NCT®

Milling Machine and Machining Center Controls

Collection of Part Program Examples

Produced and developed by **NCT Ipari Elektronikai kft**.
H1148 Budapest Fogarasi út 7
⊠ B.O. Box: H1631 Bp. pf.: 26
☎ Phone: (+36 1) 467 63 00
☎ Fax:(+36 1) 363 6605
E-mail: nct@nct.hu
Home page: www.nct.hu

Table of Contents

06.07.19

1 Programming Circular Interpolation with Positioning

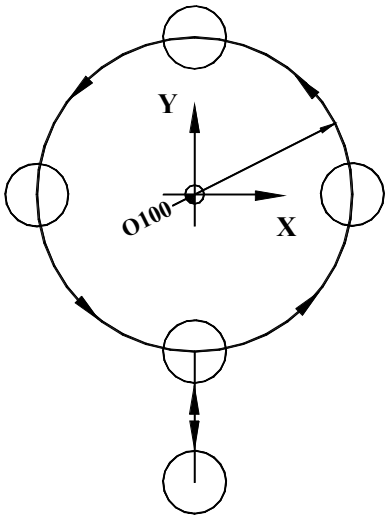


Diagram 1

```
%O7011(1.1)
N100 T1
N110 G54 G0 X0 Y-100
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G0 Z5
N150 G1 Z-10 F20
N160 G1 X0 Y-50 F50
N170 G3 J50
N180 G1 X0 Y-100
N190 G0 Z100
N200 M30
%
```

The first block contains the selection of tool needed. The second block includes orientation in plane X-Y in the appropriate coordinate system. The third block is responsible for taking tool length compensation into account. Other program examples of the collection is shown with similar program start, therefore they are not discussed further on. The program start also contains a percent symbol as well as four digits after letter O, which is the program identifier. The program is finished also with a percent symbol. These characters are to be indicated only when programming on external device, otherwise the control creates them automatically. Tool path definition is started from the fourth block. Programming of the whole circle is done with the help of address J, which is the incremental definition of circle center according to circle start position. Program is closed with code M30.

2 Programming Circular Interpolation with Contour Tracking

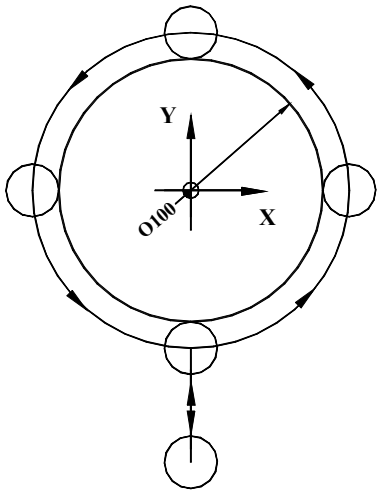


Diagram 2

```
%O7012(1.2)
N100 T1
N110 G54 G0 X0 Y-100
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G0 Z5
N150 G1 Z-10 F20
N160 G1 G42 X0 F50 D1
N170 G3 J-50
N180 G1 G40 Y-100
N190 G0 Z100
N200 M30
%
```

The difference from previous example is the contour tracking. Tracking is to be activated in block N160. In this case the control modifies the block end position so that the next block is moved by the referred tool diameter at start. Stop functions similarly, in this case compensation is 0 at the end position of block (N180).

3 Programming Circular Interpolation with Positioning in Tangential Direction

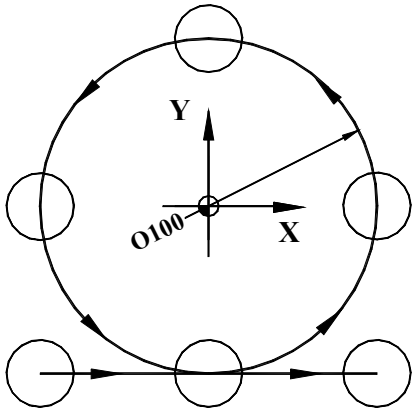


Diagram 3

```
%O7013(1.3)
N100 T1
N110 G54 G0 X-50 Y-50
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G0 Z5
N150 G1 Z-10 F20
N160 G1 X0 F50
N170 G3 J50
N180 G1 X50
N190 G0 Z100
N200 M30
%
```

The difference from previous example is the positioning Since axes reach their desired position by the use of acceleration-deceleration, thus in the previous example the tool leaves a trace at circle start position. The contour is to be approached in tangential direction in all cases in order to avoid this.

4 Programming Circular Interpolation with Contour Tracking

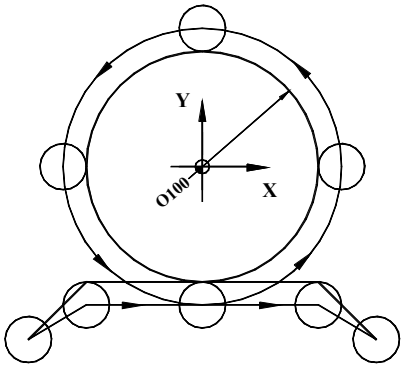


Diagram 4

```
%O7014(1.4)
N100 T1
N110 G54 G0 X-70 Y-70
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G0 Z5
N150 G1 Z-10 F20
N160 G42 X-50 Y-50 D1 F50
N170 G1 X0
N180 G3 J50
N190 G1 X50
N200 G1 G40 X70 Y-70
N210 G0 Z100
N220 M30
%
```

The difference from previous example is the start position as well as the positioning on and off the contour. Coordinates of positioning on and off must be programmed so that movement without compensation is larger than tool radius.

5 Circular Interpolation on Internal Contour

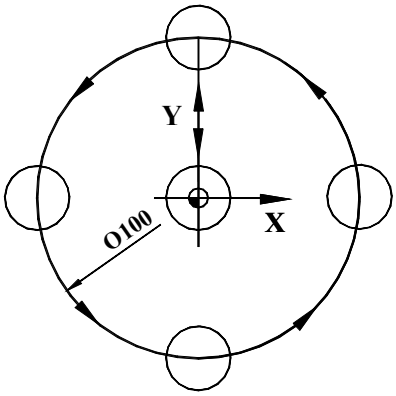


Diagram 5

```
%O7021(2.1)
N100 T1
N110 G54 G0 X0 Y0
N120 G43 Z0 H1
N130 S1000 M3 M8
N140 G0 Z5
N150 G1 Z-10 F20
N160 G1 X0 Y50 F50
N170 G3 J-50
N180 G1 X0 Y0
N190 G0 Z100
N200 M30
%
```

In this case, as in example 1.1, the problem arises from rectangular positioning.

6 Inner Circular Interpolation with Contour Tracking

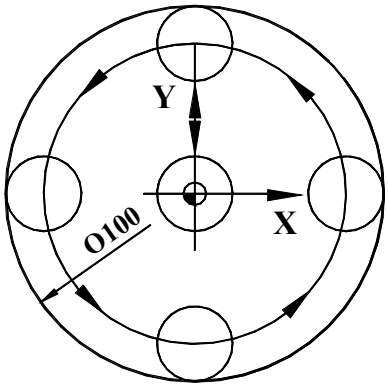


Diagram 6

```
%O7022(2.2)
N100 T1
N110 G54 G0 X0 Y0
N120 G43 Z0 H1
N130 S1000 M3 M8
N140 G0 Z5
N150 G1 Z-10 F20
N160 G1 G41 X0 Y50 F50 D1
N170 G3 J-50
N180 G1 G40 X0 Y0
N190 G0 Z100
N200 M30
%
```

Only G41-G42 change in relation to external contour tracking. G41-G42 replacement could also have been substituted by the replacement of G2-G3. Neither can this process eliminate movement over circle start position.

7 Programming Inner Circular Interpolation with Positioning in Tangential Direction

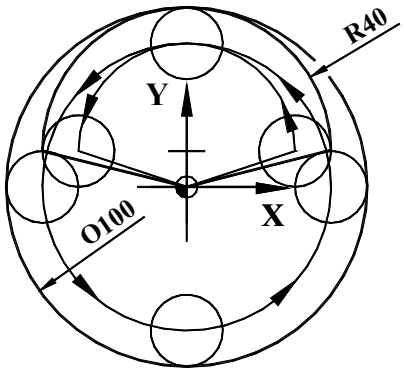


Diagram 7

```
%O7023(2.3)
N100 T1
N110 G54 G0 X0 Y0
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G0 Z5
N150 G1 Z-10 F20
N160 G41 X40 Y10 D1 F50
N170 G3 X0 Y50 R40
N180 G3 J-50
N190 G3 X-40 Y10 R40
N200 G1 G40 X0 Y0
N210 G0 Z100
N220 M30
%
```

In order to avoid surface errors, positioning to inner arc is executed in tangential direction by the use of circular interpolation, as in case of external contour Tangential circle radius is optional, however larger than tool radius and smaller than circle radius. Its center is moved according to original circle center in direction Y with the difference between contour radius and circle radius.

8 Programming Straight Contour

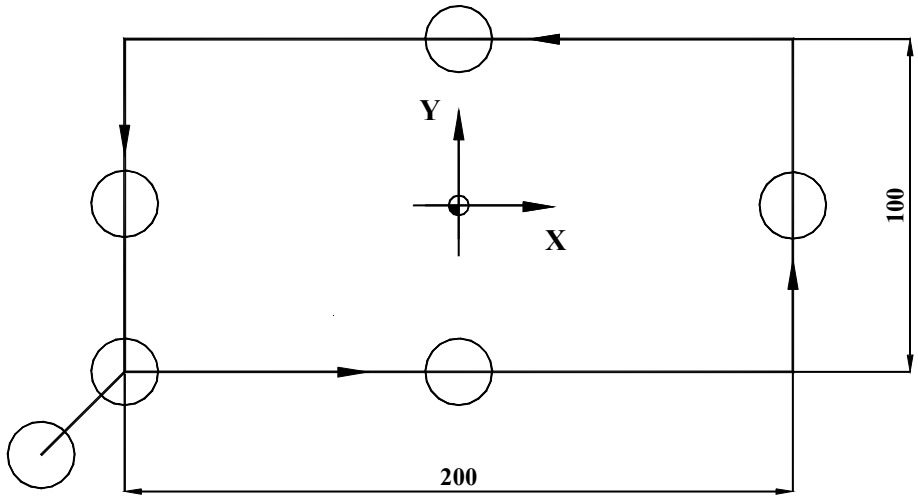


Diagram 8

```
%O7031(3.1)
N100 T1
N110 G54 G0 X-120 Y-70
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G0 Z5
N150 G1 Z-10 F20
N160 G1 X-100 Y-50 F50
N170 G1 X100
N180 G1 Y50
N190 G1 X-100
N200 G1 Y-50
N210 G1 X-120 Y-70
N220 G0 Z100
N230 M30
%
```

Round-milling of a rectangle is the simplest example. G1 codes are repeated only for easier interpretation, the repeated codes are unnecessary in view of the control

9 Programming Straight Contour with Contour Tracking

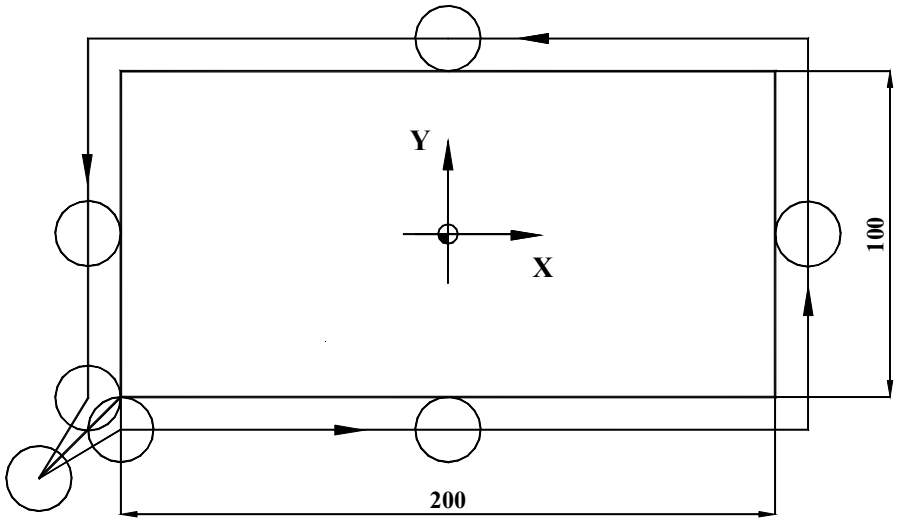


Diagram 9

```
%O7032(3.2)
N100 T1
N110 G54 G0 X-120 Y-70
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G0 Z5
N150 G1 Z-10 F20
N160 G1 G42 X-100 Y-50 F50 D1
N170 G1 X100
N180 G1 Y50
N190 G1 X-100
N200 G1 Y-50
N210 G1 G40 X-120 Y-70
N220 G0 Z100
N230 M30
%
```

The measure of positioning is to be defined, so that is it larger than tool radius, as in example 1.4. In the case drafted on the diagram burr may stay on start position corner, thus it is recommended to overset the program a few mm before the corner when positioning on and after the corner when positioning off.

10 Programming Zero Circles in Case of Contour Tracking

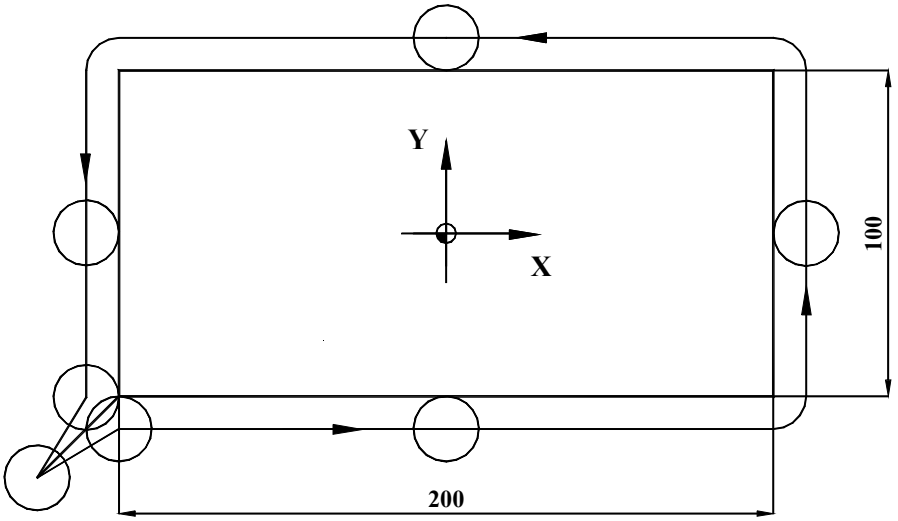
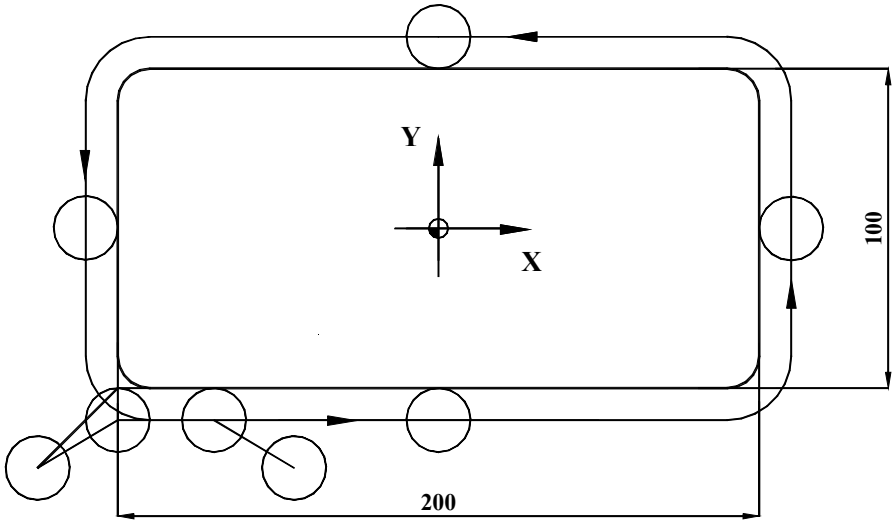


Diagram 10

```
%O7033(3.3)
N100 T1
N110 G54 G0 X-120 Y-70
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G0 Z5
N150 G1 Z-10 F20
N160 G1 G42 X-100 Y-50 F50 D1
N170 G1 X100 ,R0
N180 G1 Y50 ,R0
N190 G1 X-100 ,R0
N200 G1 Y-50
N210 G1 G40 X-120 Y-70
N220 G0 Z100
N230 M30
%
```

,R0 is interpreted by the control as circle arc with zero radius, thus this part is executed as if circle arc during contour tracking. Tool path radius equals to tool radius, while sharp corner remains on contour. This version helps in balanced loading of tool in case of a larger roughing allowance.

11 Contour Tracking with Rounding

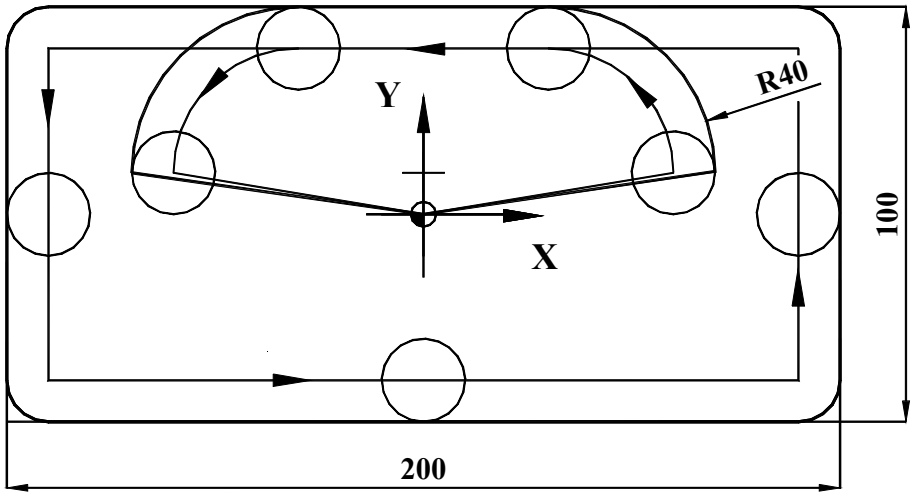


11. ábra

```
%O7034(3.4)
N100 T1
N110 G54 G0 X-120 Y-70
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G0 Z5
N150 G1 Z-10 F20
N160 G1 G42 X-100 Y-50 F50 D1
N170 G1 X100 ,R10
N180 G1 Y50 ,R10
N190 G1 X-100 ,R10
N200 G1 Y-50 ,R10
N210 G1 X-70
N220 G1 G40 X-50 Y-70
N230 G0 Z100
N240 M30
%
```

In order to round the fourth corner the tool must return to the first edge or at least to a part of it. This distance must be larger than the sum of rounding and tool radius. In case of rounding the tool path radius corresponds to the sum of rounding and tool radius.

12 Rectangle with Inner Contour Tracking and with Positioning in Tangential Direction



12. ábra

```
%O7041(4.1)
N100 T1
N110 G54 G0 X0 Y0
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G0 Z5
N150 G1 Z-10 F20
N160 G1 G41 X80 Y10 F50 D1
N170 G3 X40 Y50 R40
N180 G1 X-100
N190 G1 Y-50
N200 G1 X100
N210 G1 Y50
N220 G1 X-40
N230 G3 X-80 Y10 R40
N240 G1 G40 X0 Y0
N250 G0 Z100
N260 M30
%
```

As in example 3.2, positioning is done round circle arc, the radius of which is larger than tool radius, but smaller than half of the hole size in direction Y.

13 Programming Optional Contour with Contour Tracking

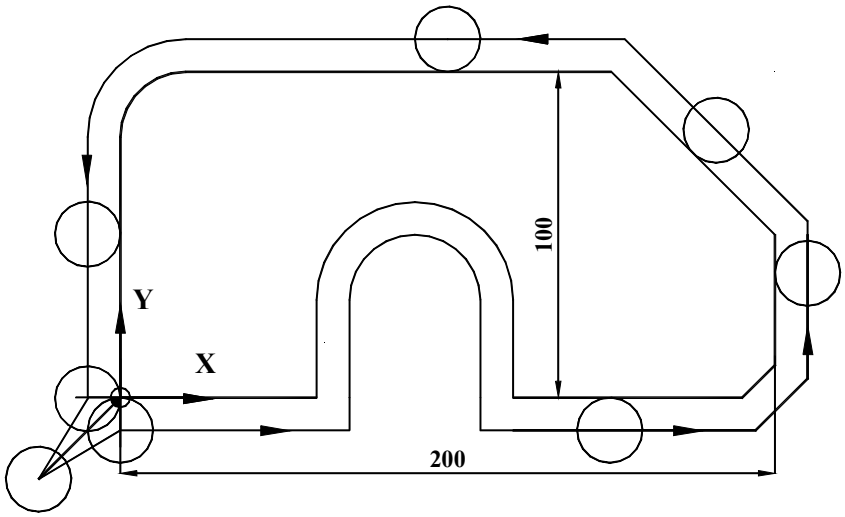


Diagram 13

```
%O7051(5.1)
N100 T1
N110 G54 G0 X-40 Y-40
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G0 Z5
N150 G1 Z-10 F20
N160 G1 G42 X0 Y0 F50 D1
N170 G1 X50
N180 G1 Y30
N190 G2 X100 R25
N200 G1 Y0
N210 G1 X200 ,C10
N220 G1 Y50
N230 G1 X150 Y100
N240 G1 X0 ,R20
N250 G1 Y0
N260 G1 G40 X-30 Y-30
N270 G0 Z100
N280 M30
%
```

An optional contour is shown in this example. Line 210 contains break, while line 240 includes rounding. Break is indicated by ,C in line 210, where the control measures the specified distances to straights defined by previous and following lines and these points are linked as resulting contour.

14 Definition of Optional Contour with Inner Contour Tracking

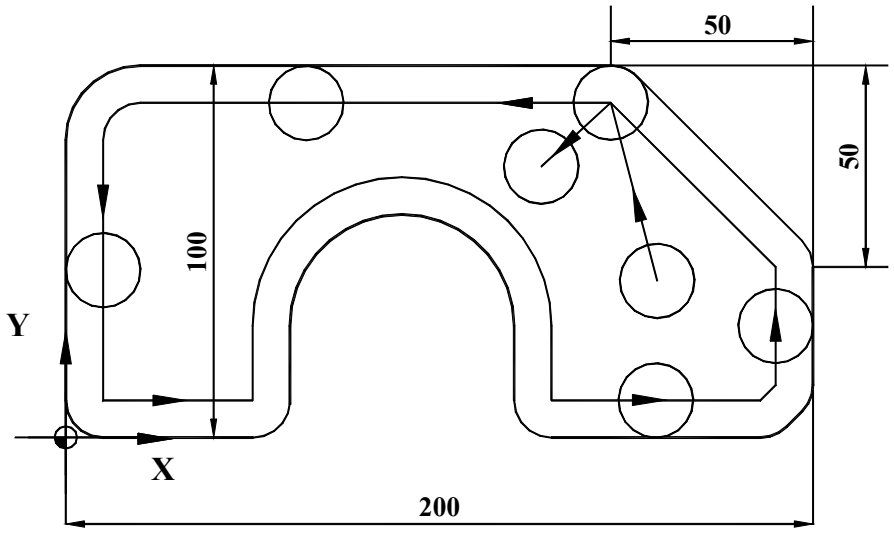


Diagram 14

```
%O7052(5.2)
N100 T1
N110 G54 G0 X160 Y50
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G0 Z5
N150 G1 Z-10 F20
N160 G1 G41 X150 Y100 I-1 J1 F50 D1
N170 G1 X0 ,R20
N180 G1 Y0
N190 G1 X50
N200 G1 Y30
N210 G2 X100 R25
N220 G1 Y0
N230 G1 X200 ,C10
N240 G1 Y50
N250 G1 X150 Y100
N260 G1 G40 XI-20 YI-20 I-1
N270 G0 Z100
N280 M30
%
```

Positioning to inner contour is done at address G41 I_ J_ in block N160. The direction vector of previous block can be specified at addresses. I and J, in this case the control positions to contour so that it is not the straight defined by the positioning block but that of the specified direction vector that is taken into account when calculating coordinates. This method works similarly in case of stop. Only direction tangents can be specified at addresses I and J, therefore it is only the sign and ratio that count, not the absolute value.

15 Optional Inner Contour (Subprogram)

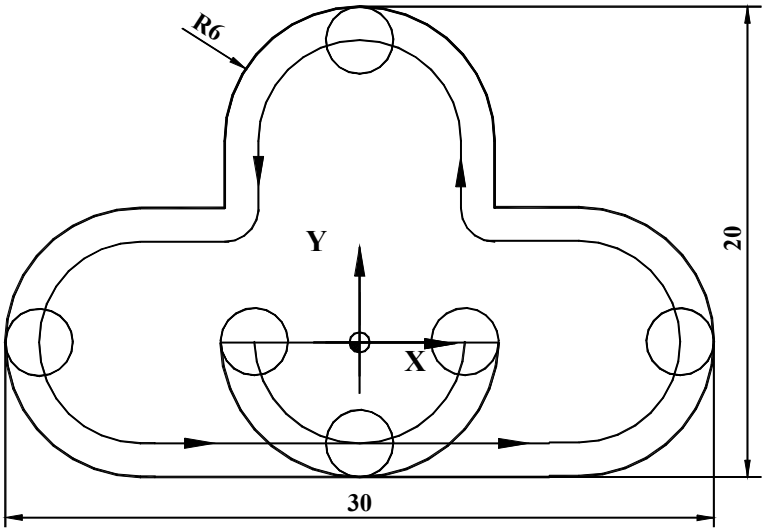


Diagram 15

```
%O7053(5.3)
N100 T1
N110 G54 G0 X0 Y0
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G0 Z5
N150 G1 Z-10 F20
N160 G1 G41 X-6 F50 D1
N170 G3 X0 Y-6 R6
N180 G1 X9
N190 G3 Y6 R6
N200 G1 X6
N210 G1 Y14
N220 G3 X-6 R6
N230 G1 Y6
N240 G1 X-9
N250 G3 Y-6 R6
N260 G1 X0
N270 G3 X6 Y0 R6
N280 G1 G40 X0 Y0
N290 G0 Z100
N300 M99
%
```

The program is written as subprogram, this is signaled by M99 instead of M30 at program end. If it is started as main program, it runs as endless cycle. This program is used later on as subprogram.

16 Subprogram Technique with Zero Position Offset

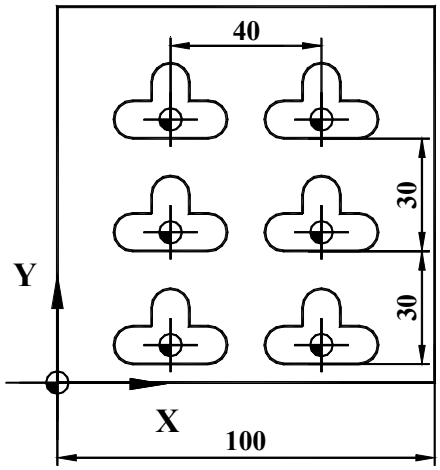


Diagram 16

```
%O7061(6.1)
N100 T1
N110 G54
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G52 X30 Y10
N150 M98 P7053
N160 G52 X70 Y10
N170 M98 P7053
N180 G52 X30 Y40
N190 M98 P7053
N200 G52 X70 Y40
N210 M98 P7053
N220 G52 X30 Y70
N230 M98 P7053
N240 G52 X70 Y70
N250 M98 P7053
N260 G52 X0 Y0
N270 G0 Z100
N280 M30
%
```

The subprogram is prepared with absolute size specification with zero position drafted on the diagram. The subprogram is detailed in example 5.3. A local coordinate-system offset and a subprogram call are included by pairs in the main program. Subprogram is closed with command M99.

17 Use of Increment with Subprogram Call

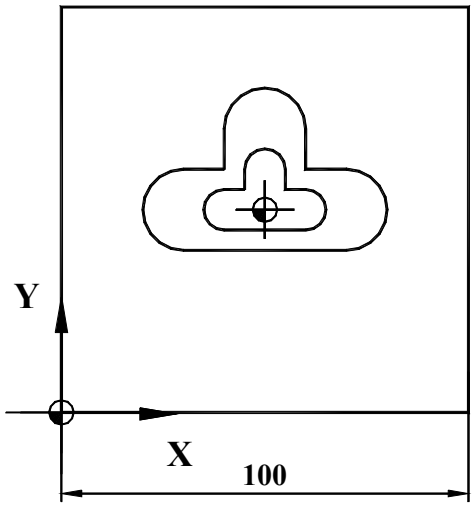


Diagram 17

```
%O7062(6.2)
N100 T1
N110 G54
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G52 X50 Y50
N150 G51 X0 Y0 P2
N160 M98 P7053
N170 G50 X0 Y0
N180 G52 X0 Y0
N190 G0 Z100
N200 M30
%
```

G52 is responsible for activating local coordinate-system, while G51 is responsible for activating increment. The control enlarges the following movements with the value given at address P around coordinates specified in block G51 as center. Contour definition is also executed in subprogram.

18 Programming Increment around Optional Position

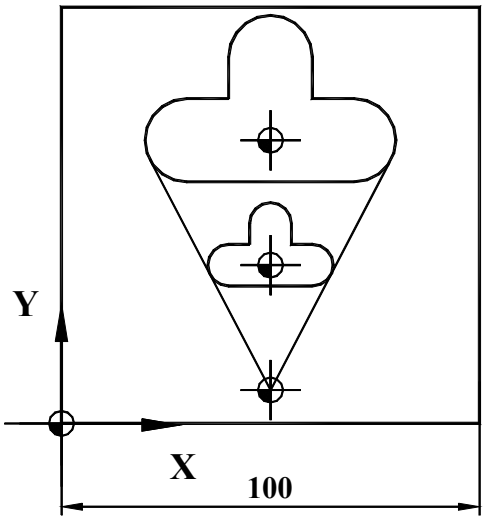


Diagram 18

```
%O7063(6.3)
N100 T1
N110 G54
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G52 X50 Y50
N150 G51 X0 Y-40 P2
N160 M98 P7053
N170 G50 X0 Y0
N180 G52 X0 Y0
N190 G0 Z100
N200 M30
%
```

The specified coordinates must be given in block G51 according to G52 zero position offset, this is why -40 is at address Y. In this case zero position is also offset in relation to the position specified in block G51 as central enlarging position.

19 Programming a Series of Bores

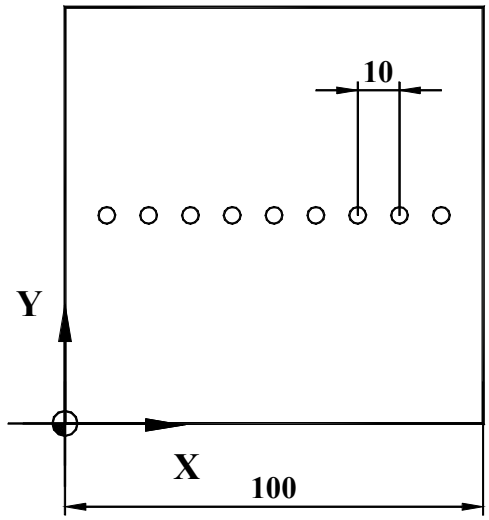


Diagram 19

```
%O7071(7.1)
N100 T1
N110 G54 G0 X0 Y50
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G81 XI10 Y50 R2 Z-10 L9
N150 G80
N160 G0 Z100
N170 M30
%
```

The programming of bores 10mm apart from each other is defined incrementally with repetition. In this case positioning has to be done one orientation block **ahead** of the first bore, since the positions were not defined in absolute value. The cycle is started by moving the increment distance, which is followed by drilling and this is repeated with the value given at address L. Drilling cycle is closed with code G80.

20 Programming Bore Series

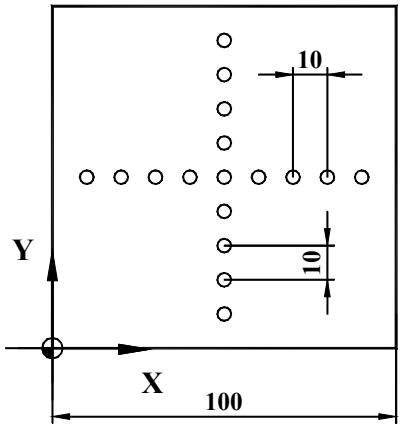


Diagram 20

```
%O7072(7.2)
N100 T1
N110 G54 G0 X0 Y50
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G81 XI10 Y50 R2 Z-10 L9
N150 G80
N160 G0 X50 Y0
N170 G81 X50 YI10 R2 Z-10 L4
N180 G80
N190 G0 X50 Y50
N200 G81 X50 YI10 R2 Z-10 L4
N210 G80
N220 G0 Z100
N230 M30
%
```

The second series of bores is programmed as a new drilling cycle, where size is again to be specified in increment. Since both cycles would include the middle bore, the second series must be split into lower and upper part.

21 Programming Center Circle

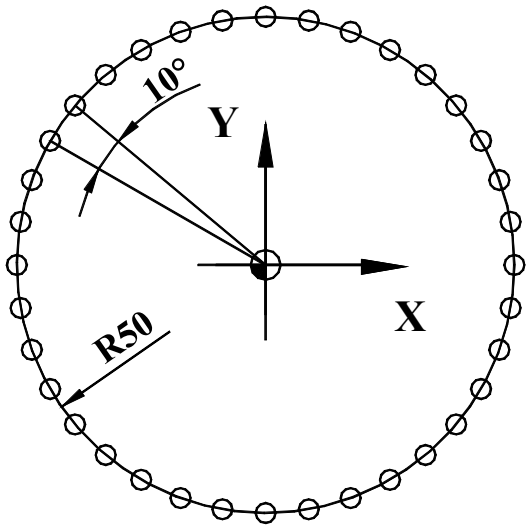


Diagram 21

```
%O7073(7.3)
N100 T1
N110 G54 G0 X0 Y0
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G16 G0 X50 Y-10
N150 G81 X50 YI10 R2 Z-10 L36
N160 G80 G15
N170 G0 Z100
N180 M30
%
```

Programming center circle differs from bore series in the use of polar coordinate data input. Orientation is also executed ahead of the first bore, however this has no significance in case of whole circle.

22 Programming Center Circle Part

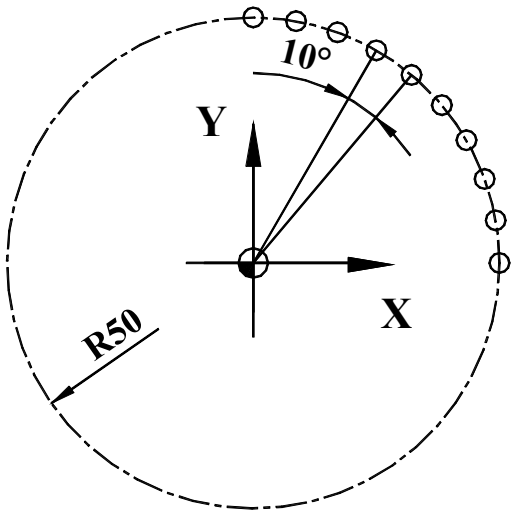


Diagram 22

```
%O7074(7.4)
N100 T1
N110 G54 G0 X0 Y0
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G16 G0 X50 Y-10
N150 G81 X50 YI10 R2 Z-10 L10
N160 G80 G15
N170 G0 Z100
N180 M30
%
```

Programming center circle part differs from programming center circle in that the multiplication of repetition number and angle difference does not result in 360 degrees. Orientation is also executed ahead of the first bore.

23 Programming Center Circle to Optional Position

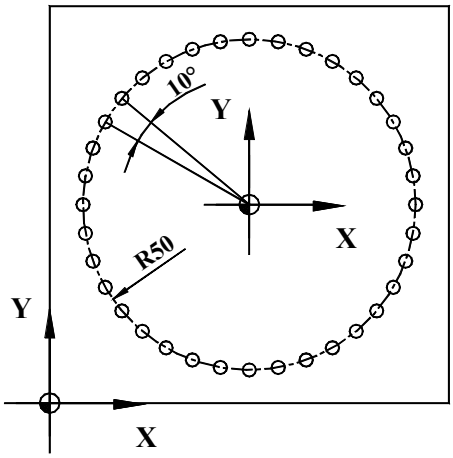


Diagram 23

```
%O7075(7.5)
N100 T1
N110 G54 G0 X0 Y0
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G52 X60 Y60
N150 G16 G0 X50 Y-10
N160 G81 X50 YI10 R2 Z-10 L36
N170 G80 G15
N180 G52 X0 Y0
N190 G0 Z100
N200 M30
%
```

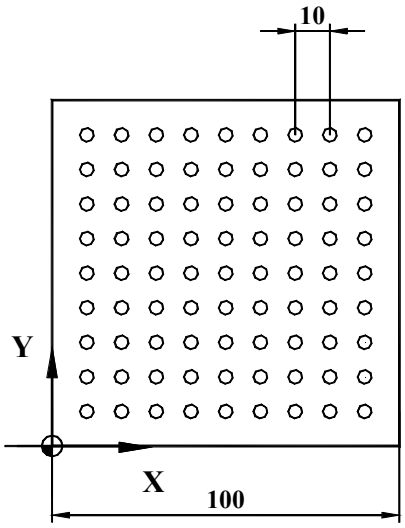
In this case work zero position is not located in the middle of work but on a corner. In such case local coordinate system (G52) must be programmed, while the further program part corresponds to the previous one. Local coordinate system must be inactivated at the end of cycle, otherwise further coordinates given in absolute values will be calculated from this zero position.

24 Programming a Series of Bores as Subprogram

```
%O7076(7.6)
N100 G81 YI10 R2 Z-10 L9
N110 G80
N120 G0 XI10 Y0
N130 M99
%
```

The programming of the series of bores is similar to the previous example, the difference is that it also contains X positioning at the end of series. Thus a bore net is resulted from the call with a given number from the main program.

25 Programming Bore Net with the Help of Subprogram



24. ábra

```
%O7077(7.7)
N100 T1
N110 G54 G0 X10 Y0
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 M98 P7076 L9
N150 G0 Z100
N160 M30
%
```

This task can be accomplished in more different ways. The simplest method is to program horizontal bore series as subprogram and the subprogram is called more times. There is a zero position offset in subprogram which must be inactivated at the end of main program. Percent symbol as well as number and name of program is shown at the start and end of program for the sake of easier separation of program parts.

26 Programming Bore Net with Cycle

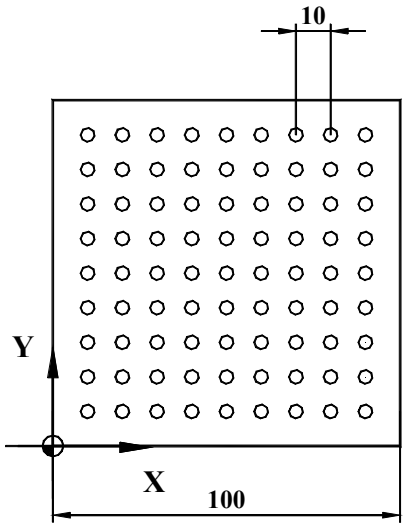


Diagram 26

```
%O7078(7.8)
N100 T1
N110 G54 G0 X10 Y0
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 #1=1
N150 WHILE[#1LE9] DO1
N160 G0 X10 Y0
N170 G81 YI10 R2 Z-10 L9
N180 G80
N190 G52 XI10 Y0
N200 #1=#1+1
N210 END1
N220 G52 X0 Y0
N230 G0 Z100
N240 M30
%
```

The second method is to organize vertical bore series into an internal cycle. In this case subprogram is not needed but so called macro variables must be adopted. These macro variables must be defined with value input and can be used optionally. The cycle is characterized by command WHILE. Cycle start is signaled with DO1, its end is signaled with END1, where numbers indicate inherence.

27 Programming Bore Net with Two Cycles Embedded in Each Other

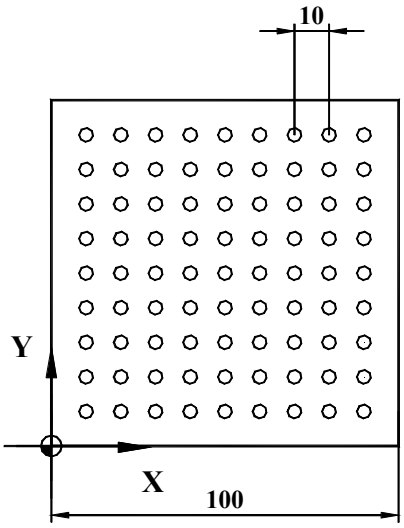


Diagram 27

```
%O7079(7.9)
N100 T1
N110 G54 G0 X10 Y0
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 #1=10
N150 WHILE[#1LE90] DO1
N140 #2=10
N150 WHILE[#2LE90] DO2
N170 G81 X#1 Y#2 R2 Z-10
N200 #2=#2+10
N210 END2
N200 #1=#1+10
N210 END1
N220 G80
N230 G0 Z100
N240 M30
%
```

In the third case when there are two cycles embedded in each other, macro variables are related to X and Y coordinates of bores. Thus the initial value of macro variable is set to the first coordinate, while the end value is set to the last coordinate. Increase in macro variables is executed with the difference between coordinates.

28 Automatic Geometric Calculation between Circle Arcs (Inner Contour)

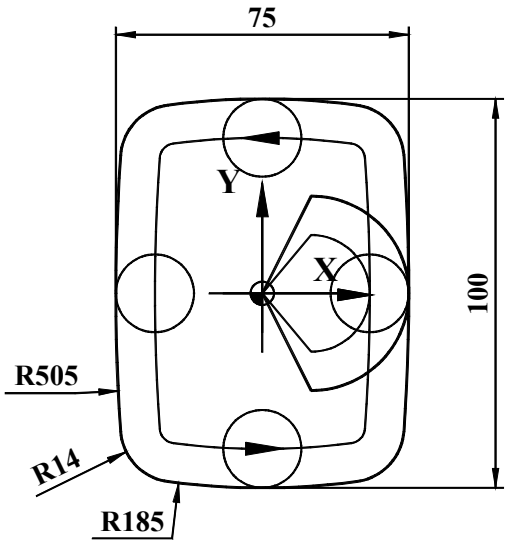


Diagram 28

```
%O7081(8.1)
N100 G54 G90 G17 G0
N110 T1
N120 G43 Z50 H1
N130 S1000 M3
N140 G0 X0 Y0
N150 G0 Z2
N160 G1 Z-19
N170 G0 X0 Y0
N180 G41 G1 X17.5 Y-20 D1
N190 G3 X37.5 Y0 R20
N200 G3 XI-505 YI505 R505 ,R14
N210 G3 I0 J-135 R185 Q-1 ,R14
N220 G3 I467.5 J0 R505 Q-1 ,R14
N230 G3 I0 J135 R185 Q-1 ,R14
N240 G3 X37.5 Y0 I-467.5 J0 R505 Q-1
N250 G3 X17.5 Y20 R20
N260 G1 G40 X0 Y0
N270 Z50
N280 M30
%
```

29 Automatic Geometric Calculation between Circle Arcs (Outer Contour)

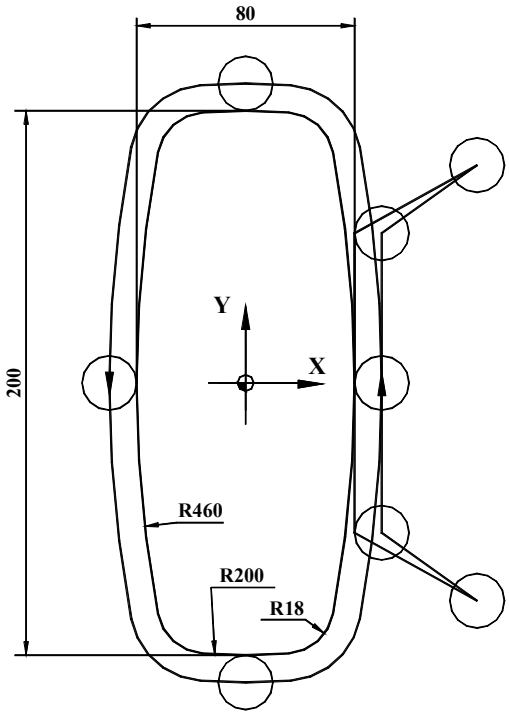


Diagram 29

```
%O7082(8.2)
N100 G54 G90 G17 G0
N110 T1
N120 G43 Z50 H1
N130 S1000 M3
N140 G0 X0 Y0
N150 G0 Z2
N160 G1 Z-19
N170 G0 X90 Y-70
N180 G42 G1 X40 Y-50 D1
N190 G1 Y0
N200 G3 XI-460 YI460 R460 ,R18
N210 G3 I0 J-100 R200 Q-1 ,R18
N220G3 I420 J0 R40 Q-1 ,R18
N230 G3 I0 J100 R200 Q-1 ,R18
N240  G3  X40  Y0  I-420  J0  R460  Q-1
,R18
N260 G1 X40 Y50
N270 G1 G40 X90 Y70
N280 Z50
N290 M30
%
```

30 Automatic Geometric Calculation between Circle Arcs (Inner Contour)

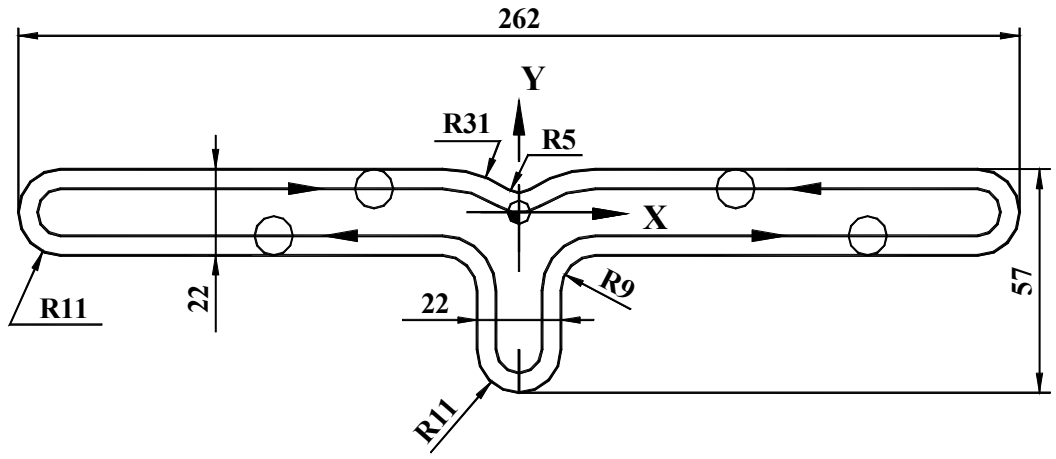


Diagram 30

```
%O7083(8.3)
N100 G17 G0 G90 G94
N110 T1
N120 G0 X0 Y0
N130 G43 Z50 H1
N140 S1000 M3
N150 G0 Z-50
N160 G0 X0 Y-35
N170 G1 G41 Y-46 D1
N180 G3 X11 Y-35 R11
N190 G1 Y-11 ,R9
N200 G1 X120
N210 G3 X120 Y11 R11
N220 G1 X20
N230 G3 X-11 Y-20 R31 ,R5
N240 G3 X-20 Y11 I-20 J-20 R31 Q1
N250 G1 X-120
N260 G3 Y-11 R11
N270 G1 X-11 ,R9
N280 G1 Y-35
N290 G3 X0 Y-46 R11
N300 G1 G40 Y-35
N310 G0 X0 Y0
N320 Z50
N330 M30
%
```

In case end milling cutter 22 is used, radiuses 11 are embodied by the milling cutter, thus the two-direction milling path coincide. In this case tangential approach is disregarded when positioning.

31 Programming Cogging - Subprogram

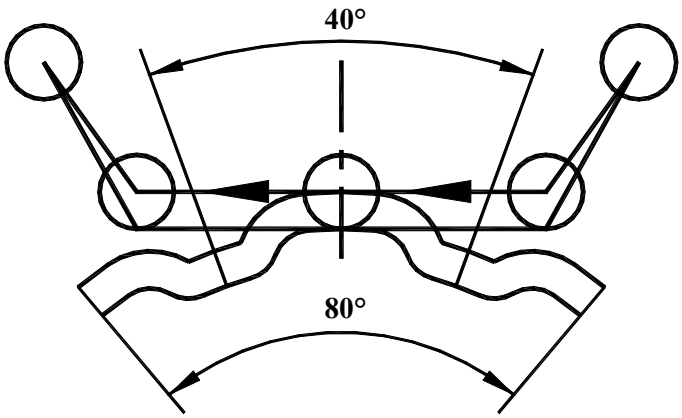


Diagram 31

```
%O7091(9.1)
N100 G1 X0 Y50
N110 G3 X-4.587 Y49.789 R50
N120 G3 X-8.846 Y46.468 R5
N130 G2 X-12.206 Y43.313 R5
N140 G3 X-15.391 Y42.286 R45
N150 G3 X-18.490 Y41.026 R45
N160 G2 X-23.093 Y41.282 R5
N170 G3 X-28.490 Y41.090 R5
N180 G3 X-32.139 Y38.302 R50
N190 G68 X0 Y0 RI40
N190 M99
%
```

In this example coordinates of a cog is defined in subprogram. Unfortunately as generally in case of cogging, the cog form cannot be defined with simple geometric calculations, therefore the form coordinates have to be calculated with a CAD/(CAM) system. The incremental rotating of coordinate system is realized in line G68.

32 Programming Cogging - Main Program

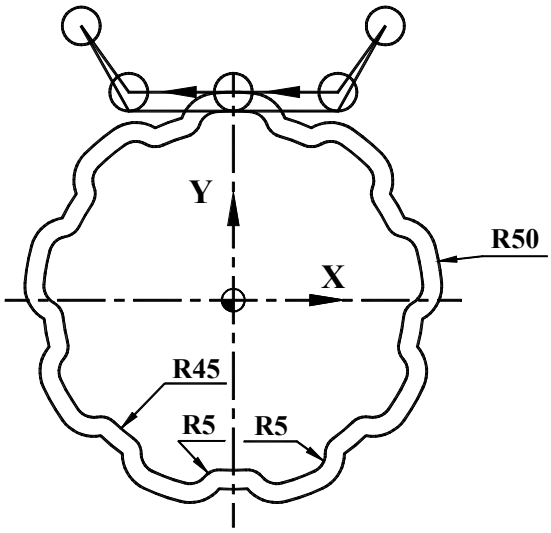


Diagram 32

```
%O7092(9.2)
N100 T1
N110 G54 G0 X40 Y70
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G0 G42 X30 Y50 D1
N150 G1 X0 Y50
N180 M98 P7091 L9
N220 G69
N230 G1 X-30 Y50
N240 G0 G40 X-40 Y70
N250 G0 Z100
N260 M30
%
```

In this program the subprogram is called as many times as is the number of cogs. At the end of program the inactivating of coordinate system rotation as well as the leaving of contour is needed.

33 Programming Cogging with a Program

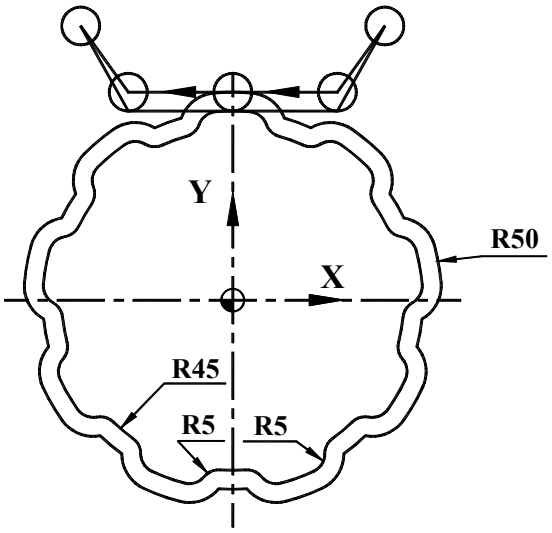


Diagram 33

```
%O7093(9.3)
N100 T1
N110 G54 G0 X40 Y70
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G0 G42 X30 Y50 D1
N150 G1 X0 Y50
N160 #1=0
N170 WHILE[#1LT9] DO1
N180 G68 X0 Y0 R[#1*40]
N190 G1 X0 Y50
N200 G3 X-4.587 Y49.789 R50
N210 G3 X-8.846 Y46.468 R5
N220 G2 X-12.206 Y43.313 R5
N230 G3 X-15.391 Y42.286 R45
N240 G3 X-18.490 Y41.026 R45
N250 G2 X-23.093 Y41.282 R5
N260 G3 X-28.490 Y41.090 R5
N270 G3 X-32.139 Y38.302 R50
N280 #1=#1+1
N290 END1
N300 G69
N310 G1 X-30 Y50
N320 G40 X-40 Y70
N330 G0 Z100
N340 M30
%
```

Cog from can be embedded in the program with the help of macro variables and by the use of an internal cycle, as it is also possible to define rotation in absolute value. The ready contour corresponds to the previous example in every aspect.

34 Programming Center Circle with Mirroring - Subprogram

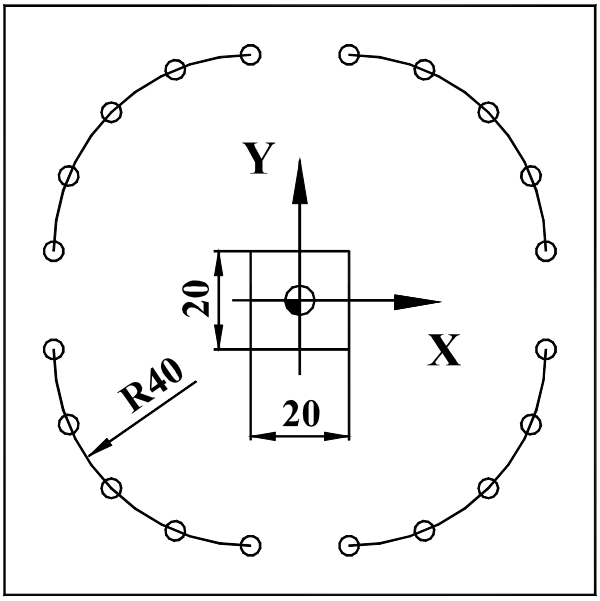


Diagram 34

```
%O7101(10.1)
N100 T1
N110 G54 G0 X0 Y0
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G52 X10 Y10
N150 M98 P7102
N160 G51.1 Y0
N170 M98 P7102
N180 G51.1 X0
N190 M98 P7102
N200 G50.1 Y0
N210 M98 P7102
N220 G50.1 X0
N230 G52 X0 Y0
N240 G0 Z100
N250 M30
%
```

In this case work zero position is not in the middle of work but on a corner. In this case local coordinate system (G52) must be programmed, afterwards the program corresponds to the previous example with the difference, that the offset is programmed in four cycles. The local coordinate system must be inactivated at the end of cycles, otherwise further coordinates given in absolute value are also calculated from this zero position.

35 Programming Center Circle with Mirroring - Subprogram

```
%O7102(10.2)
N140 G16 G0 X40 Y-10
N150 G81 X50 YI10 R2 Z-10 L5
N160 G80 G15
N170 G0 Z100
N180 M99
%
```

The main program shown in the previous example needs this subprogram for the correct operation. By the way this program almost completely corresponds to the example discussed in chapter 22. on page 26.

36 Programming Macro - Sinusoidal Curve

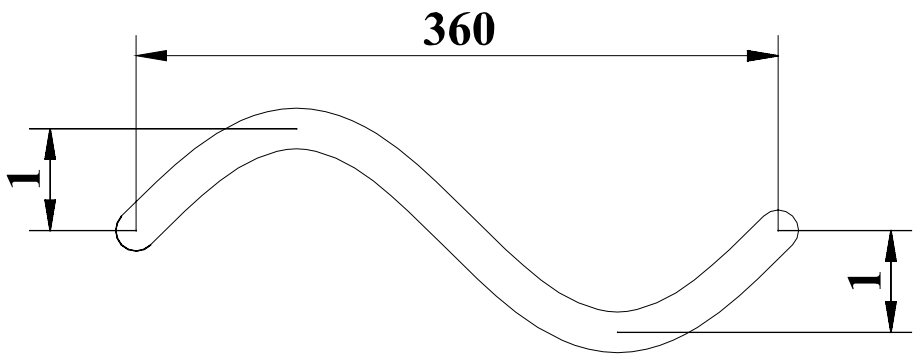


Diagram 35

```
%O7111(11.1)
N100 T1
N110 G0 G90 X0 Y0
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G1 Z-10
N150 #1=0
N160 WHILE[#1LE360] DO1
N170 #2=SIN#1
N180 G1 X#1 Y#2 F100
N190 #1=#1+0.1
N200 END1
N210 G0 Z50
N220 M30
%
```

In this example programming of a simple sinusoidal curve is shown with the help of macro variables. The circle is started at the origin, its longitudinal axis is parallel to axis X. To make it simple, tool compensation is disregarded. #1 is the so called running variable, which is 0<#1<360. (Since NC control belongs to the world of engineering, not that of mathematics, naturally degrees are measured in angle not in radian.) Variable #2 can be calculated from this with the help of the sinus function. Afterwards assignment of #1 to coordinate X and of #2 to coordinate Y is simple. This, linked in a cycle, results in the sinusoidal curve.

In case the start position of the needed curve is not the origin or if its amplitude is not 1 unit or its length is not 360 mm, line N180 can be modified as follows:

N180 G1 X[A+[#1*B]] Y[C+[#2*D]] F100,

where A and C are the offset values (X;Y), B is the length and D is the amplitude.

37 Programming Macro - Circular Milling, Cylindrical Interpolation

```
%O7112(11.2)
N100 T1
N110 G43 Z50 H1
N120 S1000 M3 M8
N130 G0 X50 Y0
N140 #1=0
N150 WHILE[#1LE360] DO1
N160 #2=50*SIN[#1]
N170 #3=50*SIN[#1]
N180 G1 X#2 Y#3 Z[50-#1]
N190 #1=#1+1
N200 END1
N210 G0 Z50
N220 M30
%
```

In this example the task is to program a spacial ellipse, plane XY of which is a circle. In this case actually a circle is programmed, but meanwhile position Z is calculated from current position X. In order to do this the circle programming must be parametric. Circle can simplest be defined with relation X=R*cosα and Y=R*sinα. Coordinate Z can be calculated easily therefrom. The running variable is the central angle.

38 Programming Macro - Hemisphere

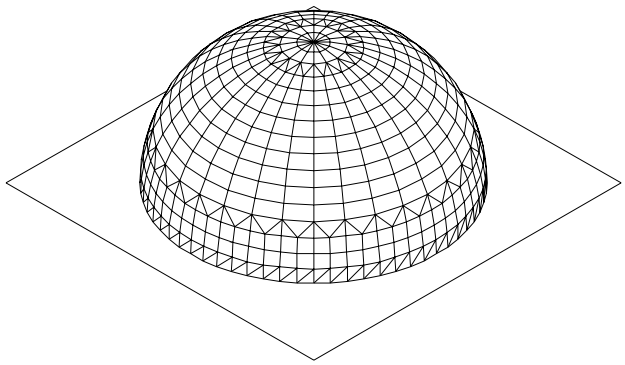


Diagram 36

```
%O7118(11.8)
N100 T1
N110 G54 G0 X0 Y0
N120 G43 Z50 H1
N130 S1000 M3 M8
N140 G0 X-50 Y-50 Z0
N150 #1=-50
N160 WHILE[#1LE50] DO1
N170 #2=-50
N180 WHILE[#2LE50] DO2
N190 #3=0
N200 IF[[[#1*#1]+[#2*#2]]GT1600] GOTO220
N210 #3=SQRT[1600-[#1*#1]-[#2*#2]]
N220 G1 X#1 Y#2 Z#3 F1000
N230 #2=#2+1
N240 END2
N250 #1=#1+1
N260 IF[#1GE50]GOTO370
N270 #2=50
N280 WHILE[#2GE-50] DO2
N295 #3=0
N300 IF[[[#1*#1]+[#2*#2]]GT1600] GOTO320
N310 #3=SQRT[1600-[#1*#1]-[#2*#2]]
N320 G1 X#1 Y#2 Z#3
N330 #2=#2-1
N340 END2
N350 #1=#1+1
N360 END1
N370 M30
%
```

On the diagram an R40 hemisphere can be seen on a 100x100 plane. Its tool path defined with macro variables is shown in the following program. Programs written with such technique have the disadvantage that the real feed cannot be effective because of the extremely many calculations, but the processor time is decisive. This can be recognized, since the tool stops “to think” between movements. The speed can be accelerated/decelerated with the fining/roughing of resolution - lines N230, N250, N330 and N350.

39 Programming Macro - Hemisphere Creation

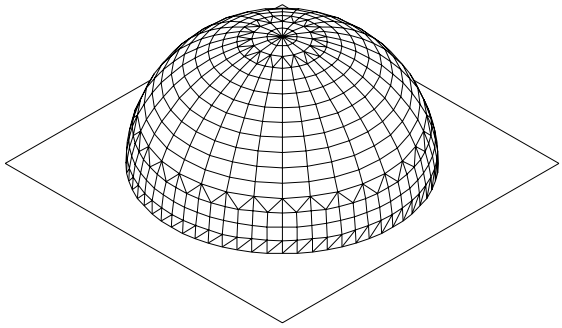


Diagram 37

```
%O7119(11.9)
N100 POPEN31
N110 DPRNT[O7120(FELGOMB)]
N120 DPRNT[T1]
N130 DPRNT[G54 G0 X-50 Y-50]
N140 DPRNT[G43 Z50 H1]
N150 DPRNT[G90 G01 S1000 M3 M8 F1000]
N160 #1=-50
N170 WHILE[#1LE50] DO1
N180 #2=-50
N190 WHILE[#2LE50] DO2
N200 #3=0
N210 IF[[[#1*#1]+[#2*#2]]GT1600] GOTO230
N220 #3=SQRT[1600-[#1*#1]-[#2*#2]]
N230 DPRNT[G1 X#1[53]Y#2[53]Z#3[53]]
N240 #2=#2+1
N250 END2
N260 #1=#1+1
N270 IF[#1GE50]GOTO370
N280 #2=50
N290 WHILE[#2GE-50] DO2
N300 #3=0
N310 IF[[[#1*#1]+[#2*#2]]GT1600] GOTO330
N320 #3=SQRT[1600-[#1*#1]-[#2*#2]]
N330 DPRNT[G1 X#1[53]Y#2[53]Z#3[53]]
N340 #2=#2-1
N350 END2
N360 #1=#1+1
N370 END1
N380 PCLOS31
N390 M30
%
```

On the diagram, as in the previous example, an R40 hemisphere can be seen on a 100x100 plane This example shows how to create a technological program without macro variables, capable of quick run. Such technological program has the advantage that the extremely many calculations are skipped under run and the real feed is effective and it is not the processor time that decides. The program creation speed changes with the fining/roughing of resolution - lines N240, N260, N340 and N360 -,while the machining speed remains unchanged. A significant difference to the previous example is the periphery handling. Opening and closing of periphery is important, otherwise the program written in backing storage can be damaged. It is hard to estimate the size of prepared program, therefore it is recommended to avoid direct storage writing and to use serial periphery instead.