Quick User Manual for
«CNC Lathe Simulator ver. 3.0»

https://sunspire.site/

1. General Description of the Software Product

The CNC lathe simulator is a multimedia application designed to provide beginners in the mechanical engineering field with a basic introduction to the principles of programming turning operations using standard (ISO) G-code.

The main task of the application is syntactic analysis (parsing) of the control program code in order to construct a graphical model of cutting tool trajectories in three-dimensional space.

Main functions of the application:

– editing the code of control programs of a lathe;

– operations with control program files;

– setting up geometric parameters of the cutting tool;

– continuous/step-by-step execution of control program blocks;

– three-dimensional visualization of tool movements in the machine's working space;

– simplified visualization of the workpiece surface;

– calculation of processing modes;

– a brief reference guide on using G-code.

The main limitations of the application are: simplified model of the cutting surface, support for only two controlled axes, simplified model of machine tool components.

Target computing device type and supported platform: IBM-compatible personal computer running Microsoft Windows, Apple Macintosh personal computer running MacOS, mobile devices running Android and iOS operating systems.

The graphical system of the software uses the OpenGL 2.0 component base. The program's graphical user interface is implemented in Russian, English and Chinese.

Multi-platform support allows the software product to be used on various computing devices, including interactive whiteboards, smartphones, tablets and desktop computers, which in turn increases the flexibility and mobility of the educational process, corresponding to the modern level of informatization of education.

Minimum system requirements:

– central processor: Intel/AMD, not less than 2,4 GHz;

– RAM: not less than 1 GB;

– video memory: not less than 512 MB;

– screen resolution: not less than 1024x768x32;

– OpenGL 2.0 (and Vulkan on mobile platforms);

– DirectX version 9.0.c (for Microsoft Windows OS);

– standard keyboard and computer mouse with scroll wheel (touch screen for mobile platforms);

– audio playback devices (audio speakers or headphones).

The software has a special version that supports hardware interaction with the physical CNC controller GSK 980 TDi. This version of the program is supplied with the hardware and software training complex. The interaction of the software and hardware components of the training complex is carried out in accordance with the ModBus application level network protocol. Direct connection of the CNC controller to the personal computer is carried out via the RS-485 interface.

2. Description of the Graphical User Interface

The main screen of the application is an interactive 3D scene with an image of a geometric model of a CNC lathe placed in a conditional surrounding space. The model is monitored using a virtual camera, the angle of which can be changed during operation.

Along with three-dimensional graphic elements, the main screen displays two-dimensional control elements – buttons, switches and basic text information about the simulated processes (Fig. 1).

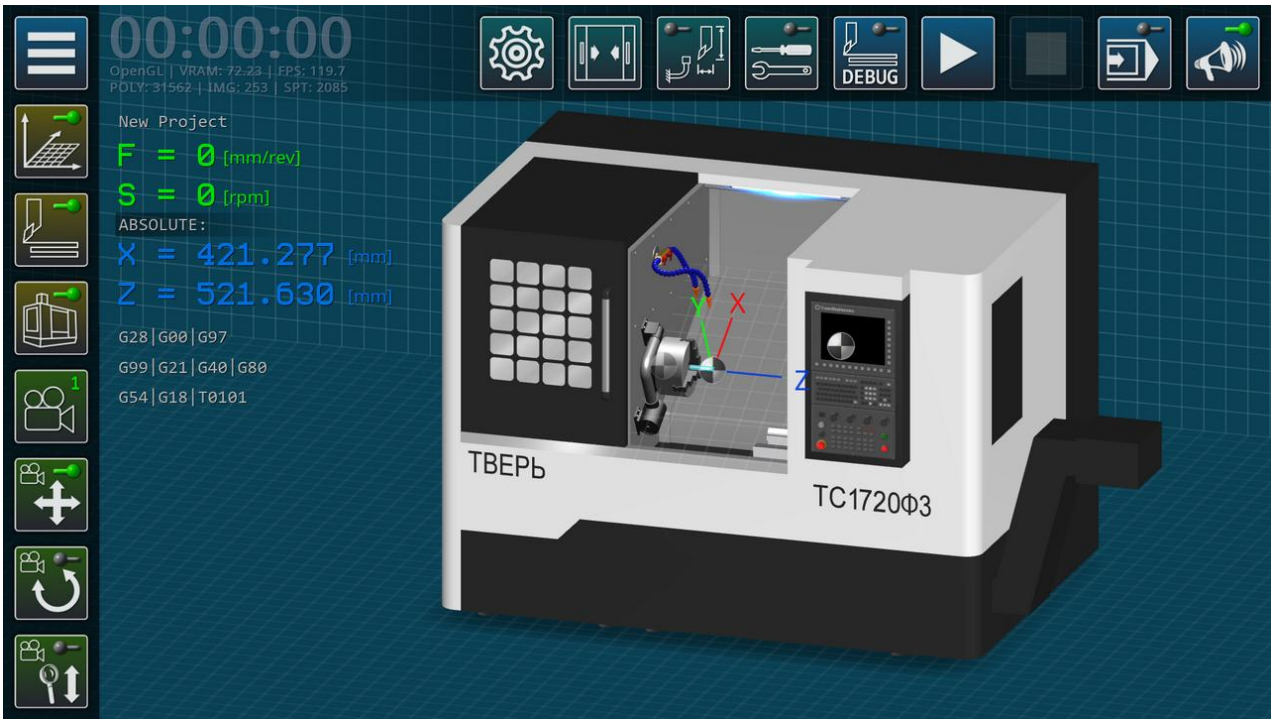

Figure 1 – Main screen of the application

On the left side of the main screen there is a vertical panel of the main function buttons (Table 1).

Table 1 – Main functional buttons of the application screen

Button Image	Function Name	Function Description
	Main Menu	Displaying the main menu of the application
	Coordinate System	Show/hide the coordinate system and coordinate grid of the machine workspace
	Tool Paths	Show/hide the graphical model of the cutting tool paths
	3D Model of the Machine	Show/hide complex graphical model of lathe machine elements
	Current Camera	Switching virtual camera mode
	Camera Move	Virtual camera planar moving mode (for touchscreen devices)
	Camera Rotate	Virtual camera rotation mode (for touchscreen devices)
	Camera Zoom	Virtual camera zoom in/out mode (for touchscreen devices)

At the top of the main screen there is a horizontal panel of buttons for controlling the simulation process and settings (Table 2).

Table 2 – Simulation process control buttons and settings

Button Image	Function Name	Function Description
	Application Settings	Displays the application settings panel for selecting localization and simulation parameters
	Automatic Door	Changing the state of the automatic door of the lathe (opening/closing)
	Tool Binding Mode	Tool offsets detection mode using a measuring sensor
	Additional Panel	Displaying an additional panel of workpiece control buttons
	Debug Mode	Fast debugging of the control program and construction of a complex model of tool paths
	Simulation Play	Starting the machine control program in continuous frame execution mode
	Simulation Pause	Pausing the machine control program
	Simulation Stop	Termination of execution of the machine control program

Table 2 – Continued

Button Image	Function Name	Function Description
	Step-by-Step Execution	Turning on/off the step-by-step execution mode of the machine control program
	Playing Sounds	Turn on/off the sound accompaniment of the simulation process

Additional functional buttons are provided for controlling the workpiece (Table 3).

Table 3 – Additional panel buttons for workpiece control

Button Image	Function Name	Function Description
	Cut Off Fragment	Show/hide a fragment of the workpiece resulting from the cutting off operation
	Restore the Workpiece	Returning the workpiece contour to its original state
	Flip Workpiece	Mirror flipping of the workpiece contour along the main axis
	Move the Workpiece to the Right	Shifting the workpiece contour to the right

Table 3 – Continued

Button Image	Function Name	Function Description
	Move the Workpiece to the Left	Shifting the workpiece contour to the left
	Coordinate System Offset	Shifting the active coordinate system to the machine's zero points
	Reset Coordinate System	Return of the original specified workpiece correctors

When you click the Main Menu button, a radial menu for selecting the application's auxiliary dialog screens is displayed on the screen (Fig. 2).

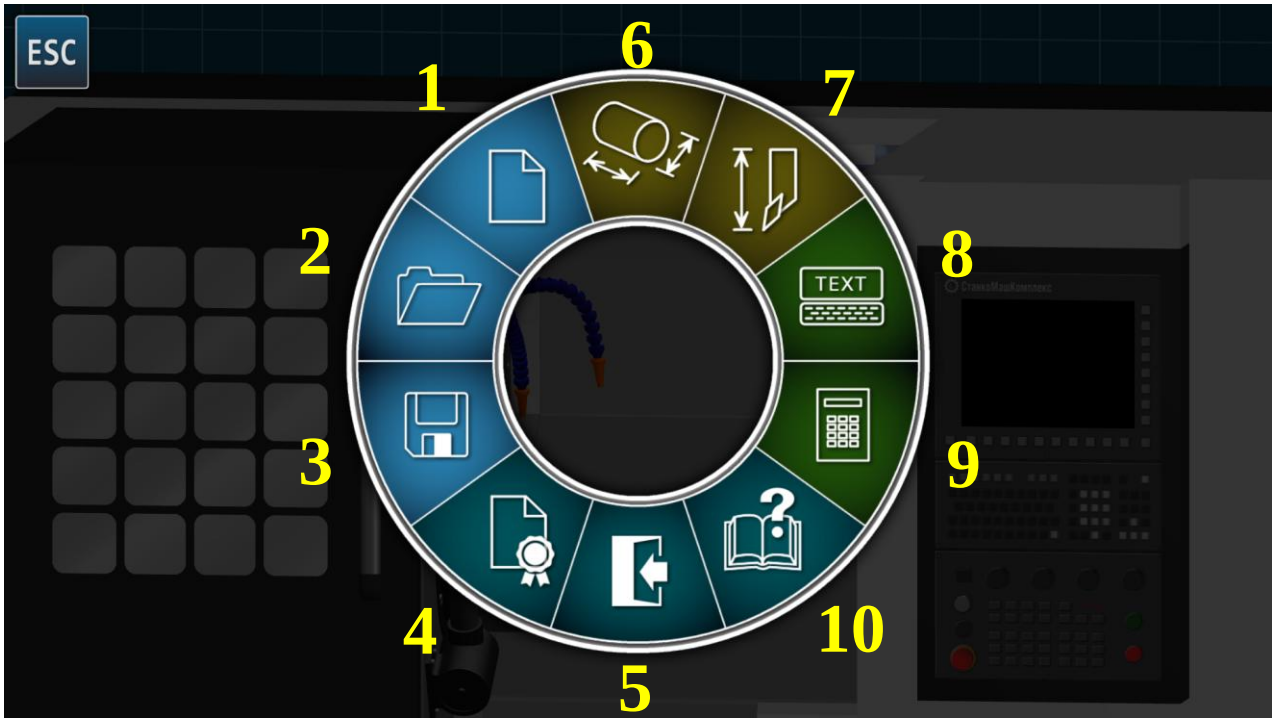

Figure 2 – Main menu of the application

Button 1 – «Create Project» – is designed to reset the current simulation parameters. This function is used if after working with an open file it is necessary to start a new project of the control program. When you click the «Create Project» button, a modal dialog box for confirming the reset of the current simulation parameters is displayed on the screen.

Button 2 – «Open Project» – is designed to select and open the machine control program file. When you click the «Open Project» button, the file opening screen is displayed (Fig. 3), in the main area of which a list of directories and a list of control program files written to the working directory of projects are displayed.

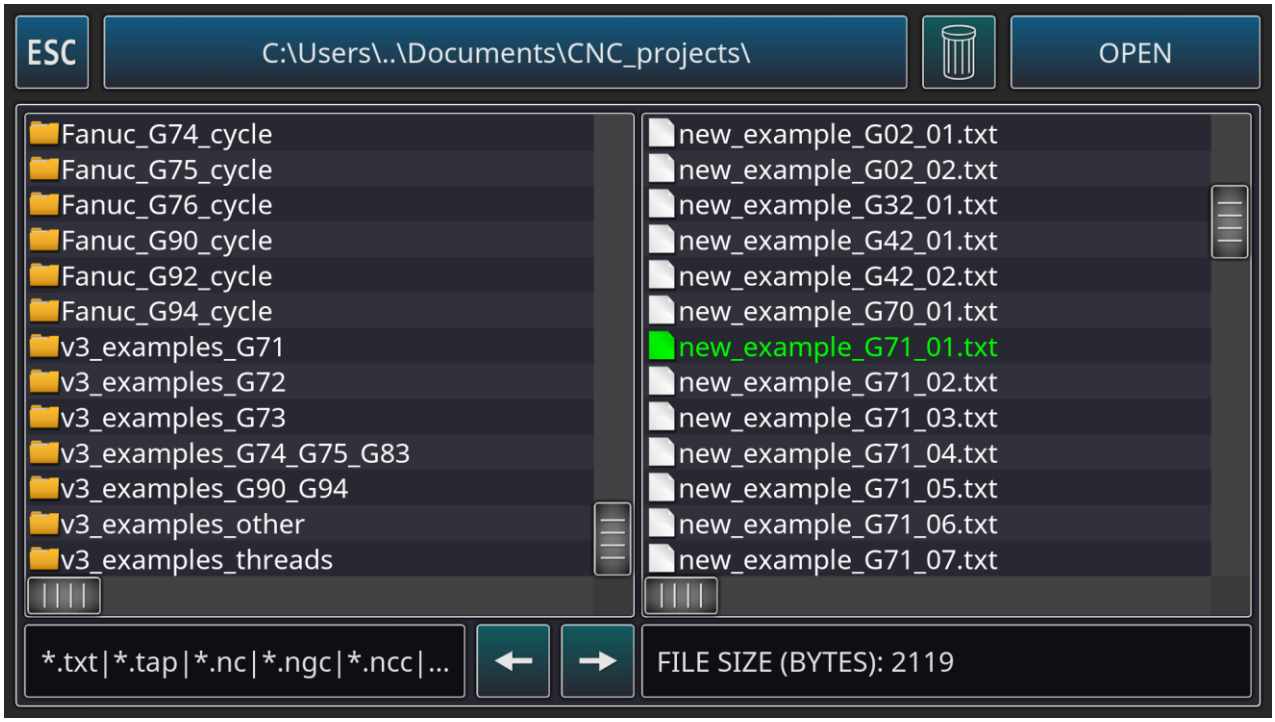

Figure 3 – Open file dialog screen

At the bottom of the file opening screen there is a file type filter, which allows you to display files of a given format in the list. The simulator supports 6 control program file formats: *.txt, *.tap, *.nc, *.ngc, *.ncc, *.csdata. The size of the selected file in bytes is displayed in the lower right part of the screen. The selected

file in the list is highlighted in green. If the file is selected, the "Open" button is activated in the upper right part of the screen.

There is also a «Delete file» button in the upper right part of the screen, which becomes active when a file is selected. In the mobile version of the application, an additional «Send» button is displayed at the top of the file opening screen, allowing you to send the selected file using an external application (for example, an email client).

Button 3 – «Save Project» – is designed to select the local path and name of the saved file of the machine control program. When you click the «Save Project» button, the file saving screen is displayed (Fig. 4), the principle of interaction with which is similar to the file opening screen. The name of the file to be saved is entered in a special text field located in the lower right part of the screen, without specifying the extension. The file extension is selected using the switch in the lower left part of the screen.

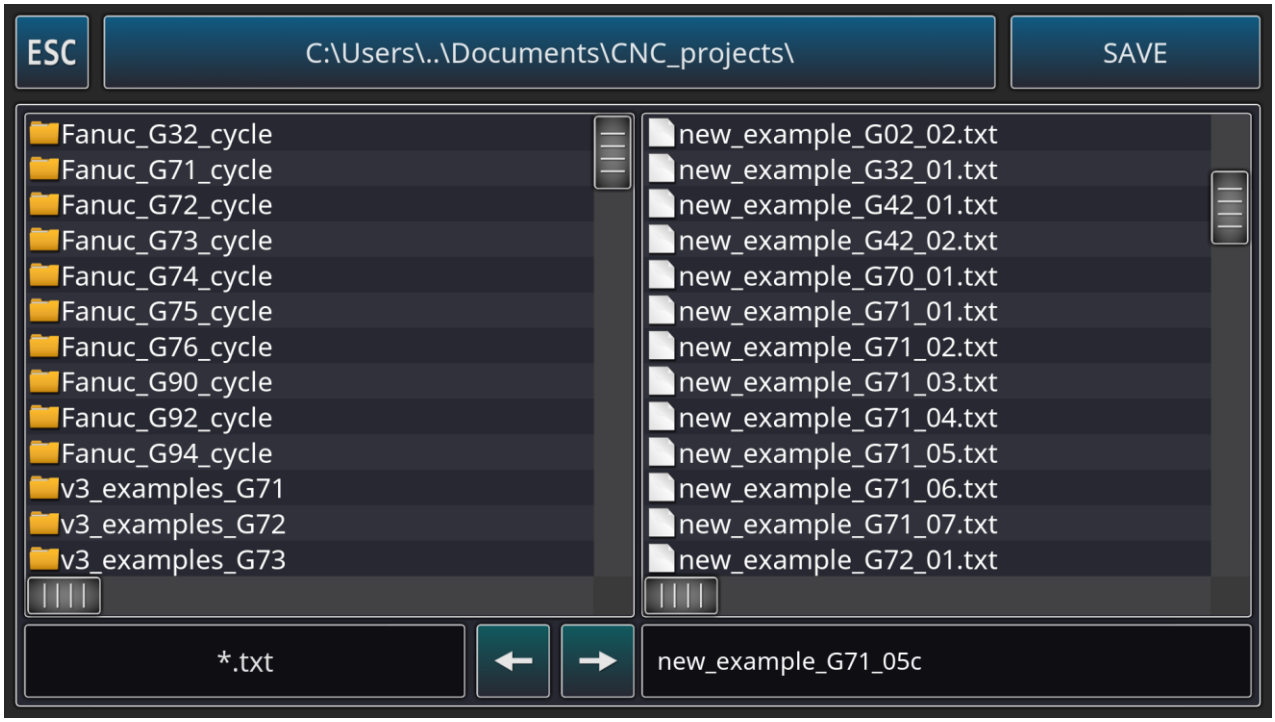

Figure 4 – File save dialog screen

The control program files and subdirectories of standard control program examples are located in the directory «**C:\Users\...\Documents\CNC_projects**» in the Microsoft Windows operating system. In the Android system, the default path to the control program files is «**/storage/emulated/0/CNC_projects**» or «**/storage/emulated/0/Android/data/com.virtlab.cncsim/files/CNC_projects**» (for later versions of Android OS). In iOS and MacOS systems, the control program files are stored in the application's working directory.

Button 4 – «License Information» – is designed to display a screen with the output of basic licensing information about the licensor and licensee of the software product.

Button 5 – «Shutdown App» – is designed to display the simulator shutdown dialog screen.

Button 6 – «Workpiece Parameters» – is designed to display the screen for setting the parameters of the workpiece being processed (Fig. 5).

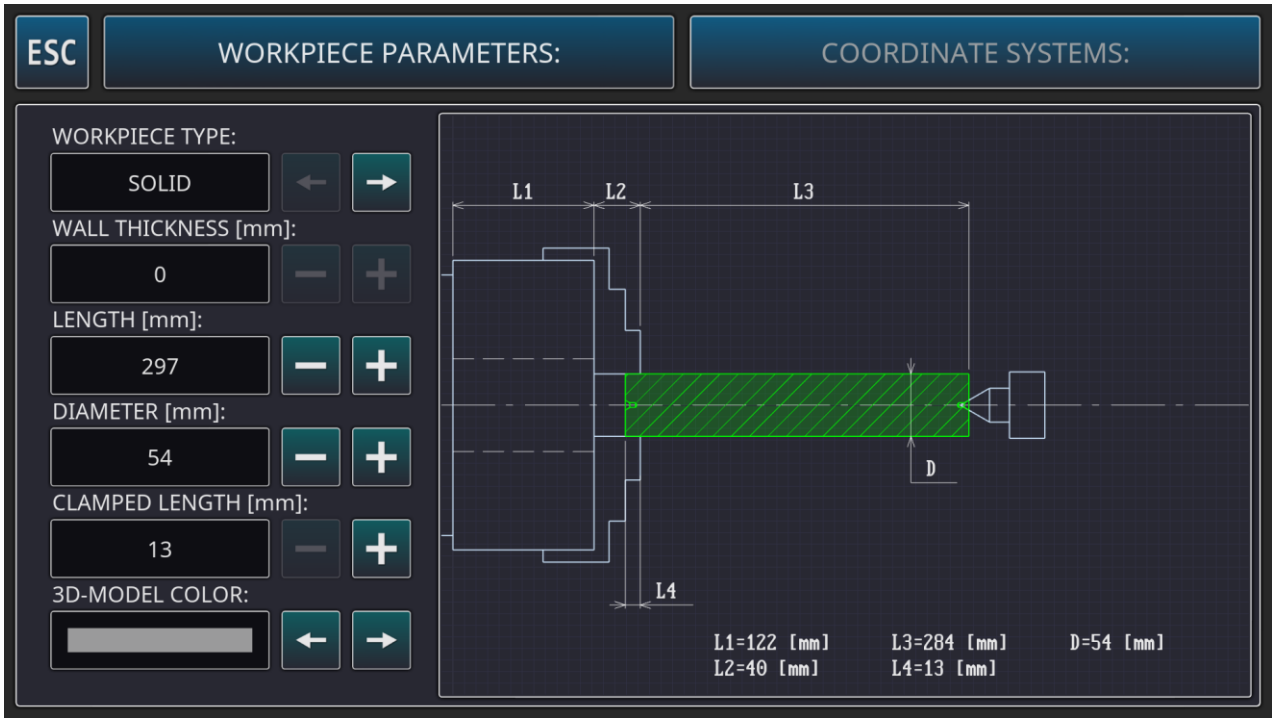

Figure 5 – Workpiece parameters setup screen

On the left side of the workpiece parameters screen there is a block of numeric and string fields with buttons for increasing/decreasing values. Using these parameters, you can set the geometric dimensions of the workpiece being processed and select the color of the modeled surface. On the right side of the screen, a drawing of the workpiece with the main dimensions and visualization of the method of fixing the workpiece is displayed. The drawing of the workpiece is updated adaptively to changes in the parameter values.

The top part of the screen contains the header buttons, which are used to switch the workpiece setup mode. By default, when the workpiece parameters dialog screen is opened, the main geometric parameters of the workpiece being processed are set up. The «Coordinate Systems» button allows you to set up the relative position of the workpiece coordinate systems (Fig. 6).

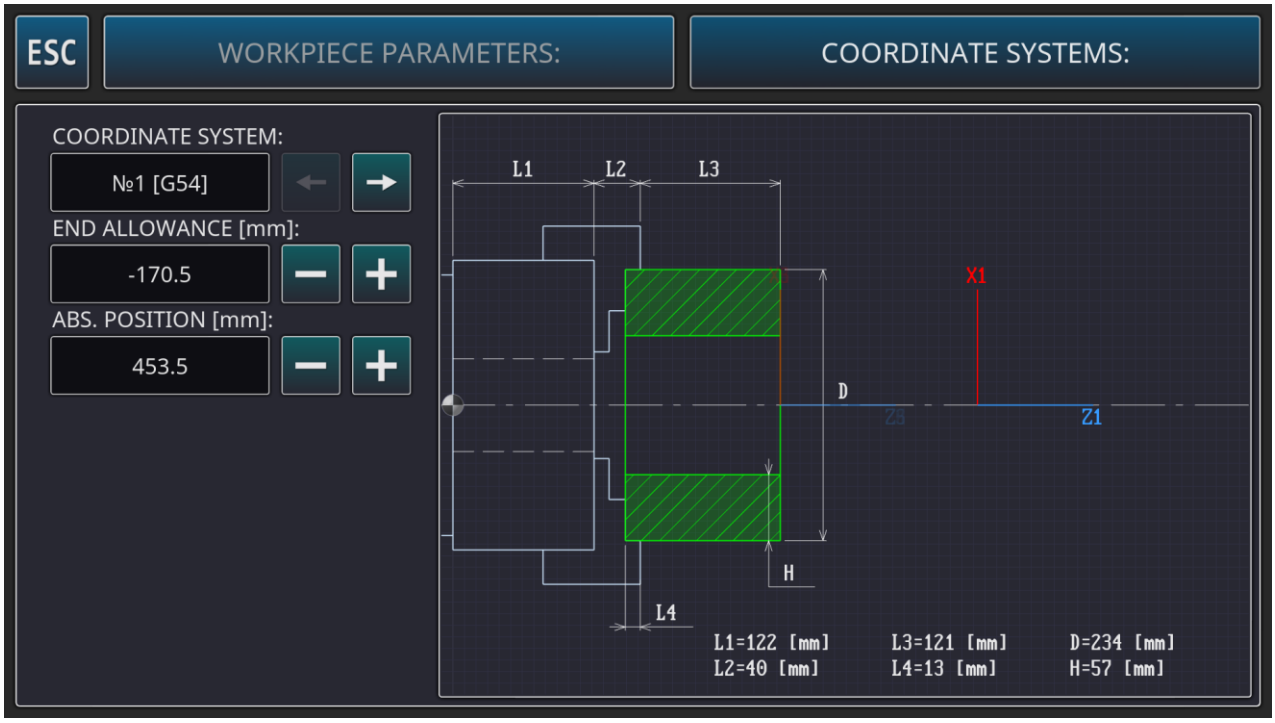

Figure 6 – Coordinate systems setup screen

Six workpiece coordinate systems are available for preliminary setup. Further switching between workpiece coordinate systems during the simulation is performed programmatically by using special control codes G54-G59. The coordinates of the

workpiece zeros can be specified in absolute values (relative to the machine zero point) or by entering the value of the end allowance relative to the right end of the workpiece.

Information about the workpiece parameters is saved at the beginning of the control program file in the form of comments.

Button 7 – «Tool parameters» – is designed to display the screen for setting the parameters of the cutting tool (Fig. 7).

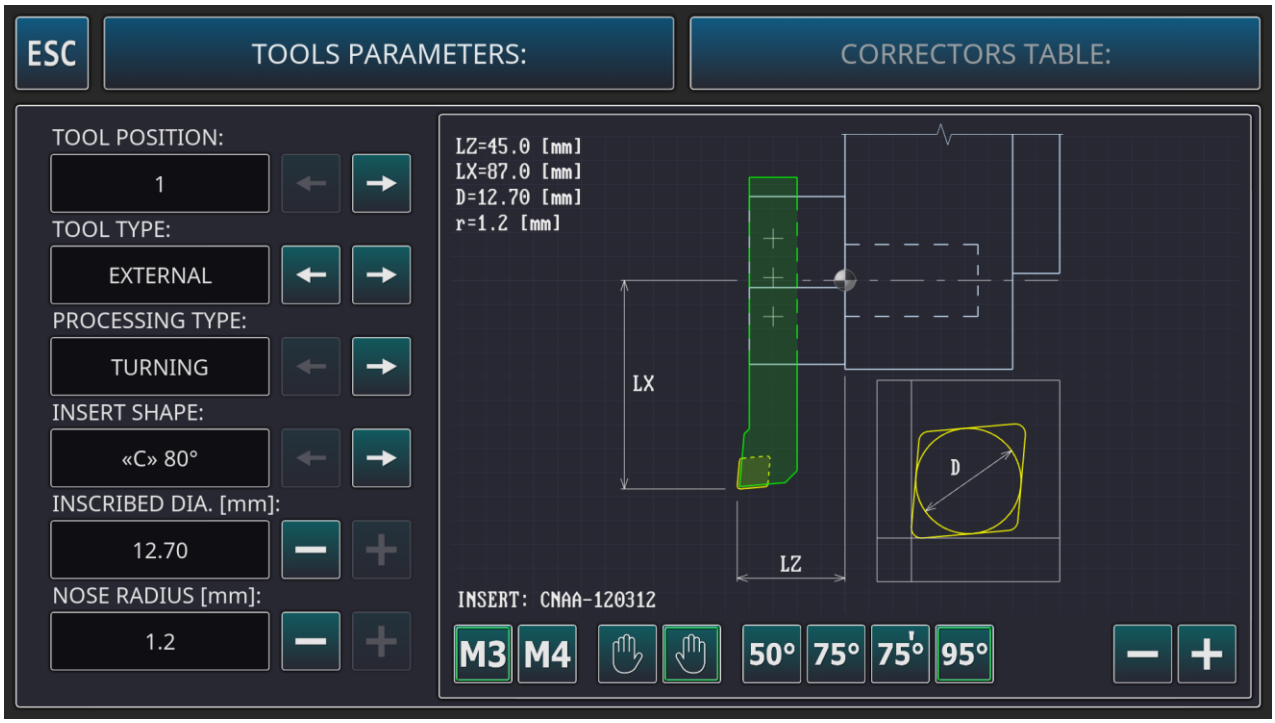

Figure 7 – Cutting tool parameters screen

On the left side of the tool parameters screen there is a block of numerical and string fields with buttons for increasing/decreasing values, as well as switching positions. Using these parameters, you can set the type of cutting tool used in the simulation, the design, and the geometric dimensions of the working part of the cutting tool.

The right side of the screen displays a simplified drawing of the selected tool and its technological equipment with the output of the main dimensions. The drawing is updated in real time when the tool parameters change.

The top of the screen contains the header buttons similar to the workpiece parameters screen. The «Correctors Table» button displays a summary table of the geometric dimensions of all the tools used with the corresponding corrector numbers used by the «T0_0_» tool switching program function.

Button 8 – «Control Program Editor» – is designed to display the text editor of the control program code on the main screen of the application (Fig. 8).

The main part of the text editor is represented by a list of control program lines available for editing from the physical keyboard. Editing of the machine control program text is performed only in the stopped simulation mode. When the simulation process is running, the control program text editing mode is not available. In this case, the control program text scrolls automatically, and the currently executed program block is highlighted in green.

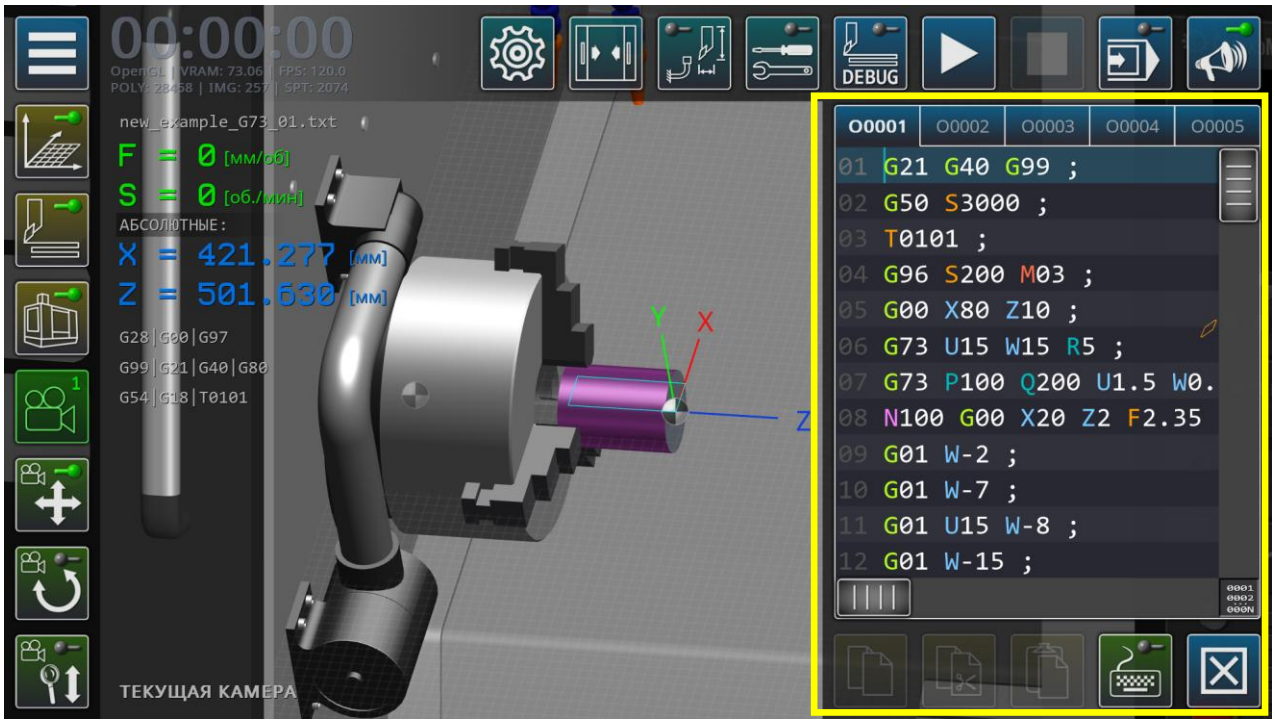

Figure 8 – Control program editor in standard (reduced) display mode

The upper part of the text editor contains tabs intended for switching control programs of the current project (5 in total). Using additional tabs, you can include

external subroutines in the control program, the numbering of which corresponds to the tab numbers.

At the bottom of the text editor are the main functional buttons: «Copy», «Cut», «Paste», «Virtual Keyboard» and «Close Editor» (Fig. 9).

Figure 9 – Basic functional buttons of the text editor

The Virtual Keyboard toggle button allows you to switch the editor to full-screen mode (Fig. 10). In full-screen mode, the text editor is displayed on the left side of the screen, and a virtual alphanumeric keyboard is displayed on the right side of the screen, which allows you to edit text on devices equipped with a touch screen without a physical keyboard (for example, on tablet computers or smartphones).

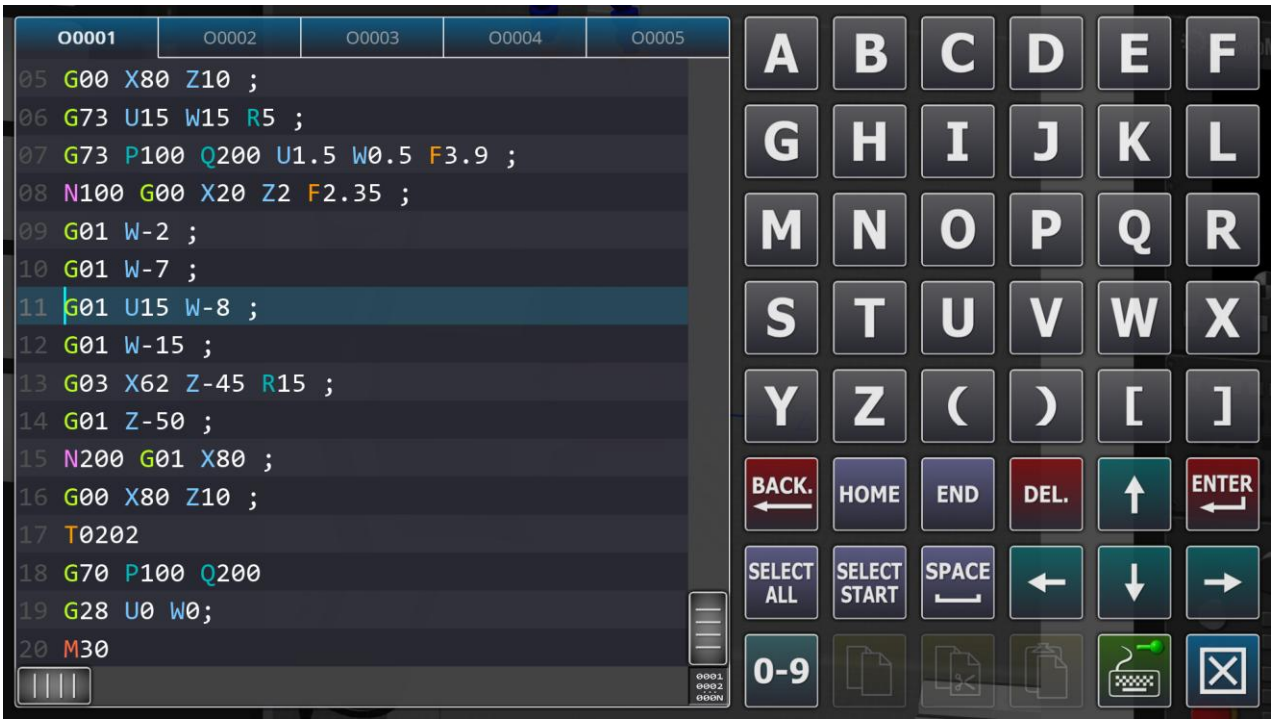

Figure 10 – Control program editor in full-screen display mode

In the editing mode, the text of the control program code is color-coded by groups of letter codes and addresses. Letter codes and addresses of one group (for example, technological parameters T/S/F or coordinates X/Z/U/W/I/K) are designated by one color. In the control program execution mode, color coding of codes and addresses is not applied.

Button 9 – «Processing Modes Calculator» – is designed to display the screen of the turning modes calculator (Fig. 11).

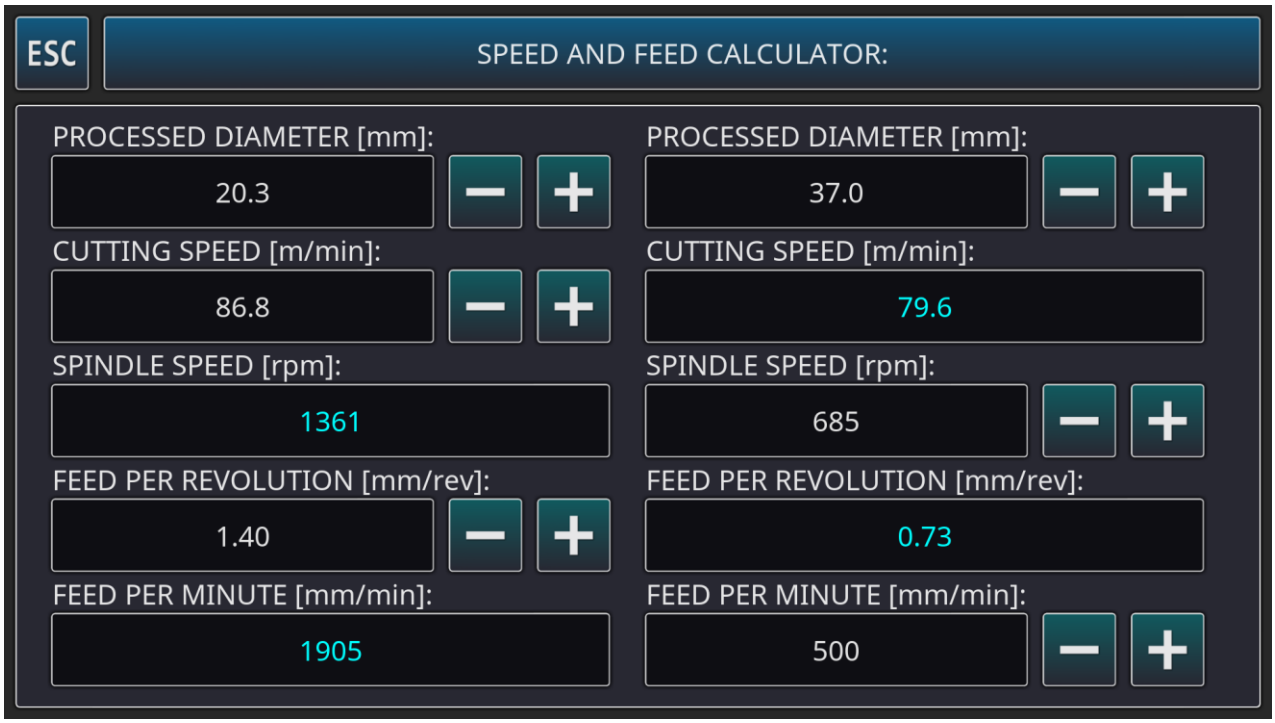

Figure 11 – Processing mode calculator screen

The calculator screen can be divided into two parts. The left part calculates the spindle speed and the feed per minute for the specified cutting speed. The right part calculates the cutting speed and the feed per revolution for the specified spindle speed. The numerical fields of the parameters, which are the initial data for the calculation, have buttons for increasing and decreasing the value. The numerical fields that do not have increment buttons are intended for displaying the calculated values.

Button 10 – «Reference Guide» – is designed to display the screen of the built-in help system of the application (Fig. 12). The reference guide contains basic information about the application and the numerical control tools used. The reference information is structured into 7 sections: «About the Application», «Model Description», «List of Commands», «G-codes Description», «M-codes Description», «Fixed Cycles», «Macro-B Operators».

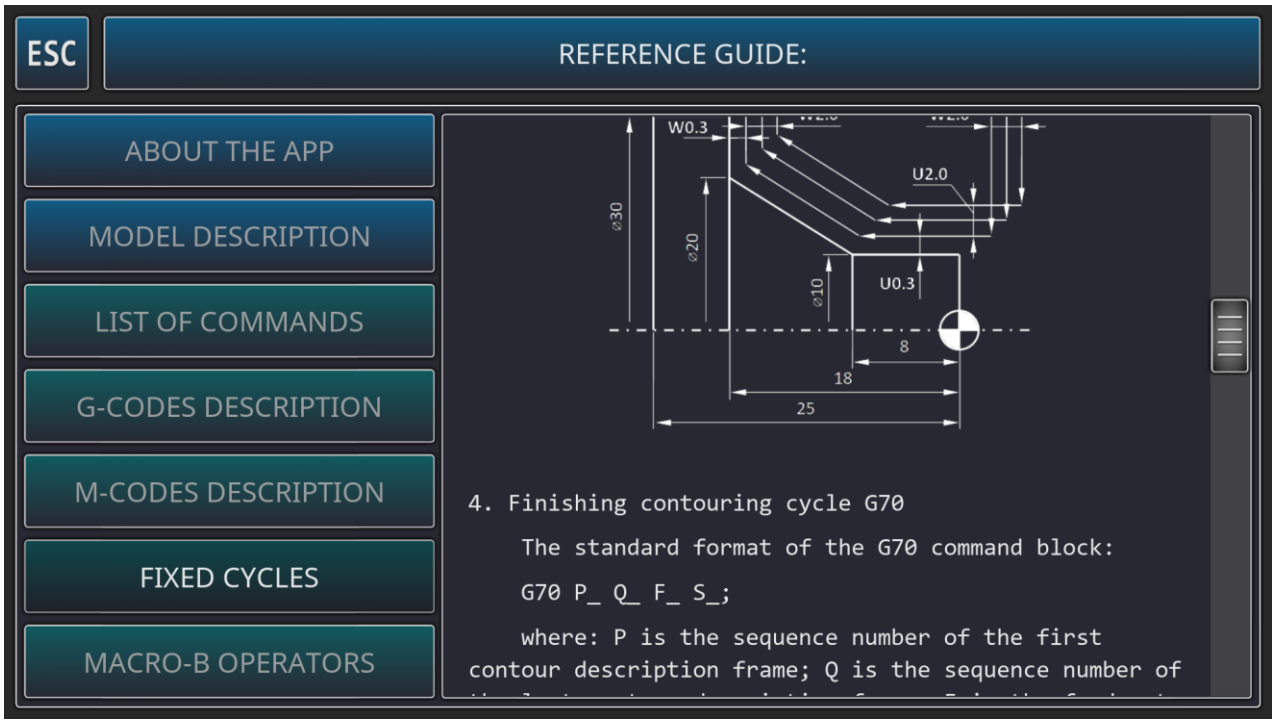

Figure 12 – Built-in help system screen

3. The Principle of Controlling a Virtual Camera Using a Mouse

Basic manipulations with the camera in mode №1 can be performed using a computer mouse. Pressing and holding the left mouse button with the accompanying movement of the mouse leads to the movement of the camera's focus point in the frontal plane of space. Pressing and holding the right mouse button with the accompanying movement of the mouse leads to rotation of the camera relative to the focus point. The rotation angles (azimuth and elevation) of the camera are limited by

the dimensions of the model space. The camera's distance is changed by rotating the scroll wheel in the forward and reverse directions (Fig. 13).

On devices with a touch screen, the camera is controlled using gestures (Touch Screen).

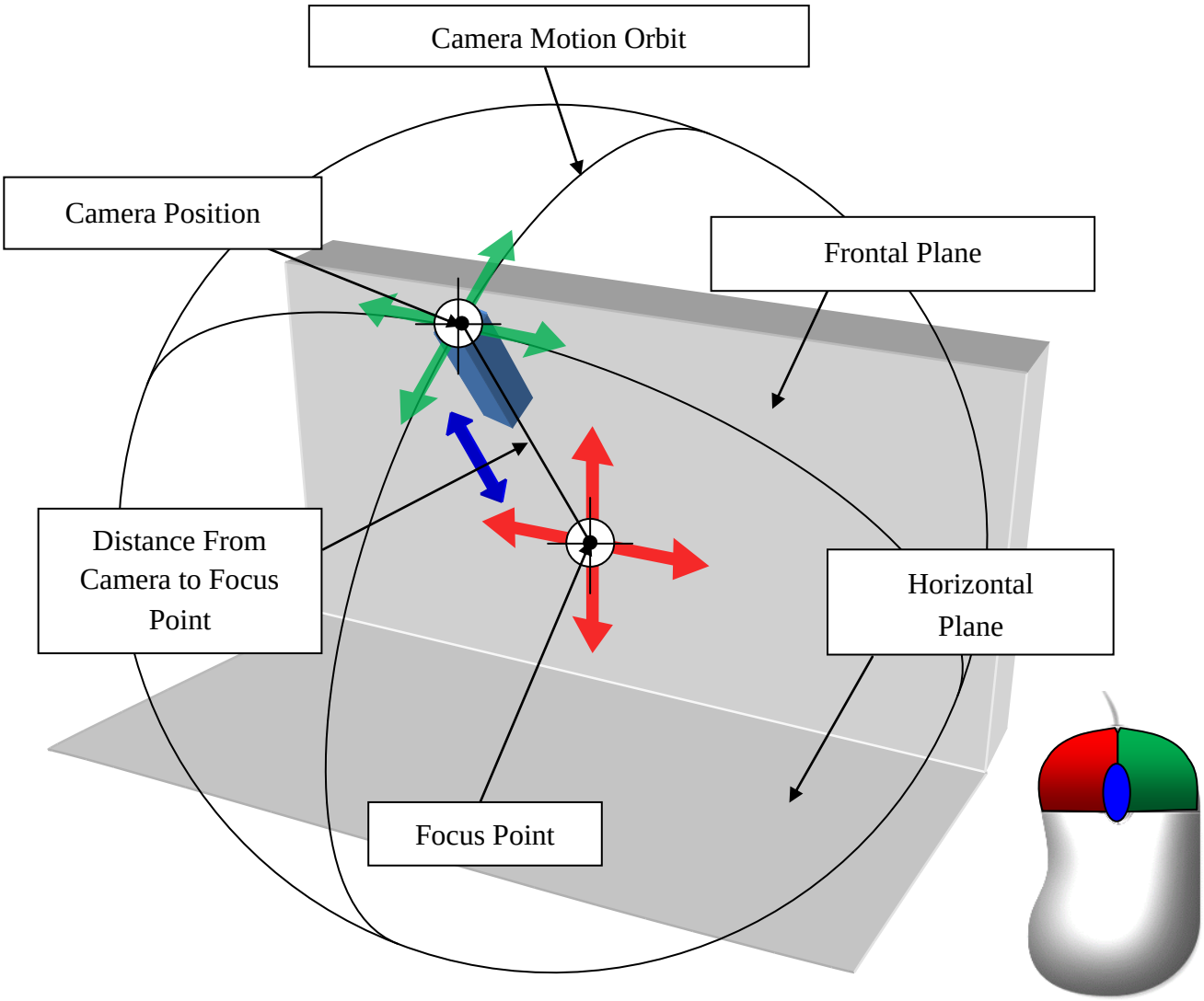

Figure 13 – Scheme of control of virtual camera using computer mouse

4. The Mode of Constructing a Model of Cutting Tool Trajectories or the Control Program Debug Mode

In debug mode, the syntactic analysis (parsing) of the control program is performed at an accelerated rate. In this case, in the application settings, you can select the appropriate simulation mode – «full» (with modeling of the shape of the workpiece) or «only trajectories» (with construction of trajectories without actual tool movements). In debug mode, the three-dimensional model of the machine is not displayed (Fig. 14).

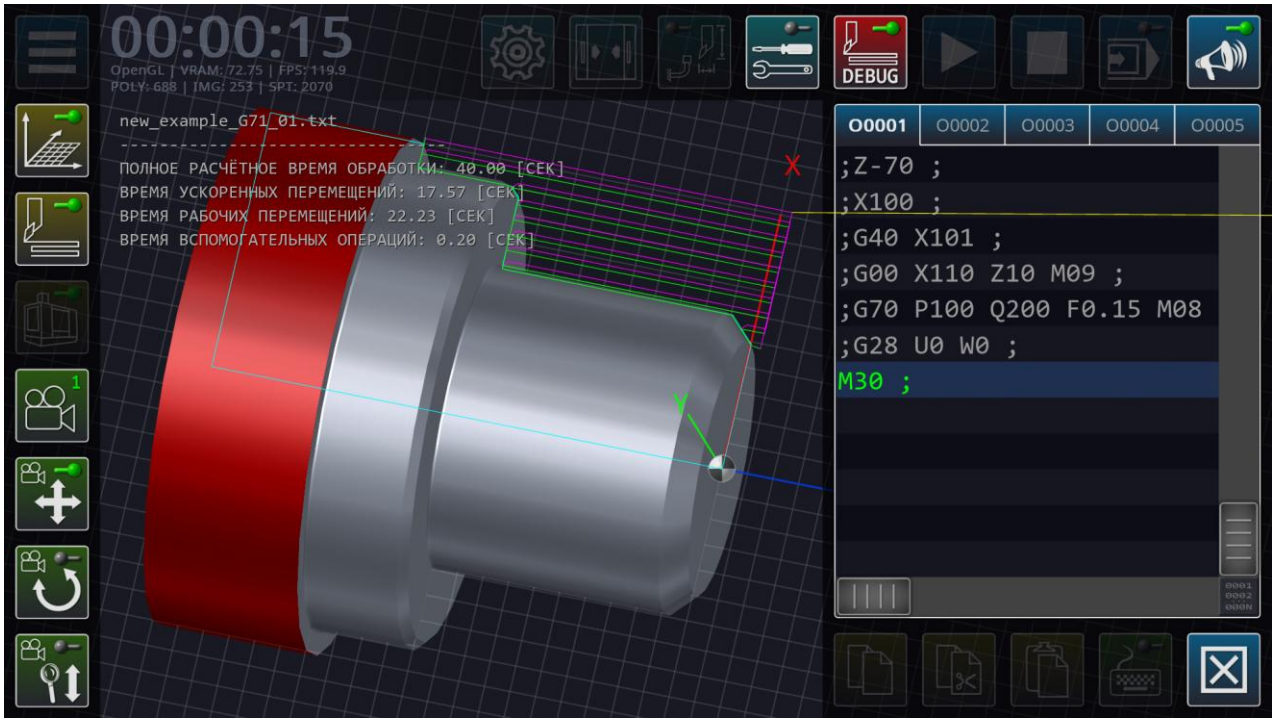

Figure 14 – View of the main screen of the simulator in the debugging mode of the control program

The left side of the main screen displays the calculated time spent on performing the main and auxiliary technological operations in minutes and seconds.

If the model of the workpiece is built, the geometric dimensions of the fragments of the longitudinal section contour of the workpiece can be measured

using the measurement function. The measurement is performed along two coordinate axes. To measure the selected section, use the left mouse button to select two control points lying on the contour of the workpiece (Fig. 15). Measurements can be performed along two arbitrary points related to different sections of the workpiece contour.

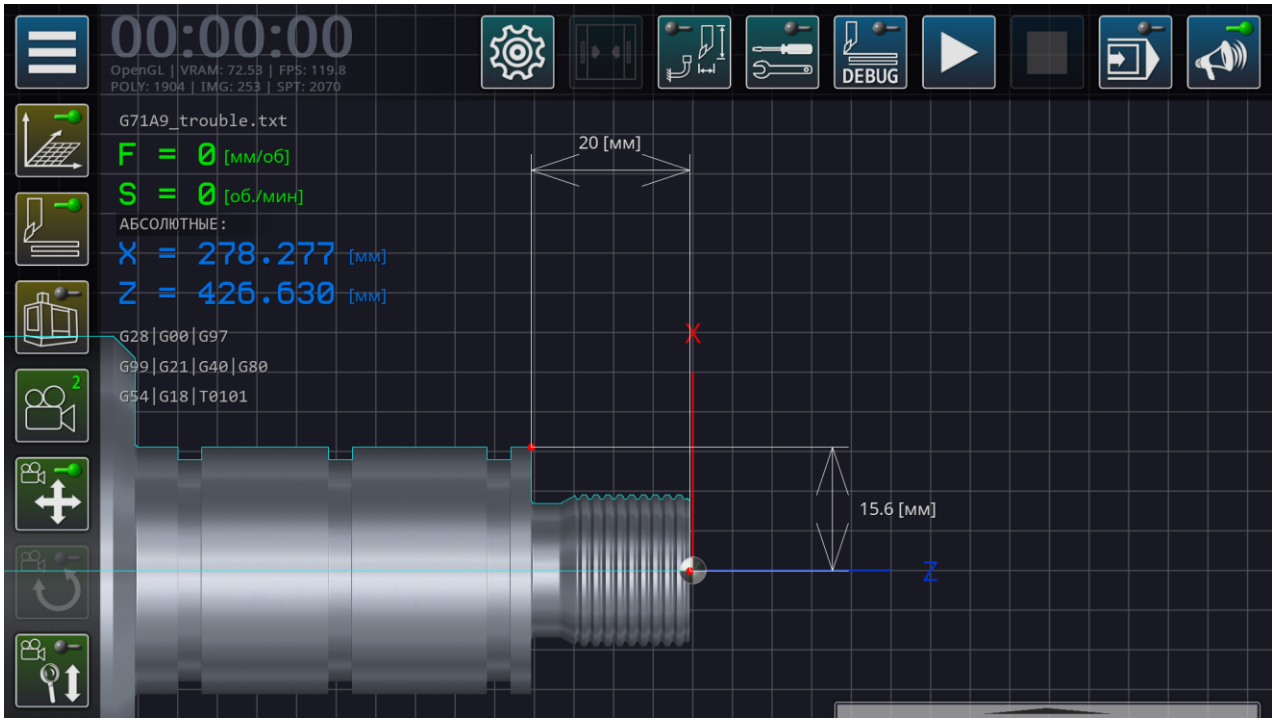

Figure 15 – Measuring contour sections of a machined part

5. Tool Binding Mode Using a Measuring Sensor

In the tool binding mode, the workpiece model is hidden, and the tool binding sensor is lowered to the working position. The main screen displays the basic geometric information about the sensor – its coordinates in the machine's reference coordinate system, as well as the size of the contact plate. The tool binding process control panel is displayed on the right side of the screen. The table of tool correctors is displayed in the lower left part of the screen (Fig. 16). The table of tool correctors duplicates the table of correctors from the tool setup section. Using the control panel buttons on the right side of the screen, you can: reset the values of all tool correctors,

select the currently measured tool, set the spatial position of the tool using the flywheels simulating the remote control of the manual pulse generator (MPG), and enter the measured value into the table.

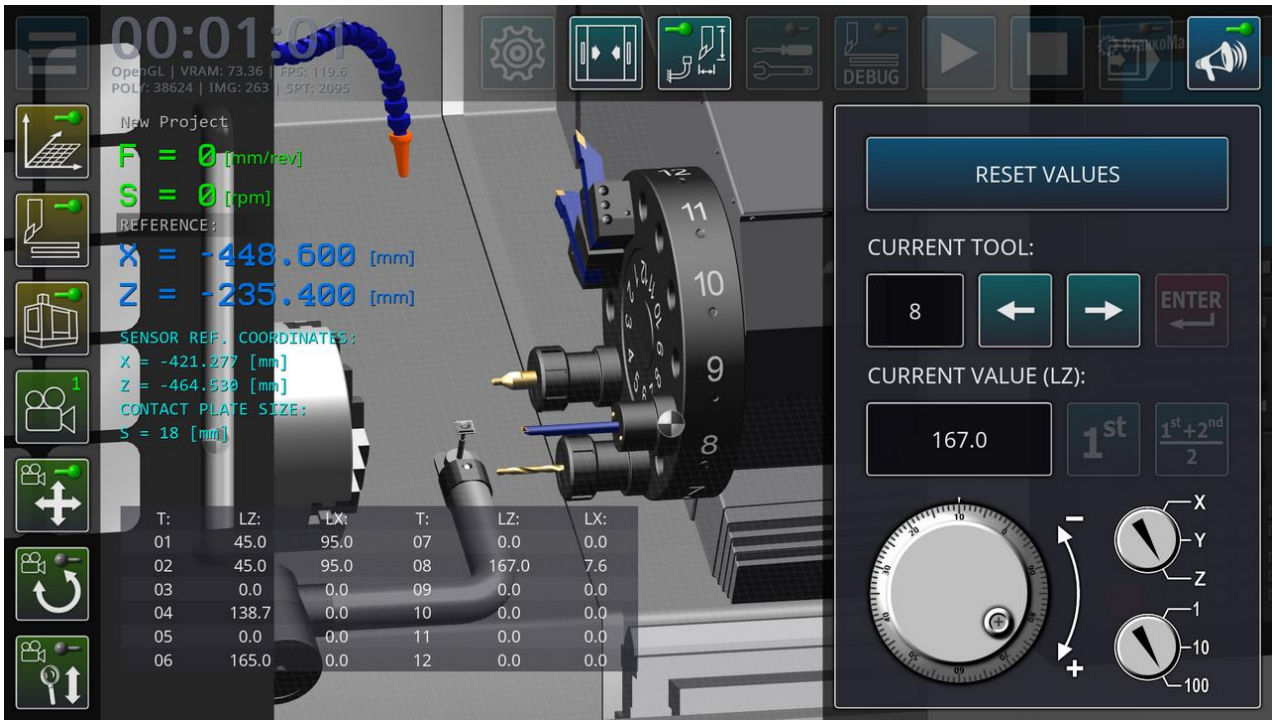

Figure 16 – View of the main screen of the simulator in the workpiece binding mode

Similar to the real MPG remote control, the control panel on the right side of the screen allows you to select the current machine axis, manual feed speed, and also feed the tool by rotating the corresponding flywheel.

If the measured tool does not touch the sensor surface, the value input buttons are inactive (Fig. 17.a). When the tool touches the sensor, the measured value input buttons become active (Fig. 17.b). The «1^{st}» button is used to enter the current measured value into the table, and the «$(1^{st}+2^{nd})/2$» button is used to enter the average between the current and previous values, which allows for tying cutters with axial symmetry when the sensor is touched not by the cutting plate, but by the left and right sides of the holder (Fig. 18).

Figure 17 – View of the control panel for the tool binding process in positioning mode (a) and in the measured value recording mode (b)

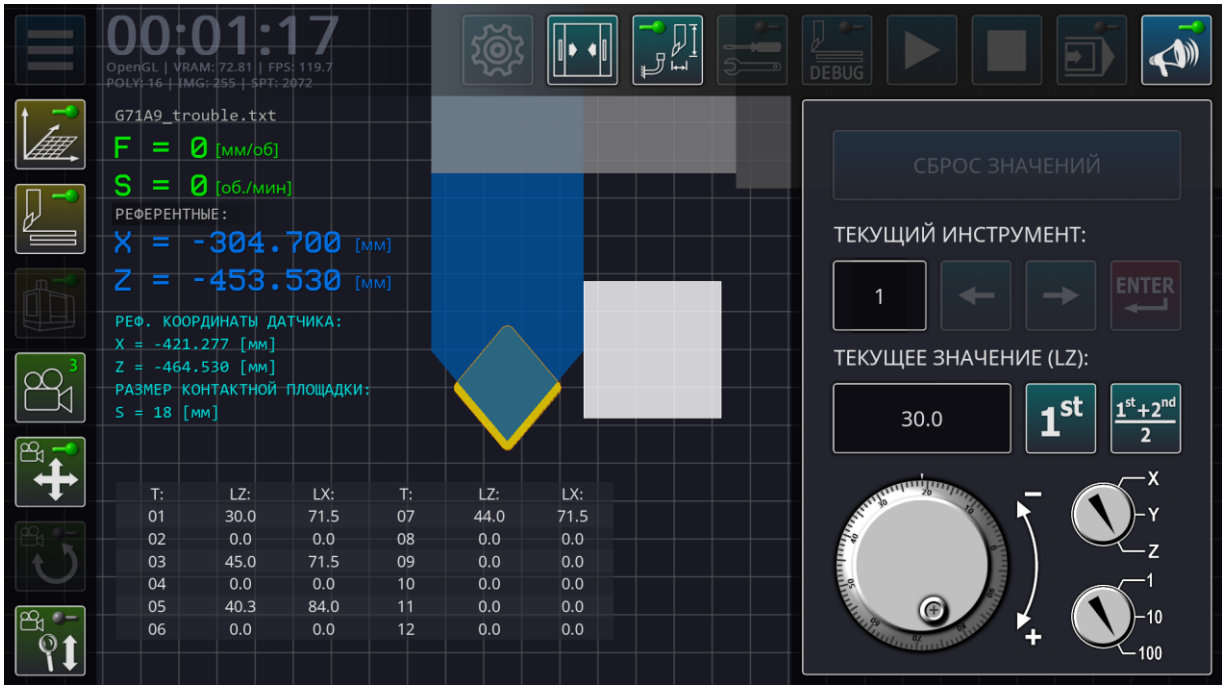

Figure 18 – Binding of symmetrical cutter

6. Brief Reference Information on Programming Turning Operations in the Simulator

This section duplicates the built-in help system of the simulator program. The following describes the basic principles of programming turning using standard (ISO) G-code. The rules for processing and executing commands in the simulation model are as close as possible to real control systems based on GSK, Fanuc, Fagor, etc. in the ISO code specification, but may differ from real systems.

The simulation model supports 3 code systems (A, B, C), including various control codes and letter addresses (Table 4).

Table 4 – Supported G-codes

Code System			Group	Function Description
A	B	C		
G04	G04	G04	00	Time delay
G28	G28	G28	00	Return to reference position
G50	G92	G92	00	Offset of the coordinate system
G52	G52	G52	00	Enter local coordinate system
G53	G53	G53	00	Machine coordinate system
G65	G65	G65	00	Call macro program
G70	G70	G72	00	Finishing cycle
G71	G71	G73	00	Longitudinal roughing cycle
G72	G72	G74	00	Cross roughing cycle
G73	G73	G75	00	Rough contour turning cycle
G74	G74	G76	00	Face groove turning cycle
G75	G75	G77	00	Side groove turning cycle
G76	G76	G78	00	Threading cycle
G00	G00	G00	01	Accelerated positioning
G01	G01	G01	01	Linear interpolation
G02	G02	G02	01	CW Circular Interpolation
G03	G03	G03	01	CCW Circular Interpolation
G32	G33	G33	01	Threading with constant pitch
G34	G34	G34	01	Variable pitch threading
G90	G77	G20	01	Longitudinal turning cycle

Table 4 – Continued

Code System			Group	Function Description
A	B	C		
G92	G78	G21	01	Threading cycle
G94	G79	G24	01	Cross turning cycle
G96	G96	G96	02	Constant cutting speed
G97	G97	G97	02	Cancel constant cutting speed
---	G90	G90	03	Absolute positioning
---	G91	G91	03	Relative positioning
G98	G94	G94	05	Feed per minute
G99	G95	G95	05	Feed per revolution
G20	G20	G70	06	Input in inches
G21	G21	G71	06	Input in mm
G40	G40	G40	07	Cancel radius correction
G41	G41	G41	07	Radius correction to the left
G42	G42	G42	07	Radius correction to the right
G80	G80	G80	10	Cancel drilling cycle
G83	G83	G83	10	End face drilling cycle
---	G98	G98	11	Return to original plane
---	G99	G99	11	Return to the R plane
G54	G54	G54	14	Set coordinate system №1
G55	G55	G55	14	Set coordinate system №2
G56	G56	G56	14	Set coordinate system №3
G57	G57	G57	14	Set coordinate system №4
G58	G58	G58	14	Set coordinate system №5
G59	G59	G59	14	Set coordinate system №6

M00 – Suspend program execution;

M01 – Suspend program execution;

M02 – End of program;

M03 – Enable clockwise rotation of the spindle;

M04 – Enable counterclockwise spindle rotation;

M05 – Stop spindle rotation;

M08 – Activation of coolant supply;

M09 – Shutdown of coolant supply;

M30 – End of program;

M97 – Calling an internal subroutine;

M98 – Calling an external subroutine;

M99 – End of subprogram;

X – Absolute coordinate along the x-axis (in diameters);

Z – Absolute coordinate along the z-axis;

U – Relative x-axis coordinate (in diameters) – for code system A;

W – Relative z-axis coordinate – for the A code system;

F – Linear travel speed of the tool;

S – Spindle speed;

T – Tool change command (example, T0202);

R – Arc radius, parameter of the drilling or turning cycle;

L – Number of subroutine calls or number of repeats of the constant loop;

Q – Drilling or turning cycle parameter;

P – Subprogram ID, time or turning cycle parameter;

I – Displacement of the center of the arc along the x-axis;

K – Displacement of the arc center in the z-axis, threading parameter.

Next in the description of G-codes the codes of system A will be considered.

6.1. Linear Interpolation Commands G00 and G01

The standard format for command blocks is G00 and G01:

G00 X_ (U_) Z_ (W_);

G01 X_ (U_) Z_ (W_) (F_);

The G00 command is used to accelerate the tool in a straight line. The code G00 is modal, i.e. it remains active until another command from the same group is called, e.g. G01, G02 or G03. This command is active in the initial state of the TNC. Accelerated positioning is used to bring the tool to the workpiece before machining and to move the tool away from the workpiece after machining.

The G01 command is used to move the tool linearly at the feed rate defined with address F in the same block as G01 or in previous blocks of the control program. The code G01 is modal. Displacement with linear interpolation is used for cutting in a straight line.

6.2. Circular Interpolation Commands G02 and G03

The standard command block format is G02 and G03:

G02 (G03) X_ (U_) Z_ (W_) I_ K_ (R_) (F_);

Circular interpolation is used to machine curved surfaces whose shape is described by an arc of a circle of a certain radius. Two methods of arc programming are used.

The first method involves specifying the coordinates of the arc center using the addresses I, K and the endpoint X (U), Z (W), and the arc radius is calculated automatically. The second method involves specifying the arc radius R and the coordinates of the end point X (U), Z (W), and the coordinates of the arc center are calculated automatically.

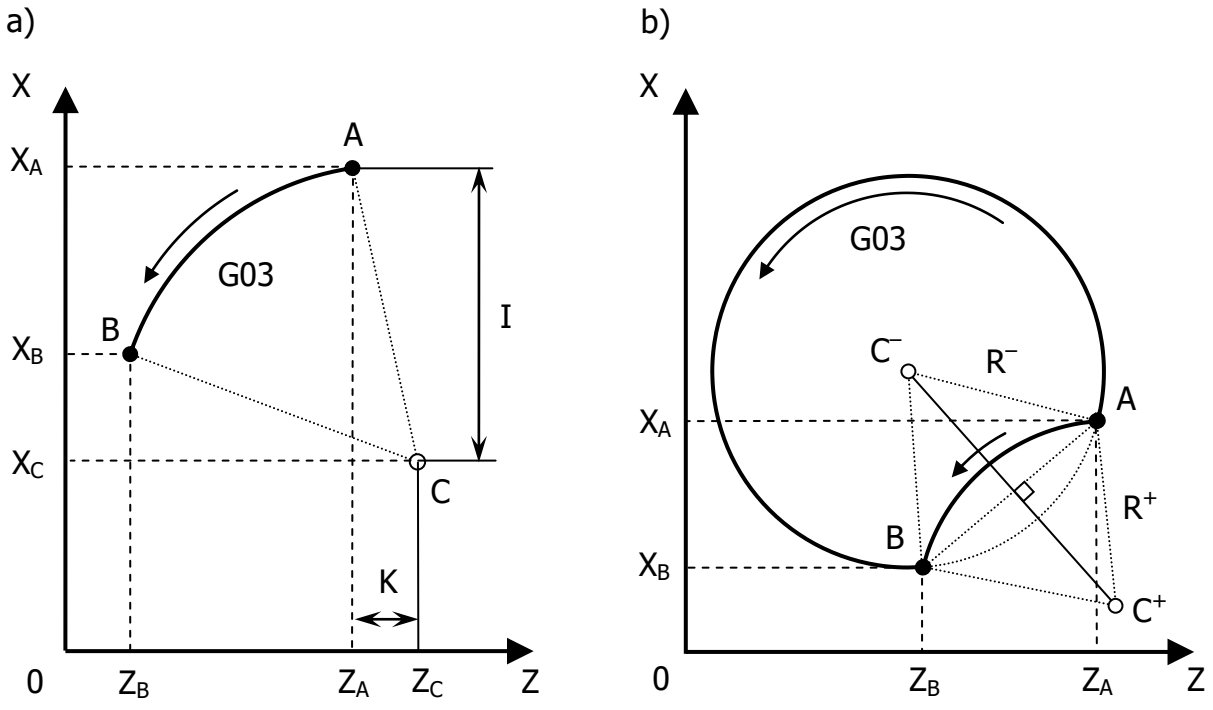

Figure 19 – Arc programming principle

Clockwise circular interpolation is specified with the G02 command and counterclockwise circular interpolation with the G03 command, respectively. If the arc center is specified with addresses I, K, the offsets relative to the arc start point in the x and z axes, respectively, are specified (Fig. 19.a). If the arc is specified with radius R, the control system takes into account the sign of the radius to calculate the coordinates of the arc center (Fig. 19.b).

When defining an arc, the control system checks the distances from the arc center to the start and end points. If the distances have a discrepancy that exceeds the permissible value, the control system will generate an error message.

6.3. Programming Chamfers and Roundings

Chamfers and roundings can be added between linear motion blocks. The additional addresses I or K are used to specify a chamfer:

G01 X_ (U_) Z_ (W_) I_ (K_);

The address I (K) is followed by a numeric value that specifies the chamfer. A chamfer can be built at segment-to-section conjugations.

An additional address R is used to specify the rounding:

G01 X_ (U_) Z_ (W_) R_;

The address R is followed by a numeric value that specifies the radius of the rounding. The rounding can be built at segment-to-section conjugations.

Mandatory conditions for chamfer and rounding programming:

- the chamfer/rounding is only programmed in the block that defines the travel G01 at the working feed;

- the trailing block must also set the travel of G01 at the working feed;

- empty blocks or comment blocks following a block that specifies a chamfer/rounding are not allowed;

- the following block must not include G-codes not related to linear interpolation;

- if the chamfer/rounding end point is outside the path, an error message is displayed.

6.4. Program Execution Delay Command G04

The standard format of the G04 command block:

G04 P_ [X_] [U_];

The G04 command is designed to perform a delay for the time specified with address P. The control system also allows the use of addresses X and U to specify the delay time. If the time is specified with a decimal point, the control system defines the set value as seconds, e.g. P2.5 – delay of the control program execution by 2.5 seconds. If the time is set without decimal point, the control system defines the set value as milliseconds, e.g. P4000 – delay of the control program execution for 4000 milliseconds or 4 seconds. Command G04 can be programmed in the same block with other commands, and the delay will be executed upon completion of the block execution.

6.5. Measure System Selection Commands G20, G21

Standard G20 and G21 command block format:

G20 (G21) _;

Command G20 sets the programming mode in inches. The G21 command sets the programming mode in millimeters. The G20 and G21 codes are modal. The default programming mode (G20 or G21) is determined by the application settings.

6.6. Command to Return to the Reference Point G28

The standard format of the G28 command block:

G28 X_ (U_) Z_ (W_);

Command G28 is intended for accelerated movement of the actuators to the machine reference point with the possibility of setting an intermediate point.

The number of coordinates defined in one block with the code G28 determines the number of axes that are used to return to datum. If the G28 command

is specified without specifying X (U), Z (W) coordinates, the control returns the tool to the reference position in two axes simultaneously without moving to an intermediate point.

The coordinates X (U), Z (W), defined in one block with code G28, define the position of the intermediate point (in the active working coordinate system) through which the control system returns the machine's actuators to zero positions.

The following is an example of how to use the G28 command.

G28 U100 W0;

In this program block, the tool is moved 100 units along the x-axis (in diameters) relative to the starting point and then moved to the reference point.

It is recommended to use the G28 command in relative positioning mode (U and W codes) to avoid collision of the tool with the workpiece.

If you call the G28 command in the automatic tool radius compensation mode (G41/G42), the correction vector is temporarily reset to zero. After executing the G28 command, automatic tool radius compensation is resumed.

6.7. Single-Pass Threading Commands G32 and G34

The standard format of G32 and G34 command blocks:

G32 X_ (U_) Z_ (W_) F_;

G34 X_ (U_) Z_ (W_) K_ F_;

The G32 command is used to cut cylindrical/conical threads with a constant pitch in one pass. The code G32 is modal, i.e. it is valid until another command from the same group is called, e.g. G00, G01, G02 or G03. Before calling the G32 command, the constant spindle speed mode (G97) and the feed rate per spindle revolution mode (G99) must be set. The thread pitch is set with address F in the same block as G32 or in previous blocks of the control program.

The G34 command is used for variable pitch thread cutting. The pitch change is programmed with address K. The numerical value after address K is the increment

of the thread pitch per spindle revolution. For increasing pitch: K>0, for decreasing pitch: K<0, respectively.

6.8. Commands for Controlling the Tool Radius Compensation Mode G40, G41, G42

The standard format for command blocks is G40, G41, and G42:

G41 (G42) _;

…

G40 _;

The control system has a built-in algorithm for calculating the equidistance when machining contours in the automatic tool radius correction mode. The algorithm for calculating the equidistance is based on the general calculation principles of the Fanuc control system. The automatic tool radius correction mode is activated with modal codes G41 and G42. If the tool is to be moved to the left of the programmed toolpath by the value of its radius at the apex, the G41 command is used (Fig. 20.a). To move the tool to the right of the trajectory, the command G42 is used (Fig. 20.b). The control system automatically calculates the path to be followed by the tool, based on the programmed part contour and the tool radius value in the table of tool correctors.

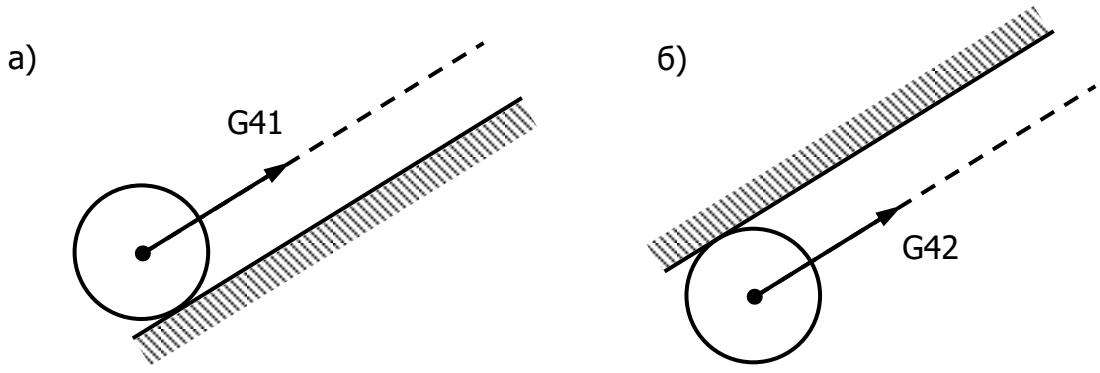

Figure 20 – The principle of tool movement in the automatic radius correction mode

Commands G41 and G42 are canceled by modal code G40. The tool radius correction vector is also temporarily reset when executing commands G28, G53, T0101. The radius compensation mode is not supported for turning cycles G71-G76, G83, G90, G92, G94.

If a program (M00, M01, M02 or M30) is terminated with tool radius correction enabled, the control will generate an error message.

The tool radius correction mode (G41 or G42) can only be activated if the commands G00 or G01 (straight-line motion) are active. If the correction mode is activated in a circular interpolation segment (G02 or G03), the control will generate an error message. Thus, if command G41 (G42) is programmed in the same block as a rectilinear movement, the entry to the equidistance is performed smoothly along a straight line, the beginning of which is located on the programmed trajectory and the end of which is located on the equidistance of the next movement (Fig. 21 – a and b).

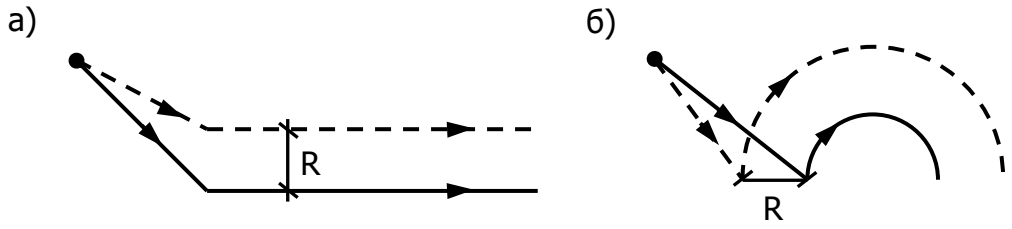

Figure 21 – Enabling automatic tool radius correction mode in one block with linear movement

While calculating the equidistance, the control system checks the following blocks of the control program containing displacements in order to determine the intersection points of the current section with the next section of the calculated path in the x-z plane. Bypassing of acute corners from the outside is performed by constructing an additional linear displacement (Fig. 22 – a and b).

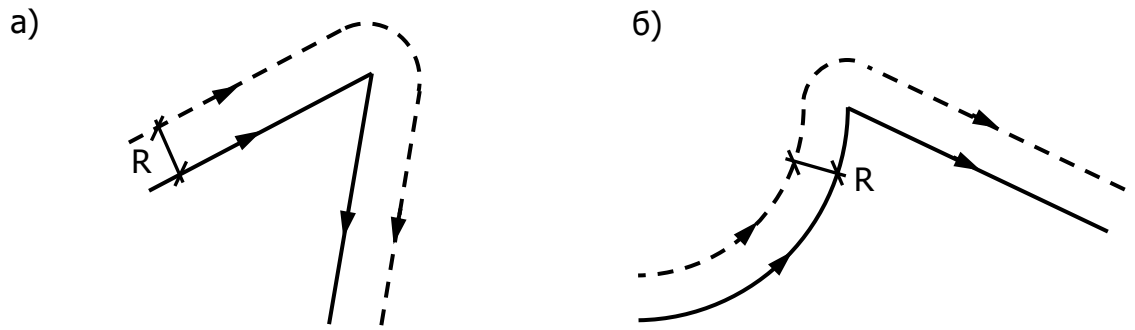

Figure 22 – The principle of bypassing an acute angle in the automatic correction mode for the tool radius

If the radius of the programmed arc is smaller than the tool radius when traversing the arc from the inside, the control system will generate an error message. In addition, if the programmed contour has self-intersections where arc sections meet other sections of the path, the equidistance is calculated according to the principle of reducing the number of self-intersections of the final path.

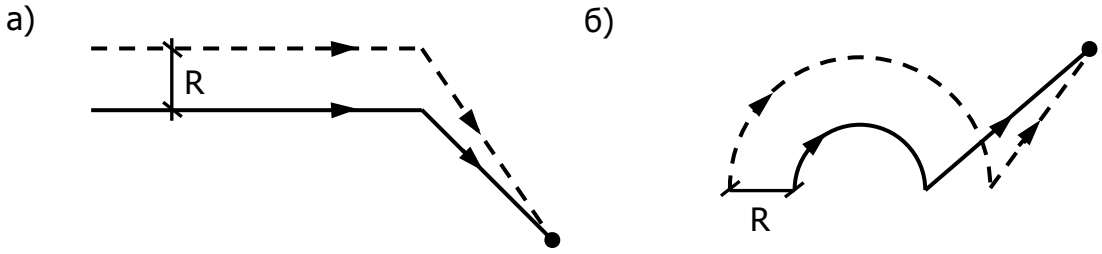

Figure 23 – Disabling the automatic tool radius correction mode in one block with linear movement

The command to cancel the automatic tool radius correction mode G40 must similarly be programmed in the linear motion block (G00 or G01). If the correction mode is canceled in the circular interpolation section (G02 or G03), the control will generate an error message. Thus, if the command G40 is programmed in the same block as the linear motion, the exit from the equidistance is performed smoothly along a straight line, the beginning of which is located on the equidistance of the

previous motion and the end of which is located on the trajectory of the current motion (Fig. 23 – a and b).

In the mode of automatic correction to the tool radius, the direction of the correction vector can be changed by calling the opposite command. For example, if after switching on the left correction mode G41 (in the following program blocks) you set the command G42, the end point of the equidistance of the given path section will be calculated from the opposite side of the trajectory (Fig. 24). Changing the direction of the correction vector is possible only on rectilinear sections of the trajectory.

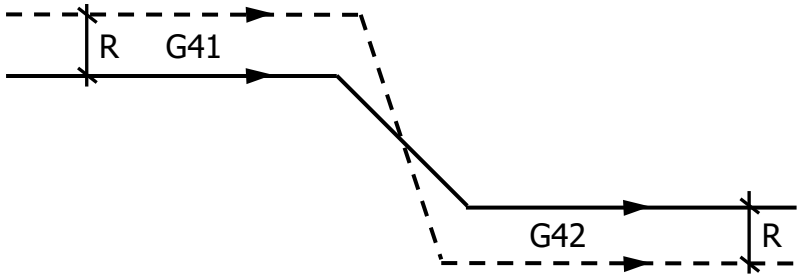

Figure 24 – Change of direction of correction vector on linear section of trajectory

Below is an example of how to use the G42 and G40 commands.

```
G50 S3000;
N10 G21 G99 G40;
G00 G28 U0 W0;
T0101;
X60 Z2;
G96 S300 M4;
G01 G42 Z0 F0.3;
Z-15;
G02 X140 Z-55 R40;
G01 X160;
```

G40 X164;

G00 X200 Z150;

G28 U0 W0;

M30;

The figure below illustrates the contour being machined in this example:

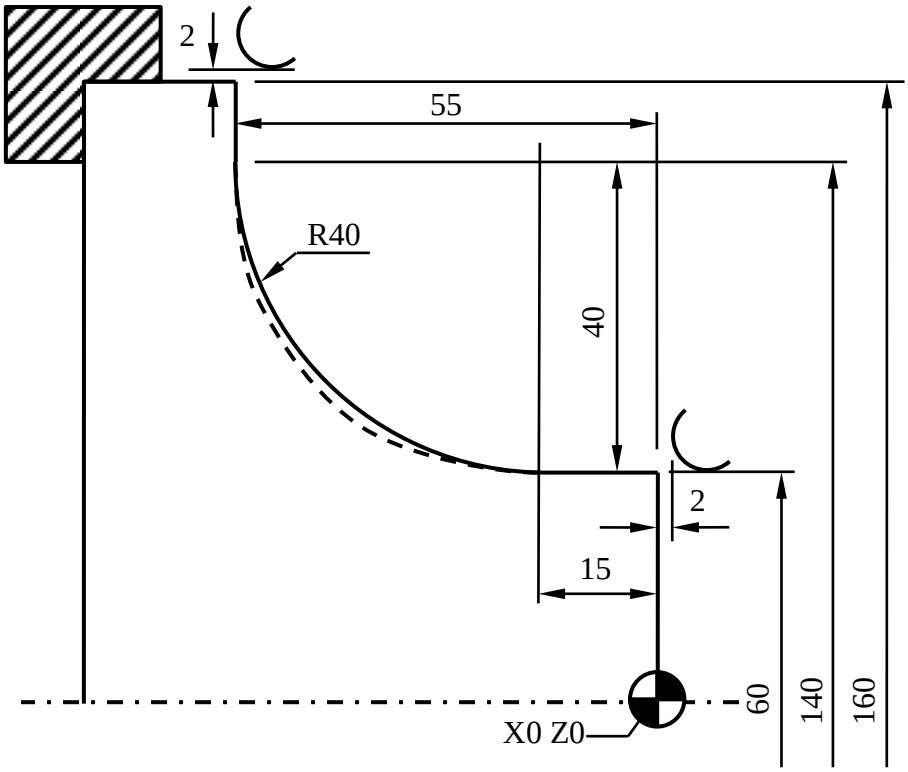

Figure 25 – An example of using radius correction on an arc section of a contour

The dotted line shows the trajectory of the imaginary tip of the cutter.

When you move in the automatic tool radius compensation mode, the actual coordinates of the tool moving along the equidistance are displayed. The equidistant line is also displayed in conjunction with the main trajectory line.

6.9. Coordinate System Offset/Spindle Speed Limit Command G50

The standard format of the G50 command block:

G50 X_ Z_; (смещение активной системы координат)

G50 S_; (ограничение скорости вращения шпинделя)

The G50 command is used to programmatically shift the active working coordinate system to an arbitrary point. This shifts the origin of the coordinate system so that the current tool position corresponds to the values defined in the same block as the G50 command.

The G50 command itself does not cause axial offsets. The offset on the G50 command can be canceled by programming another G50 offset to change the current part correction back to the original value.

Below is an example of using the G50 command to convert the G54 coordinate system.

(G54)

N1 G00 X150 Z200;

N2 G50 X0 Z0;

N3 G01 X100;

N4 G50 X250 Z200;

In block N1, the tool is accelerated to the point [x=150; z=200] of the active working coordinate system (G54). In block N2, the working coordinate system is shifted so that the current position of the tool is now defined by the coordinates [x=0; z=0]. In block N3, the tool moves to the point [x=100; z=0] in linear feed rate. In block N4, the working coordinate system returns to the original position.

When the simulation process is stopped, the programmed G50 offsets are reset to zero.

The G50 command cancels all valid G52 offsets. The G50 command must not be used in the same block as the G53 command.

If no coordinates are set after the G50 command, but address S is set, the command executes the maximum spindle speed limit in constant cutting speed mode (G96).

6.10. Local Coordinate System Setting Command G52

The standard format of the G52 command block:

G52 X_ Z_;

The G52 command creates a local coordinate system within the active working coordinate system by shifting the zero of the active working coordinate system by the values set with the X and Z addresses. The set offset is canceled by calling the G52 command again with zero values X and Z. Thus, the local coordinate system is aligned with the working coordinate system.

When a local coordinate system is set, the sequentially programmed motion commands in absolute positioning mode are the coordinate values in the local coordinate system. Each working coordinate system (G54-G59) has its own local coordinate system.

The following is an example of how to use the G52 command.

(G54)

N1 G00 X50 Z50;

N2 G52 X100;

N3 G01 X0;

N4 G02 X100 R50;

N5 G52 X0;

In block N1, the tool is accelerated to the point [x=50; z=50] of the active working coordinate system (G54). In block N2 the local coordinate system is set: the origin of the working coordinate system is shifted by 100 units in the positive direction of the x-axis, and the tool position in the working coordinate system is defined by the point [x=-50; z=50]. In block N3, the tool moves linearly to the point [x=0; z=50] of the local coordinate system. In block N4, the tool is moved clockwise in a circular motion to the point [x=100; z=50] of the local coordinate system. In

block N5 the local coordinate system is aligned with the working coordinate system (G54) along the x-axis.

When you call the G52 command, the tool coordinates in the local coordinate system are displayed when you perform a traverse.

The G52 command and its accompanying parameters are always programmed in a separate block.

6.11. Command to Change to Machine Coordinate System G53

The standard format of the G53 command block:

G53 _;

When executing command G53, the control temporarily cancels the work coordinate offsets and uses the machine coordinate system. The G53 command is non-modal, i.e. it only operates in the block in which it is defined. Linear or circular movements can be programmed in conjunction with G53.

The G53 command only works in absolute positioning mode. If the G53 command is set in relative positioning mode, the control system will generate an error message.

If you call the G53 command in the automatic tool radius compensation mode (G41/G42), the correction vector is temporarily reset to zero. After executing command G53, automatic tool radius compensation is resumed.

It is unacceptable to use the G53 command in the same block as the G28 or G50 commands.

6.12. Commands for Switching the Working Coordinate System G54-G59

The standard command block format is G54-G59:

G54 (G55, G56 …) _;

This simulation provides the possibility to use six predefined working coordinate systems. The selection and setting (as active) of the working coordinate system is done using the modal codes: G54, G55, G56, G57, G58, G59.

The zero offsets of each work coordinate system relative to the machine zero of the coordinate table are preset in the corresponding section of the workpiece parameter screen. These offsets can be changed during the execution of the control program using commands G50 and G52.

The tool coordinates in the active working coordinate system are shown in the display when traversing.

Commands G54-G59 can be defined in a separate program block or in blocks together with other commands.

6.13. Macro Program Call Command G65

The standard format of the G65 command block:

G65 P_ (L_) (A_ B_ C_ D_ E_ F_ H_ I_ J_ K_ M_ Q_ R_ S_ T_ U_ V_ W_ X_ Y_ Z_);

The G65 command is used to call a macro program whose code is located in the main control program or in a separate program (with its own block numbering). If the macro program is located in the code of the main program, it must have a header of the OXXXXXX type, specified in a separate block. The macro program code can also be specified in an additional tab of the text editor. In one block with the G65 command it is obligatory to set the parameter P. The integer parameter P defines the four-digit number of the macro program.

The optional integer parameter L defines the number of consecutive calls of the macro program, e.g. L3 means that the macro program will be called 3 times in a row. Up to 999 consecutive macro program calls are allowed. If parameter L is not specified in the G65 command block, the macro program is called once.

Any macro program called must be terminated with the termination code M99. When the macro program is completed, control is transferred to the block following the block with the macro program call command (G65).

Macro programs use the local variable space #1 –#33, to which values can be assigned by entering alphabetic arguments after the G65 command. Any changes to the values of local variables in the macro program code only affect the macro program itself and are not returned to the main program.

The basic method of passing arguments to a macro program involves the use of various letter addresses A–Z (except G, L, N, O, P). Each letter address corresponds to a certain number of a local variable:

A – #1	I – #4	T – #20
B – #2	J – #5	U – #21
C – #3	K – #6	V – #22
D – #7	M – #13	W – #23
E – #8	Q – #17	X – #24
F – #9	R – #18	Y – #25
H – #11	S – #19	Z – #26

Example: G65 P1001 L12 A35.5 B–14.2 Q8;

In this example, macro program O1001 is called 12 times in a row, and when the macro program is called, the values of local variables are passed to it: #1=35.5; #2=-14.2; #17=8.

An alternative method of passing arguments to the macro program involves the use of letter addresses A, B, C, I, J, K. Each of the addresses I, J, K can be set up to 10 times in one block with the code G65. The correspondence of the letter addresses to the numbers of local variables is as follows:

A – #1	I4 – #13	I8 – #25
B – #2	J4 – #14	J8 – #26
C – #3	K4 – #15	K8 – #27
I1 – #4	I5 – #16	I9 – #28
J1 – #5	J5 – #17	J9 – #29
K1 – #6	K5 – #18	K9 – #30
I2 – #7	I6 – #19	I10 – #31
J2 – #8	J6 – #20	J10 – #32

K2 – #9	K6 – #21	K10 – #33
I3 – #10	I7 – #22	
J3 – #11	J7 – #23	
K3 – #12	K7 – #24	

Example: G65 P1002 L3 I68.4 J-13 K4 I-18.5 J-9 K50.2 I19.2 J-1;

In this example, macro program O1002 is called 3 times in a row, and when the macro program is called, the values of local variables are passed to it: #4=68.4; #5=-13; #6=4; #7=-18.5; #8=-9; #9=50.2; #10=19.2; #11=-1; #12=88.8.

Macro programs are processed by analogy with external subroutines. It is allowed to use up to 20 nested macro programs with their own code structure – conditional and unconditional transitions, loops, etc.

6.14. Spindle Rotation Mode Switching Commands G96, G97

Standard G96/G97 command block format:

G96 (G97) S_;

The modal command G96 enables the constant cutting speed mode. After address S, the cutting speed is entered in [m/min] or [ft/min], depending on the selected measuring system (G21 or G20). The actual spindle speed in G96 mode is calculated automatically depending on the current x-axis position of the tool. Before using the G96 command, you must enter the maximum spindle speed limit with G50.

Command G97 enables constant spindle speed mode, independent of the current x-axis position of the tool. The spindle speed [rpm] is programmed directly with address S.

6.15. Feed Mode Switching Commands G98, G99

Standard G98/G99 command block format:

G98 (G99) F_;

The modal command G98 enables the minute feed mode. In this mode, the feed rate set with address F is defined in [mm/min] or [inch/min] depending on the

selected measuring system (G21 or G20). In G98 mode, the working feed rate is determined only by the F setpoint, regardless of the spindle speed.

Modal command G99 enables the recycle feed mode. In this mode, the feed rate set with address F is defined in [mm/rev] or [inch/rev], depending on the selected measuring system (G21 or G20). In G99 mode, the working feed rate is determined by the setpoint F multiplied by the spindle speed S. If the spindle is stopped, there is no translational movement of the slide.

6.16. Commands to Suspend Program Execution M00, M01

The standard command block format is M00/M01:

_ M01 (M00);

In this simulation, codes M00 and M01 are executed in the same way and are used to suspend the execution of the control program until it is manually resumed. The M00/M01 code can be programmed either in a separate block or in the same block with other commands, and the control program will be suspended when the block is completed.

In this simulation, the M00/M01 code can be used to monitor the correctness of the trajectories and machining modes in order to debug the control program.

6.17. Program End Commands M02, M30

The standard command block format is M02/M30:

_ M02 (M30);

In this simulation, codes M02 and M30 are executed in the same way and are used to complete the control program execution process. The M02/M30 code can be programmed either in a separate block or in the same block with other commands, and the control program will be completed after the block is executed.

If the M02/M30 code is detected, parsing of the control program is terminated. Spindle rotation and coolant supply are automatically switched off. Internal subroutines called by code M97 must be programmed after code M02/M30.

6.18. Spindle Rotation Control Commands M03, M04, M05

The standard command block format is M03, M04, and M05:

M03 (M04) S_;

Commands M03 and M04 are used to enable clockwise and counterclockwise spindle rotation, respectively. When the spindle rotation is enabled, the rotation speed [rpm] must be set with address S in the same block as the M03/M04 command or in the previous blocks.

Spindle rotation is switched off by command M05.

6.19. Tool Switching Command T

The standard format of the T command block:

T_;

The tool change command consists of two numbers (e.g. "T0101"), the first of which indicates the tool position in the turret and the second of which indicates the tool number in the tool corrector table. The control system will apply the axial outreach values corresponding to the specified corrector number.

6.20. Coolant Control Commands M08, M09

The standard format of the M08 and M09 command block:

_ M08;

…

_ M09;

In this simulation, code M08 is used to activate the coolant supply to the cutting zone. The coolant supply is deactivated by code M09. Codes M08 and M09 can be programmed in a separate block or in the same block with other commands, and the coolant supply/disconnect operation will be performed when the block is completed.

6.21. Subprogram Control Commands M97, M98, M99

The standard command block format is M97, M98, and M99:

M97 P_ (L_);

…

M98 P_ (L_);

…

M99;

The M97 command is used to call an internal subprogram, the code of which is located in the main control program after the M02/M30 termination command. In one block with the M97 command, it is mandatory to set the parameter P. The integer parameter P defines the number of the subprogram start block, e.g. P100 means transferring control to the subprogram start block N100, and the subprogram start block must always contain the address N.

The optional integer parameter L defines the number of consecutive subroutine calls, e.g. L3 means that the internal subroutine will be called 3 times in a row. Up to 999 consecutive subroutine calls are allowed. If parameter L is not specified in the M97 instruction block, the subprogram is called once.

The M98 command is used to call an external subprogram. Similar to macro programs (G65), an external subprogram can be located in the main control program or in a separate program (with its own block numbering). If the subprogram is located in the code of the main program, it must have a header of the form OXXXXXX, specified in a separate block. The code of the subprogram can also be specified in an additional tab of the text editor. In one block with the M98 command it is obligatory to set the parameter P. The complex parameter P defines the number of the subprogram and can also contain the number of subprogram calls. For example, the parameter P31002 means that the external subroutine O1002 will be called 3 times in a row. In this simulation, if the external subroutine is located in an additional tab of the text editor, its number must correspond to the tab number of the

text editor. If the parameter P does not contain the number of subprogram calls, it is allowed to use the parameter L similar to the format of the command M97.

Any called subprogram must be terminated with the termination code M99. When the subprogram is completed, control is transferred to the block following the block with the subprogram call command (M97 or M98).

In this simulation, a multi-level subroutine call is allowed – multiple calls to a subroutine from a subroutine. The order of calling and terminating nested subroutines must be observed.

6.22. Roughing Cycle for Longitudinal Contouring G71

The standard format of the G71 command block:

G71 U_ R_;

G71 P_ Q_ U_ W_ F_ S_;

where: [first line] U – machining depth for roughing passes (programming mode in radii) [mm]; R – distance after each pass [mm]; [second line] P – sequence number of the first contour description frame; Q – sequence number of the last contour description frame; U – value (programming mode in diameters) and direction of finishing allowance removal in X axis [mm]; W – value and direction of finishing allowance removal in Z axis [mm]; F – feed rate for roughing; S – spindle speed/cutting speed for finishing.

The following is an example of how to use the G71 cycle (Fig. 26).

G00 X32 Z3;

G71 U2 R0.2;

G71 P100 Q150 U0.3 W0.3 F0.15;

N100 G01 X10 Z0;

N110 G01 Z-8;

N120 G01 X20 Z-18;

N130 G01 X30;

N140 G01 Z-25;

N150 G01 X32;

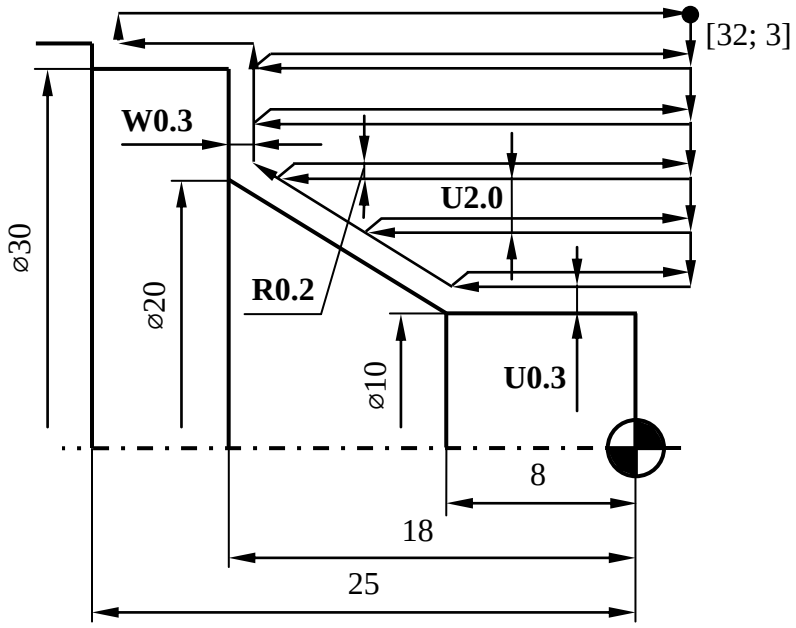

Figure 26 – Path diagram for G71 cycle execution

6.23. Cross Contouring Roughing Cycle G72

The standard format of the G72 command block:

G72 W_ R_;

G72 P_ Q_ U_ W_ F_ S_;

where: [first line] W – machining depth for roughing passes [mm]; R – distance after each pass [mm]; [second line] P – sequence number of the first contour description frame; Q – sequence number of the last contour description frame; U – value (programming mode in diameters) and direction of removal of finishing allowance in X axis [mm]; W – value and direction of removal of finishing allowance in Z axis [mm]; F – feed rate for roughing; S – spindle speed/cutting speed for finishing.

The following is an example of how to use the G72 cycle (Fig 27).

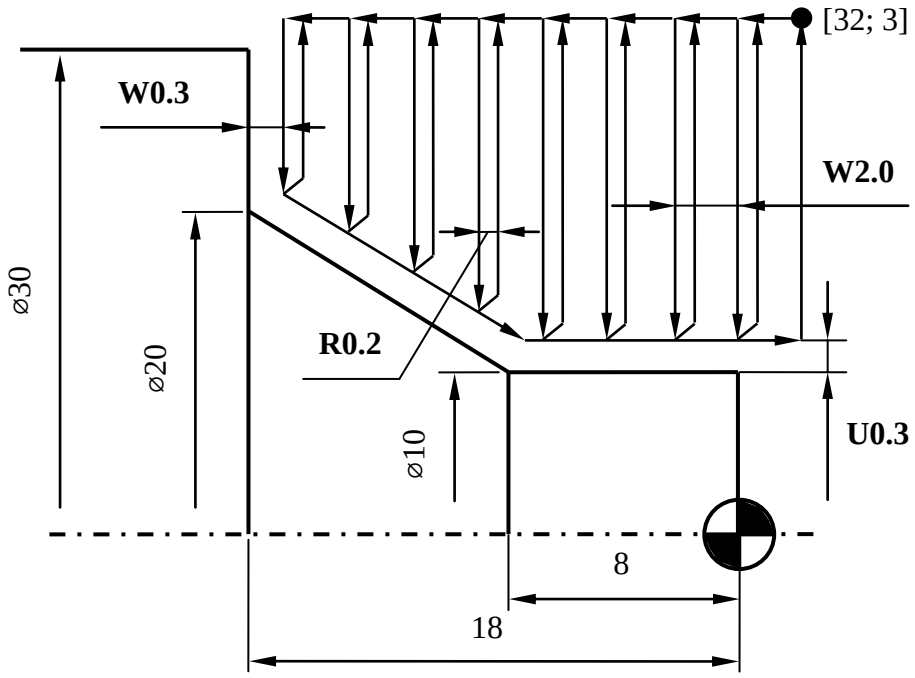

Figure 27 – Path diagram for G72 cycle execution

```
G00 X32 Z3;
G72 W2 R0.2;
G72 P100 Q130 U0.3 W0.3 F0.15;
N100 G01 X10 Z0;
N110 G01 Z-8;
N120 G01 X20 Z-18;
N130 G01 X30;
```

6.24. Contouring Cycle G73

The standard format of the G73 command block:

G73 U_ W_ R_;

G73 P_ Q_ U_ W_ F_ S_;

where: [first line] U – value (programming mode in radii) and direction of total allowance removal along X axis [mm]; W – value and direction of total

allowance removal along Z axis [mm]; R – number of consecutive passes for roughing allowance removal, including the finishing pass; [second line] P – sequence number of the first contour description frame; Q – sequence number of the last frame of the contour description; U – value (programming mode in diameters) and direction of roughing allowance removal along the X axis [mm]; W – value and direction of roughing allowance removal along the Z axis [mm]; F – feed rate for roughing; S – spindle speed/cutting speed for finishing.

The following is an example of how to use the G73 cycle (Fig. 28).

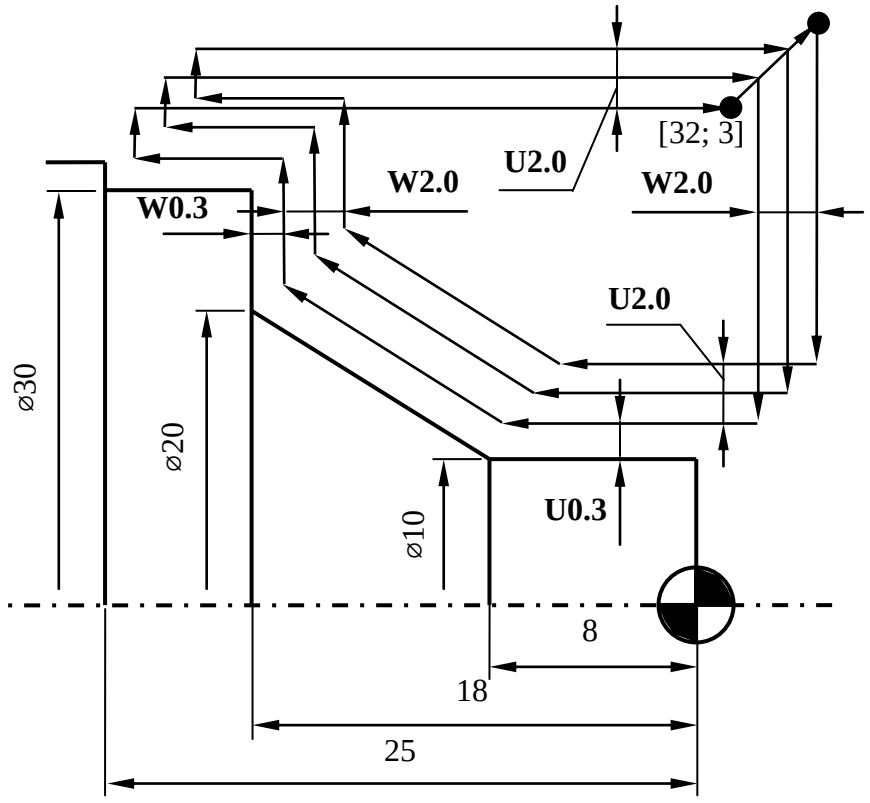

Figure 28 – Path diagram for G73 cycle execution

G00 X32 Z3;

G73 U2 W2 R3;

G73 P100 Q150 U0.3 W0.3 F0.15;

N100 G01 X10 Z0;

N110 G01 Z-8;

N120 G01 X20 Z-18;

N130 G01 X30;

N140 G01 Z-25;

N150 G01 X35;

6.25. Finishing Contouring Cycle G70

The standard format of the G70 command block:

G70 P_ Q_ F_ S_;

where: P is the sequence number of the first contour description frame; Q is the sequence number of the last contour description frame; F is the feed rate for finishing; S is the spindle speed/cutting speed for finishing.

Below is an example of how to use the G70 cycle (Fig. 29).

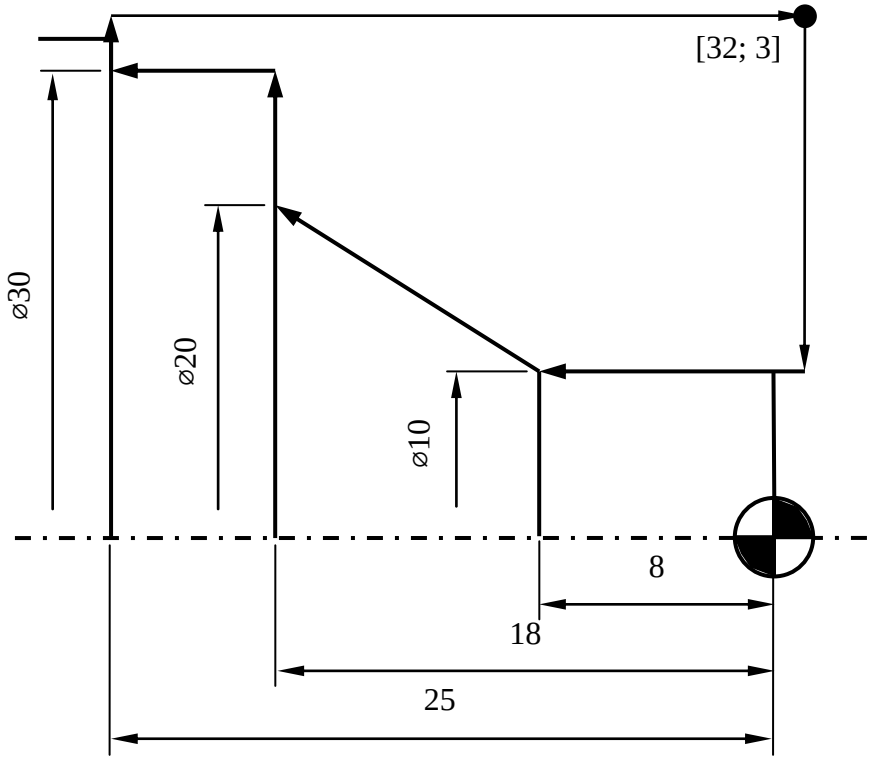

Figure 29 – Path diagram for G73 cycle execution

G00 X32 Z3;

G70 P100 Q140 F0.1 S2000;

N100 G01 X10 Z0;

N110 G01 Z-8;

N120 G01 X20 Z-18;

N130 G01 X30;

N140 G01 Z-25;

6.26. Automatic Side Groove Machining Cycle G75

The standard format of the G75 command block:

G75 R_;

G75 X_(U_) Z_(W_) P_ Q_ F_;

where: [first line] R is the distance to which the tool is retracted after completing the step of turning [mm]; [second line] X(U) is the X-axis coordinate of the end point [mm]; Z(W) is the Z-axis coordinate of the end point [mm]; P is the X-axis step of turning [μm]; Q is the Z-axis step of turning [μm]; F is the feed rate.

Below is an example of how to use the G75 cycle (Fig. 30).

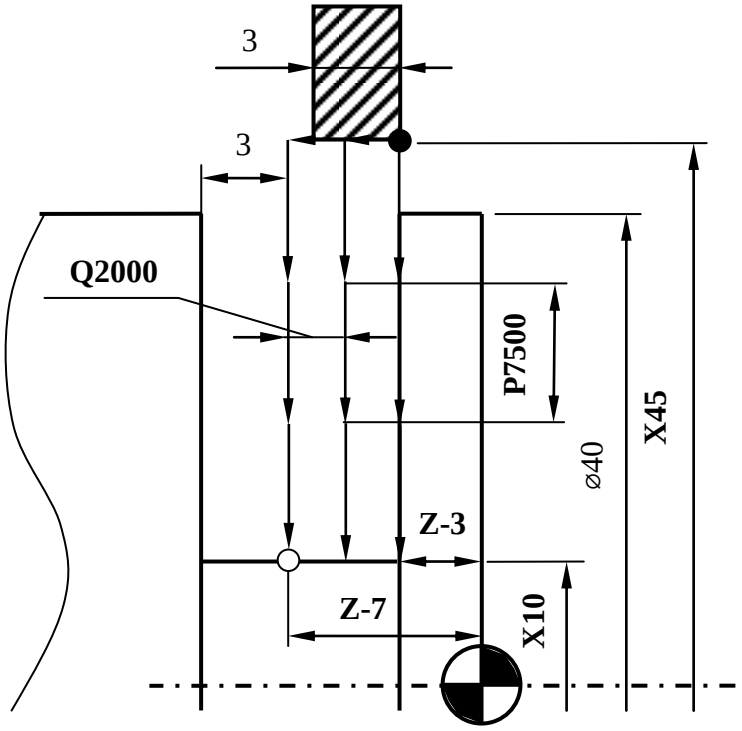

Figure 30 – Path diagram for G75 cycle execution

G00 X45 Z-3;

G75 R0.5;

G75 X10 Z-7 P7500 Q2000 F0.15;

6.27. Cycle for Automatic Machining of End Grooves G74

The standard command block format is G74:

G74 R_;

G74 X_(U_) Z_(W_) P_ Q_ F_;

where: [first line] R is the distance to which the tool is retracted after completing the step of turning [mm]; [second line] X(U) is the X-axis coordinate of the end point [mm]; Z(W) is the Z-axis coordinate of the end point [mm]; P is the X-axis step of turning [μm]; Q is the Z-axis step of turning [μm]; F is the feed rate.

Cycle G74 is similar to cycle G75 except that the groove is machined along the Z axis. This cycle can be used for intermittent drilling of end holes.

6.28. Automatic Tapping Cycle G76

The standard format of the G76 command block:

G76 Pxxyyzz Q_ R_;

G76 X_(U_) Z_(W_) R_ P_ Q_ F_;

where: [first line] xx – two-digit number of finishing passes; yy – two-digit number defining the chamfer size; zz – two-digit number defining the cutting edge angle of the tool; Q – minimum thread depth (radius programming mode) [μm]; R – total depth of threading during finishing passes [mm]; [second line] X(U) – X-axis end point coordinate [mm]; Z(W) – Z-axis end point coordinate [mm]; R – X-axis displacement value for tapered thread cutting (not programmable for cylindrical thread cutting) [mm]; P – thread height [μm]; Q – thread depth for the first pass [μm]; F – Z-axis thread pitch.

The following is an example of how to use G76 cycle (Fig. 31).

M03 S150;

G00 X90 Z10;

G76 P020060 Q1000 R0.5;

G76 X70 Z-30 P5000 Q2000 F5.5;

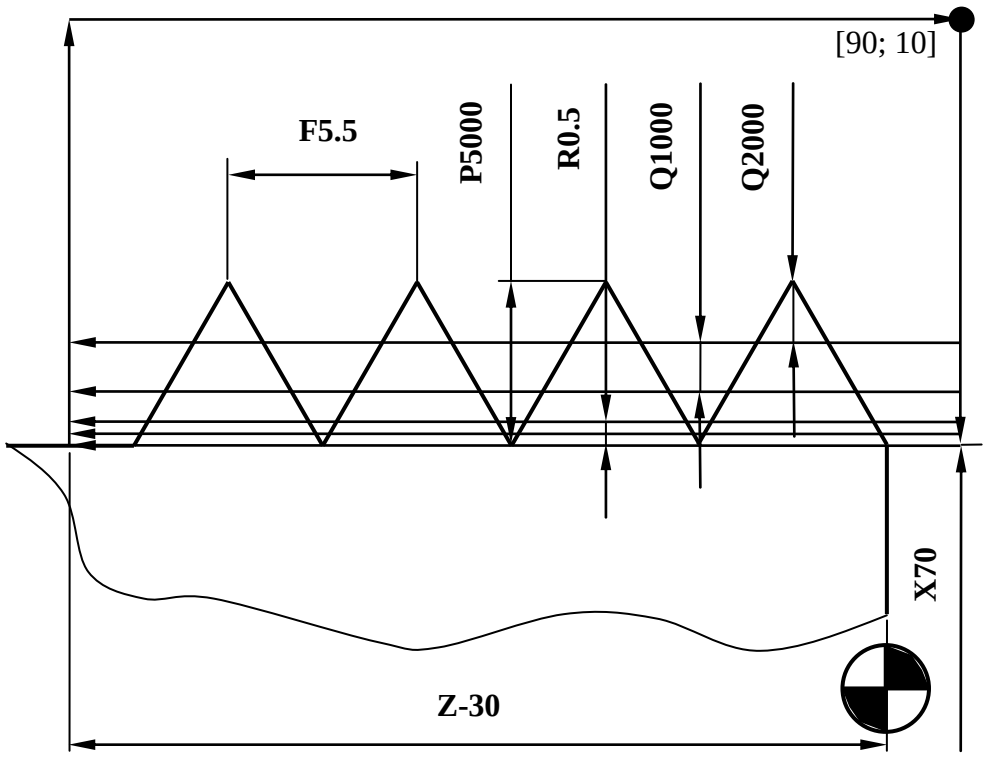

Figure 31 – Path diagram for G76 cycle execution

6.29. Longitudinal Roughing Cycle G90

The standard format of the G90 command block:

G90 X_ (U)_ Z_ (W)_ R_ F_;

where: X(U) – coordinate of the end point of the first pass along the X axis [mm]; Z(W) – coordinate of the end point of the first pass along the Z axis [mm]; R – change of the radius of the cone base [mm]; F – feed rate.

Below is an example of how to use the G90 cycle (Fig. 32).

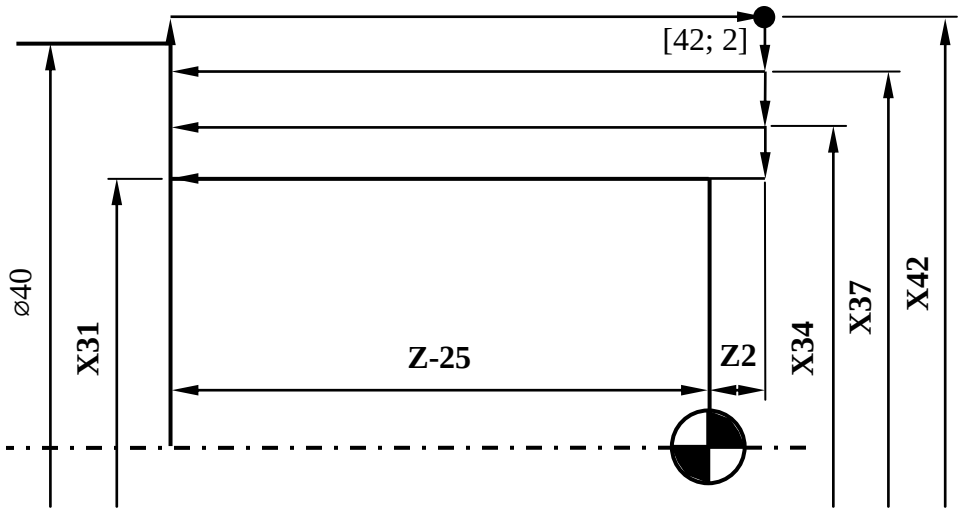

Figure 32 – Path diagram for G90 cycle execution

G00 X42 Z2;

G90 X37 Z-25 F0.1;

X34;

X31;

6.30. End Roughing Cycle G94

The standard format of the G94 command block:

G94 X_(U_) Z_(W_) R_ F_;

where: X(U) – coordinate of the end point of the first pass along the X axis [mm]; Z(W) – coordinate of the end point of the first pass along the Z axis [mm]; R – change of the radius of the cone base [mm]; F – feed rate.

Below is an example of how to use the G94 cycle (Fig. 33).

G00 X32 Z1;

G94 X10 Z-2 F0.1;

Z-4;

Z-6;

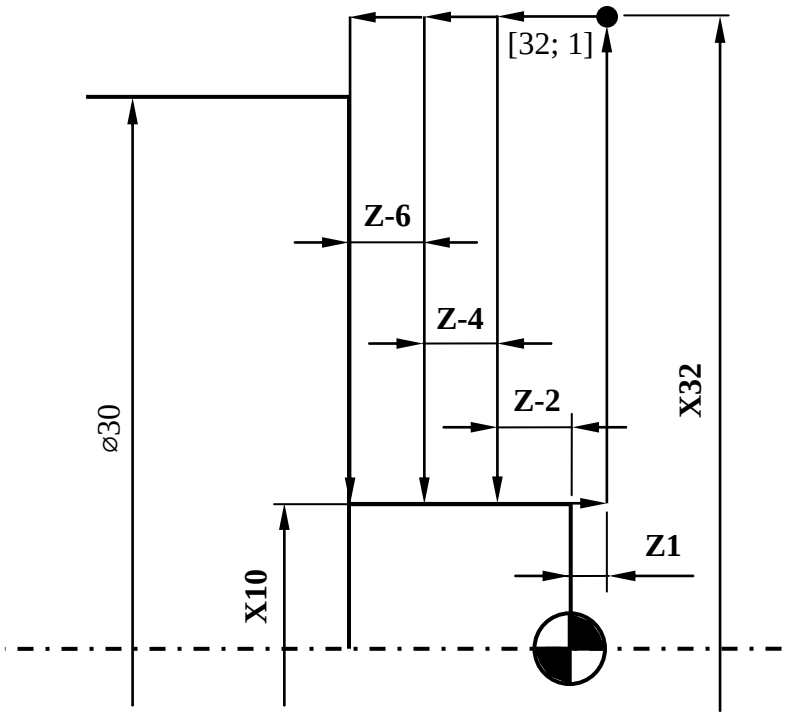

Figure 33 – Path diagram for G94 cycle execution

6.31. Thread Cutting Cycle G92

The standard format of the G92 command block:

G92 X_(U_) Z_(W_) R_ F_;

where: X(U) – coordinate of the end point of the first threading pass in the X axis [mm]; Z(W) – the coordinate of the end point of the first threading pass in the Z axis [mm]; R – amount of movement in the X axis for tapered thread cutting (not programmable for cylindrical thread cutting) [mm]; F – the thread pitch in the Z axis.

Below is an example of how to use the G92 cycle (Fig. 34).

G00 X42 Z2;

G92 X30 Z-25 F1.0;

X29;

X28;

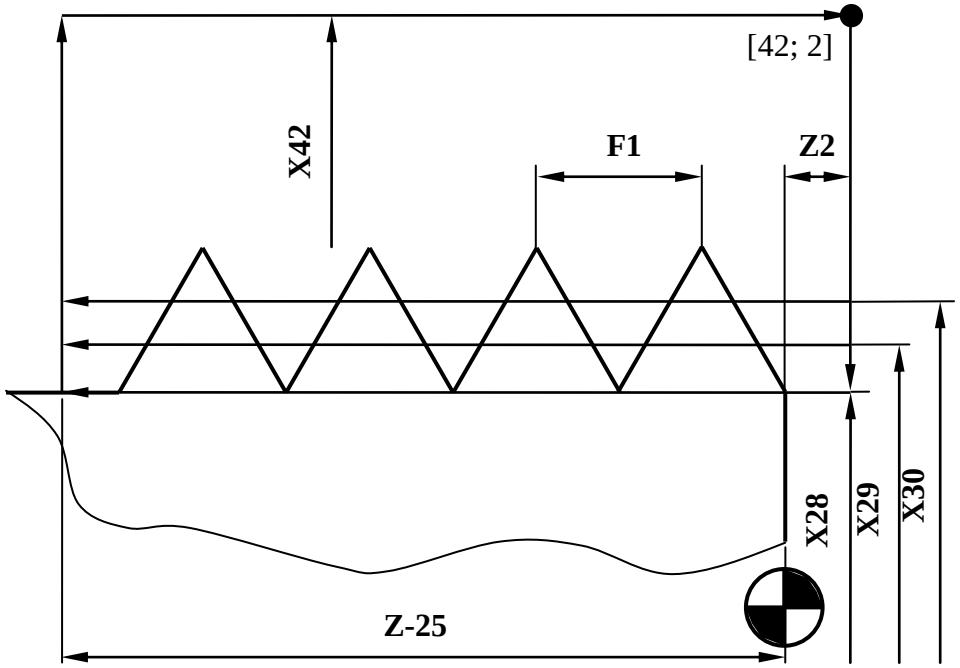

Figure 34 – Path diagram for G92 cycle execution

6.32. Standard Multi-Pass (Intermittent) Drilling Cycle G83

The standard format of the G83 command block:

G83 X_(U_) Z_(W_) R_ P_ Q_ F_;

where: X(U) – X-axis end point coordinate [mm]; Z(W) – Z-axis end point coordinate [mm]; R – Z-axis absolute coordinate of the tool retraction plane [mm]; P – dwell time at the bottom of the hole in milliseconds; Q – Z-axis drilling step [μm]; F – feed rate.

Cycle G83 drills a hole with periodic full withdrawal of the tool from the hole. The additional parameter Q defines the depth of one working pass. After each work pass, the tool is retracted either to the starting position or to the retraction plane, after which drilling continues until the final hole depth is reached.

6.33. G80 Standard Cycle Cancel Command

The standard format of the G80 command block:

G80 _;

The modal command G80 cancels the active constant cycle G83. The G80 code can be programmed either in a separate block or in one block with other commands. By default, the G80 command is active when the control program is started.

6.*34. Parametric Programming*

The simulation model allows you to interpret the main features of the Macro B parametric programming language: working with variables, unconditional and conditional transitions, as well as loops with precondition.

6.34.1. Working with Variables

The presented control system works with common and local variables of the #_ type. The variable name (number) is specified as an integer. The variable number is preceded by the grid symbol «#». The variable can be assigned a value using the assignment operator «=».

The control system interprets arithmetic and logical expressions, as well as mathematical functions using nested brackets «[...]».

Supported arithmetic operators:

1) equality: #1=#2;

2) addition: #1=#2+#3;

3) subtraction: #1=#2–#3;

4) multiplication: #1=#2*#3;

5) division: #1=#2/#3.

It is not allowed to use a zero divisor value in the division operation, otherwise the control system will generate an error message.

Supported math functions:

1) arcsine: #1=ASIN[#2];

2) sine: #1=SIN[#2];

3) arccosine: #1=ACOS[#2];

4) cosine: #1=COS[#2];

5) arctangent: #1=ATAN[#2];

6) tangent: #1=TAN[#2];

7) square root: #1=SQRT[#2];

8) absolute value: #1=ABS[#2];

9) rounding: #1=ROUND[#2];

10) exponent: #1=EXP[#2];

11) natural logarithm: #1=LN[#2];

12) rounding down: #1=FIX[#2];

13) rounding up: #1=FUP[#2].

The arguments of the trigonometric functions SIN, COS, and TAN are values in angular degrees.

The control system monitors the correctness of the ASIN, ACOS, TAN, SQRT and LN function arguments. The numerical value of the argument of the ASIN and ACOS functions must correspond to the range [-1; 1]. The value 90 is not allowed as an argument of the TAN function. Negative values are not permitted as an argument of the SQRT function. Values greater than zero are not allowed as an argument of the LN function.

The ROUND function rounds expression and variable values to integers in assignment operations. When the ROUND function is used after alphabetic addresses, rounding is performed to 3 digits after the point for metric units and 4 digits after the point for inch units.

Supported logical operators:

1) conjunction И: #1=#2 AND #3 (alternative notation: &);

2) disjunction ИЛИ: #1=#2 OR #3 (alternative notation: |).

Supported comparison operators (used in conditional expressions):

1) is equal to: EQ (alternative notation: ==);

2) is not equal to: NE (alternative notation: <>);

3) more: GT (alternative notation: >);

4) greater than or equal to: GE (alternative notation: >=);

5) less: LT (alternative notation: <);

6) less than or equal to: LE (alternative notation: <=).

Variables can be specified as values after letter addresses, e.g. X#1. Parenthesized expressions can also be used as values after letter addresses, e.g., Y[#2+SIN[#3]*#4]. Integer values of G/M codes can also be specified by variables, e.g., G#4.

When composing mathematical or logical expressions, the order of brackets must be observed. If the number of opening and closing brackets does not match in one expression, the control system will display an error message.

When calculating expressions, the order of mathematical operations is performed in accordance with the rules of mathematics – parsing of algebraic expressions is performed using the «Reverse Polish Notation» method.

6.34.2. Unconditional and Conditional Transitions

The operator of unconditional transition GOTO is intended to transfer control to a certain block of the program specified after the word GOTO, for example, when executing the operator GOTO20, the transition to the block N20 will take place. The block to which the unconditional transition is performed must contain the address N.

Below is an example of using an unconditional transition.

N10 G00 X0 Z0;

N12 GOTO16;

N14 M30;

N16 G01 U100;

In block N10, the tool is accelerated to the beginning of the active working coordinate system. In block N12, an unconditional transition to block N16 is carried

out, in which the working linear displacement by 100 units in the positive direction of the x-axis is performed.

The conditional transition provides branching of the control program according to the criterion of execution of the conditional expression. The conditional transition operator IF is specified at the beginning of the block, followed by the conditional expression in brackets. The conditional expression must contain at least one comparison operator. If the conditional expression is executed (the value of the conditional expression is true), the control system executes the right part of the block (everything that follows the conditional expression). If the conditional expression fails (the value of the conditional expression is false), control is transferred to the block following the block with the conditional expression.

The following are examples of how the conditional transition is used.

N15 IF [#1 GE 360] THEN G00 Z0;

In block N15 the following condition is checked: «if the value of variable #1 is greater than or equal to 360». If the conditional expression is true, accelerated positioning to the mark z=0 is performed.

The THEN operator after a conditional expression is optional.

N10 #3=20;

N12 IF [#1 LE #2 AND #2 GT 45] GOTO#3;

In block N10, variable #3 is assigned the value 20. In block N12 the following condition is checked: «if the value of variable #1 is less than or equal to the value of variable #2 and the value of variable #2 is greater than 45». If the conditional expression is true, the program moves to block N20.

In the presented example, the transition parameter after the GOTO operator is specified by a variable, but the block number cannot be specified by a variable, i.e. the block to which the transition is performed must contain the address N with an explicit numeric value (N20).

By analogy with the control system «StankoMachComplex» (Tver), the simulator allows the use of multi-line view conditions:

IF […] **{…** **…}** **ELIF […]** **{…** **…}** **ELSE** **{…** **…}**	**IF […]** **{…}** **ELIF […]** **{…}** **ELSE** **{…}**	**IF […] {…}** **ELIF […] {…}** **ELSE {…}**

In this record format, the number of ELIF subconditions is unlimited. The presence of curly braces is mandatory. Any expression or sequence of G-codes in the form of blocks can be specified between curly braces.

The simulator allows you to perform unconditional transitions into the body of multiline conditions. The code execution is performed similarly to the BASIC programming language.

6.34.3. Cycles with Precondition

A loop with a precondition is programmed using the syntactic construct WHILE...DO...END. The WHILE operator is specified at the beginning of the block, followed by a conditional expression in brackets. The conditional expression must contain at least one comparison operator. The conditional expression is followed by the DOn loop start operator, where n is the integer index of the nested loop from 1 to 3. This block is followed by the blocks executed in the loop. The loop ends with the ENDn operator, where n is the index corresponding to the loop start index.

If the conditional expression is executed (the value of the conditional expression is true), control is transferred to the block following the loop start block (DO). If the conditional expression is not executed (the value of the conditional

expression is false), control is transferred to the block following the end of the loop block (END).

The DO...END loop is executed as long as the conditional expression is true.

Below is an example of using a loop with a precondition.

```
N01 #1=300;
N02 #2=200;
N03 WHILE [#1 GT #2] DO1;
N04 #1=#1-10
N05 U100;
N06 W-100;
N07 U-100;
N08 W100;
N09 END1;
N10 M30;
```

In block N1, variable #1 is assigned the value 300. In block N2, variable #2 is assigned the value 200. Block N3 starts the loop with a precondition. The WHILE operator is followed by a conditional expression: «if the value of variable #1 is greater than the value of variable #2», in case of the truth of which the control is transferred to block N4. In block N4, the value of variable #1 is decremented by 10 (it happens at each iteration of the loop). In blocks N5, N6, N7 and N8, the tool moves along the square contour in the x-z relative positioning mode. Block N9 ends the cycle with the END statement. The DO...END cycle is executed 10 times in a row. At the last iteration of the loop, the value of variable #1 is 200, so the loop condition is not satisfied and control is transferred to block N10, which terminates the program execution with code M30.

The DO...END loop can be programmed without the WHILE precondition. Up to three nested loops may be used, but the indexing order of DO and END statements must be observed.

CONTENT