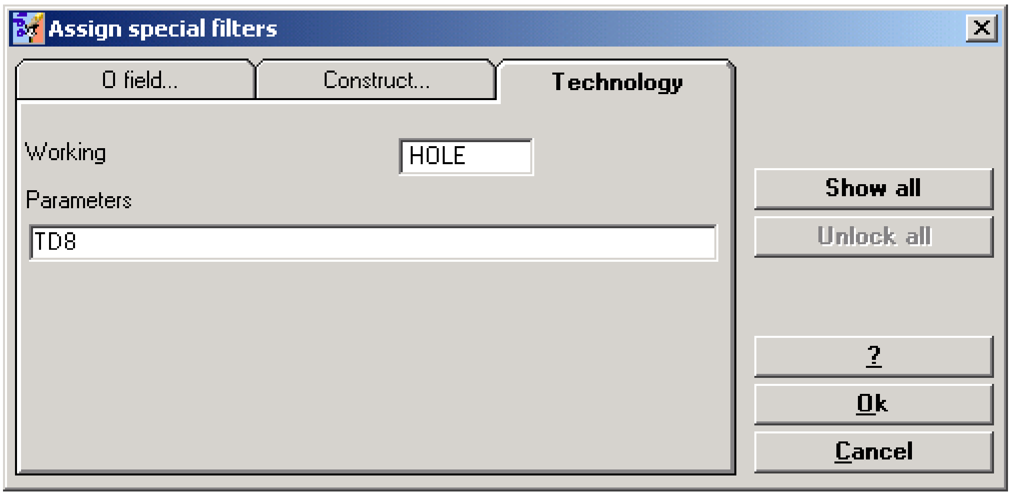
- **Name**: name assigned to the construct. (Modifiable from ***Set->Customize*** menu

- **Color**: color assigned to the construct. (Editable from ***Set->Customize*** menu)
- **View:** construct display status**:**
  - displayed (it shows a sun)
  - not displayed (it shows a snowflake);
- **Lock**: free or locked construct status:
  - free (it shows an open padlock)
  - locked (it shows a closed padlock)

The **[Show all]** and **[Unlock all]** buttons deactivate special filters set for the Construct.
The page Construct is not available, if the management of the Constructs is not enabled.

Technology

This section allows to choose workings to display by assigning the relevant working ASCII code and/or a set of parameters belonging to the working itself. No lock/unlock option is available since it is only about a display filter.



Working HOLE

Parameters

TD8

According to the data set in the example window the only workings with "*HOLE*" ASCII code and *TD* parameter value set to 8 are displayed.
Parameters are interpreted as technological settings strictly connected to workings (examples: machine, group, tool), therefore, in case of working related to a profile, only the parameters which belong to the profile opening working (setup or segment of profile) are taken into account.
The option **Total match** determines the search criteria of the workings, that verify the settings. If selected, the expanded lists are verified as well, i.e. the workings assigned by subprograms or macro-programs. If not selected, only the programmed workings are verified (list in the ASCII-text).
In the above figure, if the option is not selected, the workings, that are directly programmed, (HOLE, TD8) are verified. If the option is selected, the other workings (HOLE,TD8), deriving from programming a sub-program, are verified, as well.

It is not necessary to assign both fields. Thus, if you always refer to the example in Figure:

- if no value is assigned to the Parameters field: the only workings with "*HOLE*" ASCII code are displayed;
- if no value is assigned to the Working field: the only workings with *TD* parameter set to 8 are shown.

For the Parameters field it is possible to provide parameter settings:

- In case of ="TD=r27" parameters, the only workings with *TD* parameter set to "r27" are displayed
- In case of ="TM2 TD=r27" parameters the only workings with TM parameter set to 2 and *TD* parameter set to "r27" are shown

We can see that:

- in case of numerical setting the comparison is made with the parameter numeric value
- in case of parametric setting the comparison is made with the parameter setting

It is also possible to assign logical conditions. Examples:

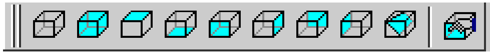
- In case of ="TMR<=3" parameters: the only workings with *TMR* parameter value lower or equal to 3 are displayed;
- In case of ="TMR#3", "TMR<>3" parameters: the only workings with *TMR* parameter different from 3 are shown;
- In case of ="TMR>3": the only workings with *TMR* parameter value greater than 3 are displayed
- In case of ="TMR>3 GR=r4" parameters: the only workings with *TMR* parameter value greater than 3 and *GR* parameter set to "r4" are shown.

In case of assignment of logical conditions it is recommended to enter numerical settings (like the value assigned to TMR in the above examples).
The **[Show all]** button deactivates special filters set for Technology.

7 Face

7.1 How To Open It

It is possible to select a face from the Face Selection Bar:



The bar includes real and fictive faces:

- real faces are the only faces actually enabled during configuration
- fictive faces, instead, are assigned in Overall View, except the faces set as construction auxiliary faces.

Each bitmap of the bar corresponds to a face or to a group of faces:

Overall View

piece-face

... from face 1 to face 6

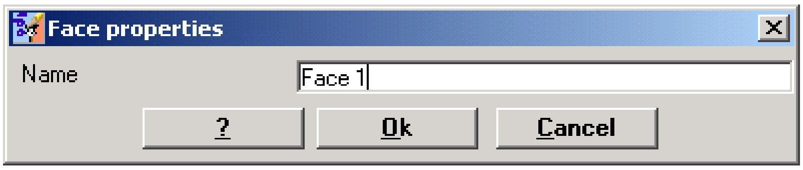
it opens the list of the fictive faces

Each selection shows a descriptive message:

- face number (except for Overall View and Piece Face)
- face name
- program lines assigned to the face

The numbering of real faces may change on different applications: in fact, it is possible to assign customized numbering to the six real faces of the basic parallelepiped.

By selecting the relevant icon the **Face properties** window appears enabling the user to assign the face name. The names of the faces are not saved in a file language, so they cannot be translated.



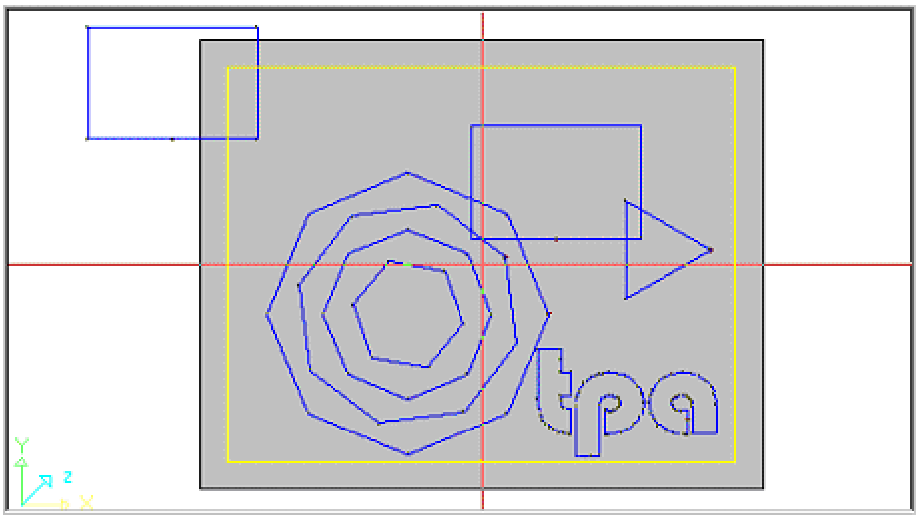
Face properties

Name: Face 1

7.2 Graphical Representation Of Face View

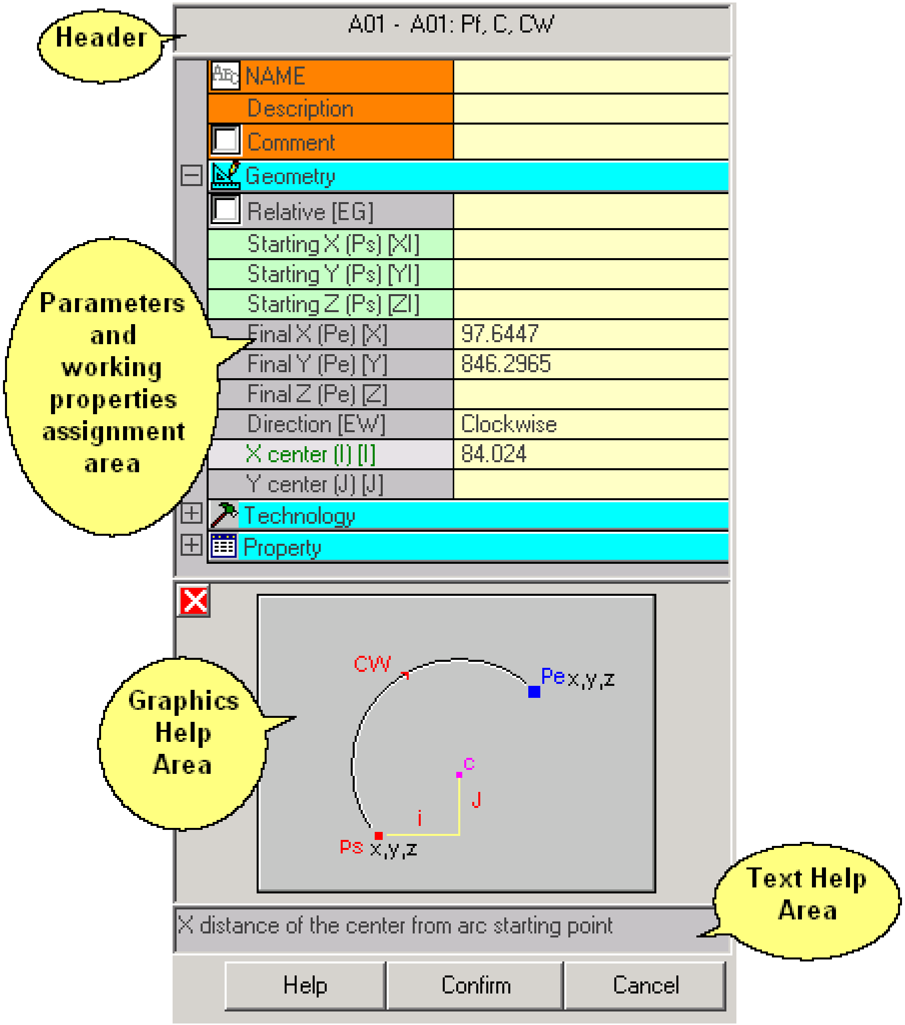
Piece Current View

It provides the planar graphical representation of the face.



The Figure is an example of face graphics. The cross-cursor as well is fully displayed on the view. The reference system origin and the face axes are defined.

Working Assignment Area
If workings have been defined in the face program, the active working data is provided in the **Working Assignment** window.



A01 - A01: Pf, C, CW

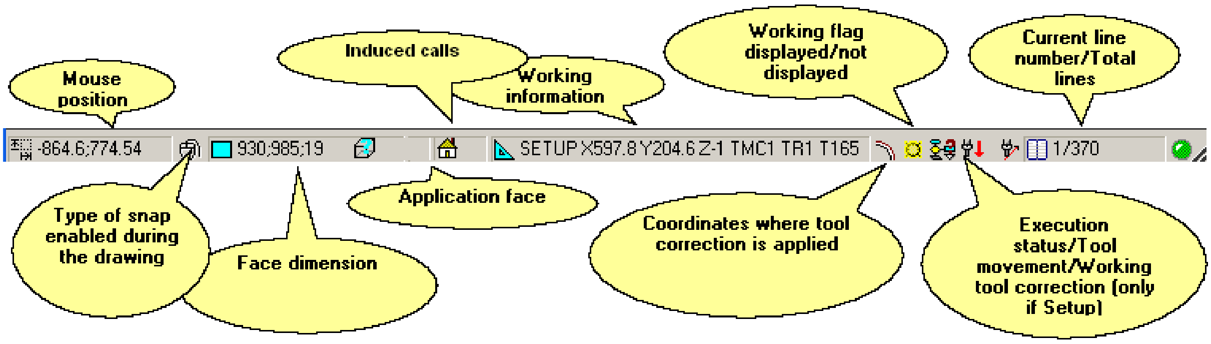
NAME	
Description	
□ Comment	
⊟ Geometry	
□ Relative [EG]	
Starting X (Ps) [XI]	
Starting Y (Ps) [YI]	
Starting Z (Ps) [ZI]	
Final X (Pe) [X]	97.6447
Final Y (Pe) [Y]	846.2965
Final Z (Pe) [Z]	
Direction [EW]	Clockwise
X center (I) [I]	84.024
Y center (J) [J]	
⊞ Technology	
⊞ Property	

X distance of the center from arc starting point

- **Header**: it appears in the form: Working ASCII name : Working description. In Figure for example A01:xy(Pf,C;CW)
- **Parameters and working properties assignment area**: items are arranged in a list as directly visible items (Name, Description, Comment,..) or grouped into nodes (Geometry, Technology, Property). Near the description of each parameter the relevant ASCII names may be displayed in square brackets. In Figure [XI] is the ASCII name of the starting X parameter (Ps); [EW] is the ASCII name of the Direction parameter.
- **Graphics Help Area**: graphical help for setting the geometric data of the selected working
- **Text Help Area**: description of working parameters

The above window can appear in two different ways. In the first display mode, the operator can modify directly working settings and confirm by clicking the **[Confirm]** button. In the second display mode, working settings can only be read. To edit these last, the operator shall select the **[Edit]** button.

Status Bar


Mouse position
Induced calls
Working information
Working flag displayed/not displayed
Current line number/Total lines
-864.6;774.54
930;985;19
SETUP X597.8 Y204.6 Z-1 TMC1 TR1 T165
1/370
Type of snap enabled during the drawing
Face dimension
Application face
Coordinates where tool correction is applied
Execution status/Tool movement/Working tool correction (only if Setup)

Clicking in the area of face Dimension, the program activates the window assigning the piece dimensions in overall view. Description of the bitmaps which are used to display the **Execution Status** of the active working with logical conditions applied:

	logical conditions not applied
↓	the working verifies logical conditions
⛔	the working does not verify logical conditions

Description of the bitmaps which are used to display the **Tool Movement** assigned to the active working:

↕	point working or isolated setup. It indicates the lowering movement and subsequent rise of the tool
↓	setup which opens a profile. It indicates the lowering movement of the tool
↓↓	setup of a multiple profile (it indicates the lowering movement of an additional tool).
↳	work on the profile or complex code which open the profile
↑	work on the profile which opens a profile and rises at the end of the segment
→	working which continues to execute a profile from the previous working to the next working
↑	working which continues a profile and rises at the end of the execution
↑	profile closing setup (hook setup)

Meaning of the bitmaps used for displaying the **selection of tool correction**, only if it is a setup working:

	correction not activated
	correction left activated
	correction right activated

Commands Visualization Area and Errors Visualization Area

Commands area displays the command exit status or provides instructions for the execution of a command.

The errors area shows the complete list of errors and warnings which have been detected in processing the face.

Place the cursor on a line of the above list, **left-click** the mouse to select the line concerned and press the **[SHIFT]** or **[CTRL]** key: the contextual help relative to the selected error appears.

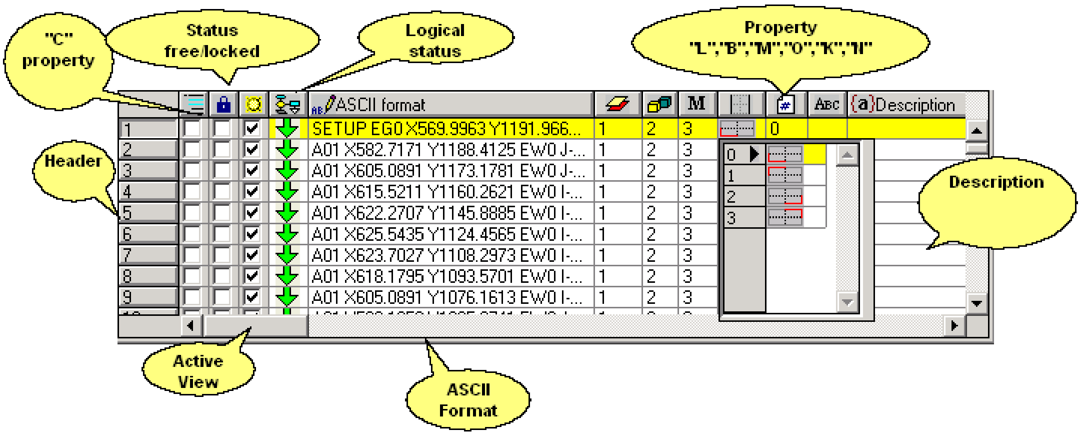
7.3 Assignments

7.3.1 Assignment Area

The assignment area provides the face program in ASCII format.
A few remarkable situations are identified and highlighted by using specific colors. The displayed list shows such situations in priority order:

- the current working appears selected (*Header* numbered button depressed) and highlighted in different color
- selected working
- comment working ("C" property flag active)
- invalid working
- induced working

The page includes a table of as many rows as the number of face workings. Different columns are assigned to each row:



					ASCII format			M			ABC	Description
1	□	□	☑	↓	SETUP EG0 X569.9963 Y1191.966...	1	2	3		0		
2	□	□	☑	↓	A01 X582.7171 Y1188.4125 EW0 J-...	1	2	3				
3	□	□	☑	↓	A01 X605.0891 Y1173.1781 EW0 J-...	1	2	3				
4	□	□	☑	↓	A01 X615.5211 Y1160.2621 EW0 I-...	1	2	3				
5	□	□	☑	↓	A01 X622.2707 Y1145.8885 EW0 I-...	1	2	3				
6	□	□	☑	↓	A01 X625.5435 Y1124.4565 EW0 I-...	1	2	3				
7	□	□	☑	↓	A01 X623.7027 Y1108.2973 EW0 I-...	1	2	3				
8	□	□	☑	↓	A01 X618.1795 Y1093.5701 EW0 I-...	1	2	3				
9	□	□	☑	↓	A01 X605.0891 Y1076.1613 EW0 I-...	1	2	3				

- **Header**: sequential numbering of workings (starting from 1). The active working is highlighted in different color. The selected workings have the relevant Header numbered buttons depressed.

- **"C" property**: the column does not appear if the property is not managed. If the field is selected the working remains in the list but it does not affect the program. At this purpose when it is referred to the previous or next working, with respect to another, it has to be meant commented workings excluded. This property can also be set from ***Apply->Assign property->Comment*** menu.

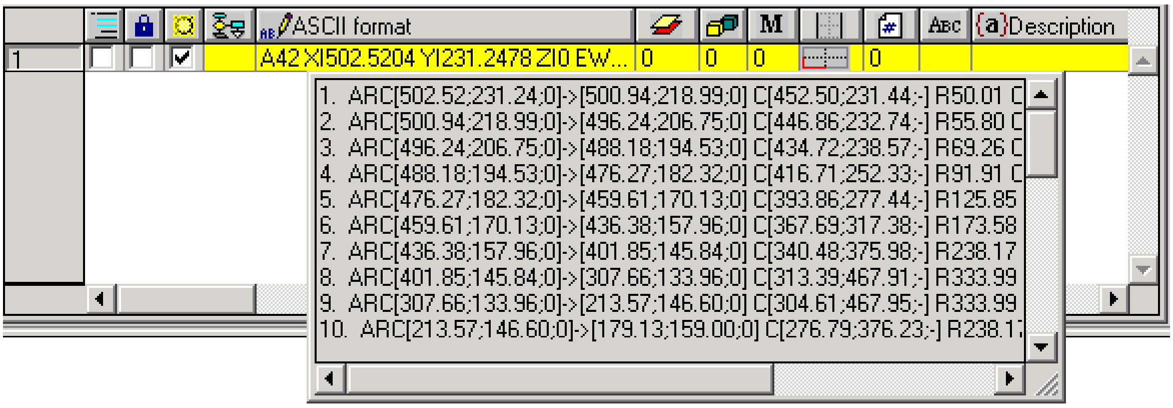
- **active view:** the field is active (visible check) if working is graphically shown. It is an optional column. A working is not graphically shown if:
- comment flag is active ("C" property)
- has logical type

- a display filter is active concerning properties (fields "L", "B", "O") or technology (operating code and/or technological parameters)
- a special view is active (sections and/or logical conditions).

- **Status free/locked** for the selected working: if the field is active it means that the working is a call induced from the application of a macro or subroutine into another face of the piece, or it has the level ("L") or construct ("B") or O property ("O") field locked. The level (or construct or O field) lock results from accidental modifications of blocks assigned to the same level (or construct or O field). When, for example, a level is locked, in any case, it can remain visible and it is possible to assign new workings on it. Only if the field is unlocked, it can be modified as usual. The column instead cannot be edited.

- **Logical status**: the relevant field shows a green arrow, if the working verifies logical conditions; otherwise it displays a crossed yellow arrow. It is an optional column.

- **ASCII Format** includes: the assignment of the operating code (it interprets the first field of the ASCII text. Examples: "G89", "L01", "A01") and of parameters, in ASCII format as defined for the relevant working. If the column was configured as editable it is also possible to modify the ASCII code of the working concerned (G89, A01, L01,…). In this case to make a modification means to replace the working with another. The column may display an indent for the immediate visualization of the logical structure of the program, which is evaluated on the basis of the if (IF, ELSE, ENDIF) and for (FOR, ENDFOR) cycles. In the case of expanded active working (multiple segment of profile, subroutine or macro recall) it is possible to open a second window, matching the expanded list, by right-clicking the mouse. Each line of the expanded list corresponds to a working, of which geometric, technological data and assigned properties are provided in the same way as bits of information are shown in the status bar for the active working

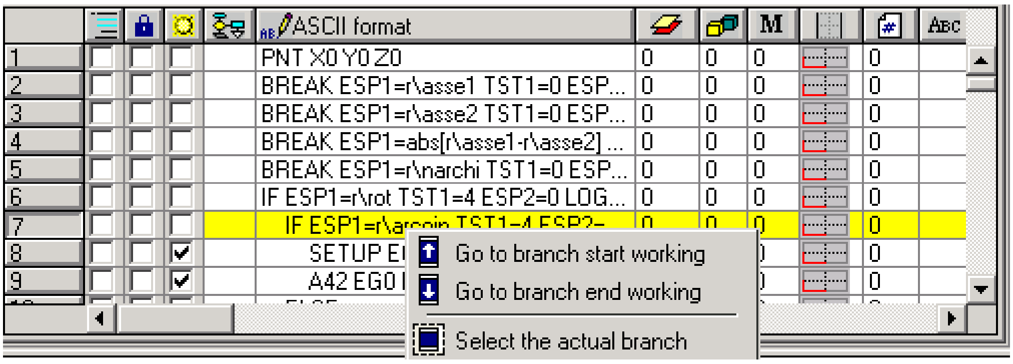


					ASCII format			M			ABC	Description
1	□	□	☑		A42 XI502.5204 YI231.2478 ZI0 EW...	0	0	0		0		

```
1.  ARC[502.52;231.24;0]->[500.94;218.99;0] C[452.50;231.44;-] R50.01 C
2.  ARC[500.94;218.99;0]->[496.24;206.75;0] C[446.86;232.74;-] R55.80 C
3.  ARC[496.24;206.75;0]->[488.18;194.53;0] C[434.72;238.57;-] R69.26 C
4.  ARC[488.18;194.53;0]->[476.27;182.32;0] C[416.71;252.33;-] R91.91 C
5.  ARC[476.27;182.32;0]->[459.61;170.13;0] C[393.86;277.44;-] R125.85
6.  ARC[459.61;170.13;0]->[436.38;157.96;0] C[367.69;317.38;-] R173.58
7.  ARC[436.38;157.96;0]->[401.85;145.84;0] C[340.48;375.98;-] R238.17
8.  ARC[401.85;145.84;0]->[307.66;133.96;0] C[313.39;467.91;-] R333.99
9.  ARC[307.66;133.96;0]->[213.57;146.60;0] C[304.61;467.95;-] R333.99
10.  ARC[213.57;146.60;0]->[179.13;159.00;0] C[276.79;376.23;-] R238.1
```

When working selected is logical, press mouse right button to open a shortcut menu to easily move inside program tabulations. The following items are listed:

- **Go to branch start working:** current working is moved to upstream program line which starts the logical condition (in figure: IF in line 6)
- **Go to branch end working:** current working is moved to downstream program line which closes the logical condition (in figure: ELSE in line 10)
- **Select the actual branch:** selection of working blocks belonging to current working level



					ASCII format			M			ABC
1	□	□	□		PNT X0 Y0 Z0	0	0	0		0	
2	□	□	□		BREAK ESP1=r\asse1 TST1=0 ESP...	0	0	0		0	
3	□	□	□		BREAK ESP1=r\asse2 TST1=0 ESP...	0	0	0		0	
4	□	□	□		BREAK ESP1=abs[r\asse1-r\asse2] ...	0	0	0		0	
5	□	□	□		BREAK ESP1=r\narchi TST1=0 ESP...	0	0	0		0	
6	□	□	□		IF ESP1=r\rot TST1=4 ESP2=0 LOG...	0	0	0		0	
7	□	□	□		IF ESP1=r\arcoin TST1=4 ESP2=	0	0	0		0	
8	□	□	☑		SETUP E			0		0	
9	□	□	☑		A42 EG0			0		0	

"L" property (level): the cell, a selection from the list and a numerical or parametric programming can be directly edited. Selection from the list allows one numerical assignment only. If a value greater than 0 is set the working is assigned on a level. A different color is assigned to display each level. The construct color prevails over the level color and this last over that of the "O" field. Therefore, if both construct and level properties are assigned to the selected working, this last is displayed with the construct color. It is an optional column. This property can also be set from ***Apply->Assign property->Level*** menu.

"B" property (construct): it is an optional column. The cell, a selection from the list and a numerical or parametric programming can be directly edited. Selection from the list allows one numerical assignment only. This property can also be set from ***Apply->Assign property->Construct*** menu. If the construct property is assigned to the working, this last is compiled but not executed.

M.. **"M" property**: it is an optional column. The cell, a selection from the list and a numerical or parametric programming can be directly edited. This property can also be set from ***Apply->Assign property->M field*** menu.

"O" property: the cell, a selection from the list and a numerical or parametric programming can be directly edited. Selection from the list allows one numerical assignment only. If a value greater than 0 is set the working can be highlighted with one of the assigned visualization colors. Tpaedi32 can interpret a use of the property to assign a programming reference (side or edge) to the working only if the maximum property value managed is lower than 3. It is an optional column. This property can also be set from ***Apply->Assign property->O field*** menu.

"K" property (block): it is an optional column. The cell, a selection from the list and a numerical or parametric programming can be directly edited. Selection from the list allows one numerical assignment only. This property can also be set from ***Apply->Assign property->Block*** menu.

"N" property (Name): it is an optional column. It is the working name. This property can also be set from ***Apply->Assign property->Name*** menu

Description: working comment.

There are some exceptions for which a working cannot be modified:

- locked status flag for the working (induced call or locked level, construct or "O" field)
- the working operating code is invalid (it is not entered into the workings database)
- the working has the "C" property flag active: it is necessary to deactivate the "C" field, before editing any other working parameters. However, these last can never be modified in case of induced call or locked level (construct or "0" field). If the working operating code is invalid the flag can be activated in any case.

"O", "M", "B", "L" and "K" property fields may not be modifiable (L, B, K always and possibly M and O, if they are not configured to be directly edited in profile) in the following cases:

- works on the profile (arcs and lines): if the working opens a profile (open profile) the value is still 0, otherwise they take the value from the profile start working

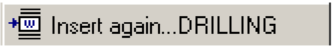
- in case of setup or complex working, with request for point hook: the properties which cannot be edited in profile are propagated by the profile start working

8 Workings

8.1 Types Of Workings

8.1.1 Simple and Complex Workings

A working insertion is performed by selecting the working itself through the Workings bar. The insertion of the last working can be repeated if selected from a shortcut menu, which is opened by pressing the right button of the mouse within the face visualization area. As an example, the menu



Simple Workings include: individual drillings, insertion of individual workings, individual setups, line and/or arc segments, logical instructions.

Point and setup workings have direct assignment for *technology and geometry*.
Main usage of setup working is profile opening: setup provides for technological information to be used for profiling. A setup can also be used alone, not followed by a profile.
On the contrary, a point working is used alone. Examples of point workings are drills and insertions.

Logical workings are featured to meet specific customization needs
For example, logical workings can be implemented for:

- real-time measure of piece;
- scheduled stop during piece execution;
- limits setting.

Geometric and technological fields can be assigned to logical workings. Anyway, they are not interpreted by *Tpaedi32:* for example, they are neither displayed in a graphic area, nor calculated in overall dimensions; moreover, any relevant positioning sets by the following working is not applied.

Complex Workings are defined by combining simple and/or complex workings. They include for example: drilling cycles (fitting, repeat), polygons, oval, ellipse, sawings.

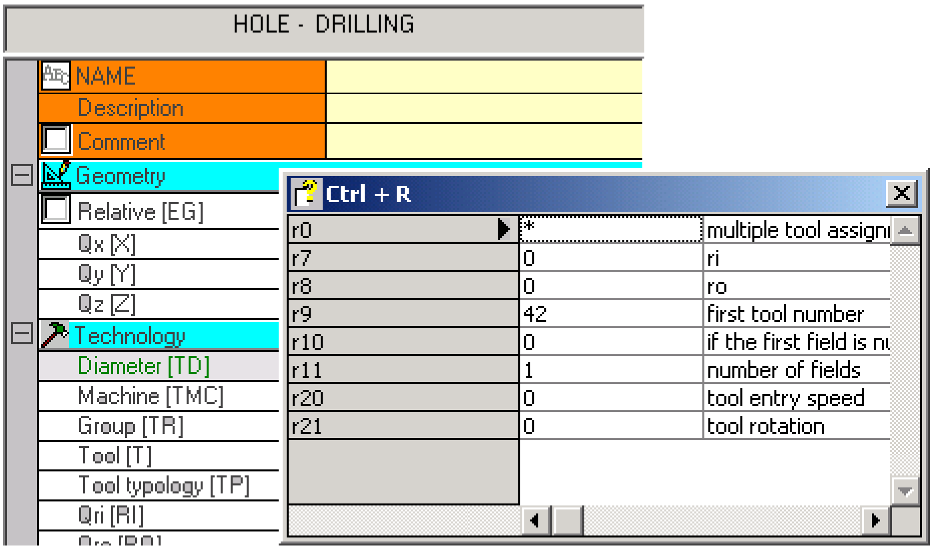
In general, the setting of each parameter can be parametric.
A **working parameter** can use each parameter setting managed in parametric programming. In particular:

- piece and/or face dimensions (l, h, s)
- program variables (o, v, r)

Working properties accept only numerical settings.

Once data has been confirmed, the working is actually entered into the face program only if no error conditions are detected in programming the working itself. In this case it is necessary to solve problems or cancel the entry operation. Only in the case of assignment of the working in macro-program it is possible to confirm the entry also in case of error. When the working has been entered, the face program is updated as a result of the insertion of the new working: the face view is updated, the entered working becomes the active working.

Guided Edit
It is possible to ask for help on the "r" variables assigned to the program, by pressing the **[CTRL+R]** key combination directly from the workings edit window:



HOLE - DRILLING

NAME	
Description	
□ Comment	
Geometry	
□ Relative [EG]	
Qx [X]	
Qy [Y]	
Qz [Z]	
Technology	
Diameter [TD]	
Machine [TMC]	
Group [TR]	
Tool [T]	
Tool typology [TP]	
Qri [RI]	
Qro [RO]	

Ctrl + R

r0	*	multiple tool assigni
r7	0	ri
r8	0	ro
r9	42	first tool number
r10	0	if the first field is nu
r11	1	number of fields
r20	0	tool entry speed
r21	0	tool rotation

select the item you wish to edit and confirm by pressing the **[ENTER]** key or **double-clicking** the item. The name which corresponds to the selected item will be automatically entered in the edit field, at the cursor position. If a symbolic name has been assigned to the "r" variable, the same name is inserted into the edit field.

This kind of help is available for the assignment of:

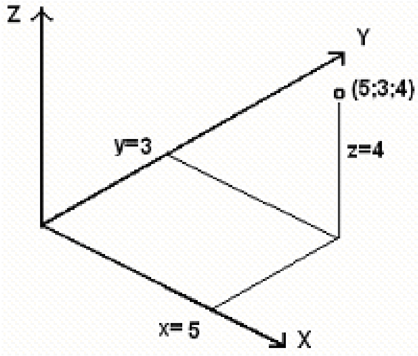
- variable geometries (fictive faces)
- workings in face program.

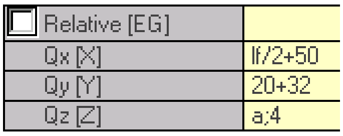
8.1.2 Point of application

A working point of application is defined by x and y coordinates assigned on XY plan and by Z coordinate, perpendicular to face plan.

Coordinates can be assigned to a Cartesian coordinates systems or polar coordinates system.

Assignment of Cartesian coordinates:





□ Relative [EG]	
Qx [X]	lf/2+50
Qy [Y]	20+32
Qz [Z]	a;4

In a Cartesian system, directly-assigned coordinates are the following:

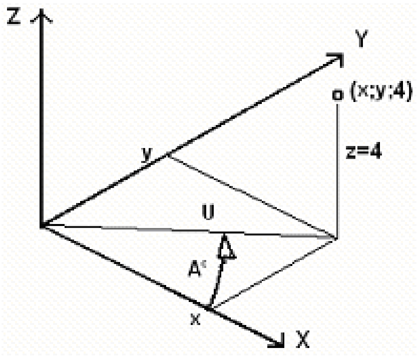
- absolute from face origin, if **Relative** box is not selected
- relative from last programmed position upward, if **Relative** box is not selected

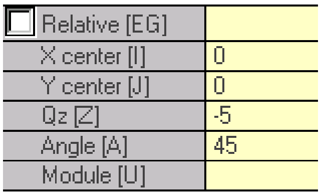
If point is assigned like in figure, with coordinates (x=5;y=3;z=4), but in relative mode with the last position programmed at (x=2; y=2;z=2), working shall have its own point of application at (x=7; y=5;z=6).

Absolute/relative selection applies to all coordinates.

When **Relative** mode is selected, absolute mode can be forced on a single coordinates, by placing "a;" before coordinate setting.

Assignment of polar coordinates:





☐ Relative [EG]	
X center [I]	0
Y center [J]	0
Qz [Z]	-5
Angle [A]	45
Module [U]	

This figure shows polar coordinates. Z coordinate remains directly assigned, as in the case where Cartesian coordinates are assigned.

Position of point in XY plan is specified by giving its distance from a centre and its angle (in degrees) in XY plan from X axis.

In figure:

- the centre is the face origin (0;0);
- distance from centre is 100;
- angle is 45°.

Now, *absolute/relative* selection applies to Z coordinate and (x;y) coordinates of centre.

When *relative* mode is selected, absolute mode can be forced on a single coordinates, by placing "a;" before coordinate setting.

8.1.3 Technology

A point or setup working has a technology assigned, which depends on how working is made, i.e. on which machine and/or group and/or tool.

Technology assignment implies plant architecture to be assessed.

The technology is defined by the plant. A plant is made of one ore more machines, in everyone of which one or more groups can work, divided into devices. When the technology is applied to a punctual or to a setup working, reference is made to a specific tool, that is fitted out in a specific position (spindle/ electrospindle) of a machine's head group.

To each head group is assigned a max. spindle configuration (depending on the application). In general, everyone can fit out a toolholder or a tool. Each machine can be provided with a tool catalog, a toolholder catalog, everyone of which can be fitted out with up to a max. number of tool (depending on the application).

General assessment criteria

Assessment criteria for tool programming applied by *Tpaedi32* are examined below, according to a chart of possible cases, on the basis of priorities applied to their assessments.

Programmation by spindle (or electrospindle) **and tool**

In the figure the fields **Spindle** are set up with value 100 and **Tool** with value 2.

For **Machine** and/or **Group** it is generally necessary to set up a value. If the field is not assigned, the value 1 is set up by default.

The value assigned to the **Spindle** field sets a spindle position on the group, while the value assigned to the **Tool** field defines the tool (or the tool-holder) to fit out on **Spindle.**

According to the technology of the machine, the value assigned to the field **Tool** can also define a tool-holder and, in the case of tool-holder fitted out with more tools, it can also show the position in use.

In the example displayed in the figure:

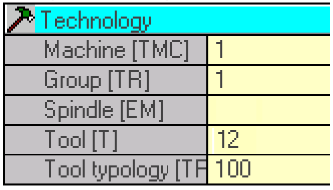
- if the **Spindle** 100 of the group 1 is associated with a tool change, the Spindle with tool number 2 is used;
- otherwise: the **Spindle** 100 must necessarily fitted out with the tool number 2.

If only one of both fields (**Tool** or **Spindle**) is set up, reference is made at the spindle programming spindle, as farther on described.

The **Tool Typology,** allowing a specific wider choice on the tool selection, can also be assigned.
As displayed in the figure, working is considered as correct and completed, if only the **Tool**=2 of **Machine**=1 and **Group**=1 will be configured with **Tool Typology**=100.

The field **Tool** can be assigned by default and be not visible in the window of the working data setup. This is the case of a group with only one configured electrospindle.

Programming by spindle (or electrospindle)



Technology	
Machine [TMC]	1
Group [TR]	1
Spindle [EM]	
Tool [T]	12
Tool typology [TF	100

Following cases can happen:

- The only field **Tool** is available and set up (in the figure with value 12);
- the only field **Spindle** is available and set up;
- both the fields are available, but only one is set up (in the figure the tool with value 12)

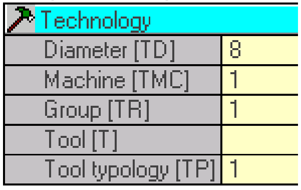
For **Machine and /or Group** in general a value must be necessarily set up. If the field is not assigned, the value 1 is imposed.

The tool is directly chosen in the file **Tool** (or electrospindle), using the current tooling.
If the spindle is not fitted out according to the technologic parameters, following cases can happen:

- choice of the *default tool*
- error situation

Like in the previous situation, **Tool Typology** can also normally be assigned.

Programming by diameter



Technology	
Diameter [TD]	8
Machine [TMC]	1
Group [TR]	1
Tool [T]	
Tool typology [TP]	1

The only Diameter field is set up: here with value 8. The values Spindle (or Electrospindle) and Tool cannot be set up.
For **Machine**, **Group** and **Tool Typology** a value can be generally set up. If a value is not assigned, a default one is not set up.
Selection criteria for program execution tool remain specific of a single application.

Programming by diameter is typical of drilling workings and, according to the declared available tools, can determine the execution of more drilling in one step.

Default tool
Neither **Spindle (**or Electrospindle**)** field, nor **Tool** spindle, **Diameter** field is set up.
The selection of **Machine** and/or **Group** and/or **Tool Typology** can be forced.
Selection criteria for program execution tool are specific of a single application.
Default Tool programming can not always be really operating.
In this case an error during the program optimization is reported.

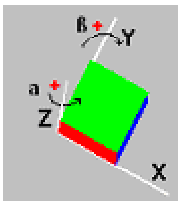
Automatic tool
Selection of automatic tool takes priority on the settings assigned to fields **Spindle** (or Electrospindle),

Tool and **Diameter.**
The choice of **Machine** and/or **Group** and/or **Tool Typology** can be forced.
Selection criteria for program execution tool are specific of a single application.

Oriented tool
When a normal working condition applies, the tool is set perpendicularly to xy plan of working face. A setup working can also assign tool orientation from face plane, to be defined as oriented setup. The following fields define tool orientation:

- rotation angle (alpha),
- slewing angle (beta).

Both rotation axes have **absolute** programming **on piece**.
Tool rotation fields, if assigned for working, are significant in any case, even if they are not setup (in this case their value is 0).



This figure represent a generic piece and three absolute Cartesian points:

- beta rotates around the y axis
- alpha rotates around the z axis.

8.1.4 The graphic display

The punctual and set-up workings are graphically represented as a two-dimensional face with a circle with the same diameter as the one of the programmed tool; in the three-dimensional graphics, workings are represented by a cylinder with the same diameter as the one of the programmed tool and the same height as the one of the tool's depth on the face.

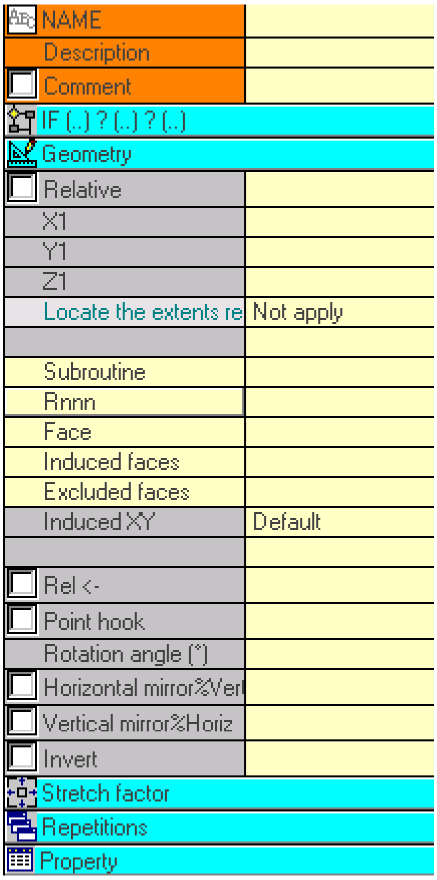
In three-dimensional graphics, in a set-up with an orientated tool, the tool representation is orientated toward the rotation and slewing angles.
A programmed workings with a multipointed tool is graphically represented by a single circle with the same diameter as the one of the first head's point.
For a more detailed description some different enabling options that concern working graphics, are available in *TpaEdi32 customization*

8.1.5 Subroutine

Subroutine

The subroutine is a piece-program file implemented with program or subroutine typology.
In the workings bar 3 types of codes have been defined for the application of a subroutine:

SUB it manages geometric transforms and multiple application with free repetition
SMAT it manages geometric transforms and multiple application with matrix repetition
SEMPTY it manages geometric transforms except for the scale factor. It does not manage multiple applications. It can generate emptyings.



NAME	
Description	
□ Comment	
IF (..) ? (..) ? (..)	
Geometry	
□ Relative	
X1	
Y1	
Z1	
Locate the extents re	Not apply
Subroutine	
Rnnn	
Face	
Induced faces	
Excluded faces	
Induced XY	Default
□ Rel <-	
□ Point hook	
Rotation angle (°)	
□ Horizontal mirror%Vert	
□ Vertical mirror%Horiz	
□ Invert	
Stretch factor	
Repetitions	
Property	

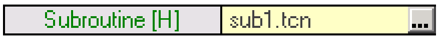
Let us see a few significant examples for the application of a subroutine:

"IF (..) ? (..) ? (..)" node: possibility to condition the application of the subroutine directly on the call working. The sub-program only applies if the condition result is TRUE.

- **Induced faces**: it lists the faces to apply in case of induced calls
- **Excluded faces**: it lists the faces not to apply in case of induced calls
- **Induced XY**: it chooses among different adaptation modes for the application point (positioning) in secondary calls (induced XY)
- **Locate the extents rectangle:** sub-program point of application is located according to overall rectangle, following these options:

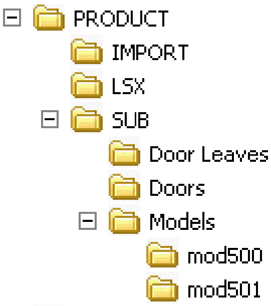
  Center XY: in correspondence with overall rectangle centre
  - **X-Y-:** in minimum overall point in both X and Y
  - **X-Y+:** in minimum overall point in X and maximum overall point in Y
  - **X+Y-:** in maximum overall point in X and minimum overall point in Y
  - **X+Y+:** in maximum overall point in both X and Y
- available geometric transforms:
  - **X1, Y1, Z1**: translation
  - **Rotation angle**: rotation
  - **Horizontal mirror, Vertical mirror**: mirror views
  - **node**: **Stretch factor**: it sets the scale factor to modify the dimension
  - **Invert**: it reverses the execution of the subroutine
- **Relative, Rel->:** absolute or relative mode with respect to the previous listed working
- **Point hook**: possibility to carry on a profile
- **Emptying**: request for the development of emptyings
- **Repetitions**: it sets a multiple application of the subroutine with possibility to choose between free or matrix repetition
- **Rnnn**: it sets the sub-routine "r" variables which can be reassigned. See Chapter ***Assignment of Subroutine Variables***
- **Subroutine**: it can be edited also in parametric form or it can be assigned by opening the file open

window. In this last case the research is set in the standard subroutine archive directory (SUB). The available file types (files of program format only) are listed in the file open window. If a file of format valid for piece-program is selected, the relevant dimensions, comment and graphical preview are displayed in the window. By closing the window, the name of the selected subroutine is shown in the SUB field. Example:


Subroutine [H] sub1.tcn

The full subroutine location path is not displayed, but only *name.extension*. In fact **a relative addressing to the standard subroutine archive directory (SUB)** is maintained. This ensures the portability of programs. In fact, if we copy our program to another machine, it is enough to copy also the *sub1.tcn* subroutine to the SUB directory to make everything work properly. In case of relative addressing, if the program has the macro extension (*.TMCR) it is searched in the macro directory and not in the subroutine (SUB) directory. It is possible to know what type of program is this by reading it and in the case of macro it is not opened. The subroutine name and extension cannot include the following characters: **\ / :* ? " < > | # %**.

The SUB directory can contain other folders, where to store subroutines.
The figure below is just an example:

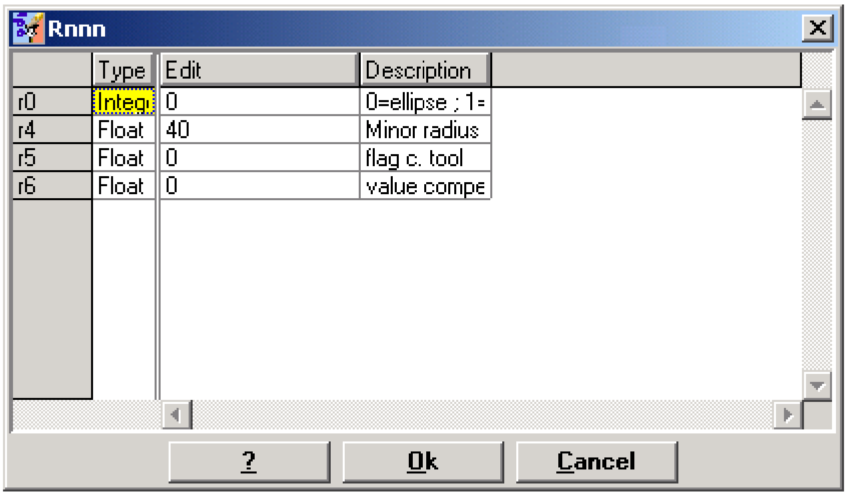


PRODUCT is the standard program archive directory; SUB is the standard subroutine archive directory and it is assigned in the program directory; in SUB the following folders are created: DOOR LEAVES, MODELS, DOORS and each of these folders can contain, in their turn, other subfolders

- if the SUB1.TCN subroutine were selected in "...\.PRODUCT\SUB\MODELS\MOD500\", the SUB field would be assigned as follows: "MODELS\MOD500\ SUB1.TCN"
- if the SUB1.TCN subroutine were selected in the "....\PRODUCT\" program directory, the SUB field would be assigned as follows: "..\SUB1.TCN": also in this case a relative addressing is maintained to ensure program portability
- if the SUB1.TCN subroutine were selected in a subfolder of the "....\PRODUCT\DOOR LEAVES\" program directory, the SUB field would be assigned as follows: "..\DOOR LEAVES\SUB1.TCN": also in this case a relative addressing is maintained to ensure program portability
- if the subroutine were selected out of the program directory: the SUB field would assign the full subroutine location path, without ensuring program portability.

Assignment Of Subroutine Variables

The **Rnnn** item enables to set the subroutine <r> variables which can be reassigned and edited only in the dedicated window. Among the set “r” variables, only those which can be reassigned in the subroutine are displayed. This option is not managed if the SUB field is not assigned or, if it is assigned, is invalid or if the subroutine has no variables which can be reassigned.
The window can appear as in the Figure below:

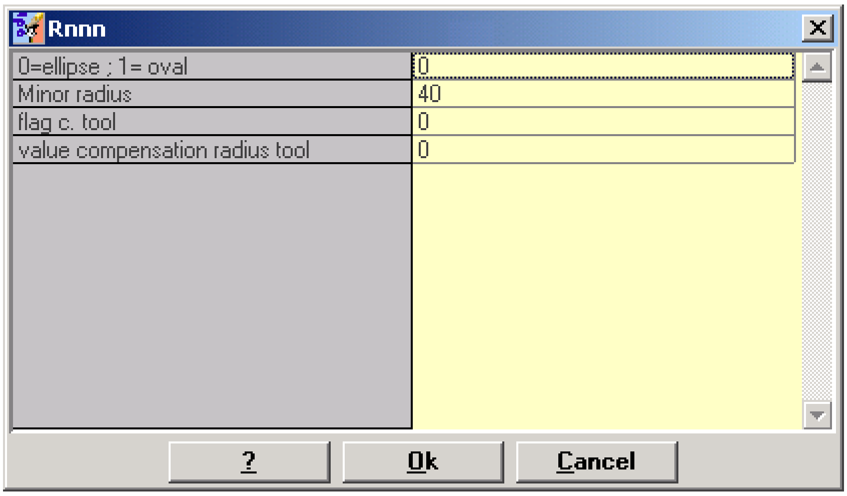


	Type	Edit	Description
r0	Integ	0	0=ellipse ; 1=
r4	Float	40	Minor radius
r5	Float	0	flag c. tool
r6	Float	0	value compe

it shows a table similar to that for the assignment of the "r" variables to the program, but with a lower number of columns:

- **header** (r0,..): it provides the variable name
- **type**: it sets the variable type (the column cannot be modified)
- **edit**: variable assignment field
- **description**: it displays the variable description (it cannot be modified).

The window layout may be different from that previously described and it can appear as follows:



0=ellipse ; 1= oval	0
Minor radius	40
flag c. tool	0
value compensation radius tool	0

it shows a list more similar to that for the assignment of other working parameters:

- **description** it provides the variable description and the variable name if the description is not assigned
- variable assignment **field**.

It is possible to choose between the two displayed windows in Set->Customize->Editor.
At the insertion of a subroutine the column fields are reset to the values assigned to the subroutine variables. If the field ist empty, one of the following two cases can occur:

- 0 value is imposed
- the value assigned to the variable in the subroutine text is imposed. Its behaviour depends on how Tpaedi32 is configured by the machine Constructor.

The buttons of the Toolbar on the left side of the chart allow to:

Import the assignments of all the variables from the subroutine

✕ Reset the value of the selected variable

✕ Reset the value of all the variables

For setting variables the same considerations followed for each other working field are taken into account. To all intents and purposes it is about information relative to the working that is being assigned, only with a higher degree of configuration. If the subroutine is edited or if the subroutine name changes, the variable window may change.

In particular, it is possible to use each valid setting parameter. The figure shows a few assignment examples:

- r0: it uses the r5 program variable
- r1: it is assigned only numerically
- r12: it uses the length of the face the subroutine is applied to

Any set “r” variables of the subroutine which cannot be reassigned are recalculated based on the new settings. In the subroutine text, for example, two not reassignable variables are assigned:

- r100=lf-r0*2
- r101=r10

The r100 variable value is assigned with:

- lf: length of the face the subroutine is applied to
- r0=r5+32

The r101 variable value is assigned with the r10 value, as assigned in the subroutine. If the subroutine does not assign r10, the variable is searched among those of the program the subroutine is applied to.

Automatic assignation

Rnn-variables are assigned when in an subroutine one ore more r-variables are used without any value assigned. When the subroutine is called by a program, the above variables are searched into the calling program. In case of multiple calls in "cascade" the research of the value to be assigned goes on until the program where the the value is assigned.

It is an useful mechanism for a fully automatic passage of one or more information into the subroutines, if an entire archive of programs always uses these information.

However, by improper use some undesired results come possibly out. This is the reason why his usage is recommended only if it is really necessary. The rule must be to use only the assigned variables.

For a better understanding this example should be analysed, as follows:

in a subroutine the variable r0 is used to determine the diameter of a drilling tool. The variable is left without assignment. In this case the r0 value is determined as worthless and of numerical type.

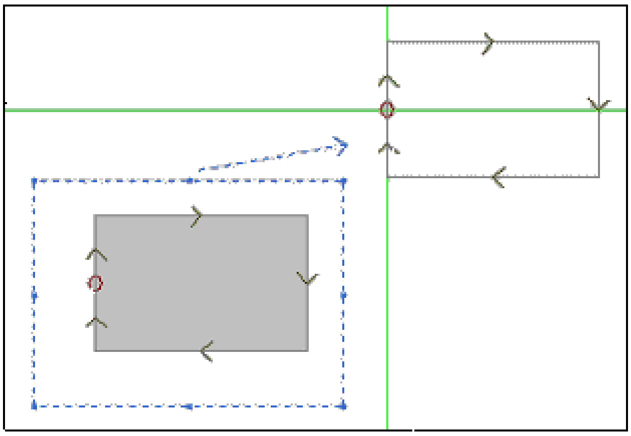
The subroutine is recalled into a new program:

- if the program does not assign the r0 variable, all remains unchanged;
- if the program assigns value 10 to the r0 variable, the subroutine application changes. The diameter of the drilling tool now is worth 10.0.

How to locate a sub-program

A sub-program is located in XY plane of face with depth to Z direction, perpendicularly to face plane: values calculated for three coordinates (x, y, z) define the ***point of application*** (to be indicated like P1). The point of application is programmed in a system of **Cartesian coordinates** to be possibly assigned in absolute or relative mode.

When **relative** mode is selected, absolute mode can be forced on a single coordinates, by placing "a;" before coordinate setting.



The rectangle selected in figure represents the sub-program development (a rectangle crossed clockwise, with initial point located in the middle of left vertical side).

The cross cursor indicates P1 point of application:
sub-program initial point is located in P1 (rectangle setup).
If coordinates in P1 point are not assigned (empty field), no translation applies from sub-program original position. For example, if two coordinates only are setup for P1 in XY plane, rectangle positioning in Z remains unchanged.
If relative positioning mode is active and working is preceded by another complex code (macro or SUB working) **Rel <-**field is also evaluated. If enabled too, point of application P1 is considered as relative to point of application (P1) of previous working.

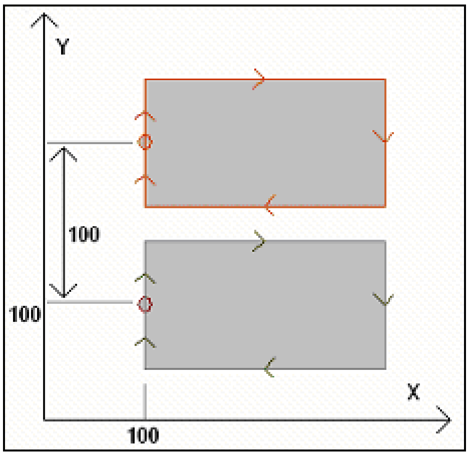


Figure corresponds to two applications of example sub-program (making a rectangle):

- on bottom, the point of application is program as absolute at (X=100; Y=100);
- on top, the point of application is programmed as relative, with **Rel<-** field selected and coordinates x = 0 and y = 100:
  - relative coordinate X=0 sets x coordinate of point P1 to the same x coordinate of point P1 in the first application (bottom rectangle)
  - relative coordinate Y=100 sets y coordinate of point P1 by adding 100 to y coordinate of point P1 in the first application (bottom rectangle)

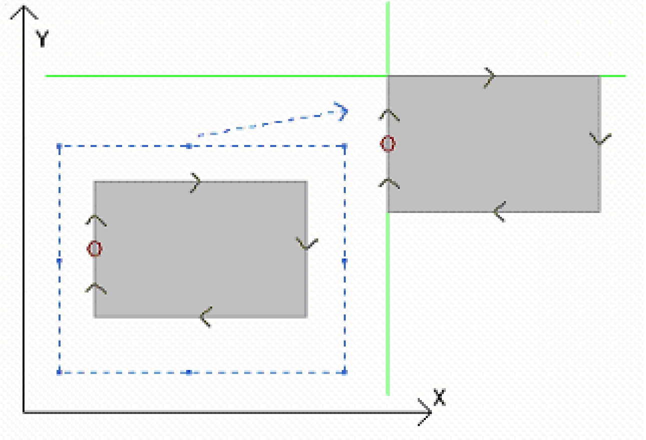
Is it possible to differently choose the sub-program point to be taken to P1, for example by referring to rectangle centre, instead of working?

Yes, in different ways.
One method provides for the selection of **Place overall rectangle** item in sub-program application SUB

code assignment window. This is a multiple selection field, with the following items:

- **No application**: field does not affect sub-program positioning
- **XY centre**: sub-program overall rectangle centre is taken to P1
- **X-Y-:** minimum overall point in both X and Y is taken to P1
- **X-Y+:** minimum overall point in x and maximum in y is taken to P1
- **X+Y-:** maximum overall point in x and minimum in y is taken to P1
- **X+Y+:** maximum overall point in both x and y is taken to P1

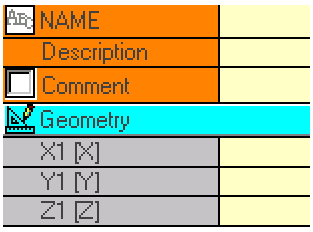
This figure shows changes in rectangle application when *X- Y+* is selected:



Please note:
Sub-program is so easy that its overall rectangle coincides with *figure* planned.

Programmed point of application

Coordinates of point to be translated to P1 point of application can be programmed in sub-program itself. Refer to **Point of application** logical instruction for programming:



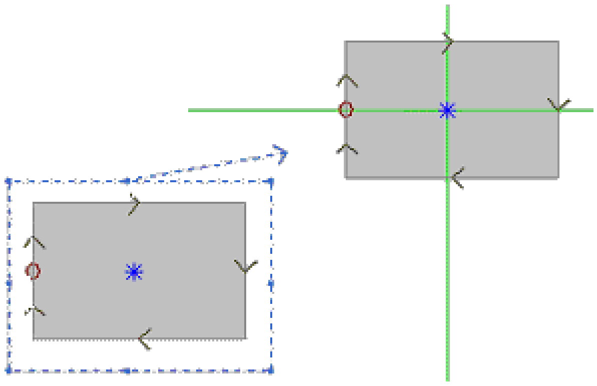
NAME	
Description	
□ Comment	
Geometry	
X1 [X]	
Y1 [Y]	
Z1 [Z]	

Three fields X1, Y1, Z1 assign the point to be located when sub-program itself is recalled.
Programming is interpreted in absolute coordinates and valid for all three coordinates: value 0.0 is taken for non set up fields, if any.
Please note:
Code is **only** interpreted for sub-program application.

A single point of application is recognized as significant, i.e. the first point checked by logical conditions. It is recommended to enter POINT OF APPLICATION instruction among the first sub-program lines.

The point of application assigned here must not necessary correspond to a work position. In the example, rectangle centre coordinates can be reasonably set up, as programmed in the sub-program.

Figure shows changes in sub-program application, by adding POINT OF APPLICATION instruction for rectangle centre.



POINT OF APPLICATION instruction in sub-program is ignored, if sub-program application SUB code sets up a valid selection at **Locate the extents rectangle** item.

Point hook

Point hook

The selection of point hook option:

- has relative mode with zero shifts for three coordinates in point of application (P1) (it makes different setups useless in point P1);
- it makes field selection useless: **Rel <-** and **Locate the extents rectangle**
- ignores POINT OF APPLICATION instruction set in the sub-program.

The point hook **always** applies to a relative programming with zero shifts.
If a profile element, where a hook can be executed (setup, arc or line, other complex working, SUB code or macro, ending its development by a profile element), is entered before (program previous line) a SUB code and if the current sub-program starts with a profile element where a hook can be executed (setup, arc or line):
the sub-program application continues profile downstream, really excluding the setup running, which starts with sub-program itself-
In this case, the point hook has recognised a situation where a **profile connection** occurs.

Final application point

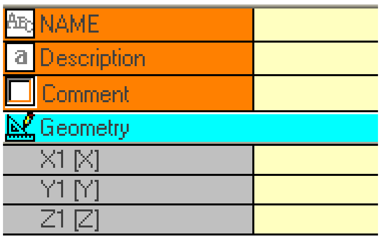
It depends on the working typology itself to determine the last working made in the development of the subroutine. When it is a matter of:

- a punctual working or a setup (e.g. a single drilling), the last point worked is determined by his application point.
- a profile trait (line or arc), the last point worked is determined by the last point of the trait
- a subroutine, the last point worked is determined by the development of the subroutine.

Let us define the following point: ***final application point.***
In a subroutine this is important for the application of:

- repetitions in execution of the subroutine itself
- subsequent workings with assignment of coordinates in a relative mode or in case of propagation of quote.

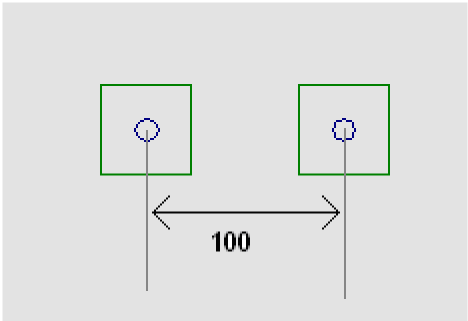
In a subroutine it is possible to program the coordinates of the FINAL APPLICATION POINT using the logical instruction appointed before.
This instruction is interpreted only within a subroutine and in the case of usage of more instructions within the same subroutine, the last one verified by the logical conditions is considered true.



NAME	
Description	
Comment	
Geometry	
X1 [X]	
Y1 [Y]	
Z1 [Z]	

The three fields of X1,Y1, Z! set up the final application point, that does not necessarily need to coincide with the coordinates of a working point.

The programming of the coordinates occurs in absolute mode and for the coordinates, that were not set up, the value 0.0 is taken up.



In the example on the side, we can see a subroutine with a hole and a construct rectangle around the hole. The FINAL APPLICATION POINT is set up, so that it coincides with the hole. In this way the call, e.g., in relative of the same subroutine, that was programmed with displacement in X equal to 100, determines its displacement in respect to the hole.

The usage of the instruction FINAL APPLICATION POINT excludes the possibility of hooking the subroutine after the working and recognizing the profile continuation. The instruction FINAL APPLICATION POINT is ignored, if in the code SUB of subroutine call is set up am **Inversion** transform.

Application Of Workings To the Correct Face

A subroutine is a piece-program file independently whether it is implemented with program or subroutine typology. Therefore, subroutine workings are applied to one or more faces. The face to which workings have been applied is configured in the **Face** field of the SUB working. It has two possibilities of functioning:

- to assign no Face setting. Induced call procedure)
- to set the Face field. (Direct call procedure)

Induced calls

Induced Call Procedure

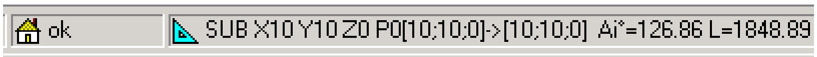
In each program face which recalls the subroutine a SUB working is entered automatically. This last recalls the corresponding subroutine face, only if it contains some workings.

For example:

- let us develop and save the ONE subroutine with assigned workings:
  - holes in face 1
  - a spline in face 3
  - holes in face 4
- now let us create the PRG1 program and select face 1:
  - let us enter a SUB code, which recalls the ONE subroutine, and leave the Face field unassigned. The workings assigned to face 1 of ONE are executed
  - opening the face selection list, we can see that a working is programmed also in faces 3 and 4. Directly in Overall View, the Piece Overall View area highlights immediately that also the workings assigned to faces 3 and 4 in ONE are executed
  - in face 3, we find a copy of the call which is entered in face 1: it is about an **induced call**. Same

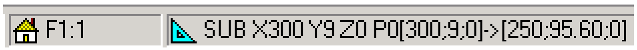
  situation in face 4
  - let us save now the PRG1 program
- let us modify the ONE subroutine by assigning some workings also to face 5
- let us reopen now the PRG1 program. We find that an induced call has been entered also into face 5
- let us modify again the ONE subroutine by cancelling all workings of face 3
- let us reopen now the PRG1 program. We can see that no induced call has been entered into face 3 (the face 3 of the subroutine is in fact empty).

If we try to modify an induced call we understand immediately that it is not possible, in the same way as we try to eliminate it.
The program lines corresponding to induced calls are automatically managed from the software: they always follow the directly programmed lines and cannot be directly modified.
On the status bar the **ok** message near the home symbol means that the current (SUB) working determines some induced calls.Current call is also indicated like **master call**



ok | SUB X10 Y10 Z0 P0[10;10;0]->[10;10;0] Ai°=126.86 L=1848.89

On the status bar the **F1:1** message near the home symbol indicates the face (F1) from which the induced call derives and the program line in face 1 (in this case line 1).Current call is also indicated like **slave call**.
If you click in this area of the Status bar the program activates the face view and the program line that correspond to the *master call*



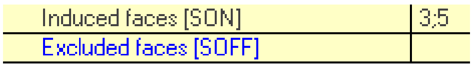
F1:1 | SUB X300 Y9 Z0 P0[300;9;0]->[250;95.60;0]

The induced call procedure is managed only at the *basic programming level*. To be clearer about this last concept let us carry on with the example above:
- let us reopen the PRG1 program and insert a few holes in face 3
- now let us develop the PRG2 program and start programming face 3:
  - let us enter a SUB code, which recalls the previously saved PRG1 program and leave the Face field unassigned: the workings assigned to face 3 of PRG1 are executed
  - opening the face selection list we can see that a working is programmed also in face 1. In Overall View, the Piece Overall View area highlights immediately that also the workings assigned to face 1 in PRG1 are executed. Now the recall of the ONE subroutine does not determine any induced call, since it is not at the basic programming level.

If PRG1 had been implemented with subroutine typology, it itself would have stopped the induced call procedure.

Selection of induced faces

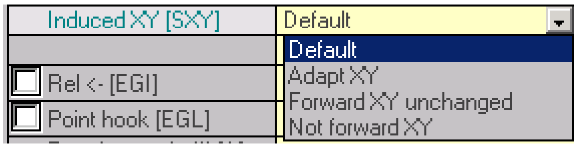
Induced call application can be selective:



Induced faces [SON]	3;5
Excluded faces [SOFF]	

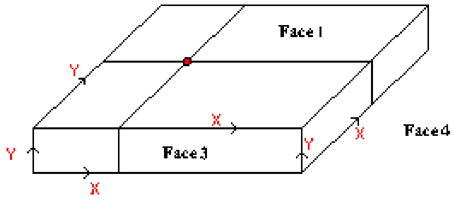
Induced faces: if set, it indicates faces involved in induced call. In figure: "3;5" setting indicates application of induced calls in faces 3 and 5 only
Excluded faces: if set, it indicates faces not involved in induced call. "3;5" setting indicates application of induced calls in all faces, except faces 3 and 5. *Excluded faces* field is only interpreted if *Induced faces* field is not set up.
In both case, list face numbers separated by character.

Positioning of induced calls



In an induced call, the point of application can be assigned in different ways, by selecting among various items in **induced XY** field:

- **Default:** field does not affect positioning (it applies mode assigned in *TpaEdi32 configuration:* it is one of the following selections)
- **Adapt XY:** it adjusts the point of application
- **Forward XY unchanged:** for each induced call, it forwards fields as set up in master call
- **Not forward XY:** for each induced call, it forwards non-setup fields.



It is necessary to adjust the point of application for induced calls as X and/or Y axes could not correspond between different faces.

Let's consider this figure (three visible faces of piece are shown)

- a sub-program application is assigned in face 1: the point of application is indicated on face 1 plane
- calls are induced in the other two faces shown, 3 and 4

Now, X and Y axes are examined in induced faces:

- face 3: X axis physically corresponds to X axis in face 1, while Y axis has no physical correspondence with Y axis in face 1;
- face 4: X axis physically corresponds to Y axis in face 1, while Y axis has no physical correspondence with X axis in face 1.

Thus, automatic association could logically appear:

- face 3: application coordinate X = application coordinate X for face 1; application coordinate Y not assigned;
- face 4: application coordinate X = application coordinate Y for face 1; application coordinate Y not assigned.

The following table examines correspondences applied with XY parameters Induced=Adapt XY:

Master face	***Induced face***	***Coordinate in induced face***
(1,2)	(4,6)	X = Y coordinate from master face (if not set up ="") Y = ""
(1,2)	(3,5)	X=X coordinate from master face Y = ""
(3,5)	(1,2)	X=X coordinate from master face Y = ""
(4,6)	(1,2)	X = Y coordinate from master face (if not set up ="") Y=""
(any other case)	(any other case)	X=X coordinate from master face Y=Y coordinate from master face

In particular, an induced call in a fictive face always applies the same X and Y settings of master call.

Direct calls

Direct Call Procedure

In this case the workings of the faces expressly declared in the Face field are entered into the program which recalls the subroutine.

Always with reference to the previous example, any face of the ONE sub-routine can be applied to face 1 of the PRG1 program, just by typing the corresponding number in the Face field.

Apply geometric transforms

When a sub-program applies, some geometric transforms can be activated, applied in the order below. If the sub-program applies a macro complex code where the required transform is not allowed, the user is warned by an error message.

Inversion
The sub-program inversion implies the inversion of execution order for developed workings: the last block ranks first and so on.
The transform also reverts the settings of:

- Tool compensation: (right or left) of each setup. (See Chapter **Workings->Profile->Tool compensation.**
- selection of entry/exit segments (always on setups) in case of right or left arc settings.

Rotation
The sub-program rotation is set up in numeric field, with rotation angle programmed (in degrees and decimal degrees) in face XY plane from X axis. Rotation occurs around sub-program point of application.

Mirror
A sub-program symmetry is setup in two selection fields:

Horizontal mirror: mirrored execution around a vertical axis
Vertical mirror: mirrored execution around a horizontal axis

If both items are selected, options are summed up.
The transform, only in case of an active selection, also inverts the settings of

- Tool compensation: (right or left) of each setup, only in case of an only active selection.
- selection of entry/exit segments, (always on setups), in case of right or left arc settings.

Scale
It applies a reduction or amplification factor to sub-program and is enabled by the following items:

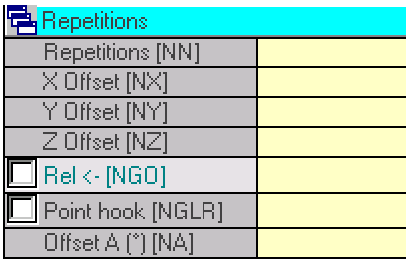
- **Enable**: if selected, it enables transform application;
- **Factor:** reduction or amplification factor (minimum set: 0.001). The following situations are interpreted:
  - lower than 1: reduction applied
  - higher than 1: amplification applied
  - =1: no action.
- **3d scale:** if selected, it also enables in-depth application (face Z axis). Selection is compulsory if the sub-program also runs arcs assigned on a plane different from xy.

Repetitions in sub-program running

SUB codes manage two different modes of sub-program automatic repetition:

- SUB implements multiple application with free repetitions
- SMAT implements multiple application with matrix repetitions

Repetitions with free distribution



Repetitions

Repetitions [NN]	
X Offset [NX]	
Y Offset [NY]	
Z Offset [NZ]	
□ Rel <- [NGO]	
□ Point hook [NGLR]	
Offset A (°) [NA]	

- **Repetitions:** number of repetitions to be <u>added</u> to base application. Minimum value to enable repetition is 1
- **X, Y, Z Offset:** deviations applied to each repetition. Values are applied as relative and added at each repetition
- **Rel<-:** if selected, it applies offsets to the application initial point of previous repetition. An Offset dimension can be forced to be absolute by indicating "a;" before the dimension.
- **Point hook:** if selected, it hooks each repetition to the previous one. In this case, it ignores settings concerning Offsets X, Y, Z and Rel <- field;
- **Offset A(°):** it sets rotation increase by applying it to each following repetition. The initial value is given by value assigned to rotation field in base application. For example, if base rotation runs a 30°-rotation and Offset A(°) is not set, all repetitions rotate by 30°; on the contrary, if Offset A(°)=10°, the first repetition rotates by 40°, the second by 50° and so on for all the others.

Any mirrored transform assigned for base application is also applied to repetitions. In particular, transforms are also applied to the corresponding offsets:

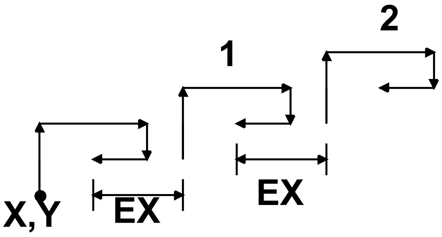
- **Horizontal mirror:** it also mirrors the offset set along the horizontal axis
- **Vertical mirror:** it also mirrors the offset set along the vertical axis

Any scale and/or inversion transform assigned for base application is also applied to repetitions.

Let's see three examples:
the following values are to be set up:

- Repetitions: 2
- Offset X: 100
- Offset Y: 0 (not set up)

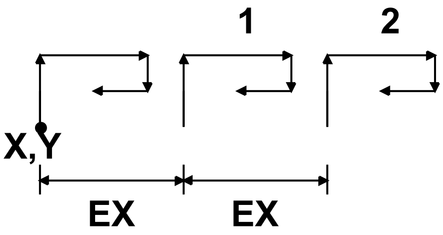
<u>Example 1:</u>



This figure shows the development following inactive **Rel<-** setting:

- **X,Y:** base point of application (can be the point of the overall rectangle, or the application point specified in the subroutine, or the first programmed point).
- **1:** corresponding to the first repetition. Its point of application adds 100 in X to base application final position and 0 in Y
- **2:** corresponding to the second repetition. Its point of application adds 100 in X to base application final position and 0 in Y

<u>Example 2:</u>



This figure shows the development following active **Rel<-** setting:

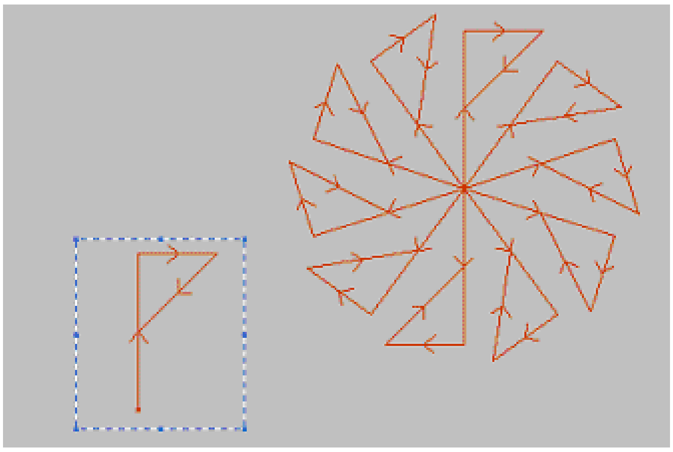
- **X,Y:** base point of application
- **1:** corresponding to the first repetition. Its point of application adds 100 in X to point P1 and 0 in Y
- **2:** corresponding to the second repetition. Its point of application adds 100 in X to first repetition point of application position and 0 in Y

Example 3:

The example uses a toy windmill, with repeated application of a single element.

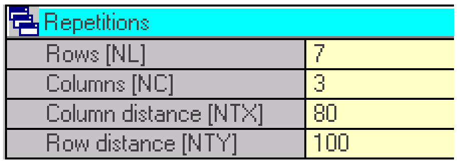
The following values are to be set up:

- Repetitions: 9
- Rel<-: enabled
- Offset A: 360/10



The section highlighted in figure corresponds to the single element, as scheduled in the sub-program. All repetitions are applied to base point of application and rotated to complete the round angle.

Repetitions with matrix distribution



Repetitions	
Rows [NL]	7
Columns [NC]	3
Column distance [NTX]	80
Row distance [NTY]	100

- **Rows, Columns:** number of rows and columns of repetition matrix. Minimum value to be enabled for repetitions is 1, in both fields. The total number of applications made is given by (Row * Columns) product, **included** base application
- **Row distance:** distance between matrix rows
- **Column distance:** distance between matrix columns
- **Rel<-:** if selected, it applies row and column offsets to the application initial point of previous repetition. An Offset dimension can be forced to be absolute by indicating "a;" before the dimension.

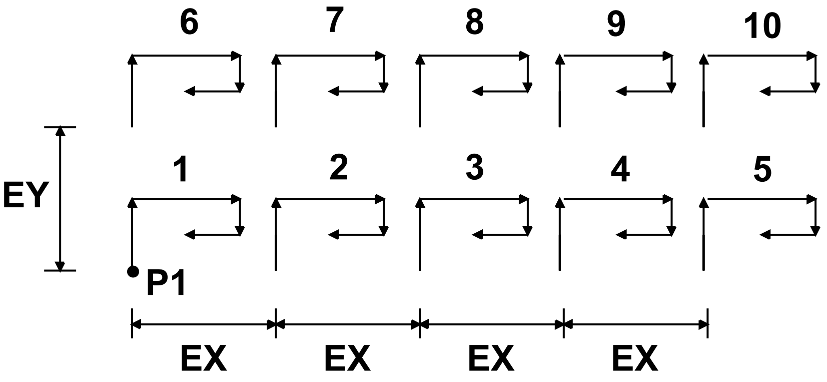
Any mirrored transform assigned for base application is also applied to repetitions. In particular, transforms are also applied to the corresponding offsets:

- **Horizontal mirror:** it also mirrors the offset set along the horizontal axis
- **Vertical mirror:** it also mirrors the offset set along the vertical axis.

Any scale and/or inversion transform assigned for base application is also applied to repetitions.

Let us see an example:
The following values are to be set up:
- Rows: 2
- Columns: 5
- Distance between rows: 100
- Distance between columns: 100

Example:



This figure shows the development following active **Rel<-** setting:
- **P1:** base point of application
- **EX:** distance between columns
- **EY:** distance between rows.

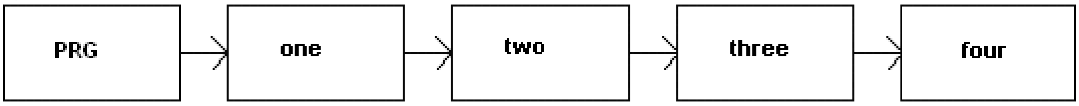
The total number of applications run is (2 *5)=10, **included** base application.

Nesting Subroutine Calls

It is possible to nest complex codes, assigned as macros or subroutine calls but with a nesting limit of 4 calls.

Let us suppose to be editing a program (PRG):
- in a face of the piece we can apply a subroutine call (ONE)
- The ONE subroutine can make calls to other subroutines. For example: to the TWO subroutine
- the TWO subroutine can make calls to other subroutines. For example: to the THREE subroutine
- the THREE subroutine can make calls to other subroutines. For example: to the FOUR subroutine
- the FOUR subroutine cannot make calls to other subroutines

The Figure below shows the situation of maximum nesting:

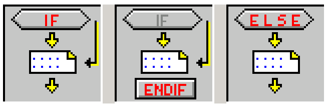


Not necessarily (*one, two, three, four*) shall be subroutines: they can be indifferently also macros. The maximum allowed number of nestings decreases by one in case a subroutine or a macroprogram are being edited.

8.1.6 Logical Instructions

Logical instructions are particular simple workings which generate no profile machining. A logical instruction can assign the conditioned execution of one or more workings or execute it itself a given function, by conditioning it or not according to the value of a logical expression (Examples: ERROR).

IF ... ELSE ... ENDIF structures
Instructions IF, ELSE, ENDIF are included in LOGIC INSTRUCTIONS group.



The condition expressed by IF instruction:

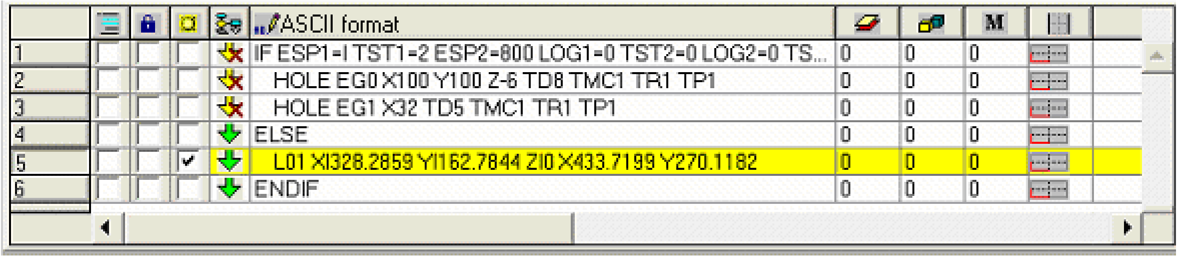
- if TRUE: determines the performance of one or more workings specified downstream IF
- if FALSE: determines the non-performance of workings involved.

ENDIF instruction limits workings conditioned by IF.
ELSE can be assigned between IF and ENDIF and denies the condition assessed by IF.

The result of logic conditions set in a program is visible by requiring the application of logic conditions.

- workings which verify logical conditions are visible
- workings which do not verify logical conditions are not visible

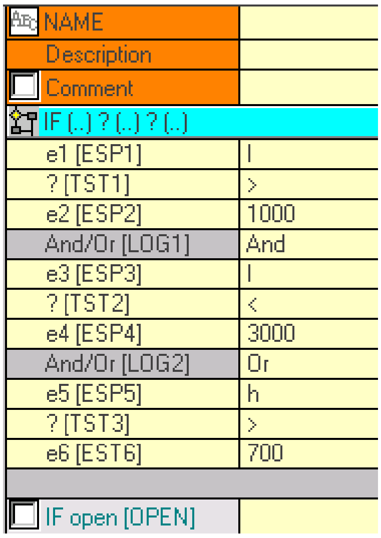
Working logical status is also shown in ASCII text:



					ASCII format			M	
1	□	□	□		IF ESP1=I TST1=2 ESP2=800 LOG1=0 TST2=0 LOG2=0 TS...	0	0	0	
2	□	□	□		HOLE EG0 X100 Y100 Z-6 TD8 TMC1 TR1 TP1	0	0	0	
3	□	□	□		HOLE EG1 X32 TD5 TMC1 TR1 TP1	0	0	0	
4	□	□	□		ELSE	0	0	0	
5	□	□	☑		L01 XI328.2859 YI162.7844 ZI0 X433.7199 Y270.1182	0	0	0	
6	□	□	□		ENDIF	0	0	0	

Figure: ELSE level is verified in IF… ELSE … ENDIF structure

Condition expressed by IF instruction can consist of three terms. Let's see an example:



NAME	
Description	
□ Comment	
IF (..) ? (..) ? (..)	
e1 [ESP1]	l
? [TST1]	>
e2 [ESP2]	1000
And/Or [LOG1]	And
e3 [ESP3]	l
? [TST2]	<
e4 [ESP4]	3000
And/Or [LOG2]	Or
e5 [ESP5]	h
? [TST3]	>
e6 [EST6]	700
□ IF open [OPEN]	

(e1) ? (e2)**: first term**

And/Or**: logic condition between the first and the second term**
(e3) ? (e4)**: second term**

And/Or**: logic condition between the first condition result and the third term**
(e5) ? (e6)**: third term**

(e..) fields appearing in a term are usually set like parameters.

Element **?** between **(e..)** fields of a term assigns a comparison condition for:

<	strict minority	(example: (e1) < (e2))
<=	minority	(example: (e1) <= (e2))
<	strict majority	(example: (e1) > (e2))
<=	majority	(example: (e1) >= (e2))
=	equality	(example: (e1) = (e2))
<>	difference	(example: (e1) <> (e2))

A term is verified like TRUE if comparison condition set up is complied with.

ATTENTION Comparisons between **(e..)** fields are always made below a minimum deviation equal to 0.001 (comparison *epsilon*): values differing by less than epsilon are considered as equal.

The logic condition between two relation terms has the following value:

And if both terms must be verified like TRUE
Or if it is enough when one term only is verified like TRUE.

Possible setups: none, one, two or tree condition terms.
When no term is set up, IF is always true. In this case, when IF also assigns an ELSE, ELSE level is never verified.

IF …ELSE… ENDIF condition structures can be entered without any limit.

Programming shown in figure corresponds to the evaluation of a logic expression:
IF (((l > 1000) and (l < 3000)) or (h > 700)) {...} ENDIF

which means:
if (l) is higher than 1000 and (l) is also lower than 3000;
or: if (h) is higher than 700
then IF instruction is verified like TRUE.

when: l=2000, h=500

(l > 1000)	TRUE
(l > 3000)	TRUE
(h > 700)	FALSE

evaluation: (TRUE and TRUE) or FALSE = TRUE or FALSE = TRUE.

open IF

IF closing with ENDIF instruction is compulsory, unless IF selects **open IF** field.
In this case, IF instruction *only* affects *the following working*, which cannot be:

- setup or profile working
- a logic instruction itself (IF, ELSE, ENDIF) or a point of application (in a sub-program).

No ELSE instruction can correspond to an open IF.

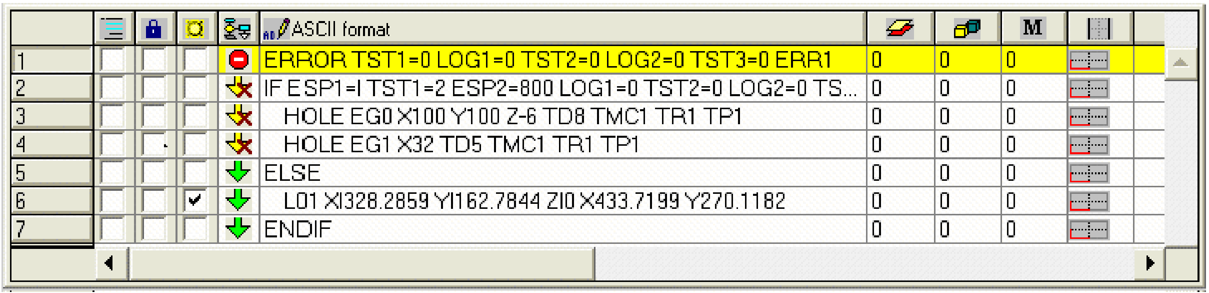
Any improper uses of the instructions IF..ELSE.. are displayed during the application of the logical conditions.
Error conditions are carried in the chapter Error in logical conditions

Error instruction

ERROR instruction allows programming error situations: error condition is expressed in the same format as IF instruction.
If instruction condition is TRUE or is not set up, the instruction interprets an error condition.
If error is generated during the call to a sub-program, its development is not performed and an error is signalled.

If error is directly generated in main program text:

- Tpaedi32 signals an error situation when logic conditions apply. In the example shown for IF instruction:
-



					ASCII format			M	
1	□	□	□	⛔	ERROR TST1=0 LOG1=0 TST2=0 LOG2=0 TST3=0 ERR1	0	0	0	
2	□	□	□		IF ESP1=I TST1=2 ESP2=800 LOG1=0 TST2=0 LOG2=0 TS...	0	0	0	
3	□	□	□		HOLE EG0 X100 Y100 Z-6 TD8 TMC1 TR1 TP1	0	0	0	
4	□	□	□		HOLE EG1 X32 TD5 TMC1 TR1 TP1	0	0	0	
5	□	□	□		ELSE	0	0	0	
6	□	□	☑		L01 XI328.2859 YI162.7844 ZI0 X433.7199 Y270.1182	0	0	0	
7	□	□	□		ENDIF	0	0	0	

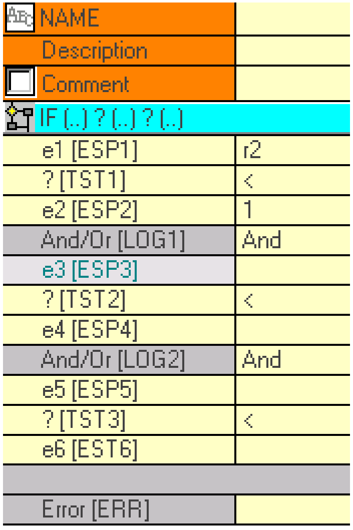
⛔ stop icon on ERROR instruction in line 1 indicates that it is verified like TRUE;

- during execution phase, it cancels program interpretation and stops its execution.

ERROR instruction can effectively manage validity checks on:

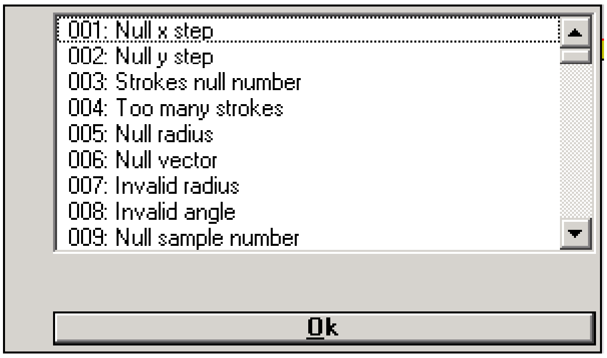
- parameters and/or variables assigned for sub-program recall
- variables assigned in program execution phase.

For error assignment, ERROR instruction manages a data-entry



NAME	
Description	
□ Comment	
IF (..) ? (..) ? (..)	
e1 [ESP1]	r2
? [TST1]	<
e2 [ESP2]	1
And/Or [LOG1]	And
e3 [ESP3]	
? [TST2]	<
e4 [ESP4]	
And/Or [LOG2]	And
e5 [ESP5]	
? [TST3]	<
e6 [EST6]	
Error [ERR]	

When **Error** item is selected, a list is shown where errors assigned are displayed (number + message).



001: Null x step
002: Null y step
003: Strokes null number
004: Too many strokes
005: Null radius
006: Null vector
007: Invalid radius
008: Invalid angle
009: Null sample number

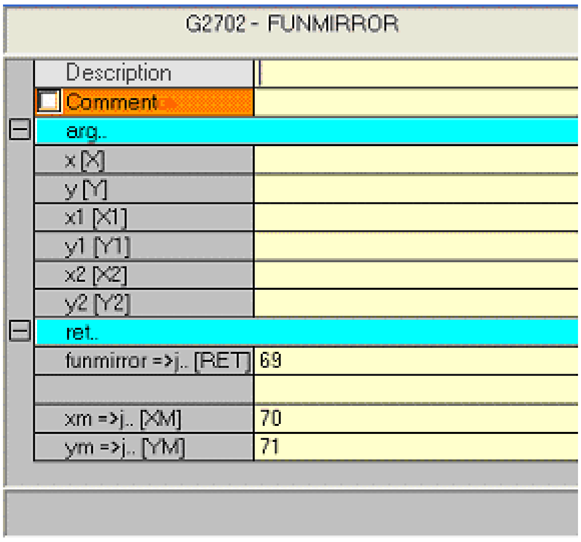
At manufacturer level, a new message can be entered or an existing one can be modified. Double click on the message selected to open a new dialog for entering or editing error messages. To apply the change to all the languages the option **Change the message in all languages** should be enabled.

Global functions

Global functions are special logic instructions which allow performing a more or less complex calculation procedure and directly assigning results in <j> variables.
They must be set up in the configuration phase of an application, by assessing specific customization needs in details.

An easy example follows.
The position of a point P is to be determined, with (r0;r1) coordinates and mirrored around a generic axis assigned by two points: P1 (r2;h/2), P2 (l/2;r3).
a possible way is obtaining formulas for transform required and assigning the first variable *r* for x coordinate and the second variable *r* for y coordinate.
If transformation concerns a single case, this solution can be surely suitable.
On the contrary, if transform is to be calculated several times in different program: formulas are to be remembered and re-written each time.
When using global functions, all formulas can be written once and recalled by using proper instruction, which does not show formula complication and makes results directly available.

Figure shows an example of how a global function could be assigned to recall the procedure which examines a point around a generic axis:



G2702 - FUNMIRROR

Description	
□ Comment	
⊟ arg..	
x [X]	
y [Y]	
x1 [X1]	
y1 [Y1]	
x2 [X2]	
y2 [Y2]	
⊟ ret..	
funmirror =>j.. [RET]	69
xm =>j.. [XM]	70
ym =>j.. [YM]	71

arg.. collects subjects required by instruction:

- coordinates of point to be examined (x;y);
- coordinates of two points on (P1(x1;y1) axis and P2(x2;y2) axis.

ret.. gathers return fields:

- **funmirror =>j..:** set index of <j> variable, which returns function result (here: j69); for example, 1 if correct result, 0 if wrong result
- **xm =>j..:** sets index of <j> variable, which returns x coordinate transformed (here: j70);
- **ym =>j..:** sets index of <j> variable, which returns y coordinate transformed (here: j71);

8.1.7 J Variables

In a few applications it is useful to assign variables during the definition of the face program. For example it is necessary to use them when the program cannot be fully defined a priori or needs some information which derives from the application of subroutines or macros; it is convenient to have local assignments

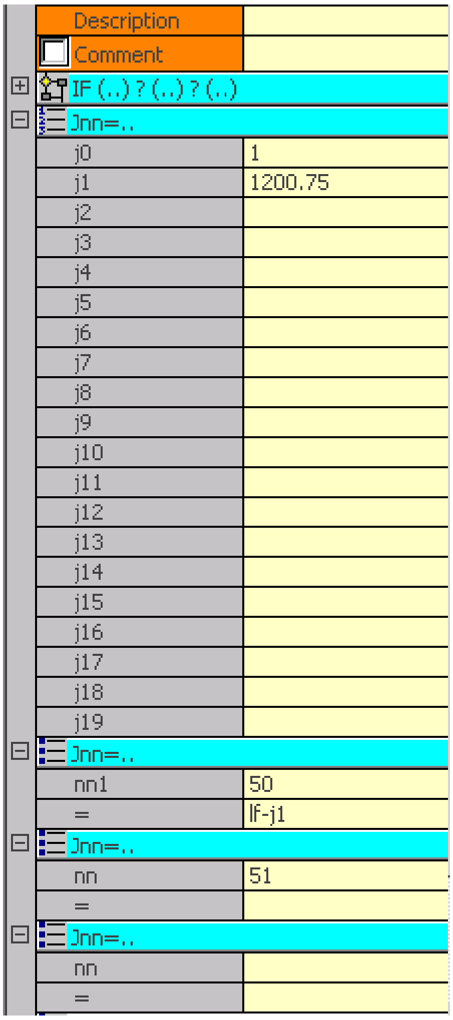
during the development of the face program, instead of grouping them in the “r” variable table.
For this purpose <j> variables are available. It is about 100 variables identified by name: from j0 to j99 of numeric type.
<J> variables are local to a face:

- there is no relationship between the assignment and reading of variables among different faces
- each face program starts with the set of variables preset to zero

<J> variables can be used in each working applied to the face, at any level:

- a <j> variable can be used to fix hole diameter, or a working coordinate, or a logic condition
- inside the face, the visibility of variables is global at any level of application. Thus:
  - the main program can set j5=1
  - the application of a subroutine can modify the j5 value (for example: j5=2)
  - after applying the subroutine, the main program can retest the j5 value finding it changed.

If induced calls of sub-routine are generated, the <j> variables are not reset, but the value set in the main call is kept.



Description	
□ Comment	
⊞ IF (..) ? (..) ? (..)	
⊟ Jnn=..	
j0	1
j1	1200.75
j2	
j3	
j4	
j5	
j6	
j7	
j8	
j9	
j10	
j11	
j12	
j13	
j14	
j15	
j16	
j17	
j18	
j19	
⊟ Jnn=..	
nn1	50
=	lf-j1
⊟ Jnn=..	
nn	51
=	
⊟ Jnn=..	
nn	
=	

Among the data concerning this instruction, descriptive texts associated to each single <j> variable are available. The rows of the window are headed as [MV0].. [MV1]...

Three general instructions for the assignment of <j> variables are defined in the workings palette:

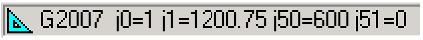
J0=···
······
Jn=···
	ASSIGN Jnn	This instruction allows to assign one or more <j> variables. If necessary, assignments are based on checking for satisfaction of logical conditions (set in **IF (..) ? (..) ? (..)**):

		• assignments are made only if the set logical conditions are verified • the first Jnn=… node groups a certain number of direct assignments: in Figure the items displayed are 20: from j0 to j19. In the example the first two variables (j0=1; j1=1200.75) are assigned • the following nodes enable to assign the same number of variables, by specifying the variable index. In the example two variables are assigned: jnn=50 with If-j1 value and jnn=51 with ifelse [r5;1,j50) value

J0=?.:..
......
Jn=?.:..
	ASSIGN Jnn with condition (.. ? .. : ..)	This instruction allows to assign one or more <j> variables based on evaluation of logical statements (set in **IF (..) ? (..) ? (..)**). For each variable a node is assigned, with three available fields:
		• the first field sets the variable index (value from 0 to 99)
		• the second field shows the assignment to make in case of logical conditions evaluated true
		• the third field displays the assignment to make in case of logical conditions evaluated false

J0=...
......
J99= ...
	ASSIGN Jnn (0 - 99)	This instruction allows to assign all <j> variables. If necessary, assignments are based on checking for satisfaction of logical conditions (set in **IF (..) ? (..) ? (..)**): assignments are made only if the set logical conditions are verified

Status bar layout:



G2007 j0=1 j1=1200.75 j50=600 j51=0

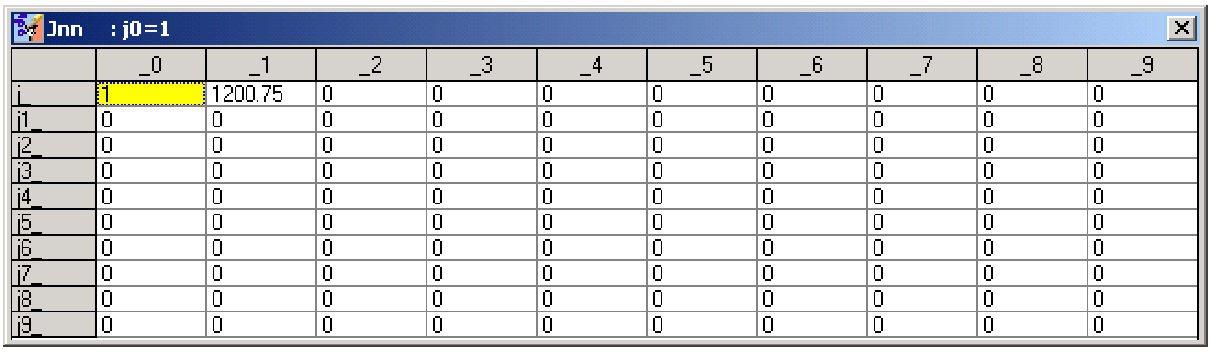
G2007   code ASCII name
j0=1…   it indicates the assignments made by the code.

When the **[FALSE]** message appears on the status bar in place of "j" variables it means that the logical conditions directly assigned to the working are evaluated false and, therefore, variable assignments are not executed.

What Is the Value of J Variables

It may be useful to know the value of <j> variables in a given point of the program. In particular it is possible to have a comprehensive framework of variables after a given working:

- move the active working to the requested point of the program (the highlighted line in ASCII text)
- then, display <j> variables by selecting the relevant command from ***Apply->J Variables*** menu or clicking on the icon in the Main Bar .



	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9
j_	1	1200.75	0	0	0	0	0	0	0	0
j1_	0	0	0	0	0	0	0	0	0	0
j2_	0	0	0	0	0	0	0	0	0	0
j3_	0	0	0	0	0	0	0	0	0	0
j4_	0	0	0	0	0	0	0	0	0	0
j5_	0	0	0	0	0	0	0	0	0	0
j6_	0	0	0	0	0	0	0	0	0	0
j7_	0	0	0	0	0	0	0	0	0	0
j8_	0	0	0	0	0	0	0	0	0	0
j9_	0	0	0	0	0	0	0	0	0	0

The window arranges the 100 <j> variables in a table of 10 rows and the same number of columns:

j_ row: it shows variables from j0 to j9;
j1_ row: it shows variables from j10 to j19;
…..
j9_ row: it shows variables from j90 to j99.

8.1.8 Insertion of Geometric Entities From Drawing Menu

The Drawing menu allows you to insert geometric entities directly, without using the workings palette. When drawing, the mouse pointer is customized and the point indication to be entered is provided in the commands bar area: you are asked to **indicate the <0.00 0.00 0.00> position ("S" snap to entity) (ESCAPE CANCEL)**.

Coordinates can be entered (x,y,z):

- directly by keyboard, following form shown in the commands bar <0.00 0.00 0.00>.
- from pop-up menu open, by pressing mouse right button

P to use the last point item is only active in the phase where initial point of some geometric entities is to be entered, such as line and arc for three points, and assigns coordinates of last element entered.

The first point of segment can be hooked to previous element in the program list. Hooking is performed by continuing the profile if really possible only. For example, if previous element is a point, command activation locates initial point of linear segment on point position, but like the beginning of a new profile. Activation can also be performed by pressing **'P'** key. The Option **T = tangent to entity** is enabled in case of the insertion of a segment carrying on a profile and forcing the linear segment to exit as a tangent line of the profile itself.

Activation can also be performed by pressing **'T'** key

The following numeric fields can be assigned: positive (without sign) or negative (with - sign) and decimal separator. Each value must be separated by a space.

If the **'S' (Snap to entity)** button is pressed or **S Snap to entity** item is selected**,** the point is entered at the coordinates determined by the snap Entity menu option.

- Programmed Point: the point is entered at the programmed point which is nearest to the cursor.
- Near Point: the point is inserted into the coordinates of the point nearest to the cursor. In general, the position is located along a trait (on an arc or a linear segment)
- Midpoint: the point is inserted into the middle point of lines or arcs
- Centre of arc: the point is inserted into the centre of arc, circle or ellipse
- Intersection point: the point is inserted into an intersection of traits
- Point on the perpendicular: the point is entered along a linear segment, an arc, a circle or an ellipse from the previous application point toward the perpendicular to the same line.
- Point on the tangent: the point is entered along a linear segment, an arc, a circle or an ellipse from the previous application point toward the tangent to the same segment.
- Point on change of quadrant : the point is inserted along an arc, a circle or an ellipse, into the point of quadrant change nearest to the position of the cursor.
- Face edge: the point is inserted into the face edge nearest to the cursor position.

On the status bar the sort of snap is displayed during the drawing.

The activation of the snap on programmed entity is restricted to the current insertion and, if necessary, it shall be recalled for the next insertion. Command selection is shown in command area by **[G On]** or **[G Off].** Snap-to-entity prevails on snap to grid.

If the **snap to grid** is active, the point of application is determined as the grid vertex nearest to the click position.

Snap activation/deactivation on grid can occur by pressing **'G'** key or selecting **Snap to grid** item from pop-up menu**.** Command selection is shown in command area by **[G On]** or **[G Off]**. Snap to grid can be activated even if the grid is not displayed.

The options **X Lock x axis** and **Y Lock y axis** prevent the cursor movement toward X and Y. The activation/deactivation can also be carried out pressing the buttons **[X]** or **[Y].** The movement blockage is employed in the current working or it is added to the selected snap entity, if the snap is enabled.

The insertion phase remains active until a graphic representation (e.g.: selection on a menu) is not made or is not cancelled with the **[ESCAPE]** button.

Commands for insertion of geometric entities can be selected on the Draw Bar or Draw menu.

+	***Draw->Point***. It enters point workings. This item is available on menu if a point working

	default code is assigned (for the current face or undifferentiated for face).
	Draw->Line. It enters linear segments. The item is available on menu, if the COPL01 working code is assigned. A linear segment is assigned, according to the indications given in the commands bar area: **segment start point** (if active, it applies the snap-to-grid) **segment end point** (if active, it applies the snap-to-grid). If the working COPL01 does not manage the setting parameters of the initial point of the segment on the plane xy, it should be possible to hook the linear segment on the previous item in the program list and the initial point of the segment is automatically positioned on the hook point.
	Draw->Arc (center,start,end). It enters an arc. The item is available on menu, if the working code COPA01, that manages the assigned parameters of the initial point of the segment on the plane xy, is assigned. An arc is assigned, according to the indications provided in the commands bar area: **center position** (if active, it applies the snap-to-grid) **radius;** **starting angle;** **ending angle.** With the assigned angles, it draws the smallest arc
	Draw->Arc (3 points). It enters an arc defined by three points. The item is available on menu, it the working code COPA04 is assigned. An arc is defined according to the indications given in the commands bar area: **arc start point** (if active, it applies the snap-to-grid); **arc mid point** (if active, it applies the snap-to-grid); **arc final point** (if active, it applies the snap-to-grid). If the working COPA04 does not manage the setting parameters of the initial point of the segment on the plane xy, it should be possible to hook the linear segment on the previous item in the program list and the initial point of the segment is automatically positioned on the hook point.
	Draw->Circle. It enters a circle. The item is available on menu, if the working code COPA45, that manages the assigned parameters of the initial point of the segment on the plane xy, is assigned. A circle is assigned according to the indications provided in the commands bar area: **center position** (if active, it applies the snap-to-grid); **radius**. The circle rotation is assigned clockwise.
	Draw->Ellipse. It enters an ellipse. The item is available on menu, if the working code COPA42, that manages the assigned parameters of the initial point of the segment on the plane xy, is assigned. An ellipse is assigned according to the indications given in the commands bar area: **center position** of the ellipse (if active, it applies the snap-to-grid); **edge point on the first axis** (ellipse starting point) (if active, it applies the snap-to-grid); **edge point on the second axis** (if active, it applies the snap-to-grid). The ellipse rotation is assigned clockwise.
	Draw->Arc of ellipse. Enters an arc of ellipse. The item is available on menu, if the working code COPA43, that manages the assigned parameters of the initial point of the segment on the plane xy, is assigned. An arc is defined according to the data provided in the Commands bar area: **center position** of the ellipse (if active, it applies the snap-to-grid); **edge point on the first axis** (ellipse starting point) (if active, it applies snap to grid); **edge point on the second axis** (if active, it applies the snap-to-grid) or value of the second axis (in the Commands bar area)

	starting angle **ending angle** The rotation of the arc of ellipse can be established during the assignment of the final angle. When **[I]** key is pressed or I reverses rotation item is selected from pop-up menu, rotation direction is changed from Clockwise **(CW)** to Counter-Clockwise **(CCW)** and vice versa. Command selection is shown in command area by **[CW]** for clockwise rotation, **[CCW]** if counter-clockwise.

□	***Draw->Rectangle***. It enters a rectangle. The item is available on menu, if the working code COPL16, that manages the assigned parameters of the initial point of the segment on the plane xy, is assigned. A rectangle is assigned according to the indications provided in the commands bar area: **rectangle first vertex** (if active, it applies the snap-to-grid); **rectangle second vertex**, opposite to the first (if active, it applies the snap-to-grid).
⬡	***Draw->Polygon***. It enters a polygon. The item is available on menu, if the working code COPL17, that manages the assigned parameters of the initial point of the segment on the plane xy, is assigned. A polygon is assigned according to the indications given in the commands bar area: **side number** (3 to 99); **center position** of the polygon (if active, it applies the snap-to-grid); **starting angle** (if active, it applies the snap-to-grid).
⮌	***Draw->Polyline***. It enters a polyline, which can be defined by a contiguous sequence of linear and/or circular (arc) segments assigned by points, according to the indications provided in the commands bar area. The item is available on menu, if the working code COPL01 is assigned. In particular it is possible to: • switch from line to arc by pressing respectively the **[L]** and **[A]** keys or from pop-up menu by selecting **L=switch to line** and **A=switch to arc** items, respectively • close a segment at the polyline starting point by pressing the **[C]** key **or from pop-up menu by selecting** C Close on the starting point **command.** If the working COPL01 does not manage the setting parameters of the initial point of the segment on the plane xy, it should be possible to hook the linear segment on the previous item in the program list and the initial point of the segment is automatically positioned on the hook point.

8.2 Profile

8.2.1 Profile Building

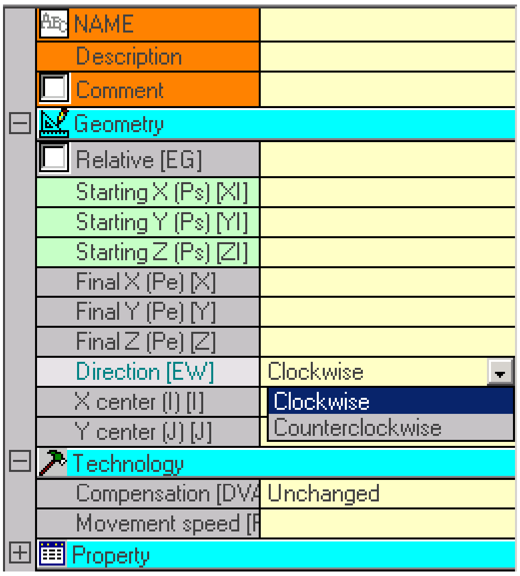
A profile is generally defined as a continuous sequence of linear and/or circular line segments, not necessarily opened by a setup working.

During the machining of the profile, the tool selected for the profile building remains engaged from the start point to the end point of the profile, without any disconnection.

Tpaedi32 allows to assign a profile:

- without putting a setup working as heading (open profile)
- by connecting profile segments defined separately (subroutine or macro calls)
- by entering a subroutine or macro.

A01 working is selected in ARCS group



NAME	
Description	
□ Comment	
⊟ Geometry	
□ Relative [EG]	
Starting X [Ps] [XI]	
Starting Y [Ps] [YI]	
Starting Z [Ps] [ZI]	
Final X [Pe] [X]	
Final Y [Pe] [Y]	
Final Z [Pe] [Z]	
Direction [EW]	Clockwise
X center [I] [I]	Clockwise
Y center [J] [J]	Counterclockwise
⊟ Technology	
Compensation [DVA	Unchanged
Movement speed [F	
⊞ Property	

Initial positioning parameters of a profile section (arc or line) allow assigning a profile without any initial setup. In this case, the profile is named **open**. A profile is interpreted like open if one of these two situations occurs:

- the profile segment (arc or line) has not null setting also in one of the parameters for the assignment of the segment start point only
- neither setup nor another profile segment is assigned before the segment profile

In figure, the former item of rotation direction selection is **Clockwise**, the latter is **Counter-clockwise**. Nevertheless, items can be exchanged between them, according to representation system assigned to the face.

The following figure shows four possible cases of graphic representation with horizontal X axis: cases are numbered 0 to 3. In all cases, an arc is represented, corresponding to the following settings:

- initial (X;Y;Z) = (100;100;0)
- final (X;Y;Z) = (200;200;0)
- centre (X;Y;Z) = (200;100;0)
- rotation = **CLOCKWISE**.

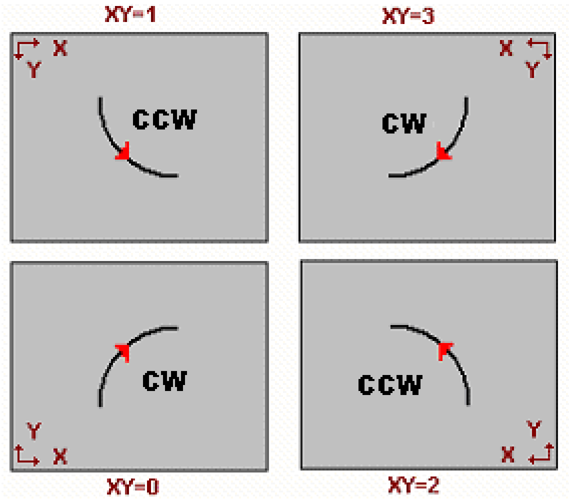


Figure shows the rotation direction writing seen when looking at the arc:

- CLOCKWISE for cases XY=0 and 3;
- COUNTER-CLOCKWISE for cases XY=1 and 2;

Really, arc is always programmed with CLOCKWISE direction (rotation in XY system=0).

If arc is programmed with A01 working, the list corresponding to the rotation direction selection is adjusted to comply with rotation direction selection and visible on the arc, i.e.:

- CLOCKWISE/COUNTER-CLOCKWISE for cases XY=0 and 3;
- COUNTER-CLOCKWISE/CLOCKWISE for cases XY=1 and 2;

PLEASE NOTE:

items shown on the list are exchanged, but value set does not change. For example, for rotation programming under parametric structure: CLOCKWISE is always 0 and COUNTER-

CLOCKWISE is always 1 (or different from 0)

Similar measures are adopted for other relevant assignments, for example:

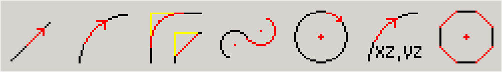
- tool correction side in setup working (items exchanged: LEFT/RIGHT)
- mirror selection for sub-program application (X MIRROR/Y MIRROR)

Please note that these measures are to be adopted when working database is drawn: particularly, be careful when defining complex custom codes. Anyway, they cannot be used for direct programming (ex.: parametric setting of correction side).

Particularly, they cannot be used when a rotation angle is assigned, for example: rotation angle for sub-program application, polar programming of point workings, setups, arcs or lines.

8.2.2 Works on profile

Works on profile are gathered in several groups in the working bar:



- Single line segments: calculating a linear segment;
- Single arcs: calculating an arc on face xy plane;
- Fillets, chamfers: calculating two linear segments or a linear segment and an arc;
- Multiple arcs: calculating two or more arcs;
- Circles: calculating an arc on face xy plane;
- Arcs on xz, yz, xyz planes: calculating an arc on xz, yz, xyz planes;
- Polygons: calculating a close figure (polygon, oval, ellipse).

In case of multiple section (working calculating more sections), single section can be *seen* on ASCII text expanded list.

8.2.3 Point of application

Works on profile have point of application in working final point. In case of multiple section, point of application is the final point of the last section calculated.

Each profile code calculates a specific geometry on a plane.
Let's see some examples, among various possibilities.

- **L2 [xy(pole, U, A), Zf]:** calculating a linear section in the area assigned in two geometric components:
  - XY plane: linear section defined in a polar system
  - Z direction: with single component perpendicular to face plane

  Working calculates a linear section in space, where each axis has a linear movement.
  Depth axis is Z.

- **A4 [xy(P1,Xf,Yf), Zf]:** calculating an helix assigned in two geometric components:
  - XY plane: circular section defined in Cartesian system like an arc for three points
  - Z direction: with single component perpendicular to face plane.

  Working calculates an helix in space, with helix axis perpendicular to face plane and circular development assigned in face plane (XY).
  Depth axis is Z.

- **A5 [xy(Xf,Zf,centre,rot), Yf]:** calculating an helix assigned in two geometric components:
  - XY plane: circular section defined in Cartesian system like an arc for three points
  - Y direction: with single component perpendicular to XZ plane.

  Working calculates an helix in space, with helix axis parallel to Y axis of face and circular development assigned in face plane (XZ).
  Depth axis is Y.

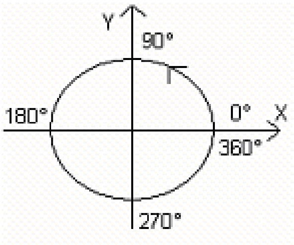
- **A9 [xyz(Xf,Yf,Zf,centre,rot)]:** calculating an arc on a generically-oriented plane in space:
  - no depth axis is assigned.

As already said, usually each profile section can directly assign also the section initial point. In this way, the section directly opens profile. If profile section does not assign the initial point, it is located on point of application of working assigned upstream.

8.2.4 Angle programming

Profile codes often use angle settings:
- angles are programmed in degrees and decimal degrees (x.xx°)
- note used is shown in figure; 0° to 360° with counter-clockwise rotation.

Negative angles cover XY plane from X axis and rotating clockwise.



8.2.5 Tangent lines and intercept lines

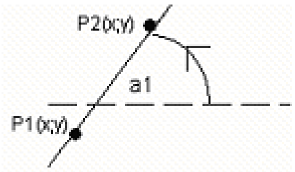
These geometric elements are used in profile codes.

The **tangent line** is a line setting the tangency condition to programmed profile section (line or arc). It can be:
- entry tangent line: if tangency is set on section initial point
- exit tangent line: if tangency is set on section final point

The **intercept line** is a line setting the belonging condition for point of application (section final point) to line itself.
An intercept line can also set tangency condition on section final point.

With a single linear section, there is no difference between entry and exit tangent line.

An **entry tangency** is defined like:
- **default tangency:** if set equal to exit tangent line of previous profile section
- **programmed tangency:** programming is required for:

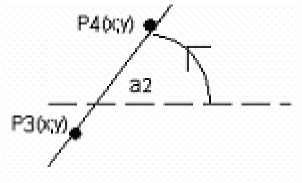


  - inclination angle of the line (a1); or
  - two points (P1 and P2) on line. Line orientation is defined from P1 to P2.

Angle programming prevails on point programming.

An **exit tangency** is defined like:
- **default tangency:** only with circular section ending on profile setup point. It is set up like initial tangent line of first profile section;

- **programmed tangency:** programming is required for:



- inclination angle of the line (a2); or
- two points (P3 and P4) on line. Line orientation is defined from P3 to P4.

Angle programming prevails on point programming.

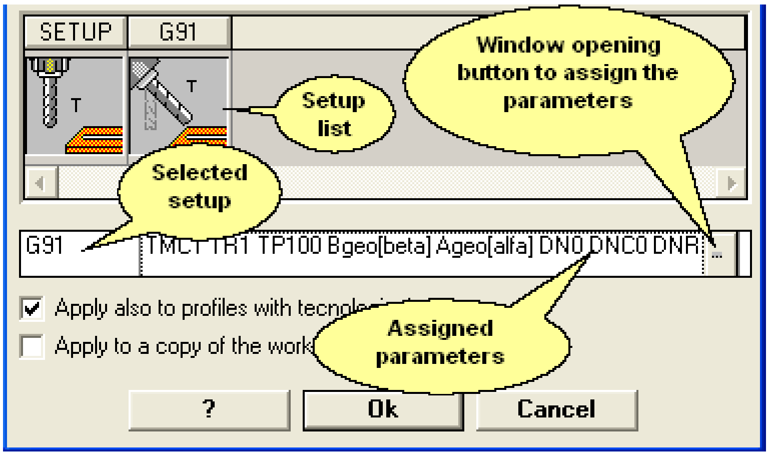
An **intercept line** is always directly programmed. Programming can occur with:
angle (a2) and a point on line (P3); or
two points (P3 and P4) on line. Line orientation is defined from P3 to P4.

8.2.6 Assign Technology

It is possible to assign technological parameters to a profile only by entering a setup working at the opening of the profile itself. The profile setup is not necessarily visible. If for example the profile is fully or also only partially defined in the application of a subroutine (or macro), the setup can be applied internally to the development of the subroutine.
A profile without opening setup has been called open. An open profile has no assigned technology.
In any case, during the machining of the profile, this last always comes with an opening setup and the relevant technology (default technology, as assigned in Customization of Tpaedi32).
Therefore, although the possibility to manage open profiles simplifies the programming procedure, it must always be clear what a program is intended to apply, during machining. If for example the profile building needs a different technology from the default one, it is up to the programmer to assign it directly.
During the execution in the machine different situations can occur according to the assignments in Configuration of Tpaedi32 from the machine constructor.

- profiles programmed open can be excluded from the execution (e.g. in the same way as the construct workings)
- profiles programmed open can be normally executed subject to assignment of the default technology.
- programming open profiles can generate an error condition and therefore the program cannot be executed. In this case the operator should assign directly the technology to each open profile.

Assign Technology Window
A technology can be assigned by direct edit or by selection in the window. See ***Tools->Apply setup to profile*** .

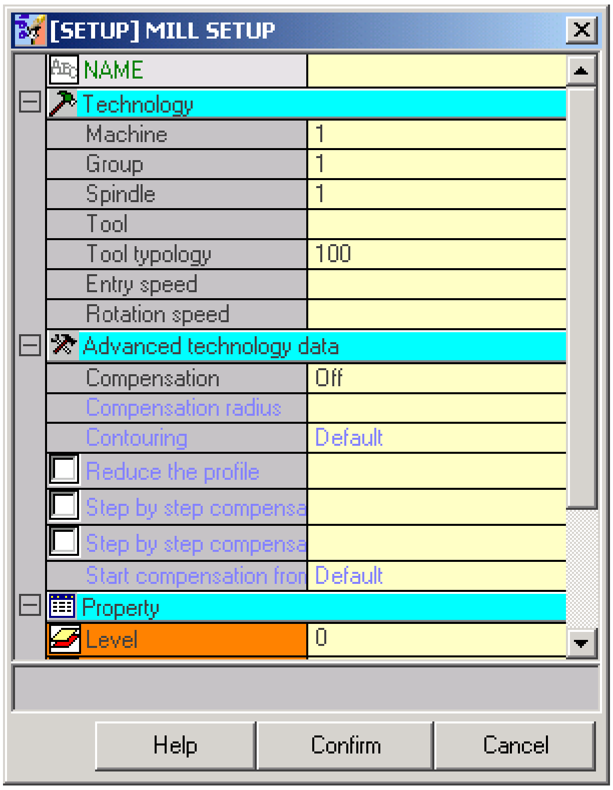


G91 TMC1 TR1 TP100 Bgeo[beta] Ageo[alfa] DN0 DNC0 DNR

☑ Apply also to profiles with tecnolo

☐ Apply to a copy of the work

The code to assign is selected in the window by choosing it among the available setup working codes displayed in the Graphical List (**Setup list**). The selected code name is displayed in text format and clicking the button shown in Figure the window relative to the selected working is opened. In this window technological parameters and working properties, which then are shown in ASCII format, are set. It is not possible to assign geometric parameters. For example, if you select the setup working the displayed window is as shown below:



[SETUP] MILL SETUP

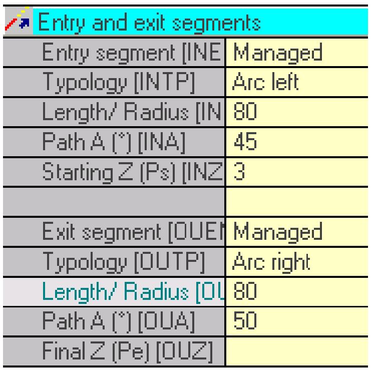
NAME	
Technology	
Machine	1
Group	1
Spindle	1
Tool	
Tool typology	100
Entry speed	
Rotation speed	
Advanced technology data	
Compensation	Off
Compensation radius	
Contouring	Default
□ Reduce the profile	
□ Step by step compensa	
□ Step by step compensa	
Start compensation fron	Default
Property	
Level	0

8.2.7 Opening and closing a profile

In a profile setup working, it is possible to assign the way the profile should be opened and/or closed.
It is possible to add an opening/closing segment and select its type (linear segment or arc), its length and the depth variation of the segment.
Opening and closing segments are **always realized with tangent continuity**:

- the opening segment moves the setup point in relation to the programmed position
- the closing segment is realized after the last programmed segment of the profile.

Opening and/or closing segments are not realized in case of isolated setups. They are displayed in view of correction tool. If the view of correction tool is active, the geometric information about the traits is displayed in the status bar.



Entry and exit segments	
Entry segment [INE	Managed
Typology [INTP]	Arc left
Length/ Radius [IN	80
Path A (°) [INA]	45
Starting Z (Ps) [INZ	3
Exit segment [OUEI	Managed
Typology [OUTP]	Arc right
Length/ Radius [OU	80
Path A (°) [OUA]	50
Final Z (Pe) [OUZ]	

Four **types** of segments can be selected:

- **Line:** linear segment
- **Arc left:** arc on the plane xy to the left of the profile
- **Arc right:** arc on the plane xy to the right of the profile
- **Arc 3d**: arc on the oriented plane

Other settable values are:

- **Length/Radius:** if the type of segment selected is Line, segment length is set in the plane of the face, otherwise the arc radius is set. Minimum settable value is 50*epsilon quote. The set value, if both the entry segment and the exit segment are enabled, but for this last one a value was not assigned, is propagated from the entry segment to the exit one.
- **Path A (°):** if the type of segment selected is Arc, the arc angle is set. If the value is not set, the default value is 45°. Minimum value is 1°, maximum value is 270°, if the arc lays on the plane xy, otherwise the maximum value is 90°. The set value, if both the entry segment and the exit segment are enabled, but for this last one a value was not assigned, is propagated from the entry segment to the exit one.
- **Movement speed:** sets the speed interpolation on the segments. If on the entry segment any value is not set, the assigned speed on the first segment of the profile is used. If on the exit segment any value is not set, the assigned speed on the final segment of the profile is used.

For the entry segment:

- **Starting Z:** sets the initial depth of the segment. The initial depth of the segment is the depth assigned for setup. Its programming is absolute and, if the value is not set, the default value is the value assigned to the **Qz** field (depth assigned to setup). If the arc lays on a oriented plane, the geometry of the segment depends on the initial segment. If the initial segment is
- an arc on the plane xy an arc on the plane xyz is executed
- an arc on the plane xz an arc on the plane xz is executed
- an arc on the plane yz an arc on the plane yz is executed
- a linear segment an arc on the plane xyz is executed

In this case the initial Z is significant, if:

- the radius of the arc is not set and it is taken from the value of the variation between the initial Z and the depth assigned for setup.
- the variation sign between the initial Z and the depth assigned for setup determines the solution of the resulting arc, so as to enter from the direction set. If the initial Z is not set, following cases can be distinguished:

1. if the profile starts with an arc, on the entry arc a rotation sense opposite to that of the first profile's arc is imposed
2. if the profile starts with a linear segment, on the entry arc the entry direction of the air quote is imposed

The value set for initial Z cannot be generally applied to the initial point of the arc, because it is determined by the value set for the angle.

For the closing segment:

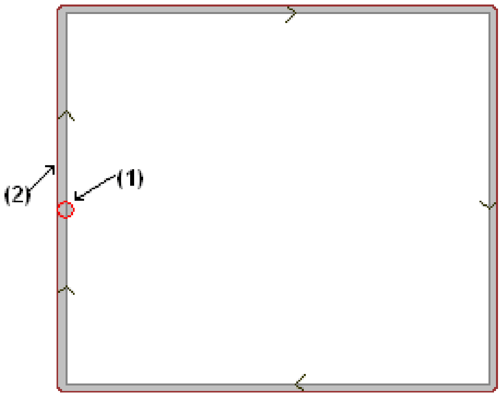
- **Final Z:** sets the final depth of the segment. The initial depth of the closing segment is the final depth assigned for the profile. Its programming is absolute and, if the value is not set, the default value is the value of the final depth of the profile. In case of arc laying onto an oriented plane, the geometry of the segment and the *final Z* are significant in the same way as the entry segment.

8.2.8 Tool Radius Compensation

The request for tool radius compensation activates a mechanism of automatic displacement of programmed trajectories (profiles), to keep into account the actual diameter of the tool which executes the same trajectory.
The command to activate or deactivate tool adjustment is selected from ***Show->Tool radius compensation*** menu or by clicking on the icon on the Special Views Bar .
Compensation tool is applied to the xy plane .
Tool radius compensation cannot be applied to arcs assigned on a plane different from the xy plane, if:

- the original arc is a circle or the arc inverts the direction of the x axis or of the y axis
- the corrected arcs determine a fillet or an intersection solution internal to the segment.

The following example shows how it works:



(1) programmed profile:

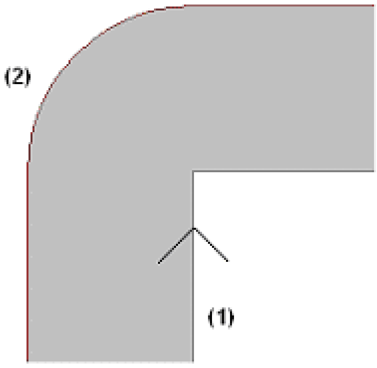
- rectangle followed clockwise;
- the small circle shown on the left vertical side of the rectangle displays the working tool diameter;

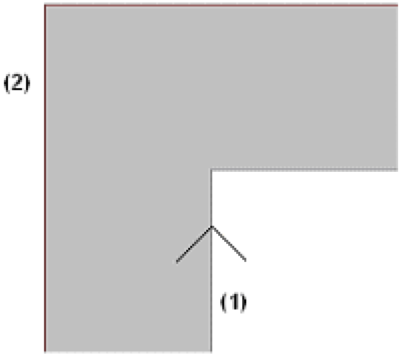
(2) profile obtained by tool radius compensation:

- it is external to the programmed profile and is followed in the same direction (clockwise);
- the distance between the two profiles is equal to the tool radius.

During the machining of the profile, the inner rectangle will have the dimensions with which it has been drawn. According to the requested compensation, the tool works in fact externally to the programmed trajectory.
If it were necessary to respect the dimensions external to the rectangle, the required compensation should be internal to the rectangle.

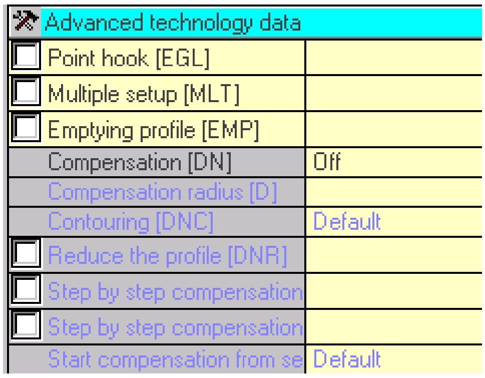
Let us see the detail of a rectangle edge of the example above:





In the left Figure the correct profile moves around the original edge with a radius arc equal to the tool radius; in the right Figure the correct profile continues up to the external intersection point of correct linear segments.
In the first case, the compensation mode with insertion of **fillets** is applied.
In the second case, the compensation mode with insertion of **edges** (otherwise called **contouring** correction) is applied.
The compensation side is established by following the direction of the programmed profile. Setting a compensation radius different from the tool radius allows to increase or reduce the default compensation. The minimum permitted value matches the resolution epsilon set by the machine manufacturer during

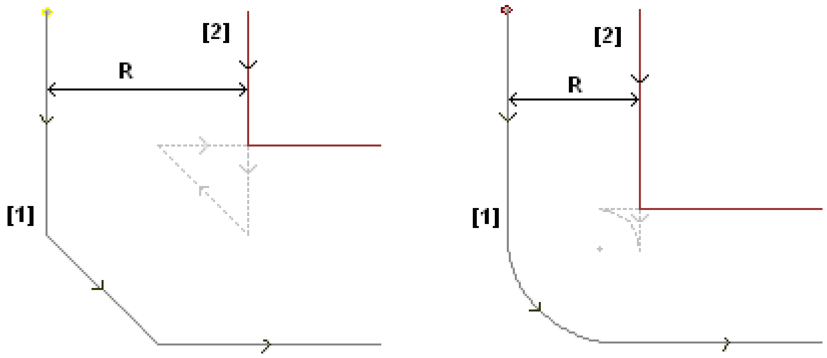
configuration. A setting value lower than epsilon is ignored.
The parameters for the execution of the tool radius compensation are assigned at profile and setup technology level.
Let us examine the parameters which, in a setup working, contribute to define the tool radius compensation:



Advanced technology data	
□ Point hook [EGL]	
□ Multiple setup [MLT]	
□ Emptying profile [EMP]	
Compensation [DN]	Off
Compensation radius [D]	
Contouring [DNC]	Default
□ Reduce the profile [DNR]	
□ Step by step compensation	
□ Step by step compensation	
Start compensation from se	Default

These parameters are grouped at the **Advanced Technology Data** item.
Let us see each parameter in detail:

- **Compensation**: it enables compensation, with direct selection of the compensation side. The listed items are three:
  - **Off** disables compensation
  - **Left** enables compensation on the left side of the profile
  - **Right** disables compensation on the right side of the profile
- **Compensation radius**: it sets the compensation radius, if it must be different from the tool radius. In configuration of Tpaedi32 a different value interpretation can be established. In fact, this one can define the positive or negative variation of correction to apply to the tool radius.
- **Contouring**: it enables the compensation modes for edges. The listed items are three:
  - **Default** enables the default mode (assigned in configuring Tpaedi32)
  - **Fillets** enables compensation with insertion of fillets
  - **Edges** enables compensation with research of edges
- **Reduce profile**: it enables the removal of segments in the correct profile, with respect to the original one, on the basis of geometric clearance restrictions exceeding compensation. The Figure below shows two typical situations, which can only be solved by enabling profile reduction:



(1) programmed profiles,
(2) profiles obtained by tool compensation
R compensation radius

The left Figure shows a profile assigned on three linear segments:

- compensation is applied on the left side of the profile
- the compensation value (R) exceeds chamfer dimensions.

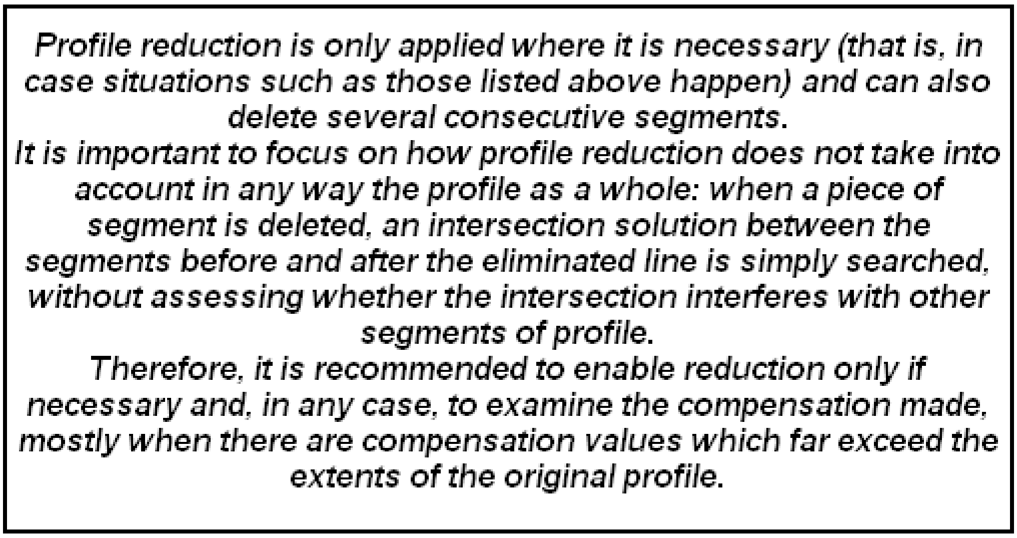
If profile reduction is not activated, profile compensation fails. An error due to excess of compensation on the inclined segment is signaled.
The correct profile (2) is obtained only if profile reduction is enabled: the intermediate segment does

not appear; in fact, it has been eliminated in the projection of segments, for building the correct profile. The dashed segments highlight which would have been the correct profile, if the compensation value applied to the intermediate segment were considered valid. It is clear that the direction of the intermediate segment would be inverted, with consequent alteration of the initial geometry.

The right Figure shows a profile assigned on two edge linear segments and an intermediate fillet:

- compensation is applied on the left side of the profile
- the compensation value (R) exceeds the fillet radius

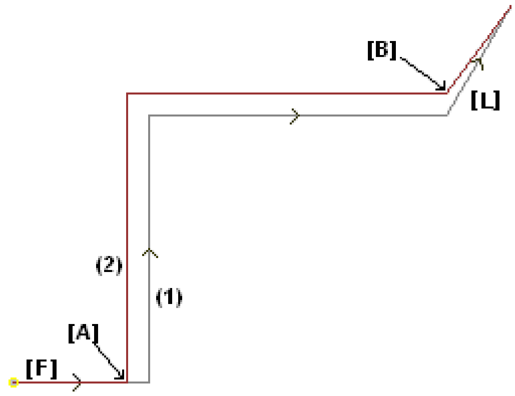
If profile reduction is not activated, profile compensation fails: an error due to excess of compensation on the arc is signaled.
The correct profile (2) is obtained only if profile reduction is enabled: the intermediate segment does not appear; in fact, it has been eliminated in the projection of segments, for building the correct profile.
The dashed segments highlight which would have been the correct profile, if the compensation value applied to the intermediate segment were considered valid: the direction of the intermediate segment would be reversed, with consequent alteration of the initial geometry.



Profile reduction is only applied where it is necessary (that is, in case situations such as those listed above happen) and can also delete several consecutive segments.
It is important to focus on how profile reduction does not take into account in any way the profile as a whole: when a piece of segment is deleted, an intersection solution between the segments before and after the eliminated line is simply searched, without assessing whether the intersection interferes with other segments of profile.
Therefore, it is recommended to enable reduction only if necessary and, in any case, to examine the compensation made, mostly when there are compensation values which far exceed the extents of the original profile.

- **Step-by-step Compensation Start**: it enables the gradual compensation startup on the first segment of the profile. Compensation is calculated from the second segment of the profile and movement on the first segment is linear: from the setup programmed point to the correct starting point of the second segment. In any case, Step-by-step compensation startup is not applied if one of the following conditions is verified:
  - the first segment of the profile is not linear
  - the profile is defined by only one segment
  - the first segment of the profile needs a disconnection in compensation (see later on)
- **Step-by-step Compensation End**: it enables the gradual compensation closure on the last segment of the profile. It is applied only if the last segment is linear. Compensation is calculated up to the last-but-one segment of the profile and movement on the last segment is linear: from the correct end point of the last-but-one segment to the end point of the programmed profile. In any case, Step-by-step compensation closure is not applied if one of the following conditions is verified:

  the last segment of profile is not linear,
  the profile is defined by only one segment,
  the last segment of the profile needs or continues with a disconnection in compensation (see later on).



(1) programmed profile,
(2) profile obtained by tool radius compensation.

The profile applies:

- gradual compensation startup ([F] which is the first segment);
- gradual compensation closure ([L] which is the last segment).

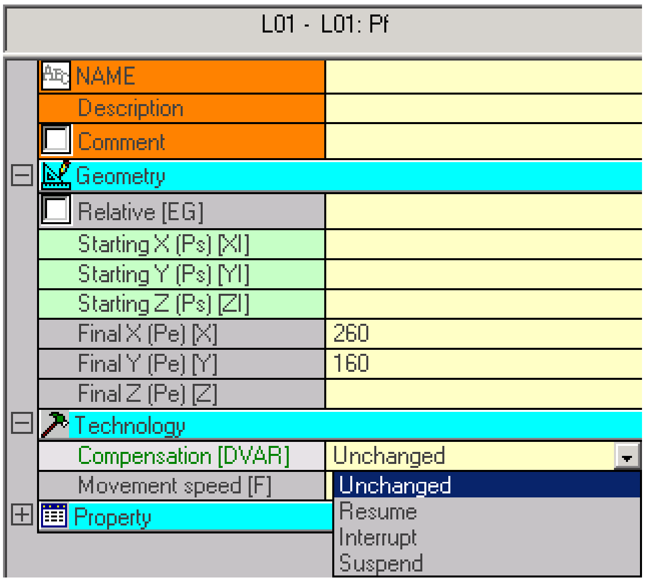
[A] is the correct starting point of the second segment of the profile;
[B] is the correct end point of the last-but-one segment of the profile

- **Start compensation from setup**: it enables compensation from the setup programmed point. The listed items are three:
- **Default** enables the default mode (assigned in configuring Tpaedi32)
- **Off** disables the compensation mode
- **Apply** enables the compensation mode;

If the item is enabled the correct profile starts from the setup programmed point to the starting compensation point on the first segment with linear movement.
In any case, Start compensation from setup is not applied if one of the following conditions is verified:

- Step-by-step compensation startup is required and applied
- the first segment of the profile needs a disconnection in compensation (see later on).

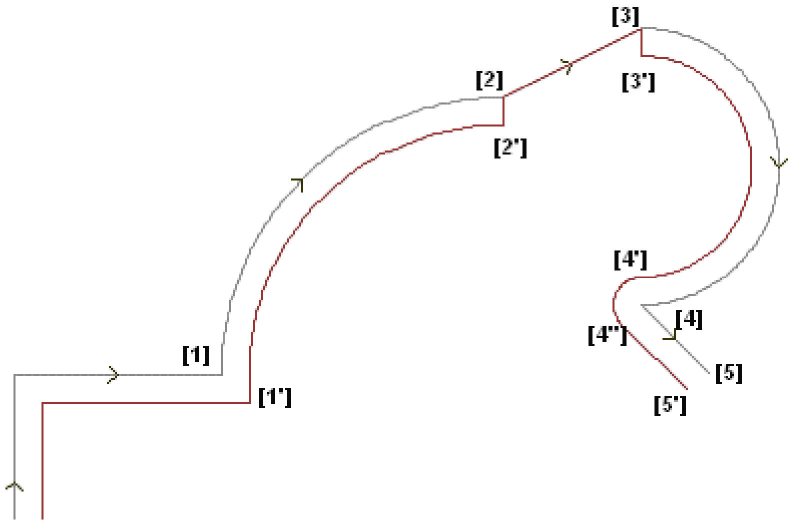
The modes of application of the tool radius compensation can be changed also during the profile development. In fact, in a work on the profile it is possible to assign the **Compensation** parameter, by selecting one of the four listed items:


L01 - L01: Pf

NAME	
Description	
□ Comment	
⊟ Geometry	
□ Relative [EG]	
Starting X (Ps) [XI]	
Starting Y (Ps) [YI]	
Starting Z (Ps) [ZI]	
Final X (Pe) [X]	260
Final Y (Pe) [Y]	160
Final Z (Pe) [Z]	
⊟ Technology	
Compensation [DVAR]	Unchanged
Movement speed [F]	
⊞ Property	

- **Unchanged**: compensation carries on unchanged with respect to the previous segment
- **Resume**: it restarts compensation, if interrupted or suspended
- **Interrupt**: it stops compensation from the current segment to restart it later
- PROFESSIONAL **Suspend:**: it interrupts compensation from the current segment to resume it later. Only in **Professional Mode** available.

Let us examine an example of application of disconnection in compensation:



The part of profile concerned is programmed on the following segments:

- ..
- [1] -> [2] (arc)
- [2] -> [3] (line)
- [3] -> [4] (arc)
- ..

Compensation is on the right side of the profile.
Let us examine the profile with tool radius compensation applied:

- ([1] -> [2]) is correct on the arc: [1'] -> [2']
- linear segment added: [2'] -> [2]
- original segment length: [2] -> [3]
- linear segment added: [3] -> [3']
- ([3] -> [4]) is correct on the arc: ([3'] -> [4']) and the fillet ([4'] -> [4'']) is added before compensating the following segment.

Compensation has not been applied on the linear segment: [2] -> [3]. In particular:

- the length and direction of the original segment [2] -> [3] coincide perfectly with those of the correct segment;
- compensation is stopped at the end of the previous segment (arc: [1] -> [2]), by defining point [2'] as it was the last segment of the profile and adding a linear segment from [2'] to point [2];
- compensation is resumed from the starting point of the following segment (arc: [3] -> [4]), by defining point [3'] as it was the first segment of the profile and adding a linear segment from [3] to point [3'].

Compensation as shown is obtained by using the **Compensation** parameter, assigned to the segments of profile.
From the example above we can infer that the correct profile results from the following settings:
[setup]: enter the compensation side: *Right*

- …
- [1] -> [2] : **Compensation**: *Unchanged*
- [2] -> [3] : **Compensation**: *Interrupt*
- [3] -> [4] : **Compensation**: *Resume*
- …

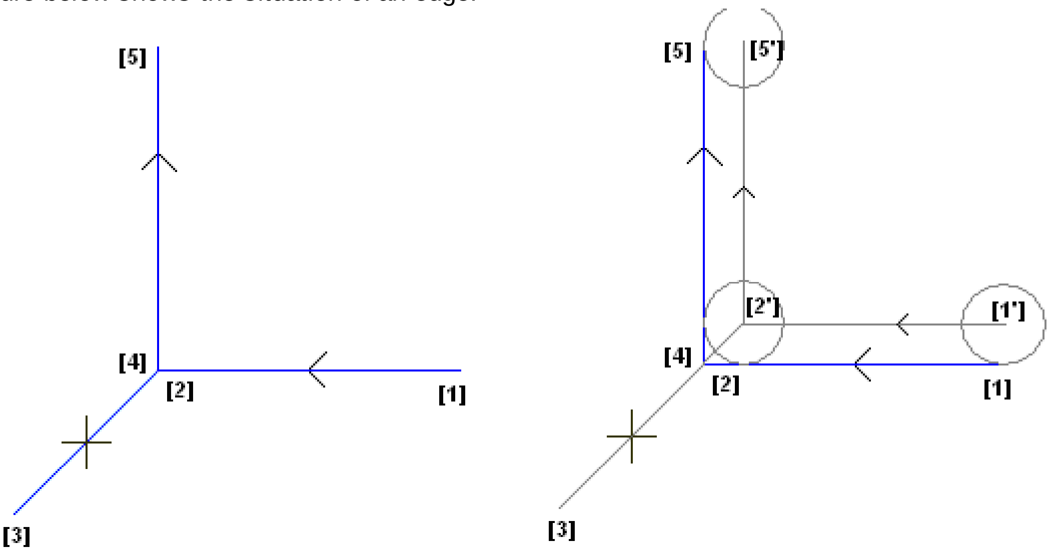
<u>It is possible:</u>

- to set a compensation stop also on the first segment of the profile;

- a compensation stop not necessarily shall be cancelled by a restart operation: it may carry on to the end of the profile.

The above-described example gives us the opportunity to focus on an aspect concerning the compensation mode applied to edges. The profile setup has necessarily required the application of the compensation mode with research of edges (Contouring: *Edges*): the intersection solution in point [1'] seems to highlight it.
But we have seen that compensation has inserted a fillet (arc: [4'] -> [4]) in point [4]: this because the compensation of the two segments which converge to point [4] has found no edge and, therefore, it has added a fillet.

An example of application of suspension in compensation concerns the working of door-frame corners.
The Figure below shows the situation of an edge:



On the left-hand side the programmed profile is displayed, with the following direction of segments:

- [1] -> [2]
- [2] -> [3]
- [3] -> [4]
- [4] -> [5]

The edge is on the two intermediate segments (2 -> 3), (3 -> 4). *Attention*: points [2] and [4] coincide.
Compensation is on the right side of the profile.

The right Figure shows what is necessary to obtain, with tool radius compensation applied:

- first correct segment: [1'] -> [2']
- linear segment added: [2'] -> [3]
- linear segment added: [3] -> [2']
- last correct segment: [2'] -> [5'].

Point [2'] is determined by intersecting the two correct segments obtained in compensation from the two original segments, respectively before and after the edge: (1 -> 2) and (4 -> 5).

Compensation as shown is obtained by using the **Compensation** parameter, assigned to the segments of profile.
From the above-described example of the frame, to obtain the correct profile as shown in the right Figure it is necessary to set each segment as follows:

- [setup]: enter the compensation side: *Right*
- …

- [1] -> [2] : **Compensation**: *Unchanged*
- [2] -> [3] : **Compensation:** *Suspend*
- [3] -> [4] : **Compensation:** *Suspend*
- [4] -> [5] : **Compensation**: *Resume*
- …

It is necessary that:

- a request for suspension is not cancelled by a restart operation. A suspension to the end of the profile determines an error message in application of tool radius compensation
- the two segments before and after suspension are geometrically consecutive, that is the first ends in the point where the second starts. Otherwise, a message error appears in application of tool radius compensation
- compensation on the two segments (segments before and after suspension) can determine a condition of intersection (and not of fillet). Otherwise, a message error appears in application of tool radius compensation.

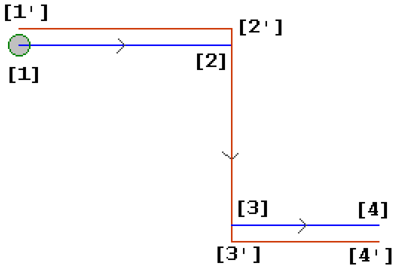
PROFESSIONAL **Variation of side to compensate** (Only in **Professional Mode** available.)

In a profile working it is possible to select the parameter **Change the compensation side,** that inverts the compensation side (from the left to the right or viceversa).

The activation of this selection is subject to limitations, as follows:

- The request may correspond to a resumption of correction after a interruption; or
- the previous segments, corresponding to the request, may assign an intersection of the compensated segments; or
- the previous segments, corresponding to the request, assign an inverted geometry.

Example of application of *inversion* to compensate



- :[1] (setup) requires left compensation
- [1] -> [2] (segment): to compensate
- [2] -> [3] (segment): stops the compensation
- [3] -> [4] (segment): restarts the compensation and requires change of side.
- ..

The same profile with tool correction applied:

- ([1] -> [2]) is correct on the left on: [1'] -> [2']
- linear segment added: [2'] -> [2]
- original segment: [2] -> [3]
- linear segment added: [3] -> [3']
- ([3] -> [4]) is correct on: ([3'] -> [4']) on the right side.

Display Modes

Commands which can be selected from ***Show->Profile thickness in compensation*** and ***Show->Original Profiles in compensation*** menu and from the View Settings Bar and can modify the display mode of tool radius compensation

Profile Thickness In Compensation

If enabled, correct profiles and the profiles which do not apply any compensation are represented with thickness equal to the tool diameter. For these profiles edge points and direction arrows are not displayed.

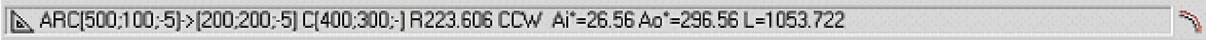
In any case the following items are represented with unit thickness:

· construct profiles
· segments of profile built in air.

If disabled, correct profiles are represented with unit thickness.

Original Profiles In Compensation
If enabled, the view shows also original profiles (incorrect profiles).
If disabled, the view shows only correct profiles and the profiles which do not apply any compensation (with direction arrows applied on the displayed segments, if required).

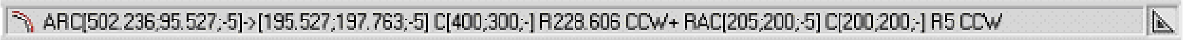
Status Bar
With Tool radius compensation activated, the coordinates relative to programmed segments or to those which have been compensated are displayed on the Status bar:
The Figure shows the programmed arc parameters:



ARC[500;100;-5]->[200;200;-5] C[400;300;-] R223.606 CCW Ai°=26.56 Ao°=296.56 L=1053.722

ARC	the writing indicates that it is about an arc (the writing is managed by message file)
[500;…]	position of the arc starting point
->[200;200;…]	position of the arc end point
C[400;…]	position of the arc center point
R=223.606	arc radius
CCW	counterclockwise rotation (CW if clockwise)
Ai°=26.56	starting angle of the segment (in degrees)
Ao°=296.56	end angle of the segment (in degrees)
L=1053.722	length of the segment (in 3d)
L°=270	angle of the arc (in degrees)

The Figure shows the correct arc parameters: to switch from one set of data to the other click on the bitmap on the right-hand of the Status bar:



ARC[502.236;95.527;-5]->[195.527;197.763;-5] C[400;300;-] R228.606 CCW+ RAC[205;200;-5] C[200;200;-] R5 CCW

ARC	the writing indicates that it is about an arc (the writing is managed by message file)
[502.236;…]	position of the starting point of the correct arc
->[195.527;…]	position of the end point of the correct arc
C[400;…]	position of the arc center point
R=228.606	radius of the correct arc
CCW	counterclockwise rotation
+	it means that compensation has added a segment after the arc
RAC	the writing indicates that it is about a fillet (the writing is managed by message file)
[205;…]	position of the fillet end point
C[200;…]	position of the fillet center point
R5	fillet radius
CCW	counterclockwise rotation of the fillet

For the correct segment neither start and end angles nor length are provided.

8.2.9 Joining Profiles

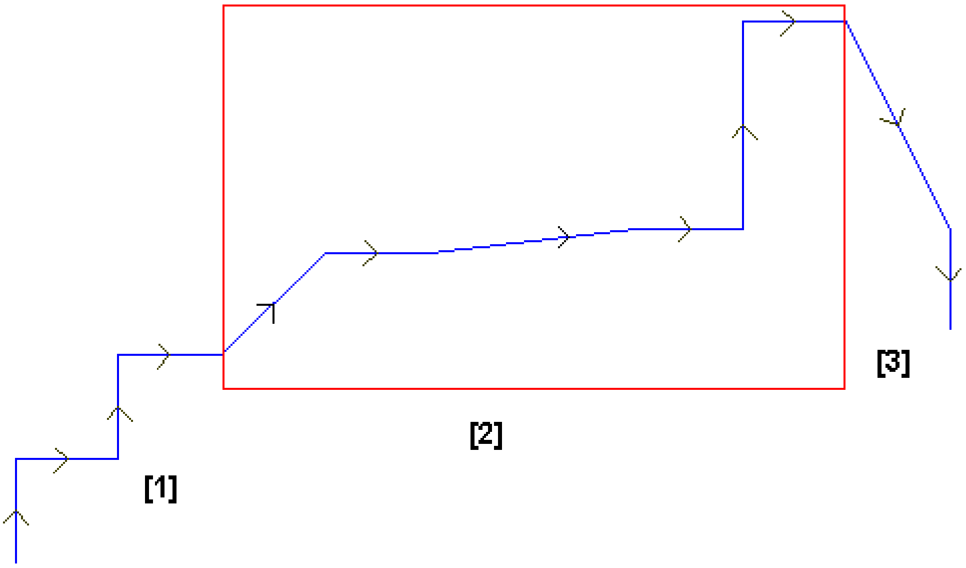
A particular aspect in defining profiles is the possibility of linking them to each other. It is about an option available as parameter of setup workings and complex codes. The corresponding item is: **Point hook**.
A point hook always needs the implementation of a relative programming of null displacements. If a profile element which can be hooked (setup, arc, line, subroutine which ends its development with a profile element) is assigned before the hook point and if the active working is a setup working or a complex code, the profile before the hook point continues the profile after the hook point, without executing any of the intermediate setups. In this case we are talking about **profile connection.**

A profile resulting from the connection of different segments is a single profile to all intents and purposes. The profile technology is usually assigned by the opening setup working. If no opening setup is assigned, also in this case we are talking about open profile.

8.2.10 Simple Profiles

Thanks to the point hook procedure, it is possible to continue a profile with parts assigned by the application of subroutines or macros. In any case, it is not said that this mechanism always allows to obtain a profile, where the tool selected for the profile building remains engaged from the starting point to the end point of the profile, without any disconnection.

Let us see a **first example**:

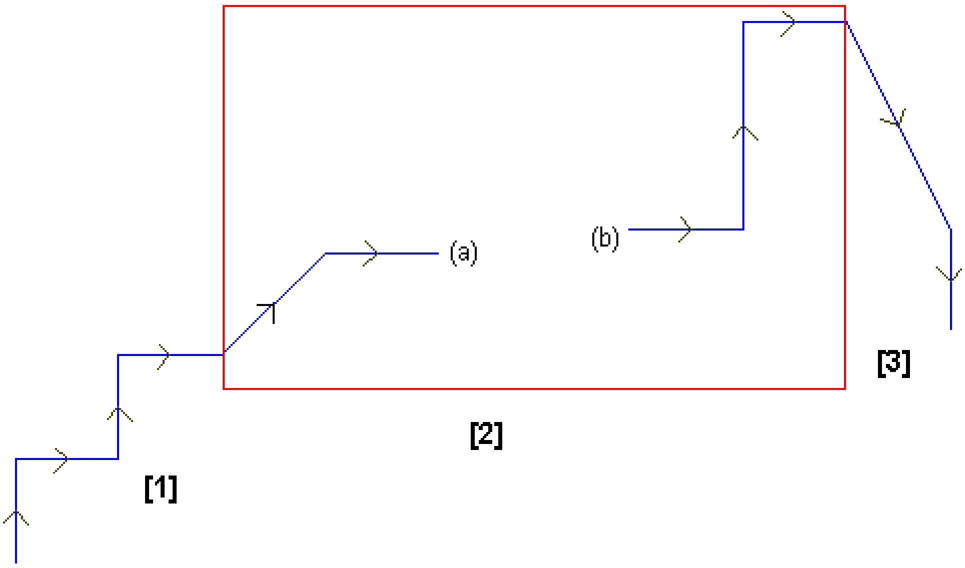


the profile shown in the above Figure consists of 3 parts:

- **[1]** the first part at the beginning of the figure (on the left-hand side) is obtained with linear segments (it does not matter if the profile is open or not)
- **[2]** the central part is enclosed in a rectangle: let us assume that it has been obtained by the application of a subroutine (in point hook)
- **[3]** the third part at the end of the figure (on the right-hand side) is obtained with linear segments and terminates the profile.

It can be stated that a profile has been built: the working tool remains engaged from the starting point to the end point of the profile, without any disconnection.

Now let us see a **second example**:



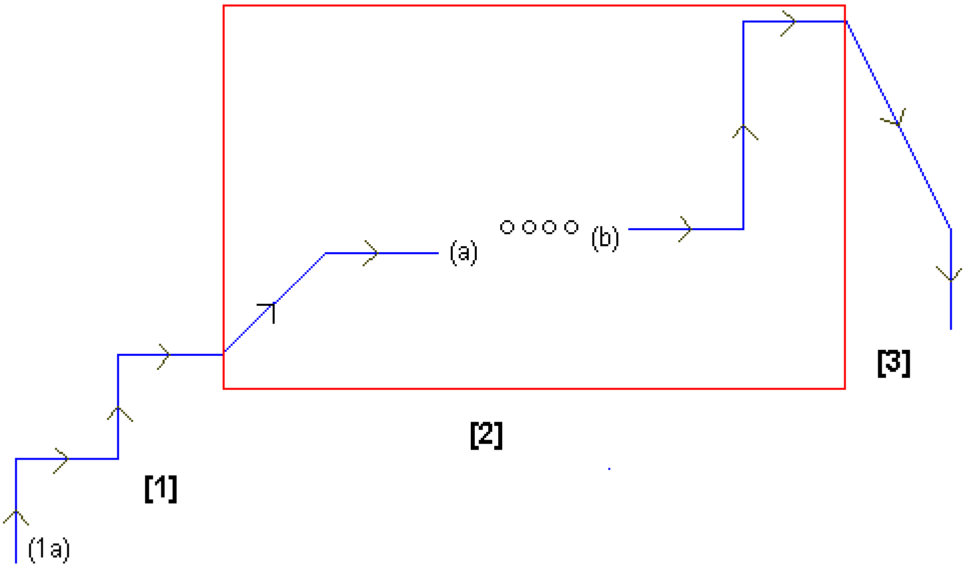
the graphical representation is similar to the previous one: the difference is that the central part of the profile shows a disconnection.

Is it still possible to say that a profile has been built?

in fact the execution shows two separate profiles:

- the first profile executes the first part **[1]** and continues to point **(a)** shown in figure
- the second profile starts from point **(b)** shown in figure and continues to complete the final part **[3]**.

A **third example** is even more far from the idea of profile:



now the subroutine marked with **[2]** executes:

- in the initial part: a profile (which is connected to the previous profile **[1]**)
- in the central part: four drilling workings
- in the final part: a profile (which is connected to the next profile **[3]**)

Logically speaking, the profile definition should be applied only to the first of the three examples examined above.

In any case, there are some functions specific to a profile for which it is not at all important to make a distinction between the above-described examples. If for example it is necessary to apply a profile tool which assigns a particular technology to the profile which starts in (**1a**), it can be useful that the tool itself considers the set of workings as a profile, without taking into account how the **[2]** block has been

defined: in this case we are talking about *profile defined in any case* or *complex* or *extended*.
In the first of the three examples examined above the profile is defined as *simple*: to all intents and purposes, the [2] block can be assimilated to a profile element.
Therefore, a profile is said to be *simple* if it consists of simple profile elements (linear segments or arcs) and/or complex codes (subroutines or macros) which can be assimilated to simple profile elements.

8.2.11 Multiple Setups

The profiles to which more than one setup is assigned are defined multiple setups or profiles. During profile building, the profile execution is repeated for a number of times equal to the number of setups configured. The first time the profile is executed with the first setup with the assigned technology, the second time the profile is executed with the second setup and the relevant technology and so on for all the other setups. In this way it is possible to execute the same profile with different tools without having to program it every time.

In Tpaedi32 only the first setup assigned to the profile is interpreted, while for the following setups the Point hook procedure is applied which makes them fully invisible, during the execution of the profile itself. If for example the application of tool radius compensation is required, the profile is compensated on the basis of the technology assigned to the first setup. See ***Tools-> Apply multiple setups***

8.3 Edit

8.3.1 Selection

The selection of workings is enabled only in face view and with face program not empty. It is not possible to make partial selections of a complex working or a subroutine. When a face view is closed, each selection is cleared.

How workings in graphical view are selected
The following key combinations are managed according to the priority order assigned as indicated by the sequence of points:

[SHIFT + (left mouse key pressed)]: it starts the area selection.
Workings which are enclosed in the relevant window are selected. In particular:

- the only workings shown in the graphical view are taken into account (it applies active views and view filters)
- workings conditioned by logical statements or commented are excluded from research
- if when the mouse key is released the **[ALT]** key appears depressed it means that selection is extended to include the profiles which are only partially enclosed in the window
- previous selections are kept
- the active line does not change

[CTRL + ALT + (left mouse key pressed)]: it selects or deselects the entire profile to which the working which is nearest to the mouse pointer position belongs (this command is only effective if the working belongs to a profile):

- the only workings shown in the graphical view are taken into account (it applies active views and view filters)
- previous selections are kept
- the active line does not change

[CTRL + (left mouse key pressed)]: it selects or deselects the working which is nearest to the mouse pointer position.

- the only workings shown in the graphical view are taken into account (it applies active views and view filters)
- previous selections are kept

- the active line does not change

In the graphic representation the selected workings are coloured according to the settings defined during the customization of
Tpaedi32.

How workings in ASCII text are selected
The following key combinations are managed according to the priority order assigned as indicated by the sequence of points:

[SHIFT + (left mouse key pressed)]: it selects from the active line to the program line pointed by the mouse cursor
- previous selections are lost
- the active line does not change

[CTRL + ALT + (left mouse key pressed)]: it selects or deselects the entire profile to which the program line pointed by the mouse cursor belongs (this command is only effective if the working belongs to a profile):
- previous selections are kept
- the active line does not change

[CTRL + (left mouse key pressed)]: it selects or deselects the program line pointed by the mouse cursor
- previous selections are kept
- the active line does not change

In the ASCII text the background of the selected workings is blue with white writing.

8.3.2 Editing the Face Program

The modification of workings is usually applied to a selected set of workings.
A selected group can be composed of one or more workings and is defined with addition and/or removal of characteristic working states.
An example of selected set of workings is represented by the application of special views and/or view filters. See Chapter ***Advanced Assignments->Special Filters***
As for general edit commands, it is necessary to create the selected set of workings before choosing the edit command.
Examples of selected set of workings are the following:
- the selected workings which verify logical conditions
- the selected workings which are assigned on a given level

Many edit commands consider as selected privileged set of workings the group which is composed of selected workings. In this case if no selection is active, the active working is edited.

8.3.3 Editing the Current Working

The direct modification of a working consists in changing parameter settings and/or working properties.
The direct modification is always applied to the active working and cannot be applied to workings in locked status (it is about an induced call or it has Level, Construct or "O" field locked) or with invalid operating code.
It is possible to modify the ASCII text directly or in the working assignment window.

Editing the active working in assignment window
If the window is displayed in direct editing mode it is possible to modify directly parameter settings and confirm by clicking the **[Confirm]** button.
If, on the contrary, the window is shown in only display mode to edit working parameters the operator

shall select the **[Edit]** button.
The active working is edited only if no error messages concerning the working itself are signalled. If this were the case, it would be necessary to solve problems or cancel modifications.
Once the modification has been confirmed, the working of the program list which follows that which has been edited becomes the active working.

Editing the active working in ASCII text
Editing is enabled, by double-clicking the relevant field, for all fields with active edit. Data editing for the **ASCII Format** field is enabled at software level.

8.3.4 Editing Properties

Property editing is enabled only in face view and with face program not empty. It is applied to selected workings, if any, otherwise to the current working and it only concerns workings which verify active view filters: selections, logical conditions, levels, special filters. It cannot be applied to workings in locked status (it is about an induced call or it has Level, Construct or "O" field locked).

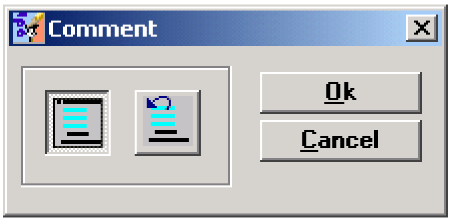
In the case of piece-face there are two situations to be taken into account:

- with the 3d face view active, the complete list of workings is subject to editing
- with the 2d face view active, the only workings applied to the face in current view are subject to editing.

"O", "M", "B", "L" and "K" property fields may not be modifiable (L, B, K always and possibly M and O, if they are not configured to be directly edited in profile) in the following cases:

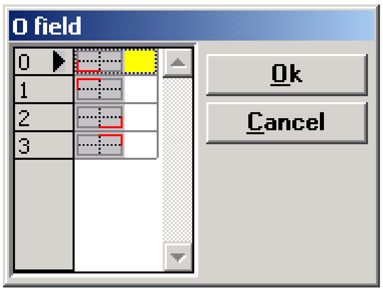
- works on the profile (arcs and lines): if the working opens a profile (open profile) the value is still 0, otherwise they take the value from the profile start working
- in case of setup or complex working, with request for point hook: the properties which cannot be edited in profile are propagated by the profile start working

"C" or Comment Property
it is an optional property.



It configures the "C" (Comment) property for selected workings. If enabled, the working remains in the list but it does not affect the program. When it is referred to the previous or next working, with respect to another, it has to be meant commented workings excluded. Clicking on the first icon, the selected working becomes a commented working; clicking on the second icon, the selected working is deleted from the list of commented workings. If a working has the "C" property flag active it can be edited only after disabling the comment flag. The "C" property flag can be activated also when the working has an invalid operating code
This property can be set from ***Apply->Assign property->Comment*** menu.

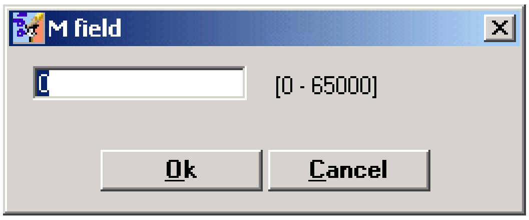
"O" Property
it is an optional property.



The meaning and the assignment mode of the "O" field depend on how Tpaedi32 is configured in Customize->Editor. In the Figure above the "O" field is used to assign a side or edge as working programming reference. This property can be set from ***Apply->Assign property->O Field*** menu.

"M" Property

it is an optional property.

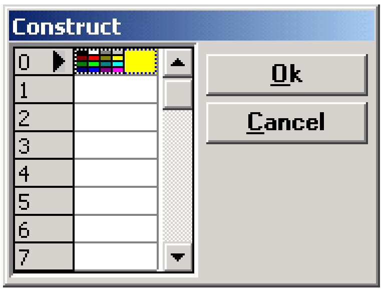


0 [0 - 65000]

It is possible to assign the "M" property value only by direct edit. This property can be set from ***Apply->Assign property->M Field*** menu.

"B" or Construct Property

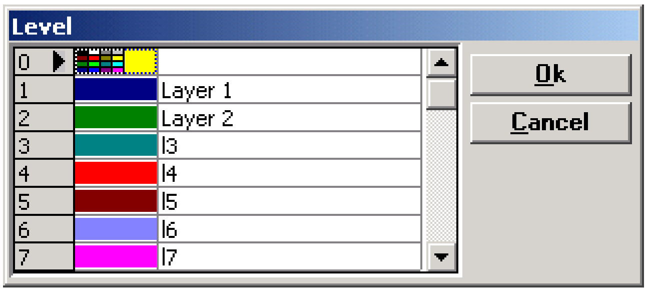
it is an optional property.



The meaning and the assignment mode of the "B" Property depend on how Tpaedi32 is configured in Customize->Editor. The Figure corresponds to the case of maximum value greater than 7 managed for the property (the vertical scroll bar is active). The property value shall be set in the list, with the only indication of the colors assigned for each value of the Construct property. If the working is indicated as construct it is compiled but not executed. This property can be set from ***Apply->Assign property->Construct*** menu.

"L" or Level Property

it is an optional property.



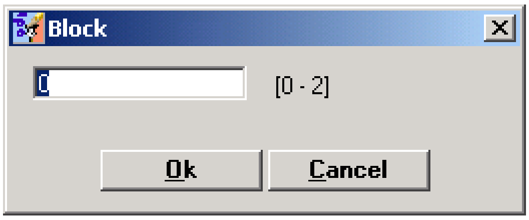
0	
1	Layer 1
2	Layer 2
3	l3
4	l4
5	l5
6	l6
7	l7

The meaning and the assignment mode of the "L" property depend on how Tpaedi32 is configured in

Customize->Editor. The Figure corresponds to the case of maximum value greater than 7 managed for the property (the vertical scroll bar is active). The property value shall be set in the list, with indication of the colors and names assigned for each value of the Level field. This property can be set from ***Apply->Assign property->Level*** menu.

"K" or Block Property

it is an optional property.



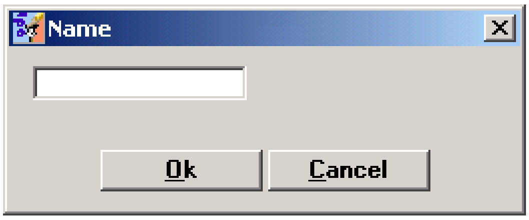
The meaning and the assignment mode of the "K" property depend on how Tpaedi32 is configured in Customize->Editor. For the "K" property two different types of numbering are active:

- automatic numbering: a maximum very great value (2*109) is automatically managed but only in direct edit
- custom numbering: a maximum value (up to 255) can be assigned by direct edit or selecting it from the list.

The edit field is set to the value corresponding to the first value which has not been assigned for the program. This property can be set from ***Apply->Assign property->Block*** menu.

"N" or Name Property

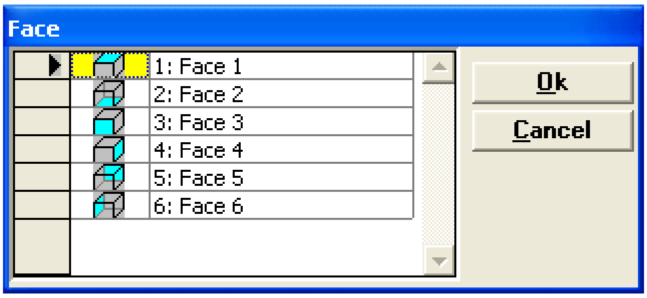
it is an optional property.



A string no more than 16 alphanumeric characters long is assigned to the "N" property. It is used for the application of complex codes of transforms to apply directly to programmed workings. See Chapter ***Tools->Advanced Tools in Face Program->Workings which apply geometric transforms***. This property can be set from ***Apply->Assign property->Name*** menu.

Face Property

Only in Face Piece available.



1: Face 1
2: Face 2
3: Face 3
4: Face 4
5: Face 5
6: Face 6

Setup is always possible, when managed from the list.

8.4 Insertion

8.4.1 Selecting the Entry Point

In face view the active working is highlighted both in ASCII text area and in face graphical view. The working assignment area shows the parameter settings of the active working.

Scrolling and selecting the active working in graphical view
The face graphical view is made interactive by mouse clicking on the display frame.
The following situations are managed:

- **direct pointing to a working (click in area)**: it moves the active working to that which is nearest to the position pointed by the mouse. In particular the research is carried out on the complete list of programs from the first to the last block, induced blocks included, but only for the workings which are displayed at the moment. Logical (IF, ELSE, ENDIF) or commented workings are excluded from the research. All face selections are cleared.
- **scrolling the program blocks**. The following keys are available:
  - **<UP arrow>**: it moves the active line to the previous block in the text list,
  - **<DOWN arrow>**: it moves the active line to the next block in the text list,
  - **<Previous page>**: it moves the active line one page up. The page size is defined in terms of number of lines displayed in the ASCII text area.
  - **<Next page>**: it moves the active line one page down,
  - **<Home>** : it moves the active line to the first program block,
  - **<End>**: it moves the active line to the last program block.

Whenever the active working changes all face selections are cleared.

Scrolling and selecting the active working in ASCII text
It is possible to scroll the program directly on the ASCII text.
The following situations are managed:

- direct pointing to a working (click in area): it moves the active line to the line pointed by the mouse. All face selections are cleared.
- scrolling the program blocks. The available keys are the same as those used to scroll the program in graphical representation. Whenever the active working changes all face selections are cleared;

Scrolling and selecting the active working with menu commands
It is possible to select a working from ***Edit->Go To*** menu.
Here is the list of available commands:

	It is recalled from menu: ***Edit->Go To->Next correspondence.*** It moves the active working selection to that which is nearest to the position pointed by the mouse. The research is carried out on the program from the current working to the last block, induced blocks included, but only for the workings which are displayed at the moment. Logical (IF, ELSE, ENDIF) or commented workings are excluded from the research. All face selections are cleared. In case of profile segments or overlapped drilling, this kind of selection allows to scroll all the workings assigned in the same position. If the search fails, the selection of ***Next correspondence*** is reset.
	It is recalled from menu: ***Edit->Go To->Go to first working***. It moves the active working to the first working in the list.
	It is recalled from menu: ***Edit->Go To->Go to last working***. It moves the active working to the last working in the list.
	It is recalled from menu: ***Edit->Go To->Go to previous working***. It moves the active working to the previous working: It works if the active working is not already the first in the list.
	It is recalled from menu: ***Edit->Go To->Go to the next working***. It moves the active working to the next working: It works if the active working is not already the last in the list.
	It is recalled from menu: ***Edit->Go To->Go to the profile start working***. It moves the active working to the profile start working: it works if the current working belongs to a profile.

	It is recalled from menu: ***Edit->Go To->Go to the profile end working***. It moves the active working to the profile end working: it works if the current working belongs to a profile.
	It is recalled from menu: ***Edit->Go To->Go to line...*** It moves the active working to the working of the progressive assigned.

Insertion of workings before or after the active working
By selecting the ***Insert before*** or ***Insert after*** items from the ***Apply*** menu it is possible to set the position where workings shall be entered, with respect to the active working. If, for example, ***Insert before*** has been selected all new workings are entered before the active working.

8.4.2 Inserting a Working at the mouse Click Position

Inserting a Working at the mouse Click Position
When a working is entered, during the bitmap selection, by holding down the **[SHIFT]** key, the setting window is not displayed immediately. The mouse pointer is customized and the following message **Indicate the position (ESCAPE Cancel)** appears in the commands bar area

- click in Face View to assign the entry point for the working.
- It is possible to enter the selected working until the SHIFT key is held down and cancel the entry operation by pressing the ESCAPE key.

If the snap-to-grid is active the entry point is determined by the coordinates of the grid vertex nearest to the click point, otherwise the entry point is the click point.
The interactive entry mode can be activated for complex and setup workings as well as point workings, if they allow to set the entry point.

8.4.3 General Purpose Commands

Copy, Paste, Cut, Delete, Delete All, Cancel, Select All
It is possible to apply the ***Copy, Paste, Cut, Delete, Delete All, Select All***commands to workings. These commands are enabled only in Face View and with the active program not empty except for the Cancel command.
In the case of piece-face two are the possible situations to be taken into account:

- with the 3d face view active: the complete list of workings can be deleted
- with the 2d face view active: the only workings applied to the face in current view can be deleted

They operate on selected workings, if any, otherwise on the active working. They are only used for the workings which verify active view filters: selections, logical conditions, levels, special filters.
They cannot be applied to workings in locked status (it is about an induced call or it has Level, Construct or “O” field locked).
In the case of macro-program the profile is evaluated by including the logical workings which can disconnect the profile itself.
These commands can be selected from ***Edit*** menu.

Edit->Cut: it cuts the selected workings and copies them to the program's Clipboard. In case only one working belonging to a profile is to be cut it is possible to remove the whole profile (confirmation is required).
Let us see the above commands in detail:

	Edit->Copy: it copies the selected workings to the program's Clipboard. In case only one working belonging to a profile is to be copied, it is possible to copy the whole profile (confirmation is required).
	Edit->Paste: it pastes the Clipboard content at the insertion point (before or after the active working). In case a working is directly entered in the middle of a profile you are asked if you wish to move the entry point before or after the profile itself.

✂	***Edit->Cut***: it cuts the selected workings and copies them to the program's Clipboard. In case only one working belonging to a profile is to be cut it is possible to remove the whole profile (confirmation is required).
✕	***Edit->Delete***: it deletes the selected workings without copying them to the program's Clipboard. In case only one working belonging to a profile is to be deleted it is possible to remove the whole profile (confirmation is required).
↶	***Edit->Cancel***: this command is to be used to reverse the last editing action of the face program. Once the command has been executed, information about the cancelled action is provided in the Commands bar area. The list of actions (commands) which can be cancelled is cleared when the active program is closed
	Edit->Select All: it selects all face workings
	Edit->Delete All: it cancels all workings without copying them to the program's Clipboard.

8.5 Creating Complex Workings

A complex working is a set of workings, which in their turn can be simple or complex.
The use of complex workings is clear:

- the whole complexity of the working is hidden to the user
- the working selection and assignment is not different from simple workings.

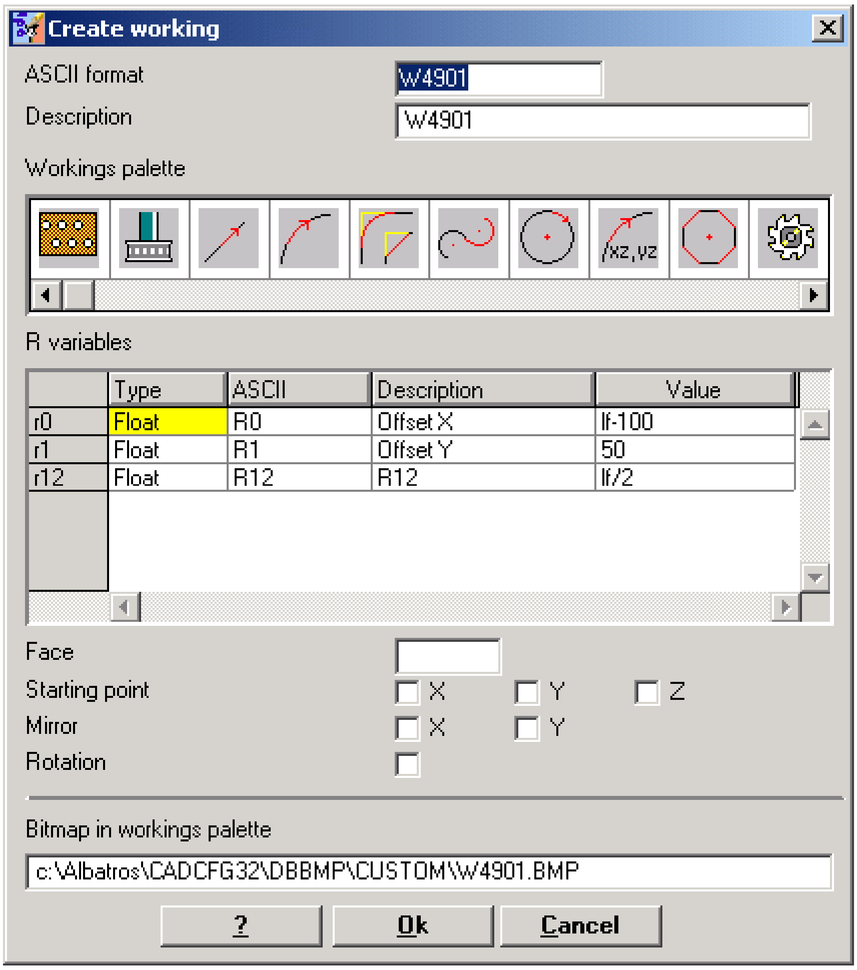
A complex working is based on a subroutine or a macro-program. This means that a program has been written with subroutine or macro-program typology and contains the workings which have been programmed to be executed by the complex working.
When a complex working is entered, the subroutine (or macro-program) is opened and applied with rules very similar to those seen in case of application of a general subroutine code. It can even be stated that the result is the same.
In any case, the differences between the two methods are remarkable from the point of view of practical applications. Let us examine them in detail:

- a working complex has its own unique identification: bitmap in workings palette, description, operating code, help file
- during assignment it has a customized parameter presentation: node splitting, field heading and the only fields which are really important. In particular:
  - the indication of the subroutine (or macro-program) the working refers to is completely hidden
  - the "r" variable settings of the subroutine are displayed together with other data (positioning coordinates, geometric transforms,..) and not in a separate window.

A macro-program can only be managed from the Constructor level, but a subroutine can be defined at each access level. The control end user can write and recall subroutines.
For this reason the possibility to assign complex codes also by the end user has been implemented.
The complex codes thus assigned will belong to the workings managed by the editor, in the same way as the original program workings.
For this purpose two commands are available. They can be selected from the ***File*** menu with the program closed:

- ***File->Manage Workings->Create Workings***
- ***File->Manage Workings->Delete Workings***

The first command opens the search window of the subroutine (the research is set in the archive subroutine directory).
Once the command has been selected and the selection confirmed, the window displaying the assignments available for the complex code is opened:



ASCII format: W4901

Description: W4901

Workings palette

R variables

	Type	ASCII	Description	Value
r0	Float	R0	Offset X	lf-100
r1	Float	R1	Offset Y	50
r12	Float	R12	R12	lf/2

Face:

Starting point: □ X □ Y □ Z

Mirror: □ X □ Y

Rotation: □

Bitmap in workings palette

c:\Albatros\CADCFG32\DBBMP\CUSTOM\W4901.BMP

- **ASCII Format**: ASCII name of the working. The default name is displayed in the form W + (opc), where "opc" stands for operating code. This last is automatically assigned to the working (the first free code among those available for this set of workings). It is possible to define a set of alphanumeric characters ranging between 2 and 10. The first character shall be a letter. They are considered as reserved names, then not usable: "WC", "WB", "WL", "WO", "WM", "WK", "WS", "WP", "WF", "WN", “SIDE”, “EMX”, “EMY”. The choosen name can not already be defined for another working. The name ASCII of the working heads a programm row in the ASCII Format list. The field setting is obligatory.

  Description: descriptive name for the working (example: "Special Drilling"). The field is initialized with the name of the subroutine (e.g. W4901), it has a length of max. 30 carachters and it is not inserted in the file language, thus it cannot be translated.
- **Workings Palette**: it provides the workings palette buttons. The image matching the set of workings to which the complex working is to be added is selected
- **R variables**: it lists the subroutine variables which can be reassigned, that become reassignable parameters of the subroutine. For each variable it is possible to assign an ASCII name, a descriptive name, a default value.
- **Type:** typology of the variable. The managed typologies are three: number with decimals (double), number without decimals (integer), string, It is a not editable field.
- **ASCII Name:** Variable ASCII. From 1 to 10 alfanumeric characters can be set. The first character must be a letter. Following names are considered as reserved (therefore not usable): "WC", "WB", "WL", "WO", "WM", "WK", "WS", "WP", "WF", "WN", “H”, “X”, “Y”, “Z”, “SIDE”, “EMX”, “EMY”, “A” next to the ASCII name of the working itself. The ASCII name is displayed the assignment area in the column ASCII format.
- **Description:** Parameter descriptive name (e.g.: "Offset x"). The field is initialized with the symbolic name of the variable or, if this last is not assigned, with the description of the variable or, if not

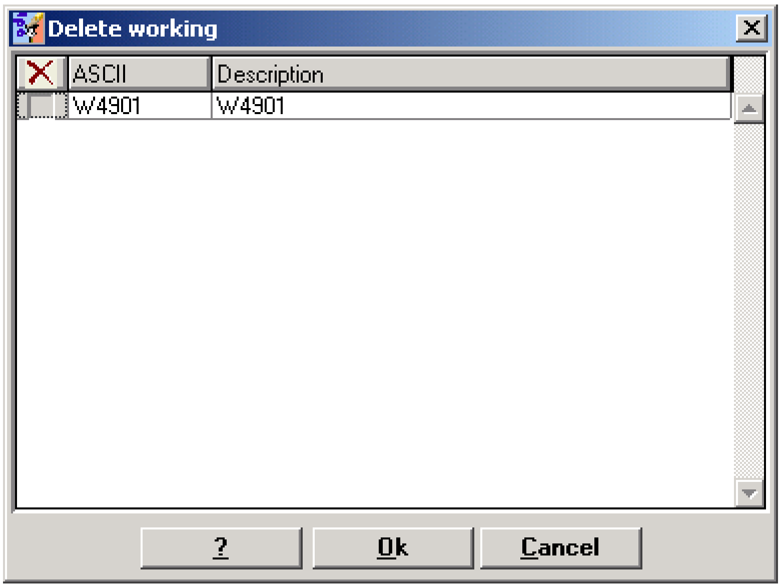
assigned, as R+(nn), with nn=number of the variable (example: "R0", "R27"). The field has a max. length of 30 characters and it is not inserted in the file of language, therefore it cannot be translated.

- **Default Value**: default value suggested during the working. The field is initialized according to the value set in the subroutine. If the variable has a numeric typology, it is possible to set a number. If a default value is not required, the filed must be let empty.
- **Face**: it defines the face (custom) number which shall be applied. If the field is assigned empty, the face selection parameter is let available in the working. The field is initialized at 1. If the field is assigned empty, the face selection parameter is let available in the working, so that a direct choice of the applied face is allowed, rather than a functioning according to the mechanisms of the induced calls. To the face parameter the ASCII name: "SIDE", that was already indicated as a reserved name, is automatically assigned.
- **Starting point**: items corresponding to the positioning coordinate/s to include in the working. To the setting parameters of the Initial Point the ASCII Names "X","Y", and "Z", already indicated as reserved names, are automatically assigned.
- **Mirror**: items corresponding to the mirror/s to include in the working. To the setting parameters of the symmetries the ASCII names: "EMX","EMY", that were already indicated as reserved names, are automatically assigned.
- **Rotation**: item to include the rotation field. The ASCII name: "A", that was already indicated as a reserved name, is automatically assigned to the rotation parameters.
- **Bitmap in workings palette**: it shows the full path of the bitmap which is displayed in the workings palette (it must be usually prepared).

Once parameter settings have been confirmed, a new complex code is assigned: now it is available in the workings palette.

If it is necessary to update the complex code assigned to a subroutine it is enough to reselect the command for the concerned subroutine. In this way, the program verifies that a complex code has already been assigned to the subroutine in question, retrieves information already entered, updates the code by adding or deleting information (r variables) and redisplays the above window.

The second command enables to delete one or more complex workings assigned by the **Create working** command



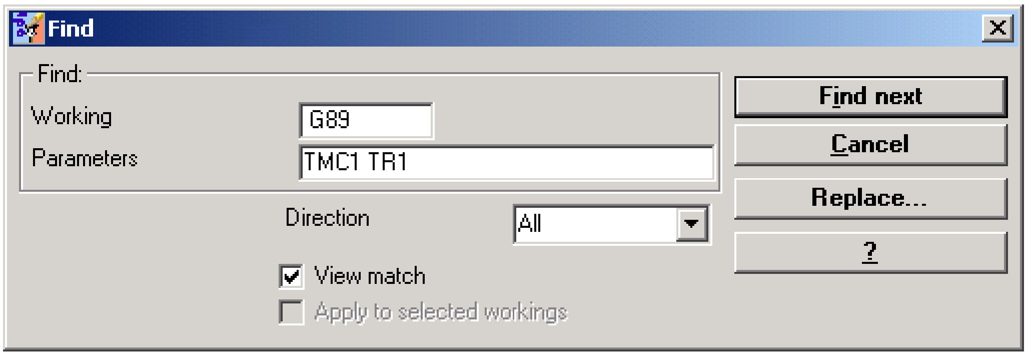
☒	ASCII	Description
□	W4901	W4901

Workings are listed in the table, with indication of two fields: **ASCII** name and **Description**.

To delete workings the first box ☒ is selected.

8.6 Find

This command is enabled only in Face View and with face program not empty. The window where to set search criteria (Find window) can be accessed from ***Edit->Find*** menu and from the Main Bar by selecting the relevant icon and appears as follows:



Find:

Working: G89

Parameters: TMC1 TR1

Direction: All

☑ View match

□ Apply to selected workings

It finds the first working which meets the assigned search criteria:

- **Working**: it is the ASCII code assigned to the working to find (in the example above: "G89")
- **Parameters**: assignments of ASCII text added (in the example above: "TMC1 TR1")
- **Direction**: search direction in the list (All workings or from the active working on)
- **View match**: it takes into account the only workings represented in current view (it applies active views and view filters)
- **Apply to selected workings**: it considers only the selected workings

In case of piece-face two are the possible situations to be taken into account:

- with the 3d face view active, the research concerns the complete list of workings
- with the 2d face view active, the research concerns the only workings applied to the face in current view

The found working becomes the active working.
The **[Find next]** button allows to continue the search session, the **[Replace]** button opens the *Replace* dialog box, the **[Cancel]** button terminates the search session and closes the window.
The Find next command which can be recalled from ***Edit->Find next*** menu and from the Main Bar by clicking on the relevant icon is available only if the face has programmed workings and after starting the research by selecting the ***Find*** command.

8.7 Replace

This command is enabled only in Face View and with face program not empty. The window where to set search and replace criteria (Find and Replace window) can be accessed from ***Edit->Replace*** menu and from the Main Bar by selecting the relevant icon .
It finds the first working which satisfies the assigned search criteria and replaces the old data with new settings.
In the **Find** area the data to be replaced are set:

- **Working**: it is the ASCII code assigned to the working to find
- **Parameters**: assignments of ASCII text added

In the **Replace with** area the new data are set

- **Working**: it is the new ASCII code assigned to the working
- **Parameters**: it is the new parameter assigned to the working

- **Direction**: search direction in the list (All workings or from the active working on)
- **View match**: it takes into account the only workings represented in current view (it applies active

views and view filters)

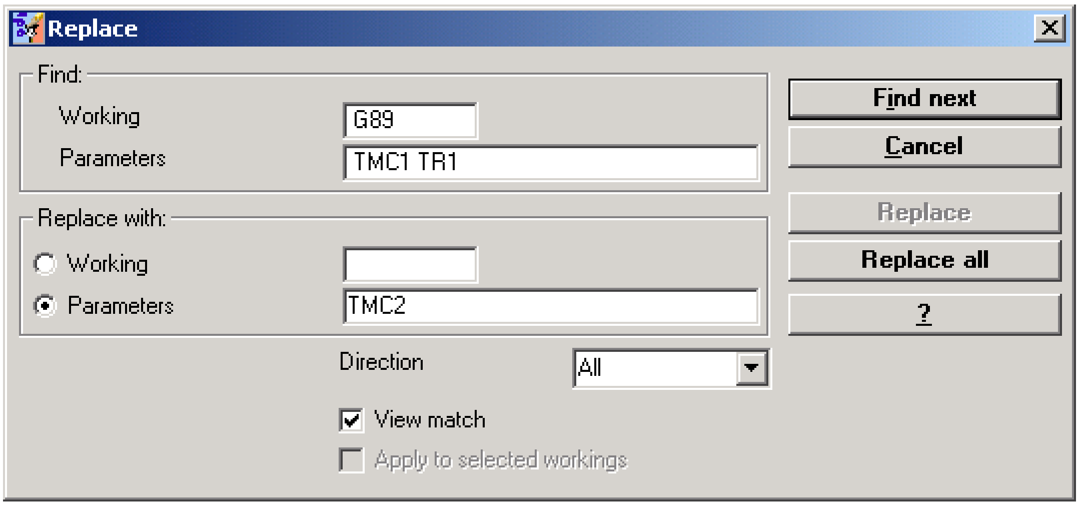
- **Apply to selected workings**: it considers only the selected workings.

The found working becomes the active working.

In case of piece-face two are the possible situations to be taken into account:

- with the 3d face view active, research and replacement concern the complete list of workings
- with the 2d face view active, research and replacement concern the only workings applied to the face in current view

The setting parameters shown in the Figure below can be replaced by specific working parameters: the "G89" workings are the only ones to be replaced with the "TMC1 TR1" assignments and, in this last case, the "TMC1" assignment is replaced with "TMC2".



Replace

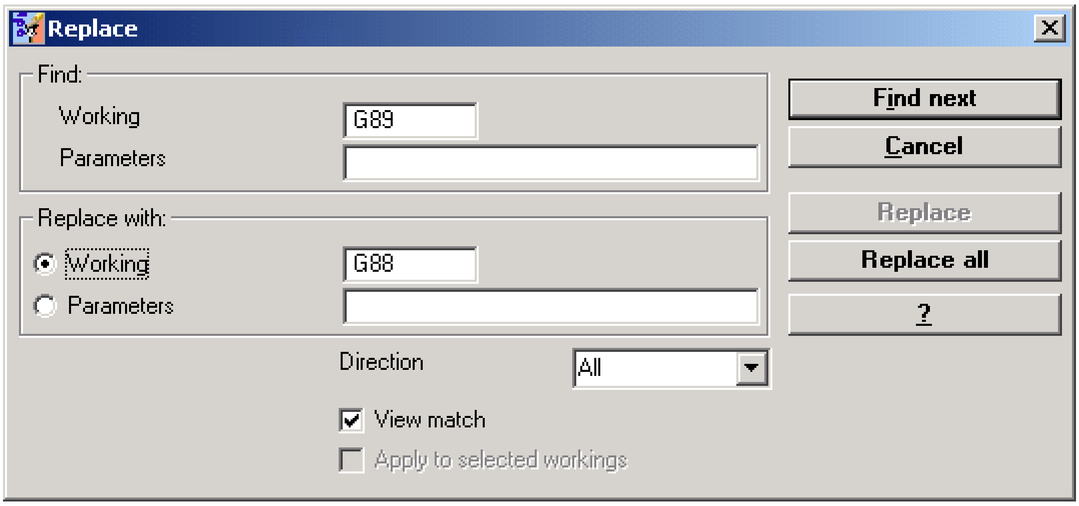
Find:
Working G89
Parameters TMC1 TR1

Replace with:
○ Working
● Parameters TMC2

Direction All

☑ View match
□ Apply to selected workings

The setting parameters shown in this second Figure need a replacement of operating code. The "G89" workings are the only ones to be replaced with "G88" workings.



Replace

Find:
Working G89
Parameters

Replace with:
● Working G88
○ Parameters

Direction All

☑ View match
□ Apply to selected workings

Let us suppose to open a program which uses invalid working codes in the current program configuration (in the example: "G89" does not match any working operating code).
A solution to make valid workings is to replace "G89" workings with a valid working (in the example above: "G88"). In any case, the search function shall deactivate the **View match** option, since "G89" workings cannot be represented in graphical view.

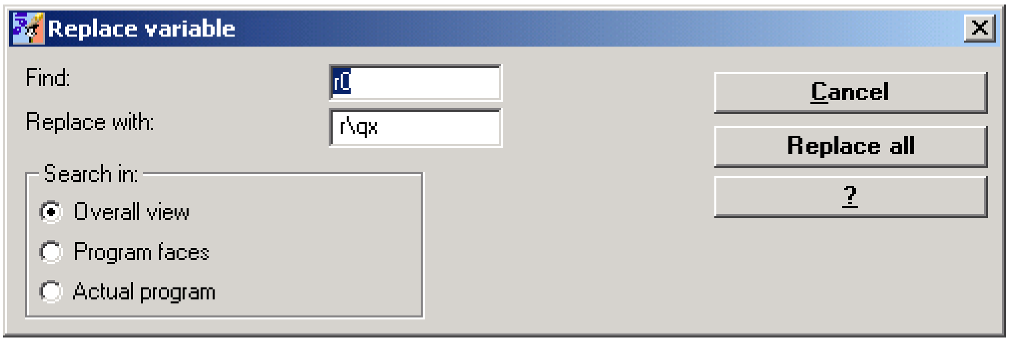
The **[Find next]** button allows to continue the search session without replacements, the **[Replace]** button replaces the data of the working found with new settings, the **[Replace all]** button replaces all face workings which meet the assigned criteria with new settings, the **[Cancel]** button terminates the replace session and closes the window.

8.8 Replace variable

The command is enabled in overall view and in face view and with not empty face program. The research and replacement data setting window is recalled by ***Edit->Replace Variable*** menu.

It finds the first working which satisfies the assigned search criteria and replaces the old data with new settings.
It is possible to assign search and\or replacement criteria for o, v, r, j, $ variables (only in macro-program test).
It is possible to replace a variable of a given type with another of different type: for example v5 with r15.

In overall view the dialog box shows the following data:



Find: r0

Replace with: r\qx

Search in:

● Overall view

○ Program faces

○ Actual program

- **Find**: variable to be replaced
- **Replace with**: value to replace
- **Search in**:
  - **Overall view:** searches and replaces variable in overall view
  - **Program faces:** searches and replaces in all program faces
  - **Actual program:** searches and replaces in overall view and in all program faces

In face view the required data are similar to those which are requested in piece overall face.

9 Piece-Face

9.1 Foreword

The **piece-face** is a particular face, which can be added to the management of real and fictive faces. This face can have multiple applications.

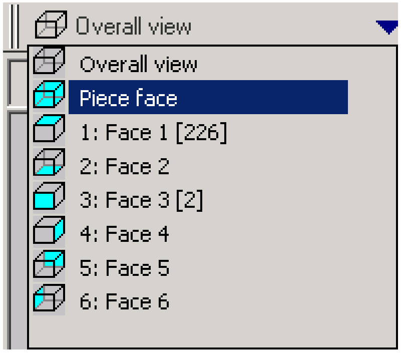
The piece-face is not geometrically identified. It can be stated that it represents the piece as a whole, including all faces which characterize it. Also in piece-face it is possible to assign a working program.
Conventionally, the piece-face is assigned with:

- the absolute reference system of the piece
- the piece dimensions
- the identification number 0.

The piece-face program allows to assign workings directly to different faces, in a single program list. The working assignment is referred to its own application face: positioning on the 3 face axes of the coordinate system, face parameterizations.

9.2 How To Open It

It is possible to select the piece-face from Face Selection Bar:

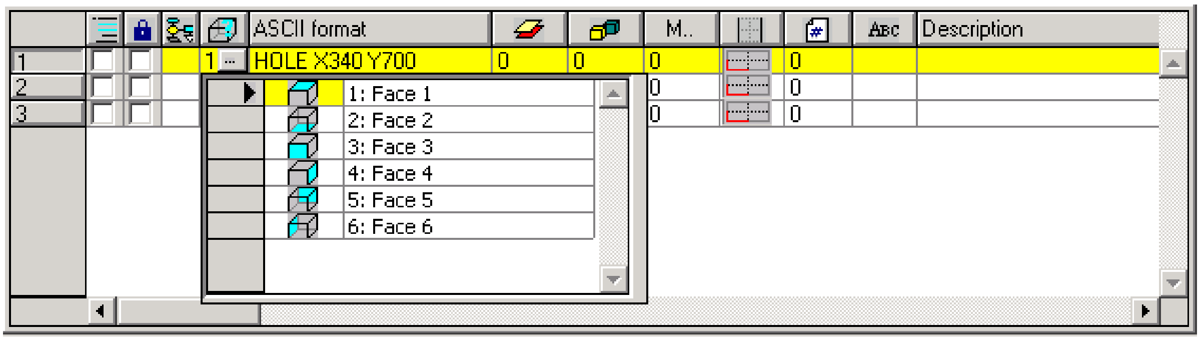


Opening the piece-face, the piece view areas are configured differently from the face view, for example, a real face. Let us start to examine the case of an empty face program:

- the overall view area does not display any active face
- the piece current view area is set as a three-dimensional (3d) graphical representation of the piece. Also in this case no active face is highlighted.

9.3 Assignments

The assignment area provides the face program in ASCII format.
Also the face program has the same structure already examined for the other faces (see Chapter ***Face->Assignment Area***), with the addition of an information element (property) for each program line: the application face or **F Field**.



					ASCII format			M..			Abc	Description
1	□	□		1 ...	HOLE X340 Y700	0	0	0		0		
2	□	□						0		0		
3	□	□						0		0		

1: Face 1
2: Face 2
3: Face 3
4: Face 4
5: Face 5
6: Face 6

9.4 F Field

F Field: it indicates the face the working is applied to. It is possible to select a face in a window where all real and fictive faces managed on the piece are listed. Parametric programming is not admitted. The F field is important for all workings except for logical instructions (IF, ELSE, ENDIF, assignment of J variables...). In these cases the selection is not managed. The **face selection list** can change according to the selected working. For example a sawing work can usually be applied only to face 1 (top side) and 2 (bottom side), therefore only face 1 and 2 are displayed in the face selection list. In case, before the selected working, some faces are assigned automatically, the selection list for the F field of a working entered in piece-face includes also the AUTO item, which corresponds to the application in automatic face. See Chapter ***Piece-Face->Automatic Faces***. The F field assignment is also managed in Working Assignment Area.

Assignment of profiles and F field propagation
When configuring Tpaedi32 it is possible to choose between two different operating modes:

- **recognition of profile not conditioned by F field**: in this operating mode greater importance is given to the continuity of profiles than to the assignment of the application face. In the case of works on profile (arcs and lines) the recognition of open profile does not take into account the F field assignments relative to the current and the previous workings:
  - if the segment opens a profile it keeps its own original F field programming
  - otherwise: it propagates the F field from the current segment to the previous segments

  In the case of setup or complex working with **request** for point hook, the F field is propagated from the current to the previous workings.

- **recognition of profile conditioned by F field**: in this operating mode greater importance is given to the assignment of the application face than to the continuity of profiles. In the case of works on profile (arcs and lines), the recognition of open profile keeps into account the F field assignments relative to the current and the previous workings and different settings cause the interruption of the continuity of the profile. F field propagation from the current to the previous segments is never applied. In the case of setup or complex working with **request** for point hook: the F field is not propagated from the current to the previous workings and the point hook does not determine the continuity of the profile if the F field setting of the previous workings is different.

9.5 Representation

The piece-face view is set as a three-dimensional (3d) representation of the piece. To activate a plane representation it is necessary to select the relevant icon in the View Settings Bar.
However, the case of the piece-face is particular, compared to real and fictive faces:

- in fact, the displayed face changes as the application face of the current working changes
- the represented face is also the selected face to operate on

If, for example, you wish to insert drawing entities in face 10 with the 3d view active, it is necessary to activate the 2d view first:

- move the current working to one applied in face 10
- insert geometric entities as usual: the entered workings are applied in face 10.

If no workings are applied to the face the relevant icon shall be selected in the View Settings Bar. This command is available only in piece-face and recalls a piece face selection menu. The command has a double function:

- it activates the 2d view graphical representation
- it places the face selected in the list as the active face to be displayed

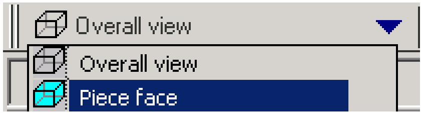
9.6 Utility Of the Piece-Face

It may be asked: "*Why do not use always and only the piece-face to assign the list of program workings?".* The possibility is concrete and certainly finds applications in many fields. In any case, it must be observed that the piece-face has a limitation: a piece-face program **cannot be used as subroutine.**
That is why the piece-face is only available in case of piece with program typology. In case of subroutine or macro-program typology, the face is not managed.

If the end user of Tpaedi32 needs to write subroutines, to apply them, he must first program workings in the views of the enabled faces. It is possible to use a subroutine also in the piece-face of a program:

- the F field set in the subroutine call states in which face the subroutine is applied
- the selection of which face of the subroutine to recall follows the same procedures outlined above (Face item which can be found, for example, among the parameters of the SUB working).

In any case, the solution to make programmable only the piece-face of programs can be implemented: Tpaedi32 can be configured in such a way to make only the piece-face accessible and the Face Selection Bar appears as in the Figure below:



This is only valid in the case of pieces with program typology.
In the case of pieces with subroutine or macro-program typology, the *Face Selection Bar*:

- does not manage the piece-face selection,
- manages the selection on the assigned real (those enabled) and fictive faces.

9.7 Generation Of Execution Sequences

One feature of the piece-face program is that the execution sequences are directly defined. During the assignment of sequences, the workings programmed in piece-face are not shown.
The workings in piece-face are executed before any other list of face workings, in the list order as they are programmed.
One of the main reasons for which the piece-face is useful is that it allows to group in safe manner the

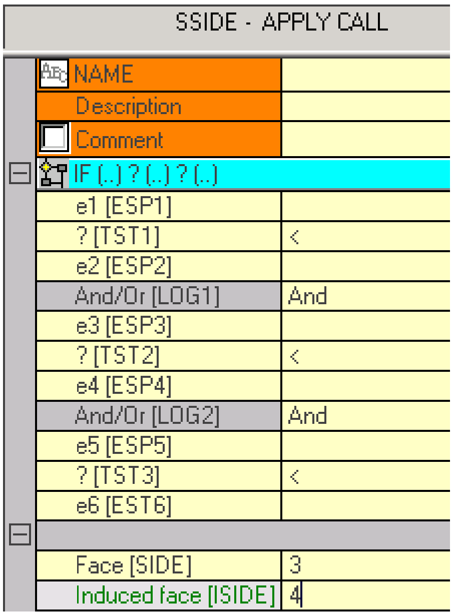
program workings which need to be executed according to a predefined sequence.
A typical example is represented by the creation of fictive faces (for example by a cut working) during execution: it is necessary to ensure that the face creation is implemented before working the face itself. In this case it can be useful to assign the cut working in piece-face. In any case, the program will be modified, but with the warranty that the face is created immediately.

9.8 Application Of Subroutines To the Correct Face

Induced calls
The behaviour of the piece-face may be equivalent to that of any other face. (See Chapter ***Workings->Types of Workings->Subroutine->Application of Workings to the Correct Face***). The actual behaviour depends on how Tpaedi32 is configured.
In fact, it is possible to select between three different types of management for induced calls:

- **Disabled management**: the piece-face disables the management of induced calls
- **Programmed management**: the piece-face resolves only programmed induced calls (now induced calls are entered into the piece-face itself, after the directly programmed workings).
- **Automatic management**: the behaviour of the piece-face is equivalent to that of any other face

Programmed induced calls
The programming of an induced call can only result from the application of a subroutine or a macro in the piece-face. An induced call can be assigned to any application face selected among the real or fictive faces of the piece.
The programming is implemented by the SSIDE working, which can be selected in the SUBROUTINES group. This working can only be entered in a subroutine or macro text and becomes operative only when the subroutine is inserted in the piece-face:



SSIDE - APPLY CALL

NAME	
Description	
□ Comment	
IF (..) ? (..) ? (..)	
e1 [ESP1]	
? [TST1]	<
e2 [ESP2]	
And/Or [LOG1]	And
e3 [ESP3]	
? [TST2]	<
e4 [ESP4]	
And/Or [LOG2]	And
e5 [ESP5]	
? [TST3]	<
e6 [EST6]	
Face [SIDE]	3
Induced face [ISIDE]	4

- **node "IF (..) ? (..) ? (..)"**: the conditions of application are assigned with a direct control-expression IF, followed by up to three conditional statements between two clauses. If the conditional statements are verified the instruction can interpret an induced application of the subroutine call
- **Face**: it sets the subroutine face to apply in the induced call
- **Induced face**: it sets the application face of the induced call

In the example the instruction programs an induced call in face 4, with application face 3 of the subroutine.
The SSIDE instruction does not resolve the induced call if the ***Programmed management*** of induced

calls is not enabled in Configuring Tpaedi32.

Solutions of <j> variables in programmed induced calls
The development of a programmed induced call can use <j> variables.
For example:

- sub-program ONE is written with workings assigned on faces 1 and 3:
  program in face 1:
    - line 1: code <ASSIGN Jnn> and value 100 is assigned to j0 variable
    - line 2: code <SSIDE-APPLY CALL> value 3 is set on Face field, value 4 on Induced Face field

  program in face 3:
    - line 1: code <HOLE> X=j0 coordinate is set up. Hole shall be at coordinate x = 0.0.
    - line 2 code <ASSIGN Jnn> and value j0=j0+100 is assigned;
    - line 3: code <HOLE> X=j0 coordinate is set up. Hole shall be at coordinate x = 100.0.
- program PRG1 is written, with working assigned on face-piece:
    - line 1: code <SUB> sub-program ONE is applied to face 1: sub-program in face 1 assigns j0 variable and develops programmed induced call in face 3;
    - line 2 code <SUB>: sub-program ONE induced call in face 3: the first hole is now made on coordinate X=100, the second X=200.

This example shows how a programmed induced call initially uses J variables like assigned at induced call time: afterwards, new assignments are added to initial situation.

9.9 Automatic Faces

PROFESSIONAL

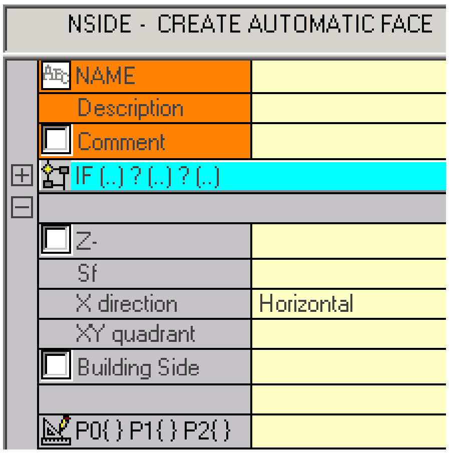
It is an optional operating mode.
The automatic faces are faces directly created during the programming of the piece-face. The face numbering is automatically and sequentially managed (101 to 500).
The visibility of the automatic faces is limited to the piece-face.
The creation of an automatic face enables the following assignment of workings to it: always and only during the programming of the piece-face. On the contrary it is not possible:

- to gain direct access to an automatic face view
- to create and/or assign workings to an automatic face from a face different from the piece-face

An automatic face cannot be directly selected by the number automatically assigned to the face: the last assigned face is the only one which is always accessible.
Therefore, the mechanism of use of the automatic faces is the following:

- …
- a number (the first unassigned number, for example 105) is automatically assigned to an automatic face;
- workings are applied to the last created automatic face (105);
- …
- a number (the first unassigned number, for example 106) is automatically assigned to an automatic face;
- workings are applied to the last created automatic face (106);
- …

In any point of the piece-face program is then available only a specific automatic face: the last created before the selected working. In this sense the general expression *Application in automatic face* is used.
An automatic face is created by the NSIDE working, which can be selected in the LOGICAL INSTRUCTIONS group. If the NSDIDE working is recalled in a face different from the face of the piece, a fictive face as reference face cannot be assigned.



NSIDE - CREATE AUTOMATIC FACE

NAME	
Description	
□ Comment	
IF (..) ? (..) ? (..)	
□ Z-	
Sf	
X direction	Horizontal
XY quadrant	
□ Building Side	
P0{ } P1{ } P2{ }	

- **node "IF (..) ? (..) ? (..)"**: the conditions of application are assigned with a direct control-expression IF, followed by up to three conditional statements between two clauses. If the conditional statements are verified the instruction can interpret the creation of automatic face
- **P0 {} P1 {} P2 {}**: it opens a window identical to the fictive face assignment window to set the edges of the automatic face. The automatic face assignment reflects the modes available for the fictive face assignment:
  - reference face
  - three distinct points (P0, P1, P2) to determine uniquely a plane as a geometric surface
  - Z axis direction
  - thickness
  - graphical representation mode
  - added parameters
  - configuration as building face (which can be used only as reference face for the assignment of a following automatic face and which cannot be programmed)

The graphical representation in piece-face also includes automatic faces, with the exception of building faces.

The graphical representation in piece-face also includes automatic faces.

Programmed induced calls

A programmed induced call can be applied to the automatic face. It is about the primary use of the SSIDE instruction. In fact, to require the application in automatic face it is enough to leave the *Induced Face* field blank.

10 Tools

10.1 Introduction

By the term Tools we mean all those commands which are specifically dedicated to edit workings by making modifications of geometric and/or technological nature.
Tpaedi32 can make available many tools which we subdivide into four groups:

General tools: they are powerful tools for editing workings. It is possible to modify the position and dimension of a working or a set of workings, or enter new copies or convert them into the simple workings which compose them.

Profile tools: they operate exclusively on profiles already inserted in face.

Advanced tools in face program: a few of these tools operate on profiles already inserted in face, others generate they themselves profiles, on the basis of particular procedures.

Overall program tools: they usually act on profiles and work in the same way as the tools belonging to the set of Profile tools, but they operate on the piece as a whole. They do not apply view filters and can be activated automatically also at the program startup. In fact these tools are typically used in case of a program import

The first three sets of tools work only in face view, while the last works only in piece overall view.
For all tools the last confirmed settings are always saved and they are retrieved next time the tool is used.
Tools are applied to the only workings which verify the active view filters: selections, logical conditions, levels, special filters.
If the tool is directly applied to the (selected or active) original workings, the modification cannot be applied to workings in locked status (it is about an induced call or it has Level, Construct or “O” field locked).
It is possible to apply a few tools to a copy of the selected working by recalling the ***Tools->Apply tool to a copy of the workings*** menu item or from the Tool Options Bar by clicking on the relevant icon.
It is also possible to apply the selected tool to the workings copied to Clipboard by selecting the ***Tools->Apply tool to workings in clipboard*** menu item or from the Tool Options Bar by clicking on the relevant icon .
Profile and Advanced tools in Face program which generate new profiles are opened by a copy:

- of the original setup, if the profile is not opened
- of the reference setup (as assigned in Customization of Tpaedi32).

10.2 Positioning Modes

It is possible to define the position of workings on the face by acquiring the coordinates of the mouse position or assigning cartesian coordinates by selecting one of the two items from ***Tools->Positioning*** menu or from the Tool Options Bar by clicking on the relevant icons and .
If the cartesian positioning system is selected, in many tools coordinates shall be entered in absolute or relative cartesian positioning or in polar positioning by indicating the pole center, the module and the angle.
By selecting the mouse positioning the coordinates are assigned only in absolute positioning. If snap-to-grid is enabled, the coordinates obtained with the mouse coincide with the snap point nearest to the mouse click point.

10.3 General Tools

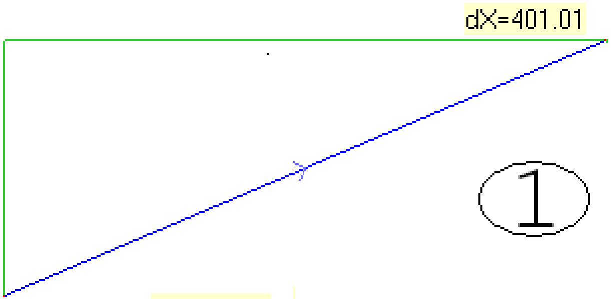
10.3.1 Measure

Measure the linear distance between two points lying on the plan of the face. The command is recalled from Menu ***Tools->Measure*** or from the icon . The position of the points is assigned with the mouse, directly in the graphics area. The information are given in the area of *Commands State.* Pressing the right mouse button a popup menu opens that allows to activate the snap. The distance between the two points is drawn through a linear segment. The geometric data related to the distance are displayed, as well: change in X, Y and reale distance on the xy plane.

10.3.2 Dimensioning

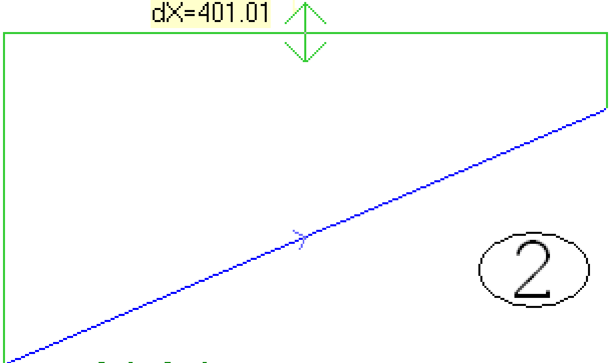
PROFESSIONAL

This tool is available only if in the database of the workings a specific working code for measuring has been assigned and if working properties Construct and Name has been enabled.
The measuring tool enables the addition of data measurement to the program. The segment to measure is chosen with the mouse, directly in the graphics area. Indications about the sessions sequence are given in the Commands area. Pressing the right mouse button a popup menu opens that allows to activate the snap.
From Menu: ***Tools->Dimensioning*** the type of measuring required can be chosen:
Horizontal: inserts an horizontal measuring line and its position.
Vertical: inserts a vertical measuring line and its position.
Horizontal+Vertical: inserts a vertical measuring line and its position and an horizontal measuring line and its position.
Diagonal: inserts an horizontal measuring line on the intercepted points and its position.
An example of horizontal dimensioning follows:

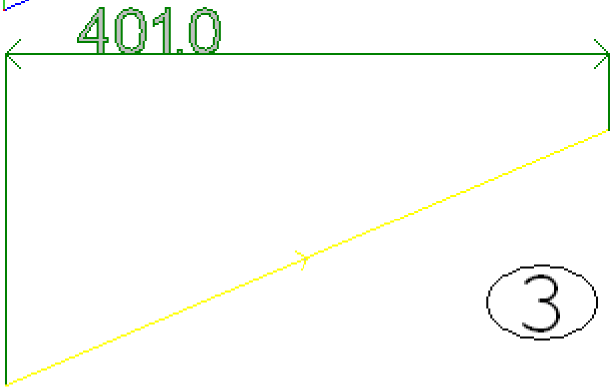


The figures show the sequence of necessary steps that complete the'Insertion:

1) identification of two extreme points of the linear segment to calculate the horizontal dimensioning (dX=401.01)



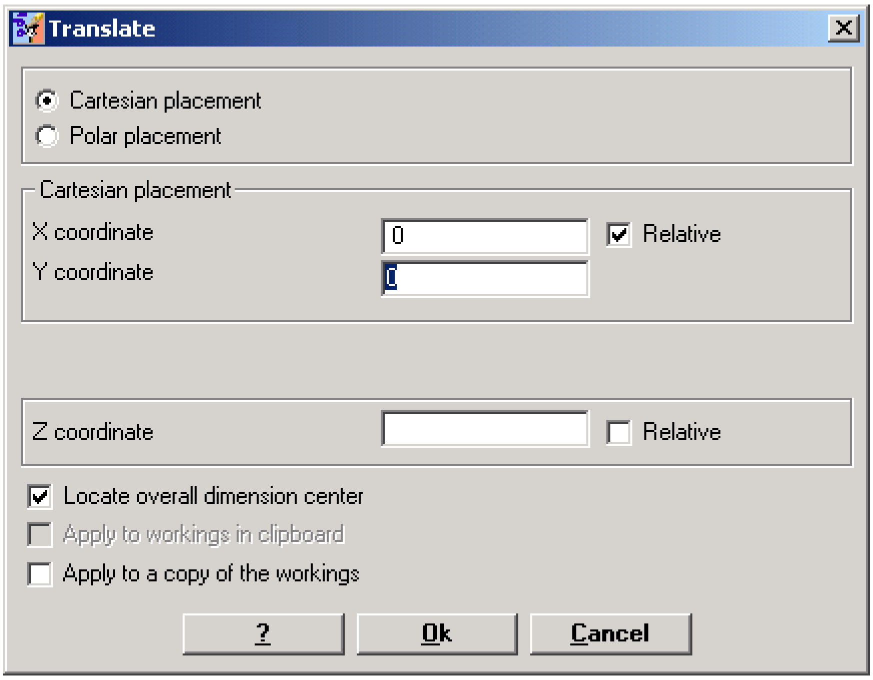
2) setting of the vertical position of the written dimension.



3) After confirming the command, a horizontal segment with arrows at its edge points and the value of the segment (401.0 in this case) is inserted.

10.3.3 Translation

This tool moves workings to the assigned position. It is recalled by selecting the ***Tools->Translate*** menu item or the relevant icon .
The translation of a working which belongs to a profile involves the translation of the entire profile.



Translate

● Cartesian placement
○ Polar placement

Cartesian placement

X coordinate 0 ☑ Relative
Y coordinate 0

Z coordinate ☐ Relative

☑ Locate overall dimension center
☐ Apply to workings in clipboard
☐ Apply to a copy of the workings

In the case of absolute positioning, if the tool must operate on the workings stored in Clipboard, the first working in the list is placed at the specified position, otherwise the active working is moved to the specified position.

If the **Locate overall dimension center** option is enabled the overall dimension center of selected workings is translated.

The position is assigned in the window if the Cartesian Positioning flag is enabled. In this case it manages:

- X and/or Y coordinate in absolute or relative positioning and/or
- Z coordinate in absolute or relative positioning

In case of absolute positioning:

- if the tool copies workings to Clipboard the first working entered is moved to the specified position
- otherwise the active working is moved to the specified position

The position is assigned with the mouse if the Mouse Positioning flag is enabled.
In this case it manages:

- XY coordinates in absolute positioning;
- indications are provided in the *Commands Status* area
- if active, it applies the snap to grid.

10.3.4 Rotation

It rotates the selected workings. This tool is recalled by selecting the ***Tools->Rotate*** menu item or by clicking on the relevant icon .

The rotation of a working which belongs to a profile involves

- the rotation of the entire profile, if *Mouse Positioning* is active or if the rotation is not applied to the current working, but it is applied to the workings in the local Clipboard or to the selected workings..
- The rotation of the part of profile between the current working and the end of the profile (select the option ***Apply from current working until the end of the profile***). In this case only, if the selection of the option it is inserted in any case a copy of the entire profile. Furthermore, the rotation centre coincides with initial point of the current working.

The data for the application of rotation are assigned:

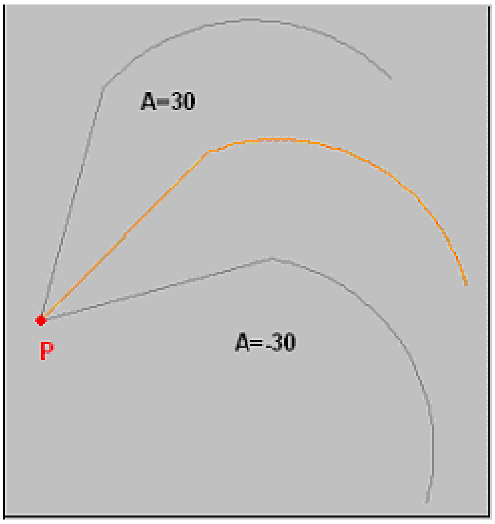
- in the window if **Cartesian Positioning** is active:
  - **x, y coordinates of the rotation center** in absolute or relative positioning. In this last case if

rotation is applied to workings in Clipboard, the x, y coordinates of the rotation center are relative to the point of application of the first working entered, otherwise they concern the active working.

- a **rotation angle** in absolute or relative positioning.

- with the mouse if **Mouse Positioning** is active:
  - XY coordinates in absolute positioning of the rotation center and a final rotation angle. Indications are provided in the *Commands Status* area. If active, it applies the snap to grid, but only for the positioning of the rotation center. At the and of the placement the window where to set the data, that can be modified, is displayed.

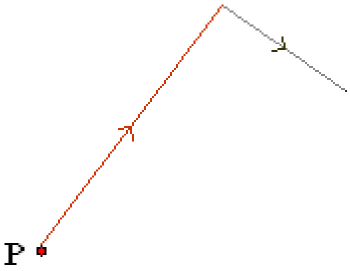
Let us see an example of rotation:



active working in P; center coordinates (0;0) in relative positioning: the center is thus positioned in P; rotation angle in relative positioning, of value:

- 30° (for upward rotation)
- -30° (for downward rotation).

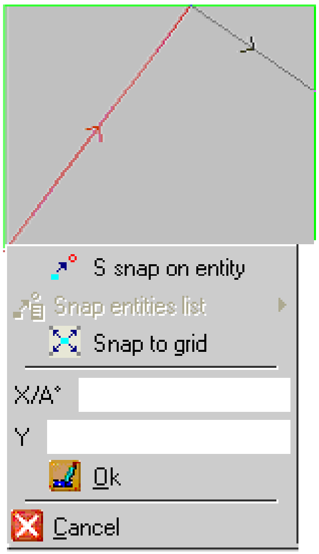
The rotation can also be executed in interactive way. Let us see an example:



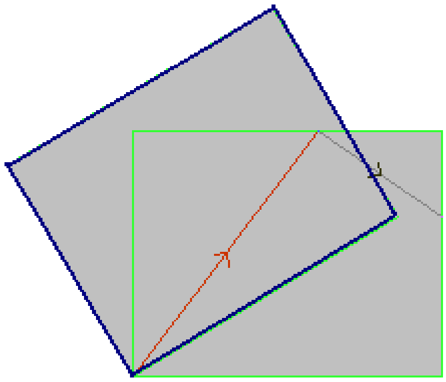
We want to rotate the profile displayed in the image beside around the point **P** in a interactive way.

Let us first select the positioning button with the mouse

then the tool Rotation



Workings, to which the rotation is applied, are framed into the overall rectangle. At this point, in order to set the rotation point press the right button of the mouse. The pop-up menu appears. Enable the option **Snap on entity** and position the mouse pointer next to the point **P** and confirm by left-clicking the mouse.



The overall rectangle now moves according to the movement of the mouse pointer.To confirm the rotation press the mouse left button. Alternatively, press the right button of the mouse to open the pop-up menu. Enable the Snap option or assign directly the coordinates X,Y that set the position of the rotated axis or the inclination angle of the overall rectangle (A °), then confirm with the **[OK]** button.

The rotation tool cannot be applied to all workings. For example, all complex workings which verify one or both the following conditions are excluded:

- they recall a subroutine or a macro to which no rotation can be applied as established in the workings database.
- they themselves are configured in the workings database as workings to which no rotation can be applied.

Typical examples are sawing works undertaken with tool that cannot be oriented.

10.3.5 Mirrors

Mirror tools reflect the selected workings with respect to a specified axis. Four types of mirror are defined and they can be selected from the ***Tools->Mirror*** menu item or by clicking on the relevant icons in the General Tools Bar.

Let us see the above commands in detail:

	Tools->Mirror->Vertical mirror
	Tools->Mirror->Horizontal mirror
	Tools->Mirror->Horizontal + Vertical Mirror
	Only from menu. ***Tools->Mirror->Generic Mirror***

In the case of **Vertical mirror** or **Horizontal Mirror** with Cartesian positioning in the dialog box a parametric coordinate is provided for the mirror axis with respect to the height and length of the active face. This because the combination of rotation axis and face axis (x or y) depends on the representation system configured for the face itself.

If the tool is applied to a profile, also the settings of

- tool compensation (right or left);
- selection of entry/exit segments, in case of right or left arc settings

are inverted.

In case of Mirrors around a vertical axis or Mirrors around a horizontal axis with mouse positioning the mouse pointer can be moved only along the selected mirror axis.

The mirror of a working which belongs to a profile involves

- the mirror of the entire profile if *Mouse Positioning* is active or if the rotation is not applied to the current working, but is is applied to the workings in the local Clipboard or to the selected workings,or if the mode of *Mirror around a generic axis* is selected;
- the mirror of the part of profile between the current working and the end of the profile (select the option ***Apply from current working until the end of the profile).*** In this case only, if the selection of the option *Apply tool to a copy of the workings* is active, in any case a copy of the entire profile is inserted. Furthermore, the mirror axis coincides with initial point of the current working.

The mirror tool cannot be applied to all workings. For example, the complex workings which verify one or both the following conditions are excluded:

- they recall a subroutine or a macro to which the selected mirror cannot be applied as established in

the workings database
- they themselves are configured in the workings database as workings to which the selected mirror cannot be applied

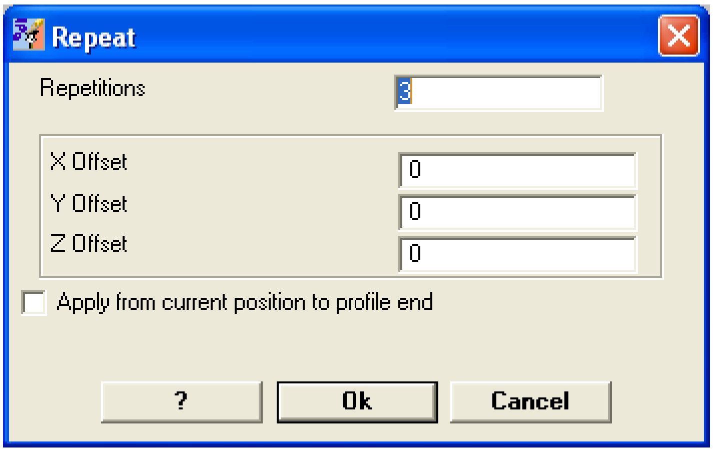
Typical examples are sawing works undertaken with tool that cannot be oriented.

10.3.6 Repetitions

Repeat

The ***Repeat*** tool, which can be selected from the ***Tools->Repeat*** menu item or from the Tool Bar by clicking on the relevant icon , copies the active working or the selected workings to the face as many times as it is required in the **[Repetitions]** box. The repetition of a working which belongs to a profile involves:

- the repetition of the whole profile, if the tool is not applied to the current working, but is applied to the workings into the Clipboard or to the selected workings;
- otherwise the part of profile between the current working and the end of the profile can be repeated (select the option ***Apply from current position to profile end***). In this case, the positioning offsets are determined automatically. To define the working coordinates it is needed to assign the following values: **[X Offset]**, **[Y Offset]** and **[Z Offset]**.



Repetitions 3

X Offset 0

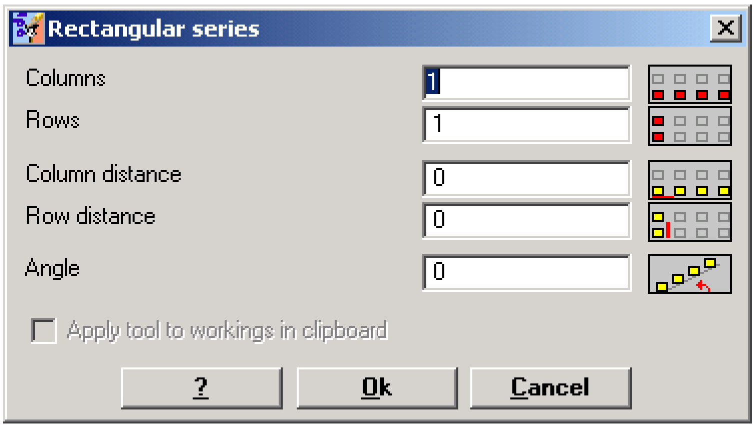
Y Offset 0

Z Offset 0

□ Apply from current position to profile end

Rectangular series

The **Rectangular series** tool, which can be selected from the ***Tools->Rectangular series*** menu item or from the Tool Bar by clicking on the relevant icon copies the active working or the selected workings according to a matrix scheme.



Columns 1

Rows 1

Column distance 0

Row distance 0

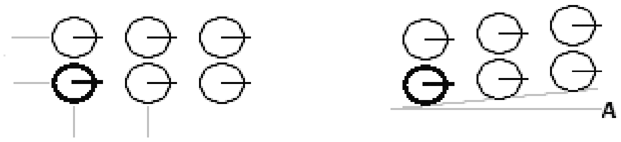
Angle 0

□ Apply tool to workings in clipboard

The following values shall be typed:

- a number of **[Columns]** and a number of **[Rows]**: the range of set, also parametric, values must be included between 1 and 1000. Both fields must be filled in with a value greater than 1 and the total number of repetitions cannot be greater than 1000.
- **[Column distance]** and **[Row distance]**: they are significant values with sign.
- **[Angle]**: rotation angle

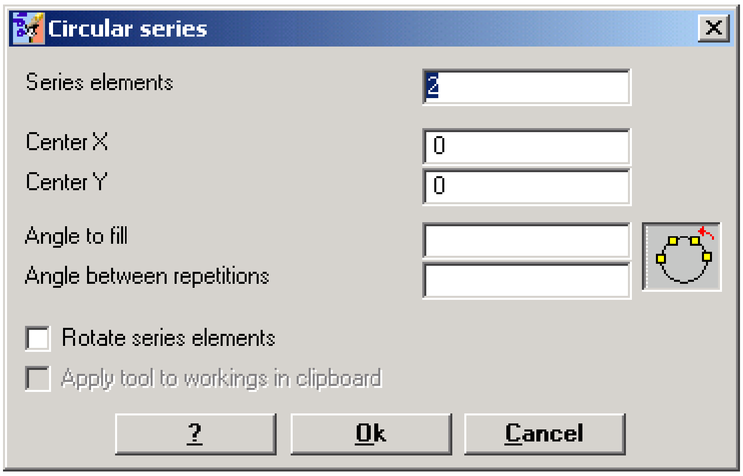
Let us see an example



Repetition of the working marked with a bold line for a number of **3** columns and a number of **2** rows.
The left Figure shows a series with **A**=0.
The series on the right-hand displays the same series with **A**#0.

Circular series

The ***Circular series*** tool, which can be selected from the ***Tools->Circular series*** menu item and from the Tool Bar by clicking on the relevant icon , copies the active working or the selected workings along a circular path where the arc starting point is the active working. The position of the middle of the arc is determined with the mouse. Then, a dialog box opens, as follows:



Circular series

Series elements: 2

Center X: 0

Center Y: 0

Angle to fill:

Angle between repetitions:

☐ Rotate series elements

☐ Apply tool to workings in clipboard

The following values shall be typed:

- **[Center X]**, **[Center Y]**: center of the arc along which the repetition of the selected workings takes place. The given coordinates are those identified with the cursor of the mouse and can be changed.
- **[Series elements]**: number of series elements, included the first one. The range of set, also parametric, values must be included between 2 and 1000;
- **[Angle to fill]**: the range of set, also parametric, values must be included between 0.001° and 360°
- **[Angle between repetitions]**: it is interpreted only in a few cases which are described later

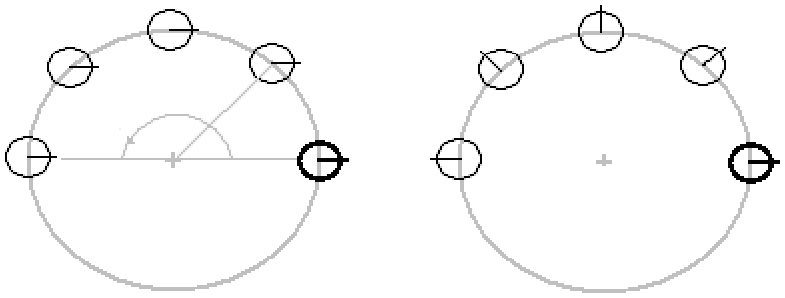
Of the three remaining parameters, shown in the dialog box, only two must be set, the third is automatically calculated according to the priority order:

Series elements and **Angle to fill** must be set: the [Angle between repetitions] field is ignored and the angle between elements [Angle to fill] is automatically calculated.

Series elements is unset: both fields relative to angles must be set. The number of series elements is automatically calculated.

Angle to fill is unset. Both the [Series elements] and [Angle between repetitions] fields must be set.

It is possible to rotate the series elements. Let us see an example:



Repetition of the working marked with a bold line for **5** elements; Angle to fill: **180°**; counterclockwise rotation.
The left Figure shows the series with rotation flag of copied objects inactive.
The series on the left-hand displays the same elements series with rotation flag of copied objects active.

10.3.7 Explode

The Explode tool breaks complex workings or multiple segments of profile into simple workings which compose them. This tool can be selected from the ***Tools->Explode*** menu item and from the Tool Bar by clicking on the relevant icon ***.***
If the tool is applied to workings in Clipboard, the only workings entered are those which correspond to the exploded view of original workings.
The Explode tool cannot always be applied. For example the complex workings which verify one or both the following conditions are excluded:

- they recall a subroutine or a macro to which the transform cannot be applied as established in the workings database.
- they themselves are configured in the workings database as workings to which no transform can be applied.

10.4 Profile

10.4.1 Change the edge into arc

It changes an edge into an arch. This tool is recalled from the menu ***Tools->Change the edge into arc*** and from the Toolbar from the icon .
The tool operates on extended profiles, but its application is possible only on an edge identified by two simple straight lines. The tool works directly on the current profile. In a dialog box is proposed the plan on which the three edges lie, as a plan on which to calculate the arc. If on the selected plane the geometrical conditions defining an arc do not exist, the transformation is not carried out. The instrument is disabled, if:

- there are not any programmed workings
- the current working is an induced working
- the current working or the following one are not straight lines
- the three vertices of the edge are not distinguished

10.4.2 Lengthen a profile segment

It lengthens or shortens the trait modifying its final point. The tool is called in the menu ***Tools->Lengthen*** and Advanced Tool and Path Bar
by clicking on the relevant icon .

The active trait must belong to a profile, to be simple and of arc or line typology. Furthermore, a linear trait cannot be of null length and an arc cannot make a circle.

How the tool is activated depends on the system of the active positioning.
If the **cartesian placement** is active or if the current segment defines an arc on a different plane from xy, the value of the linear **Length** of the trait or the value of the **Angle subtended** of the arc (in **°)** must be specified. After opening the window, the values obtained from the original trait are suggested.
If the **mouse positioning** is active, the new position of the final point is found by the cursor position within the graphic area, clicking on the mouse. It is considered the nearest point to the cursor position on the active trait.
The tool does not work if the current working does not verify the active sight filters (selections, levels logical conditions, special filters) of if the profile is in a lock state (it is an induced call or his level, construct or O field are locked).
The toll is deactivated, if:

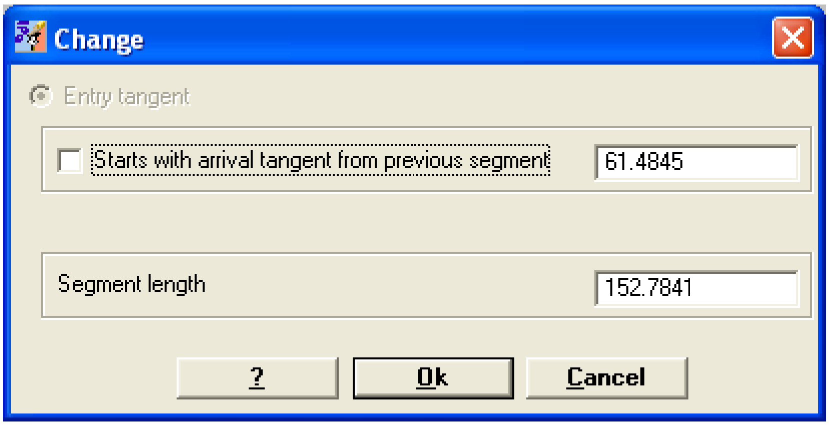
- there is not any programmed working
- the current call is induced
- the current call is not of any linear segment or arc typology.

10.4.3 Change a profile segment

It changes the active trait by modifying its geometry. The tool is called in the menu ***Tools->Change*** and from Advanced Tool and Path Bar

by clicking on the relevant icon .
The active trait must belong to a profile, to be simple and of arc or line typology. The arc or the conic segments (ellipse), that are assigned on a different plane from the one of the face, are excluded. Furthermore, a linear trait cannot be of null length and an arc cannot make a circle.
How the tool is activated depends on the system of the active positioning.
If the **cartesian placement** is active and the selected trait is a **linear segment**, it is possible to set up:

- the trait inclination in unit (°), by imposing the tangency with the previous trait, or by assigning the inclination value (in the window an initialized value to the value of the original trait inclination is displayed)
- the length of the trait (in the window an initialized value to the value ot the original trait length is displayed)



□ Starts with arrival tangent from previous segment 61.4845

Segment length 152.7841

If the **cartesian placement** is active and the selected trait is an **arc segment** it is possible to set up:

- the starting inclination in unit (°), by imposing the tangency with the previous trait, or by assigning the inclination value (in the window an initialized value to the value or the original trait inclination is displayed)
- the final inclination of the trait in unit (°) by assigning the inclination value (in the window an initialized value to the value of the original trait inclination is displayed)

If the **mouse positioning** and the selected trait is a **linear segment**, the final point of the trait is found by the cursor position.

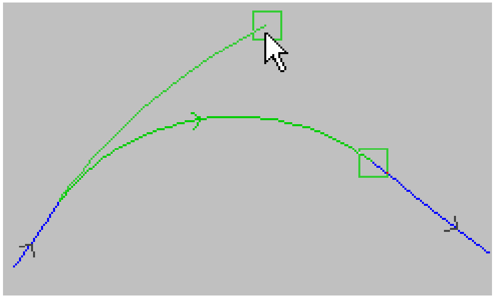
If the **mouse positioning** and the selected trait is a **arc segment**, the final point is found according three different modes:

- the final point of the trait is found by the cursor position.
- in the local menu (that can be activated through the right mouse taste) the option **Move the centre**, moves the centre of the arc .
- in the local menu (that can be activated through the right mouse taste) the option **Move the middle point,** moves the middle point of the arc.

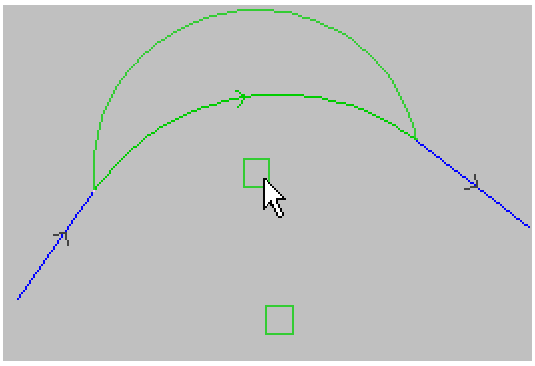
The consequent change of the arc can be displayed in the graphic area, by moving the mouse.

Hereunder three situation to modify the segment of arc typology:

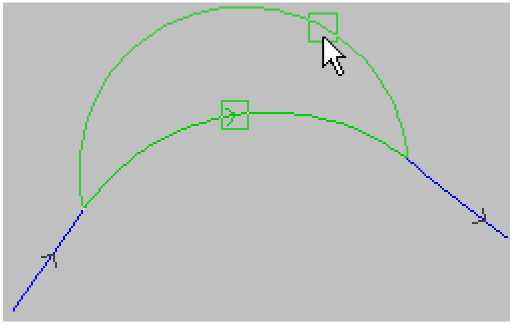
1- Displacement of the final point of the arc



2- Displacement of the center of the arc



3- Displacement of the middle point of the arc



The change of the segment can involves the change of the operational code, a segment becomes a linear L01 working and an arc becomes an arc through three points.

The tool does not work, until the current working does not verify the active sight filters (selections, levels logical conditions, special filters) or if the profile is in lock state (it is an induced call or it is of a level, construct or locked O field).

The toll is disabled, if:

- no workings are programmed
- the current working is induced
- the current working is not of any linear segment or arc typology.

If the current working is a conic arc (ellipse or oval) or an arc on a plane different from xy, the tool **Lengthen a profile segment** is applied to the above segments.

10.4.4 Split the Profile

This tool splits the current profile in the specified point. It can be selected from the ***Tools->Split the profile*** menu item and from the Tool Bar by clicking on the relevant icon ***.***
The cut position is indicated:

- after the current working
- in the specified position: in the window or with the mouse (it does not apply the snap to grid, even if it is active).

If the original profile is an open profile (i.e, it does not start with a setup working), if possible, the new profile created after the cut will be an open profile, otherwise the new profile created after the cut is opened with a copy

- of the original profile's setup, if the profile is not opened, otherwise
- of the reference setup, according to the assignment given in *Customization of Tpaedi32.*

10.4.5 Invert the Profile

This tool reverses the direction of selected profiles. It can be accessed from the ***Tools->Invert the profile*** menu item and from the Tool Bar by clicking on the relevant icon ***.***
The tool is applied to:

- all profiles which have at least a selected element
- the current profile.

The tool also inverts the settings of

- tool compensation (right or left);
- selection of entry/exit segments, in case of right or left arc settings.

The tool operates directly on the concerned profiles or on a copy of these last.

10.4.6 Move Setup To a Closed Profile

This tool moves the current profile setup to the specified point. The profile must be closed. It can be selected from the ***Tools->Move setup to a closed profile*** menu item and from the Tool Bar by clicking on the relevant icon ***.***
The setup position is indicated

- after the active working, or
- in the specified position in the window or with the mouse (it does not apply the snap to grid, even if it is active).

The tool operates directly on the current profile or on a copy of this last.

10.4.7 Scale the Profile

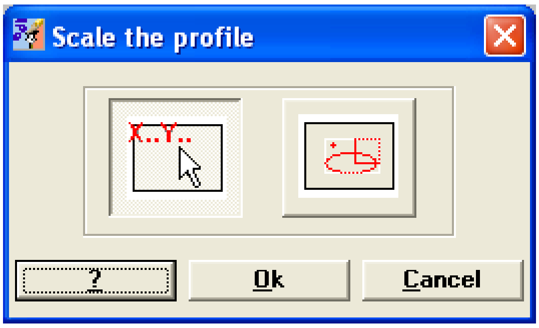
This tool applies a scale factor to one or more profiles, with possibility to operate in xy plane or in xyz space. The tool can be selected from the ***Tools->Scale the profile*** menu item and from the Tool Bar by clicking on the relevant icon ***.***
It is applied to:

- all profiles which have at least a selected element
- the current profile

The tool operates directly on the concerned profiles or on a copy of these last.
This tool can be applied

- to the entire profile if *Mouse Positioning is active,* or if it is not applied to the current profile;
- to the part of profile between the current working and the end of the profile, if the option *Apply from*

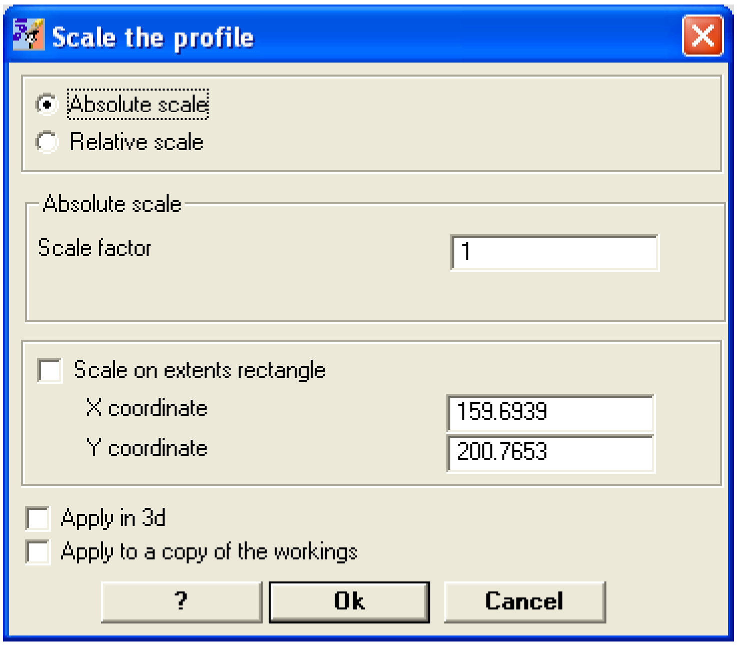
current position until the end of the profile is selected. In this case only, if the selection of the option *Apply tool to a copy of the workings* is active, it is inserted in any case a copy of the entire profile. Furthermore, the basic point coincides with initial point of the current working.



The three bitmaps in sequence enable the assignment of the base point:

- **by coordinates**: the position coordinates are obtained by clicking with the mouse left button on an empty place on the face plane. It applies the snap to grid, if active
- **on extents rectangle**: the base point is automatically assigned at the overall dimension center of the profiles to be resized with the transform.

The coordinates of the basic point are displayed again in the dialog box and can be modified.



● Absolute scale
○ Relative scale

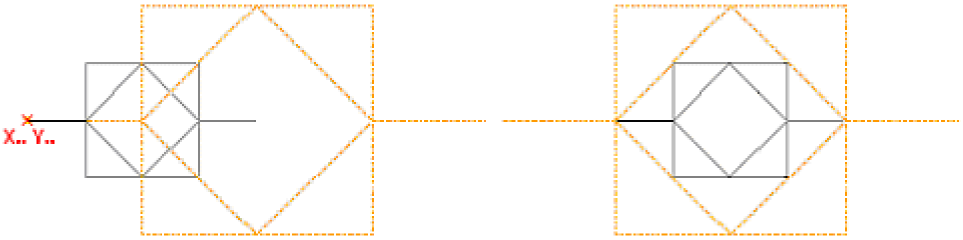
Absolute scale
Scale factor: 1

□ Scale on extents rectangle
X coordinate: 159.6939
Y coordinate: 200.7653

□ Apply in 3d
□ Apply to a copy of the workings

- **Absolute scale**: it directly assigns the **scale factor** value. To enlarge a profile it is necessary to set a value greater than 1; to reduce a profile it is necessary to set a scale factor ranging between 0.01 and 1
- **Relative scale**: the scale factor is obtained by using reference length values. For example a 5.0 long segment would be expanded to 10.0, by the application of an absolute scale factor equal to 2.0.
- **Apply in 3d**: if enabled, the scale is also applied to the face depth, otherwise only to the xy plane of the face.

Let us see an example:



Required scale factor: **0,5**

Base Point

Left Figure: on the left-hand side of profiles

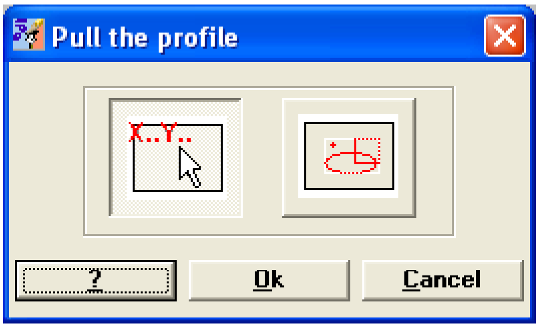
Right Figure: centered on the overall dimension rectangle

The *Scale the profile* tool halves the dimension of each segment of profile and each profile halves its own distance from the specified base point.

If also a entry and/or exit trait is assigned to the profiles, the scale factor is also applied to themselves.

10.4.8 Pull the Profile

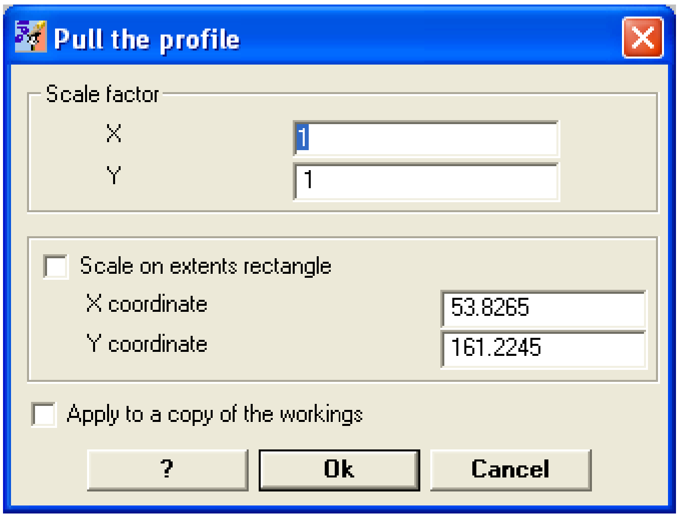
A scale factor is applied to one ore more profiles, where the scale applied in x and in y can differ. The tool is recalled from Menu **Tools->*Pull the profile*** and from Toolbar in icon .



The three bitmaps in sequence enable the assignment of the base point:

- **by coordinates**: the position coordinates are obtained by clicking with the mouse left button on an empty place on the face plane. It applies the snap to grid, if active
- **on extents rectangle**: the base point is automatically assigned at the overall dimension center of the profiles to be resized with the transform.

The coordinates of the basic point are displayed again in the dialog box and can be modified.



Pull the profile

Scale factor

X: 1

Y: 1

□ Scale on extents rectangle

X coordinate: 53.8265

Y coordinate: 161.2245

□ Apply to a copy of the workings

The scale factors are marked out on 2 axes.

The arcs stretching generates automatically an arc of ellipse.
In case of profile/profiles with assigned entry and/or exit traits, the scale is also applied to the geometry of the traits also if the scale factors are the same.

The tool is deactivated:

- if there aren't any programmed workings
- if the current working is induced and the option "Apply to a workings copy" is not selected
- if the current working does not belog to a profile
- if the current working is complex, without previous or next hooks and if it does not finish with profile elements.

10.4.9 Apply Setup To Profile

This tool applies a technological setup to one or more profiles. Profiles can be opened, that is without starting setup or headed with geometric setup, or with setup technology already assigned.
The tool can be selected from the Tools->Apply setup to profile menu item and from the Tool Bar by clicking on the relevant icon .
It is applied to:

- all profiles which have at least a selected element
- the current profile.

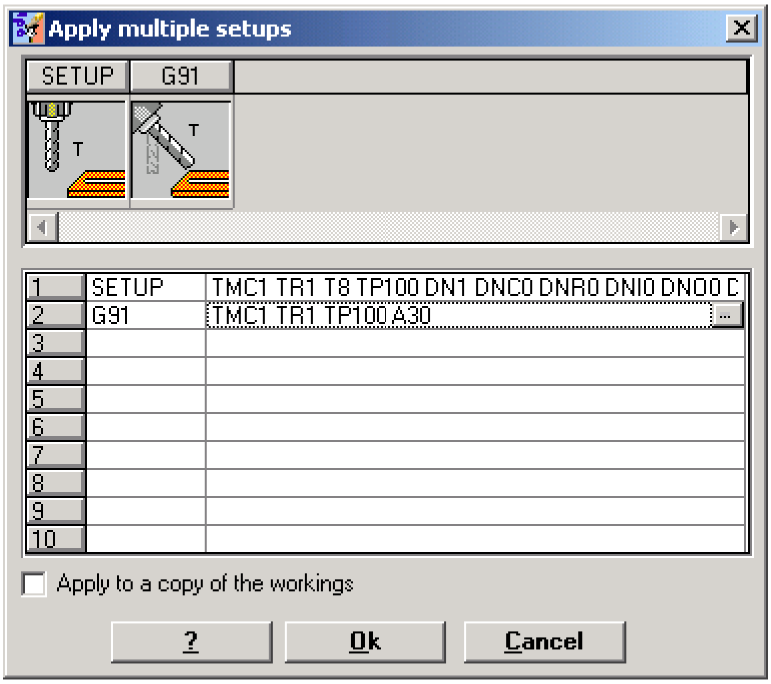
In case of piece-face the tool is only enabled with the 2d face view active and operates only on the profiles applied to the face in current view.
The type of setup to be assigned is selected in a window where all default setup workings are provided.
If the option **Apply also to profiles with technological setup** is enabled, the tool is applied also to the profiles that have an assigned technology.
This involves the change of the setup code and/or of the profiles technology itself.
All technological parameters for the setup assignment process are set in the standard Assign Technology window. See Chapter ***Workings->Profile->Assign Technology***.

10.4.10 Apply Multiple Setups

This tool applies multiple setups to one or more profiles. It can be selected from the ***Tools->Apply multiple setups*** menu item and from the Tool Bar by clicking on the relevant icon .
It is applied to:

- all profiles which have at least a selected element
- the current profile.

The type of setup to be assigned is selected in a window where all default setup workings are provided.



Apply multiple setups

1	SETUP	TMC1 TR1 T8 TP100 DN1 DNC0 DNR0 DNI0 DNO0 D
2	G91	TMC1 TR1 TP100 A30
3		
4		
5		
6		
7		
8		
9		
10		

☐ Apply to a copy of the workings

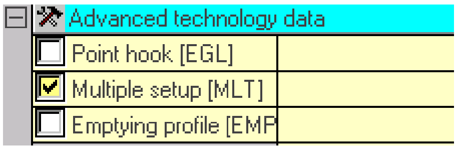
All technological parameters for the setup assignment process can be set in the workings window which is displayed by clicking on the button ... in table. The setup geometry parameters (example: coordinates) cannot be set.

In table it is possible:

- to assign up to 10 setups
- to change the setup order. To move a row it is necessary to select the row header box by clicking the mouse left button, drag then the mouse to the required position and release the button only at the end of the operation
- to delete a setup assignment (**[Canc]** key)
- to enter a setup in the middle of already assigned setups (**[Ins]** key)

We are talking about multiple setups, but it would be better to refer to multiple profiles.

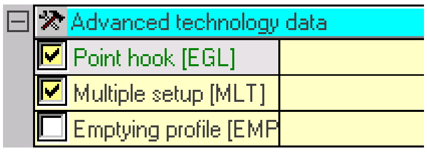
In the above-described example, the tool assigns two setups at the top of each concerned profile. Let us better examine how the tool has assigned the two setups of each profile: in particular let us see a few items included in the *Advanced Technology Data* working node:



Advanced technology data
☐ Point hook [EGL]
☑ Multiple setup [MLT]
☐ Emptying profile [EMP

Following setups:

- *Point hook*: disabled
- *Multiple setup*: enabled



Advanced technology data
☑ Point hook [EGL]
☑ Multiple setup [MLT]
☐ Emptying profile [EMP

Second setup:

- *Point hook*: enabled
- *Multiple setup*: enabled

During execution, a profile headed as examined here is *duplicated* as many times as the number of multiple setups specified. In our example:

- a first time the profile is executed with the first setup (with the relevant technology assigned)
- a second time the profile is executed with the second setup (with the relevant technology assigned)

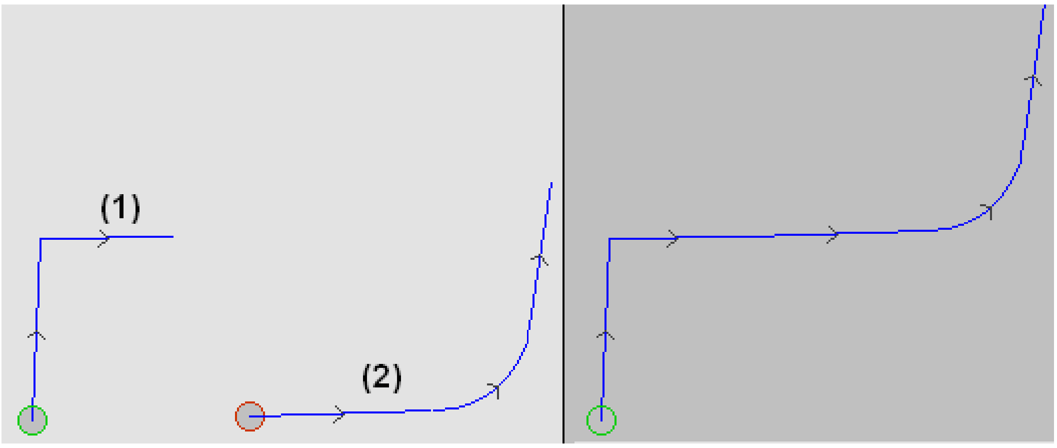
and that without programming the profile twice.

During the profile representation and management in *Tpaedi32*: only the first assigned setup is interpreted for the profile, while for the followings the Point Hook recognition is applied. This last makes the other setups invisible, during the profile execution. Thus: if, for example, the application of tool radius compensation is required, the profile is compensated according to the parameters assigned to the first setup.
The setups after the first one differ only on the status bar in the image showing the movement of the tool.

10.4.11 Profiles Joint

With translation

The tool joins two or more profiles, by translating them so that the starting point of the second profile coincides with the final point of the first profile. Select ***Tools->Profiles joint->Joint profiles by translation*** menu and click on the relevant icon from the Tool Bar.
In case of piece-face the tool is only enabled with the 2d face view active and only operates on the profiles applied to the face in current view.
The tool also acts on extended profiles. Single profiles are identified by directly selecting the translation point in the graphic area by mouse. Indications are given in *Commands display* area.
In case of piece-face the tool is only enabled with the 2d face view active and only operates on the profiles applied to the face in current view.

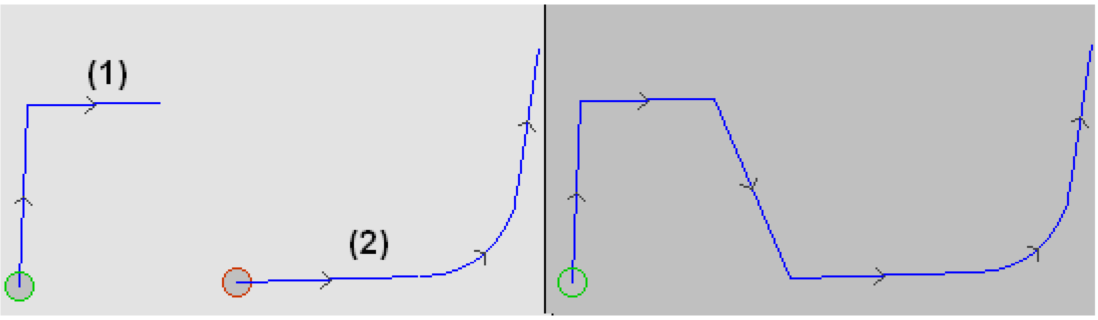


On the left the starting situation is represented, displaying two separate profiles. The profile **(1)** is the first selected profile, while the profile **(2)**is the second selected profile.
On the right the final situation is represented after the tool application. The profil **(1)** remains in the original position, while the profile **(2)** is translated in correspondence of the final point of the straight line **(1).** As a result we obtain one only profile with setup at the profile starting point **(1).**

With connection segment

To join together profiles without moving them from their position, by inserting connecting segments, select the ***Tools->Profiles joint->Connect profiles with segment*** menu item and click on the relevant

icon from the Tool Bar.
The tool also acts on extended profiles. Single profiles are identified by directly selecting the translation point in the graphic area by mouse. Indications are given in *Commands display* area.
In case of piece-face the tool is only enabled with the 2d face view active and only operates on the profiles applied to the face in current view.



On the left the starting situation is represented, displaying two separate profiles. The profile **(1)** is the first selected profile, while the profile **(2)**is the second selected profile.
On the right the final situation is represented after the tool application. Both the profiles remain in their original position, and a straight line between the point of the final straight line **(1)** and starting point of the straight line **(2)** is inserted. As a result, we obtain one only profile with setup at the profile starting point **(1).**

Connect subsequent profiles

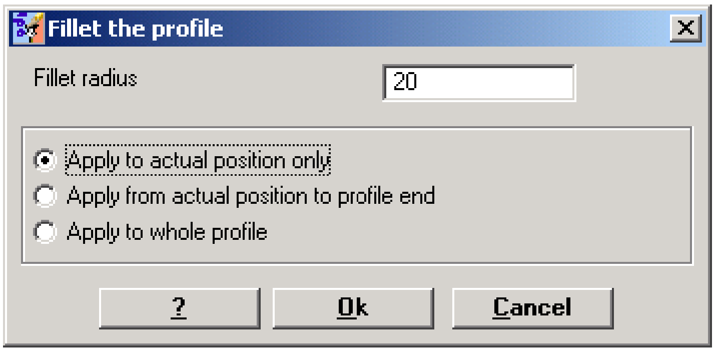
To join together profiles where the final/initial point of one coincides with initial/final point of the following one, select the ***Tools->Profiles joint->Connect subsequent profiles*** menu item and click on the relevant icon from the Tool Bar.
The first element to be set up is **Connection distance,** which represents maximum distance allowed between two profiles (calculated between final point of first profile to initial/following point of the second one) to enable automatic connection. Value allowed is between system *epsilon* and value (100 * system *epsilon*).
With geometric distance > (higher than) system *epsilon*, automatic connection is assigned by shifting the second profile to obtain geometric continuity.

To ensure that the distance of connection is also valued on a component of depth (Z axis) you need to select the option in **Apply in 3d**.

It is then required to indicate the first profile by mouse pointer. Afterwards, **connections automatic search *is suggested***. If the answer is positive, Tpaedi32 automatically connects all subsequent profile, otherwise segments are to be indicated by mouse.

10.4.12 Fillet the profile

This tool inserts fillet arcs at one or more profile edges. The edge is replaced by the arc. It can be selected from the ***Tools->Fillet the profile*** menu item and from the Tool Bar by clicking on the relevant icon **.**


Fillet the profile

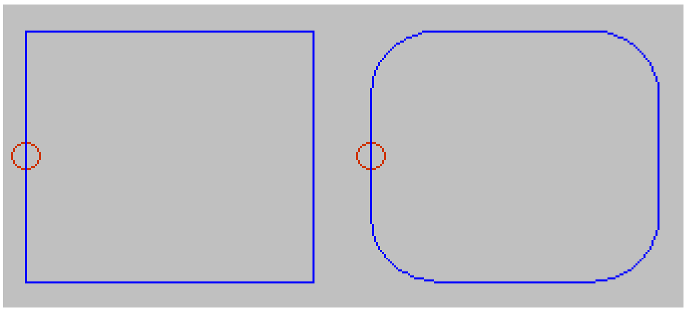
Fillet radius 20

● Apply to actual position only
○ Apply from actual position to profile end
○ Apply to whole profile

- **Fillet radius**: radius of the fillet arc inserted on the edge. The entered arc keeps tangency continuity with adjacent segments.

The tool is applied:

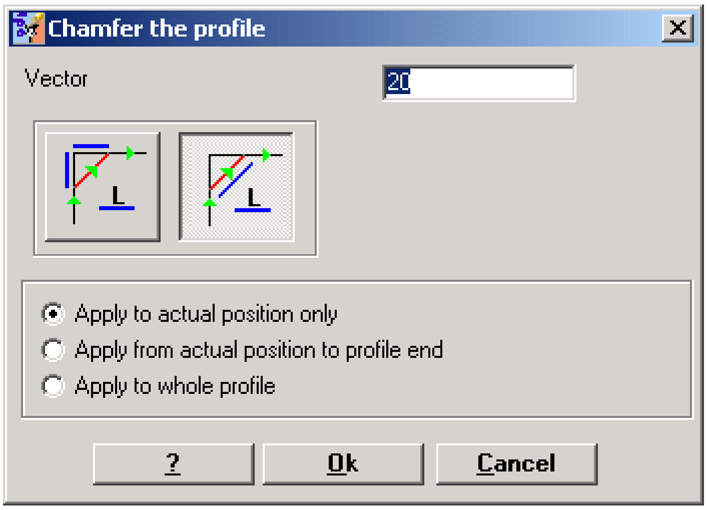
- **to current position only**: it inserts a fillet on the edge formed by the active working with the following working
- **from current position to profile end**: it inserts fillets on each profile edge from the active working
- **to entire profile**: it inserts fillets at all profile edges



On the left a rectangle with sharp edges is displayed.
On the right the result after the tool application to whole profile is finally displayed.

10.4.13 Chamfer the Profile

This tool chamfers the profile at one or more profile linear edges. The linear edge is replaced by a chamfered edge. The tool can be selected from the ***Tools->Chamfer the profile*** menu item and from the Tool Bar by clicking on the relevant icon .


Chamfer the profile

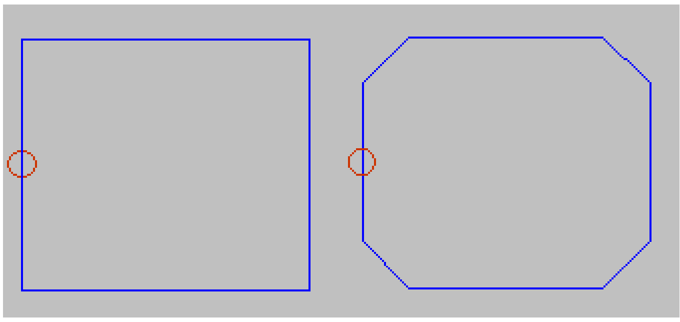
Vector 20

● Apply to actual position only
○ Apply from actual position to profile end
○ Apply to whole profile

- **Vector**: it can assign the distance from the edge or the chamfer length

The tool is applied:

- **to current position only**: it inserts a chamfer (straight connection) on the edge formed by the active working with next working
- **from current position to profile end**: it inserts chamfers on each profile edge from the active working
- **to the profile**: it inserts chamfers at all profile edges

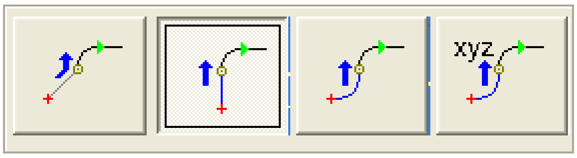


On the left a rectangle with sharp edges is displayed.
On the right the result after the tool application to whole profile is finally displayed.

10.4.14 Apply Entry Point To Profile

This tool applies a linear or circular entry segment to the current profile. If the current profile is an open profile a setup working is added as starting point of the added segment, otherwise the setup is moved to the new profile starting point. The tool can be selected from the ***Tools->Apply entry point to profile*** menu item and from the Tool Bar by clicking on the relevant icon .

The segment is assigned in the window, if Cartesian positioning is active. By a graphic palette it is possible to choose what type of segment to enter:



- **a linear segment with given coordinates**: it inserts a linear segment from the profile starting point to the point defined by the assigned coordinates.
- **a linear tangent segment**: it inserts a linear segment whose length is defined by **Module,** while the linear segment direction is assigned, so that the start direction of the original profile is maintained.
- **a circular tangent segment**: it inserts a circular segment, while the linear segment direction is assigned, so that the start direction of the original profile is maintained.
- **a circular segment on 3d-plane**: it inserts a circular segment, that is defined by the radius and the value of the angle, while its direction is so assigned, that the start direction of the original profile is maintained.
- **insert a coverage line**: it inserts a segment with set length, duplicating the geometry or the last profile line. It is possible to select a coverage line only if the original profile is a closed profile and if it closes with a profile line.

The segment is assigned in the window, if the Cartesian position is active or if you choose to insert a coverage line.

The segment is assigned with the mouse if Mouse positioning is active. If active, it applies the snap to grid.

After selecting the line, a window opens and displays the data completing the assignments required. To insert a line, following data are required:

- **linear segment with given coordinates:**

  X coordinate, Y coordinate:absolute or relative coordinates of the starting point of the linear segment.

  Z coordinate: starting depth of the linear segment. If set in relative mode, the value is applied at the starting depth of the profile.
- **linear tangent segment:**

Module: linear segment length

Apply in 3d: if enabled, it imposes the tangent continuity in the space while starting the profile. The segment direction and the starting deep of the added linear segment are determined by the first segment of the profile.

Get the angle from the profile: if enabled, it imposes the tangent continuity while starting the profile; otherwise it imposes the segment direction, as defined from the **Angle** field value.

Z coordinate: starting depth of the linear segment. If set in relative mode, its value is applied according the starting depth of the profile.

- **circular tangent segment:**

  X coordinate, Y coordinate:absolute or relative coordinates of the starting point of the arc.

  Z coordinate: starting depth of the linear segment. If set in relative mode, its value is applied according the starting depth of the profile.

  This inserts:

- **circular segment on 3d-plane:**

  Radius: radius of the arc

  Followed angle: arc value in degree. This value should be between 1.0° and 90°. If an arc cannot be determined, a linear segment whose length is equal to the **Radius** is defined with continuous tangent at the profile entry. The segment solution is analogue to what applied into setup workings as far as the segment of the profile entry is concerned.

- **as a coverage** :

  Apply coverage at the beginning: if enabled, it allows to insert at the beginning of the profile a (total or partial) segment, covering the last profile segment. In this case the field are editable:

  Module: length of the joined segment. The field is initialized according the length value of the last profile segment. If a value null or higher than the initialized value is set, a total coverage will be obtained.

  Z coordinate: Starting depth of the segment. If set in relative mode, the value is applied according the starting depth of the profile. The depth coordinate is not considered, when the covering segment is an arc, developing on a different plane from xy.

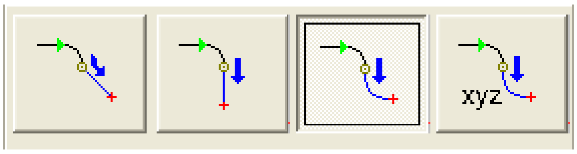
  Apply coverage at the end: if enabled, it allows to insert at the end a (total or partial) covering segment for the last segment of the profile. In this case the fields are editable: the option cannot be selected, if the profile geometry does not allow to insert at the end the covering segment.

  Module: length of the joined segment. The field is initialized according to the value of the first profile segment. If a value null or higher than the initialized value is set, a total coverage will be obtained.

  Z coordinate: final segment depth. If set in a relative mode, the value is applied according to the starting profile depth. The depth coordinate is not considered, when the covering segment is an arc, developing on a different plane from xy.

10.4.15 Apply Exit Point To Profile

This tool applies a linear or circular exit segment to the current profile. It can be selected from the ***Tools->Apply exit point to profile*** menu item and from the Tool Bar by clicking on the relevant icon .
By a graphic palette it is possible to choose what type of segment to enter:



- **a linear segment with given coordinates:** it inserts a linear segment from the profile end point to the point defined by the assigned coordinates.
- **a linear tangent segment**: it inserts a linear segment whose length is defined by **Module,** while the segment direction is assigned, so that the direction of the original starting profile is maintained.
- a **circular tangent segment**: it inserts a circular segment, while the segment direction is assigned, so that the direction of the original starting profile is maintained.
- **a circular segment on 3d-plane**: it inserts a circular segment, that is defined by the radius and the value of the angle, while its direction is so assigned, that the start direction of the original profile is

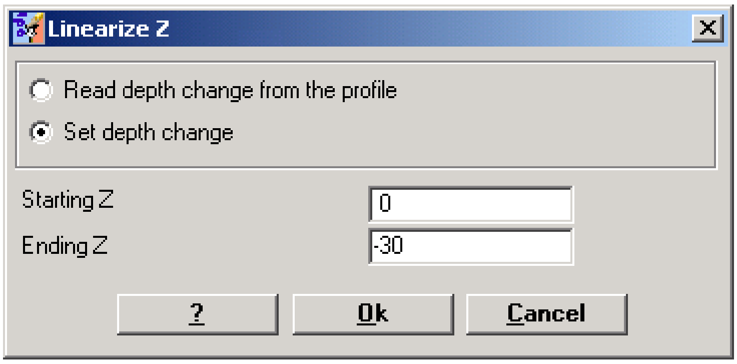
maintained.

- **coverage:** inserts a segment of set length, that duplicates the geometry of the first profile segment. It is possible to select the coverage segment only if the original profile is a closed profile and if it starts with a segment profile.

The segment is assigned in the window, if the Cartesian positioning is active or if a covering segment is chosen.
The segment is assigned with the mouse, if the Cartesian positioning with the mouse is active. If active, it applies the snap on the grid.

10.4.16 Linearize Z

This tool linearizes the variation of the current profile depth. The tool works only on simple profiles with only arcs on the plane yx. It can be selected from the ***Tools->Linearize Z*** menu item and from the Tool Bar by clicking on the relevant icon **.**



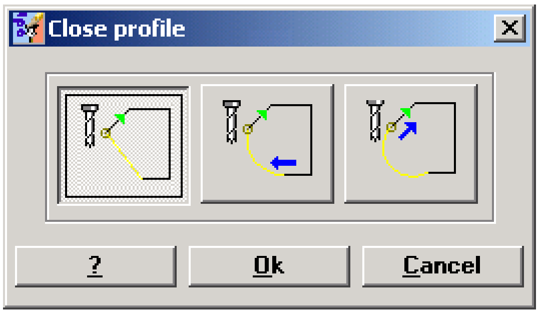
○ Read depth change from the profile
● Set depth change

Starting Z 0
Ending Z -30

- **Read depth change from the profile**: it reads the starting and the end z coordinates assigned to the profile and changes the working depth along the entire path so as to obtain a constant variation.
- **Set depth change**: given the starting and the end Z coordinates, it assigns a Z coordinate to each segment of profile so as to obtain a constant change.

10.4.17 Close the Profile

This tool applies a linear or circular closing segment to the current profile. The profile cannot be closed. The tool can be selected from the ***Tools->Close the profile*** menu item and from the Tool Bar by clicking on the relevant icon .
By a graphic palette it is possible to choose the characteristic of the segment which is entered to close the profile

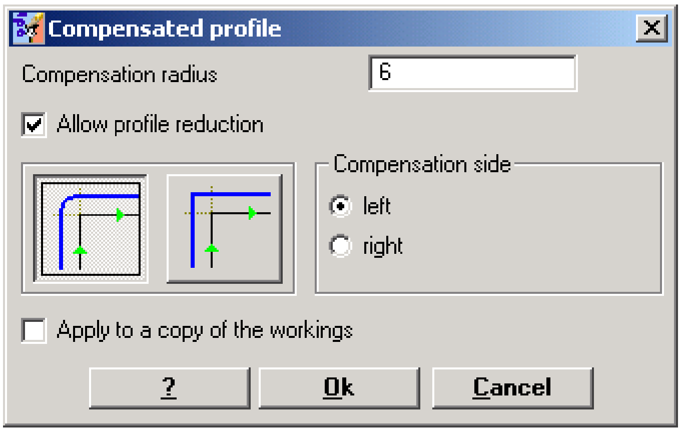


- **with a linear segment**: it closes the profile by inserting a linear segment which connects the last profile point to the setup point.

- **with a tangent arc starting from last segment**: it closes the profile by inserting an arc tangent to the last segment of profile
- **with a tangent arc ending at the first segment**: it closes the profile by inserting an arc tangent to the first segment of profile

10.4.18 Compensated profile

It inserts a new profile achieved by compensating the tool from the current profile. The tool is recalled from the menu ***Costructions->Compensated Profile*** and from Path Bar and Advanced Tool icon. .



Compensated profile

Compensation radius: 6

☑ Allow profile reduction

Compensation side

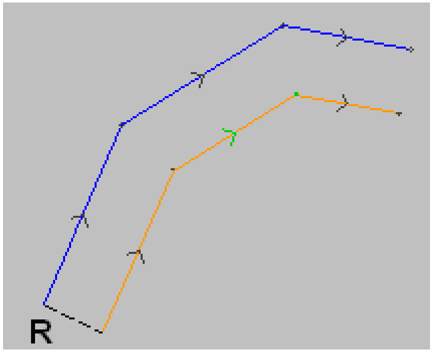
● left

○ right

☐ Apply to a copy of the workings

- **Compensation radius:** this is the absolute compensation value
- **Insert fillets:** it is enabled by selecting the first bitmap. It inserts fillets (smooth round joints) at the intersection of segments. It inserts fillets (smooth round joints) at the intersection of segments
- **Reduce fillets to intersections:** this option is enabled by selecting the second bitmap. It allows not to insert fillets at the intersection of segments, but to leave an edge.
- **Allow profile reduction:** this option enables segments removal in the compensated profile, in comparison to the original one and according to the geometric overall dimensions that exceed the compensation itself.

Example:

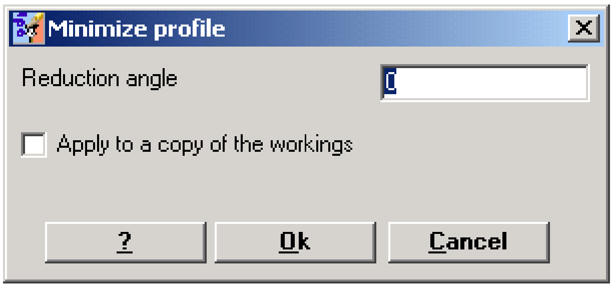


The original profile is external. The compensated profile is internal, because the requested compensation is **on the right side.**.
R represents the compensation radius.
If in the original profile an entry and/or exit segment is set in the original profile, this segment is also assigned in the setup of the correct profile.

10.4.19 Minimize profile

When a profile consists of a great number of consecutive segments it is possible to reduce the number of segments by assigning a **[Reduction angle]** in the window which appears by selecting the ***Constructions->Minimize profile*** menu item and by clicking on the relevant icon in the Advanced Tool and Path Bar. The angle of reduction is expressed in degrees and defines the cone of maximum angle within which the consecutive linear traits are joined. The values set should be between 0,001 ° and 90.
The consecutive linear segments which are enclosed in a maximum assigned angle cone are joined

together.



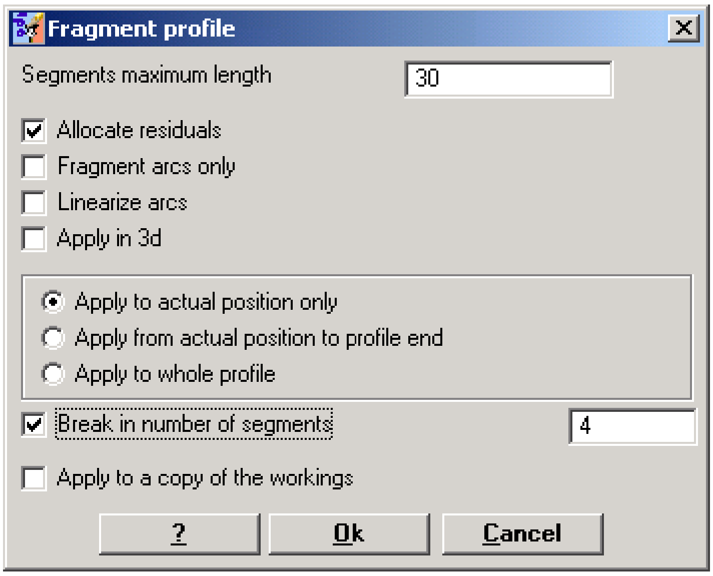
Minimize profile

Reduction angle: 0

□ Apply to a copy of the workings

10.4.20 Fragment profile

It is possible to fragment a profile by selecting the tool from the ***Constructions->Fragment profile*** menu item and from the Advanced Tool and Path Bar by clicking on the relevant icon .
It fragments the segments of a profile into maximum assigned length segments.



Fragment profile

Segments maximum length: 30

☑ Allocate residuals
□ Fragment arcs only
□ Linearize arcs
□ Apply in 3d

● Apply to actual position only
○ Apply from actual position to profile end
○ Apply to whole profile

☑ Break in number of segments: 4

□ Apply to a copy of the workings

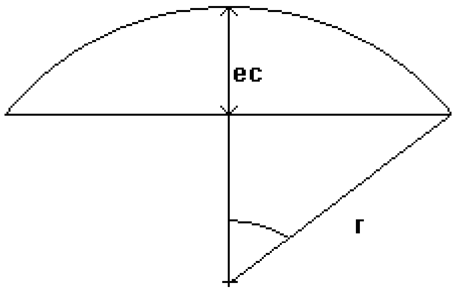
In the dialog box below the following items must be set:

- **Segments maximum length**: it sets the maximum length of profile fragments.The minimum value is positive and equal to 10.0*epsilon of Tpaedi32 resolution.
- **Apply chordal error to the arcs:** if enabled, this option fragments the arcs by applying the *chordal error* set. In this case the maximum length of the segments is applied only to the fragmentation of the linear segments.
- **Allocate residuals**: on each individual segment the tool calculates the number of fragments into which the segment itself is partitioned. The remaining part is allocated on each partition. For example if a linear segment is 52 mm long and a length of fragments equal to 10 mm is assigned, if the item is not selected the linear segment is fragmented into 6 partitions: 5 10 mm-long and 1 (the last) 2 mm-long fragments. Viceversa if the item is selected the linear segment is fragmented into 6 partitions of the same length. The length is recalculated and it will be equal to (52/6) mm
- **Fragment arcs only**: the tool is applied only to the segments of arc
- **Linearize arcs**: it breaks arcs into segments which are converted into linear segments
- **Apply in 3d:** the maximum length of segments is also applied to the depth component
- **Break in number of segments:** it allows to fragment an individual segment into an assigned number of partitions. In this case the **Segments maximum length** parameter setting is automatically calculated. The field only accepts a numeric input ranging between 2 and 99. The item is enabled only if:

- the tool is applied to the current working
- the current working has a line or arc typology and executes a single geometric segment

If in the setup of the original profile some entry and/or exit segments are set, these continue to be assigned directly in the setup and they are not subject to fragmentation.

Splitting an arc



The figure shows the geometric meaning assigned to the value of the chordal error set. A typical chordal error is 0.05 mm. Splitting an arc according to the criteria of the chordal error determines sampling on an arc whose length changes according to the radius of the arc. If the radius increase, the length of the segments increases accordingly.

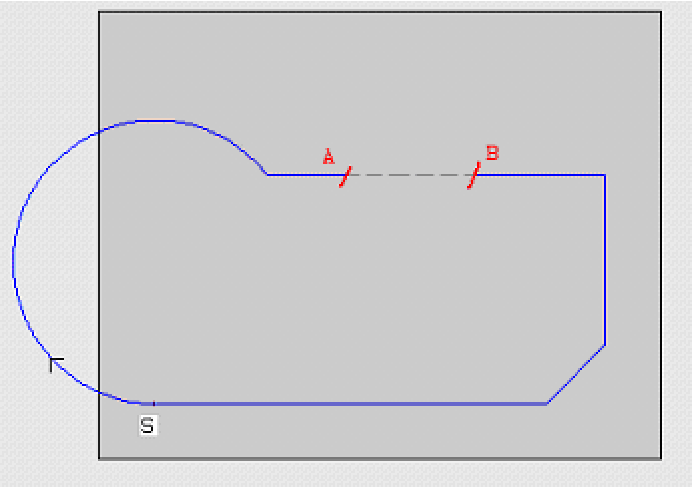
If the option **Chordal error** is selected:

- for each splitting a maximum chordal error equal to 50% of the radius of the arc is accepted
- for an arc a sampling number is resolved anyway and it is not lower than the entire fractions of 45° (of the arc dimension)
- for each splitting precise limits are accepted for the sampling angle, that is calculated. Its minimum value is 1° and its maximum one is 45°.

10.4.21 Disconnect Profile

This tool disconnects the current profile, by assigning cut points. It can be selected from the ***Constructions->Disconnect profile*** menu item and from the Advanced Tool and Path Bar by clicking on the relevant icon .

The tool allows to delete a part of the current profile or split a segment of the profile into two separate segments.



The figure above shows an example of profile:

- (S) indicates the profile starting point
- the arrows indicates a counterclockwise direction
- the example profile is closed

On the profile 2 cut points have been marked and namely: (A) and (B) (the two points can lie on different

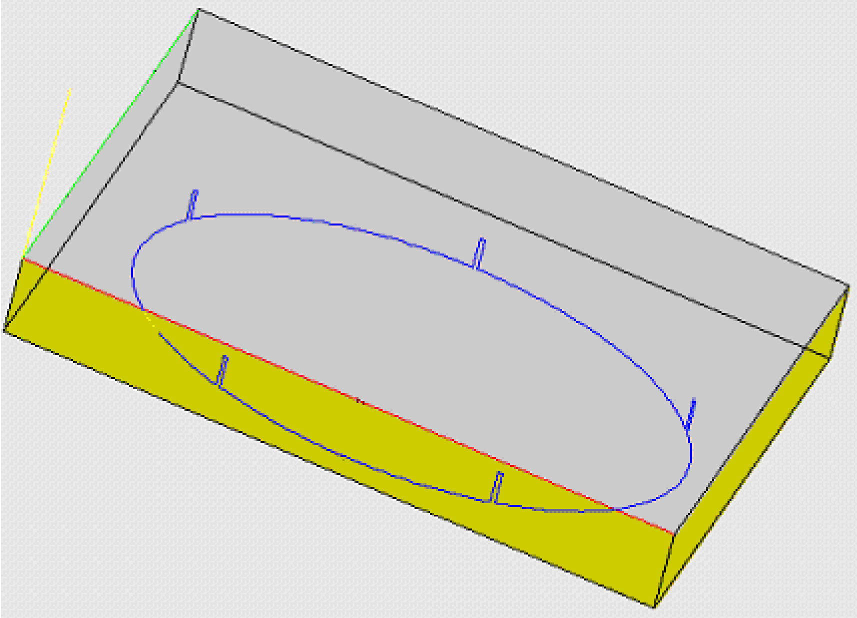
segments).
When the tool is applied, the part of profile between the two points (attention: in the original direction) is deleted.
The original profile is then broken by the tool into 2 profiles:

- the 1st profile starts from (S) to (A);
- the 2nd profile starts from (B) to (S).

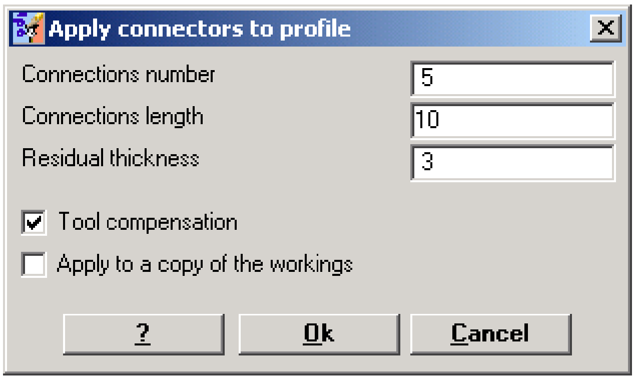
If, on the contrary, it was required to break the concerned linear segment in only one point (example: (A): it was enough not to mark the second cut point.
In this case we have a single profile with one more segment.

10.4.22 Apply Connectors To Profile

To apply connectors to the current profile select this tool from the ***Constructions->Apply connectors to profile*** menu item and from the Advanced Tool and Path Bar by clicking on the relevant icon .
A connector results from the fragmentation of an original segment of profile into a short segment, executed at such depth that a residual thickness is left in the piece.
Typically, this tool is used in case of closed profiles, where milling depth exceeds piece thickness (feedthrough profile). In these cases, the direct execution of the profile would cause a part of the piece (the inside of the milled area) to detach, with possible fall of the profile itself when milling.
Then, the application of connectors allows to leave connectors along the milled area thus avoiding the above-mentioned detachment.
Next Figure shows an example of closed feedthrough ellipse, with 5 connectors applied



The connections allocation on the profile can be executed with the aid of the **Manual connections allocation** command by selecting with the mouse the profile point where to apply the connection and pressing the **[Escape]** key or selecting the **Automatic connections allocation** command to terminate the insertion. Tpaedi32 distributes homogeneously the number of connections assigned by the user on the profile, by keeping constant the distance of connections along the profile.
In both cases a second setting window is displayed:



Apply connectors to profile

Connections number: 5

Connections length: 10

Residual thickness: 3

☑ Tool compensation

☐ Apply to a copy of the workings

- **Connections number**: it sets the number of connections to allocate automatically. The field accepts a numeric input ranging from 2 to 255
- **Connections length**: it sets the connection length (in the xy plane of the face). The value cannot be lower than the resolution epsilon set by the machine manufacturer in configuring the software
- **Residual thickness**: it sets the thickness which the tool leaves in the piece while creating the connection. This field accepts positive values at least equal to the software resolution epsilon.
- **Tool compensation**: it changes the actual connection length so as to keep into account the tool diameter. The connection is made a little narrower to allow the tool to enlarge it up to reach the required length, during execution.

If in the setup of the original profile some entry/and exit segments are set, these continue to be assigned directly in the setup.

10.4.23 Apply feed in Z

If modifies the current profile, by inserting consecutive steps until an assigned final depth.The tool is recalled from menu ***Constructions->Apply feed in Z*** and from Advanced Tool and Path Bar from the icon .

The tool works only on simple profiles. It is typically used in profiles, that should be executed at a depth measure, that cannot be obtained with one only passage.

In a dialog box parameters are set, as follows:

- **starting Z**: it displays the starting profile depth. It is not editable.
- **ending Z**: finale depth of the recursive profile development
- **Z feed**: sets of how much should change the depth at every development
- **Manage profile inversion**: this option is only activated, if the command is applied to a closed profile. If selected, the option inverts the alternating profile execution at every depth change. If not selected, the inversion is excluded. If the profile is not closed, the profile inversion is managed in any case.
- PROFESSIONAL **RfCompensation: change of side:** the command is not applied in case of closed profile where the inversion of the profile is not requested. If enabled, to each additional pass an inversion of compensation side (from the right to the left or viceversa) is added. This option is managed only if it is enabled by the machine Constructor during the configuration of Tpaedi32 . Only in **Professional Mode** available.

The tool is disabled, if:

- there are not any programmed workings
- the current working does not belong to a profile

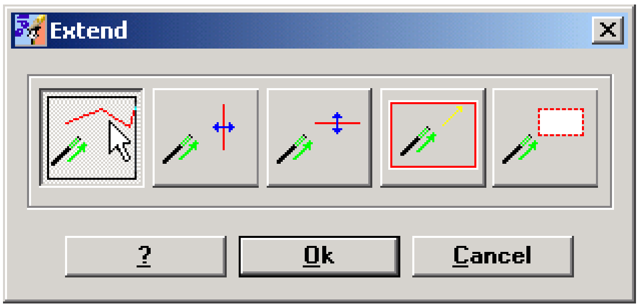
If in the setup of the original profile some entry/and exit segments are set, these continue to be assigned directly in the setup.

10.4.24 Extend

This tool extends the active segment belonging to a profile up to intersect a selected delimitation element. In case of several intersection solutions, that nearest to the original segment is considered valid. The tool can be selected from the ***Constructions->Extend*** menu item and from the Advanced Tool and Path Bar by clicking on the relevant icon .

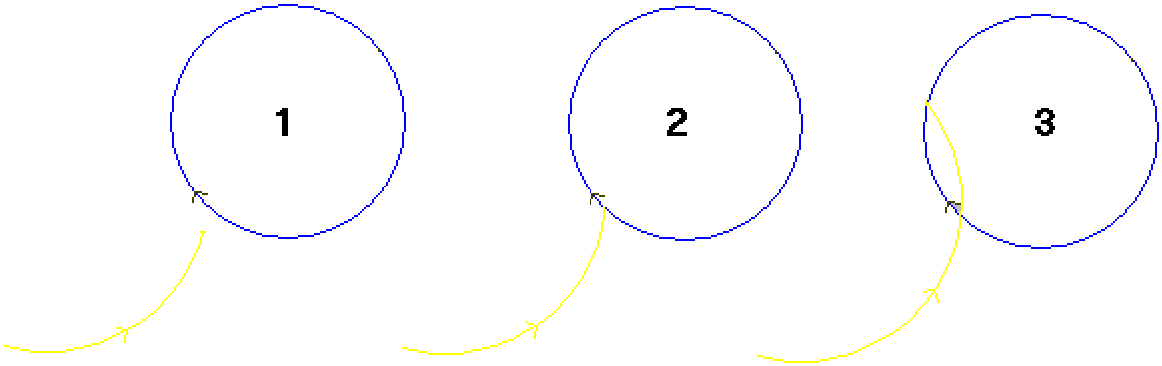
To apply the tool the active segment must belong to a profile and have arc typology, but it must not be a circle, or line typology of not null length.

Use a graphic palette to choose the selection mode of delimitation element:



- **extension to a program element:** delimitation element is set up by programmed working and it does not include punctual and setup workings, selected by mouse pointer inside a graphic area. Section is extended to intersect the profile closest to selection point. If more intersection solutions exist, the closest to the point of origin is valid.
- **extension to a vertical line:** delimitation element is set up by a vertical line, directly located in the graphic area.
- **extension to a horizontal line:** delimitation element is set up by a horizontal line, directly located in the graphic area.
- **extension to intersection with face:** delimitation element is set up by face overall rectangle. Segment is extended to intersect face side.
- **extension to intersection with rectangle:** delimitation element is set up by a rectangle. Rectangle overall dimensions are directly set up in the graphic area.

The Figure below shows the case of two applications of the tool.



The arc represents the segment which must be extended up to the point of intersection with the circle

- Figure **1** matches the starting situation
- Figure **2** corresponds to the first application of the Extend command
- Figure **3** corresponds to the second application of the Extend command

10.4.25 Repeat profile

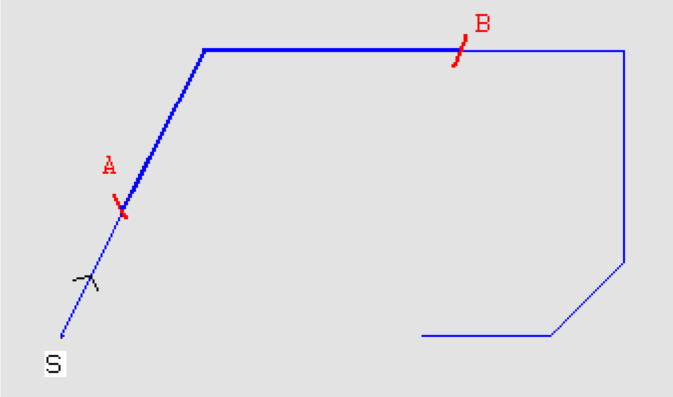
It repeats part of the current profile. The segment to repeat is defined between two cut points. The tool can be selected from ***Constructions->Repeat profile*** menu and from the Advanced Tool and Path Bar by clicking on the relevant icon .

The profile is obtained as a reply of the segments between the two cut points and it is opened with a copy of the following setups:

- original one, if the profile is not open; otherwise
- reference one, (as assigned in Customization of Tpaedi32);

The tool can work directly on the current profile or on a copy of the same.

The tool allows to select a part of the current profile



In the figure it is shown a profile example:

- (S) indicates the profile starting point;
- the arrow indicates a counterclockwise direction
- the profile is a open profile.

On the profile 2 cut points are indicated and namely (A) and (B). The two points can lie on different segments..

When the tool is applied, the part of the profile between the two points, (attention: in the original direction) is selected.

If the tool needs to work on the current profile, the parts of the profile are eliminated

- from (S) to the point (A);
- from (B) to the end of the profile.

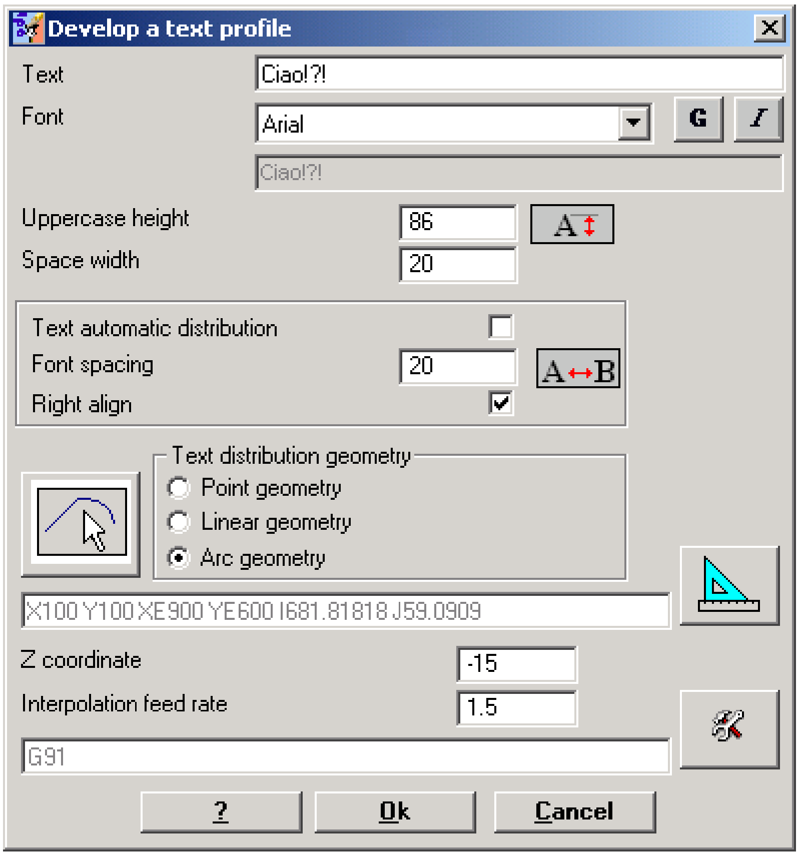
If the tool needs to work on a copy of the current profile, a new profile is added and assigned from the point (A) to the point (B).

10.5 Advanced Tools In Face Program

10.5.1 Text Generation

PROFESSIONAL

This tool allows to enter text into the face program, directly in form of profiles. It can be selected from the ***Constructions->Develop a text profile*** menu item and from the Advanced Tool and Path Bar by clicking on the relevant icon



Text: Ciao!?!

Font: Arial

Ciao!?!

Uppercase height: 86

Space width: 20

Text automatic distribution □

Font spacing: 20

Right align ☑

Text distribution geometry

○ Point geometry

○ Linear geometry

● Arc geometry

X100 Y100 XE900 YE600 I681.81818 J59.0909

Z coordinate: -15

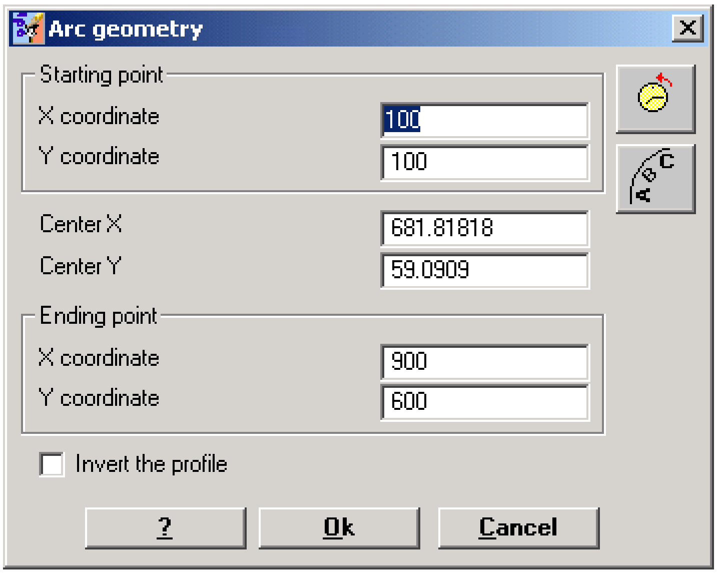
Interpolation feed rate: 1.5

G91

- **Text**: characters to enter
- **Font**: type face. The list makes all installed True Type characters available. The **[G]** and **[I]** buttons format the text respectively in bold and in italic.
- **Uppercase height**: it sets the height of the A letter in units of measurement of the piece
- **Space width**: it sets the width assigned to spaces, if any, in the Text message
- **Text automatic distribution**: it is applied only if the text is distributed along a segment (arc or line segment). The starting point of the message is displaced so as to terminate the message exactly on the segment end point
- **Font spacing**: it defines the space between the individual characters of the message (inter-character spacing). It is an option alternative to automatic distribution.
- **Right align**: it right aligns the text. The selection is only possible if the automatic distribution of the individual characters of the message is not enabled
- **Text distribution geometry**: text distribution can be assigned with reference to:
  - a **point** by assigning the X, Y writing start position and the text inclination angle
  - a **linear segment** with X, Y writing start position and segment end position, with possibility to apply the segment geometry inverted
  - a **circular segment** with X, Y writing start position and arc end position, center and rotation, with possibility to apply the text inside or outside the arc and the segment geometry inverted

The bitmap allows to acquire the text distribution linear or circular segment from a program segment. Once the segment has been selected the text distribution geometry check box is updated to the type of segment selected.

The left Figure shows the arc assignment window.



Arc geometry

Starting point

X coordinate 100

Y coordinate 100

Center X 681.81818

Center Y 59.0909

Ending point

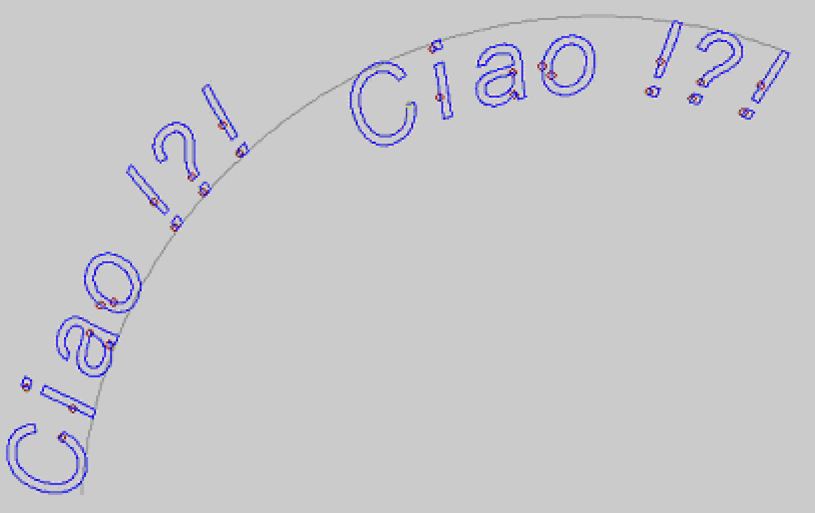
X coordinate 900

Y coordinate 600

□ Invert the profile

The inserted profiles are assigned with technological parameters as defined by the Technology button (working code, technology, working properties). See Chapter ***Workings->Profile->Assign Technology***.

Let us see an example:



The Figure above shows an example of generation of 2 equal writings, with selection of no automatic text distribution along a circular segment in clockwise direction.

The left writing has been generated with no selection added.

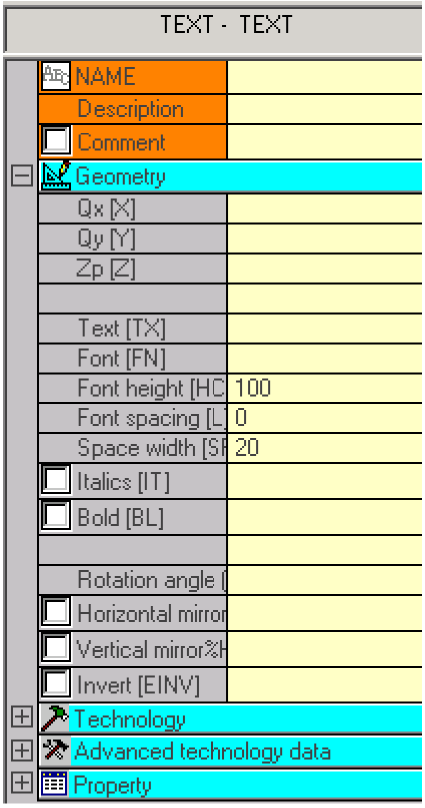
The right writing has been generated with selection of:

- right alignment
- text inside the arc

It is also possible to enter texts by recalling dedicated macros in the list of workings.

Let us see an example:

let us select the TEXT working in the FOR MILLING group:



TEXT - TEXT

NAME	
Description	
☐ Comment	
⊟ Geometry	
Qx [X]	
Qy [Y]	
Zp [Z]	
Text [TX]	
Font [FN]	
Font height [HC	100
Font spacing [L]	0
Space width [SF	20
☐ Italics [IT]	
☐ Bold [BL]	
Rotation angle [	
☐ Horizontal mirror	
☐ Vertical mirror%H	
☐ Invert [EINV]	
⊞ Technology	
⊞ Advanced technology data	
⊞ Property	

The working allows to assign:

- **Text**: text to enter
- **Font**: selection of the type face to apply to the text to enter (the list makes all installed True Type characters available)
- **Font height**: it sets the height of the A letter (in units of measurement of the piece)
- **Font spacing**: space between consecutive characters
- **Space width**: it sets the width assigned to spaces, if any, in the Text message
- **Bold and Italic**: to apply these formats to text
- **Qx,Qy Zp**: starting text coordinates and text depth
- **Horizontal mirror and Vertical mirror**: they enable the required mirror setting
- **Rotation angle (°)**: it sets the text inclination angle
- **Invert**: it enables inversion in the profile execution
- **Technology**: tool selection, tool radius compensation, speed,…
- **Working property**: it sets Level, O Field, Construct …

10.5.2 Profiles Cut

PROFESSIONAL

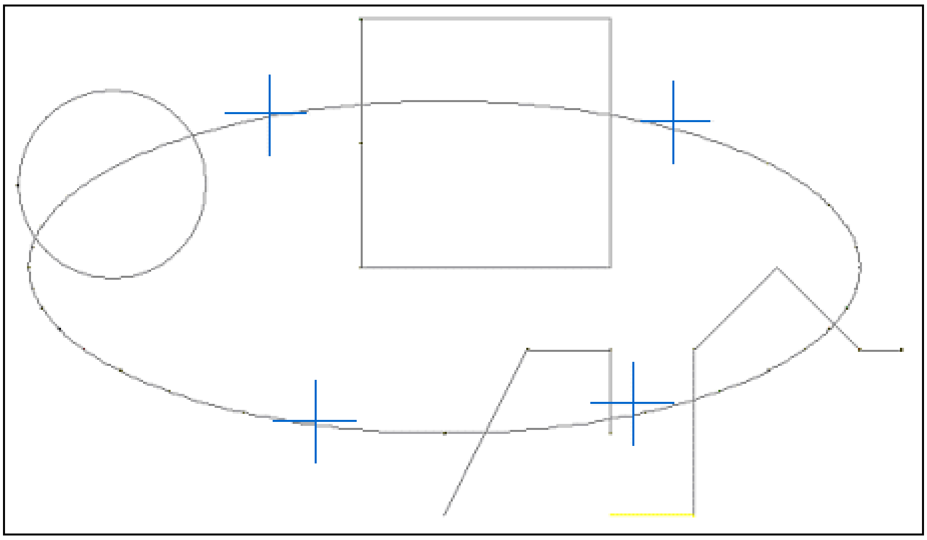
This tool allows to cut a profile at an edge defined by one or more profiles. It can be selected from the ***Constructions->Tools->Profiles cut*** menu item and from the Advanced Tool and Path Bar by clicking on the relevant icon

The profiles defined as cutting edges are the following:

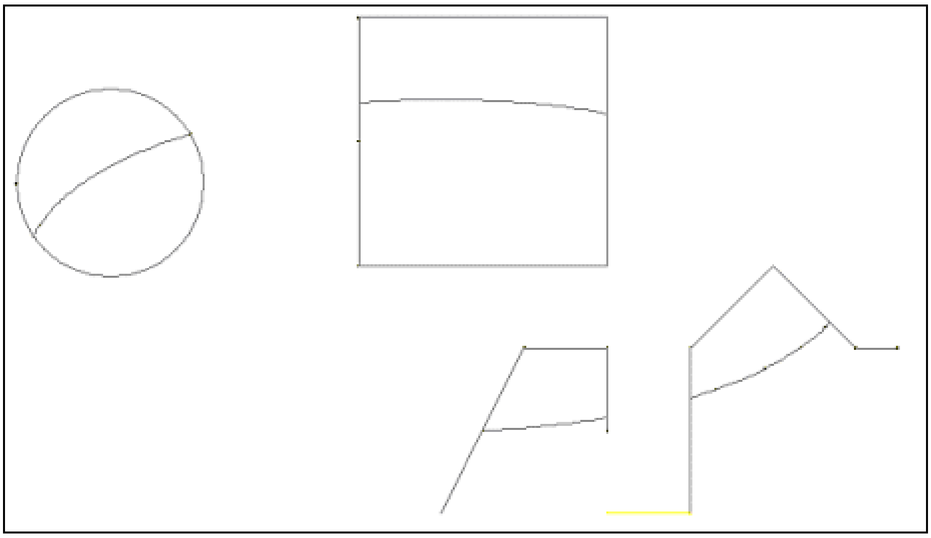
- selected profiles (if any), or
- all face profiles.

The cutting edges are assigned with the mouse, by following the instructions provided in the Commands Status area.

The Figure shows a program which consists of intersecting profiles. The 4 crosses identify the cutting edges



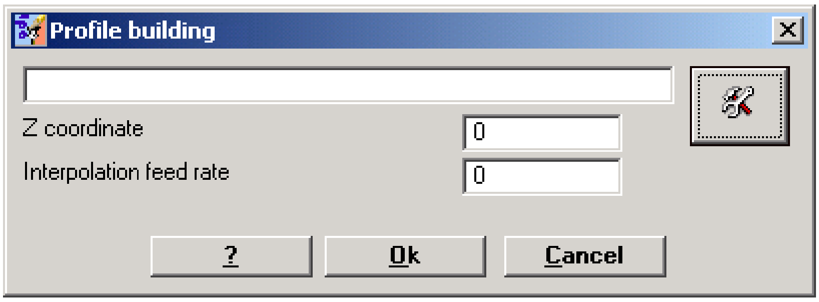
After the application of the tool the program appears like in Figure



10.5.3 Profile Building

PROFESSIONAL

This tool allows to build a new profile by selecting one or more programmed segments of profile. The selected segment must have a point of intersection with the previous segment of profile.
It can be selected from the ***Constructions->Profile building*** menu item and from the Advanced Tool and Path Bar by clicking on the relevant icon ⎍.
In case of piece-face the tool is only enabled with the 2d face view active and operates only on the profiles applied to the face in current view.
It is requested to assign technological parameters for the new profile:



Profile building

Z coordinate: 0

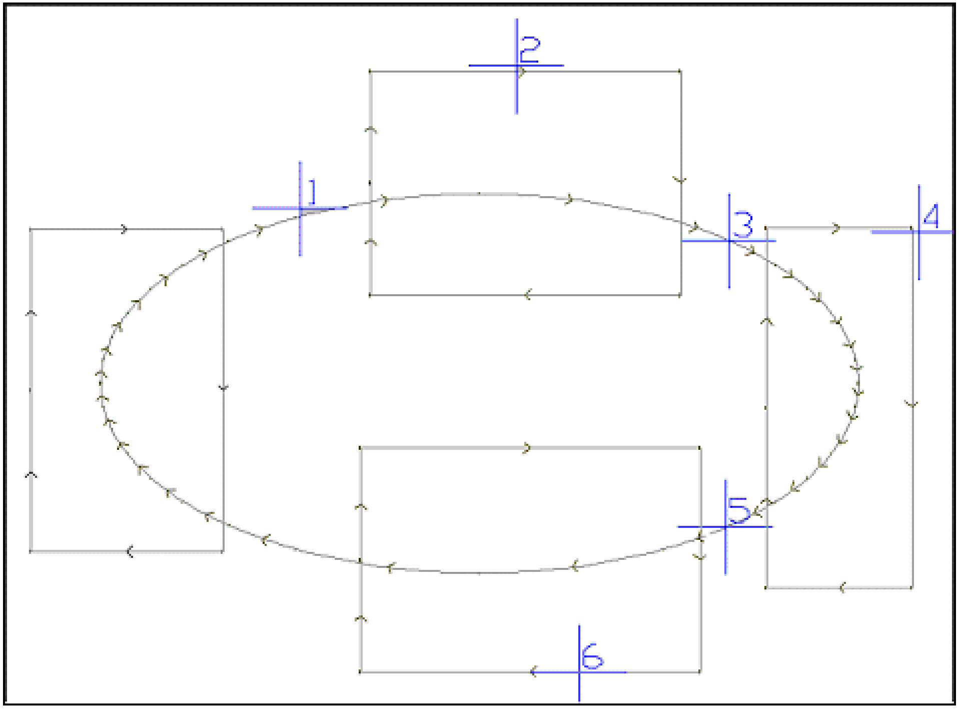
Interpolation feed rate: 0

Technological parameters are assigned by the Technology button (working code, technology, working properties). See Chapter ***Workings->Profile->Assign Technology.***

- **Z coordinate**: it sets the profile depth coordinate (Z) assigned on the setup
- **interpolation feed rate**: it sets the profile execution speed

The segments which belong to the new profile are assigned with the mouse, by following the instructions provided in the Commands Status area.

Let us see an example:



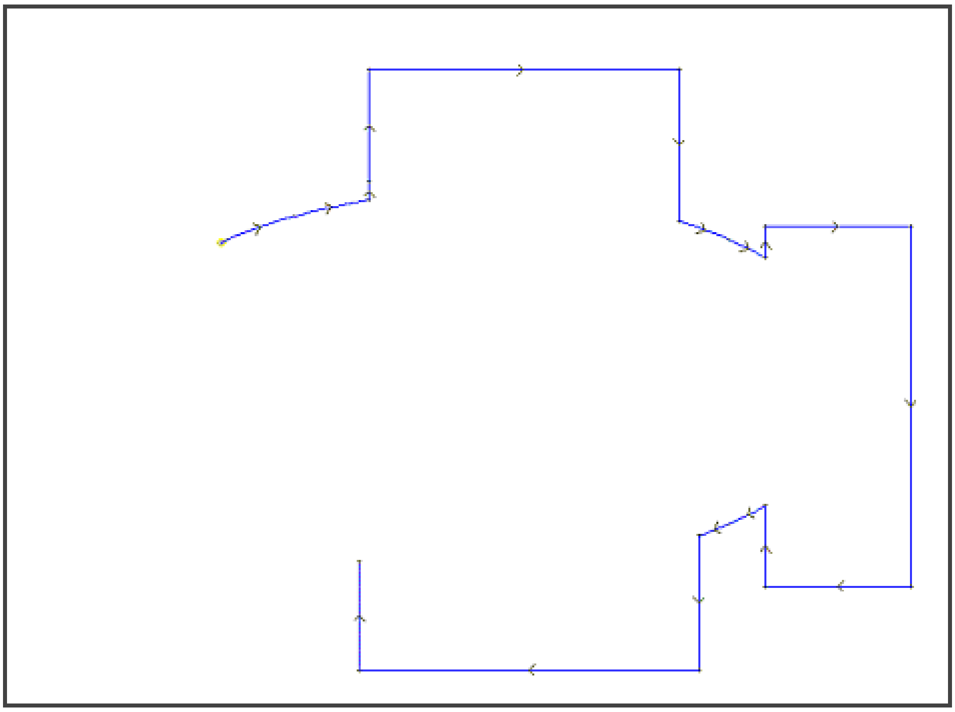
The Figure above shows a program which consists of intersecting profiles. The 6 crosses identify the parts the profile under construction is made of. Crosses are numbered and indicate the order by which the selected parts are added to the new profile:

- cross 1 marks the starting point. The segment of profile which is nearest to the position clicked with the mouse is searched
- cross 2 chooses how to continue the profile. The segment of profile which is nearest to the position clicked with the mouse and which continues geometrically the segment already selected as segment (1) is searched. The geometric continuity can also determine the inversion of segment (1) and/or segment (2), with respect to the direction of execution of original profiles
- cross 3 chooses how to continue the profile. The segment of profile which is nearest to the position clicked with the mouse and which continues geometrically the segment already selected as segment (2) is searched. Now the geometric continuity can cause the inversion of the only segment (3), with respect to the direction of execution of original profiles

- ..
- up to cross 6.

Once all segments have been selected, the command is confirmed by clicking the right mouse button. At this point the acquired positions are processed and a new profile is added to the face program, without in any way modifying original profiles.

In next Figure we can see the profile built by following the above-described instructions:



10.5.4 Generate Spline From Polyline

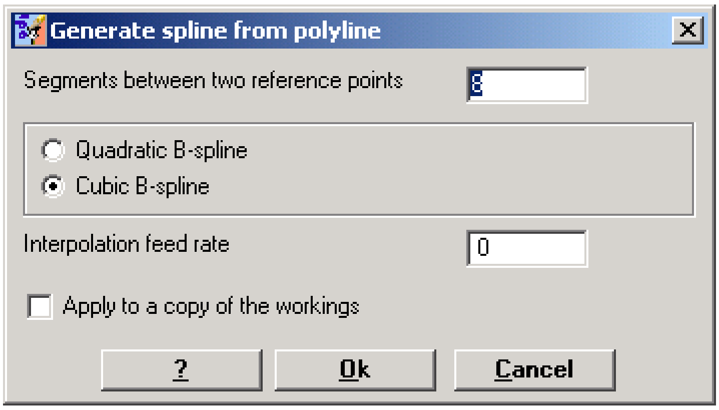
PROFESSIONAL

This tool allows to generate a new profile at the edges defined on already programmed profiles.

For each identified profile the tool uses vertices as reference points for a curve which is open and closed exactly as the original profile.

The generated curve passes through the first and the last profile point and approaches the other points. The arcs of the original profile are manipulated as line segments. Possible circles are deleted from the profile for the evaluation of reference points. Moreover the original profile cannot assign arcs in a plane different from the xy plane.

The spline curve is generated by a sequence of linear segments. The approximation level is assigned by the number of linear segments which are displayed between two reference points. It can directly operate on the current profile or on a copy of this last. The tool can be selected from the ***Constructions->Generate spline from polyline*** menu item and from the Advanced Tool and Path Bar by clicking on the relevant icon



Generate spline from polyline

Segments between two reference points: 8

○ Quadratic B-spline

● Cubic B-spline

Interpolation feed rate: 0

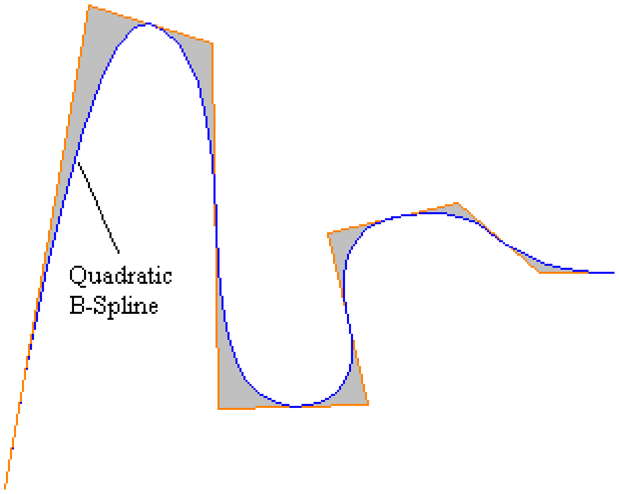
□ Apply to a copy of the workings

- **Segments between two reference points**: number of linear segments which are displayed between two reference points (values between 8 and 50 are accepted)
- **Quadratic B-spline**: spline curves are calculated with quadratic functions
- **Cubic B-spline**: spline curves are calculated with cubic functions
- **Interpolation feed rate**: it sets the spline curve execution speed

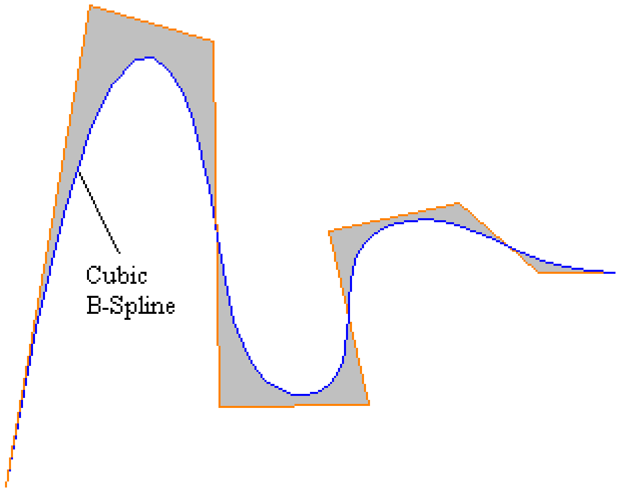
The original profile must be assigned on a minimum number of segments:

- **3** in the case of quadratic B-Spline
- **4** in the case of cubic B-Spline.

Here is an example of spline curve generation by selecting the **quadratic spline** item



From the same profile by selecting the **cubic spline** item the following curve is generated:



In the Figures above the grey areas included between the original curve and the spline curve are only displayed to highlight the distance between the two curves.

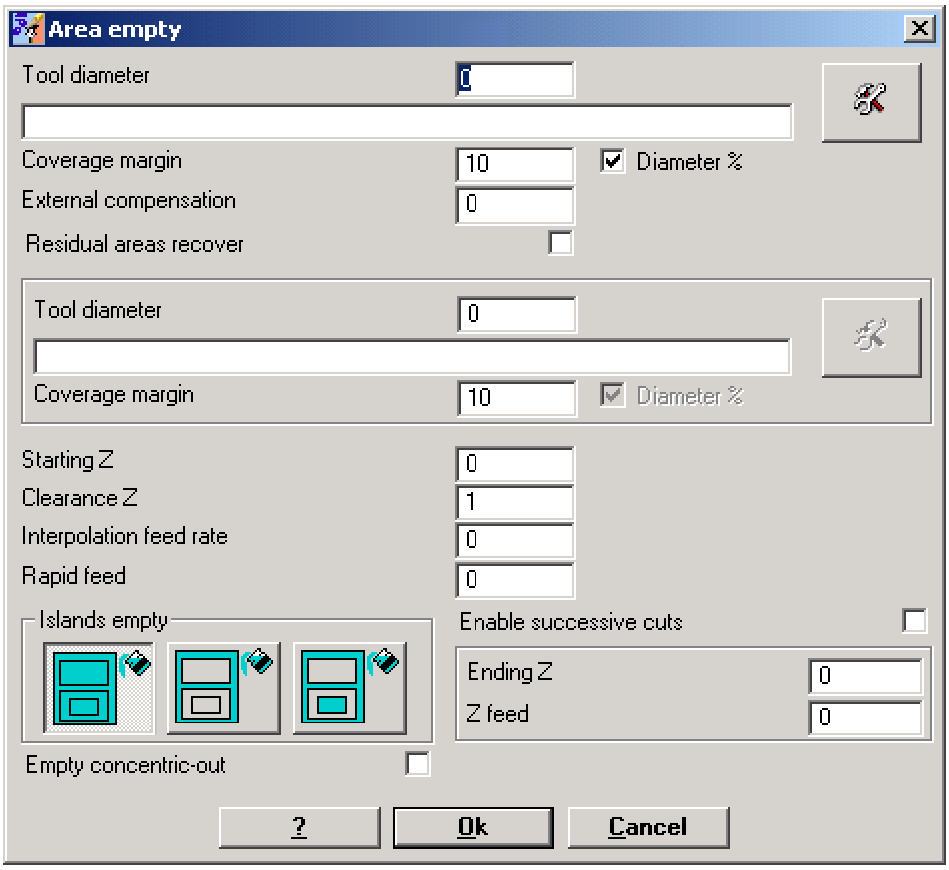
10.5.5 Area Empty

PROFESSIONAL

This tool allows to empty an area defined by a closed profile, by directly inserting emptying profiles into the face. It can be selected from the ***Constructions->Area empty*** menu item and from the Advanced Tool and Path Bar by clicking on the relevant icon .

In the case of piece-face the tool is only enabled with the 2d face view active and operates only on profiles applied to the face in current view.

It does not take into account the profiles which have the Emptying profile parameter active. This parameter is managed in setup workings to mark profiles generated during emptying.



Area empty

Tool diameter 0

Coverage margin 10 ☑ Diameter %

External compensation 0

Residual areas recover □

Tool diameter 0

Coverage margin 10 ☑ Diameter %

Starting Z 0

Clearance Z 1

Interpolation feed rate 0

Rapid feed 0

Enable successive cuts □

Ending Z 0

Z feed 0

Empty concentric-out □

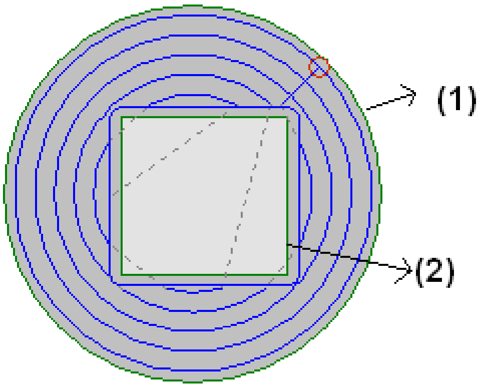
- **Tool diameter**: it assigns the tool diameter. By the Technology button . the setup working code to be used for the emptying procedure is selected. See Chapter ***Workings->Profile->Assign Technology***.
- **Coverage margin**: it indicates the amount by which the successive passes of the tool overlap. The relevant field value can be expressed in absolute (mm) o in **%** of the tool **diameter** which is set
- **External correction**: it indicates the amount of deviation from programmed profile on first correction. Value is expressed in piece measuring units (mm or inch) and subtracted from radius compensation value used for the first correction. Field accepts a positive value >= (higher than or equal to) (epsilon coordinates * 10) and <= (lower than or equal to) radius compensation. For example, with (*epsilon* coordinates) = 0.001 mm and program measuring unit in [mm], field minimum value is equal to 0.01.
- **Residual areas recover**: if selected it enables the items of the next box. It allows to use two different

emptying technologies:

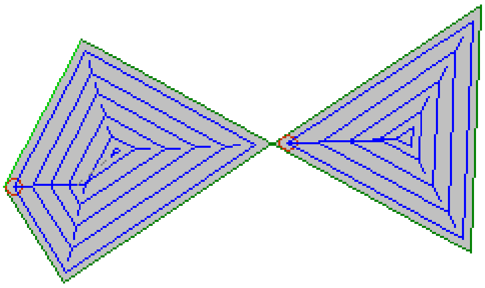
- the above-assigned default technology
- a second technology to be used in the case some areas have not been completely emptied by the default tool. For this purpose the now assigned tool must have a diameter lower than the default one, since it must operate in areas of lower overall dimensions. The technology of the recovery tool is assigned in the same way as the default one

- **Starting Z**: it sets the emptying depth coordinate which matches the coordinate of the first pass, in case several passes are required
- **Clearance Z**: it sets the safe clearance height coordinate that the tool retracts to for movements in air
- **Interpolation feed rate**: it sets the speed of movements during emptying
- **Rapid feed**: it sets the speed of xy movements executed in air. The lowering segments from Clearance Z to the working depth are executed at the same tool entry speed, as previously set up (button Technology). If no lowering tool speed is set, the lowering segments are executed at the same movement speed in the air.
- **Enable successive cuts**: it enables the repetition of the emptying cycle in case of several cuts, executed at different depths
  - **Ending Z**: it is the final depth to reach during the execution of the last cut
  - **Z Feed**: it is the depth variation in case of successive cuts
- **Islands empty**: bitmap representation from left to right:
  - **Ignore islands**: it empties the inside of the area defined by the profile, by ignoring inner closed contours
  - **Empty outside**: it empties the inside of the area by excluding inner closed contours
  - **Alternate empty**: it empties the inside of the area defined by the profile. If it meets a closed contour inside of it, it stops the emptying process until it meets another closed contour
- **Empty concentric-out**: if active, it needs to execute the emptying process from the inside of the area. This item can be selected only if [Ignore islands] is active.

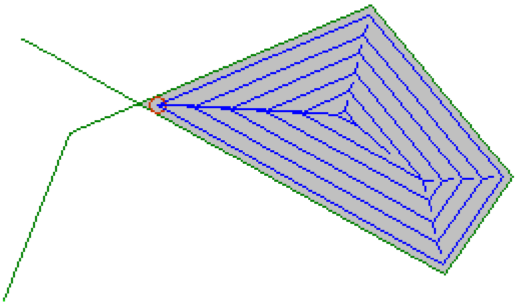
It is possible to execute emptying operations also in the form of complex working.



The figure represents the emptying of a circle (1) with rectangular island (2). The area between the two profiles is emptied according a profile that carry on towards the inside with successive reductions. The emptying profile is interrupted in correspondence of the rectangular island . The tool goes up and moves in the air above the island (outlined lines), going down to the working quote in the area to empty.



The figure represents the emptying of a profile, which generates more closed areas and each of those is emptied indipendently.



The figure represents the emptying of a profile, which is not closed. Emptying check is made on existing closed areas.

10.5.6 Create Fictive Face From Geometry

PROFESSIONAL

The command is enabled in face view with face program not empty. This tool can be selected from the ***Apply->Create fictive face*** menu item.

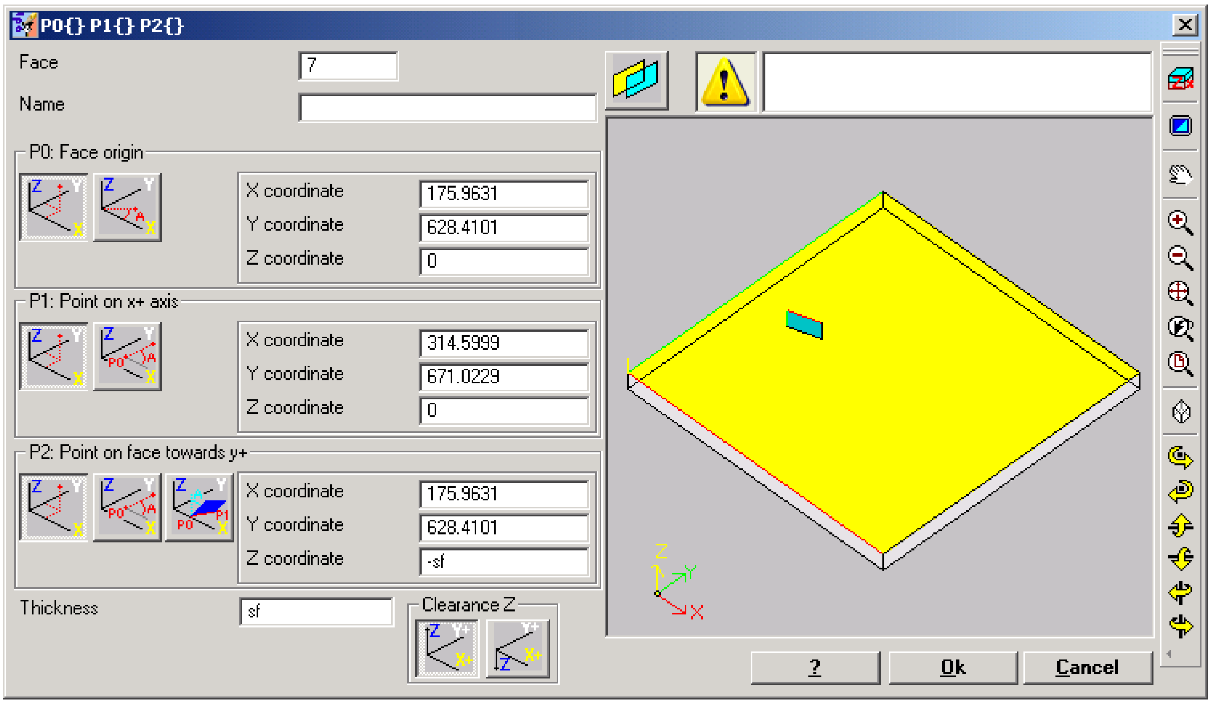
In the case of piece-face:

- the command is not active if the current working is applied in automatic face
- the command is only enabled if the 2d face view is active, with face view corresponding to the application face of the current working

It is about a tool for simplified creation of fictive faces, on the basis of a geometric entity composed of linear segments already programmed on the face.

If a linear segment, belonging to an oriented profile, is found, it is requested if you want to **orient the face according to the profile**. An affirmative reply creates an inclined face towards the vertical direction face, a negative reply creates a vertical face towards the face.

Once individual segments have been assigned with the mouse, the following fictive face setting window is displayed:



P0{} P1{} P2{}

Face: 7

Name:

P0: Face origin

X coordinate: 175.9631

Y coordinate: 628.4101

Z coordinate: 0

P1: Point on x+ axis

X coordinate: 314.5999

Y coordinate: 671.0229

Z coordinate: 0

P2: Point on face towards y+

X coordinate: 175.9631

Y coordinate: 628.4101

Z coordinate: -sf

Thickness: sf

Clearance Z

Fields are preset on the basis of the values read by the identified linear segment.

The insertion of the fictive face into the list of program faces takes place after confirming the exit from the window and assessing the geometric correctness of the face.

Once the fictive face has been entered, it is possible to carry on with this command by indicating another linear segment or exit the command window by pressing the **[ESCAPE]** key.

> *A face entered by this procedure is in no way related to the linear segment used for its setting.*
> *It is possible to change or delete a segment, without consequent modification or automatic deletion of the face itself.*

10.5.7 Workings Which Apply Geometric Transforms

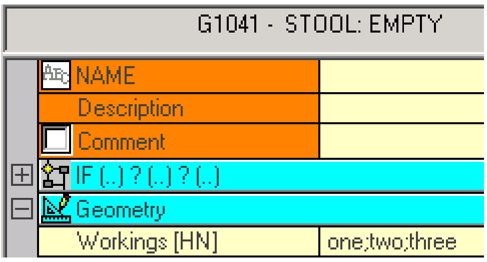
PROFESSIONAL

To understand what is meant by workings which apply geometric transforms and how they are used let us examine the following example.
Let us suppose to have to execute the emptying of a closed area. It is possible:

- to apply the Emptying tool directly to the profile. See Chapter ***Tools->Advanced tools in face program->Area Empty***. In this mode the emptying profile generated will not accept possible changes made to the original profile. In particular the emptying operation does not take into any account possible parametric programming of the original profile
- to save the profile in a subroutine and apply the transform by recalling a complex emptying code. See Chapter ***Workings->Types of workings->Subroutine***. This possibility allows to have the parametric profile, which can be modified, and consequently adapt the emptying procedure: in any case it is necessary to use a subroutine.

The workings which apply geometric transforms combine the positive characteristics of the two above-listed procedures: they apply the Emptying tool directly to a profile, without passing through a subroutine, but they leave the possibility to change the original profile and, automatically, also the emptying process.
In the Workings Bar 3 workings have been defined and namely:
STOOL it applies general transforms (translation, mirrors, rotation, scale, repetitions)
STOOL EMPTY it executes emptying operations
STOOL SPLINE it generates Spline curves

To which workings do complex workings apply?
Complex workings can operate on all workings which have the **Name** (or **N Property)** set at the **Workings** item. In fact, a Name or N Property can be assigned to each working. Several workings can have the same Name.



G1041 - STOOL: EMPTY

NAME	
Description	
Comment	
⊞ IF (..) ? (..) ? (..)	
⊟ Geometry	
Workings [HN]	one;two;three

In the example in the Figure above the complex emptying working is applied to all workings, whose **Name** (or **N Property)** is "two" or "three" or "one".

If the parametric programming takes place in face 0, the only workings assigned to the same face by the complex transform code are accepted. In any case, comment, logical or complex workings, for which it is not possible to create exploded views, are excluded. If the transform needs an emptying operation or a generation of Spline curves the only concerned workings are works on profile.

10.6 Overall Program Tools

10.6.1 Apply technology

This tool which can be selected from the ***Apply->Apply Technology*** menu item is similar to the Apply setup to Profile tool. It applies technology (technological parameters) to point workings and open profiles, that is without opening setup or headed with geometric setup.
The type of setup or point code to be assigned is selected in a window where all workings of selected typology, made available by the software, are displayed.
All technological parameters are set in the standard Assign Technology window. See Chapter ***Workings->Profile->Assign Technology.***
No technology can be assigned to open profiles or profiles headed with geometric setup or geometric points, if defined within a complex working.

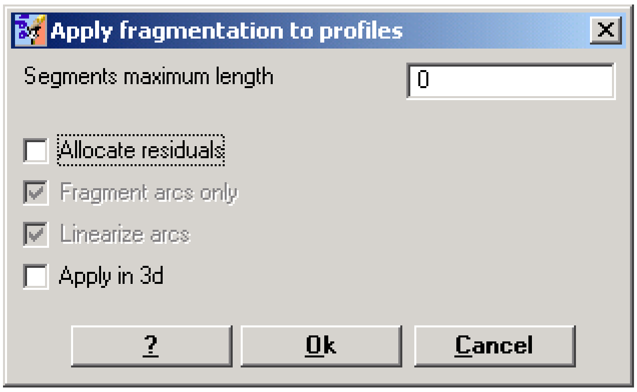
In a punctual working the parameter *Diameter* is assigned according to the following rules:

- If the working of punctual geometric code has no diameter value set, **substitution** is made
- If the working of punctual geometric code has a diameter value set, **substitution is not** made

10.6.2 Apply Reduction or Fragmentation To Profiles

The tool which can be selected from the ***Apply->Apply reduction to profiles*** menu item is similar to the Minimize profile tool. It reduces the number of segments by assigning a **[Reduction angle]**. The consecutive linear segments which are enclosed in a maximum assigned angle cone are joined together.
Profiles defined within a complex working cannot be minimized.

This tool which can be selected from the ***Apply->Apply fragmentation to profiles*** menu item is similar to the Fragment profile tool.
It fragments the segments of a profile into maximum assigned length segments. Fragmentation concerns the arcs only, where fragmented segments can be linearized. The command does not fragment the profiles defined within a complex working.



Segments maximum length 0

□ Allocate residuals
☑ Fragment arcs only
☑ Linearize arcs
□ Apply in 3d

In the dialog box below the following items must be set:

- **Segment maximum length**: it sets the maximum length of profile fragments
- **Apply chordal error to the arcs:** if enabled, this option fragments the arcs by assigning the *chordal error* set.
- **Allocate residuals**: on each individual segment the tool calculates the number of fragments into which the segment itself is partitioned. The remaining part is allocated on each partition.
- **Fragment arcs only**: the tool is applied to the only segments of arc
- **Linearize arcs**: it breaks arcs into segments which are converted into linear segments
- **Apply in 3d**: it evaluates the length of the segment in the xyz space, otherwise only in the xy plane

10.6.3 Apply connection to profiles

This tool which can be selected from the ***Apply->Apply connection to profiles*** menu item is similar to the Connect subsequent profiles tool
For each concerned face, profiles are connected with check of geometric continuity between the starting and the end points by also assessing the **profile inversion** if enabled. The tool identifies the starting connection working with the first (isolated or not) setup or profile segment working. To ensure that the distance of connection is also valued on a component of depth (Z axis) you need to select the option in **Apply in 3d**.
Profile connections cannot be created within a complex working.

11 Parametric Programming

11.1 Introduction

Program assignments usually enable a parametric setting.
Let us consider for example program variables of "o" and "v" type. They are numeric variables whose set field can generally assign a number or a numeric expression.
A greater specialization is required for "r" variables. The “r” variable type is not fixed but it can be assigned between two numeric-type variables (Double and Integer) and one non-numeric-type variable (String).
The Double type configures the "r" variable in the same way as an "o" or "v" variable (for these last the variable type assignment is automatic). The set field can generally assign a number or a numeric expression and the calculated value holds the decimal part.
In case of Integer type the set field can generally assign a number or a numeric expression but the calculated value truncates the decimal part.
Let us set, for example, for an “r” variable, the expression: "1000/3":

- in case of Double type variable, the calculated value will be = 333.333333
- in case of Integer type variable, the calculated value will be = 333.

In case of String type variable the set field generally assigns an alphanumeric expression and also the value assigned to the variable is a string. The String-type variable is typically used for subroutine assignment, as we can see in the following configuration example: "doors\prg1.abc"
The same considerations outlined for “r” variables apply to working parameters. In any case the variable type selection is transparent during programming, since it is set in workings database assignment.

11.2 Variables and Numeric-Type Parameters

A numeric expression is any expression which can be evaluated as a number. The elements of the expression can include any combination of keywords (the functions which can be used in parametric programming), variables (example: piece dimensions), constants (example: Greek pi) and operators (example: +, -, *, /,|) whose result is a number.
A numeric expression must be assigned:

- with lower-case characters
- the use of spaces is limited to string functions or variable arguments
- maximum allowed number of characters: 100.

Examples of numeric expression are the following:

- 20: the expression can be directly solved. It directly assigns the numerical value
- (100+32)/2: it uses numbers, mathematical operators, brackets
- r27+100: it uses numbers, variables, mathematical operators
- sqrt[r27+r15]-r5: it uses variables, mathematical operators, single-argument mathematical function.

The meaning of the above-listed expressions is intuitive. Let us follow step-by-step how each expression is evaluated:

- (100+32)/2=(132)/2=132/2=66
- (value of r27=50) = r27+100=50+100=150
- (value of: r27=50, r15=30, r5=-5) = sqrt[r27+r15]-r5=sqrt[50+30]-(-5)=sqrt[80]-(-5)=9.944271-(-5)= 9.944271+5=14.944271

Precedence of Operators
When an expression includes several operations, each part is evaluated and solved according a pre-established order, defined "precedence of operators".
Mathematical and logical operators are evaluated on the basis of the order of precedence specified in the following list:

- Multiplication (*), divisions (/, #), module (%), step adjustment (?) and logical operators (&, |);
- Addition and subtraction (+, –).

When there are operators with the same order of precedence in an expression (example: multiplication

and one division), each operation is evaluated in the order in which it appears, from left to right. Same for an addition and a subtraction within the same expression.
Using round brackets, it is possible to ignore the order of precedence and let that some parts of an expression are evaluated before others; The maximum limit of nested brackets depends only on the maximum allowable string length (100 characters). Expressions in round brackets are evaluated first. However, the normal precedence of operators is observed within round brackets.

11.3 Functions

The use of functions allows to make more complex computations that those allowed by operators. An example of function is "*sqrt[r27+r15]-r5*" which uses the *sqrt* mathematical function which calculates argument's square root.
Functions are divided into two categories:

- single-argument functions: an example is the *sqrt* function;
- multi-argument functions: an example is the *pown* function.

Single-argument functions can be used with two formalisms:

- numerical formalism: the argument is a positive number. Example "*sqrt25*": the argument (25) is written directly after the function name;
- non-numeric formalism: the argument is a negative number (example: -25) or it is expressed in parametric form (examples: "r25", "100-32"). Example "*sqrt[r25]*": now the argument is written in square brackets.

The non-numeric formalism is compulsory also for a few special single-argument functions, which belong to the References to piece variables group.

Multi-argument functions can only use the non-numeric formalism, with *name[op1;op2;…;opn]* syntax:

- *name* is the function name. Example: **pown**;
- *[…]* they delimit the function operands
- *op1* first argument
- *;* separator between two arguments
- *op2* second argument
- .
- *opn* last argument.

The number of arguments of a multi-argument function can be fixed or variable: in the following paragraphs we will examine in detail each single function, by giving particular attention to the required number of arguments and to which arguments it is necessary to assign and which not.
The way the syntax of a function is written is important to interpret the number and the use of arguments and reflects a general formalism. Let us see a few examples:

- *pown[nb;ne]* 2-argument function: both of them shall be assigned
- *min[n1;…;n30]* function with variable number of arguments: the allowable number ranges between 1 and 30;
- *case[nc;nc1:nv1;nc2:nv2;…;nvdef]* function with variable number of arguments: the first 3 (*nc;nc1: nv1;nc2:nv2*) shall be assigned, then follows a number of optional arguments (….;) and the last assigned (*nvdef*) has a particular interpretation;
- *prmac[(nm); nkind;(vdef)]* the 1st and the 3rd parameter are in round brackets (*nm*), (*vdef*): this means that the argument can be assigned empty (in this case: the function applies a default value). Since *vdef* is the last argument of the function it is also possible not to assign it at all.

There is no limit in terms of function nesting: it depends only on the maximum allowable string length (100 characters).

11.4 Variables and String-Type Parameters

Examples of alphanumeric expression are the following:

- doors\prg1.abc: the expression can be directly solved: it directly assigns the value (string)
- doors*r1.abc: it uses variables
- *pr[r45]: it uses variables, variable reference function.

The meaning of the expressions above is less intuitive than the case of numeric expressions. Let us examine step-by-step how each expression is evaluated:
(r1 is a string variable, with value="prg1") -> doors*r1.abc= doors\prg1.abc
(value of r45=2)-> *pr[r45]= *pr[2]-> (r2 is a string variable, with value="prg1") ->="prg1".
An alphanumeric expression can be assigned:

- also with upper-case characters
- the use of spaces is allowed (except for start and end spaces)
- characters included between ' ' (space) and '}' (decimal values between 32 and 125) can be used
- maximum allowed number of characters: 100.

While the formalism of a numeric expression fully satisfies the general solution criteria of an expression, an alphanumeric expression is interpreted on the basis of the two predefined and above-described formalisms, which must be observed:

- **"doors*r1.abc"**
  in this formalism the "*rn" expressions have a parametric interpretation, where “n” specifies the "r" variable to use (n=0-299).
  In the example:
- if r1 is a string-type variable, in this case the value (string) of r1 is replaced at the “*r1” expression, as stated above;
- but if r1 is a numeric-type variable, in this case the string corresponding to the entire part of the r1 value is replaced at the “*r1” expression;
- in case of r1 unassigned variable, in this case the "0" string is replaced at the “*r1” expression.

  There is no limit in the number of replacements. Thus, for example the following assignments are valid:
  "doors*r1.*r3"
  "abc*r5*r1.*r3":

  It is also possible to extract a part of the addressed string by a "*rn" expression.
  Syntax: "…*rn[ni;nc]…"   where:
- n = r variable index (example: 5 for r5). It can only be numeric;
- ni = start position from which the string assigned for r5 is read (significant from 1). It can be assigned:
  - numeric (example: ni=3),
  - with numeric-type r variable (example: ni=r2),
  - with $ variable -if in macro text- (example: ni=$0);
- nc = number of read characters, from ni (optional). It can be assigned:
  - numeric (example: ni=3),
  - with numeric-type r variable (example: ni=r2),
  - with $ variable -if in macro text- (example: ni=$0).

    Furthermore, it is also possible to handle the use of symbolic names for r variables, in the two following forms:
- "…….*r\name\….." attention: the symbolic name must be terminated by the '\' character
- "…….*r\name[ni;nc]….." attention: here the symbolic name is terminated by the '[' character

Example: "door leaves\ *r5[3;1].cnc"
let r5 be the assigned string variable = "abcdef';
ni=3: it reads r5 from the third character;
nc=1: it reads 1 character;
->   the solution is "door leaves\c.cnc".

Example: "door leaves\ *r5[3].cnc"

let r5 be the assigned string variable = "abcdef";
ni=3: it reads r5 from the third character;
nc is unassigned: it does not truncate the string;
-> the solution is "door leaves\cdef.cnc".

Example: "door leaves\ *r5.cnc"
let r5 be the assigned string variable = "abcdef";
-> the solution is "door leaves\abcdef.cnc".

Example: "door leaves\ *r\str1\.cnc"
let r5 be the assigned string variable = "abcdef", with name ="str1";
-> the solution is "door leaves\abcdef.cnc".

Example: "door leaves\ *r\pippo[3].cnc"
let r5 be the assigned string variable = "abcdef" , with name ="pippo";
ni=3: it reads r5 from the third character;
nc is unassigned: it does not truncate the string;
-> the solution is "door leaves\cdef.cnc".

- **"*pr[r45]"**
  this second formalism is more rigorous than the previous. In fact, it interprets the only "*pr[…..]" form, where the pr[..] function argument can assign any numeric expression.
  The solution of the pr[..] function argument is a numerical value of Integer type (n), which identifies in its turn a rn variable.
  - rn is normally a string type variable; it follows that the rn's value (string) assigns the string value of the alphanumeric expression;
  - but if rn is a numeric type variable, it follows that the string corresponding to the entire part of the rn's value assigns the string value of the alphanumeric expression.

  Let's consider the following example:
  r3 variable of numeric format = 250.8
  r5 string variable = *pr[3]= “250”;
  - if rn is not assigned, the "0" string is replaced at the *pr[…..]” expression.

11.5 Numeric Formats of Special Use

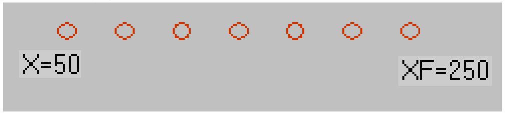
Let us examine here a form of special parameter prefix which, also if not directly used in programming, can be generated in applying tools (rotation, mirror,..).
It is about the "a;….." form of programming which can be set by working numeric parameters with meaning of coordinates.
Examples of valid assignments are the following:
"a;500" the parameter value is numeric
"a;l/2" the parameter value itself is parametric.

The "a;….." form means that the corresponding coordinate is in absolute programming.
Let us consider for example an arc working: the center coordinates are interpreted in relative positioning with respect to the arc starting point. It is possible to force an interpretation of coordinates in absolute positioning by using the "a;….." form

11.6 Expression Terms

11.6.1 Operators

Arithmetic

+	Addition. Example: 100.6 + 7 = 107.6
-	Subtraction. Example: 100.6 - 7 = 93.6
*	Multiplication. Example: 100 * 7 = 700
/	Division. The denominator cannot be zero. Error conditions: 125: null denominator; Example: 100 / 7 = 14.285714 Division between two operands.
%	Division modulus between two operands. Modulus (division reminder). The denominator cannot be zero. Example: 100 % 7 = 2 The computational procedure is as follows: • it does division (100 / 7) = 14.285714 • it separates the decimal part of the result: (14.285714 – 14) = 0.285714 • it multiplies by the divisor: 0.285714 * 7 = 2
#	Integer division. The denominator cannot be zero. Example: 100 # 7 = 14 The computational procedure is as follows: • it does division: 100 / 7 = 14.285714 • it separates the integer part of the result: integer (14.2857) = 14
?	Step adjustment between two operands. The denominator cannot be zero. Example: 100 ? 7 = 7.14285 The computational procedure is as follows: • it does division (100 / 7) = 14.285714 • it separates the decimal part of the result: (14.285714 – 14) = 0.285714 • it multiplies by the divisor: 0.285714 * 7 = 2 (=100 % 7) • it divides the module by the integer of the division: 2 / 14 = 0.14285 • it adds to the divisor: 7+ 0.14285 =7.14285. The first three points calculate the module. Then, the ? operand returns the divisor, modified so as to obtain an integer division result. In the particular case of a division result lower than 1, the operation returns the dividend. Example: 10 ? 15 = 10 {the solution is: 10/15= 0.6666} Example: programming a X fitting in the particular case, in which the final X coordinate coincides with the drilling position and it is necessary adjust the pitch between the two holes. Example initial x=50, final, final x=250, Pitch = 200?32. The resulting pitch is 33.33 and the last hole is made at the coordinate X = 250. 

Logical

Following symbols should be considered as operators of advanced programming.

&	The bitwise AND operator compares each bit of the first integral operand to the

	corresponding bit of the second integral operand, with truncation of the decimal part of the result. Example: 10.456 & 3.56 = 10 & 3 = 2 In fact the bit representations of 10 and 3 are considered: 10 = 1 0 **1** 0 and 3 = 0 0 **1** 1 = 0 0 1 0 = 2 decimal value
|	The bitwise OR operator compares each bit of the first integral operand to the corresponding bit of the second integral operand, with truncation of the decimal part of the result. Example: 10.456 | 3.56 = 10 | 3 = 11 In fact the bit representations of 10 and 3 are considered: 10 = **1** 0 **1** 0 or 3 = 0 0 **1 1** = 1 0 1 1 = 11 decimal value

Brackets, Separators

(..)	Level of brackets: they can be nested without limit. The first brackets of the expression to be evaluated are the innermost brackets. Example: "12*((r0+r3)*sqrt[12])" • 1st operation: (r0+r3) {example =10} • 2nd operation: (10*sqrt[12])=34.64 • 3roperation: 12*34.64=415.6921.
[..]	Delimiters for: parametric or negative function argument; assignment of multi-operand function. Examples: sqrt[r12] single-argument function with parametric argument sin[-45] single-argument function with negative argument min[r12;1;67] multi-argument function
. ,	Separator between integer and decimal part of a numeric argument. The separator to use is only one and is indicated in Configuring Tpaedi32 Examples: 128.6 .965
;	Separator between arguments of multi-argument function. Example: pown[5;2]
"…"	Direct string assignment (for example for function*: strcmp*). The space character is also allowed. Example: strcmp[5; "pippo"] evaluates r5 and compares with "pippo"; string strcmp [5; "ciao …"] evaluates r5 and compares with "ciao …" string

11.6.2 Variable Arguments

Variable Arguments

pi	Greek pi (π = 3.1415..) . Usable: always.

eps	Threshold (epsilon) for linear position, takes a value conforming to the current unit of measure of the program: • 0.001 for mm • 0.001/25.4 for inch multiplied by the Epsilon Multiplier factor. (Assigned in Configuring Tpaedi32) Usable: always.
cnq	Conversion factor for linear position, takes value conforming to the current unit of measure of the program: • 1 for mm • 1/25.4 for inch Usable: always. The *cnq* variable argument is to be used in writing subroutines and/or macro-programs, for comparisons and/or direct assignments with coordinate (or speed) values, when it is predictable that the subroutine (or macro-program) can be used in a program indifferently written in [mm] or [inch] units. Example: • a subroutine (ONE) is written in [mm] units • the subroutine executes drilling works, spaced each other by an offset in x • the distance between holes is assigned in an r variable (reassignable): example r3 • a minimum 20 mm distance is wished to be applied: then, r3 is compared with number 20. There are no problems if also the program which applies ONE is written in [mm]. If, on the contrary, the program which applies ONE is written in [inch]: r3 is now set to [inch]. In this case it is no longer possible to compare r3 directly with 20. Both cases are valid if the comparison is made with "20*cnq": 20*cnq is valid: • 20 if the program is written in [mm] units • 0.7874 if the program is written in [inch] units
l h s	Piece dimensions: X (l), Y(h), Z(s). Usable: always.
face	Face number (only for face programming). There are several conditions of use: • in working parameter: it returns the number of the face the working is applied to (value from 1 to 99): • in case of real face number (value from 1 to 6): it is about the face custom number; • in case of piece-face: it matches the F field (in case of application in automatic face: it returns the number assigned to the automatic face – example: 120-); • in the list of program variables of "r" type: it returns the –1 value; • in assignment of (o, v) variables or variable geometry: it returns the value –1. Usable: always.
lf hf sf	Face dimensions: X(lf), Y(hf), Z(sf). There are several conditions of use: • in working parameter: they return the dimensions of the face the working is applied to: • in case of piece-face: they match the F field (in case of application in automatic face: they return the automatic face dimensions); • in the list of program variables of "r" type: they return the corresponding piece dimensions; • in assignment of (o, v) variables or variable geometry: they return the corresponding piece dimensions. Usable: always.

prgt	Piece typology: 0=program, 1=subroutine,2=macro Usable: always
prgrd	Access piece level: 0=Operator level, 1=Installer, 2= Constructor level Usable: always.
prgwr	Edit piece level: 0=Operator level, 1=Installer, 2= Constructor level Usable: always.
'ch'	Replaces a numeric decimal value corresponding to the ASCII coding of the ch. character. Valid codes: from 32 (' ') to 125 ('}'). Usable: always. Example: 120+'a'=120+97=217 'a'-' ' = 97-32=65

Execution mode

prgrun	Program execution environment: 0 = edit; 1 = running. The prgrun argument can be useful to differentiate custom error messages: for example, it is possible to implement error messages relative to technological malfunctions only in Execution mode. Usable: always.
prgn	Flag of normal execution: 1 = normal execution; 0 = different execution. In edit mode (*prgrun* is set to 0): the value corresponds to what assigned to the piece in *Execution mode*. The prgn argument allows to differentiate the execution of a program on the basis of the way the program itself is deployed. Examples: to execute or not some workings, to assign the direction of execution of a sawing work. Usable: always.
prgx	Flag of mirrored execution X: 1 = mirrored execution X; 0 = different execution. In edit mode (*prgrun* is set to 0): the value corresponds to what assigned to the piece in *Execution mode*. The *prgx* argument allows to differentiate the execution of a program on the basis of how the program itself is deployed. Usable: always.
prgy	Flag of mirrored execution Y: 1 = mirrored execution Y; 0 = different execution. In edit mode (*prgrun* is set to 0): the value corresponds to what assigned to the piece in *Execution mode*. The *prgy* argument allows to differentiate the execution of a program on the basis of how the program itself is deployed. Usable: always.
prgxy	Flag of mirrored execution XY: 1 = mirrored execution XY; 0 = different execution. In edit mode (*prgrun* is set to 0): the value corresponds to what assigned to the piece in *Execution mode*. The *prgxy* argument allows to differentiate the execution of a program on the basis of how the program itself is deployed. Usable: always.
prarea	Execution area. In edit mode (*prgrun* is set to 0): the value corresponds to what assigned to the piece in *Execution mode*. The *prarea* argument allows to differentiate the execution of a program and/or make specific evaluations on the basis of where the program itself is deployed. Usable: always.
prqx prqy prqz	X, Y, Z Step in execution area. In edit mode (*prgrun* is set to 0): the value corresponds to what assigned to the piece in *Execution mode*. The *prgx/y/z* arguments allow to differentiate the execution of a program and/or make specific evaluations on the basis of where the program itself is deployed.

	Usable: always.
prun1 prun2 prun3 prun4 prun5	Additional parameters in execution. In edit (prgrun = 0) mode, the value corresponds to the value set for the piece in: *Execution mode.* In Tpaedi32 the parameters have always the same value 0. They can be assigned during the external optimisation. Usable: always.

Environment Settings

The arguments can be used during the writing of sub-routines and/or macro-programs, to compare and/or direct assignments of values (coordinates, rotation axes), where it is expected that the sub-routine (or macro-program) can be used in a non-predifined configuration.
arguments of advanced programming should be considered.

sysface	Face geometry. 0= cartesian piece 1= faces in transparency 2= custom systems Usable: always.
sysquad	Operating quadrant. (value from 1 to 4). Usable: always.
sysz	Indicates the direction of the z axis applied to faces: 0= negative z (-Z) axis enters into the piece; 1= positive z (+Z) axis enters into the piece. Usable: always.
sysxz	Indicates the arc typology on the xz-plane: 0 = xz-plane; 1 = zx-plane. Usable: always
sysbeta	Indicates the managed rotation for the slewing axis: 0=positive towards x positive 1=positive towards x negative Usable:: always
sysfeed	It indicates the programming unit of the linear speed rates: • in a program with [mm] unit, possible selections are: 0=[mt/min] 1=[mm/min] • in a program with [inch] unit, possible selections are: 0=[inch/sec] 1=[inch/min].

Piece Variables

o0-o7 o\name	Piece variables of "o" type. There are several conditions of use: • in program text: the only assigned and manipulated variables can be used (there may be a number of variables lower than 8, or at worst no variables at all); • in macro text: there are 8 variables and they can always be used; The second above-mentioned form corresponds to the symbolic formalism, with: • fixed part "o\"; • variable part: the symbolic name assigned to the variable. They cannot be used: • in assignment of 'o', 'v' variables; • in assignment of custom functions. Error conditions:

	• 114: use in invalid context; • 121: invalid index to "o" variable.
v0-v7 v\name	Piece variables of "v' type. There are several conditions of use: • in program text: the only assigned and manipulated variables can be used (there may be a number of variables lower than 8, or at worst no variables at all); • in macro text: there are 8 variables and they can always be used; The second above-mentioned form corresponds to the symbolic formalism, with: • fixed part "v\"; • variable part: the symbolic name assigned to the variable. It cannot be used: • in assignment of 'o', 'v' variables; • in assignment of custom functions. Error conditions: • 113: use in invalid context; • 120: invalid index to "v' variable.
r0-r299 r\name	Piece variables of "r" type. In variable or numeric parameter assignment: it takes the r variable value (0.0 in case of non-numeric variable). In variable or non-numeric parameter (string) assignment and if the "*rn" (with n= 0 – 299) remarkable form is recognized: • if *rn* identifies a string variable: it returns the corresponding string value; • if *rn* identifies a numeric variable: it returns the string corresponding to the integer part of the variable (numeric) value; • if *rn* identifies an unassigned variable: it returns the string "0". It is also possible to extract a part of the addressed string by a "*rn" expression. Syntax: "…*rn[ni;nc]…" where: • n = r variable index (example: 5 for r5). It can only be numeric; • ni = start position from which the string assigned for r5 is read (significant from 1) . It can be assigned: • numeric (example: ni=3), • with numeric-type r variable (example: ni=r2), • with $ variable -if in macro text- (example: ni=$0); • nc = number of read characters, from ni (optional). It can be assigned: • numeric (example: ni=3), • with numeric-type r variable (example: ni=r2), • with $ variable -if in macro text- (example: ni=$0). The second above-mentioned form (recognized in variable or numeric parameter assignment) corresponds to the symbolic formalism, with: • fixed part "r\"; • variable part: the symbolic name assigned to the variable. They cannot be used: • in assignment of 'o', 'v' variables • in assignment of custom functions. Error conditions: • 112: use of r variable in invalid context; • 117: invalid index to "r" variable • 102: in case of "…**rn[ni;nc]**…" format, with ni or nc assigned with invalid syntax. Example: "doors*r28.*r29" where: • r28 is a string variable with ="p007" (string) value • r29 is a numeric variable, with =12.5 (numeric) value

	the computation of the expression will result in the following (string) value: ="doors\p007.12".

j0-j99	Program global variables. There are several conditions of use: • in working parameter: it applies the real values of j variables; • in the list of program variables of "r" type: it applies always null values. <u>They cannot be used</u>: • in assignment of 'o', 'v' variables; • in assignment of custom functions; • in assignment of variable geometries (fictive face edges). <u>Error conditions</u>: • 115: use in invalid context; • 118: invalid index to "j" variable.
$0-$299	Auxiliary variables: they can be used only in writing a macro-program. <u>They cannot be used</u>: • in 'o', 'v', 'r' variable assignment; • in assignment of custom functions; • in assignment of variable geometries (fictive face edges); • in program text. <u>Error conditions</u>: • 111: use in invalid context; • 119 : invalid index to "$" variable.

References To Piece Variables

They are parametric forms that allow to synthesize the reading of program variables.
They are normally used to write macros. They should be considered as advanced programming forms.

pr[.]	Reference to r variable. The parametric form (use of [.] brackets) is compulsory and the expression is evaluated by taking the rn variable value, with n=value calculated in square brackets. In variable or numeric parameter assignment: it returns the numeric value of the rn variable (0.0 in case of non-numeric variable); In variable or non-numeric parameter (string) assignment, if the "*pr[..]" remarkable form is recognized: • if *rn* identifies a string variable: it returns the corresponding "$" variable calculated; • if *rn* identifies a numeric variable: it returns the "$" variable corresponding to the integer part of the variable value; • if *rn* identifies an unassigned variable: it returns the string "0". <u>It cannot be used</u>: • in assignment of 'o', 'v' variables; • in assignment of custom functions. <u>Error conditions</u>: • 112: use of r variable in invalid context; • 117: invalid index to "r" variable. Examples: pr[12]: it returns the r12 value pr[10+5]: it returns the r15 value pr[r1], let r1=7: it returns the r7 value.
pj[.]	Reference to a program global variable. The parametric form (use of [.] brackets) is compulsory and the expression is evaluated by taking the jn variable value, with n=value calculated in [.].

	There are several conditions of use: • in working parameter: it applies the real values of j variables; • in the list of program variables of “r” type: it applies always null values. It cannot be used: • in assignment of 'o', 'v' variables; • in assignment of custom functions; • in assignment of variable geometries (fictive face edges). Error conditions: • 115: use in invalid context; • 118: invalid index to "j" variable.
p$[.]	Reference to auxiliary variable in macro-program text. The parametric form (use of [.] brackets) is compulsory and the expression is evaluated by taking the $n variable value, with n=value calculated in [.]. The function can only be used in writing a macro-program. It cannot be used: • in 'r', 'o', 'v' variable assignment; • in assignment of custom functions; • in assignment of variable geometries (fictive face edges); • in program text. Error conditions: • 111: use in invalid context; • 119: invalid index to “$” variable.

Assignments Relative To the Application of Subroutine or Macro

They are arguments returning information about subroutine or macros application. They should be used in the text of the same sub-routine or macro.

subx suby subz	Return the x, y, z positioning coordinates: • in cycle application, they take the value of the point of application, with point hook and relative programming solved; • in 'r' variable assignment: they take value 0.0. The *subx/y/z* arguments allow to know the point of application set in a subroutine call, within the subroutine itself. They cannot be used: • in assignment of 'o', 'v' variables; • in assignment of custom functions; • in assignment of variable geometries (fictive face edges). Error conditions: • 109: use in invalid context;
subang suba subang0	Returns the rotation angle: • in cycle application, they take the rotation angle value; • in 'r' variable assignment: they take value 0.0. The *subang* (*suba*) argument allows to know the rotation value set in a subroutine call, within the subroutine itself. The argument *subang0* lets the user to know the rotation value set in the subroutine call, within the subroutine itself, but including all the possible previous calls. Let us consider a subroutine applied with a 20° rotation within an external call with a -5° call: • the local rotation evaluated at the more internal level is returned by *subang* and the result is 20°, • the overall rotation evaluated at the more internal level is returned by *subang0* and the result is: 20°+(-5°)=15 They cannot be used: • in assignment of 'o', 'v' variables;

	• in assignment of custom functions; • in assignment of variable geometries (fictive face edges). Error conditions: • 109 : use in invalid context;
subinv subi	Returns the assignment of inversion: • in cycle application, they take the inversion parameter value (1 if required); • in 'r' variable assignment: they take value 0. The *subinv* argument allows to know if a subroutine call has required the inversion of execution, within the subroutine itself. They cannot be used: • in assignment of 'o', 'v' variables; • in assignment of custom functions; • in assignment of variable geometries (fictive face edges). Error conditions: • 109: use in invalid context.
submir subm submir0	Returns the assignment of mirror: • in cycle application, they take the value corresponding to the required mirrored execution (0=inactive; 1= mirror x; 2= mirror y; 3= mirror x+y); • in 'r' variable assignment: they take value 0.0. The *submir* (*subm*) argument allows to know if a subroutine call has required a mirrored execution, within the subroutine itself. The argument *submir0* lets the user to know the mirroring value set in the subroutine call, within the subroutine itself, but including all the possible previous calls. Let us consider a subroutine applied with a x+y mirror within an external call with a x mirror: • the local mirror evaluated at the more internal level is returned by *submir* and results in x+y • the overall mirror evaluated at the more internal level is returned by *submir0* and results in: y They cannot be used: • in assignment of 'o', 'v' variables; • in assignment of custom functions; • in assignment of variable geometries (fictive face edges). Error conditions: • 109: use in invalid context.
sublink subl	Returns the assignment of point hook: • in cycle application, they take value 1 if a point hook is required; • in 'r' variable assignment: they take value 0. The *sublink* argument allows to know if a subroutine call has required a point hook for a given execution, within the subroutine itself. They cannot be used: • in assignment of 'o', 'v' variables; • in assignment of custom functions; • in assignment of variable geometries (fictive face edges). Error conditions: • 109: use in invalid context
substr subs substr0	Returns the assignment of stretch: • in cycle application, it takes the required stretch value; • in 'r' variable assignment: they take value 1.0. The *substr* (*subs*) argument allows to know if a subroutine call has required a stretch operation for a given execution, within the subroutine itself. The argument *substr0* lets the user to know the resizing value set in the subroutine call, within the subroutine itself, but including all the possible previous calls. Let us consider a subroutine applied with a 2.0 resize within an external call with a 0.5

	resize: • the local resize evaluated at the more internal level is returned by *substr* and the result is: 2.0 • the overall resize evaluated at the more internal level is returned by *substr0* and the result is: 2.0*0.5=1.0 They cannot be used: • in assignment of 'o', 'v' variables; • in assignment of custom functions; • in assignment of variable geometries (fictive face edges). Error conditions: • 109: use in invalid context.
subemp sube	Returns the assignment of emptying: • in cycle application, they take value 1 if emptying is required; • in 'r' variable assignment: they take value 0. The *subemp* argument allows to know if a subroutine call has required to apply emptying operations, within the subroutine itself. They cannot be used: • in assignment of 'o', 'v' variables; • in assignment of custom functions; • in assignment of variable geometries (fictive face edges). Error conditions: • 109: use in invalid context.
subface subf	Returns the applied face. • in cycle application, they take the number value of the applied face: • in case of real face number (value from 1 to 6): it is about the face custom number • in 'r' variable assignment: they take value -1. They cannot be used: • in assignment of 'o', 'v' variables; • in assignment of custom functions; • in assignment of variable geometries (fictive face edges). Error conditions: • 109: use in invalid context.

submaster	Returns multiple call information: • in cycle application, it takes the value: • -2: if it corresponds to a master call; • >0: if it corresponds to an induced call. It returns the master face number if it is a real face number (value 1 to 6): it is the face custom number-); • -1 in any other case; • in “r” variable assignment: it takes value -1. They cannot be used: • in assignment of 'o', 'v' variables; • in assignment of custom functions; • in assignment of variable geometries (fictive face edges). Error conditions: • 109: use in invalid context.
subvl subvb subvo subvm subvk	Return the value assigned to the property field: L (subvl) B (subvb) O (subvo) M (subvm) K (subvk)

	• in cycle application, they take the property value • in 'r' variable assignment they take value 0. They cannot be used: • in assignment of 'o', 'v' variables; • in assignment of custom functions; • in assignment of variable geometries (fictive face edges). Error conditions: • 109: use in invalid context.
rempty[nn]	It verifies that the r variable, whose index ist indicated by nn, is assigned. Returns 0 if the variable is assigned, otherwise 1. The parametric form is obligatory (use of the [] square brackets). If the argument of the function is parametric and has "rn" or "r\name" remarkable form, the function works directly on the rn variable. Following situations can occur: • in case of subroutine or macro, it returns value 0, if the r variable of the cycle is assigned in the call or if it corresponds to a variable of the subroutine or of the not reassignable macro. • in the assignation of r variable or not in the cycle assignation takes value 0, if the r variable of the program is assigned. They cannot be used: • in assignment of 'o', 'v' variables; • in assignment of custom functions; Error conditions • 112: use of r variable in invalid context; • 117: invalid index to "r" variable

Setting of custom sections

The arguments allow a direct access to the data assigned in the custom sections, limited to the numerical items (whole numbers or numebers with comma, selection from the list).
If the section or the option indicated is not assigned, the argument takes a null value (0).
Usage:
typically in sub-routines and/or macro-programs writings, that should be considered as advanced programming forms.

The arguments cannot be used in:

- variable assignment (‘or ’, ‘v.’, ‘R’)
- variable geometry assignment
- in assignment of custom section
- in assignment of custom functions.

szs\name	Returns a numeric item in the Special settings section: • fix part “szs\”; • variable part (“name”): the symbolic name assigned for the item in the section. Example: “szs\aaa” : returns the field value named "aaa".
szs\name	returns a numeric item in the section of additional Infos:
szo\name	Returns a numeric item in the section of Optimization settings:
szl\name	Returns a numeric item in the section of Constraint settings.

11.6.3 Auxiliary Functions

They are normally used to write sub-routines and/or macro-programs.
They should be considered as forms of advanced programming

nfa	Returns the real number of the (custom) face specified as argument. The function operates in case of argument of value included between 1 and 6: the argument is interpreted as face custom number and the function returns the face real number. Otherwise: it returns the integer part of the argument in any case. Usable: always. Error conditions: none. Examples: assign a face custom numbering as follows: face 1 -> custom number: 5 face 2 -> custom number: 6 face 3 -> custom number: 1 face 4 -> custom number: 4 face 5 -> custom number: 2 face 6 -> custom number: 3 nfa5=1 nfa2=5
nfc	Returns the custom number of the (real) face specified as argument. The function operates in case of argument of value included between 1 and 6: the argument is interpreted as face real number and the function returns the face custom number. Otherwise: it returns the integer part of the argument in any case. Usable: always. Error conditions: none. Examples: (assign the face custom numbering as indicated for function: *nfa*) nfc1=5 nfa2=6

11.6.4 Mathematical and statistical functions

abs	Returns the absolute value of the argument. The abbreviated form is accepted: a. Usable: always. Error conditions: none.
sqrt	Extracts the square root of the argument. The value of the argument must be positive (>=0.0). The abbreviated form is accepted: q. Usable: always. Error conditions: • 127: negative argument. Examples: *sqrt[25]* = 5 *sqrt*[-25] <- it causes error (127)
int	Returns the integer part of the argument, obtained by truncation. The abbreviated form is accepted: i. Usable: always. Error conditions: none.

	Examples: *int*[-12.8] = -12.0 *int*[12.9] = 12
inv	Returns the reciprocal of the argument (1/x). The argument cannot be null. The abbreviated form is accepted: v. Usable: always. Error conditions: • 125: null argument. Examples: *inv2* = 0.5 *inv* 0 <- it causes error (125)
pow	Squares the argument. The abbreviated form is accepted: p. Usable: always. Error conditions: none. Examples: *pow3* = 9 *pow0* = 0
pown[nb;ne] pown[nb;ne;ne2]	Raises the first argument to the power resulting from the second argument: • nb (1° argument) = base • ne (2° argument) = exponent. It is applied to the integer part. • ne1,ne2 (2° argument and 3° argument) = the exponent is calculated as ne=(ne1/ ne2) and used without truncation of the entire part. The arguments of this function can be 2 or 3. Particular cases: • nb#0 and ne=0 it returns 1 • ne=0 it returns 0. • ne2=0.0 uses the 2-arguments form Usable: always. Error conditions: • 123: number of operands =0; • 124: number of operands #2,3; • 128: *ne* argument nb=0.0, *ne* <0 (negative) Examples: pown[5;2] = 5 * 5 =25 pown[5;3.5] = pown[5,3] = 5 * 5 * 5 = 125 pown[5, 0] = 1 pown[0, 5] = 1 pown[5;1;2]= 5 ^1/2 =2.236
round	Rounds the argument to the nearest integer. Usable: always. Error conditions: none. Examples: round[12.8] = 13 round[12.3] = 12 round[12.5] = 12 round[-10.3] = -10 round[-10.7] = -11
odd	Returns 1 if the integer part of the argument is odd; otherwise 0. Usable: always. Error conditions: none. Examples:

	*odd.*12.8 = *odd*12 = 0 *odd*13.8 = *odd*13 = 1
hypot[c1;c2] hypot[c1;c2; c3]	Returns the hypotenuse of the right triangle which has assigned legs.The arguments of the function can be 2 or 3. If the arguments are 3, the triangle is assigned in the space. Usable: always. Error conditions: • 123: number of operands =0; • 124: number of operands #2 e #3; Example: hypot[5;2] = sqrt[5*5+2*2] = 5.385
min[n1;..;n30] max[n1;..;n30] ave[n1;..;n30] sum[n1;..;n30]	Return the minimum, maximum, average value between the arguments or the sum of the arguments. The maximum number of arguments is 30. Usable: always. Error conditions: • 123: number of operands =0; • 124 : number of operands >30. Examples min[5;12;3;25] it returns 3 max[5;12;3;25] it returns 25 ave[5;12;3;25] it returns (5 + 12 + 3 + 25) /4 = 11.25 sum[5;12;3;25] it returns 5 + 12 + 3 + 25 = 45
minr[n1;n2] maxr[n1;n2] aver[n1;n2] sumr[n1;n2]	Return the minimum, maximum, average value or the sum between the values assigned to r variables in the (n1, n2) range. The arguments of this function must be 2. The concerned variables may be of any type: • numeric (double, integer): the function reads the value; • string: the function takes value 0.0. The n1 and n2 arguments must identify a range of variables included between r0 and r299. The functions are always interpreted differently according to the conditions of use: • in (complex) working for assignment of subroutine variable reassignment or in generic parameter: the function moves research to previous expansion levels (up to the list of r variables of the main program); • in (complex) working for assignment of subroutine variable non-reassignment: the function moves research to previous expansion levels (up to the list of r variables of the main program) but starting from its own level; • in the list of r variables: the function applies variables from the list. They cannot be used: • in assignment of 'o', 'V variables • in assignment of custom functions. Error conditions: • 112: use of r variable in invalid context; • 123: number of operands =0; • 124: number of operands #2; • 117: invalid index to "r" variable. Example r10=min[2;5] with values r2=5; r3=12; r4=3; r5=25 the function returns 3
minj[n1;n2] maxj[n1;n2] avej[n1;n2] sumj[n1;n2]	Return the minimum, maximum, average value or the sum between the values assigned to j variables in the (n1, n2) range. The arguments of this function must be 2. The n1 and n2 arguments must identify a range of variables included between j0 and j99.

	There are several conditions of use: • in working parameter: it applies the real values of j variables; • in the list of program variables of “R” type: it applies always null values; They cannot be used: • in assignment of 'o', 'v' variables • in assignment of custom functions • in assignment of variable geometries (fictive face edges). Error conditions: • 123: number of operands =0; • 124: number of operands #2; • 115: use in invalid context; • 118: invalid index to “j” variable. Example *maxj*[2;5] (with values j2=5; j3=12; j4=3; j5=25) -> the function returns 25.
min$[n1;n2] max$[n1;n2] ave$[n1;n2] sum$[n1;n2]	Return the minimum, maximum, average value or the sum between the values assigned to $ variables in the (n1, n2) range. The arguments of this function must be 2. The n1 and n2 arguments must identify a range of variables included between $0 and $299. They cannot be used: • in 'r', 'o', 'v' variable assignment; • in assignment of custom functions; • in assignment of variable geometries (fictive face edges); • in program text. Error conditions: • 123: number of operands =0; • 124: number of operands #2; • 111: use in invalid context; • 119: invalid index to “$” variable. Example *ave$*[2;5] (with values $2=5; $3=12; $4=3; $5=25) -> the function returns 11.25

11.6.5 Trigonometric Functions

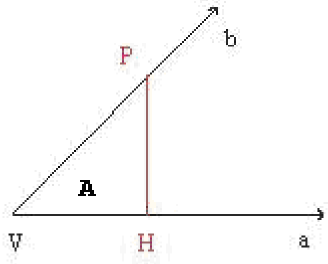
Outlines of trigonometry

The brief outlines of trigonometry given below provide a frame of reference to solve geometric problems, which recur in the programming.

The main unit of measurement of the plane angles are: The centesimal degree and the radiant angle.

In Mathematics the linear measure of the angles, whose unit of measurement is the radiant, is normally used; however, the most widely used unit of measurement of the angles is definitely the degree. For this reason, the following trigonometric functions require or return angular values expressed in degrees.

It is useful to remember: 1 radiant = (180/p) °, with (p = 3.1415..) known as pi (π).



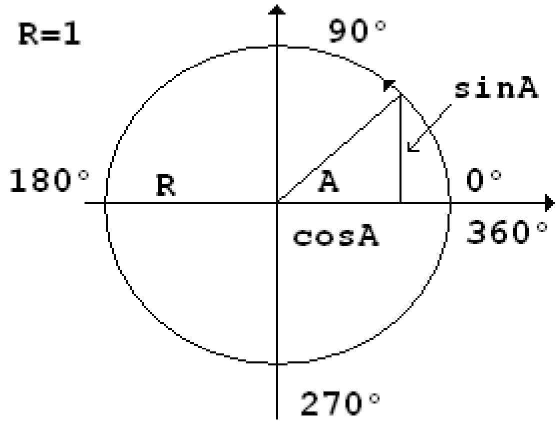
The user should remember that an angle is defined as positive when it rotates in counterclockwise direction.
Let us consider an angle **(A):** (in radians) of vertex **V** and sides **a** and **b**. Let us take on the half-line **b** any point **P** distinct from the vertex **V**Let us project it on the half-line **a**: be **H** the point of the perpendicular described by **P** on **a**.

Now, let us consider the right triangle **VHP** and the ratio between the oriented segments:
HP/VP; VH/VP; HP/ VH
It is shown that these ratios only depends on the angle A and not on the the point P chosen on the half-line b.
The three written ratios define the three functions of the angle **A** called:

sinus of **A**	$\frac{HP}{VP} = sin\,\mathbf{A}$ in particular with **VP**=1, is: **HP**=sin**A**
cosine of **A**	$\frac{VH}{VP} = cos\,\mathbf{A}$ in particular with **VP**=1, is: **HP**=cos**A**
tangent of **A**	$\frac{HP}{VH} = tg\,\mathbf{A}$

Furthermore it is demonstrated the significant relationship, as follows: (sin**A**)2 + (cos**A**)2 = 1.0.

With reference to the figure below, the correspondences are as follows:



Aº	**sinA**	**cosA**	**tgA=sinA/cosA**
0	0.0	1.0	0.0
0÷90	0.0 ÷1.0	1.0 ÷0.0	0.0÷ +(infinite)
90	1	0.0	+(infinite)
90÷180	1.0 ÷0.0	0.0÷(-1.0)	-(infinite) ÷ 0.0
180	0.0	-1.0	0.0
180÷270	0.0÷(-1.0)	(-1.0) ÷0.0	0.0 ÷ +(infinite)
270	-1.0	0.0	-(infinite)
270 ÷360	(-1.0) ÷0.0	0.0 ÷1.0	-(infinite) ÷ 0.0
360	0.0	1.0	0.0

Functions

sin	Computes the sine of the argument (in °).The value of the function is included in the range (-1.0 ÷ 1.0). The abbreviated form is accepted: s. Usable: always. Error conditions: none. Example *sin[90]*= 1 *sin[-90]*= -1
cos	Computes the cosine of the argument (in °).The value of the function is included in the range (-1.0 ÷ 1.0). The abbreviated form is accepted: c. Usable: always.

	Error conditions: none. Example *cos[90]*= 0 *cos*[gr[pi]] = -1
tan	Computes the tangent of the argument (in °). The abbreviated form is accepted: t. Usable: always. Error conditions: 132: invalid angle for tangent calculation. Example *tan[45]*= 1 *tan[90]*= causes error 132 *tan[-90]*= causes error 132
asin,as	Computes the arc-sine of the argument. The value returned by the function is in ° (degrees), included between 0 and 180°. The value of the argument must be included between - 1 and 1. The abbreviated form is accepted: d. Usable: always. Error conditions: • 126 : argument outside the range of values (- 1; 1). Example *asin1*= 90 *asin*[-*1*] = -90
acos,ac	Computes the arc-cosine of the argument. The value returned by the function is in ° (degrees), included between 0 and 180°. The value of the argument must be included between: - 1 and 1. The abbreviated form is accepted: e. Usable: always. Error conditions: • 126 : argument outside the range of values (- 1; 1). Example *acos0*= 90 *acos*[-*1*] = 180
atan,at	Computes the arc-tangent of the argument. The value returned by the function is in ° (degrees), included between -90° and 90°. The abbreviated form is accepted: f. Usable: always. Error conditions: none. Example *atan1*= 45 *atan*[-*1*] = -45
gr	Converts the argument from radians into degrees (°) : 1 radian=(180/Greekpi) °. The abbreviated form is accepted: g. Usable: always. Error conditions: none. Example gr[pi] = 180
atan2[y,x]	It calculates the arc-tangent of (y/x). The value returned by the function is in ° (degrees), included between -180° and 180°. If both arguments are null, it returns value 0. Usable: always. Error conditions: • 123: number of operands =0; • 124: number of operands #2;

	<u>Example</u> *atan2[1;0]* = 90

11.6.6 Functions Which Operate On Strings

`They should be considered as forms of advanced programming`

strlen[nn]	Returns the number of characters set for the r variable addressed by the argument. The examined string is: • di impostazione: se variabile numerica, • if string variable. In case of unassigned variable, the function returns 0. *Formato notevole* If the function argument is a parameter and has "rn" or "r\name" remarkable format (in this case the parametric form is compulsory: <u>It cannot be used:</u> • in assignment of 'o', 'v' variables; • in assignment of custom functions; <u>Error conditions:</u> • 112: : use of r variable in invalid context; • 117: : invalid index to "r" variable. <u>Esempio 1</u> Sia assegnata la variabile di tipo stringa r5="submio*r4", con r4="pippo" Risolve la parametrizzazione di r5: "submio\pippo" *strlen5* restituisce il valore 12 che corrisponde al numero di caratteri in "submio\pippo"; *strlen[r5]* restituisce il valore 12 che corrisponde al numero di caratteri in r5="submio\pippo"; <u>Esempio 2</u> Sia assegnata la variabile numerica r5="r4/12" *strlen5* restituisce il valore 5 che corrisponde al numero di caratteri in "r4/12".
getat[nn;np]	Restituisce il valore decimale corrispondente ad un carattere estratto dalla stringa: • di impostazione: se variabile numerica, • valore: se variabile di tipo stringa. Returns the value 0, in case of: • variabile non assegnata, • posizione del carattere non valida (minore di 1 o maggiore della lunghezza della stringa). <u>Argomenti</u>: nn = indice di variabile r. Il valore di nn deve essere compreso tra 0 e 299; np = posizione del carattere in valore stringa (significativo da 1). *Formato notevole* Se il primo argomento della funzione è parametrico ed ha formato notevole "rn" o "r\name": la funzione opera direttamente sulla variabile rn. <u>Non è utilizzabile</u>: • in assegnazione di variabile 'o', 'v' • in assegnazione di funzioni custom <u>Situazioni di errore</u>: • 112: utilizzo di variabile r in contesto non valido; • 123: numero di operandi =0; • 124: numero di operandi #2; • 117: indice di variabile r non valido. <u>Esempio 1</u> sia assegnata la variabile di tipo stringa r5="s2*r4", con r4="pippo" risolve la parametrizzazione di r5: "s2\pippo"

	getat[5;2] restituisce 50 che corrisponde al valore decimale del carattere '2' *getat[5;6]* restituisce 112 che corrisponde al valore decimale del carattere 'p' Esempio 2 sia assegnata la variabile di tipo numerico r5="r4/12" *strlen[5,2]* restituisce 52 che corrisponde al valore decimale del carattere '4'.
strcmp[n1;n2]	Restituisce il valore del confronto tra la prima stringa e la seconda: • 0 se le due stringhe sono uguali, • <0 se la prima stringa è minore della seconda, • >0 se la prima stringa è maggiore della seconda. Il confronto non tiene conto delle differenze tra lettere maiuscole o minuscole. Argomenti: n1= indice di prima variabile r n2= indice di seconda variabile r. Gli argomenti possono avere impostazione parametrica. Opera su variabile r di tipologia qualunque: • se di tipologia stringa: applica alla $ risolta, • se tipologia numerica: applica alla $ di data-entry, • se la variabile non è assegnata: applica a stringa vuota. Un argomento (o entrambi) possono assegnare direttamente una stringa, compresa tra doppi apici. *Formato notevole* Se un argomento della funzione è parametrico ed ha formato notevole "rn" o "r\name": la funzione opera direttamente sulla variabile rn. Non è utilizzabile: • in assegnazione di variabile 'o', 'v' • in assegnazione di funzioni custom. Situazioni di errore: • 112: utilizzo di variabile r in contesto non valido; • 123: numero di operandi =0; • 124: : number of operands #2; • 117: indice di variabile r non valido. Esempi: strcmp[5; "pippo"] evaluates r5 and compares with "pippo" string strcmp[r5; "pippo"] evaluates r5 and compares with "pippo" string strcmp["pippo";r6] evaluates r6 and compares with "pippo" string strcmp[r5;r6] evaluates and compares r5 and r6.

toolex[nn;nfield] tooltip[nn;nfield]	Si applicano a variabile r di tipologia stringa ed interpretano la stringa valore. Arguments: nn = indice di variabile r nfield = indice di campo (vedi oltre). Gli argomenti possono avere impostazione parametrica. Le funzioni restituiscono: toolex: il valore del campo nfield risultante nella stringa valore tooltip: 1 se il campo nfield risulta numerico, 0 se la variabile non è assegnata o è numerica o nfield non valido. Un campo è riconosciuto: numerico (senza segno e senza decimali) o non numerico. Per entrambe le funzioni è gestito il caso particolare nfield = 0: restituiscono il numero di campi riconosciuti in stringa valore. *Formato notevole* Se il primo argomento della funzione è parametrico ed ha formato notevole "rn" o

	"r\name": la funzione opera direttamente sulla variabile rn. Non è utilizzabile: • in assegnazione di variabile 'o', 'v' • in assegnazione di funzioni custom. Situazioni di errore: • 112: utilizzo di variabile r in contesto non valido; • 123: numero di operandi =0; • 124: numero di operandi #2; • 117: indice di variabile r non valido. Esempio: r5="12|25;64" **toolex[5]=5** Numero campi riconosciuti = 5 **tooltip[5;1]=1** **toolex[5;1]=12** 1° campo valore = 12 campo numerico **tooltip[5;2]=0** **toolex[5;2]=124** 2° campo valore = decimale di '|' = 124 campo non numerico .... 3° campo valore = 25 campo numerico 4° campo valore = decimale di ';' = 59 campo non numerico 5° campo valore = 64 campo numerico n-esimo° campo (n > 5) valore = 0 campo non numerico. L'esempio riportato indica come la funzione può essere utilizzata per interpretare una programmazione corripondente a maschera di utensili (formalismo simile a CNC90).

11.6.7 Logical Functions

ifelse[nc;n1;n2]	Minimum ternary operator: • it returns n1 if nc is different from zero, • otherwise it returns n2. The arguments of this function must be 3. The equality comparison between *nc* and the zero (0) value is evaluated at less than *epsilon* = 0.001. Usable: always. Error conditions: • 123: number of operands =0; • 124: number of operands #3; The function always evaluates both *n1* and *n2*, although it returns only one of the two values. In any case, a few particular conditions are filtered with reference to mathematical errors which may occur in the evaluation of the term not returned. Specifically, the following are not considered errors: • 125: null denominator, in execution of divisions or i*nv* function; • 127: negative argument, in *sqrt* function; • 126: argument outside the range of values (- 1; 1), in *asin*, *acos* functions. • 128: argument *ne* <0 or >10, in *pown* function. Example: ifelse[50;100;l/2] = 100 ifelse[0;100;l/2] = l/2
ifcase[nc1;nesp;nc2; n1;n2]	Complete ternary operator: it evaluates the (*nc1* ? *nc2*) condition, with ? =*nesp*: • if the condition is verified, that is it is true, it returns *n1*, • otherwise *n2*. The function returns *n2* although an invalid value is assigned to *nesp*.

	The arguments of this function must be 5. The *nesp* argument is interpreted to assign the condition between nc1 and nc2: • Value 0 corresponds to < (less than) • Value 1 corresponds to <= (less than or equal to) • Value 2 corresponds to > (greater than) • value 3 corresponds to >= (greater than or equal to) • Value 4 corresponds to = (equal) • Value 5 corresponds to <> (not equal). For the *nesp* argument it is also possible to assign the corresponding symbolic forms, instead of the numeric value. Example: the "greater than or equal to" condition can be set to value 3 or as ">=". The inequality relation can be expressed as "<>" or as "#". The comparison condition between *nc1* and *nc2* is evaluated at less than *epsilon* = 0.001*.* Usable: always. Error conditions: • 123: number of operands =0; • 124: number of operands #5; The function always evaluates both *n1* and *n2*, although it returns only one of the two values. In any case, the same particular conditions listed above for the *ifcase* function are filtered with reference to mathematical errors which may occur in the evaluation of the term not returned. Examples ifcase[5; >=;12;3;25] = 25 ifcase[5;<;12;3;25] = 3 ifcase[5;<> ;12;3;25] = 3
case[nc;nc1:nv1;nc2:nv2;.;nvdef]	Condition test operator: it evaluates the (nc = nc1), (nc = nc2) conditions, by returning the "nv" value assigned to the first verified condition. It tests if a condition, among those assigned, is verified, by returning the value assigned to the condition evaluated to true. The arguments are the following: • nc : value to evaluate • nc1 : first value of comparison with *nc* • nv : value returned by the function if *nc = nc1* • nvdef : default value returned if no equality has been detected. It is not a compulsory argument (if it is not assigned: it is set to 0); if assigned, it must be typed as last argument. The separator character between *nc* and *nc1* is **;** , while that between *nc1* and *nv1* is compulsory **:** . The maximum number of handled cases is 10, *nvdef* included. All arguments can be numeric or string values. The comparison condition between *nc* and the assigned *nc** values is evaluated at less than *epsilon* = 0.001*.* Usable: always. Error conditions: • 123: number of operands =0; • 124: number of operands <2 or >11. The function always evaluates *nv1, nv2…. nvdef*, although it returns only one of the two values. In any case, the same particular conditions listed above for the *ifcase* function are filtered with reference to mathematical errors which may occur in the evaluation of the terms not returned. Example case[h;100:r0;200:h-100;l:l/2;h]

	• if h=100 the (r0) variable value is returned • if h=200 the value resulting from (h-100) is returned • if h=l the value resulting from (l/2) is returned • if no equality has been detected the (h) value is returned.
not[nc]	Argument negation operator. If nc=0 returns 1, if nc#0 (different from 0), returns 0. The equality comparison between nc and the zero (0) value is evaluated at less than epsilon = 0.001.

11.6.8 Technological Functions

Data of a plant are defined according to well determined structures, that can be similar to matrices. Therefore, it is generally possible to enter each single parameter. The definition of a machine includes:

- a first parameters group for movements assignments (axes, working area limits), working areas, head groups setting
- each head group must setup the available spindles/electro spindles (correctors, orientations, speed, fitted out with
  tool or toolholder)
- a tools catalog
- a toolholder catalog (each toolholder can hold one or more tools).

Each parameter

- can be generally entered through a numerical reference (type). E.g., see the function *prtool,* where *nkind* must specify the parameter type. For the parameter of remarkable interest, *nkind* can show a symbolic name containing the following formalism:
  - fix part "p\";
  - variable part: the assigned symbolic name for the parameter. The combination of parameter's symbolic name and numerical reference (typology) occurs automatically.
- can be in some cases entered through a features indication. See function *prfi* that read directly the parameter assigning the diameter of a tool.
- can be addressed in an absolute way as a matrix cell by indication of row and column. Functions (prmxmac, prmxgru,…) enter every single parameter in this way. These functions require an excellent knowledge of the structures of machines parametrics. Therefore their use is destined for developers.

Technological parameters assigned with symbolic formalism

p\gron	Type of parameter enabling a head group
p\face	Type of parameter assigning working face/faces of a spindle
p\ofx p\ofy p\ofz	Parameter types: • offset (x/y/z) of a group • offset (x/y/z) of a spindle
p\xmax p\xmin	Types of parameters of the min. and max. positioning of a head group on the X-axis.
p\ymax p\ymin	Types of parameters of the min. and max. positioning of a head group on the Y-axis.
p\zmax p\zmin	Types of parameters of the min. and max. positioning of a head group on the Z-axis.
p\cmax p\cmin	Types of parameters of the min. and max. positioning of a head group on the C-axis.

p\betamax p\betamin	Types of parameters of the min. and max. positioning of a head group on the Beta-axis.
p\attr	Type of outfit parameter of a spindle or of a toolholder position.
p\fitool	Type of parameter of a tool diameter
p\tiptool	Type of parameter of a tool
p\lltool	Type of working length of a tool
p\ltottool	Type of total length of a tool
p\lauxtool	Type of auxiliary length of a tool
p\ariatool	Type of parameter of tool clearing quote
p\feedmin	Type of parameter of min. working speed of a tool
p\feedmax	Type of parameter of max. working speed of a tool
p\feed	Type of parameter of default working speed of a tool
p\rpmmin	Type of parameter of min. rotation speed of a tool
p\rpmmax	Type of parameter of max. rotation speed of a tool
p\rpm	Type of parameter of default rotation speed of a tool

Access Functions To a Generic Plant Group

They should be considered as functions of advanced programming.

primp[nkind; (vdef)]	Returns a generic Plant group parameter: • nkind = parameter type (this item is compulsory) • vdef = default value (in case of parameter not found). If unset or empty, the default value is 0 If nkind is =0 (worthless), the function returns the machine number set up in the plant. If kind is =0 (worthless), the function returns the number of the configured machines in the plant. Usable: always. Error conditions: • 123: number of operands =0; • 124 : number of operands >2.; • 130: *nkind* argument omitted (empty assignment=. Example: primp[1100]: returns the parameter value 1100 (0 if parameter not found)

Access Functions To a Machine Level For the Configuration Of Head Groups

They should be considered as functions of advanced programming.

prmac[(nm); nkind; (vdef)]	Returns a generic Machine parameter: • nm = machine (this item is compulsory) (in case of empty assignment: the default value is 1) • kind = parameter type (this item is compulsory) • vdef = default value (in case of not found parameter). The max. value that can be set for nm is given by the configuration of the plant technology. If it is nkind=0 (worthless), the function returns 1 (different from 0), if the machine is configured and present in the plant. Usable: always. Error conditions:

	• 123: number of operands =0; • 124: number of operands <2 or >3; • 130: *nkind* argument omitted (empty assignment). Examples • prmac[2;1100]: this function returns the parameter value (1100) of machine 2 {0 in case of not found parameter} • prmac[;1100;100]: this function returns the parameter value (1100) of machine 1 (default machine) {100 in case of not found parameter) • prmac[2;0]: returns 1 if the machine 2 is configured
prgr[(nm); (ng); nkind; (vdef)]	Returns a generic head group parameter: • nm = machine (this item is compulsory) (in case of empty assignment: the default value is 1) • ng = head group (this item is compulsory) (in case of empty assignment: the default value is 1) • kind = parameter type (this item is compulsory) • vdef = default value (in case of not found parameter). If unset or empty, the default value is 0. The max. usable values for nm and ng are given by the configuration of the plant technologies and machine. If it is nKind =0 (worthless), the function is brought back to 1 (different from 0) if the group is configured and present. Usable: always. Error conditions: • 123: number of operands =0; • 124: number of operands <3 or >4; • 130: *nkind* argument omitted (empty assignment). Examples • prgr[2;3;1100]: this function returns the parameter value (1100) of group 3 of machine 2 {0 if parameter not found} • prgr[2;;1100;100]: this function returns the parameter value (1100) of group 1 (default group) of machine 2 {100 if parameter not found) • prgr[2;3;p\ofx]: returns the value (x offset of the group) of the parameter (p\ofx) of group 3 of machine 2.

Tool Access Functions

prface[(nm);(ng);(np)(ns); nt;(side)] prface[(nm),(ng);(nt);side] prface[ng; nt; nside] prface[nt; nside]	Tests if the tool can work on the inside face: • nm = machine (this item is compulsory) (in case of empty assignment: the default value is 1) • ng = head group (this item is compulsory) (in case of empty assignment: the default value is 1) • np = if the outfit found by (ns;nt) identifies a toolholder, the parameter np shows the position of the tool fitted out in the toolholder. • ns = spindle (compulsory) • nt = tool/toolholder to be used on the ns spindle or spindle. (if *nt* is not assigned, value 0 is imposed) • nside = face (if left out or assigned as empty sets off *nside* = current face; if assigned, it interprets the face custom number). The function returns value 1 if the test is verified, otherwise it returns 0. Some particular cases can be examined, such as: (ns<=0; nt=0)

In this case any valid technology is not shown. The function returns value 0;

(ns>0; nt=0)

The *ns* spindle is fitted out as configured in the head group. In particular:

- if *ns* is not fitted out (this is a position of electrospindle with tool change): verification is made according with the information on the working face assigned to the electrospindle
- if *ns* is fitted out with a toolholder, *np* displays the tool position on the toolholder (by default if no=0: first point).

  Verification is only made on *ns* configuration.

(ns<=0; nt#0)

The spindle is now displayed in *nt* and it is fitted out as shown in the head group configuration (see: (ns>0; nt=0, with spindle now in *nt*).

(ns>0; nt#0)

If *nt* has a significant value (in a valid range tool or toolholder range), the spindle *ns* is considered fitted out with *nt.* In particular, if *nt* displays a toolholder, *np* displays the tool position on the toolholder.

In case of outfit on a toolholder:

- if *np* is not assigned: it is verified on the first fitted out tool.
- if *np* is assigned as not valid (<=0 or as well as the max. value allowed), the function is brought back to the value 0.

The max. usable values for (*nm, ng, np,ns, nt*) are given by the technologies configurations of plant, machine, group, tools and toolholders catalog. Manner of fitting out a tool rather than a toolholder also depends on technologies configuration.

The *nside* face number assigns the z-axis plane and orientation.

The equivalent forms of the reduced formats are as follows:

prface[nm;ng;nt;side]->prface[nm;ng;0;-1;nt;side]

prface[nm;ng;nt;side]->prface[1;ng;0;-1;nt;side]

prface[nt; nside] -> prface[1;1;0;-1;nt;side].

Usable: always.

Error conditions:

- 123: number of operands =0;
- 124: number of operands <3 or >6,
- 130: arguments *ns* and *nt* both left out (empty assignation)

Examples

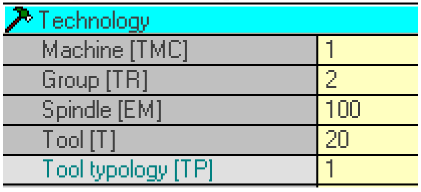
prface[;;;;90]: head if the spindle identified as (nm=1; ng=1; np=0; ns=1; nt=90) can work on the active face.

With reference to the technological program of a working, following correspondences to the parameters of the

prface are shown in order to check the work face in case of:

programming of technology for spindle and tool:

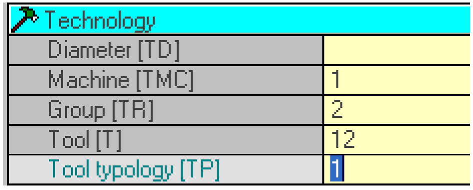
prface[1;2;0;100;20]



Technology	
Machine [TMC]	1
Group [TR]	2
Spindle [EM]	100
Tool [T]	20
Tool typology [TP]	1

- programming of technology for spindle and tool:

prface[1;2;0;100;20]

	• programming of technology for spindle where two equivalent forms are possible: prface[1;2;0;0;12] prface[1;2;0;12;0]  Technology Diameter [TD] Machine [TMC] 1 Group [TR] 2 Tool [T] 12 Tool typology [TP] 1
prfi[(nm);(ng);(np);(ns); nt] prfi[(nm);(ng);nt] prfi[ng;nt] prfi[nt]	Returns the diameter of the tool: • nm = machine (this item is compulsory) (in case of empty assignment: the default value is 1) • ng = head group (this item is compulsory) (in case of empty assignment: the default value is 1) • np= if the tooling found by (ns;nt) specifies a toolholder, the parameter *np* displays the position of the tool as fitted out on the toolholder. • ns = spindle (compulsory) • nt = tool/toolholder to fit out on *ns* spindle or spindle. (If *nt* is not assigned a value 0 is assigned) Two particular cases can be examined: (ns<=0; nt=0) a valid technology is not displayed - the function is brought back to 0.0 (ns>0; nt=0) The *ns* spindle is fitted out as configured in the head group. In particular, • if *ns* is not fitted out (this is a position of electrospindle with tool change): the function returns the value 0.0; • if *ns* is fitted out with a toolholder, *np* displays the tool position on the toolholder (by default, if np=0: first point) (ns<=0; nt#0) The spindle is now displayed in *nt* and fitted out as configured in the head group (see case: (ns>0; nt=0) with spindle now in *nt).* If *nt* identifies correctly a tool or a toolholder, this function reads the diameter for the tool or the toolholder. In particular: if *nt* displays a toolholder, *np* displays the tool position in the toolholder (by default if np=0: first point); The max. usable values for (*nm, ng, np, ns, nt*) are given by the technologies configurations of plant, machine, group, tools and toolholders catalog. Manner of fitting out a tool rather than a toolholder also depends on technologies configuration. The equivalent forms of the reduced formats are as follows: prfi[nm;ng;nt] -> prfi[nm;ng;0;-1;nt] prfi [ng; nt] -> prfi[1;ng;0;-1;nt] prfi [nt] -> prfi[1;1;0;-1;nt] Usable: always. Error conditions: • 123: number of operands =0; • 124: number of operands <3 or >5; • 130: arguments nt and ns both left out (empty assigment) Examples

	prfi[1;1;90]: returns the diameter of the spindle identified as (nm=1; ng=1; np=0; ns=1; nt=90)
prtool[(nm);(ng);(np);ns;(nt);nkind; (vdef)] prtool[(nm);(ng);nt;nkind;(vdef)] prtool[nm; ng; nt; nkind] prtool[ng; nt; nkind] prtool[nt; nkind]	Returns a generic tool parameter: • nm = machine (this item is compulsory) (in case of empty assignment: the default value is 1) • ng = head group (this item is compulsory) (in case of empty assignment: the default value is 1) • if the tooling found by *(ns;nt*) specifies a toolholder, the *np* parameter displays the position of the tool as fitted out on the toolholder. • ns = spindle (compulsory) • nt = tool/toolholder to fit out on the *ns* spindle or spindle. (If *nt* is not assigned, the value 0 is assigned) . • nkind = parameter type (this item is compulsory) • vdef = default value (returned if parameter not found).(if vdef is not assigned , the value 0 is assigned). By particular cases: (ns<=0; nt=0) can be considered. If a valid technology is not displayed, the function value becomes 0 again; (ns>0; nt=0) The *ns* spindle is fitted out as configured in the head group. In particular, • if *ns* is not fitted out (this is a position of electrospindle with tool change), the function can return a value different from *vdef*, only if the parameter is significant for a spindle position (example of valid parameter: corrector x; examples of not valid parameters: point diameter, point lenght); • if *ns* is fitted out with a toolholder, *np* displays the tool position on the toolholder (by default if *np*=0: first point); (ns<=0; nt#0) The spindle is now displayed in *nt* and fitted out as configured in the head group (see (ns>0; nt=0), with the spindle now in *nt).* (ns>0; nt#0) If *nt* has a significant value (in a valid tool or toolholder range): the spindle *ns* is considered as fitted out with *nt*. In particular: if *nt* shows a toolholder, *np* shows the tool position on the toolholder. In case of outfit on toolholder: • if *np* is assigned as not valid (as well as the the max. allowed value): the function becomes 0 again. In case of outfit on toolholder: • if *np* is not assigned or <= 0 : check is made on the first fitted out tool. • if *np* is assigned as not valid (as well as the max. value allowed): the function becomes 0 again.
	The max usable values for (*nm, ng, np, ns, nt*) are given by the technologies configurations of plant, machine, group, tools and toolholders catalog. Manner of fitting out a tool rather than a toolholder also depends on technologies configuration. As compared to Edicad: the conversion parameter is no longer handled. The equivalent forms of the reduced formats are as follows: prtool[nm;ng;nt;nkind;vdef] àprtool[nm;ng;0;-1;nt;nkind;vdef] prtool[nm;ng;nt;nkind]à prtool[nm;ng;0;-1;nt;nkind;0.0] prtool[ng;nt; nkind] à prtool[1;ng;0;-1;nt;nkind;0.0] prtool[nt;nkind]à prtool[1;1;0;-1;nt;nkind;0.0].

	Usable: always. Error conditions: • 123: number of operands =0; • 124: number of operands <2 or >7; • 130: nt and ns arguments both left out or nkind left out (empty assignment). examples: *prtool*[1;2;;100;3;100]: returns the value of the tool parameter (100) identified as (nm=1; ng=2; np=0;ns=100, nr=3). It returns 0 if the parameter required is not found. *prtool*[1;1;90;p\fitool]: reads the spindle diameter in position (nm=1; ng=1; nt=90). The p\fitool parameter can be replaced by the value 1002. *prtool*[1;1;90;p\tiptool]: reads the value of spindle type in position (nm=1; ng=1; nt=90). The p\fitool parameter can be replaced by the value 1001. See following function prtip[ ].
prtip[(nm);(ng);(np);ns;(nt)] prtip [nm; ng; nt] prtip [ng; nt] prtip [nt]	Returns the type of a tool: • nm=machine (compulsory) (in case of empty assignment: it uses =1) • ng=head group (compulsory) (in case of empty assignment: it uses =1) • np= if the outfit resulting from (*ns;nt*) defines a toolholder, it shows the tool fitted out on the toolholder. • ns=spindle (compulsory) • nt=tool/toolholder to fit out on the *ns* spindle or spindle (in case of empty assignment: it uses =0) The same particular cases examined for the *prtoo* function can be considered. The equivalent forms of the reduced formats are as follows: prtip[nm; ng; nt] ->prtip[nm;ng;0;-1;nt] prtip[ng; nt]->prtip[1;ng;0;-1;nt] prtip[nt]->prtip[1;1;0;-1;nt]. Usable: always. Error conditions: • 123: number of operands =0; • 124: number of operands <1 or >5; • 130: nt and ns arguments both left out.
prfulcrox [(nm);(ng);(np);ns;(nt)] prfulcroy [(nm);(ng);(np);ns;(nt)] prfulcroz [(nm);(ng);(np);ns;(nt)]	Return the (x, y, z) position of the selected tool pivot point, in condition of machine set to 0: • nm = machine (this item is compulsory) (in case of empty assignment: the default value is 1) • ng = head group (this item is compulsory) (in case of empty assignment: the default value is 1) • np=if the outfit found by (ns;nt) specifies a toolholder, the *np* parameter shows the position of the tool fitted out on the toolholder. • ns = spindle (compulsory) • nt = tool/toolholder to fit out on the *ns* spindle or spindle • (in case of empty assignment: is uses =0) Particular cases can be considered: (ns<=0; nt=0) A valid technology is not shown: the function becomes as value 0.0 again. (ns>0; nt=0) The *ns* spindle is fitted out as configured in the head group. In particular: • if *ns* is not fitted out (this is a position of electrospindle with tool change),

	the function returns the value of the spindle's fulcrum; - if *ns* is fitted out with a toolholder, *np* shows the tool position on the toolholder (by default if *np*=0: first point); (ns<=0; nt#0) The spindle is now displayed in *nt* and fitted out as configured in the head group (see the case: (ns>0; nt=0), with spindle now in *nt)*. (ns>0; nt#0) If *nt* has a significant value (in a valid tool or toolholder range): the spindle *ns* is considered as fitted out with *nt*. In particular: if *nt* shows a toolholder, *np* shows the tool position on the toolholder. In case of outfit on toolholder: - if *np* is not assigned or <=0: it evaluates for the first assigned tool; - if *np* is assigned as not valid (as well as the max. value allowed): the function becomes 0 again. The max. usable values for (*nm, ng, np, ns, nt*) are given by the technologies configurations of plant, machine, group, tools and toolholders catalog. Manner of fitting out a tool rather than a toolholder also depends on technologies configuration. Usable: always. Error conditions: - 123: number of operands =0; - 124: number of operands #5; - 130: nt and ns arguments both left out. (empty assignment)

Function of direct access to matrix plants

For example, these functions can be used to write sub-routines and/or macro-programs, to evaluate details of the plant technology and should be considered functions of advanced programming.

prmxmac[(nm);nrow;ncol,(vdef)]	Returns the value of a matrix generic cell of machine configuration: - nm = machine (compulsory) (in case of empty assignment: it uses =1) - nrow = matrix row (compulsory) (significant from value 1) - ncol = matrix column (compulsory) (compulsory) (significant from value 1) - vdef = default value (returned if the parameter is not found) If not set or set up as empty: it uses =0 Usable: always. Error conditions: - 123: number of operands =0; - 124: number of operands different from 3,4; - 130: nrow argument or left out ncol (empty assignment).
prmxgru[(nm);(ng);nrow; ncol,(vdef)]	Returns the value of a matrix generic cell of groups configuration: - nm = machine (compulsory) (in case of empty assignment: it uses =1) - ng = head group (compulsory) (significant from 1) - nrow = matrix row corresponding to ng group (compulsory) (significant from 1) - ncol = matrix column (compulsory) (significant from 1) - vdef = (returned if the parameter is not found). If not set or set as empty: it uses =0.

	Usable: always. Error conditions: • 123: number of operands =0; • 124: number of operands different from 4 and 5; • 130: nrow argument or left out ncol (empty assignment).
prmxtool[(nm);ntool;(nkind);(ncol);(vdef)]	Returns the value of a generic information from the matrix of the tool catalog: • nm = machine (compulsory) (in case of empty assignment: it uses =1) • ntool = tool number (compulsory) • nkind = parameter type • ncol = matrix column (significant from 1) • (returned if the parameter is not found). If not set or set as empty: it uses =0 If ntool is worthless, following two cases can be distinguished: • ncol>=0: the function returns the maximum identification number of the configured tools in the tool catalogue. • ncol<0: as above; however, the identification number is assigned now with a numeration number (only where allowed by parametric cases) If ntool has assigned a valid worth, it selects a tool. Following cases can be distinguished: • nkind =0, ncol =0 (worthless): the function returns 1 (different from 0) if the ntool tool is configured; • nkind different from 0: it reads the parameter with nkind typology (nkind: considered in absolute value), assigned for the ntool tool; • nkind =0, ncol different from 0: it reads the parameter in the column ncol (ncol: considered in absolute value), assigned for the ntool tool. If a negative value is assigned to ntool, it selects a tool, that is identified by a numeration parameter. There are the cases already shown for **ntool** positive. Error conditions: • 123: number of operands =0; • 124: number of operands < 2 or > 5; • 130: nrow argument or left out ncol (empty assignment). Examples: prmxtool[1;0;0;0]: returns the'the maximum identification number of the configured tools in the tool catalogue of the machine 1 prmxtool[1;3;p\fitool]: Returns the diameter of the'tool 3 of the tool catalogue of the machine 1 prmxtool[1;3;0;6]: returns the parameter of column 6 of the'tool 3 of the tool catalogue of the machine 1
prmxhtool[(nm);ntool;(nrow);(nkind);(ncol);(vdef)]	Returns the value of a generic information from toolholder matrix (catalog): • nm = machine (compulsory) (in case of empty assignment: it uses =1) • ntool = toolholder number (compulsory) (significant from 1) • nrow = toolholder row (compulsory) (significant from 0) • nkind = parameter type • ncol = matrix column (significant from 1) • vdef = value by default (in case of not accessible cell) - (if not set or set as empty : it uses =0) If ntool is worthless, the function returns the toolholders number, that are

	configured in the toolholder catalog. If ntool is not worthless: it selects a toolholder. Following two cases can be distinguished: • nkind =0, ncol =0 (worthless): the function returns 1 (different from 0) if the toolholder ntool is configured; • nkind different from 0: it reads the parameter of type nkind (nkind: considered in absolute value), assigned or the ntool tool to the configuration row nrow; • nkind =0, ncol different from 0: it reads the parameter in the ncol column (ncol: considered in absolute value), assigned for the ntool tool in the row configuration nrow. If it is nrow=0: shows the configuration row of the toolholder. If it is nrow>0: shows the configuration row of the nrow-th fitted out tool (significant for the max. number of the tools, that can be fitted out). Usable: always. Error conditions: • 123: number of operands =0; • 124: number of operands < 2 o > 6; • 130: ntool, nrow argument or left out ncol (empty assignment) Examples: prmxhtool[1;0]: returns the number of the tool-holders configured for the machine 1 prmxhtool[1;3]: returns 1 if the tool-holder 3 of machine 1 is configured prmxhtool[1;3;0;6]: returns the parameter defined in the column 6 of the configuration row of the tool-holder 3 of the machine 1 prmxhtool[1;3;4;p\ofx]: returns the parameter correction x of the point 4 of the tool-holder 3 of machine 1 prmxhtool[1;3;4;;37]: returns the column parameter 37 of the point 4 of the tool-holder 3 of machine 1
prmxstore[(nm);nstore;(nrow);(nkind);(ncol),(vdef)]	Returns the value contained in a generic matrix configuration cell of a carousel: • nm = machine (compulsory) (in case of emtpy assignment: it uses =1) • nstore = carousel number (significant from 1) • nrow = matrix row of the carousel defined by nstore (compulsory) (significant from 0) • nkind = parameter type • ncol = matrix column (significant from 1) • vdef = value by default (in case of not accessible cell) - (if not set or set as empty : it uses =0) If the nstore parameter is worthless, the function returns the number of the configured carousels. If the nrow parameter is different from 0 follwing cases can be distinguished: • nkind =0, ncol =0: the function returns 1 (different from 0) if the position defined by nrow is fitted out . • nkind different from #0: returns the parameter with nkind typology (the value of nkind is considered in absolute value) assigned for the nrow position. • nkind=0, ncol different from 0: it reads the parameter in the ncol column (ncol: considered in absolute value), and nrow row Usable: always.

	<u>Error conditions</u>: • 123: number of operands =0; • 124: number of operands < 2 o > 6; • 130: nstore argument left out (empty assignment) <u>Examples:</u> prmxstore[1;0]: returns the number of the tool collectors configured for the machine 1 prmxstore [1;3]: returns the maximum position equipped for the tool collector 3 of machine 1 prmxstore [1;3;4]: returns 1 if the position 4 of the tool collector 3 of machine 1 is equipped. prmxstore [1;3;4;1200]: returns the parameter of typology 1200 from the position 4 of the tool-holder 3 of machine 1 prmxstore [1;3;4;0;12]: returns the parameter given in the column 12 from the position 4 of the tool-holder 3 of machine 1

11.6.9 Multi-Purpose Geometry Library Functions

The first function argument specifies the remarkable name of the selected geometric function. Later on it is written in bold face and each case is described separately. For all examined cases the following considerations must be taken into account:

<u>Usable</u>:

- always, in the versions that do not use working names
- The versions using working names cannot be used, when 'o and 'v variables, variable geometries (fictive faces edges), custom functions are assigned.

<u>Error conditions</u>:

- 123: number of operands =0;
- 124: wrong number of operands.
- 116: invalid context in those versions that use working names.

No geometric error conditions have been detected; in any case a default condition is assumed.

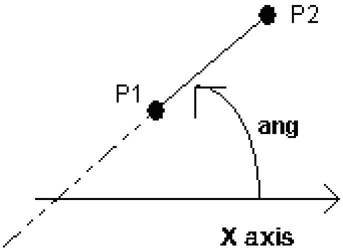
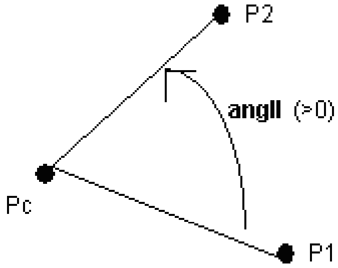
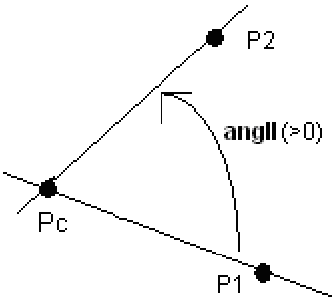
Function versions that use working names search the version indicated before the current working. Search is broken at the first correspondence. The working name is indicated with the formalism "wname" and must be placed between double quotation marks.

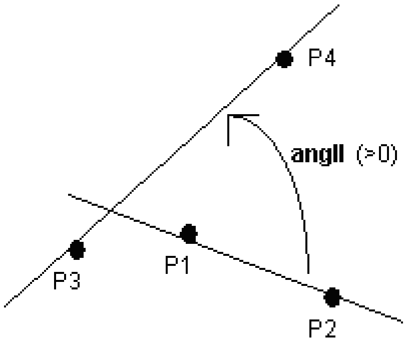
A term can be summed to the search. Following syntax types are recognized:

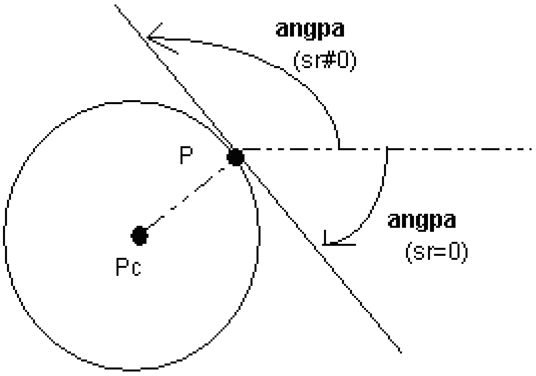
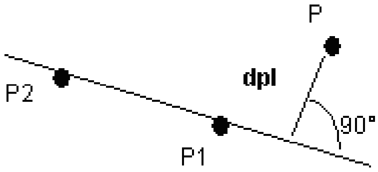
- “wname+2”: shows that the searched working is located two lines after "wname";
- “wname-2”: shows that the searched working is located two lines before "wname";
- “wname+”;nn: where the displacement is assigned in an added argument (in parametric form, as well). “wname-” can also be used.

The functions should be partly considered as functions of advanced programming.

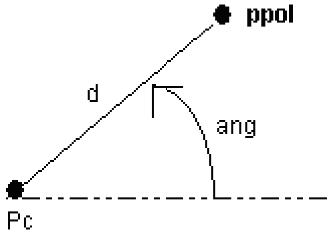
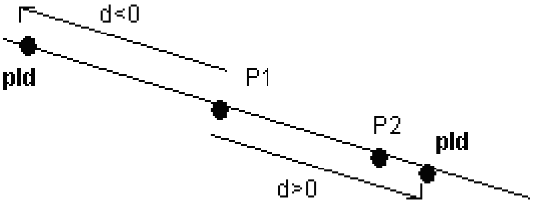
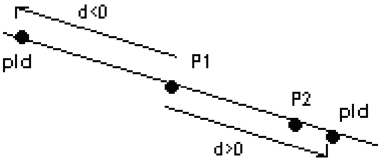
geo[**angc**;ang;(sgn)]	Returns the *ang* angle reduced to: 0° – 360°. ang = angle sgn = if value different from 0, the *ang* original sign is preserved (in case of empty or unassigned field: the default value is =0) <u>Examples</u> *geo[angc*;4500] = 180
geo[**ang**;x1;y1;x2; y2]	Returns the angle between the x axis and the oriented line P1-P2: x1;y1 = abscissa and ordinate of point P1 x2;y2 = abscissa and ordinate of point P2

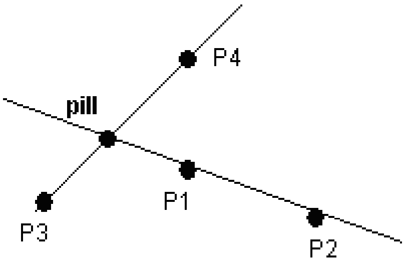
	 The angle value is included in the range [0 <= *ang* < 360], in units of degrees [°]. If the two points P1 and P2 coincide: the function returns 0. <u>Examples</u> *geo[ang*;100;100;400;400] = 45
geo[**ang**; "wname"] geo[**ang**; "wname+nn"] geo[**ang**; "wname", nn"]	The function is similar to the function geo[**ang**;x1;y1;x2;y2]. The oriented line P1-P2 is defined by the working name. Working *"wname+nn"* must correspond to a linear segment, that is assigned with one only line. If the working is not correctly defined, the function returns the value 0.0.
geo[**angll**;xc;yc;x1; y1;x2;y2]	Returns the angle between the oriented lines Pc-P1 and Pc-P2 (Pc=center) xc;yc = abscissa and ordinate of point Pc x1;y1 = abscissa and ordinate of point P1 x2;y2 = abscissa and ordinate of point P2. The angle value is included in the range [–180 < angll <= +180], in units of degrees [°]: positive: if the line Pc-P1 closes on the line Pc-P2 in counterclockwise direction; negative: otherwise;  If the two points P1 and P2 coincide: the angle value returned is 0; If point Pc is not distinct from P1 or P2: the angle value returned is always 0. <u>Examples</u> *geo[angll;100;100;200;0;400;100]=* 45
geo[**angll**;xc;yc;zc; x1;y1;z1;x2;y2;z2]	Returns the angle between the oriented lines Pc-P1 and Pc-P2 (Pc=centre) in the space:  xc;yc;zc = coordinates of point Pc x1;y1;z1 = coordinates of point P1

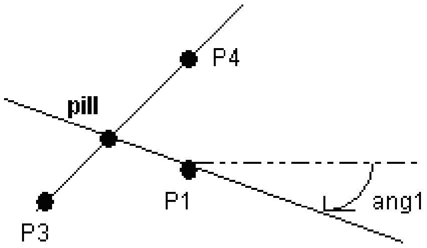
	x2;y2;z2 = coordinates of point P2 The value of the angle is included in the range [–180 < *angll* <= +180], in units of degrees [°]. If the points P1 and P2 coincide, the value of the angle is 0. If the Pc point is not distinct from P1 or P2, 0.0 is returned anyway. <u>Examples</u> *geo[angll;400;0;-100;400;0;0;450;-20;0]* = 28.3
geo[**angll**;x1;y1;x2;y2;x3;y3;x4;y4]	Returns the angle between the oriented lines P1-P2 and P3-P4 x1;y1 = abscissa and ordinate of point P1 x2;y2 = abscissa and ordinate of point P2 x3;y3 = abscissa and ordinate of point P3 x4;y4 = abscissa and ordinate of point P4. The angle value is included in the range [–180 < angll <= +180], in units of degrees [°]: positive: if the line P1-P2 closes on the line P3-P4 in counterclockwise direction; negative: otherwise;  If the two lines are parallel or coincident, the function returns the value 0. <u>Examples</u> *geo[angll*;100;100;200;0;0;100;400;100] = 45
geo[**angll;**"wname1+nn",x3;y3;x4;y4] geo[**angll;**"wname1+nn","wname2+nn"	This function is similar to the function geo[**angll**;x1;y1;x2;y2;x3;y3;x4;y4]. It returns the angle between the oriented lines, where the first or both the lines are defined by a working name "wname1"= name of the working assigning the first line "wname2"= name of the working assigning the second line "wname1" and "wname2" workings must define the linear segments made of one only line. If the workings are not correctly defined, the function returns the value 0.0.
geo[**angpc**;x;y;xc;yc;(sr)]	Returns the tangency angle of a point P on circle: x;y = abscissa and ordinate of point P xc;yc = abscissa and ordinate of the circle center Pc sr = direction of rotation on circle (0=clockwise, #0 counterclockwise). In case of invalid or unassigned field value, the default angle value is 0. The angle value is included in the range [0 <= *ang* <= 360], in units of degrees [°]. In case the two points P and Pc coincide: the function returns the value 0.

	 Examples *geo[angpc*;0;100;0;0;1] = 180
geo [**angpc**;"wname+nn"]	This function is similar to the function geo[**angpc**;x;y;xc;yc;(sr)]. It returns the tangent angle of a Point P on an circle arc. Working "wname" must define an arc on xy plane made of one only line. The point P is the final point of the arc. If the working is not correctly defined, the function returns the value 0.0.
geo[**dist**;x1;y1;x2; y2]	Returns the distance between the two points P1 and P2 (in plane): x1;y1 = abscissa and ordinate of point P1 x2;y2 = abscissa and ordinate of point P2. Examples *geo[dist*;0;100;100;-200] = 316.2278
geo[**dist**;x1;y1;z1; x2;y2;z2]	Returns the distance between the two points P1 and P2 (in space): x1;y1;z1 = abscissa, ordinate and height of point P1 x2;y2;z2 = abscissa, ordinate and height of point P2. Examples *geo[dist*;0;100;10;100;-200;-10] = 316.8596
geo[**dpl**;x;y;x1;y1; x2;y2]	Returns the distance of point P from the line P1-P2: x;y = abscissa and ordinate of point P x1;y1 = abscissa and ordinate of point P1 x2;y2 = abscissa and ordinate of point P2.  The distance is null if point P belongs to the line. Examples *geo[dpl*;0;200;0.;0;100;100] = 141.4214 *geo[dpl*;50;50;0.;0;100;100] = 0 <- the point (50;50) is on the line
geo[**dpl**; "wname1 +nn",x1;y1;x2;y2] geo[**dpl;** x; y;"wname2+nn"] geo[**dpl;**"wname1 +nn","wname2+nn"	This function is similar to the function geo[**dpl**;x;y;x1;y1;x2;y2]. It returns the distance of the point P from the line P1-P2. The point and/or the line can be determined by a working name. "wname1"= name of the working assigning the point P. It may correspond to a point or to a setup. In a profile line (line or arc) the point P is the final point of the line. "wname2"= name of the working assigning the linear segment. It must define a linear segment made of one only line. If the working is not correctly defined, the function returns the value 0.0.

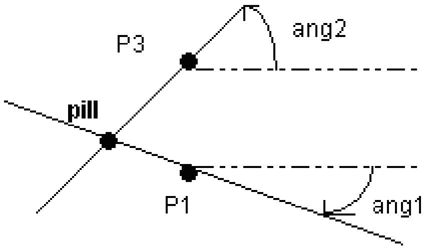
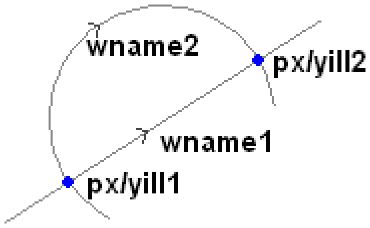
geo[**pxpol**;xc;yc;
ang;d]
geo[**pypol**;xc;yc;
	Return the abscissa (x) or the ordinate (y) of the point assigned with polar modes:
	xc;yc = abscissa and ordinate of point Pc (origin of the polar system)

ang;d]	ang = angle d = vector (it is applied in absolute value).  If the vector d is null: the function returns the initial coordinate of point Pc. <u>Examples</u> *geo[pxpol*;0;0;30;100] = 86.6025 *geo[pypol*;0;0;30;100] = 50
geo[**pxld**;x1;y1;x2;y2;d] geo[**pyld**;x1;y1;x2;y2;d]	Return the abscissa (x) or the ordinate (y) of the point on the line P1-P2 (assigned on the plane), at distance d from P1: x1;y1 = abscissa and ordinate of point P1 x2;y2 = abscissa and ordinate of point P2 d = distance. If d>0 (positive): the point is calculated from P1 to P2; If d<0 (negative): the point is calculated from P1 in direction opposite to P2; If d=0 (positive): the point coincides with P1.  <u>Examples</u> *geo[pxld*;0;0;100;0;200] = 200 *geo[pyld*;0;0;100;0;200] = 0
geo[**pxld**;x1;y1;z1;x2;y2;z2;d] geo[**pyld**;x1;y1;z1;x2;y2;z2;d] geo[**pzld**;x1;y1;z1;x2;y2;z2;d]	return the abscissa (x), the ordinate (y) or the coordinate z of the point on the line P1-P2 (assigned in the space), at a distance d from P1.  x1;y1;z1 = abscissa and ordinate of the point P1 x2;y2;z2 = abscissa and ordinate of the point P2 d = distance If d>0 (positive): the point is calculated from P1 to P2; If d<0 (negative): the point is calculated from P1 in direction opposite to P2; IF d=0 (positivE): the point coincedes with P1. If the points P1 and P2 coincide, the coordinate of P1 is returned P1.
geo[**pxld**;"wname+nn",d] geo[**pyld**;"wname+nn",d] geo[**pzld**;"wname+nn",d]	The functions are similar to the functions geo[**pxld**;x1;y1;z1;x2;y2;z2;d], geo[**pyld**;x1;y1;z1;x2;y2;z2;d] e geo[**pzld**;x1;y1;z1;x2;y2;z2;d]. They return the abscissa (x) or the ordinate (y) of the point on the line, that is defined by a working name, at a distance d from P1. The working "wname1" must define a linear segment made of one only line. If the working is not correctly defined, the function returns the value 0.0.

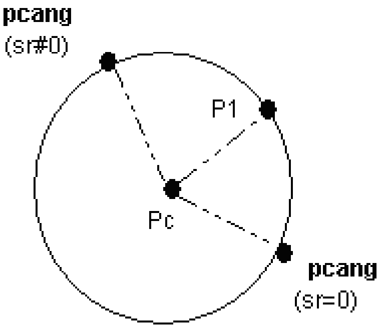
geo[**pxill**;x1;y1;x2;
y2;x3;y3;x4;y4]
geo[**pyill**;x1;y1;x2;
y2;x3;y3;x4;y4]
	Return the abscissa (x) or the ordinate (y) of the intersection point between the two lines P1-P2 and P3-P4:
	x1;y1 = abscissa and ordinate of point P1
	x2;y2 = abscissa and ordinate of point P2
	x3;y3= abscissa and ordinate of point P3
	x4;y4 = abscissa and ordinate of point P4.
	
	Return, in any case, the coordinate of point P1 if:
	one or both lines are assigned with a null segment (P1=P2 and/or P3=P4);
	the two lines are coincident;
	the two lines are parallel.
	
	<u>Examples</u>
	geo[pxill;0;0;300;300;0;300;300;0] = 150

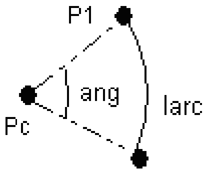
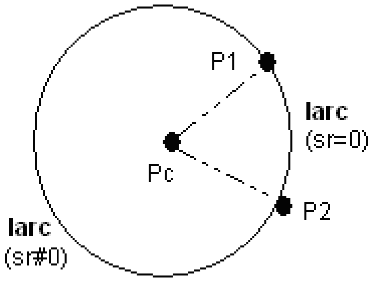
geo[**pxill**;x1;y1;
ang1;x3;y3;x4;y4]
geo[**pyill**;x1;y1;
ang1;x3;y3;x4;y4]
	Return the abscissa (x) or the ordinate (y) of the intersection point between the two lines P1-ang1 and P3-P4:
	x1;y1 = abscissa and ordinate of point P1
	ang1 = inclination angle of the first line
	x3;y3= abscissa and ordinate of point P3
	x4;y4 = abscissa and ordinate of point P4.
	
	Return, in any case, the coordinate of point P1 if:
	the second line is assigned with a null segment (P3=P4);
	the two lines are coincident;
	the two lines are parallel.
	

geo[**pxill**;x1;y1;
ang1;x3;y3;ang2]
geo[**pyill**;x1;y1;
ang1;x3;y3;ang2]
	Return the abscissa (x) or the ordinate (y) of the intersection point between the two lines P1-ang1 and P3-ang2:
	x1;y1 = abscissa and ordinate of point P1
	ang1 = inclination angle of the first line
	x3;y3= abscissa and ordinate of point P3
	ang2 = inclination angle of the second line.

	 Return, in any case, the coordinate of point P1 if: the two lines are coincident; the two lines are parallel.
geo[**pxill**;”wname1+nn”;x3;y3;x4;y4] geo[**pxill**;”wname1+nn”;x3;y3;ang2] geo[**pxill**;”wname1+nn”; ”wname2+nn”] geo[**pxill**;”wname1+nn”; ”wname2+nn”(;nsol)] geo[**pyill**;”wname1+nn”;x3;y3;x4;y4] geo[**pyill**;”wname1+nn”;x3;y3;ang2] geo[**pyill**;”wname1+nn”; ”wname2+nn”] geo[**pxill**;”wname1+nn”; ”wname2+nn”(;nsol)]	The functions are similar to the functions geo[**pxill**;x1;y1;ang1;x3;y3;x4;y4] and geo[**pyill**;x1;y1;ang1;x3;y3;x4;y4]. They return the abscissa (x) or the ordinate (y) of the intersection point between the two lines respectively defined by a "wname1" and "wname2" working name. The workings must define a linear segment or an arc made of only one segment. nsol = in the form with "wname1" and "wname2", if two intersection points are detected, it is possible to indicate which one of the two should be returned. If nsol=1, the first solution is returned, if nsol=2 the second solution is returned. If the variable is not assigned, the default is 1:  In the example displayed in the figure: "wname1" is a linear segment. "wname2" is an arc. Both the segments have 2 intersection points: ▪ l’the intersection which is returned as a default is the one nearest to the point of beginning of the "wname1" segment (corresponding to nsol=1 or not assigned nsol). The second intersection is returned, if a 2 value nsol is assigned. If the workings are not correctly defined, the function returns the value 0.0.
geo[**pxme**;x1;y1;x2;y2] geo[**pyme**;x1;y1;x2;y2]	Return the abscissa (x) or the ordinate (y) of the intermediate point of the segment P1-P2 (in plane): x1;y1 = abscissa and ordinate of point P1 x2;y2 = abscissa and ordinate of point P2.
geo[**pxme**;x1;y1; z1;x2;y2; z2] geo[**pyme**;x1;y1; z1;x2;y2; z2] geo[**pzme**;x1;y1; z1;x2;y2; z2]	Return the abscissa (x), the ordinate (y) or the height (z) of the intermediate point of the segment P1-P2 (in space): x1;y1;z1 = abscissa, ordinate and height of point P1 x2;y2;z2 = abscissa, ordinate and height of point P2.
geo [**pxme**;"wname+nn"] geo [**pyme**;"wname+nn"] geo	The functions are similar to the functions geo[**pxme**;x1;y1; z1;x2;y2; z2], geo [**pyme**;x1;y1; z1;x2;y2; z2], geo[**pzme**;x1;y1; z1;x2;y2; z2]. They return the abscissa (x), the ordinate (y) or the depth (z) of the middle point of the line defined by a working name. The working must define a linear segment or an arc made of one only line. The function calculates the linear distance between the extreme points of the lines. The length of an arc is calculated by the function

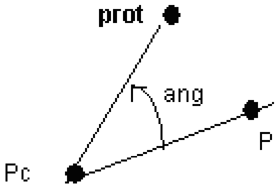
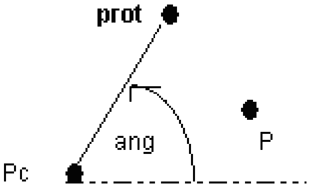
[**pzme**;"wname+nn"]	geo [**larc**;"wname"]. If the working is not correctly defined, the function returns the value 0.0.

geo[**pxcang**;x1;y1;
xc;yc;ang;(sr)]
geo[**pycang**;x1;y1;
xc;yc;ang;(sr)]
	Return the abscissa (x) or the ordinate (y) of a point P of a circle, determined from point P1 and with clockwise or counterclockwise angle of rotation:
	x;y = abscissa and ordinate of point P1
	xc;yc = abscissa and ordinate of the circle center Pc
	ang = angle of rotation
	sr = direction of rotation on circle (0=clockwise, #0 anticlockwise). In case of empty or unassigned field, the default angle value is 0.
	
	If points P1 and Pc coincide: the function returns the initial coordinate of P1.
	
	Examples
	geo[pxcang;0;100;0;0;45] = 70.7107
	geo[pxcang;0;100;0;0;45;1] = -70.7107

geo[**larc**;x1;y1;xc; yc;ang]	Returns the length of the arc of circle, defined from the point P1 and with the size of an angle *ang* :  x1;y1 = abscissa and ordinate of the point P1 xc;yc = abscissa and ordinate of the center PC of the circle ang = size angle If the points P1 and Pc coincide, or if ang=0.0: the value is returned worthless examples *geo[larc*;0;0;100;0;90] = 157.0796
geo[**larc**;x1;y1;x2; y2;xc;yc;(sr)]	Returns the lenght of the circle arc, defined from the point P1 until the point P2, clockwise or anticlockwise:  x1;y1= abscissa and ordinate of the point P1 x2;y2= abscissa and ordinate of the point P2 xc;yc= abscissa and ordinate of the centre PC point of the circle sr = direction of rotation of circle (0=clockwise, different from 0 counterclockwise). In case of empty or not assigned field: it uses =0) If the points P1 and Pc, or P2 and Pc coincide, the value is returned worthless.

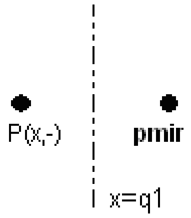
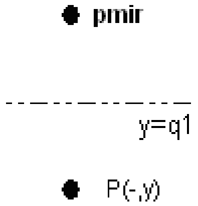
	examples *geo[larc*;0;0;100;100;100;0] = 157.0796 *geo[larc*;0;0;100;100;100;0;1] = 471.238
geo[**larc;** "wname"]	The function is similar to the function geo[**larc**;x1;y1;x2;y2;xc;yc; (sr)]. It returns the length of the arc of the circle defined by a working name. The working mus define an arc lying on a any plane. P1 and P2 are respectively the starting point and the final point of the arc. The length is calculated on the plane of the arc development. If the working is not correctly defined, the function returns the value 0.0.

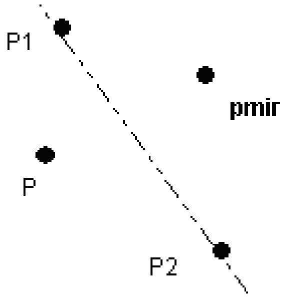
Point rotation functions

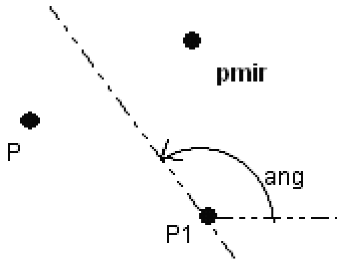
geo[**pxrot**;x;y;xc;yc;ang] geo[**pyrot**;x;y;xc;yc;ang]	Return the abscissa (x) or the ordinate (y) of a point P rotated by the angle ang with center Pc (incremental rotation): x;y = abscissa and ordinate of point P xc;yc = abscissa and ordinate of point Pc ang = incremental rotation angle: If ang>0 (positive): the point rotates in counterclockwise direction; If ang<0 (negative): the point rotates in clockwise direction. If point P coincides with the center or if ang=0: the function returns the initial coordinate.  Examples: *geo[pxrot*;70.7107;70.7107;0;0;-45] = 100 *geo[pyrot*;70.7107;70.7107;0;0;-45] = 0
geo[**pxrota**;x;y;xc;yc;ang] geo[**pyrota**;x;y;xc;yc;ang]	Return the abscissa (x) or the ordinate (y) of a point P rotated by the angle ang with center Pc (absolute rotation): x;y = abscissa and ordinate of point P xc;yc = abscissa and ordinate of point Pc ang = final rotation angle. If point P coincides with the center: the function returns the initial coordinate.  examples: *geo[pxrota*;70.7107;70.7107;0;0;90] = 0 *geo[pyrota*;70.7107;70.7107;0;0;90] = 100

Mirror functions

geo[**pmir**;q;q1]	Returns the coordinate of a point P mirrored around a vertical or horizontal axis: • if q stands for x coordinate: the mirror axis is vertical: it has the equation x=q1; q is the abscissa of point P (x=q)

	 - if q stands for y coordinate: the mirror axis is horizontal: it has the equation y=q1; q is the ordinate of point P (y=q)  examples: *geo[pmir*;100;500] = 900

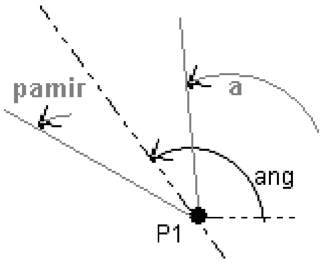
geo[**pxmir**;x;y;x1;
y1;x2;y2]
geo[**pymir**;x;y;x1;
y1;x2;y2]
	Return the abscissa (x) or the ordinate (y) of a point P mirrored around the P1-P2 axis:
	x;y = abscissa and ordinate of point P
	x1;y1 = abscissa and ordinate of point P1
	x2;y2 = abscissa and ordinate of point P2.
	
	Examples:
	geo[pxmir;0;0;0;500;500;0]=500

geo[**pxmir**;x;y;x1;
y1;ang]
geo[**pymir**;x;y;x1;
y1;ang]
	Return the abscissa (x) or the ordinate (y) of a point P mirrored around the P1- ang axis:
	x;y = abscissa and ordinate of point P
	x1;y1 = abscissa and ordinate of point P1
	ang = inclination angle of the line.
	

geo [**pxmir;**"wname1	The functions are analogue to the functions geo[**pxmir**;x;y;x1;y1;x2;y2], geo [**pymir**;x;y;x1;y1;x2;y2] and geo[**pxmir**;x;y;x1;y1;ang],geo[**pymir**;x;y;x1;y1; ang].

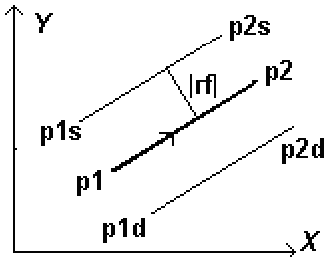
+nn",x1,y1,x2,y2] geo[**pxmir;**"wname1+nn",x1,y1,ang] geo[**pxmir**;x, y,"wname2+nn"] geo[**pxmir**;"wname1+nn","wname2+nn"] geo[**pymir;**"wname1+nn",x1,y1,x2,y2] geo[**pymir;**"wname1+nn",x1,y1,ang] geo[**pymir**;x, y,"wname2+nn"] geo[**pymir**;"wname1+nn","wname2+nn"]	They return the abscissa (x) or the ordinate (y) of the point P, mirrored around an axis. The point P and/or the axis are assigned by the name of a working. wname1"=name of the working assigning the point P. It may correspond to a point or a setup. In a profile segment (line or arc) the point P is the final point of the segment. wname2"=name of the working assigning the linear segment. It must define a linear segment made of one only segment. If the working "wname1" is not correctly defined, the function returns value 0.0. If the working "wname2" is not correctly defined, the function returns the coordinates of the point P.

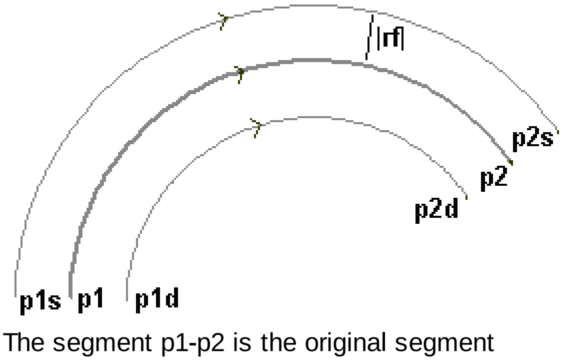
Angle rotation functions

geo[**pamir**;a;ang]	Returns the angle (a) mirrored around the ang axis: a = angle to mirror ang = inclination angle of the axis.  examples: *geo[pamir*;30;90] = 150
geo[**pamir**;a;x1;y1;x2;y2]	Returns the angle (a) mirrored around the P1-P2 axis: a = angle to mirror x1;y1 = abscissa and ordinate of point P1 x2;y2 = abscissa and ordinate of point P2. examples: *geo[pamir*;30;0;0;0;100] = 150
geo[**pamir**;a;"wname+nn"]	This function is analogue to the function geo[**pamir**;a;x1;y1;x2;y2]. It returns the angle (a) mirrored around the axis defined by a working name. The workingshould define a linear segment made of one only segment. If the working "wname2" is not correctly defined, the function returns the a value.

Segment correction function with offset

geo[**px1rf**;x1;y1;	They return the abscissa (x) or the ordinate (y) of the segment starting (1) or final

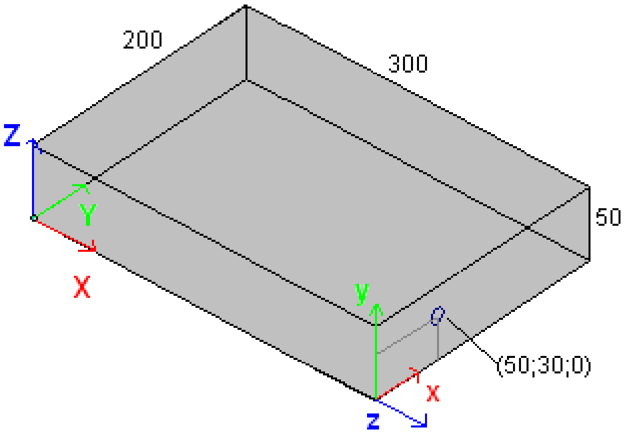
x2,y2,rf(;nret)] geo[**py1rf**;x1;y1; x2,y2,rf(;nret)] geo[**px2rf**;x1;y1; x2,y2,rf(;nret)] geo[**py2rf**;x1;y1; x2,y2,rf(;nret)]	point (2), after correcting a linear segment. x1,y1= abscissa and ordinate of the starting point of the segment P1 x2,y2= abscissa and ordinate of the starting point of the segment P2 rf=correction value to apply on the segment. If the value is positive, the correction will be carried out to the left side of the segment; if the value is negative, the correction will be carried out to the right side of the segment. nret= flag of outcome request. If a positive value is set, the function returns 1, if the geometric data are correct; otherwise it returns 0.  The segment p1-p2 is the original segment If the value of rf is positive, the correct segment is p1s-p2s. If the value of rf is negative, the correct segment ist p1d-p2d In both cases the correct segment has a distance from the original one equal to the absolute value of rf. If the coordinates of the points P1 and P2 are the same, the function returns the coordinate of the starting point without applying the correction.

geo[**px1rf**; x1;y1;
x2,y2,xc,yc,sr,rf;
(nret)]
geo[**py1rf**; x1;y1;
x2,y2,xc,yc,sr,rf;
(nret)]
geo[**px2rf**; x1;y1;
x2,y2,xc,yc,sr,rf;
(nret)]
geo[**py2rf**; x1;y1;
x2,y2,xc,yc,sr,rf;
(nret)]
	They return the abscissa (x) or the ordinate (y) of the segment starting (1) or final point (2), after carrying out the correction to an arc.
	x1,y1= abscissa and ordinate of the starting point of the segment P1
	x2,y2= abscissa and ordinate of the starting point of the segment P2
	xc,yc=abscissa and ordinate of the center of the arc.
	sr=direction of circle rotation. If sr=0, it is a clockwise direction, otherwise it is considered anti-clockwise
	rf=correction value to apply on the segment. If the value is positive, the correction will be carried out to the left side of the segment; if the value is negative, the correction will be carried out to the right side of the segment.
	nret= flag of outcome request. If a positive value is set, the function returns 1, if the geometric data are correct; otherwise it returns 0.
	
	
	The segment p1-p2 is the original segment
	If the value of rf is positive, the correct segment is p1s-p2s.
	If the value of rf is negative, the correct segment ist p1d-p2d
	In both cases the correct segment has a distance from the original one equal to the absolute value of rf.
	If the arc is not valid (different initial and final radius) or the internal requested correction has a greater value than the radius of the arc, the function returns the point coordinate without applying the correction.

geo[**px1rf;**	They return the abscissa (x) or the ordinate (y) of the segment starting (1) or final

"wname+nn",rf;
(nret)]
geo[**py1rf;**
"wname+nn",rf;
(nret)]
geo[**px2rf;**
"wname+nn",rf;
(nret)]
geo[**py2rf;**
"wname+nn",rf;
(nret)]
	point (2), after applying the correction to it. Before the current working, the line is defined by a working name and can be a linear segment or an arc.
	"wname"=name of the working assigning the line to which the correction needs be applied.
	rf=correction value to apply on the segment. If the value is positive, the correction will be carried out to the left side of the segment; if the value is negative, the correction will be carried out to the right side of the segment.
	nret= flag of outcome request. If a positive value is set, the function returns 1, if the geometric data are correct; otherwise it returns 0.
	The selected working must define an arc on a plane xy or a linear segment made of one only line.

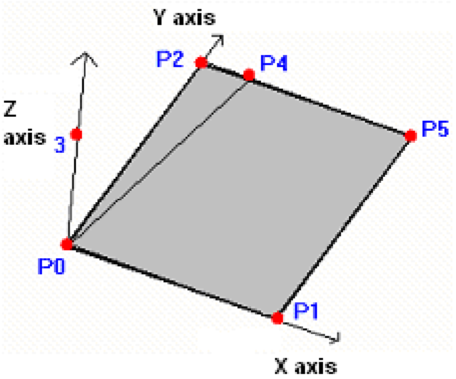
Function of coordinates conversion and of faces information reading

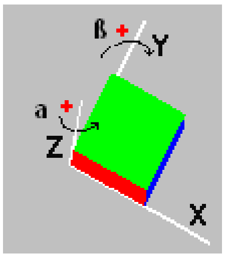
geo[**pxp**;x;y;z;
(nside)]
geo[**pyp**;x;y;z;
(nside)]
geo[**pzp**;x;y;z;
(nside)]
	Returns the coordinate (*x; y; z)*of the indicated point, that has been converted from the local face system (*nside*to the absolute coordinates of the piece:
	x;y;z = point coordinates in the local system of the face
	nside = custom face number
	The face number is optional: if unassigned, it takes the active face.
	The function returns the initial coordinate if:
	- *nside* assigns an invalid face value;
	- *nside* is not specified and the active face is the overall piece face;
	- in assignment of (o, v, r) variables or variable geometry and *nside* indicates a fictive face.
	If *nside* indicates a real face: the function operates although the face is not assigned on the piece.
	If the working is programmed in face-piece and *nside=100*, it is considered the last automatic face, that has been assigned, before the selected working.
	Let this figure be considered:
	
	 200
	300
	Z
	Y
	X
	50
	y
	o
	(50;30;0)
	x
	z
	- length of the piece=300; height of the piece=200; thickness of the piece=50
	- on face 4 a point is shown at the positions 50;30;0
	- in absolute coordinates of piece, the point has coordinates 300;50;30
	Examples:
	*geo[pxp;100;100;-5;1]*: this function returns the abscissa of the point (100;100;-5) from the local system of face 1 to the absolute coordinates of the piece;
	*geo[pxp;100;100;-5]*: this function returns the abscissa of the point (100;100;-5) from the local system of the active face to the absolute coordinates of the piece

geo[**pxf**;x;y;z;
(nside);(nsorg)]
geo[**pyf**;x;y;z;
	Returns the coordinate (*x; y; z*) of the indicated point, that has been converted from the local face system (*nsorg*) to the local system of the face (*nside*):
	x;y;z = point coordinates in the local system of the face nsorg

(nside);(nsorg)] geo[**pzf**;x;y;z; (nside);(nsorg)]	nside = number of the target face (custom number) (in case of empty assignment: the default face value is =current face) nsorg = number of the source face (custom number) (in case of empty assignment: the default face value is =overall piece) It is allowed to handle forms reduced: • to 4 arguments. Example: geo[**pxf**;x;y;z]: • *nside*: it takes the active face; • *nsorg*: it takes the overall piece face value (-1); • to 5 arguments. Example: geo[**pxf**;x;y;z; nside]: • *nsorg*: it takes the overall piece face value (-1). In case of unassigned or invalid *nsorg* : the function converts from the absolute coordinates of the piece to the local system of the face (*nside*). The function returns the initial coordinate if: • *nside* assigns an invalid face value; • *nside* is not specified and the active face is the overall piece face; • in assignment of (o, v, r) variables or variable geometry and *nside* indicates a fictive face. If *nside* and/or *nsorg* indicates a real face: the function operates although the face is not assigned on the piece. If the working is programmed in face-piece and nside=100 or nsorg=100, it is considered the last automatic face that has been assigned, before the selected working. Examples: *geo[pxf;100;100;-5;1;7]*: this function returns the abscissa of the point (100;100;-5) from the local system of face 7 to the local system of face 1; *geo[pxf;100;100;-5;;7]*: this function returns the abscissa of the point (100;100;-5) from the local system of face 7 to the local system of the active face; *geo[pxf;100;100;-5;1]*: this function returns the abscissa of the point (100;100;-5) from the absolute coordinates of the piece to the local system of face 1; *geo[pxf;100;100;-5]*: this function returns the abscissa of the point (100;100;-5) from the absolute coordinates of the piece to the local system of the active face;

geo[**px**;(np);
(nside);(dd);
(ndest)]
geo[**py**; (np);
(nside);(dd);
(ndest)]
geo[**pz**; (np);
(nside);(dd);
(ndest)]
	Returns the coordinate (*x; y; z*) of the face edge (*nside*) in coordinates of the local system of the face (*ndest*):
	np = face edge identification number (in case of empty assignment: it uses =0)
	• 0: face origin (point P0)
	• 1: point along the x+ axis (point P1)
	• 2: point along the y+ axis (point P2 recalculated)
	• 3: point along the z axis, in z-air direction (point P3)
	• 4: initial point P2
	• 5: fourth point of the face rectangle (point P5)
	nside = number of the source face from which the edge is read (custom number) (in case of empty assignment: the default face value is =current face)
	dd = assignment of the required point from P0 (it is not significant if np=0) (in case of empty assignment: no value is applied)
	ndest = number of the target face on which the edge is read (custom number) (in case of empty assignment: the default face value is =overall piece face)

	 Y axis Z axis 3 P2 P4 P5 P0 P1 X axis It is allowed to handle forms reduced: • to 2 arguments. Example: geo[**px**;np]: • *nside*: it takes the active face; • *dd*: the distance is the default distance (face dimension along the x-axis); • *ndest*: it takes the overall piece face value (-1); • to 3 arguments. Example: geo[**px**;np;nside]: • *dd*: the distance is the default distance (face dimension along the x-axis); • *ndest*: it takes the overall piece face value (-1); • to 4 arguments. Example: geo[**px**;np;nside;dd]: • *ndest*: it takes the overall piece face value (-1). If *np* has an invalid value: the function operates as with *np*=0. If *np*=4*:* the function does not interpret *dd.* The function returns the null coordinate (=0.0) if: *nside* assigns an invalid face value; *nside* is not specified and the active face is the overall piece face; in assignment of (o, v, r) variables or variable geometry and *nside* indicates a fictive face. If *nside* indicates a real face, the function operates although the face is not assigned on the piece. If the working is programmed in face-piece and nside=100 or ndest=100, it is considered the last automatic face that has been assigned, before the selected working. Examples: *geo[px;0]*: this function returns the abscissa of a point P0 of the active face, converted to the absolute coordinates of the piece. *geo[px;0;7;;1]*: this function returns the abscissa of a point P0 of face 7, converted to the local system of face 1
geo[**alfa**;(nside)] geo[**beta**;(nside)]	Return respectively the rotation angle (alfa) and the slewing angle (beta) to assign to a tool in order to make it work perpendicular to the *nside* face: nside = face number (custom number) (in case of empty assignment: the default face value is =current face). In any case the function returns 0, if: • *nside* assigns an invalid face value; • *nside* corresponds to a face which is not assigned on the piece; • the face is assigned with invalid geometry; • in assignment of (o, v, r) variables or variable geometry and *nside* indicates a fictive face. If *nside* indicates a real face: the function operates although the face is not assigned on the piece. If the working is programmed in face-piece and nside=100, it is considered the last automatic face that has been assigned, before the selected working.

	 With reference to the figure above: • beta rotates around the y axis • alfa rotates around the z axis. With reference to the six real faces of the parallelepiped, alfa and beta values are assigned as follows: *alfa* and *beta* are assigned as follows: *Face* — *Beta; Alfa* — *Notes* 1 — (0;0) — Alfa: any 2 — (180;0) — Alfa: any 3 — (-90;90); (90;-90) 4 — (-90;180); (90;0) 5 — (-90;-90); (90;90) 6 — (-90;0); (90;180) If in the configuration of Tpaedi32 the inversion of the slewing axis is set, the slewing values signes in the table have to be inverted.
geo[**lface**;(nside)] geo[**hface**;(nside)] geo[**sface**;(nside)] geo[**zface**;(nside)]	Return respectively: length (lface), height (hface), thickness (sface) and z-axis orientation (zface) of the *nside* face: nside = face number (custom number) (in case of empty assignment: the default face value is =current face). In any case the function returns 0, if: • *nside* assigns an invalid face value; • *nside* corresponds to a face which is not assigned on the piece; • in assignment of (o, v,r) variables or variable geometry: *nside* indicates a fictive face. If *nside* indicates a real face: the function operates although the face is not assigned on the piece. If the working is programmed in face-piece and nside=100, it is considered the last automatic face that has been assigned, before the selected working.
geo[**isface**; (nside)]	Returns the existence flag for the face in the piece (1= face exists, 0= face does not exist): nside = face number (custom number) (in case of empty assignment: the default face value is =current face). The function returns 0 if: • *nside* assigns an invalid face value; • *nside* corresponds to a face which is not assigned on the piece; • in assignment of (o, v,r) variables or variable geometry and *nside* indicates a fictive face. If the working is programmed in face-piece and nside=100, it is considered the last automatic face that has been assigned, before the selected working.
geo[**simil**; (nMode);(nside)]	Returns the number of the real face verifying criteria similarity of the face that is defined by nside: nMode = similarity criterion. If the parameter is not assigned, the default is 0: • 0: verifies that the'tool has the same entry direction on the face. The face (nside) is generated by translation of one or more axes and possible rotation on the plane xy. • 1 (different from 0): the face (nside) is generated by simply translation of one

	or more axes. nside = number of the face (custom number). If the parameter is not assigned, the number of the current face is used. In any case the function returns 0, if: • nside assignes a non valid face value • nside shows a fictive face not assigned in the piece • nside finds a fictive face during the assignments of variables (o,v,r) or of variable geometry If the returned value is significant (different from 0), it corresponds to a face custom number. If nside assigns a value of real face: the function becomes nside again. If the working is programmed in face-piece and nside=100, it is considered the last automatic face that has been assigned, before the selected working.
geo[**pr1**;(nside)] geo[**pr2**;(nside)] geo[**pr3**;(nside)]	return the additional parameters of fictive or automatic face. nside = number of the face (custom number). If the parameter is not assigned, the default is the number of the current face.

Access functions to programmed working information

The function should be considered as function of advanced programming.

geo [**param**;"wname+ nn";"pname";(nret)] geo [**param**;"wname+ nn";pID; (nret)]	Returns the working parameter value, as follows: • "wname" = name of the working searched before the current working. According to the syntax, the name should be placed between double quotation marks. • "pname" =paramter name (ASCII). According to the syntax, the name should be placed between double quotation marks, in lower-case letters. • pID =parameter's identification number. pID use form is recommended to advanced users only. • nret = result enquiry flag. If the field is empty or not assigned, it is used the value 0 as a default one. If the field is assigned with positive value (1), the function returns: • 1 if the parameter and working search is correct • 0 if parameter and working search is not correct "wname" working research is broken at the first correspondence. If the working is not correctly found, the function returns 0.0. For the'"pname" argument (ASCII parameter name) some remarkable cases are managed, as follows: • working generic information: • "#cop" reads the operative code of the working • "#tip" reads the working typology; • "#prog" reads the consecutive sequence number of workings. • specific information of complex working (sub-program or macro-program) • "#subxi", "#subyi", "#subzi": read the corresponding coordinate of the first position worked; • "#subxe", "#subye", "#subze": read the corresponding coordinate of the 'last position worked; • "#subxn", "#subxp": read the position corresponding to the x-coordinate of minimum/maximun overall dimension of the working; • "#subyn", "#subyp": read the position corresponding to the y-coordinate of minimum/maximun overall dimension of the working; • "#subzn", "#subzp": read the position corresponding to the z-coordinate of minimum/maximun overall dimension of the working. examples: *work[param*;"w1", "x"] : returns the value of the X-quote of the "w1" working,

	according to the programming and to the assignment (absolute/relative,..).

11.6.10 Custom functions

Custom functions are only available in Professional mode.

fun0[n1;..;n30] .. fun99[n1;..;n30]	They run the corresponding custom function, assigned with the default name The maximum arguments number is 30. Error conditions: • 123: operands number =0; • 124: operands number >30; • 134: too many calls of custom functions (max. 5) • 135: use of custom function in a not valid context (it signals that a custom function call already stored in stack or an illicit use of a private custom function is programmed).
funxxxxx[n1;..;n30]	It runs the corresponding custom function, assigned with customized name. The maximum arguments number is 30. Error conditions: • 123: operands number =0; • 124: operands number>30; • 134: too many calls of custom functions (max. 5) • 135: use of custom function in a not valid context (it signals that a custom function call already stored in stack or an illicit use of a private custom function is programmed).

12 Error Messages

12.1 General Errors

It is about messages, directly displayed in the window, which are strictly connected to the execution of program commands. They can identify:

- actual errors resulting from a failed procedure. (example: failure to load a program from a file),
- simple warnings: messages which inform about a specific situation. (example: the request for a tool applied to unsuitable workings).

12.1.1 1 - Error in the procedure

Description
It is about a general error which is not otherwise identified.

Context
Any context is fine. However, it is necessary to specify that, while each different error condition is usually precisely identified by a detailed message, the documentation provided in this section is only applicable to general cases, although the number of situations which generate this error is limited.

12.1.2 2 - Error in memory allocation

Description
there is not enough system memory to execute the required procedure. It is a critical error: you are recommended to close the program and perform the necessary system checks.

Context
any

12.1.3 5 - Error in file access

Description
An error in file access (in reading or saving) has occurred. It can identify such situations: improper file addressing, access to file not allowed, empty file, invalid file format.

Context
This error message may result from the request for any of the following:

- program loading or saving
- piece matrix loading or saving
- copy of workings to Clipboard (the error can be due to the creation of an auxiliary temporary file)
- custom function file loading or saving (in loading this error can be caused by the identification of an invalid file format)

12.1.4 6 - Error in accessing to clipboard

Description
An error in accessing to Clipboard has made impossible to save or retrieve data. This kind of error is strictly connected to an improper system functioning.

Context
This error message may result from the request for any of the following:

- copy of workings to Clipboard (working edit commands: Copy,Delete)
- Clipboard data retrieval (working edit commands: Paste; general tools: Translation, Rotation, Mirrors,

Repetitions, Exploded view)

12.1.5 7 - Error accessing an Undo temp file

Description
An error in accessing one of the temporary files created to support the UNDO function has occurred. This error can be due to external temp-file corruption or to error conditions resulting from the access to the storage peripheral device. It is about an error strictly connected to an improper system functioning.

Context
The error message can result from the execution of each program edit command which it is possible to cancel:

- working edit commands: Edit, Insert, Paste, Delete, Selective replacements (parameters and/or properties);
- tools

12.1.6 13 - System level doesn't allow execution of this operation

Description
The activated command cannot be executed because the user access level is lower than that required by the command.

Context
This error message may result from the request for any of the following:

- macro-program loading (required access level: Constructor)
- loading of a program which has a read access level higher than the level set
- loading of a program which has applied Professional level tools in a system working with a standard level key
- storage of a program that has a write access level higher than the level set

12.1.7 18 - Current working is invalid

Description
General error related to the application of a command to the current working.

Context
This error message may result from the request for any of the following:

- working edit commands: Edit, Insert, Paste, Delete, Selective replacements (parameters and/or properties)
- tools

12.1.8 36 - Maximum number of workings per face was overcome

Description
It is no longer possible to enter workings into the current face, because the maximum allowed number has been reached (1.000.000).

Context
This error message may result from the request for any of the following:

- program loading
- application of subroutine (or macro), due to: excessive number of read lines or excessive number of lines resulting from the application of repetitions or emptying operations
- working insert commands: Insert, Paste
- application of tools which need the insertion of workings with consequent overgeneration of lines

12.1.9 37 - Can't assign after an induced working

Description
It has been required to enter workings after an induced working. Induced workings can be automatically generated as a result of a programmed application of subroutine (or macro); they are always queued into the list of face workings and they cannot be edited directly.

Context
This error message may result from the request for any of the following:
- working insert commands: Insert, Paste

12.1.10 38 - Can't insert this working on the current face

Description
It has been required to insert a working on a face where the working itself is disabled.

Context
This error message may result from the request for any of the following:
- insertion of a few types of workings on Piece-Face. In fact, while for each face the workings which are not handled are usually disabled, in Piece-Face all workings are always enabled and an error message appears if it is tried to enter a disabled working.

12.1.11 39 - The tool can't use a required working

Description
The tool cannot be activated because, among the workings which have been configured for the application, a working essential for the tool to function is not available. It is always about a basic work on profile for a type of segment. Basic profile codes:
- L01 [code = 2201] for linear segment
- A01 [code = 2101] for arc assigned in xy plane
- A05 [code = 2105] for arc assigned in xz plane
- A06 [code = 2106] for arc assigned in yz plane
- A10 [code = 2110] for arc assigned in xyz plane.

Context
Tools which edit or generate profiles:
- all profile tools (Break profile, Displace setup point,…)
- Advanced profile tools (Text generation, Emptying of closed areas, Profiles cut, Profile building)

12.1.12 40 - Can't edit an induced working

Description
It has been required to edit an induced working.

Context
This error message may result from the request for any of the following:
- working edit commands: Edit, Delete, Selective replacements (parameters and/or properties)
- general or profile tools which need to operate directly on original working/s

12.1.13 41 - Errors assigning working properties

Description
A value outside the minimum and maximum range limits, which can be assigned to a working property

(Level, Construct, M Field, O Field, K Field), was entered

12.1.14 42 - No modifications or replacements have been effected

Description
The activated command has not generated any changes.

Context
This error message may result from the request for any of the following:

- working edit commands: Edit, Insert, Selective replacements (parameters and/or properties)
- general, profile, advanced profile tools

12.1.15 49 - This tool may only be applied to profiles

Description
A profile tool has been activated for executions which are not works on profile.

Context
This error message may result from the request for a profile tool, with selected workings which do not belong to a profile.

12.1.16 50 - Tool didn't interpret transforms

Description
No changes have been produced by the tool activated according to the parameters set.

Context
This error message may result from the request for any of the following:

- general or profile tools
- Area Empty tool: no closed areas have been identified;

12.2 Specific errors in applying tools

It is about messages, directly displayed in the window, which are strictly connected to the execution of program commands: in this particular case, the error is caused by the unsuccessful operation of the relevant tool.

12.2.1 51 - This tool may only be applied to profile simple

Description
A profile tool has been activated for a complex profile
Context
This error message may result from the request for any of the following tools:

- *Linearize z (profile depth)*
- *Apply Connectors To Profile*
- *Disconnect profile*
- *Close open profile*
- *Generate path for tool radius compensation*
- *Generate spline curves*

12.2.2 53 - Minimize the profile: the reduction angle exceeds 90.0°

Description:
A value higher than 90° was assigned to the reduction angle parameter.

Context:
The event can be signalled after calling the tool :

- *Minimize profile*

12.2.3 54 - Fragment profile: maximum length of traits is null

Description
The maximum fragmentation length parameter is set to an invalid value (< 5.0 * epsilon)

Context
This error message may result from the request for the following profile tool: Fragment profile.

12.2.4 55 - Apply connections to profile: invalid number of connections [min: 2; max: 255]

Description
The number of assigned connections is outside the range of values from 2 to 255.

Context
This error message may result from the request for the following profile tool: *Apply connections to profile*, in automatic distribution mode.

12.2.5 56 - Apply connections to profile: invalid length of connections or compensation exceeding tool diameter

Description
The length of connections is set to null value (< epsilon); or with tool compensation flag active, the length of connections is set to a value lower than the tool diameter

Context
This error message may result from the request for the following tool: *Apply connections to profile*

12.2.6 59 - Apply connections to profile: invalid or unassigned thickness of connections

Description
The residual thickness of connections is set to null value (< epsilon)

Context
This error message may result from the request for the following tool: *Apply connections to profile*.

12.2.7 60 - Apply connections to profile: can't distribute connections on profile (reduce number of connections)

Description
The profile length is not enough to distribute all required connections. To solve this problem it is necessary to set a lower number of connections.

Context
This error message may result from the request for the following tool: *Apply connections to profile* with

request for automatic distribution of connections.

12.2.8 61 - Inversion of profile : complex codes encountered that cannot be inverted

Description
The current profile that is wished to be inverted is assigned with complex codes (subroutines and/or macros) which:

- cannot be assimilated to a profile and do not handle the inversion parameter; or
- which cannot be inverted since they must respect restrictions set during the assignment of the working database for the application (they themselves apply workings for which the possibility to invert the execution has been excluded)

Context
This error message may result from the request for the following profile tool:*Invert profiles.*

12.2.9 62 - Apply tool: complex code of profile end doesn't terminate with a profile trait

Description
The current profile to be inverted terminates with a complex code (subroutines and/or macros) whose development does not end with a segment of profile

Context
This error message may result from the request for the following profile tool: *Invert profiles, Apply entry to profile* (selecting a joined segment as a coverage in exit)

12.2.10 63 - Displace setup in profile : position coincident with current setup

Description
The position where the setup point should be moved corresponds to a point on profile which coincides with the current setup position (< epsilon)

Context
This error message may result from the request for the following profile tool: *Displace setup in closed profile.*

12.2.11 64 - The tool can be applied at a closed profile

Description
The current profile is not closed. The starting point must coincide with the end point (< epsilon)

Context
This error message may result from the request for the following profile tool: *Displace setup in closed profile, Apply entry to profile* (selecting a joined segment as a coverage in exit).

12.2.12 67 - Fillet or bevelling: radius assigned is null

Description
The radius assigned to fillet or chamfering has null value (< epsilon)

Context
This error message may result from the request for the following profile tools: *Apply fillets to profile, Apply chamfering to profile, Generate path for tool radius compensation*

12.2.13 68 - Cut profile : this position is already a setup

Description
The position where the profile should be cut corresponds to a point on profile which coincides with the current setup position (< epsilon)

Context
This error message may result from the request for the following profile tool: *Cut profile*

12.2.14 69 - Cut profile : this position already terminates a profile

Description
The position where the profile should be cut corresponds to a point on profile which coincides with the profile end position (< epsilon)

Context
This error message may result from the request for the following profile tool: *Cut profile*.

12.2.15 70 - Enter / Exit profile : reference working unassigned in working database

Description
The tool cannot be activated because, among the workings which have been configured for the application, a working essential for the tool to function is not available. It is always about a basic work on profile for a type of segment. The necessary basic codes are the following:

- COPL01 for linear segment;
- COPA17 for circular segment at the beginning of profile;
- COPA16 for circular segment at the end of profile;

Context
This error message may result from the request for the following profile tools: *Apply entry to profile, Apply exit to profile*.

12.2.16 71 - Enter profile : can't link an entry before profile

Description
It is not possible to assign a profile opening or, in any case, a point hook before the current profile since:

- the profile starts with a complex code (subroutine or macro) which, in any case, cannot be assimilated to a profile, at the beginning of its own development or
- the profile starts with a complex code (subroutine or macro) which does not handle the hook parameter or which cannot be hooked because it must respect restrictions set during the assignment of the working database for the application

Context
This error message may result from the request for the following profile tools: *Apply setup, Apply multiple setups, Apply entry to profile*.

12.2.17 72 - Enter profile: displacement unassigned for initial point of profile

Description
The position where the setup point should be moved coincides with the current setup position (< epsilon)

Context
This error message may result from the request for the following profile tool: *Apply entry to profile*

12.2.18 73 - Exit profile : displacement unassigned for final point of profile

Description
The position where the final point of profile should be moved coincides with the current end point of profile (< epsilon)

Context
This error message may result from the request for the following profile tool: *Apply exit to profile*.

12.2.19 75 - Join profiles: second profile not properly identified

Description
No profile with geometric continuity with respect to the first selected profile has been identified

Context
This error message may result from the request for the following profile: *Connection between consecutive profiles*.

12.2.20 78 - Join profiles: profiles are separated

Description
The selected profiles have no geometric continuity such as to allow to join separate profiles into a single profile (< epsilon)

Context
This error message may result from the request for the following profile tool: *Join profiles*.

12.2.21 79 - Stretch profile : non modifiable complex codes were encountered

Description
The current profile is assigned with complex codes (subroutines and/or macros) which cannot be modified by the selected tool:

- they cannot be assimilated to a profile and do not handle the scale parameters; or
- they cannot be reduced or amplified since they must respect restrictions set during the assignment of the working database for the application (they themselves apply workings for which the possibility to scale the execution has been excluded)
- they execute arcs in planes different from xy and a scale limited to the only xy plane is required

Context
This error message may result from the request for the following profile tool: *Scale the profile.*.

12.2.22 80 - Stretch profile : amplification or reduction factor unassigned or equal to 1.0

Description
The assigned scale factor is equal to 1.0 or is unassigned.

Context
This error message may result from the request for the following profile tool: *Scale the profile*.

12.2.23 82 - The tool required too many repetitions (max 1000)

Description
An excessive number of repetitions has been required: the total number of repetitions to enter cannot be greater than 1000.

Context
This error message may result from the request for the following tools: *Repetition of workings, Rectangular series of workings, Circular series of workings.*

12.2.24 85 - Apply tool: the profile has circular segment laying on a plane different from xy

Description
The activated tool cannot operate on one or more selected profiles because the profiles themselves have circular elements (arcs) laying on a plane different from xy

Context
This error message may result from the request for any of the following tools:
- *Scale the profile*, with request for a scale limited to the only xy plane
- *Linearize z* (profile depth)
- *Generate path for tool radius compensation*
- *Generate spline curves*

12.2.25 86 - Profile exit: can’t hook on a downward exit

Description
It is not possible to assign a hook point after the profile itself since the profile terminates with a complex code (subroutine or macro) which:
- cannot be assimilated to a profile, at the end of its own development or
- which cannot handle the hook parameter or cannot be hooked since it must respect restrictions set during the assignment of the working database for the application;

Context
This error message may result from the request for the following profile tool: *Apply exit to profile*.

12.2.26 88 - Apply setup to profile: can't apply setup because reference code is missing

Description
The activated tool has not worked because of the impossibility to assign a reference setup code to profile.

Context
This error message may result from the request for profile tools. The assignment of a reference setup code is performed during the Configuration of Tpaedi32.

12.2.27 92 - The tool didn't introduce displacements on any axis

Description
No translation is generated by the parameter setting.

Context
This error message may result from the request for the *Translation* tool.

12.2.28 93 - The tool introduced a null rotation

Description
No rotation is generated by the parameter setting. It could have been assigned a relative rotation angle of null value or an absolute rotation angle and a relative centre with zero value.

Context
This error message may result from the request for the *Rotation* tool.

12.2.29 94 - The tool didn't introduce repeated applications

Description
The total number of repetitions (repeated applications) to enter is not enough.

Context
This error message may result from the request for the following tools:
Repetition of workings with number of required repetitions lower than 1
Rectangular series of workings with total number of repetitions (rows*columns) lower than 1
Circular series of workings with number of elements to execute lower than 2

12.2.30 98 - Create text: insufficient font height (min = eps * 100)

Description
Too small font size has been assigned. The value cannot be lower than (epsilon * 100).

Context
This error message may result from the request for the advanced *Text Generation* tool.

12.2.31 99 - Develop text: invalid arc developed

Description
The arc on which the text is distributed is incorrectly assigned: the radius is null or the initial radius is different from the final radius.

Context
This error message may result from the request for the advanced *Text Generation* tool.

12.2.32 294 - Surface clearing: profile isn't closed

Description
The emptying of open area/s has been required.

Context
This error message may result from the request for the advanced *Area Empty* tool.

12.2.33 295 - Surface clearing: profile unsuitable for the assigned tool

Description
It has been required to clear an already empty closed area or such that a first partial clearing pass, with the assigned technology, is not even allowed.

Context
This error message may result from the request for the advanced *Area Empty* tool.

12.2.34 296 - Surface clearing: tool radius assigned null [min: 10*eps]

Description
During surface clearing, an invalid radius compensation value (< 10.0*epsilon) has been assigned

Context
This error message may result from the request for any of the following:
- advanced *Area Empty* tool;
- in applying a surface clearing cycle (application of subroutine or macro).

12.2.35 297 - Surface clearing: coverage exceed the tool radius

Description
The assigned coverage value exceeds the tool radius compensation

Context
This error message may result from the request for any of the following:
- advanced *Area Empty* tool
- in applying a surface clearing cycle (application of subroutine or macro).

12.2.36 298 - Surface clearing: depth range includes Z=0.0

Description
The initial and final depth values are assigned with opposite sign in surface clearing with execution of subsequent passes

Context
This error message may result from the request for any of the following:
- advanced *Area Empty* tool
- in applying a surface clearing cycle (application of subroutine or macro).

12.2.37 299 - Surface clearing: invalid Z clearance coordinate

Description
Initial depth values and Z clearance coordinate are assigned with the same sign

Context
This error message may result from the request for any of the following:
- advanced *Area Empty* tool
- in applying a surface clearing cycle (application of subroutine or macro).

12.2.38 300 - Surface clearing: eccessive number of profiles to be evaluated (>300)

Description
It has been required to execute a surface clearing which has generated the evaluation of an excessive number of closed areas (maximum allowed value = 300).

Context
This error message may result from the request for any of the following:
- advanced *Area Empty* tool
- in applying a surface clearing cycle (application of subroutine or macro).

12.3 Errors In Parametric Programming

It is about error messages displayed:
- directly in the window, with reference to the incorrect setting of a software command or tool; or
- in the list of errors, with reference to incorrect setting parameters: variables, variable geometries, face program.

For this kind of error, the relevant expression is evaluated with the following assignments:

- value 0.0, in case of numeric variable or parameter;
- evaluated string coincident with programmed string, in case of numeric variable or parameter.

For a detailed description of the functions where errors can be detected refer to Chapter Parametric Programming

12.3.1 101 - Parametric programming: string too long

Description
An expression consisting of a too high number of characters has been typed. The maximum allowed number of characters is 100.

12.3.2 102 - Parametric programming: invalid syntax

Description
The syntax used in parametric programming is invalid.
Here are a few programming rules which can be useful to interpret a syntax error:

- characters whose value is enclosed between the space (' ') and the closed curly bracket ('}') are valid;
- the space can only be used in assignment of a string (string-type variable or parameter or string-type argument). Examples:
  - "**strcmp[r5;"pippo 1"]**" is valid
  - "**120+ 12**" causes syntax errors
- the " character is interpreted to assign direct messages (it heads and closes a message). Example: strcmp[r5;"pippo"];
- the ' character is interpreted to assign a character value (heading and closure). Example: 120+'a';
- the use of the relative syntax must assign symbolic names with:
  - a non-empty symbolic name: "o\" causes error;
  - name length must not exceed 16 characters: "o\abracadabraaaaaaaaaaaa" causes error;
  - a variable of the type indicated by the specified name must be assigned: "o\aaa" causes syntax error if the “o” variable is not assigned with the "aaa" symbolic name.
- the use of variable "o" functions relative to the development of custom functions causes syntax error, if they are used in a program

Example:
Examples of invalid syntax are the following:
"100+16-" -> becomes valid if changed to "100+16"
"32*(r0+r3" -> becomes valid if changed to "32*(r0+r3)"
"abs(r5)" -> becomes valid if changed to "abs[r5]"
"o\aaa" -> if the “o” variable is not assigned with the "aaa" symbolic name

12.3.3 105 - Parametric programming: value exceeds range allowed (-3.4E+30; 3.4E+30)

Description
The value calculated for the numeric expression exceeds the range allowed for a value with decimal point

12.3.4 106 - Parametric programming: solution of string parameter too long (max = 260 chars)

Description
The string which results from a parametric programming exceeds the maximum allowed number of characters for a string-type parameter which is 260

12.3.5 109 - Parametric programming: invalid context for subprogram arguments

Description
Error relative to the use of variable arguments concerning the application of subroutine or macro: subx, suby,…., subface.

Context
Invalid contexts of use are the following:

- assignment of 'o', 'v' variables;
- assignment of custom functions;
- assignment of variable geometries (fictive face edges).

12.3.6 111 - Parametric programming: invalid context for use of variables "$"

Description
Error relative to the use of <$> variables or functions with <$> variables: $0-$299, p$[.], min$[.], max$[.], ave$[.], sum$[.].

Context
Invalid contexts of use are the following:

- assignment of 'r', 'o', 'v' variables;
- assignment of custom functions;
- assignment of variable geometries (fictive face edges);
- program text.

12.3.7 112 - Parametric programming: invalid context for use of variables "r"

Description
Error relative to the use of <r> variables or functions with <r> variables: r0-r299, pr[.], minr[.], maxr[.], aver[.], sumr[.], strlen, getat[.],strcmp[.], toolex[.]. tooltip[.].

Context
Invalid contexts of use are the following:

- assignment of 'o', 'v' variables;
- assignment of custom functions.

12.3.8 113 - Parametric programming: invalid context for use of variables "v"

Description

Context
Invalid contexts of use are the following:

- assignment of 'o', 'v' variables;
- assignment of custom functions.

12.3.9 114 - Parametric programming: invalid context for use of variables "o"

Description

Context
Invalid contexts of use are the following:

- assignment of 'o', 'v' variables;
- assignment of custom functions.

12.3.10 115 - Parametric programming: invalid context for use of variables "j"

Description

Error relative to the use of < j > variables or functions with <j> variables: j0-j99, pj[.], minj[.], maxj[.], avej [.], sumj[.].

Context

Invalid contexts of use are the following:

- assignment of 'o', 'V variables;
- assignment of custom functions;
- assignment of variable geometries (fictive face edges).

12.3.11 116 - Parametric programming: invalid context for use of working name

Description:

error concerning the use of a geometric library function whose syntax uses the name of a working.

Context:

Invalid contexts of use are the following:

- assignment of 'o', 'V variables;
- assignment of custom functions;
- assignment of variable geometries (fictive face edges).

12.3.12 117 - Parametric programming: invalid index to a variable "r"

Description

An invalid index to an <r> variable is indicated or calculated: valid values are included between 0 and 299. However, in <r> variable programming, the range of values is more limited: an <r> variable can only use lower index variables; thus, for example: r10 can use r9, but not r11.

Examples:

Examples of wrong programming are the following:

"r400", "pr[400]", "*pr[400]"	the maximum index to an “r” variable is 299
"r20"	used for example in r10 variable assignment

12.3.13 118 - Parametric programming: invalid index to a variable "j"

Description

An invalid index to a <j> variable is indicated or calculated: valid values are included between 0 and 99.

12.3.14 119 - Parametric programming: invalid index to a variable "$"

Description

An invalid index to a <j> variable is indicated or calculated: valid values are included between 0 and 99.

12.3.15 120 - Parametric programming: invalid index to a variable "v"

Description

An invalid index to a <v> variable is indicated or calculated: valid values are included between 0 and 7, with the maximum allowed value which depends on the number of <v> variables which the software is enabled to handle:

- the maximum value is equal to 7 with 8 handled variables
- the maximum value is equal to 6 with 7 handled variables
- …

- no value is valid, with any <v> variable handled

12.3.16 121 - Parametric programming: invalid index to a variable "o"

Description
An invalid index to an <o> variable is indicated or calculated: valid values are included between 0 and 7, with the maximum allowed value which depends on the number of <o> variables which the software is enabled to handle:

- the maximum value is equal to 7 with 8 handled variables
- the maximum value is equal to 6 with 7 handled variables
- …
- no value is valid, with no handled <o> variable

12.3.17 122 - Parametric programming: function with too many operands (max 30)

Description
A function has been called with a number of operands greater than 30, which is the maximum allowed limit for the number of operands of a function.

12.3.18 123 - Parametric programming: function with no operands

Description
A multi-operand function without operands has been called. An example of wrong programming is the following: "ifcase[]"

12.3.19 124 - Parametric programming: function with wrong number of operands

Description
A multi-operand function with a wrong number of operands has been called. An example of wrong programming is the following: "ifelse[r0;l/2]

12.3.20 125 - Parametric programming: division by zero

Description
In a mathematical operation a division by zero has been executed. The error results from the use of division mathematical operators(/, %, #, ?)

12.3.21 126 - Parametric programming: value of trigonometric function (sin, cos) beyond range -1 +1

Description
If the operand is not included between –1.0 and +1.0, an inverted trigonometric function (asin, acos) has been executed

12.3.22 127 - Parametric programming: square root of negative value

Description
A *sqrt* function which returns the square root of a number has been executed with negative operand

12.3.23 128 - Parametric programming: invalid exponent [min: 0; max: 10]

Description
The *pown* involution function with the 2nd operand (exponent) not included between 0 and 10 has been executed

12.3.24 129 - Parametric programming: invalid geometrical library function

Description
The multi-purpose geometry library function *geo[..]* has been executed with the 1st operand which does not match a valid name. An example of wrong programming is "geo[aaa;l/2]": "aaa" does not match a valid name

12.3.25 130 - Parametric programming: function missing required argument

Description
A compulsory function argument is missing. Example of wrong programming is "primp[;100.0]": the 1st argument, which is considered compulsory, is missing

12.3.26 132 - Parametric programming: the angle is not valid for tangent calculation

Description
The error is due to the execution of the trigonometric function *tan*, when a invalid value for tangent calculation is assigned to the operand (angle). The assigned angle, reduced to the numeric range (0° - 360°) cannot be 90° or 270°, because in these cases calculating the tangent loses its meaning.

12.3.27 134 - Parametric programming: too many nested custom function calls (max: 5)

Description
The number of nested custom calls is greater than 5. This error may only result from a wrong assignment of custom functions, as performed during the software configuration.

12.3.28 135 - Parametric programming: invalid use of custom function

Description
This error message signals that an already stacked custom function is programmed or an unauthorized use of private custom function

12.3.29 136 - Parametric programming: invalid use of arguments arg# res#

Description
Error relative to the use of arguments reserved for program custom functions

12.3.30 137 - Parametric programming: arg# argument invalid index or name

Description
An invalid index to an <arg> variable is indicated or calculated, or no variable is assigned with the specified symbolic name. The error message can appear only in writing custom functions

12.3.31 138 - Parametric programming: res# argument invalid index or name

Description
An invalid index to a <res> variable has been indicated or calculated, or no variable is assigned with the

specified symbolic name. This error message can only appear in writing custom functions

12.3.32 139 - Parametric programming: error calling custom function

Description
An error has been detected at custom function level. For more information refer to the specific documentation on the custom functions assigned during the software configuration

12.3.33 140 - Parametric programming: error using functions reserved to custom functions

Description
Functions reserved for program custom functions have been used

12.3.34 141 - Parametric programming: invalid index to argument var#

Description
An invalid index to a <var> variable is indicated or calculated, or no variable is assigned with the specified symbolic name. This error message can only appear in writing custom functions

12.4 Errors in processing variable geometries

It is about error messages concerning the assignment of *variable Geometries* displayed as follows:

- directly in the window; or
- in the list of errors, during the assignment of variable Geometries or the visualization/ assignment of general program Data

12.4.1 22 - Can't eliminate face with workings assigned

Description:
A fictive face, on which workings have been programmed, cannot be cleared

Actions:
Eliminate all the programmed workings on the fictive face, then clear the fictive face itself

12.4.2 144 - Invalid or not assigned reference face

Description
An invalid number has been assigned to the reference face. It is always about a fictive face number (greater than 6) and it may specify that the selected face:

- is unassigned
- has an identification number greater than or equal to the current fictive face
- has invalid geometry (no distinct or aligned points)

12.4.3 145 - Face vertices are not all distinguished

Description
The fictive face or the automatic face which is being defined does not have all three vertices distinguished.

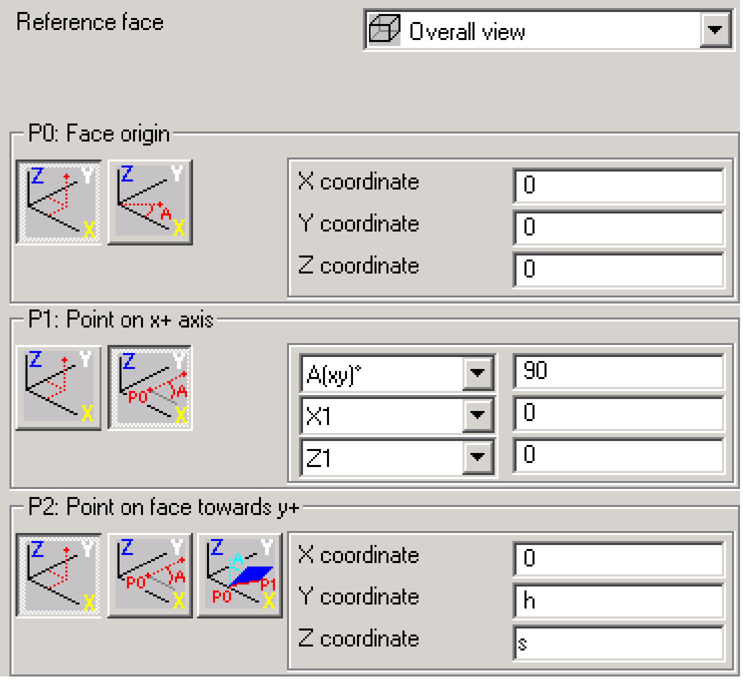
12.4.4 146 - Face vertices are aligned

Description
The three vertices of the fictive or the automatic face which is being defined are assigned aligned.

12.4.5 147 - Invalid face polar geometry

Description
A geometric error has occurred during the assignment of a face edge in polar coordinates. In particular, the error message appears in case of infinite computation of polar coordinates vector.

Example:
An example of wrong programming is shown in the Figure below:



Reference face: Overall view

P0: Face origin

X coordinate	0
Y coordinate	0
Z coordinate	0

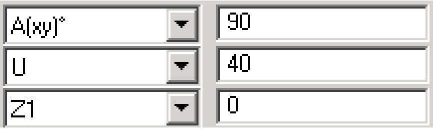
P1: Point on x+ axis

A(xy)°	90
X1	0
Z1	0

P2: Point on face towards y+

X coordinate	0
Y coordinate	h
Z coordinate	s

The error concerns point P1. A possible solution may be:

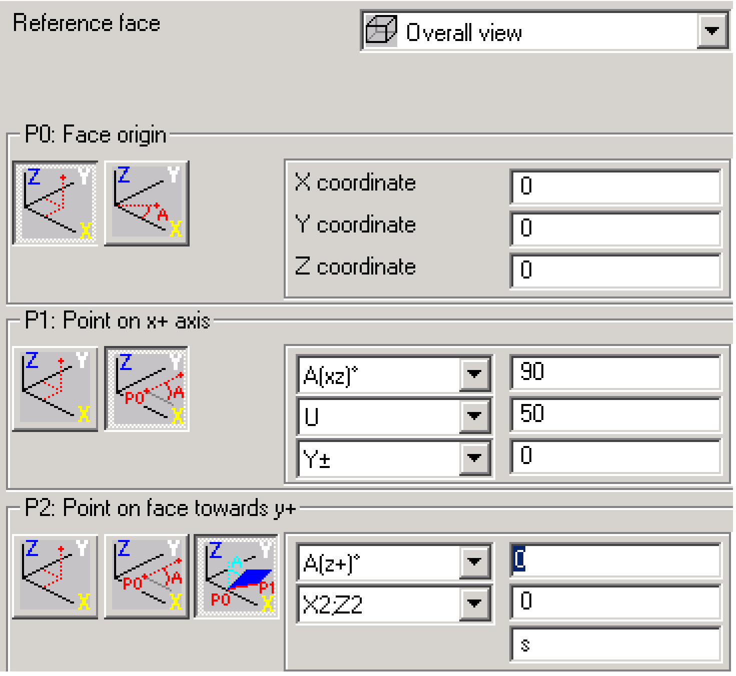


A(xy)°	90
U	40
Z1	0

12.4.6 148 - Invalid rotation plane

Description
An error has occurred in the assignment of the face rotation plane (no distinguished or aligned points).

Example:
An example of wrong programming is shown in the Figure below:



Reference face: Overall view

P0: Face origin

X coordinate: 0

Y coordinate: 0

Z coordinate: 0

P1: Point on x+ axis

A(xz)°: 90

U: 50

Y±: 0

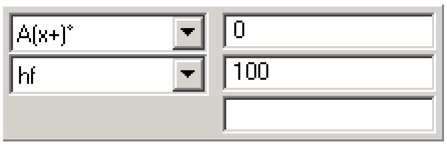
P2: Point on face towards y+

A(z+)°: 0

X2Z2: 0

s

The error concerns point P2. A possible solution may be



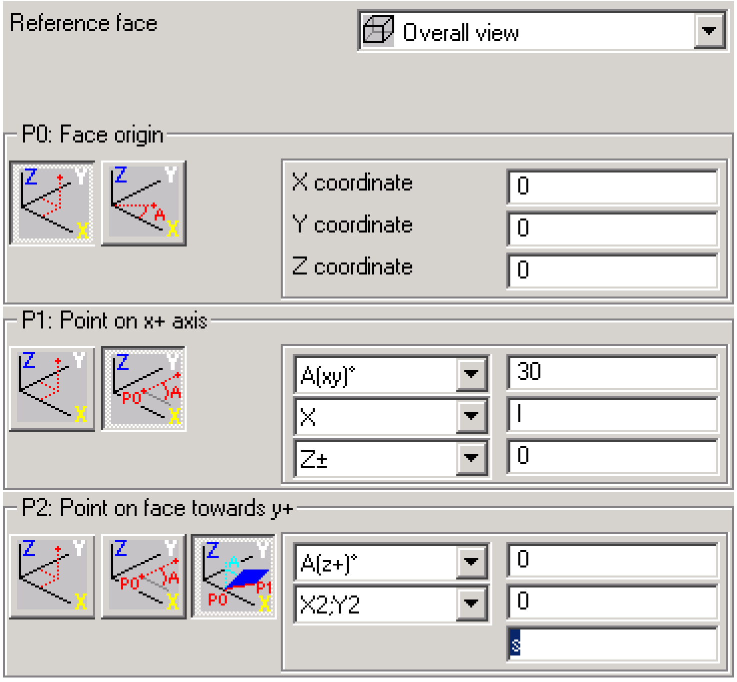
A(x+)°: 0

hf: 100

12.4.7 149 - Impossible to assign the face third point

Description
An error has occurred in the assignment of the third edge by face rotation plane

Example:
An example of wrong programming is shown in the Figure below:



Reference face: Overall view

P0: Face origin

X coordinate	0
Y coordinate	0
Z coordinate	0

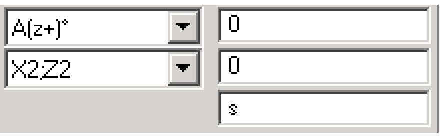
P1: Point on x+ axis

A(xy)°	30
X	l
Z±	0

P2: Point on face towards y+

A(z+)°	0
X2;Y2	0
	s

The error concerns point P2. A possible solution may be:



A(z+)°	0
X2;Z2	0
	s

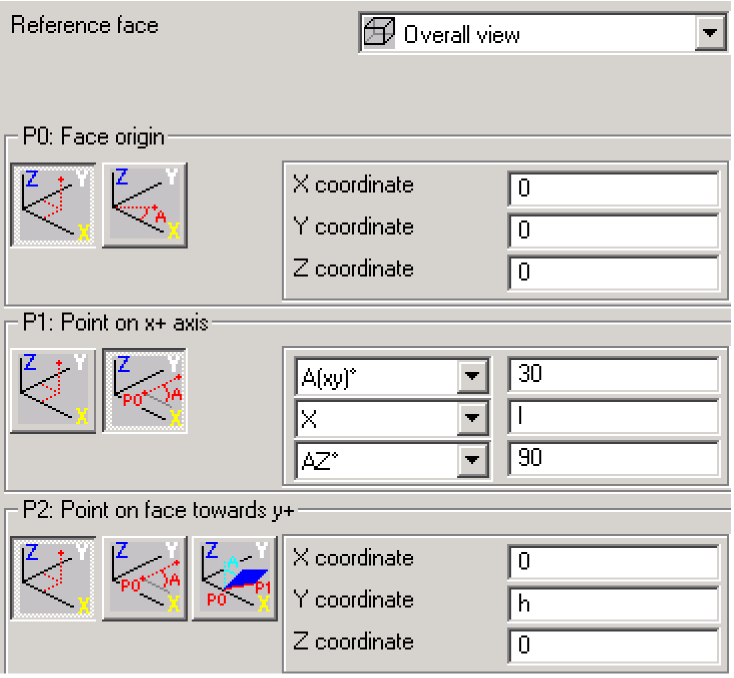
12.4.8 150 - Invalid point depth

Description

An error has occurred in the assignment of the point third coordinate axis in polar coordinates. In particular: this error may be generated in case of coordinate assignment with angular increment mode, if the angle has absolute value equal to 90°.

Example:

An example of wrong programming is shown in the Figure below:



Reference face: Overall view

P0: Face origin

X coordinate	0
Y coordinate	0
Z coordinate	0

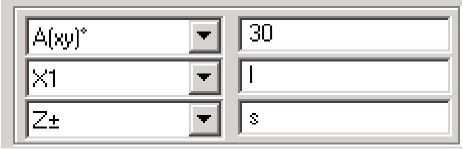
P1: Point on x+ axis

A(xy)°	30
X	l
AZ°	90

P2: Point on face towards y+

X coordinate	0
Y coordinate	h
Z coordinate	0

The error concerns point P1. A possible solution may be:



A(xy)°	30
X1	l
Z±	s

12.5 Errors in compiling face program

Only in a few cases it is about critical errors, which make the program execution impossible.
In any case: a default solution is adopted (refer to the documentation on workings for a detailed description of each single error message)

12.5.1 151 - Invalid Code <operative code name>

Description
The specified operative code (in place of <operative code name>) is not assigned in the working database implemented for the application or it is assigned with a different typology.

Actions:

- The corresponding working is executed by propagation of working coordinates from the previous working
- it is about a **critical error**, which makes the program execution impossible:
- to solve this error it is necessary to replace the working with a valid working (for example, by replacing the operative code).

Context
This error may appear while reading a program from a file

12.5.2 152 - Parameter <parameter name>: invalid value

Description
A value outside the range defined in configuring the application has been assigned to the selected parameter (in place of <parameter name>)

Actions:

- it is about a **critical error**, which makes the program execution impossible

12.5.3 153 - Parameter <parameter name>: set format $nn

Description
In a FOR instruction the first term of one of the three expressions has not been correctly set

Context
This error may only appear in programming a macro-program.

Example:
An example of correct assignment is the following:

FOR (***$0*** = r1 to ***$0*** <= r2; ***$0*** = $0+r3)
{
.
} ENDFOR

where the first term of each expression is in bold. For each term the $0 form is used (it is not necessary that the same variable is always indicated), as required.

Here is an example of wrong assignment:

FOR (***$0*** = r1 to ***<u>$0+5</u>*** <= r2; ***$0*** = $0+r3)
{
.
} ENDFOR

where the wrong assignment is underlined.

12.5.4 161 - Too many or not available automatic faces

Description
It is not possible to assign a reference to an automatic face for an excessive number of created faces (maximum allowed number: 400) or because there are no assigned faces at all.

Actions:

- The corresponding working is executed without assignment of automatic face
- it is about a **critical error**, which makes the program execution impossible
- to solve this error it is necessary to modify programming instructions (by deleting the relevant working or reducing the total number of assigned automatic faces or using a non-automatic reference face).

Context
The error message may appear while processing a working code which assigns an automatic face.

12.5.5 162 - Field F : invalid value

Description
An invalid value has been set for the *application Face* (F field) working property.

Actions:

- it is about a **critical error**, which makes the program execution impossible:

- to solve this error it is necessary to modify programming instructions.

Context
This error message may only appear in programming the piece-face and may be caused by one of the following cases of assignment:

- unauthorized assignment of automatic face
- failed assignment of automatic face
- failed assignment of non-automatic real or fictive face.

12.6 Errors in works on profile

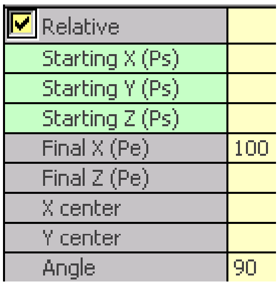
12.6.1 192 - Tool radius computed as infinite

Description
An infinite value has been computed for the vector radius of a polar system.

Context
Here are the concerned workings:

- L04 [code = 2204]
- L05 [code = 2205]
- L06 [code = 2206]
- L07 [code = 2207]

Example:
An example of wrong programming corresponds to code L04 with the following assignments:

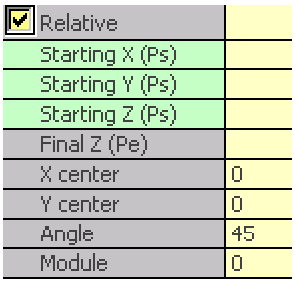


☑ Relative	
Starting X (Ps)	
Starting Y (Ps)	
Starting Z (Ps)	
Final X (Pe)	100
Final Z (Pe)	
X center	
Y center	
Angle	90

12.6.2 193 - Tool radius null

Description
A null value has been computed for the polar radius coordinate or the arc radius.
This message should be viewed as a **warning** rather than as an error.

Example:
An example of wrong programming corresponds to code L02 with the following assignments:

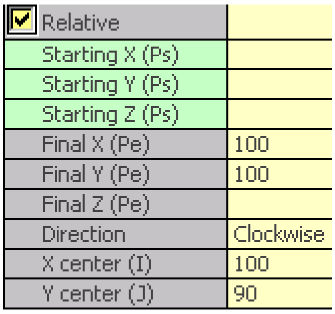


☑ Relative	
Starting X (Ps)	
Starting Y (Ps)	
Starting Z (Ps)	
Final Z (Pe)	
X center	0
Y center	0
Angle	45
Module	0

12.6.3 194 - Invalid arc

Description
An arc has been incorrectly or insufficiently assigned (unassigned center, the initial radius is different from the final radius).

Example:
An example of wrong programming corresponds to code A01 with the following assignments:



☑ Relative	
Starting X (Ps)	
Starting Y (Ps)	
Starting Z (Ps)	
Final X (Pe)	100
Final Y (Pe)	100
Final Z (Pe)	
Direction	Clockwise
X center (I)	100
Y center (J)	90

12.6.4 195 - Invalid intersection line

Description
An intercept-line is not correctly assigned (unassigned or no distinct points, or geometrically invalid). An intercept-line must be assigned with:
- two distinct points, or
- a point and an angle.

12.6.5 196 - Invalid entry tangent

Description
An entry Tangent line has not been correctly defined (unassigned or geometrically invalid). An intercept-line must be assigned with:
- an angle, or
- two distinct points.

This message should be viewed as a **warning** rather than as an error.

12.6.6 197 - Invalid exit tangent

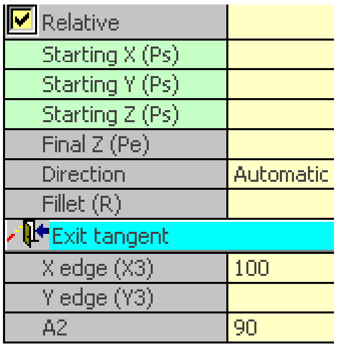
Description
An exit Tangent line has not been correctly defined (unassigned or geometrically invalid). Remember that an intercept-line must be assigned with:
- an angle, or
- two distinct points.

This message should be viewed as a **warning** rather than as an error.

12.6.7 198 - Point computed external to traits

Description
In a chamfering or in a fillet, the point computed is external to the original programmed segment.

Example:
An example of wrong programming corresponds to code A29 with the following assignments:

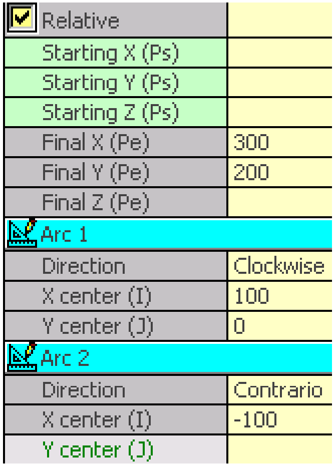


☑ Relative	
Starting X (Ps)	
Starting Y (Ps)	
Starting Z (Ps)	
Final Z (Pe)	
Direction	Automatic
Fillet (R)	
Exit tangent	
X edge (X3)	100
Y edge (Y3)	
A2	90

12.6.8 199 - Intersection non-existent

Description
Error message which may appear in case of double arcs, if no solution is found.

Example:
An example of wrong programming corresponds to code A32 with the following assignments:



☑ Relative	
Starting X (Ps)	
Starting Y (Ps)	
Starting Z (Ps)	
Final X (Pe)	300
Final Y (Pe)	200
Final Z (Pe)	
Arc 1	
Direction	Clockwise
X center (I)	100
Y center (J)	0
Arc 2	
Direction	Contrario
X center (I)	-100
Y center (J)	

12.6.9 200 - Invalid arc (points are not distinguished)

Description
An arc is incorrectly assigned, because of the coincidence between arc points an/or points with the center. Error conditions:

- arc defined by three points: the three points are not distinct;
- arc defined by two points and the centre: the centre coincides with an arc point

12.6.10 201 - Invalid arc (points aligned)

Description
In arc assignment by points, these last have been assigned aligned. If the arc takes place in the space, the error reports also the cases of circle or of aligned initial and final points and centre.

12.6.11 202 - Oval : invalid radius

Description
In constructing an oval profile the minor radius is assigned greater than or equal to the minor half-axis. This message should be viewed as a **warning** rather than as an error.

12.6.12 203 - Oval reduced to a circle

Description
In constructing an oval profile the two half-axes are defined equal.
This message should be viewed as a **warning** rather than as an error.

12.6.13 204 - Oval: null or invalid axis/axes

Description
In constructing an oval profile one or both semi-axes are null (< epsilon).

12.6.14 205 - Ellipse/Oval start point outside conic extents

Description
In constructing an oval or an ellipse the starting point falls outside the assigned conic extents.

12.6.15 206 - Rectangle: invalid axis/axes or radius

Description
In constructing a rectangle one or both axes are defined null (< epsilon) or the set fillet radius is such that it exceeds the rectangle extents.

12.7 Errors in subroutine or macro

12.7.1 210 - Invalid subprogram name

Description
The subroutine (or macro) name is incorrectly assigned. Error conditions:
- it is assigned with invalid characters: "#%;/\;"
- it is assigned with more than one character "."

12.7.2 211 - Subprogram doesn't exist

Description
The subroutine (or macro) does not exist or cannot be read.

12.7.3 212 - File selected hasn't a valid format for subprogram

Description
The specified subroutine (or macro) has an invalid format. This error message may also appear as a result of an attempt to apply a macro with a generic subroutine code without being enabled to do it.

12.7.4 213 - Invalid face number

Description
It has been required to apply an invalid identification number (number lower than 1 or greater than 99) to a face.

Context:
The message can display one of the following situations:
- A face number lower than 1 or higher than 99 has been assigned ;
- in the programm of face- piece:
  - a subroutine induced call is applied on an authomatic face;

- a SSIDE working (programmed induced call working) has defined an invalid face of the subroutine to apply;
- a SSIDE working (programmed induced call working) has defined an invalid application face.

12.7.5 216 - Subprogram read failure

Description
An error has been detected in reading the subroutine (or macro)

12.7.6 217 - Subprogram name unassigned

Description
No name has been assigned to the subroutine (or macro)

12.7.7 218 - Curve creation can't be applied

Description
It has not been possible to generate a Spline curve, since no profiles to which to apply the transform have been recognized. If it has not been required to delete original workings, this message should be viewed as a **warning** rather than as an error.

12.7.8 219 - Emptying not applicable

Description
It has not been possible to apply empting, since no profiles to which to apply it have been recognized. If it has not been required to delete original workings, this message should be viewed as a **warning** rather than as an error.

12.7.9 220 - Rotation can't be applied

Description
It has not been possible to apply the required rotation for the following reasons:

- restrictions within the subroutine (or macro) development: the development applies, on its turn, subroutine (or macro) to which the transform cannot be applied (in working database configuration);
- the transform cannot be applied to the application itself (in configuration);
- the subroutine (or macro) development includes circular elements (arcs) assigned in planes different from xy, but the auxiliary working A10 [code= 2110] is not configured.

12.7.10 221 - Inversion can't be applied

Description
It has not been possible to apply the required inversion for the following reasons:

- restrictions within the subroutine (or macro) development: the development applies, on its turn, subroutine (or macro) to which the transform cannot be applied (in working database configuration);
- the transform cannot be applied to the application itself (in configuration).

12.7.11 222 - Mirror 'x' can't be applied

Description
It has not been possible to apply the required mirror for the following reasons:

- restrictions within the subroutine (or macro) development: the development applies, on its turn, subroutine (or macro) to which the transform cannot be applied (in working database configuration);
- the transform cannot be applied to the application itself (in configuration).

12.7.12 223 - Mirror 'y' can't be applied

Description
It has not been possible to apply the required mirror for the following reasons:
- restrictions within the subroutine (or macro) development: the development applies, on its turn, subroutine (or macro) to which the transform cannot be applied (in working database configuration);
- the transform cannot be applied to the application itself (in configuration).

12.7.13 224 - Stretch can't be applied

Description
It has not been possible to apply the required stretch for the following reasons:
- restrictions within the subroutine (or macro) development: the development applies, on its turn, subroutine (or macro) to which the transform cannot be applied (in working database configuration);
- the transform cannot be applied to the application itself (in configuration);
- the subroutine (or macro) development includes circular elements (arcs) assigned in planes different from xy, but stretch is only required in xy plane.

12.7.14 226 - Too many nested subprogram calls (max = 4)

Description
The subroutine (or macro) development has too many nested subroutine (or macro) function calls. The maximum allowed number of nested function calls is:
- 4: in case of program typology;
- 3: in case of subroutine or macro typology.

12.7.15 227 - Custom error number <custom error code>

Description
The subroutine (or macro) development has detected a custom error, by interpreting a programmed error instruction:
- ERROR [code= 2009];
- BREAK [code= 2005]: it can only be used in macro text.

The programmed error number replaces <custom error code> in the message.
This error message can also result from an ERROR instruction directly programmed in the program text. In this case, it is generated when the program is saved and means that the program cannot be executed.

12.7.16 228 - Can't assign font (invalid name)

Description
The assigned font name is invalid. Error conditions:
- unassigned font name;
- the name is invalid because it consists of an excessive number of characters (maximum allowed number of chars: 32);
- the name is invalid to assign a font handled by the Windows system;
- the name is invalid to assign a TrueType font.

Context
The error message may appear in case of application of text generation subroutine (or macro)

12.7.17 229 - Can't assign device for font creation

Description
A system error has occurred in the assignment of the display device for font creation.

Context
The error message may appear in case of application of text generation subroutine (or macro)

12.8 Errors in logical conditions

It is always about critical errors which make the program execution impossible.

12.8.1 230 - Number of unloaded ENDIF greater than loaded IF

Description
A number of ENDIF conditional statements greater than the number of IF conditions has been defined. Check correspondence between IF, ELSE and ENDIF. It is useful to remember that an open IF must not be closed by ENDIF.

Example:
Here is an example of wrong programming:

```
IF …
  IF …
  …
  ENDIF
..
ENDIF
…
ENDIF              <- Endif without the relevant If
```

12.8.2 231 - Number of unloaded ENDIF lower than loaded IF

Description
A number of IF conditional statements greater than the number of ENDIF conditions has been defined. Check correspondence between IF, ELSE and ENDIF.

Example:
Here is an example of wrong programming:

```
IF …               <- If without the relevant Endif
  IF …
  …
  ENDIF
..
```

12.8.3 232 - Invalid code after an open IF

Description
The working after an open IF is invalid. An open IF may condition a working as follows:

- point working
- custom logic working
- complex working (subroutine or macro).

Only if in macro-program text:

- setup working

- work on profile (circular or linear segment)
- non-custom logic working for ($, j) variable assignment.

12.8.4 233 - Number of unloaded ENDFOR greater than loaded FOR

Description
A number of ENDFOR commands greater than the number of FOR conditions has been defined. Check correspondence between FOR and ENDFOR.

Context
This error may only appear in programming a macro-program.

12.8.5 234 - Number of unloaded ENDFOR lower than loaded FOR

Description
A number of FOR conditions greater than the number of ENDFOR commands has been defined. Check correspondence between FOR and ENDFOR.

Context
This error may only appear in programming a macro-program.

12.8.6 235 - Too many FOR instructions (max = 300)

Description
More than 300 FOR cycles have been assigned. This value is the maximum number of instructions handled by a face program.

12.8.7 236 - Too many iterations in FOR cycles (max = 10000)

Description
The application of a macro has caused the execution of more than 10000 FOR cycles: this control is activated to exclude the possibility to enter infinite or endless loops, which would block the program.

12.8.8 237 - ENDIF instruction used in completion of FOR cycle

Description
Check the correspondence between IF- ELSE and ENDIF, FOR and ENDFOR.

Example:
Here is an example of wrong programming:

```
FOR …
  IF …
  …
  ENDIF
..
ENDIF              <- Endif which unloads FOR
```

12.8.9 238 - ENDFOR instruction used in completion of IF cycle

Description
Check the correspondence between IF- ELSE and ENDIF, FOR and ENDFOR.

Example:

Here is an example of wrong programming:

```
IF …
  IF …
  …
  ENDIF
..
ENDFOR              <- Endfor which unloads If
```

12.9 Errors in global function assignment

12.9.1 240 - Function name unassigned

Description
Function name unassigned.

Context
This error message means that an incorrect working is assigned to the working database.

12.9.2 241 - Invalid function name

Description
Invalid function name.

Context
This error message may identify one of the following situations:

- an incorrect working is assigned to the working database;
- an incorrect custom function is assigned during the program configuration.

12.9.3 242 - Error during function execution: no return values are assigned

Description
the global function development is invalid.

Context
This error message identifies a parametric programming error situation: this error is detected during function interpretation and development.
It also means that no variable (j) is assigned.

12.10 Errors in multiple setups (profiles)

The procedure of development of multiple setups is activated only on request for program optimization or piece matrix creation.
It is always about critical errors which make the program execution impossible.

12.10.1 245 - Development of multiple profiles exceeds maximum number of workings that may be assigned on a face

Description
The procedure of development of multiple setups cannot be terminated in the selected face, since the maximum allowed number of workings has been reached (equal to 500000).

12.11 Errors in the assignment of technological parameters for profile and point workings

The procedure of development of multiple setups is activated only on request for program optimization or piece matrix creation.
It is always about critical errors which make the program execution impossible.

12.11.1 250 - Can't apply setup with an open profile as reference code is missing

Description
The concerned operation cannot be terminated since the necessary setup working is missing. In particular both the technological and geometric setups are missing. The technological setup is assigned at the software configuration item level.

12.11.2 251 - Can't apply a technologic per point as reference code is missing

Description
The concerned operation cannot be terminated since the necessary technological working per point, for replacing geometric points, is missing. The technological working per point is assigned at the software configuration item level.

12.11.3 252 - It is not possible to assign open profiles

Description
Open profiles or profiles headed with geometric setup have been found, but their execution is not supported by the software configuration. In this case, it is necessary to assign a technological setup for each profile.

12.12 Errors in applying tool radius corrections

It is always about critical errors which make the program execution impossible.

12.12.1 261 - Tool radius correction exceeds arc radius

Description
Internal correction is required for an arc, with compensation value greater than the arc radius. If it is necessary to apply compensation, this error can be solved by enabling the *Reduce profile* command.

12.12.2 262 - Corrections exceeds line

Description
Correction is required for a linear segment, with application of such compensation value that the segment direction is inverted. If it is necessary to apply compensation, this error can be solved by enabling the *Reduce profile* command.

12.12.3 265 - Error during correction on #xy plane with intersection solution of the segments

Description:
Correction is required for a circular segment assigned on a plane different from xy, while the intersection condition is verified within the arc.
This error can be solved by deleting the segment or disconnecting segment compensation.

12.12.4 266 - Error for tool correction in plane #xy

Description
Correction is required for a circular segment assigned in a plane different from xy, when the condition of original arc, inverting one of the two x or y-coordinates, while the execution of the segment, is verified. This error can be solved by deleting the segment or disconnecting segment compensation, or by modifying the geometry or the arc value.

12.12.5 267 - Tool compensation: inverting a correction should resolve an intersection or resume an interruption

Description:

- In a profile the inversion of a correction has been requested as a not inverted segment
- A correction, restarted after being suspended, has not been set or an intersection of the correct segments cannot be solved. The error can be solved by setting a segment with interruption of compensation.

12.12.6 268 - Suspension of correction without consecutive resumption

Description
In a profile a suspension of correction has been executed without consecutive resumption. This error can be solved by setting the resumption of correction in the selected point.

12.12.7 269 - A suspension and consecutive resumption of correction can't compute a connection

Description
In a profile a wrong suspension and consecutive resumption of correction has been detected: the application of correction between the two concerned segments can't compute a connection.

12.12.8 270 - A suspension and consecutive resumption of correction must verify geometric continuity of traits

Description
In a profile a wrong suspension and consecutive resumption of correction has been detected: the two concerned segments of profile must have geometric continuity (the first segment stops where the second starts).

12.13 Errors of fragmentation and linearization of arcs in planes different from xy

The procedure of development of multiple setups is activated only on request for program optimization or piece matrix creation.
It is always about critical errors which make the program execution impossible.

12.13.1 255 - 3D arc linearization exceeds maximum lines number

Description
It is not possible to terminate the concerned operation in the selected face since the maximum allowed number of workings has been reached (equal to 500000).

12.13.2 256 - Impossible to linearize 3D arcs because linear reference code is missing

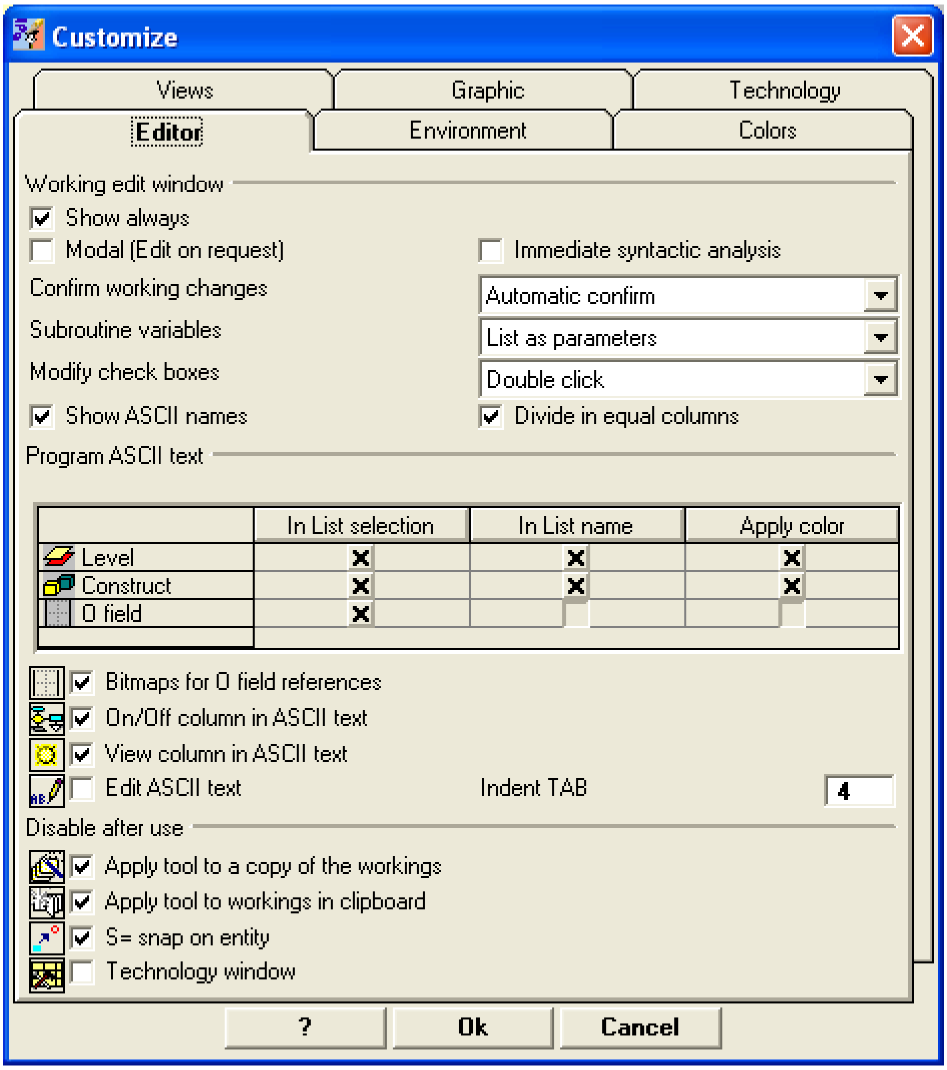
Description
It is not possible to terminate the concerned operation since the working with operating code L01 [code= 2201] is unavailable.

13 Customization of Tpaedi32

For customizing Tpaedi32 select the **Set->Customize** menu item.

13.1 Editor

Customize Editor



Working edit window

☑ Show always

□ Modal (Edit on request)

□ Immediate syntactic analysis

Confirm working changes: Automatic confirm

Subroutine variables: List as parameters

Modify check boxes: Double click

☑ Show ASCII names

☑ Divide in equal columns

Program ASCII text

	In List selection	In List name	Apply color
Level	☒	☒	☒
Construct	☒	☒	☒
0 field	☒	□	□

☑ Bitmaps for 0 field references

☑ On/Off column in ASCII text

☑ View column in ASCII text

□ Edit ASCII text

Indent TAB: 4

Disable after use

☑ Apply tool to a copy of the workings

☑ Apply tool to workings in clipboard

☑ S= snap on entity

□ Technology window

Working edit window area

- **Show always**: if selected, the working edit window is always displayed. If unselected, the window is displayed only when a working is entered from palette or when modifications are made. In this last case to modify an already entered working, either double-click or press the **[F2] key** on the relevant ASCII text line. In case the **Edit ASCII text** option is enabled and the button **[SHIFT]** is pushed at the same time, the modification is directly made on the ASCII text, otherwise the edit window is opened
- **Modal (Edit on request)**: if enabled, the window is configured in display mode only (direct editing is not possible). Select the **[Edit]** button to edit the working. Default setting: enabled
- **Immediate syntactic analysis**: if enabled, every time the setting of a working parameter is changed, the parametric setting is checked for validity. In case of wrong setting, the error is immediately signalled. Default setting: disabled
- **Confirm working changes**: select an item from the relevant list if the **Modal** box is unchecked. In

this case, with always disabled editing mode, it is possible to indicate how a change, not explicitly confirmed, is handled (by the **[Confirm]** button). Three options are listed:

  - **Not automatic confirm**: changes which have not been confirmed by the **[Confirm]** button are lost (default)
  - **Automatic confirm**: changes are automatically saved
  - **Confirm on request**: changes are confirmed on system request.

- **Subroutine variables**: two options are listed to select the layout window for <r> variable assignment, in application of subroutine:
  - **List as parameters**: variables are listed and arranged in two columns per item: header column (variable description) and assignment column (default)
  - **List as variables**: variables are listed and arranged in a table similar to that used for program <r> variables.
- **Modify check boxes:**
  - **double click**: this option allows to change the checkbox state in the working assignment area by double-clicking with the mouse (default value)
  - **single click**: this option allows to change the checkbox state in the working assignment area by single-clicking with the mouse
- **Show ASCII names**: if enabled, the window shows parameter ASCII names, in addition to the relevant descriptive messages. Otherwise: only these last are displayed. Default setting: disabled
- **Divide in equal columns**: if enabled, the window adjusts the width of the two (Header and Assignment) columns so that they are of the same size. Otherwise: the header column is autosized so as to fit and fully display the longest item. Default setting: enabled

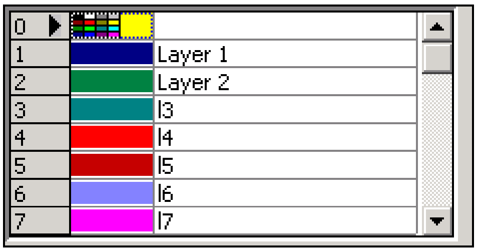
<u>Program ASCII text</u>

- **Working property assignment mode**: it is a table with no more than 4 rows, one for each working property which can be selected in list: "O", "L", "B", "K". Three selection columns are provided for each row:
  - **In List selection**: if enabled, the property is selected in list, otherwise an edit box is handled
  - **In List name**: if enabled, also the name assigned to the property is displayed in list;
  - **Apply color**: if enabled, this option applies the color assigned to the property, in the working preview. If disabled: with In List Selection active, the names assigned to the property are in any case displayed

  Here are the default settings:
- O field: In List Selection: active; In List Name: inactive; Apply color: inactive;
- L field: In List Selection: active; In List Name: active; Apply color: active;
- B field (construct): In List Selection: active; In List Name: inactive; Apply color: active;
- K field (block): all fields are inactive. In List Selection and/or Apply Color fields can be activated for the "K" field only if the custom numbering is enabled.

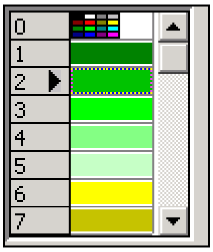
Here are a few examples:

a selection list for the "L" field, with the relevant applied colors and assigned names

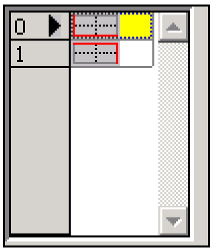


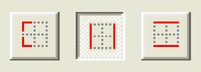
0 ▶	🟨	
1	🟦	Layer 1
2	🟩	Layer 2
3	🟩	l3
4	🟥	l4
5	🟥	l5
6	🟦	l6
7	🟪	l7

a selection list for the "B" field, with color application and without name assignment



a selection list for the "O" field (maximum value =1), with visualization of the relevant reference bitmaps and without name assignment.

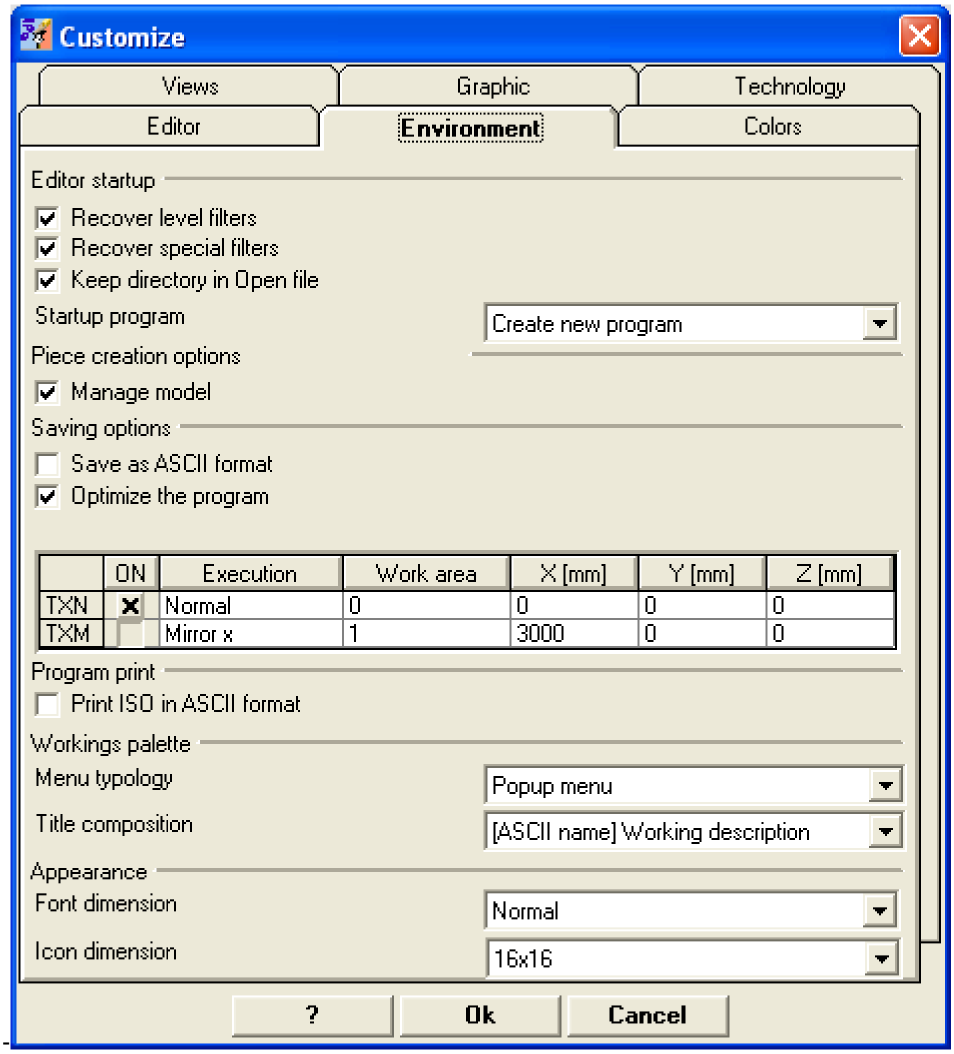


- **Bitmaps for O field references**: it is possible to select this option, by checking the relevant box, only if the O field has a maximum setting value not greater than 3. In these cases, if enabled, the graphic list shows the bitmaps relative to the O field interpretation as reference on the selected face. Otherwise: the selection list for the O field displays the colors applied (if color application is activated) and/or the names assigned to every single value. Default setting: enabled. The first four values are associated with the four edges of the face:
  - O Field = 0 edge at X=0,Y=0
  - O Field = 1 edge at X=0,Y=hf. In this case the bitmaps can be associated with two programmable values (0,1), so as to point out their technological relationship. Three selections are possible, as provided in the bitmaps. 
  - O Field = 2 edge at X=lf,Y=0
  - O Field = 3 edge at X=lf,Y=hf
  - O Field = 4 edge at X=0, Y=0 that remains fixed, apart from the execution mode of the piece (normal or mirror)
- **On/Off column in ASCII text**: if the working ASCII text is enabled, this option allows to display the column which returns the verification status of logical conditions. Default setting: enabled
- **View column in ASCII text:** if the working ASCII text is enabled, this option displays column for working graphical representation.Default setting: enabled
- **Edit ASCII text**: if enabled, this option activates the direct editing of the working ASCII text (in the program list window). Default setting: disabled
- **Indent TAB**: this item sets the number of spaces corresponding to a tab indentation (TAB) in program ASCII text. This field can be set to values ranging from 0 and 9. The default setting is 2. Tabulations are used to highlight the program logical structure and then they are evaluated on if (IF, ELSE, ENDIF) and for (FOR, ENDFOR) cycles.

Disabled are after use

- **Apply tool to a copy of the workings**: if enabled, this option deactivates the selection after one use. Default setting: disabled
- **Apply tool to workings in clipboard**: if enabled, this option deactivates the selection after an application. Default setting: disabled
- **Snap on Entity:** if enabled, the selection of Snap on Entity is not disabled. Default setting: enabled.
- **Window technology:** if enabled, this option closes the technology window after having selected directly the technology (during the insertion and/or the change of a working). Default is disabled

13.2 Environment



Editor startup

☑ Recover level filters

☑ Recover special filters

☑ Keep directory in Open file

Startup program: Create new program

Piece creation options

☑ Manage model

Saving options

□ Save as ASCII format

☑ Optimize the program

	ON	Execution	Work area	X [mm]	Y [mm]	Z [mm]
TXN	☒	Normal	0	0	0	0
TXM	□	Mirror x	1	3000	0	0

Program print

□ Print ISO in ASCII format

Workings palette

Menu typology: Popup menu

Title composition: [ASCII name] Working description

Appearance

Font dimension: Normal

Icon dimension: 16x16

Editor startup area

- **Recover level filters**: if enabled, at the startup of *Tpaedi32*, the free or locked status of *Levels* are recovered as they were set the last time the program was started. Otherwise: neither visibility nor editing restriction on *Levels* is predefined. Default setting: disabled. This option is not available, if the Level property is not managed.
- **Recover special filters**: if enabled, at the startup of *Tpaedi32*, the free or locked status of *Special Filters* are recovered, as they were set the last time the program was started. Otherwise: neither visibility nor editing restriction on *Specials Filters* is predefined. Default setting: disabled. This option is not available, if the special Filters are not managed.
- **Keep directory in Open file**: if enabled, the *Open File* window displays the directory and the file typology corresponding to those which were selected the last time the program was started. Default setting: disabled.
- **Startup program**: this option selects the working state at the start up of the program. Three options are listed:
  - create new program: this item applies the default prototype file. This is the default setting
  - load last file
  - do not open the program

Saving options area

- **Save in ASCII format:** if enabled, the program is memorized as ASCII format. The ASCII format lends itself to a more intuitive reading and it can be used for instance to generate programs for Tpaedi32.

Hereafter the row that corresponds to the registration of a drilling work (operation code: 81; name ASCII: HOLE)

for the ASCII format and for the internal format respectively: HOLE WS1 EG0 X100 Y100 Z-12 TD10 TMC1 TR1 TP1 W#81{::WTp WS=1 #8015=0 #1=100 #2=100 #3=-12 #1002=10 #201=1 #203=1 #1001=1 }W. Wenn a program in ASCII format is read, the removal of program lines can be determined, if it is not found any correspondence between the charged workings and the workings defined in the workings database. Default setting is disabled.

- **Optimize the program**: if enabled, after the program has been stored, program optimization is performed. Optimization modes are assigned in the "**TXN**", "**TXM**" table. Default setting: disabled
- **Save piece matrix**: if enabled, when the program is optimized, the piece matrix is recorded on the disc (file .TXN and/or TXM) and it corresponds to the optimized matrix for the execution (any path and/or steps and/or tool changes may be optimized).

The item is enabled only if **Optimize the program** is selected. Default setting: disabled

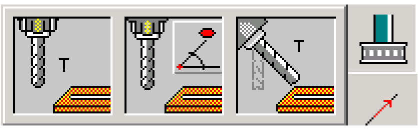
- **"TXN", "TXM" Table**: this table defines the Execution modes which are assigned for program optimization, when the piece matrix is saved. The table is enabled only if **Optimize the program** is selected. The table consists of two rows: one for the creation of a matrix in normal execution, the other for the creation of a matrix in mirrored execution. Each row is divided into six columns:
  - **ON**: if selected, it enables optimization
  - **Execution**: type of execution in list (for mirrored matrix only)
  - **Work area**: work area identification number
  - **X[mm] Y[mm] Z[mm]**: step offset, on the 3 axes of the coordinate system. Coordinates are to be set in: [mm] or [inch] (As assigned in *Tpaedi32* configuration)

Program Print area

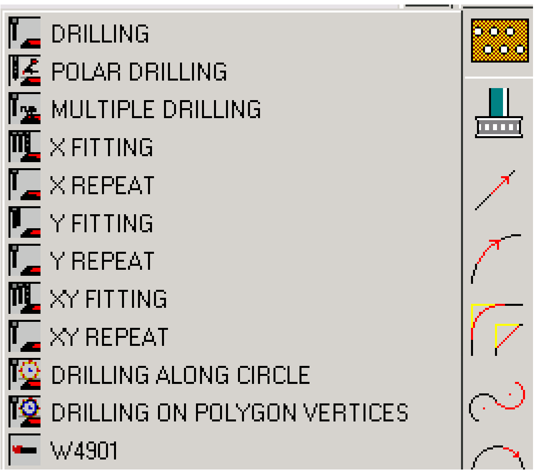
- **Print ISO in ASCII format**: if enabled, the text print-out displays the working ASCII format. Otherwise: the text print-out shows the default working format. Default setting: disabled

Workings Palette area

- **Menu typology**: this item allows to choose between two options relative to the visualization of workings palettes (those opened by selecting the workings palette buttons)
  - **Normal Toolbar**: this is the default setting. Workings palettes display the buttons (a bitmap for each button) shown in Figure:



  - **Popup Menu**: workings palettes are displayed with a pull-down menu, as shown in Figure:



Each listed item shows the relevant working bitmap reduced to the size of an icon together with a working description, as defined by the following **Title composition** setting.

- **Title composition**: three options are listed:
  - **Working description**: this is the default setting (example: FORO)
  - **[ASCII Name] Working description** (example: [G81] FORO)
  - **ASCII Name** (example: G81)

  If workings palettes are displayed as Normal Toolbar, the title is used in the tooltip text
  If workings palettes are displayed as popup Menu, the title is used in the drop-down menu.

Appearance area

- **Font size**: this option selects the font size in piece assignment area, ASCII text, workings menus. Three options are listed:
  - normal
  - large
  - very large.

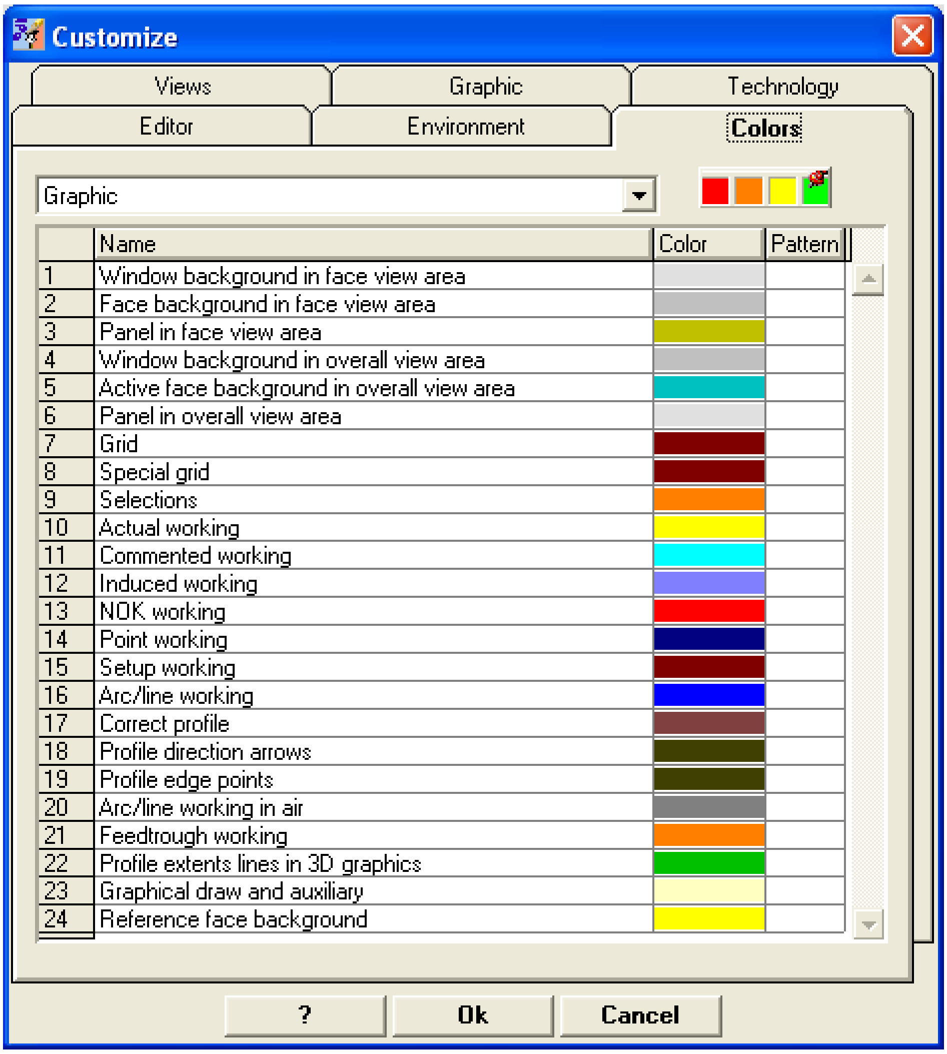
The selection does not change the characters of other types of window (e.g. tool assignment, file selection, tooltip)

- **Icons dimensions :** this function selects the dimensions of the toolbar icons in the main menu and of the icons displayed in the piece assignment area. The images are stored in folders, which have the same name as the dimention of the selected icon. (For example: "16x16").Three or four options are possible:
  - default
  - 16x16
  - 24x24
  - 32x32

  The installed Tpaedi32 provides the images in BMP format, matching the two dimensions (16x16) and (24x24). Possible changes of the images itself should leave the names, the dimensions and the 24 colours-mapping unchanged. The dimensions, whose corresponding folders are available, are shown. Choosing the default option requires the use of the internal resources of Tpaedi32.

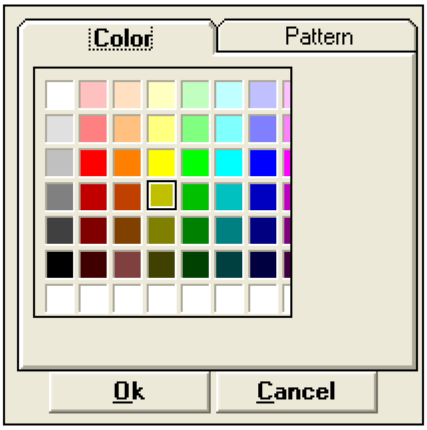
13.3 Colors

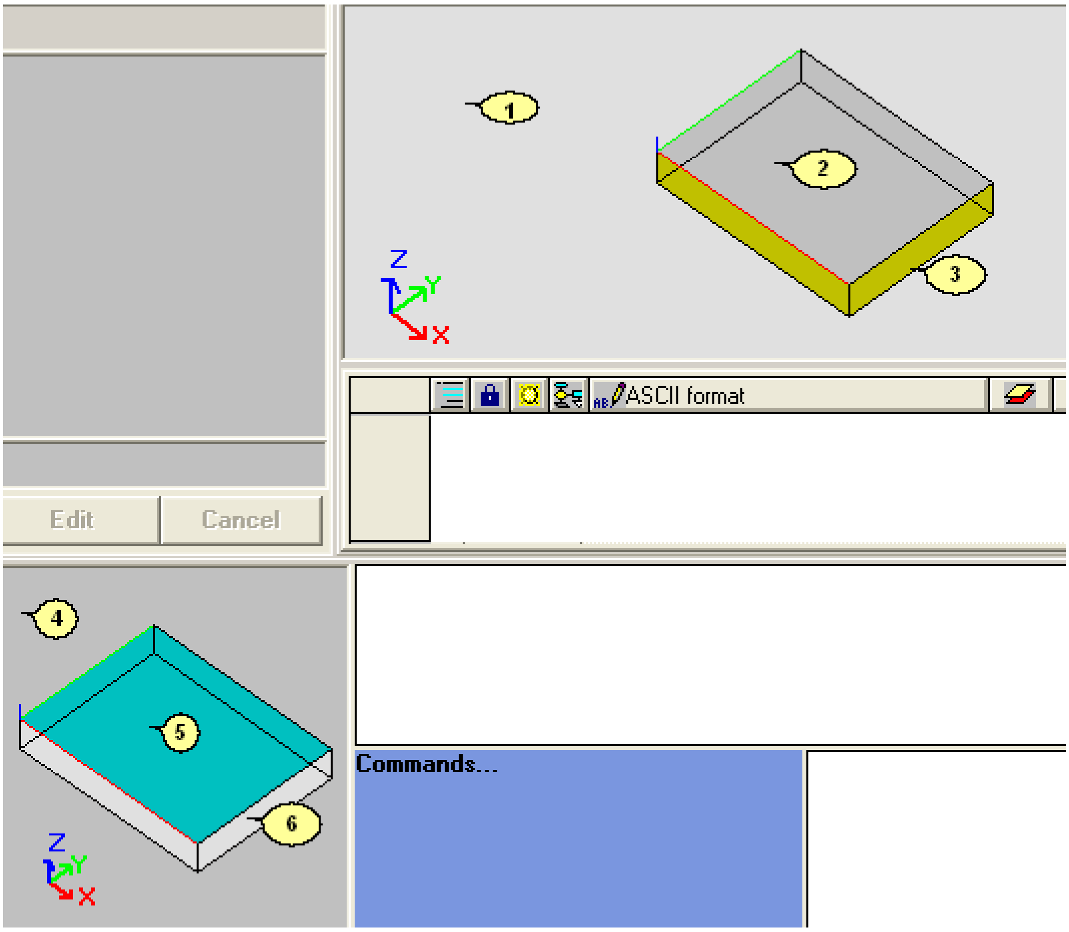
Graphic display colors



	Name	Color	Pattern
1	Window background in face view area	⬜	
2	Face background in face view area	⬜	
3	Panel in face view area	🟨	
4	Window background in overall view area	⬜	
5	Active face background in overall view area	🟦	
6	Panel in overall view area	⬜	
7	Grid	🟥	
8	Special grid	🟥	
9	Selections	🟧	
10	Actual working	🟨	
11	Commented working	🟦	
12	Induced working	🟪	
13	NOK working	🟥	
14	Point working	🟦	
15	Setup working	🟥	
16	Arc/line working	🟦	
17	Correct profile	🟫	
18	Profile direction arrows	🟫	
19	Profile edge points	🟫	
20	Arc/line working in air	⬛	
21	Feedtrough working	🟧	
22	Profile extents lines in 3D graphics	🟩	
23	Graphical draw and auxiliary	🟨	
24	Reference face background	🟨	

To set a color click on the relevant field of the **Color** column, the color selection window (*Colors*) is displayed:



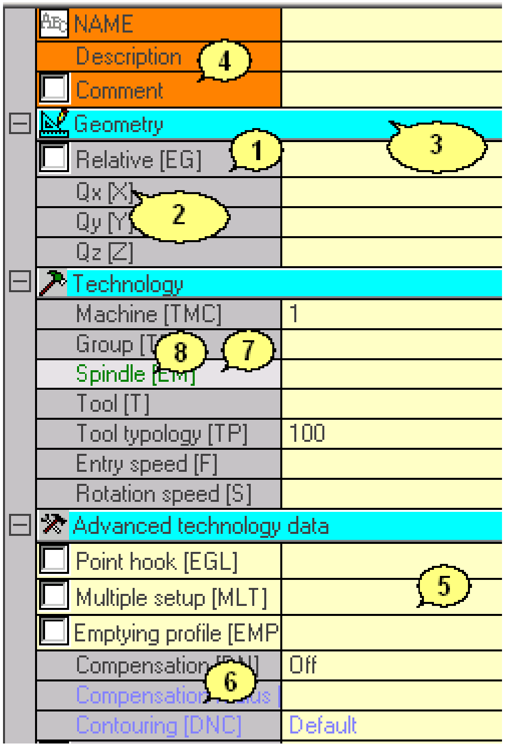


- **Window background in face view area**: view window background color (in current face view area) (figure: color **1**). A pattern coloring the background of the view window can also be set, replacing the use of the color. The pattern is chosen in the column **Pattern** after enabling the selection by clicking left on the top side. The patterns displayed are charged from the image file called PATTERN, memorized in the configuration folder (cadcfg\custom). The file format can be BMP, JPG, GIF. The whole image is divided into cells of 32x32 pixels. If the file does not exist, a default image is displayed.
- **Face background in face view area**: current face background color (in current face view area) (figure: color **2**). A pattern filling the face can also be set, replacing the use of the color. The pattern is chosen in the column **Pattern** after enabling the selection by clicking the checkbox left on the top side. The patterns displayed are charged from the image file called PATTERN, memorized in the configuration folder (cadcfg). The file format can be BMP, JPG, GIF. The whole image is divided into cells of 32x32 pixels. If the file does not exist, a default image is displayed.
- **Panel in face view area**: panel background color (in current face view area) (figure: color **3**). It is also possible to set a pattern filling the panel, replacing the use of the color.
- **Window background in overall view area**: view window background color (in piece overall view area) (figure: color 4). It is also possible to set a pattern coloring the background of the piece overall view window, replacing the use of the color.
- **Active face background in overall view area**: current face background color (in piece overall view area) (figure: color 5). It is also possible to set a pattern filling the active face, replacing the use of the color.
- **Panel in overall view area**: panel background color (in piece overall view area) (figure: color **6).** It is also possible to set a pattern filling the panel, replacing the use of the color.
- **Grid**: color of grid elements
- **Special Grid**: color of special grid elements
- **Selections**: identification color of selected workings (in face view)

- **Actual working**: current working identification color (in face view)
- **Commented working**: commented working identification color (in ASCII text list)
- **Induced working:** Induced working identification color (in ASCII text list)
- **NOK working:** identification color for workings compiled with error (in ASCII text list)
- **Point working**: graphic display color for point workings
- **Setup working**: graphic display color for setup workings
- **Arc/Line working**: graphic display color for works on profile -arcs and lines-
- **Correct profile**: graphic display color for profiles with applied tool radius compensation
- **Profile direction arrows**: graphic display color for direction arrows applied to profile
- **Profile edge points**: graphic display color for edge points applied to segments of profile
- **Arc/Line working in air**: graphic display color for segments of profile executed in air: if an arc or a linear segment is fully or partially executed in air, the concerned part is marked by this color. *Attention*: property (level, B field, ..) colors are given priority over general colors
- **Feedthrough arc/line working**: graphic display color for feedthrough segments of profile exceeding the face thickness: if an arc or linear segment is fully or partially executed so as to be feedthrough, the concerned part is marked by this color. *Attention*: property (level, B field, ..) colors are given priority over general colors.
- **Working of feedthrough profile:** graphic display color for the segments of profiles, exceeding the face thickness: if an arc or a linear segment is fully or partially executed so as to be feedthorugh, the concerned part is marked by this color. Properties' colors (level, B field,...) are given priority over general colors.
- **Profile extents lines in 3d graphics**: display color for profile extents lines in 3d graphics
- **Graphical draw and auxiliary**: color used in:
  - drawing functions
  - interaction of tools with the mouse (translation point, mirror axis)
  - expanded list management in program ASCII text
- **Reference face background**: reference face background color, in form of fictive face edge assignment

🟥🟧🟨🟩 The button sets up the graphical colors according to a default colors set.

Working data-entry check display colors



NAME	
Description	
□ Comment	
⊟ Geometry	
□ Relative [EG]	
Qx [X]	
Qy [Y	
Qz [Z]	
⊟ Technology	
Machine [TMC]	1
Group [T	
Spindle [EM]	
Tool [T]	
Tool typology [TP]	100
Entry speed [F]	
Rotation speed [S]	
⊟ Advanced technology data	
□ Point hook [EGL]	
□ Multiple setup [MLT]	
□ Emptying profile [EMP	
Compensation [DN]	Off
Compensatio ius	
Contouring [DNC]	Default

- **Control background**: control background color (figure: color **1**)

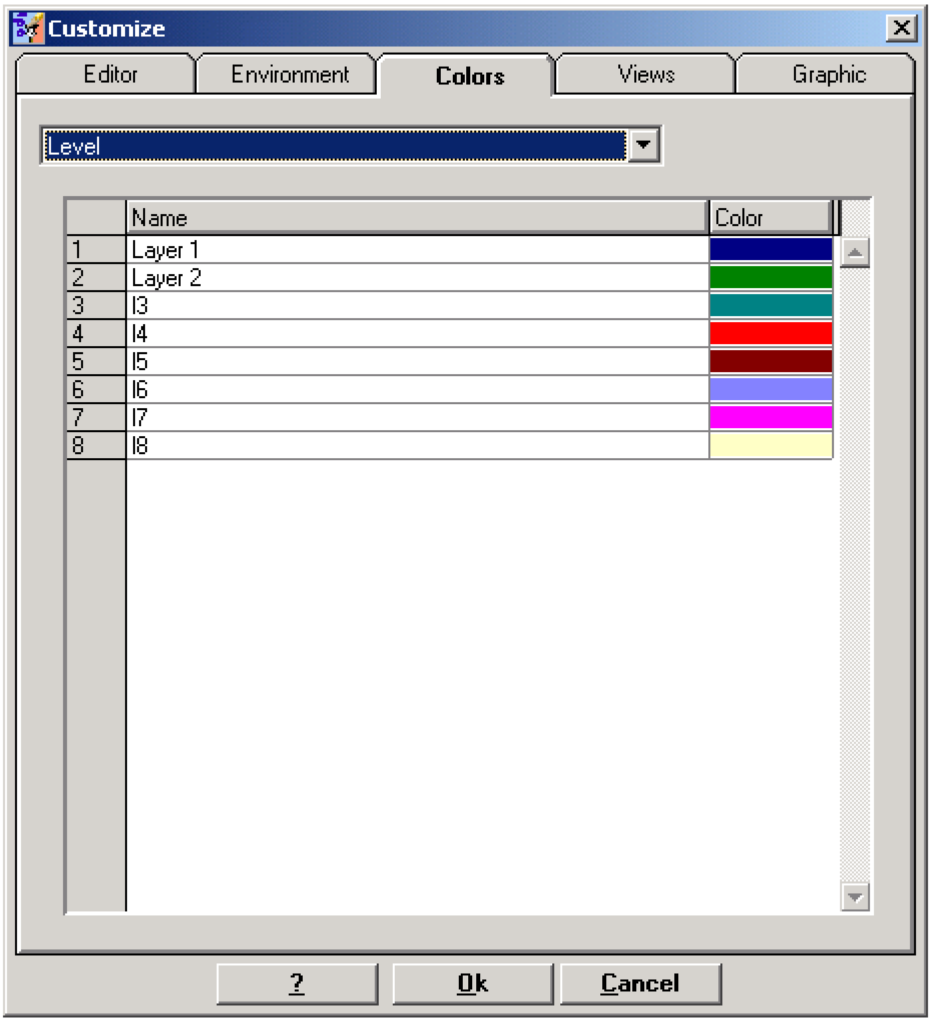
- **Text color**: text color (figure: color **2**)
- **Node heading background**: node background color (figure: color **3**)
- **Properties items background**: background colors for property rows (figure: color **4**)
- **Values column background**:background color for the right-hand data-entry column (figure:color **5**)
- **Element text not editable**: text color for non editable text elements (figure: color **6**)
- **Selected item background**: background color for the selected item (figure: color **7**)
- **Selected text element color**: text color for the selected text element (figure: color **8**)

🟥🟧🟨🟩 The button sets up the graphical colors according to a default colors set.

Level display colors

Select the item from the list

This item is unavailable if the L (Level) property is not managed



Level

	Name	Color
1	Layer 1	🟦
2	Layer 2	🟩
3	l3	🟦
4	l4	🟥
5	l5	🟫
6	l6	🟦
7	l7	🟪
8	l8	🟨

The table consists of a number of rows equal to the maximum value which can be assigned to levels (in figure: 8)

- **header**: level number
- **Name**: name to be assigned to the level. If not set, a default name is assigned
- **Color**: click on the field to select a color for the concerned level

All workings which have a "L" (Level) field value different from 0 can be displayed with the color here assigned to the level value. The color field depends on the level default use and is an aid to the graphical representation.

It is possible to manage a level value which excludes the working graphics. See Chapter ***Graphics->Graphics Specials***

Construct display colors

Select the **Construct** item from the list. This item is unavailable if the property is not managed.
The table consists of a number of rows corresponding to the maximum value which can be assigned to constructs and its layout is equal to that of the previous property:

- **header**: construct number
- **Name**: name to be assigned to the construct. If not set, a default name is assigned
- **Color**: click on the field to select a color for the relevant construct

All workings which have a "B" (Construct) field value different from 0 can be displayed with the color here assigned to the construct value.

O Field display colors

Select the **O Field** item from the list. This item is unavailable if the property is not managed.
The table consists of a number of rows corresponding to the maximum value which can be assigned to the property and its layout is equal to the previous case:

- **header**: property number
- **Name**: name to be assigned to the property. If not set, a default name is assigned
- **Color**: click on the field to select a color for the relevant property

All workings which have a "O" field value different from 0 can be displayed with the color here assigned to the property value.

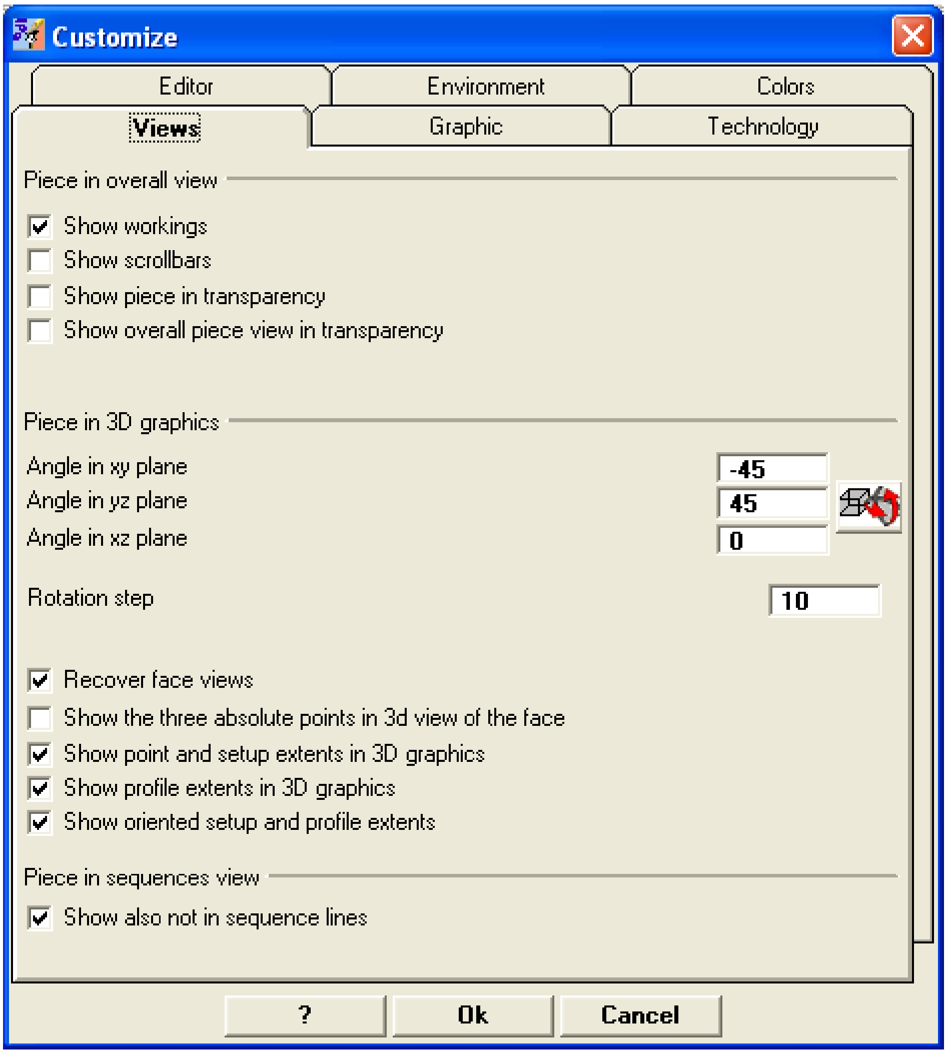
K (Block) Field display colors

Select the **Block** item from the list. This item is unavailable if the property is not managed or if it is handled with automatic numbering.
The table consists of a number of rows corresponding to the maximum value which can be assigned to the property and its layout is equal to the previous case:

- **header**: property number;
- **Name**: name to be assigned to the property. If not set, a default name is assigned
- **Color**: click on the field to select a color for the relevant property

All workings which have a "K" (Block) field value different from 0 can be displayed with the color here assigned to the property value.

13.4 Views



Piece in overall view

☑ Show workings
□ Show scrollbars
□ Show piece in transparency
□ Show overall piece view in transparency

Piece in 3D graphics

Angle in xy plane -45
Angle in yz plane 45
Angle in xz plane 0

Rotation step 10

☑ Recover face views
□ Show the three absolute points in 3d view of the face
☑ Show point and setup extents in 3D graphics
☑ Show profile extents in 3D graphics
☑ Show oriented setup and profile extents

Piece in sequences view

☑ Show also not in sequence lines

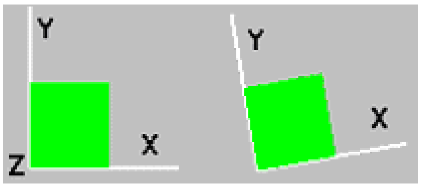
Piece in overall view area

- **Show workings**: if the relevant box is checked, this option enables the piece overall view, with applied workings. Default setting: enabled
- **Show scrollbars**: if selected, this item enables the scrollbar display and handling in overall view. Default setting: disabled
- **Show the piece in transparency:** if selected, this option enables the piece representation in transparency. In General View and in Face View the backgournd, the piece and the face are represented using one only color or pattern. (Color selected in the page Colors under the option Window Background in Face View Area). The transparency effect can be also obtained while setting the same colors or patterns in all the representation parts of the piece. However, in this case the graphical efficiency fails. Default setting: disabled.
- **Show overall piece view in transparency:** if selected, this option enables the piece representation in Overall View. The piece, the background and the current face are represented using one only color or pattern. (Color selected in the page Colors under the option Window Background in Overall View Area). The transparency effect can be also obtained while setting the same colors or patterns in all the representation parts of the piece. Default setting: disabled.

Piece in 3d graphics area

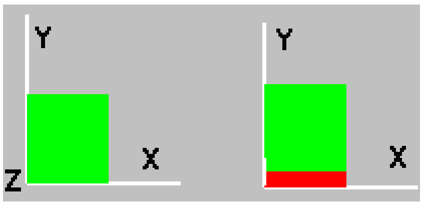
- **Angle in xy plane**: it is the default rotation angle in the screen display plane. The figure below shows the effect of rotation in xy plane: the left figure displays zero rotation angles; the right figure shows a non-zero angle in xy plane.

-



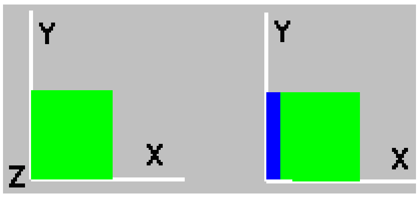
If the assigned value is positive, it rotates in counterclockwise direction; if negative, it rotates in clockwise direction. The field can only accept values within the range –360.00 - +360.00. Default value: -45.

- **Angle in yz plane**: it is the default rotation angle around the horizontal axis of the screen display plane. The figure below shows the effect of rotation in yz plane: the left figure displays zero rotation angles; the right figure shows a non-zero angle in yz plane.
-

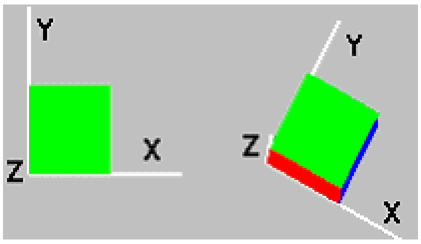


If the assigned value is positive, it rotates upwards; if negative, it rotates downwards. The field can only accept values within the range –360.00 - +360.00. Default value: 45.

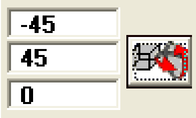
- **Angle in xz plane**: it is the default rotation angle around the vertical axis of the screen display plane. The Figure below shows the effect of rotation in xz plane: the left figure displays zero rotation angles; the right figure shows a non-zero rotation angle in xz plane.
-



If the assigned value is positive, it rotates leftwards; if negative, it rotates rightwards. The field can only accept values within the range –360.00 - +360.00. Default value: 0



The Figure above shows the effect of a non-zero rotation on all 3 planes.



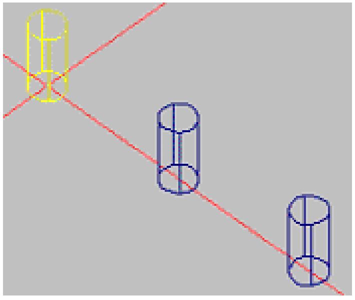
-45
45
0

: it sets the three value matching the current rotation of the piece. The button can be seen only if a program is open and if the 3d graphic view is selected.

- **Rotation step**: unit increment value applied in piece rotation. The field can only accept values within the range –360.00 - +360.00. Default value: 10
- **Recover face views:** if selected, it displays the plane or three-dimensional view of the face and, in

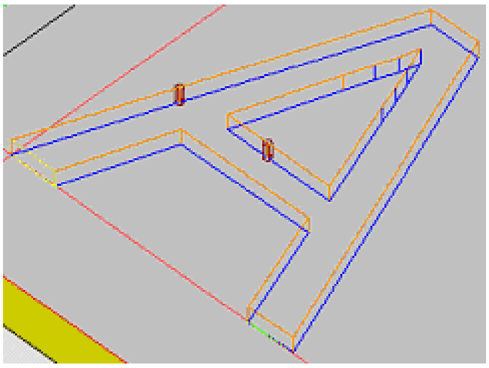
case of three-dimensional view, the rotation angles according to the last views set. If not selected, the overall view of the piece is displayed by applying the three value of the rotation angles as set in the options **Angle in xy, xz e yz plane** described above; the face views are displayed in plane. The default is not enabled.

- **Show the three absolute points in 3d view of the face:** if selected, the three absolute points in 3d view of the face appears. if not selected, the three represented absolute points are the three points showing the face orientation. The default is not enabled.
- **Show point and setup extents in 3d graphics**: if the relevant box is checked, this option enables the display of point and setup extents in 3d graphics: the point and setup extents show the space occupied by the tool (tool extents) on piece.



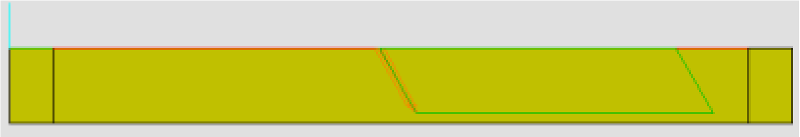
The Figure above shows an example of point (or setup) 3d graphics and displays the relevant extents: each working is represented in 3d by a small cylinder. Default setting: disabled

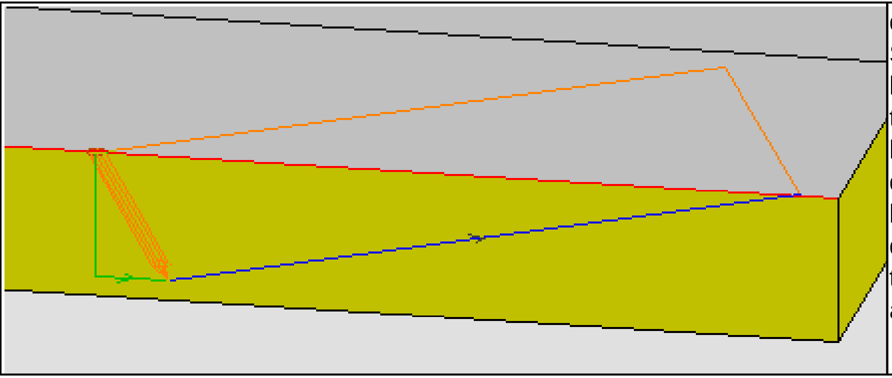
- **Show profile extents in 3d graphics**: if selected, this option enables the display of profile extents in 3d graphics: the profile extents highlight the space occupied by the tool (tool extents) on piece. This option is not applied to oriented setup workings, whose reference is the option **Show oriented setup and profiles extended.**



The figure above shows an example of profile 3d graphics and displays the relevant extents: each segment of profile highlights the space occupied by the tool (tool extents) on piece. Default setting: disabled.

- **Show oriented setup and profiles extended :** it is possible to select the extent display mode, that is the tool overall dimension in the three-dimensional view of setup working and oriented profiles. If the option is not selected, the programmed profiles with oriented setup are shown as programmed profiles with vertical setup and the options **Show point and setup extents in 3D graphic** and **Show profiles extents in 3D graphic,** if selected, are applied. If the option is selected, an oriented setup working is represented by a cylinder drawn according to the working parameters. The profile associated with the oriented setup is shown with oriented lines, according to the assigned parameter in the setup. If the option **Show profiles extents in 3D graphic** is not enabled, the only oriented line is shown. The default is disabled.

	Both the drawings are an example of thee-dimensional view of oriented profile, where the display of the overall

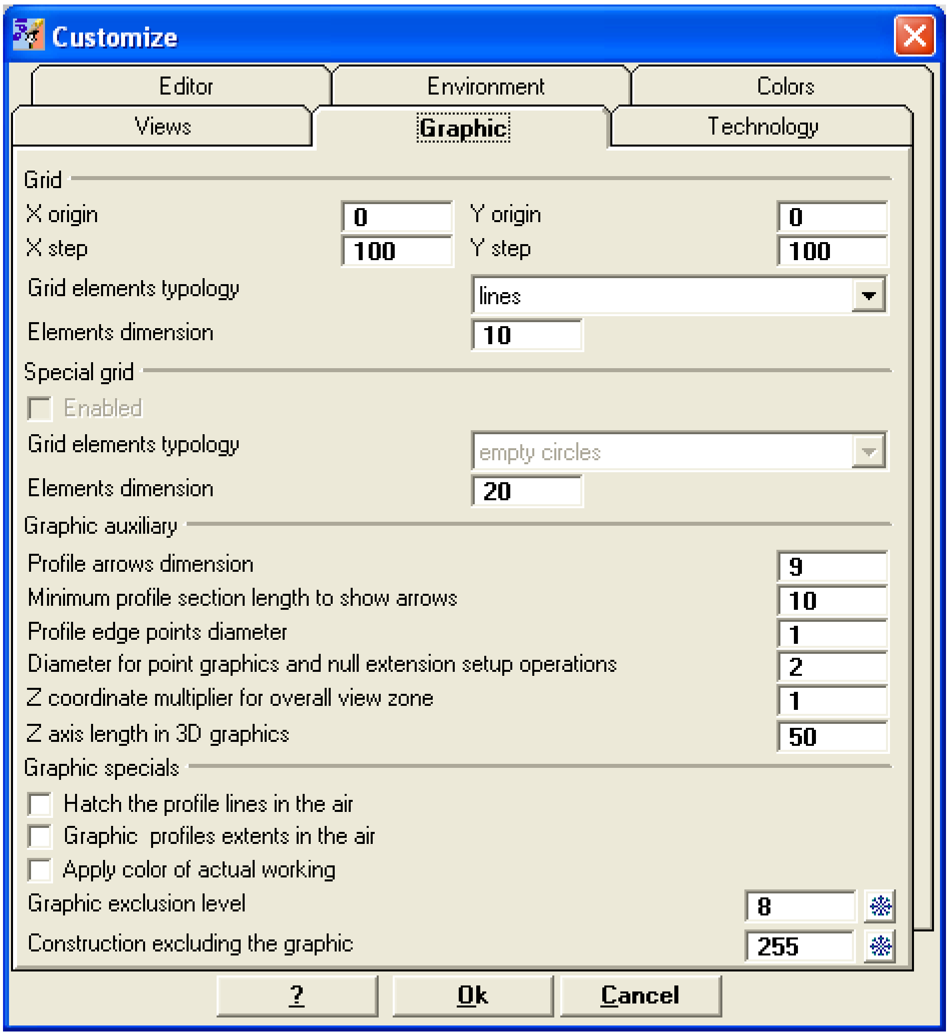


dimensions is enabled. Setup is indicated on the left, with the tool point turned downward. Profile line points out the overall dimensions of the tool between the piece (line down) and the line touching the plane of the face (line above).

Piece in sequences view area

- **Show also not in sequence lines**: if selected, this option enables a complete display in sequences view. If it is not selected, a few items are not shown in sequences view, that is: open profiles, induced workings, workings for which the sequence management is disabled. Default setting: disabled.

13.5 Graphic



Grid

X origin 0 Y origin 0

X step 100 Y step 100

Grid elements typology lines

Elements dimension 10

Special grid

□ Enabled

Grid elements typology empty circles

Elements dimension 20

Graphic auxiliary

Profile arrows dimension 9

Minimum profile section length to show arrows 10

Profile edge points diameter 1

Diameter for point graphics and null extension setup operations 2

Z coordinate multiplier for overall view zone 1

Z axis length in 3D graphics 50

Graphic specials

□ Hatch the profile lines in the air

□ Graphic profiles extents in the air

□ Apply color of actual working

Graphic exclusion level 8

Construction excluding the graphic 255

Grid area

- **X origin**: grid x origin, on the current face plane. To be set in: [mm] or [inch] (units of measurement for setting parameters). Default value: 0.0.
- **Y origin**: grid y origin, on the current face plane. To be set in: [mm] or [inch] (units of measurement

for setting parameters). Default value: 0.0.

- **X step**: grid step along the face view x axis. To be set in: [mm] or [inch] (units of measurement for setting parameters). The relevant field can accept a minimum value of 1 mm. Default value: 32 mm
- **Y step**: grid step along the face view y axis. To be set in: [mm] or [inch] (units of measurement for setting parameters). The relevant field can accept a minimum value of 1 mm. Default value: 32 mm
- **Grid element typology**: grid display options are listed here below:
  - **lines**: the grid consists of horizontal and vertical lines, spaced according to set step values. The intersection points of lines are the grid points. This is the default option
  - **empty circles**: the grid is displayed with empty circles, centered on grid points
  - **crossed circles**: the grid is displayed with crossed circles, centered on grid points
  - **empty squares**: the grid is displayed with empty squares, centered on grid points
  - **crossed squares**: the grid is displayed with crossed squares, centered on grid points
- **Elements dimension**: grid element size. In case of empty or crossed circles, this option assigns the circle diameter. In case of empty or crossed squares, this option assigns the square side. To be set in: [mm] or [inch] (units of measurement for setting parameters). The relevant field can accept a minimum value of 1 mm. Default value: 10 mm.

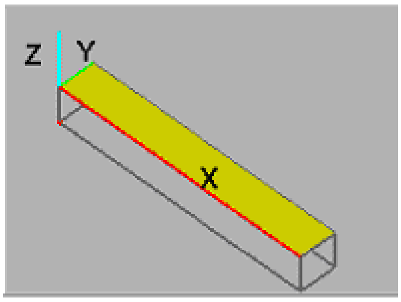
Special grid area

- **Enabled**: if this option is selected, it enables the special grid management. It is about a grid directly assigned by single points and it is defined during configuration by the machine manufacturer. Default setting: disabled
- **Grid elements typology**: grid display options are listed here below:
  - **empty circles**: the grid is displayed with empty circles, centered on grid points. This is the default option
  - **crossed circles**: the grid is displayed with crossed circles, centered on grid points
  - **empty squares**: the grid is displayed with empty squares, centered on grid points
  - **crossed squares**: the grid is displayed with crossed squares, centered on grid points
- **Elements dimension**: grid element size. In case of empty or crossed circles, this option assigns the circle diameter. In case of empty or crossed squares, this option assigns the square side. To be set in: [mm] or [inch] (units of measurement for setting parameters). The relevant field can accept a minimum value of 1 mm. Default value: 10 mm.

Graphic auxiliary area

- **Profile arrows dimension**: this option allows to set the length of the two profile direction arrow segments. To be set in: [mm] or [inch] (units of measurement for setting parameters). The relevant field can accept a minimum value of 1 mm. Default value: 20 mm.
- **Minimum profile section length to show arrows**: this option allows to set the minimum profile section length to show the direction arrow. To be set in: [mm] or [inch] (units of measurement for setting parameters). The relevant field can accept a minimum value of 0 mm. Default value: 5 mm.
- **Profile edge points diameter**: this option allows to set the profile edge points diameter. To be set in: [mm] or [inch] (units of measurement for setting parameters). The relevant field can accept a minimum value of 0.1 mm. Default value: 1 mm.
- **Diameter for point graphics and null extension setup operation**: this option allows to set diameter for point graphics and null extension setup operation. If the working is a sort of construct working, the assigned value is not considered.

  To be set in: [mm] or [inch] (units of measurement for setting parameters). The relevant field can accept a minimum value of 0.5 mm. Default value: 5 mm.
- **Z coordinate multiplier for overall view zone**: this option allows to set the piece thickness (Z coordinate) multiplier, in overall view area. The following values are accepted: minimum=1, maximum=10. Default: 1. In current view area no Z coordinate multiplier is applied.
- **Z axis length in 3d graphics**: this option allows to set the length of the face z axis, in 3d graphics. To be set in: [mm] or [inch] (units of measurement for setting parameters). The relevant field can accept a minimum value of 1 mm. Default value: 50 mm.

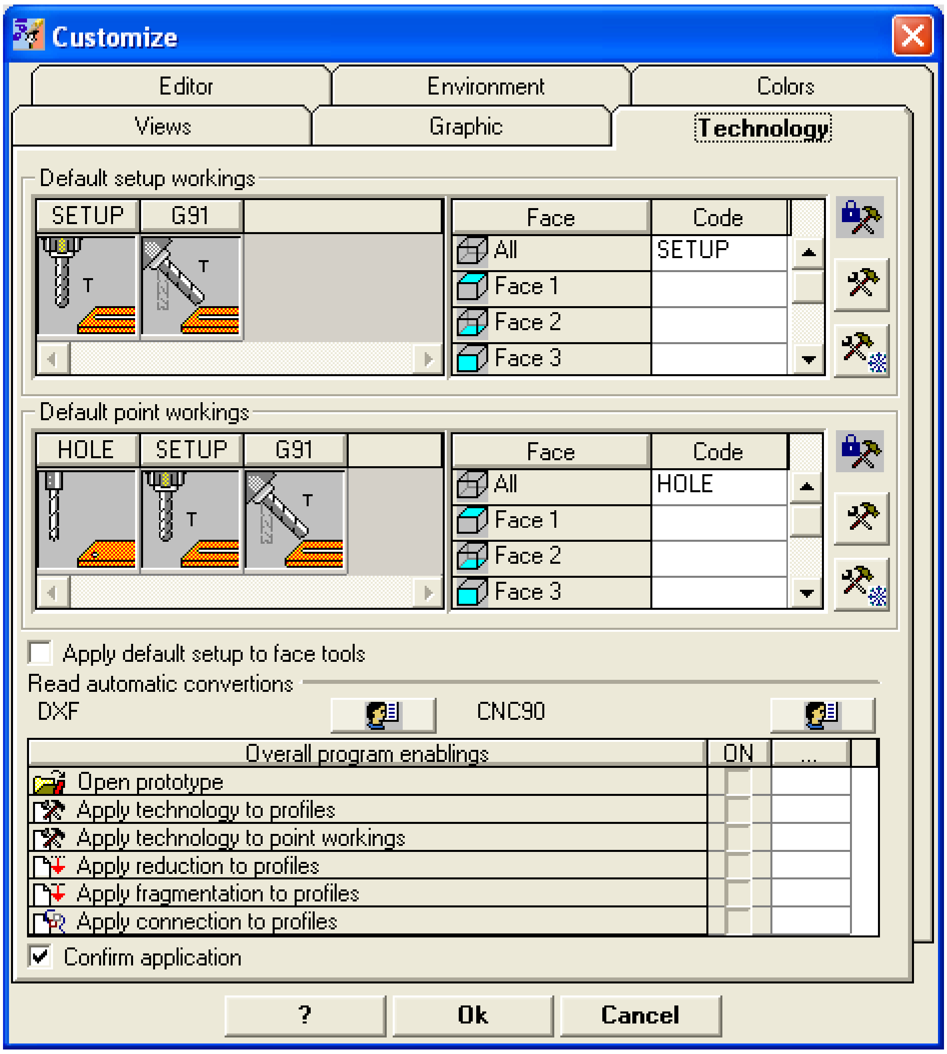


The Figure above shows a 3d view. Conventional notations have been assigned to the three face axes: X and Y on the face plane and Z as depth axis. A Z-axis extension in air is displayed: it starts from the face origin and its length is assigned by the current setting (multiplied by the **Z coordinate multiplier**, but only if in overall view area).

Graphic specials area

- **Hatch the profile lines in air:** if selected, it enables the display of the profile segments traced in the air. Default setting is disabled.
- **Grafic profiles extent in the air:** if selected, it enables the view of the tool extent also on the profile segments executed in the air. The default is not enabled.
- **Apply color of actual working**: if selected, this option enables the representation of the current working with the color set on page Colors under the option Actual Working. If not selected, the current working is represented using the color, taken from the programmation, for instance, it can be used the color taken from the level or construct properties or the color defining the working typology (punctual or profile segment). Default setting is enabled.
- **Graphic exclusion level**: Level ("L" field) value which excludes the working graphics. This parameter is used, for example, when the working graphics is fully based on the use of constructs. The following values are accepted: minimum=0 (in this case: it never operates), maximum=255. Default value: 255. By double-clicking the mouse on the symbol (snowflake) beside the edit field the graphic display can be enabled again. By double-clicking the mouse on the symbol (snowflake) beside the edit field the graphic display can be enabled again. This option is not available, if the property Level is not managed.
- **Construction excluding the graphic:** construct value ("B" field) excluding the graphic display of the working. The parameter's utility can be applied, when a construct working is used to generate other workings, for instance after working applying geometric transforms . Values between 0 (here: it never works) and 255. Default value is 255. By double-clicking the mouse on the symbol (snowflake) beside the edit field the graphic display can be enabled again. This option not available, if the property Construct is not managed.

13.6 Technology



Default setup workings

SETUP | G91

Face	Code
All	SETUP
Face 1	
Face 2	
Face 3	

Default point workings

HOLE | SETUP | G91

Face	Code
All	HOLE
Face 1	
Face 2	
Face 3	

□ Apply default setup to face tools

Read automatic convertions

DXF

CNC90

Overall program enablings	ON	...
Open prototype		
Apply technology to profiles		
Apply technology to point workings		
Apply reduction to profiles		
Apply fragmentation to profiles		
Apply connection to profiles		

☑ Confirm application

Default setup workings

The displayed data can be changed only if the program is closed. If the page is open during the edit of a program, the bitmap is shown to indicate that the data cannot be changed.

In this area technological data are set for the setup workings that derive from the application of profile tools (e.g. Profiles cut), from the application of complex workings such as text generation of emptying, from the execution of open profiles or headed profiles with geometric setup code. (A geometric setup code can only derive from a conversion from another size such as DXF...)

In the right table it is possible to set setup codes and different technological parameters for every face.

Face	Code
All	It sets a common technology for each face. It is applied when a face has not his own technology set.
Face 1	It sets a technology for the face 1 (above)
...	
Face 6	It sets a technology for the face 6 (the rear face)
Fictive Faces	It sets a technology for all the fictive and automatic faces

How to proceed to set the technology:

- select the row of the face for which a technology must be set (right table ***Face***, ***Code***).
- double click the mouse on one of the displayed setup workings (in the figure SETUP and G91). The working name of the selected working is carried in the right table in the column **Code**. The displayed workings are those available in the working palette, except those with polar programming.
- click the mouse on 🛠 and set the technological data. The dimensional parameter (quotes and speed) must be assigned with particular care, because they must be set according to the measurement unit defined in the the configuration ([mm] or [inch] for the quotes; [m/min] or [inch/sec]-[inch/min] for the speeds. Each value can be set in a parametric form. For each setting, a signal is shown, when the parametric programming is not valid.

Default point workings

The displayed data can be changed only if the program is closed. If the page is open during the edit of a program the bitmap 🛠 is displayed to show the data that cannot be changed.

In this are the technological data are set for workings with geometric punctual code. The setting procedure of technology is the same as that described above for the technology assignment of setup workings. The only noticeable thing is that in the list of the usable codes there are both the punctual and the setup ones. (A geometric setup code can only derive from a conversion from another format such as DXF..).

In a punctual working by default the parameter Diameter is assigned according to the following rules:

- If the working of punctual geometric code has not any diameter value set, **substitution** is made
- If the working of punctual geometric code has a diameter value set, **no substitution** is made

🛠 Default Technology

The button Default Technology is displayed in the window only if the workings database the working for the assignment of technology default is present. The choice of technology is unique for all the faces and not dependent on the number of faces chosen in the column **Face**.

If you select the button, a window for the setting of technologic parameters is displayed respectively for:

- set up workings and complex codes of profile technology. This is a matter of complex codes, whose setting in the working database is a Sybtypology equal to 1, e.g. Ellipse and Pocket workings.
- Punctual workings and complex codes of drilling type. It is a matter of complex codes, whose setting in the working database is a Sybtypology equal to 0, e.g. Fitting and Repeat workings.

Data in the default technology window can be changed if it is necessary to:

- change the default assignment of a parameter as regards it was proposed during the insertion
- change the default assignment of a parameter and make it unchangeable.

E.g. if among the technology parameters a value = 1 is set in the machine parameter, this parameter will be displayed with value 1 at every working insertion. Anyway, it remains changeable. This parameters setting is preferable to the later, as it does not change the workings interpretation and does not change the programming already carried out.

If the programming of the Machine = 1 parameter is to be forced, it must be set:

- (1): value closed in round brackets or "v,1". This numbering works in such a way that the parameter may be seen in the workings insertion window, but not be changed.
- (1): value closed in square brackets or "h,1".. This numbering works in such a way that the parameter may not be seen in the workings insertion window, and not be changed.

Further settings are as follows:

- (): no value in round brackets or "V'. This notation allows the parameter to be seen into the window of working insertion; but it cannot be changed and it cannot have any assigned setting.
- []: no value in square brackets or "h". This notation allows the parameter to be seen but into the window of working insertion; it also cannot be changed and it cannot have any assigned setting.
-

This forced parameters setting can be only used in particular cases. An example ist a plant made of a **Machine** and a **Group** only. Another example is to choose a technology excluding the **Electrospindle** programming.

Those parameters, already defined as not changeable parameters in the working database, cannot be changed. Machine, Diameter,Group, Subgroup, Tool typology, Tool, Speed and Rotation parameters only can be forced. The set value can be expressed in a parametric form. For each setting, a signal is shown, when the parametric programming is not valid. Also in a macro-program the programming of a parameter can be forced. If this parameter is assigned in a r variable, during the storing process all modifications get lost and the forced value is set again.

- **Apply default setup to the face tools:** If it is selected into the face tools, that can assign a technology (Build a profile, Emptying, Text) the technology is assigned directly as shown in the Figure.
- **Default setup workings** In this case, in the tool the selection window of the operating code is not displayed. Therefore the setting window of setup parameters is directly opened and shows the values already assigned in the frame described as above. Default setting: disabled

Overall program enablings

Automatic activation mode of *Overall program tools* is assigned corresponding an importation from external format. Tpaedi32 applies one or more complex tools automatically.

The table displays six rows,one for each procedure, that can possibly be activated. For each row following columns are defined:

- **ON:** if selected it activates the automatic execution of the tool during the program importation.
- **Parameters (..) :** clicking on the cell a window of the assignments to apply in tool execution opens. Settings and speed quotes must be set in configuration measurement unit ([mm] or [inch] for the quotes; [m/min] o [inch/sec] for the speeds). For the option ***Open prototype*** a window is opened, where the complete path is carried, from which the imported program is initialized.
- **Confirm application:** if selected, confirmation is required while opening the program, before applying the tools; otherwise it applies the tools directly.

The activation of the option ***Open prototype*** determines the initialization of the program according to the prototype default file (the file *PIECE.TCN*, stored in the *cadcfg\custom* folder). If the file exists, the program initializes some information, reading them from the same prototype file: the 'o' and 'v' variables

the sections that are managed among Special Settings, joined Infos, Constrains, optimizing Settings.

the names of the faces

To activate the option *Apply technology to the punctual workings*, the same rule mentioned into the paragraph *Default point workings* and concerning the Diameter, is applied.

13.6.1 Read automatic conversions

le may contain graphical objects that do not need to be converted, while the geometric entities to convert need some additional information. Hence the need to have a mechanism that filters among the information those that are really necessary and that allow to introduce the missing information. Both the mechanisms of filter and the mechanisms of the integration use layers (levels) of DXF files , which allow the creation of groups of objects. The filter action is defined in the page **Levels.** Here the level to filter or, alternatively, those to convert are indicated. In the first case it concerns the functioning in relative logic, in the second one it concerns positive logic.

The integration action can use two alternative methods:

- **basic procedure**: the drawing is converted as geometric piece description and all the technological definitions are introduced directly in *Tpaedi32* after the conversion.

- **custom producedure**: the technological information are introduced in the phase of DXF drawing and the conversion can directly integrate them into the format for *Tpaedi32* . This procedure can be used only if the presence of a customized USB hardware key is detected.

Basic procedure

The converter carry out automatically the conversion according to the following guidelines:

- points are convert into point-related workings, with assigned diameter brought from the drawing.
- lines are convert into setup working, followed by a L01 linear : Pf(X,Y,Z)
- arcs are converted into setup working, followed by an A01 arc : xy(Xf,Yf,center,rot),Zf;
- polylines are converted in setup working, followed by A01 or L01 workings, according to the type of segment to translate.
- piece dimensions are deducted from the *piece level* (see later) or they are determined from the global overall dimensions of the converted workings.

Working codes with which points and setup, that open the profiles, are assigned in a

- special list. In particular:
  - *Apply technology to profiles* assigns setup workings;
  - *Apply technology to point workings* assigns the point-related working.

- Otherwise: it assigns geometric codes, therefore without technological parameters, that can be later replaced in the program.

Custom procedure

The basic mechanism of the custom procedure requires to characterize with the same level (block) the geometric entities that have a common technological characterization and to the level (block) a name is applied, that contains the technological data.

Also the custom procedure applies the selected logic for the levels (included or excluded). Some geometric elements, that correspond to any face defining the basic program parallelepiped of Tpaedi32 can be inserted in a two-dimensional (2D) DXF-File. In this case the level of the geometric entities (see on page ***Reference systems***) is interpreted.

The level usage is instead not necessary in case of three-dimensional (3D) drawings. According to the configured UCS, the correspondent face Tpaedi32 is selected.

According to the type of selected logic and to the included or excluded levels, the converter converts automatically in accordance with the following rules:

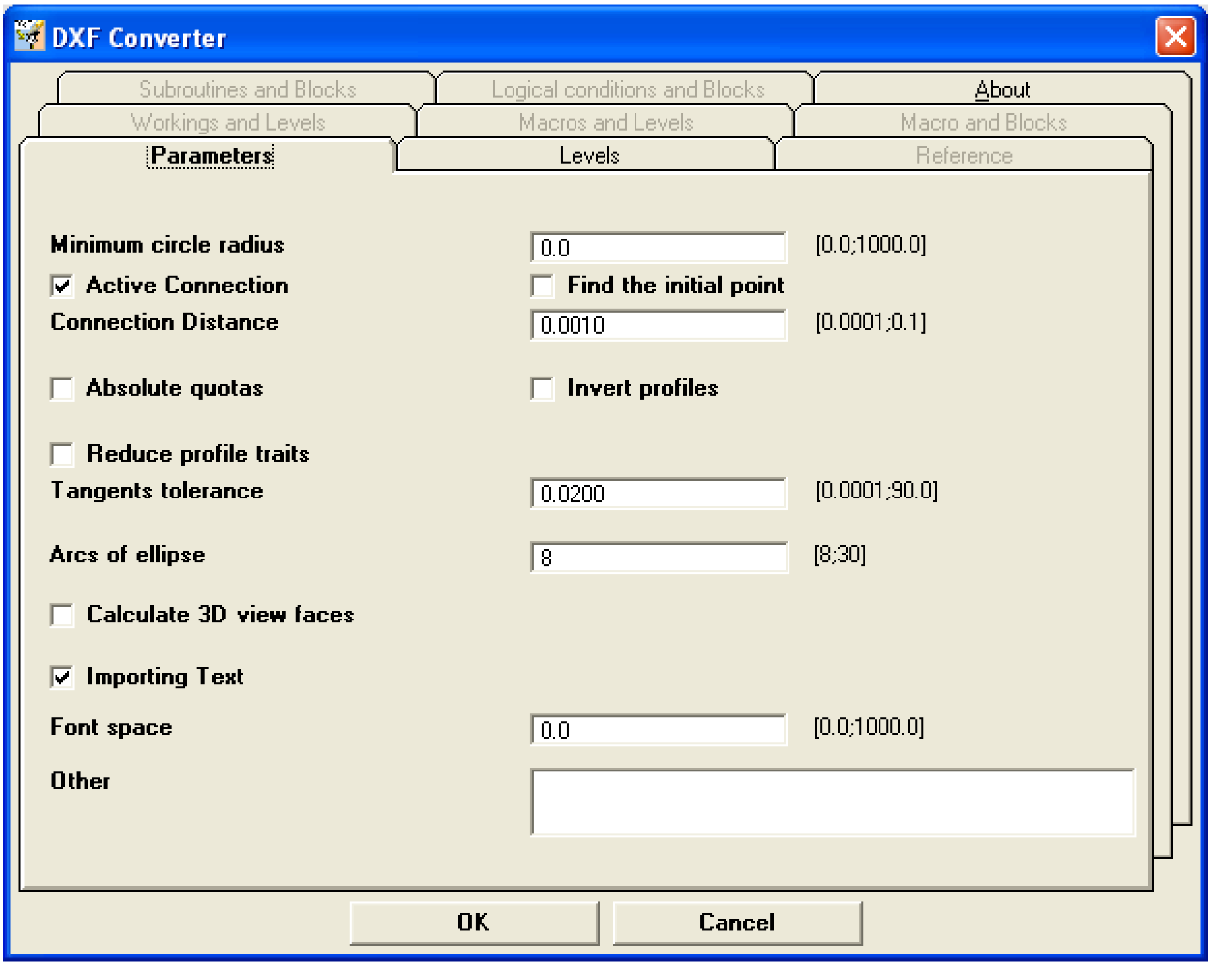
- entities with a level (block) where the prefix in defined in the homonymous list are translated with the corresponding workings (see on pages **Workings and levels, Macros and levels, Subroutines and blocks, Macros and blocks,** ***Logical Conditions***). The technological characteristics of the working are inserted, considering the defined parameter list for the prefix.
- entities whose levels were not treated in the point before are converted according the default criteria, that are described in the basic procedure.
- Piece dimensions are brought from *piece level* (see later) or they are determine from the overall dimensions of the translated workings.
- The geometric entities, that can be translated, are converted either to working macros in one of the basic faces (1 to 6) according to the presence in the level of the face identifier, or of the UCS set in the 3D DXF. If there are levels with face indicators there must be also the level with the panel specifications.
- The Z coordinate is brought in from the drawing, if it not specified in the level.

General criteria

While creating a DXF files you need to consider the following restrictions of a general nature:

- DXF drawings must have a **1:1** ratio, with millimeters used as a unit of measure
- The converter only interprets elements such as: **points, circles, segments, arcs, polylines made by arcs and segments, ellipses.**
- in the **basic procedure** the drawing is supposed to lay in XY plane for the faces 1 and 2, in the XZ plane for the faces 3 and 5, in the plane XZ for the faces 4 and 6. Drawing **reference** is defined on the **bottom left** corner.

Parameters



Minimum circle radius 0.0 [0.0;1000.0]

☑ Active Connection

□ Find the initial point

Connection Distance 0.0010 [0.0001;0.1]

□ Absolute quotas

□ Invert profiles

□ Reduce profile traits

Tangents tolerance 0.0200 [0.0001;90.0]

Arcs of ellipse 8 [8;30]

□ Calculate 3D view faces

☑ Importing Text

Font space 0.0 [0.0;1000.0]

Other

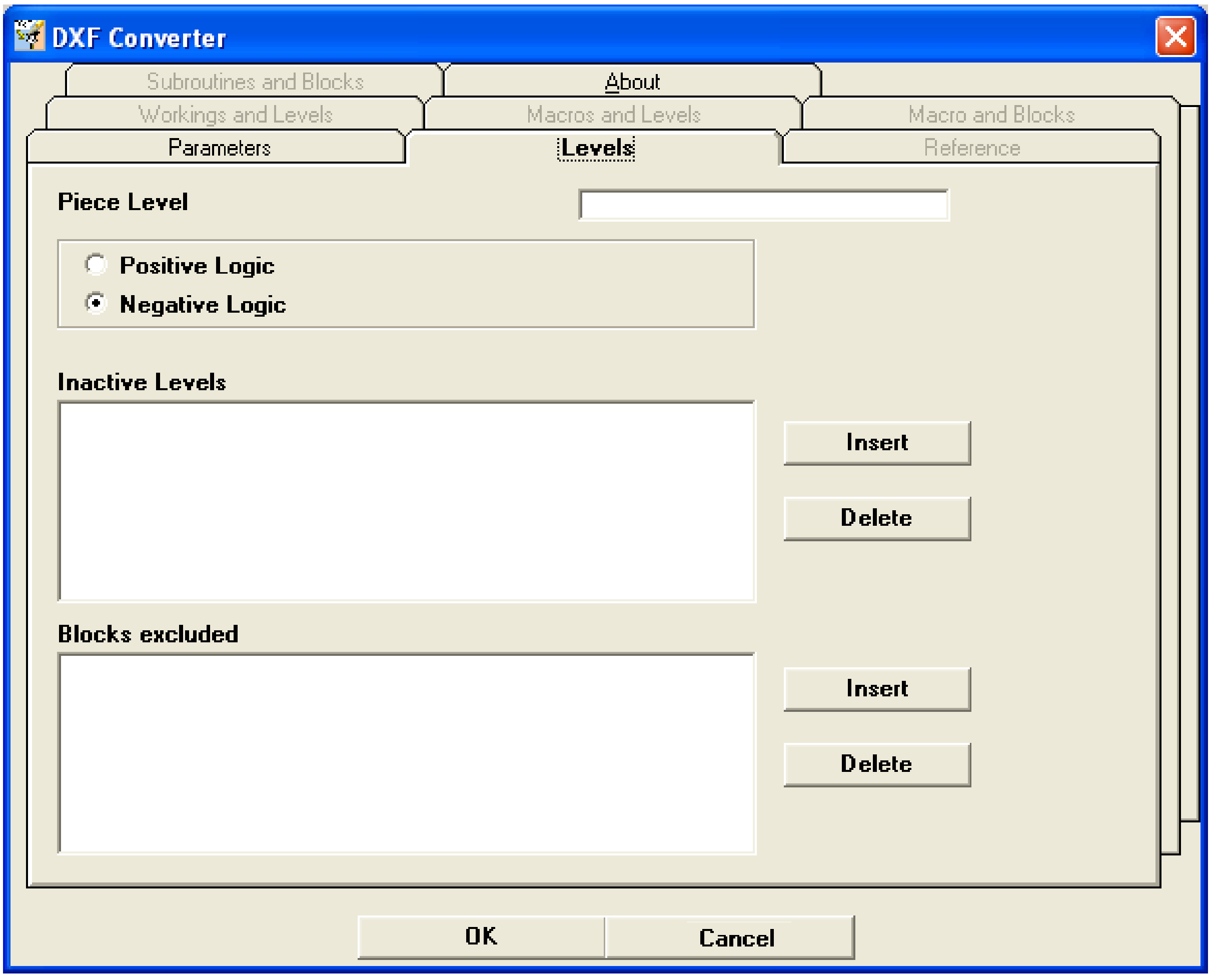
- **Minimum circle radius**: a circle with a radius lower than the Minimum radius value will be interpreted as a point, otherwise as an arc with a radius equal to that of the circle and start point coinciding with the end point.
- **Active Connection:** if selected, it enables the connection of the geometric elements of arc, line or polyline type of the same level and whose start and/or end points coincide. Two exterior points of a geometric entity must be considered coincident, if the difference of their quotes is lower than the **Connection Distance.** .Default is not active.
- **Find the initial point:** the selection is only considered if **Active Connection** is set. If selected, the conversion search in the DXF file the points whose coordinates correspond with the start or end coordinates of an arc. These points will become the workings start points. In this way a non oriented geometry turns in an oriented geometry.Default is not active.
- **Connection Distance:** setting is only considered, if **Active Connection** is set. It sets the longest distance used in the connection procedure.
- **Absolute Coordinates:** sets the coordinates of the polyline points as absolute**,** with respect of P(0,0) reference**.** If the option is not set**,** the conversion turns the coordinates of the polyline points from the absolute, with respect of P(0,0) reference, to the relative ones, with respect of the start point of the polyline . Default is not active.
- **Invert profiles:** enables inversion of the polylines
- **Reduce profile traits:** enables the reduction of the consecutive profile segments having the same tangent. Default is not active.
- **Tangents tolerance:** tolerance value of the tangents arcs and lines in order to reduce the profiles.
- **Arcs of ellipse:** solved arc numbers for each ellipse quadrant.
- **Compute 3D view faces :** if selected, this option adds up the heights values to the working depths, otherwise it reeds the depth of the Z coordinate. This option is used to import a 3D piece (advanced

performance) and the workings on the faces. Default is not active.

- **Import Texts:** if selected, this option imports the assigned texts into levels valid for import. If not selected, this options does not execute any importation, even if the texts are assigned into valid levels. Default is active.
- **Font space:** it sets the space dimension between the characters of a writing. The values should be between 0.0 and 1000.0. The default value is 0.0.
- **Piece thickness:** sets the program default thickness, used if the thickness is not set. The field accepts values from 1.0 to 1000.0.
- **Assign construct geometric workings** The option is only available with **custom procedure**. If the option is selected, all the profile or point-shaping workings, that do not match configured working prefixes or macros, are imported as a construct. If the option is not selected, or in case of **basic procedure,** the construct is not applied to workings.
- **Other:** field of text type. It should be used for assignments, that have been added between a release and the following one. As for the syntax, follow the specific directions. The length of the text cannot exceed 250 characters.

The above parameters and the guidelines are applied both in the basic procedure and in the custom procedure.

Levels



Piece Level

○ Positive Logic
● Negative Logic

Inactive Levels

Blocks excluded

- **Piece Level:** name of the level (layer) identifying in the DXF files the geometry of the overall dimensions of the piece. The geometric entities typically belonging to this level are the four segments forming a circle, originated in (0,0). In the piece level it is possible to set the panel thickness, by specifying the "ZP" string followed by a value, the program comment, by specifying the "ZP" string, followed by the description. If in the DXF the entities distinguished by this level are missing, the piece dimensions are taken by the largest overall dimensions of the translated graphical objects and translated to the axes origin.

- **Positive Logic/ Negative Logic:** it indicates the active working mode
  - **Positive**: the conversion only works on the geometric entities assigned on the levels in the list below (levels included)
  - **Negative**: The conversion does NOT work on the geometric entities assigned on the levels in the list below (levels included)
- **Inactive Levels**: list of the included or excluded levels. It is possible to insert in the list until 100 level names. Each name can be made of no more than 32 characters. The not allowed characters are **<>^:?*|`=".** The buttons **[Insert]** and **[Delete]** allow to add or to delete an option from the list. To import all the levels of the programs, the option **Negative Logic** should be selected, otherwise the names of the levels to convert have be edited in the list.
- **Blocks excluded**: list of the included or excluded blocks. It is possible to insert in the list until 100 block names. Each name can be made of no more than 32 characters. The not allowed characters are **<>^:?*|`=".** The buttons **[Insert]** and **[Delete]** allow to add or to delete an option from the list.

Reference systems

This page is accessible if a HW key is available and recognized as valid. The list allows you to assign the rules to import workings from geometric entities of the DXF drawing in any of the six basic faces of the piece. To provide to the system the necessary information for the conversion it is necessary to find the correspondence between the DXF drawing reference systems and the faces of the system *Tpaedi32.*

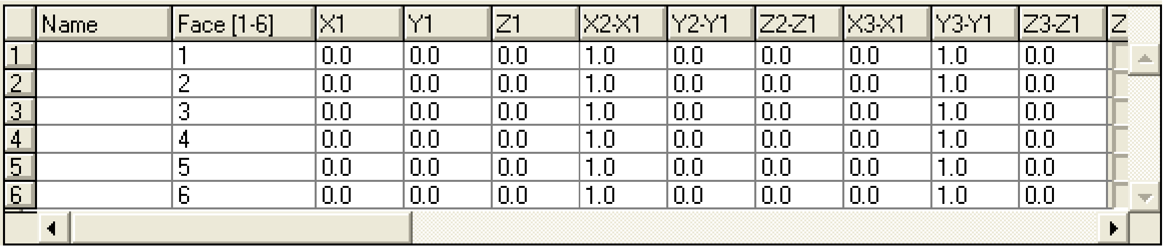
A face is determined by:

- **name**: Sequence of alphanumeric characters (max.10) available in the name of a level. The name is used in the 2D DXF drawing. All the geometric entities belonging to one side are characterized by different levels, whose name includes the name of the face. In the 3D DXF drawing, the indication of the face, to which an entity belongs, shall be deducted from the geometric information of the face (triad of the points of the plan and direction of the depth axis).
- **P1,P2 and P3 three points,** defining, with reference to a right-ward cartesian system, the oriented plan and therefore the face position in the space. Point P1 is the reference of zero in every face workings. Vectors P2-P1 and P3-P1, perpendicular to each other, define the plane orientation of the face;
- **depth axis direction:** with reference to the assigned plane. The depth axis can enter (left-wise axes set) or exit (right-wise axes set) from the face plane.

Every point of a face is assigned by means of three coordinates (X,Y,Z):

- Point P1 (X1,Y1,Z1): Places the face origin in the system of reference absolute (0,0,0) or relative to the level piece. The coordinates of Point P1 may also be expressed in the parametric form using as variables:
  - dimensions: L,H,S;
  - programmed coordinates: X,Y,Z
  - numeric values
  - mathematical operators of sum (+) and subtraction (-)
- Point P2 (X2-X1,Y2-Y1,Z2-Z1): places the point P2 in relative with reference to the P1 point. Quote is expressed in incremental form and typically values +1.0 o -1.0;
- Point P3 (X3-X1,Y3-Y1,Z3-Z1): places point P3 in relative with reference to the P1 point.

With the following settings:



	Name	Face [1-6]	X1	Y1	Z1	X2-X1	Y2-Y1	Z2-Z1	X3-X1	Y3-Y1	Z3-Z1	Z
1		1	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	
2		2	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	
3		3	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	
4		4	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	
5		5	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	
6		6	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	

all the predefined system reference user are reproduced in Autocad.

Workings and levels

This page is accessible if a HW key is available and recognized as valid. In this page are set the criteria for translation for the levels that are to translate. A working and its parameters are associated to the present characters in the name of the level of a DXF entity. In the DXF file the level is characterised by a 32 characters name:

- the first characters, that will be called prefix, are associated to a working that is present in the working database of the *Tpaedi32* workings, among those of punctual or setup type.
- the remaining characters are associated to the working parameters defined by the prefix.

The prefix:

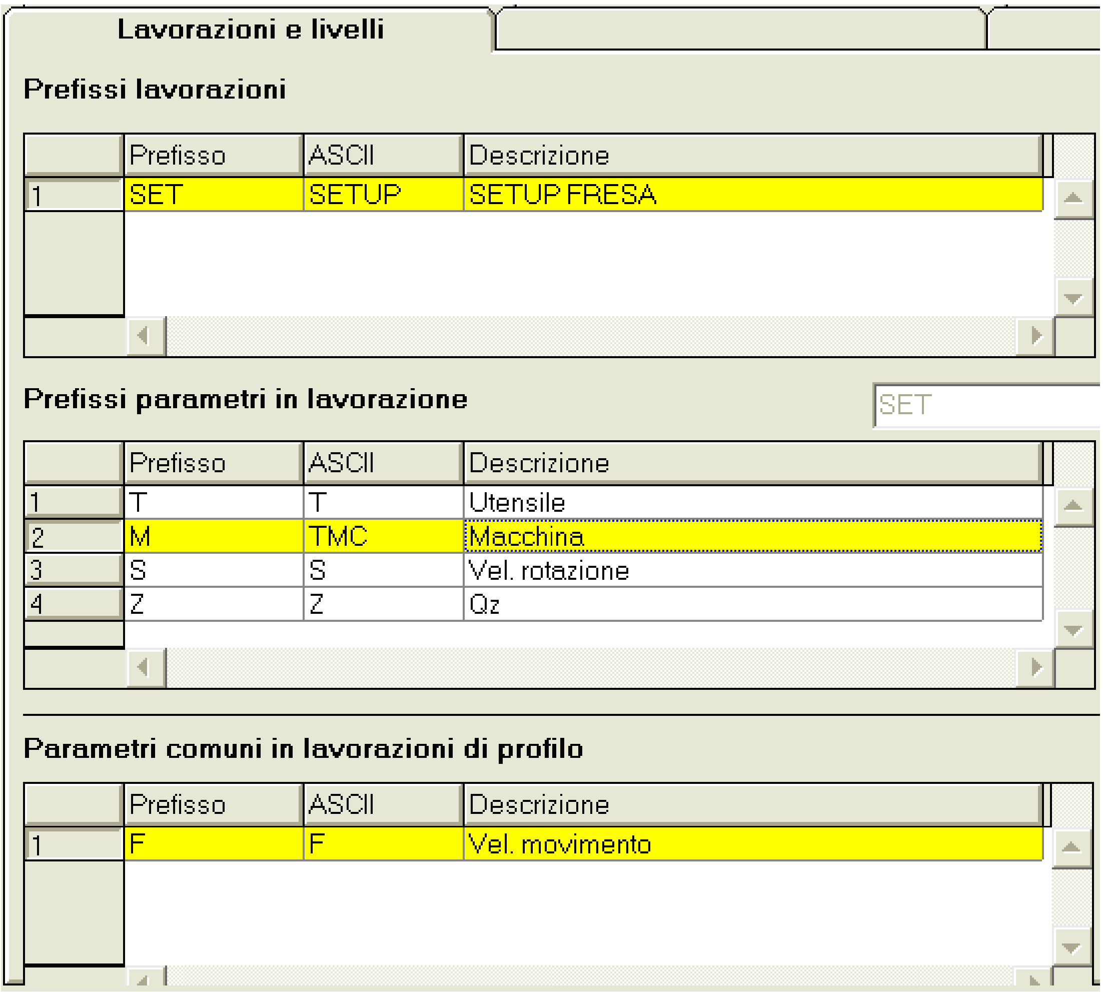
- is made of 1 to 30 alphanumeric characters
- cannot be repeated
- must be assigned

The indicators are assigned on a single alphabetical character. For each prefix it is possible to have up to 26 indicators.

In the level as assigned in DXF files:

- after an indicator may appear a number (with or without decimal figures) showing the value to be assigned to corresponding parameter
- as decimal separator it is possible to use the point (.), the comma (,) or the underscore (_).

In a dialog box up to 2 of the parameters common to all the profile workings (arcs or lines) may be assigned, as resulting from the working database of *Tpaedi32.* For instance:



Lavorazioni e livelli

Prefissi lavorazioni

	Prefisso	ASCII	Descrizione
1	SET	SETUP	SETUP FRESA

Prefissi parametri in lavorazione

SET

	Prefisso	ASCII	Descrizione
1	T	T	Utensile
2	M	TMC	Macchina
3	S	S	Vel. rotazione
4	Z	Z	Qz

Parametri comuni in lavorazioni di profilo

	Prefisso	ASCII	Descrizione
1	F	F	Vel. movimento

In the first list the prefix SET is associated to the working MILL SETUP. For this reason, following parameters are defined (second list):

Tool	associated to the indicator T
Machine	associated to the indicator M
Rotation speed	associated to the indicator S
Qz	associated to the indicator Z

The third list shows that there are some parameters not necessarily related to one only working, but to all profile workings. For instance, Feed(lav) is associated to the indicator F. As a result, if the translation reeds a level of the type SETM1T10S4000F5Z12_8, this last one is translated as
Milling on basic face
Tool 10
The speed rotation rate of the milling machine is 4000 rpm
The speed feed rate is 5 mt/min
Working depth is 12.8 mm.
After each indicator in the level a digit appears, representing the parameter value associated to the indicator; it can be an integer or a number with decimal, according to the type of parameter. When in the names of the DXF levels points and commas are not used, these can be replaced by the underscore. The application point of the working is taken from the DXF file.

Macros and levels

This page is accessible if a HW key is available and recognized as valid. In this page the criteria for translation are set for the levels that are to translate. A working and its parameters are associated to the present characters in the name of the level of a geometric entity of a point of line type of DXF. In this page it is only dealt with the correspondent workings to fix cycles (macroprogram codes).

For each working it is now possible to assign:

- up to 26 parameters of generic assignation (over one only alphabetical character)
- 3 parameters corresponding to the three coordinates of the application point.

An association Level-Macro was entered to code the sawing work. If in the DXF file the reference level of a line is associated to a macro (typically sawing work), the geometrical information of the line is transmitted to the macro. In particular the configuration variables with "&A" value are interpreted for the angle, "&U" for the module, "&X"/"&Y"/"&Z" for the final X/Y/Z of the line.

The option **Assign Z application point from the level** is applied to all the working assigned in the page. If the option is assigned and the level sets a "Z..." field, this one is assigned to the Z coordinate of the application point of the macro. If the option is not selected, the Z coordinate of the application point of the macro is assigned only with the corresponding parameter, if set in the table.

Macros and blocs

This page is accessible only if a hardware key is present and recognized as valid. In this page the criteria of translation are set for the levels to translate. A working is associated to the characters into the bloc's name and to the bloc's attributes its parameters. Only the workings that correspond to fixed cycles are considered (macro-program codes). The prefix name corresponds to the bloc's name and it can have a length from 1 to 25 alphanumerical characters, when the file, that records the bloc, is assigned.

For each working it is now possible to assign:

- up to 26 parameters of reassignable r-variables of the macro-program (over one only alphabetical character)
- 3 parameters corresponding to the quotes of the application point.

Subroutines and blocs

This page is accessible only if a hardware key is present and recognized as valid. In this page the criteria of translation are set for the blocs to translate. A working is associated to the characters into the bloc's name and to the bloc's attributes its parameters. The prefix name corresponds to the bloc's name and it can have a length from 1 to 25 alphanumerical characters, when the file, that records the bloc, is assigned. To the prefix a subroutine is associated, assigned with the name of the search, possibly with its extension and concerning the product\sub folder of the subroutines. A length of no more than 100 characters is accepted

For each working it is now possible to assign up to 25 parameters of of reassignable r-variables of the subroutine (over one only alphabetical character).

Logical conditions and blocs

This page is accessible only if a hardware key is present and recognized as valid. In this page the criteria of translation are set for the levels to translate. A logical working is associated to the characters into the bloc's name and to the bloc's attributes its parameters. The prefix name corresponds to the bloc's name and it can have a length from 1 to 30 alphanumerical characters, when the file, that records the bloc, is assigned.

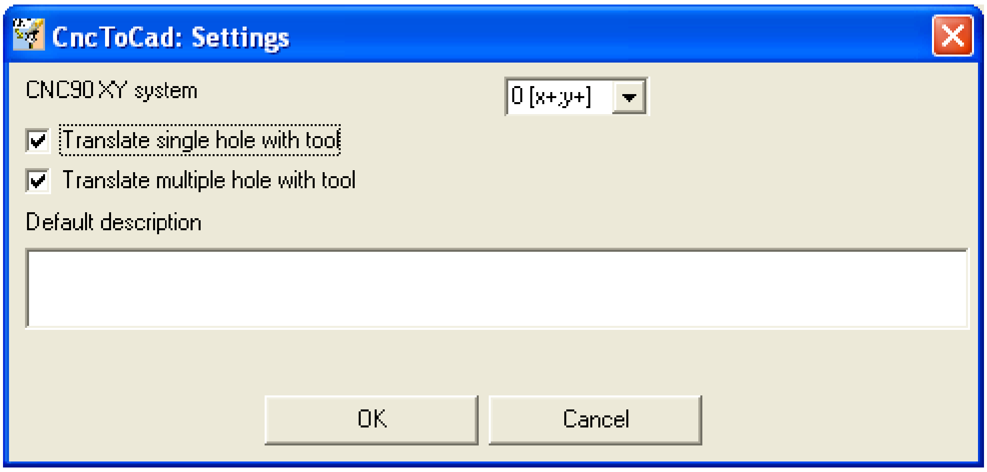
For each working it is now possible to assign:

- up to 26 parameters assigning working parameters (over one only alphabetical character)
- 3 parameters corresponding to the coordinates of the application point.

Information

It shows general information about the component (version, key presence, name of the current plant).

CNC90 **:** clicking on the button a dialog box opens, in order to set the parameters and the selections regulating the importation procedures of the files in CNC90 format.



CNC90 XY system 0 [x+;y+]

☑ Translate single hole with tool

☑ Translate multiple hole with tool

Default description

- **CNC90 XY System:** reference system used in CNC90. Default is 0. In a CNC90 system this is the value of the parameter *Reference System,* located in Technological Parameters->Settings-> Flags.
- **Translate single hole with tool:** if enabled, it turns the programmed drilling workings with only a tool into drilling workings according to the tool, otherwise it turn them in drilling workings for each diameter. The option must be selected if the folders and the files, allowing the technological correspondence between the environment CNC90 and the environment Tpaedi32 to be executed, have not been configured. Default setting is disabled.
- **Translate multiple hole with tool**: if enabled, it turns the programmed drilling workings with more than a tool in drillings working according to the tool. Only the first programmed tool is reported. If disabled, workings are turned in drillings for diameter: one working according to the programmed tool, with a step on xy-plane according with the declared correctors tools. The option must be selected, if the folders and the files allowing to execute the technological correspondence between the environment CNC90 and the environment Tpaedi32 to be executed have not been configured. Default setting is disabled.
- **Default Description**: this is the ccomment that the conversion procedure assigns to the program, if missing in the source program. The default setting is a not assigned comment.

14 Conversion programs

14.1 From Tpaedi32 format to Edicad format

Software CADTOEDI.EXE transcodes files created by ***Tpaedi32*** to be imported into ***Edicad*** environment.
Conversion only applies to pieces with program or sub-program types.

14.1.1 Transcoding mode

General information on piece

Piece general assignment are converted as follows:

- **Measuring unit and dimensions:** recovered in *Edicad* format. To recover a program in inches [inch], it is necessary to deactivate conversion on measuring unit, in *Tpaedi32* configuration, during creation phase of piece matrix. Otherwise, program in *Edicad* format is converted in [mm].
- **Comment:** recovered in *Edicad* format, 250 characters maximum.
- **"o" variables**: the first three "o" are specified in piece offsets in *Edicad* environment in numeric format. Each parametric form is calculated.
- **"v" variables**: specified in piece system variables in *Edicad* environment in numeric format. Each parametric form is calculated.
- **"r" variables**: assignments for "r" variables are lost.
- **Variable geometries**: fictive faces set are recovered. Assignment is referred to three edges, in numeric format. Each parametric form is calculated. Also assignment of a reference face or definition of a fictive face on geometry different from Cartesian assignment consisting of three edges is directly solved: piece in *Edicad* format consists of three coordinates of three face edges, anyway referred to piece absolute system. Also fictive faces are recovered when assigned empty or like construction auxiliary faces. In variable geometries, also faces assigned by program in face-piece are also recovered: in this case, face numbering refers to fictive faces, taking the first available numbers. Thickness settings for fictive faces are lost, as well as z axis direction setting.
- **Custom sections**: each assignment is lost.
- **Sequences:** for programmed working in piece, sequence assigned in primary list is recovered. Workings derived from expanded lists (such as sub-program application) are ordered again in compliance with sequence order, as assigned on all secondary lists; they appear in *Edicad* format with primary list sequence.

Programmed workings

In each face, conversion is made by maximum 32500 workings assigned in matrix.
Conversion involves all faces assigned to *Tpaedi32*, included face-piece. In this case:

- face-piece workings are sorted to the relevant assignment faces, before any workings directly assigned in faces.
- sequences assigned to face-piece workings allow keeping on top of sequences list, in the same progressive order as they were programmed in face-piece
- automatic faces are converted into fictive faces.

If the workings number exceeds 32500 for face:

- if *Tpaedi32* environment configures active sequences, exceeding workings are lost
- otherwise workings exceeding the maximum number are sorted to more programs. In this case, at least two programs are created in *Edicad* environment, after conversion.

Program in *Edicad*format only assigns workings with logic condition checked, as assigned in piece filed in *Tpaedi32* environment.
Moreover, conversion does not include system logic workings (cycles IF ..ELSE.. ENDIF, ERROR. J variable assignment).
Complex workings (profile, macro or sub-program application) are exploded during conversion into

Edicad format.
Each parametric assignment is calculated and replaced by numeric setting.

Point workings

Point workings have operating code between 1 and 1000.
Operating code [81] in *Tpaedi32* environment calculates a hole programming by tool and diameter; in *Edicad* environment, code [81] calculates a hole programming by diameter, while code [82] calculates a hole programming by tool.
Considering the above, code [81] is translated into:

- code [81]: if no tool is assigned (tool field with value 0);
- code [82]: if a value different from zero is assigned to tool field.

For all the other cases of point workings, conversion refers to operating code as assigned in matrix.
For all point workings, the following conversion rules apply:

***Tpaedi32* format**	***Edicad* format**
L field	Sets in Level field (value up to 8 only)
O field	Sets in Origin field (value up to 3 only)
field M	Sets in M field
X coordinate of point of application	Sets in Qx field
Y coordinate of point of application	Sets in Qy field
Z coordinate of point of application	Sets in Zp field
Machine	Sets in corresponding field, only if value is different from 0
Group	Sets in corresponding field, only if value is different from 0
Tool	Sets in corresponding field, only if value is different from 0
Tool typology	Sets in corresponding field, only if value is different from 0
Tool Diameter	Sets in corresponding field, only if value is different from 0
Rotation speed	Sets in corresponding field, only if value is different from 0
Operating speed	Sets in corresponding field, only if value is different from 0
Entry slowing value	Sets in corresponding field, only if value is different from 0
Exit slowing value	Sets in corresponding field, only if value is different from 0
Custom parameters	Sets the same custom parameter, only if value is different from 0

Setup

Setup workings have operating code between 1 and 1000.
Conversion shows an operating code as assigned in matrix. The following conversion rules apply to all setup workings:

***Tpaedi32* format**	***Edicad* format**
L field	Sets in Level field (value up to 8 only)
O field	Sets in Origin field (value up to 3 only)
M field	Sets in M field

X coordinate of point of application	Sets in Qx field
Y coordinate of point of application	Sets in Qy field
Z coordinate of point of application	Sets in Zp field
Machine	Sets in corresponding field, only if value is different from 0
Group	Sets in corresponding field, only if value is different from 0
Tool	Sets in corresponding field, only if value is different from 0
Tool typology	Sets in corresponding field, only if value is different from 0
Tool Diameter	Sets in corresponding field, only if value is different from 0
Rotation speed	Sets in corresponding field, only if value is different from 0
Operating speed	Sets in corresponding field, only if value is different from 0
Entry slowing value	Sets in corresponding field, only if value is different from 0
Exit slowing value	Sets in corresponding field, only if value is different from 0
C axis coordinate (rotation)	Sets in corresponding field
B axis coordinate (slewing)	Sets in corresponding field
Custim parameters	Sets the same custom parameter, only if value is different from 0

If matrix is generated with tool compensation applied, the assignments relevant to milling machine setup compensation are not set up by conversion.
For blade setup cases, compensation parameters are always set up by conversion, as not applied in matrix.

The following conversion rules apply:

***Tpaedi32* format**	***Edicad* format**
Tool compensation selected (off/ Left/Right)	Sets in corresponding field
Compensation radius	Sets in corresponding field, only if value is different from 0

Tool compensation result can be different in *Tpaedi32* and *Edicad*: *Tpaedi32* improves recovery of situations differently managed or not managed in *Edicad* and adds new performances which would be lost when converting into *Edicad* (profile reductions, compensation change in profile).
Matrix generated with tool compensation applied cancels this difference: programs imported into *Edicad* are already compensated.

Logical workings

Logical workings have operating code between 1 and 1000.

Conversion refers to operating code as assigned in matrix.
For all custom logical workings, the following conversion rules apply:

***Tpaedi32* format**	***Edicad* format**
L field	Sets in Level field (value up to 8 only)
O field	Sets in Origin field (value up to 3 only)
M field	Sets in M field

Machine	Sets in corresponding field, only if value is different from 0
Group	Sets in corresponding field, only if value is different from 0
Tool	Sets in corresponding field, only if value is different from 0
Tool typology	Sets in corresponding field, only if value is different from 0
Tool Diameter	Sets in corresponding field, only if value is different from 0
Rotation speed	Sets in corresponding field, only if value is different from 0
Operating speed	Sets in corresponding field, only if value is different from 0
Entry slowing value	Sets in corresponding field, only if value is different from 0
Exit slowing value	Sets in corresponding field, only if value is different from 0
Custom parameters	Sets the same custom parameter, only if value is different from 0

Profile

Linear typology
Conversion refers to operating code L01 [2201].
The following conversion rules apply:

***Tpaedi32* format**	***Edicad* format**
X coordinate of point of application	Sets in Xf field, only if different from previous segment
Y coordinate of point of application	Sets in Yf field, only if different from previous segment
Z coordinate of point of application	Sets in Zf field, only if different from previous segment
Interpolation speed	Sets in corresponding field, only if value is different from 0
Custom parameters	Sets the same custom parameter, only if value is different from 0

Arc typology (xy plane)
Conversion refers to operating code A01 [2101].
The following conversion rules apply:

***Tpaedi32* format**	***Edicad* format**
X coordinate of point of application	Sets in Xf field, only if different from previous segment
Y coordinate of point of application	Sets in Yf field, only if different from previous segment
Z coordinate of point of application	Sets in Zf field, only if different from previous segment
Centre X coordinate	Sets in *Cx* field
Centre Y coordinate	Sets in *Cy* field
Rotation direction	Sets in rotation field (0=clockwise, 1=counter-clockwise)
Interpolation speed	Sets in corresponding field, only if value is different from 0
Custom parameters	Sets the same custom parameter, only if value is different from 0

Arc typology (xz plane)
Conversion refers to operating code A05 [2105].
The following conversion rules apply:

***Tpaedi32* format**	***Edicad* format**
X coordinate of point of application	Sets in Xf field, only if different from previous segment
Y coordinate of point of application	Sets in Yf field, only if different from previous segment
Z coordinate of point of application	Sets in Zf field, only if different from previous segment
Centre X coordinate	Sets in *Cx* field
Centre Z coordinate	Sets in *Cz* field
Rotation direction	Sets in rotation field (0=clockwise, 1=counter-clockwise)
Interpolation speed	Sets in corresponding field, only if value is different from 0
Custom parameters	Sets the same custom parameter, only if value is different from 0

In *Tpaedi32* configuration, a broken line with linear segments can be chosen for recording in piece matrix, instead of arc in xz plane. In this case, each linear segment of broken line is converted by operating code L01 [2201] and considerations referred to linear typology profile working apply.

Arc typology (yz plane)
Conversion refers to operating code A06 [2106].
The following conversion rules apply:

***Tpaedi32* format**	***Edicad* format**
X coordinate of point of application	Sets in Xf field, only if different from previous segment
Y coordinate of point of application	Sets in Yf field, only if different from previous segment
Z coordinate of point of application	Sets in Zf field, only if different from previous segment
Centre Y coordinate	Sets in *Cy* field
Centre Z coordinate	Sets in *Cz* field
Rotation direction	Sets in rotation field (0=clockwise, 1=counter-clockwise)
Interpolation speed	Sets in corresponding field, only if value is different from 0
Custom parameters	Sets the same custom parameter, only if value is different from 0

In *Tpaedi32* configuration, a broken line with linear segments can be chosen for recording in piece matrix, instead of arc in yz plane
In this case, each linear segment of broken line is converted by operating code L01 [2201] and considerations referred to linear typology profile working apply.

Arc typology (xyz plane)
An arc assigned on a generic plane (xyz) is converted into a piece matrix in a broken line with linear segments.
Criteria defining the broken line generation mode are assigned into *Tpaedi32* configuration.
Each linear segment of broken line is converted by operating code L01 [2201] and considerations referred to linear typology profile working apply.

Tecnologie e Prodotti per l'Automazione s.p.a.
Via Carducci 221
I - 20099 Sesto S.Giovanni (MI)
Tel. +39 02.365.27.550
Fax. +39 02.24.81.008
www.tpaspa.it