NUM 720 F

Programming Manual

En-938702/D

MOD. 10-92
ED. 09-88

www.num.com

CNC HighEnd Applications

NUM 720F – 938702

October 1992

NUM 720 F

PROGRAMMING MANUAL

AMENDMENT LIST D

Conforms to software 720 F N° 201291 index F

IMPORTANT :

Some functions described in this document may well not be offered by your system, if the particular option does not form part of the installated configuration, or if the function does not exist in that version of the product (cf version E in particular).

Modified pages :

0.1–0.4
2.1–2.10
4.2, 4.3, 4.9
7.8
8.2, 8.6
9.4, 9.5, 9.9
10.1, 10.7, 10.8
11.2, 11.20
A1.1, A1.3, A1.4, A1.9–A1.11, A1.15, A1.17, A1.18

Added pages :

0.5, 0.6
2.11, 2.12
8.15–8.46
9.4.1–9.4.2
9.17–9.22
11.21–11.26
12.1–12.4
13.1–13.10
A1.19–A1.24

CONTENTS

1 – GENERAL

1.1 – ISO CODE (NFZ 68010) – EIA CODE (RS244.B)

- Programming is of the variable, address type.
- Programming is possible by means of two codes :
  . ISO code according to standards NFZ 68010, 68030, 68032
  . EIA code according to standards RS244.A and 273.A.

<u>Characters recognized</u>

MEANING	ISO	EIA
10 digits	from 0 to 9	from 0 to 9
Letters of the alphabet	A to Z	A to Z
Start of program	%	EOR
Start of comment	(	,
End of comment	)	%
+ Sign	+	+
– Sign	–	–
Decimal point	.	.
Greater than	>	
Less than	<	
Multiplied	*	
Equal	=	
Division	/	
Commercial A	@	
End of block	LF	EOB
Block jump	/	/
Program subdivision	:	letter O
End of tape	X OFF	BS

<u>NOTE</u> :

As the EIA standards make no provision for comments, the characters "," and "%" have been retained and have the same effect on the system as the parentheses in ISO.

<u>IMPORTANT</u> :

As the equivalent of the characters >, <, *, = and @ are not defined in the EIA code, neither parametric programming nor the input or output of tool dimensions is permitted in this code.

<u>Characters which are recognized by the system but have no effect on its operation</u>

MEANING	ISO	EIA
Horizontal Tab.	HT	TAB
Carriage return	CR	
Space	SP	SP
Delete	DEL	DEL
	RUB OUT	RUB OUT

<u>NOTE</u> :

In EIA code, a missing character is indicated by a parity error.

Punching combinations for special characters in ISO code

No. of data tracks		8	7	6	5	4		3	2	1
Meaning	Character	Punching combinations								
Less than	(			●	●	●	•	●		
Greater than	)	●		●	●	●	•	●	●	
Multiplied	*	●		●		●			●	
Equal	=	●		●	●	●	•	●		●
Division or block jump	/	●		●		●	•	●	●	●
Commercial A	@	●	●				•			
AND	&	●		●			•	●	●	
OR	!			●			•			●
Dollar	$			●			•	●		
Comma	,	●		●		●	•	●		
Decimal point	.			●		●	•	●	●	
Apostrophe	'			●			•	●	●	●
Semi-colon	;	●		●	●	●	•		●	●
Hash	#	●		●			•		●	●
Question mark	?			●	●	●	•	●	●	●
Inverted commas	"			●			•		●	

ISO CODE : NFZ 68010 – EIA CODE : RS244.B

		ISO								
	Hole tracks	8	7	6	5	4		3	2	1
Meaning	Character	Hole combination								
Program start, rewind stop	%	●		●			·	●		●
+ sign	+			●		●	·		●	●
– stop	-			●		●	·	●		●
Digits	0			●	●		·			
	1	●		●	●		·			●
	2	●		●	●		·		●	
	3			●	●		·		●	●
	4	●		●	●		·	●		
	5			●	●		·	●		●
	6			●	●		·	●	●	
	7	●		●	●		·	●	●	●
	8	●		●	●	●	·			
	9			●	●	●	·			●
Angular coordinates about X axis	A		●				·			●
Angular coordinates about Y axis	B		●				·		●	
Angular coordinates about Z axis	C	●	●				·		●	●
Tool compensator	D		●				·	●		
External parameter	E	●	●				·	●		●
Feedrate or Dwell	F	●	●				·	●	●	
Preparatory function	G		●				·	●	●	●
Sub-program no.	H		●			●	·			
Interpolation address	I	●	●			●	·			●
Interpolation address	J	●	●			●	·		●	
Interpolation address	K		●			●	·		●	●
Program variable	L	●	●			●	·	●		
Auxiliary function	M		●			●	·	●		●
Block number	N		●			●	·	●	●	
	O	●	●			●	·	●	●	●
Various parameters	P		●		●		·			
	Q	●	●		●		·			●
	R	●	●		●		·		●	
Spindle speed	S		●		●		·		●	●
Tool number	T	●	●		●		·	●		
Secondary axis // to X	U		●		●		·	●		●
Secondary axis // to Y	V		●		●		·	●	●	
Secondary axis // to Z	W	●	●		●		·	●	●	●
Primary axis, X	X	●	●		●	●	·			
Primary axis, Y	Y		●		●	●	·			●
Primary axis, Z	Z		●		●	●	·		●	
Program subdivision	:			●	●	●	·		●	
Optional block	/	●		●		●	·	●	●	
Carriage return	CR	●				●	·	●		●
Line feed, (End of block)	LF					●	·		●	
Start of comment	(			●		●	·			
End of comment	)	●		●		●	·			●
Space	SP	●		●			·			
End of tape character	XOFF	●			●		·		●	●
Horizontal tab.	HT					●	·			●
Delete	DEL	●	●	●	●	●	·	●	●	●
Null	NUL						·			

EIA									
	8	7	6	5	4		3	2	1
Character	Hole combination								
EOR					●	·		●	●
+		●	●	●		·			
-		●				·			
0			●			·			
1						·			●
2						·		●	
3				●		·		●	●
4						·	●		
5				●		·	●		●
6				●		·	●	●	
7						·	●	●	●
8					●	·			
9				●	●	·			●
a		●	●			·			●
b		●	●			·		●	
c		●	●	●		·		●	●
d		●	●			·	●		
e		●	●	●		·	●		●
f		●	●	●		·	●	●	
g		●	●			·	●	●	●
h		●	●		●	·			
i		●	●	●	●	·			●
j		●		●		·			●
k		●		●		·		●	
l		●				·		●	●
m		●		●		·	●		
n		●				·	●		●
o		●				·	●	●	
p		●		●		·	●	●	●
q		●		●	●	·			
r		●			●	·			●
s			●	●		·		●	
t			●			·		●	●
u			●	●		·	●		
v			●			·	●		●
w			●			·	●	●	
x			●	●		·	●	●	●
y			●	●	●	·			
z			●		●	·			●
o		●				·	●	●	
/			●	●		·			●
						·			
EOB	●					·			
?			●	●	●	·		●	●
%		●		●	●	·		●	●
SP				●		·			
BS			●		●	·		●	
TAB			●	●	●	·	●	●	
DEL		●	●	●	●	·	●	●	●
NUL						·			

1.2 – STRUCTURE OF THE PROGRAM TAPE

1.2.1 – Workpiece program

- The tape must begin with the character :
  . % in ISO
  . EOR in EIA (end of record).

  The ISO – EIA code is recognized by the system on reading this character.

- The tape must end with the character :
  . X OFF in ISO
  . BS in EIA (Backspace).

  FORMAT OF THE CONTROL BLOCKS

- Blocks with variable, address format.

- Decimal point : as the unit used is the millimeter, .01 represents 1/100 mm

- It is possible to omit :
  . the "+" sign taken by default for dimensions,
  . leading and trailing zeros,
  . spaces, tabulations and RUB OUT.

NOTE :

Maximum memory capacity is 64 Kb.

1.2.2 – Tool compensations on separate tape

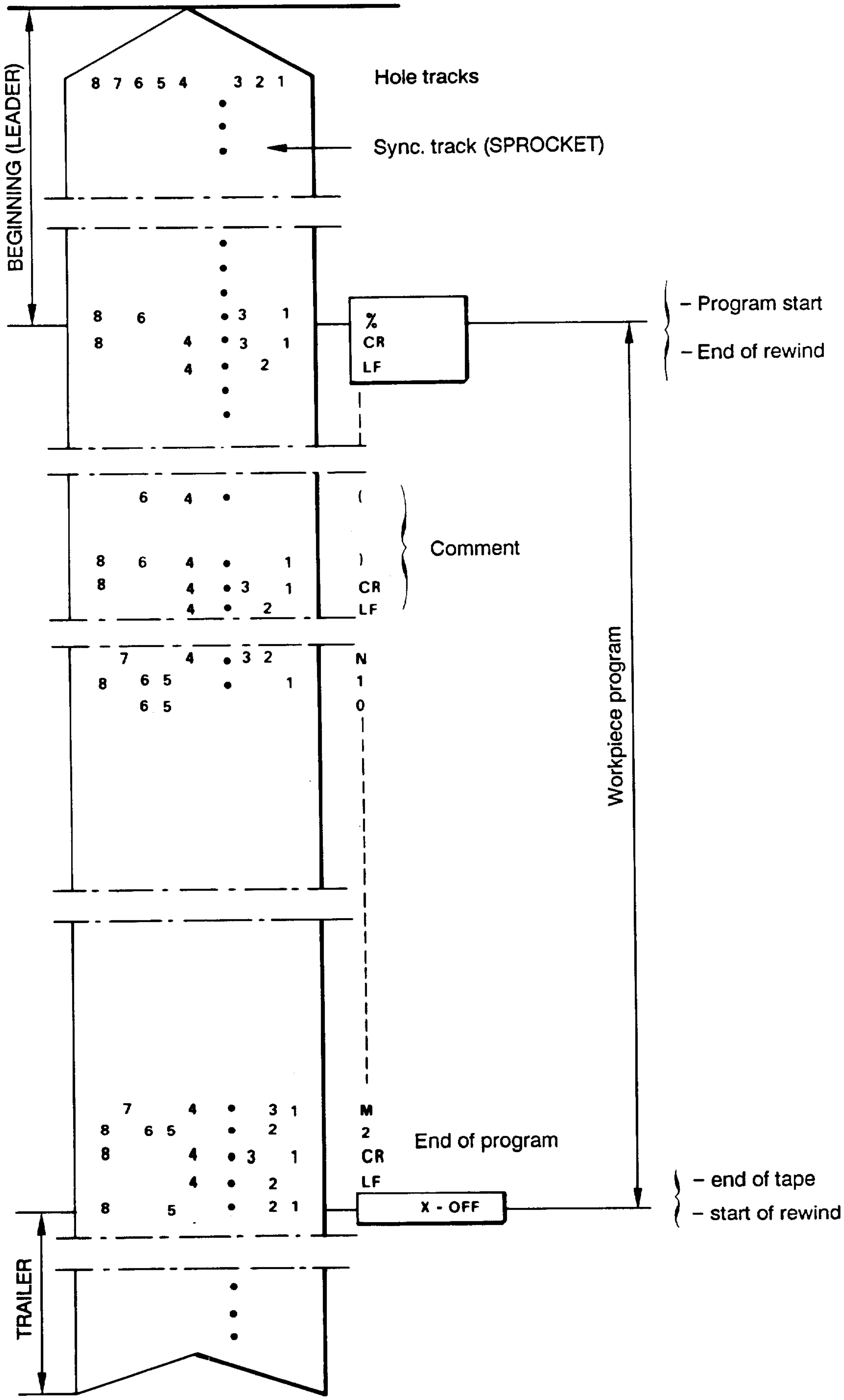
ISO Format

% (comment of max. 40 characters)				CR	LF	
D1	L ± 4.3	R ± 3.3	@ ± 3.3	CR	LF	(@ = cutter tip radius)
D2	L ± 4.3	R ± 3.3	@ ± 3.3	CR	LF	
.						
.						
.						
D32	L ± 4.3	R ± 3.3	@ ± 3.3	CR	LF	
X OFF	LF					

NOTE :

The tool compensation tape must be in ISO code, since @ does not exist in EIA code.

STRUCTURE OF THE PROGRAM TAPE (Example in ISO)



NOTES

2 - PROGRAMMING

2.1 - GENERAL FORMAT OF DATA AND DEFINITION OF ADDRESSES (ISO/DIS 6983/1 APPENDIX C)

Programming dimensions which may contain a decimal part are expressed thus :

X + 36 equivalent to X = 36 mm
X – .3 equivalent to X = –0.3 mm

% 04	Program number
N05	Sequence number (0 to 32767)
G02	Preparatory functions
H04	Sub-program number, used with G77
X+053	Movement along the X axis
Y+053	Movement along the Y axis
Z+053	Movement along the Z axis
I+053 J+053 K+053 I053 J053 K+053	In G2 or G3, absolute or incremental coordinates from the centre of the circle Complete turns, in helical interpolation In G45, finishing depth of cut In G45, finishing width of cut In G84, tap pitch for rigid tapping
A+053	Angular coordinate in A (around X axis)
B+033	Angular coordinate in B (around Y axis)
C+033	Angular coordinate in C (around Z axis)
F05	Feedrate

EA+033	In G1, angle, in degrees, from the straight line with respect to axis: X in G17 Z in G18 Y in G19
EB+053	EB + in G1, G2 or G3, radius between any two elements. EB – in G1, chamfer between two straight lines In G45, blend radius or fillet
ED+033	Angular offset
EF+022 EF 05	Dwell in cycle G82, 87 or 89 Special feedrate on a fillet or chamfer
EG 03	Programmed acceleration override (1–100%)
EI 053	In G45, axial finishing speed
EJ 053	In G45, lateral finishing speed
EK 053	Up/down spindle speed ratio for G84 rigid tapping
EM±	Maximum dimensions of the parallelepiped for 3D display
EP 053	In G45, axial roughing speed
EQ 053	In G45, lateral roughing speed
ER+053	Position of withdrawal plane in canned cycles
EX 053	In G23, intermediate point on a circle In G45, X dimension of pocket in incremental mode
EY 053	In G23, intermediate point on a circle In G45, Y dimension of pocket in incremental mode
EZ 053	In G23, intermediate point on a circle In G45, Z dimension of pocket in incremental mode
P+053	In G83 or G87, value of the first penetration In G16 (P+ or P–), direction of tool on X axis In G45 (P043), setting of rough working axial cut depth

Q+053	In G83 or G87, value of the last penetration In G16, (Q+ ou Q–), direction of tool in X direction In G45 (Q043) setting of rough working lateral cut depth
R+053	Radius of the circle in circular or helical interpolation mode In G16, (R+ ou R–), direction of tool in Z direction In G92, radius of the reference circle for application of tangential feed
R±	In G0 or G1, positioning at a programmed distance from a point
F052	In G93, F042 , feedrate programmed in V/L In G94, feedrate expressed in mm/min ; max. 15mm/mn. In G04, F022, length of dwell in seconds ; max. 99 secs.
M03	Auxiliary functions : 15 decoded, 240 coded
NC05	Spline curve number. NC0 : end of definition of a spline curve
S05	In G97, spindle rotation speed in rpm In G77, S04 number of cycles of a sub–program
T03	Tool number from 0 to 255
D02	Compensation number from 0 to 32
L03	Program variables from 0 to 19, 100 to 199 and 900 to 939
E113	Program parameters

2.2 - G FUNCTIONS

CODE	CANCELLATION	DESCRIPTION	SEE §
G00	G01-G02-G03 G23	Rapid mode linear interpolation	4.1.1
G01*	G00-G02-G03 G23	Linear interpolation at programmed rate	4.1.1
G02	G00-G01-G03 G23	Circular interpolation at programmed tangential feed rate, clockwise direction	4.1.2
G03	G00-G01-G02 G23	Identical to G02, but anti-clockwise	4.1.2
G04	End of block	Dwell programmable with address F	10.2
G06	End of block	Spline curve execution command	11.18
G09	End of block	Precise stop at end of block before continuing to the following block	10.1.3
G10	End of block	End of machining on reception of the "End of block stop" signal, or on detection of a measured threshold, with optional jump to the following block	11.2
G12	End of block	Handwheel enhancement of program feed	10.1.5
G16*	End of block	Definition of tool axis using PQR	6.3
G17*	G18 - G19	XY plane selected for circular interpolation and radius compensation	4.1.2
G18	G17 - G19	ZX plane selected for circular interpolation and radius compensation	4.1.2
G19	G17 - G18	YZ plane selected for circular interpolation and radius compensation	4.1.2

CODE	CANCELLATION	DESCRIPTION	SEE §
G23	G00–G01–G02 G03	Programming of a circle by 3 points	4.1.2
G40*	G41–G42	Cancelling of tool radius compensation	7
G41	G40 – G42	Tool radius compensation to left side of profile	7
G42	G40 – G41	Tool radius compensation to right side of profile	7
G45	End of block	Rectangular and oblong pocket cycle	8.2
G46	End of block	Definition of a special pocket and facing cycle block with any contour	8.3
G48	End of curve definition	Start of spline curve definition	11.18
G49	End of block	Deallocation of spline curve memories	11.18
G51	End of block	Mirror function. Enabled or disabled using axes addresses	3.2.5
G52	End of block	Absolute programming of dimensions with respect to measurement origin	3.2
G53	G54	DAT 1, DAT 2 and DAT 3 disabled	3.2.1
G54*	G53	DAT 1, DAT 2 and DAT 3 enabled	3.2.1
G59	End of block	Programmed origin offset. Added to offset enabled by G54	3.1.3/3.2.2
G70	G71	Data input in inches	11.8
G71*	G70	Data input in metric	
G73*	G74	Cancellation of the scaling factor	3.2.6

CODE	CANCELLATION	DESCRIPTION	SEE §
G74	G73	Scaling factor enabled	3.2.6
G75	End of block	Definition of an emergency withdrawal sub-program	11.4
G76	End of block	Transfer of the current parameter values into the part program	11.17
G77	End of block	Unconditional call of a sub-program or of a series of blocks with return	11.3.1
G79	End of block	Conditional or unconditional jump to a block without return	9.3/11.3.2
G80*	G81 to G89	Cancellation of machining cycle	8.1
G81	G80 G82 to G89	Centering and drilling cycle	8.1
G82	G80 G81 G83 to G89	Counter boring and drilling cycle	8.1
G83	G80 to G82 G84 to G89	Woodpeck drilling cycle	8.1
G84	G80 to G83 G85 to G89	Tapping cycle or Rigid tapping cycle	8.1 8.4
G85	G80 to G84 G86 to G89	Boring cycle	8.1
G86	G80 to G85 G87 to G89	Boring cycle with spindle stop at bottom of hole	8.1
G87	G80 to G86 G88 to G89	Chip-breaker cycle	8.1
G88	G80 to G87 G89	Boring and hard facing cycle	8.1
G89	G80 to G88	Boring cycle with dwell at bottom of hole	8.1

CODE	CANCELLATION	DESCRIPTION	SEE §
G90*	G91	Absolute programming with respect to program origin	3.2.2
G91	G90	Incremental programming with respect to block starting point	3.2.2
G92	End of block	Pre–selection of program origin	3.2.3
G92 R	G92–R0–M02	Tangential feedrate programming	10.1.6
G93	G94	Feed rate expressed in inverse time (V/L)	10.1.2
G94*	G93	Feed rate expressed in mm/min	10.1

* Indicates the default condition. ie when the system is switched on, or after a reset.

Special addresses and data for machining pockets with an arbitrary contour

ED 053	Roughing cutter diameter
EH+053	Material plane dimensions
EI+052	Axial finishing feed
EJ+052	Lateral finishing feed
EP+052	Axial feed
EQ+052	Lateral machining feed
EX EY +053 EY	Coordinates of the roughing end point in the plane of the pocket
J 053	Machining allowance
LX LY +53 LZ	Coordinates of the roughing or finishing start point in the plane of the pocket
NP 05	Pocket number
NR	Type of work : NR + : inward NR – : outward
NU 02	Type of block. Always preceded by function G46
P 053	First penetration for G83 or G87
Q 063	Last penetration for G83 or G87
R 01	Tool rotation direction (3 or 4)

2.3 – DECODED M FUNCTIONS

CODE	FUNCTION		CANCELLATION	DESCRIPTION	SEE §
	BEFORE	AFTER			
M00		X	Cycle start	Programmed stop	
M01		X	Cycle start	Optional stop	
M02		X	% or EOR	End of workpiece program	
M03	X		M4–M5–M0–M19	Clockwise spindle rotation	
M04	X		M3–M5–M0–M19	Anti–clockwise spindle rotation	
M05*		X	M3–M4	Spindle stop	
M06		X	Report	Tool change	
M07	X		M9–M2	Coolant 2, on	
M08	X		M9–M2	Coolant 1, on	
M09*		X	M7–M8	Coolants off	
M10		X	M11	Axis clamp	
M11	X		M10	Axis unclamp	
M12		X	Cancellation of feed–stop	Setting–up of "Recall" mode	11.15
M19		X	M3–M4–M5	Spindle orientation	10.4

CODE	FUNCTION		CANCELLATION	DESCRIPTION	SEE §
	BEFORE	AFTER			
M40 to M42	X			3 spindle ranges	
M48*		X	M49	Enables the spindle and feed potentiometers	
M49	X		M48	Disables spindle and feed potentiometers	
M997	X		M998-M999-M2	Forces block continuation	9.1.2
M998*	X		M999-M997	Allows EDIT, MDI modes and sub-program calls by the PLC	9.1.2
M999	X		M998-M2-M997	Disallows EDIT, MDI modes and sub-program calls by thePLC	9.1.2

* Indicates default functions. ie when the system is switched on, or after a reset.

NOTE :

- Only function M6 is not modal. It is reset to zero as soon as the M acknowledgement is detected by the CNC (CRM).
- Several decoded M functions (eg M3, M8, M41) can be programmed in one block.
- The other coded M functions defined by the manufacturer are "after" functions. Only one of these functions is authorized in any one block.
- Function M19 is sent to the PLC, which then takes over the responsibility for orientating the spindle

2.4 - ADDRESS EQUIVALENCE TABLE

This table is used to obtain an equivalence between one address and another. It allows a program to be written for one set of axes, eg XZ, and makes it possible to use the same program for another set, eg YX.

The @ symbol followed by an address (A to Z) designates an equivalent address.

An equivalent address is declared by programming :

$$@\left\{\begin{matrix}A\\ \cdot\\ \cdot\\ Z\end{matrix}\right\} = \left\{\begin{matrix}A\\ \cdot\\ \cdot\\ Z\end{matrix}\right\}$$

A value is assigned to an equivalent function by programming :

$$@\left\{\begin{matrix}A\\ \cdot\\ \cdot\\ Z\end{matrix}\right\}$$ Followed by the value

<u>EXAMPLE</u> :

@ X = Z

@ X 300 is then equivalent to Z 300

The table of address equivalence is initialized when the system is switched on, or during a reset or M02 (@ A = A @ B = B ... @ Z = Z).

Declaration of a new equivalent address delays preparation of the block in which it is programmed until the preceding block has been executed.

The table of address equivalence is displayed by pressing the "Program variables" key, which operates as a bistable : press once to display the program variables, press again to display the table of address equivalence.

See example in paragraph 9.5.

NOTES

3 – DIMENSION PROGRAMMING SYSTEM

3.1 –DEFINITIONS

The system always processes dimensions with reference to a DATUM or measurement origin, irrespective of the programming mode chosen (G90, G91, G52).

3.1.1 – Origins

The origins are as follows :

- M/C measurement origin : Om

  This is a reference point defined on each axis by the machine manufacturer and is usually established by contacting a switch.
  It is used to establish the absolute measurement origin.

- Workpiece origin : Ow

  This origin is independent of the measurement system and is set to a known point, perhaps on a location fixture. From the known point, the workpiece may be be positioned directly or may be offset, by using shims or a comparator

- Program origin : OP

  The program origin is independent of the measurement system. It represents the origin of the reference coordinate system from which the programmer has chosen to write his program.

3.1.2 – DAT 1

After referencing the m/c, the initial display shows the X, Y and Z dimensions pre-assigned to the machine axes by the m/c builder. The operator then establishes the workpiece origin as follows.

- By physically jogging the slides to the required positions, the "current position" values can be entered semi-automatically.
  * X Y Z LF is input in DAT 1 mode. The displayed position will then zero, and will correspond to the absolute value of the current position with respect to the machine origin (MOS).

- If these values are already known, then they can be entered directly through the keyboard, in DAT 1 mode.

3.1.3 – Other offsets

- DAT 2

  This offset is always added to DAT 1. It is available simply for the sake of convenience, to hold a shim thickness for example, in order to more easily set up the workpiece origin.

  DAT 2 is often known by the programmer, who can then instruct the operator, (via a set up sheet) into what value to enter through the keyboard. It is taken into account by the system in all cases except when in G52 or G53.

  DAT 3 (optional)

  This offset can be used on a machine fitted with a servo–controlled rotary table.

  It represents the difference between the theoretical axis of rotation of the workpiece and that of the actual table. The program origin must be set on the theoretical axis of rotation of the workpiece.

- A PROGRAMMED OFFSET, eg G59 X..Y..Z..

  From the workpiece origin, a programmed offset may be used to move the part program origin to any point. This is to allow for easy interpretation of the part drawing dimensions, or repetitive shapes, etc.

  . In absolute programming (G90)
  G59 Xn defines the value of the offset to be applied to all X dimensions that follow.

  A new G59 Xn cancels and replaces the previous one, and all subsequent X axis demands take the value of the last G59 programmed into account.

  . In incremental programming (G91)
  G59 Xn defines the value of the offset to be applied to the first X dimension after the G59.

  A new G59 will, in the same way modify the first dimension which follows it. Thus, the absolute value of the position will be modified by the sum of all G59s previously programmed on that axis.

NOTE :

To cancel the offset or offsets applied in G59, it is recommended :

- *in absolute programming (G90) :*
  to program G59 X0 Y0 Z0.

- *in incremental programming (G91) :*
  to revert to absolute programming (G90) and program G59 X0 Y0 Z0.

3.2 – RULES FOR THE USE OF OFFSETS

Programmed axis demands are always expressed from an origin.

The dimensions programmed may be given as follows :

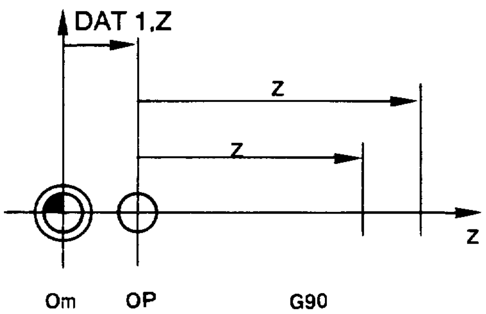
. Absolute programming (G90)
The dimensions are taken with respect to the program origin, Op.

. Incremental programming (G91)
The dimension are taken with respect to the previous position.

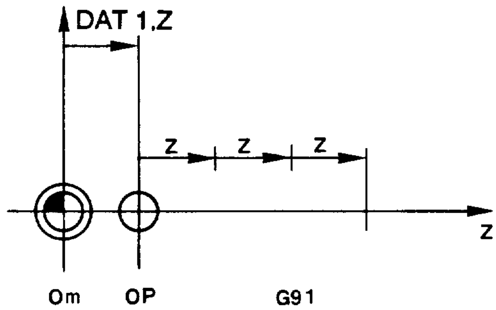
. Absolute programming (G52).
The dimensions are taken with respect to the measurement origin, Om.

With G52, no offsets or tool compensations are taken into account. This function is cancelled at the end of the block. It must be programmed before the axes addresses and without radius compensation (G41 – G42). Do not forget to cancel any active ED.

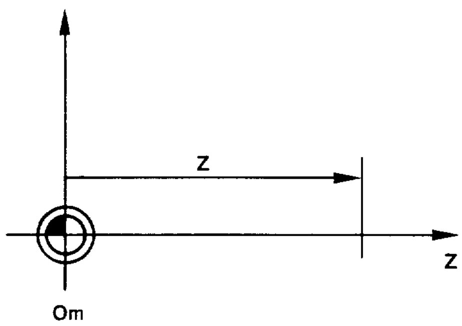
This programming mode is used to position an axis to a fixed point with respect to the machine structure (eg for automatic tool change sequences).



Absolute programming



Incremental programming



G52 : Absolute programming with respect
to the measurement origin
(m/c reference.)

NOTE :

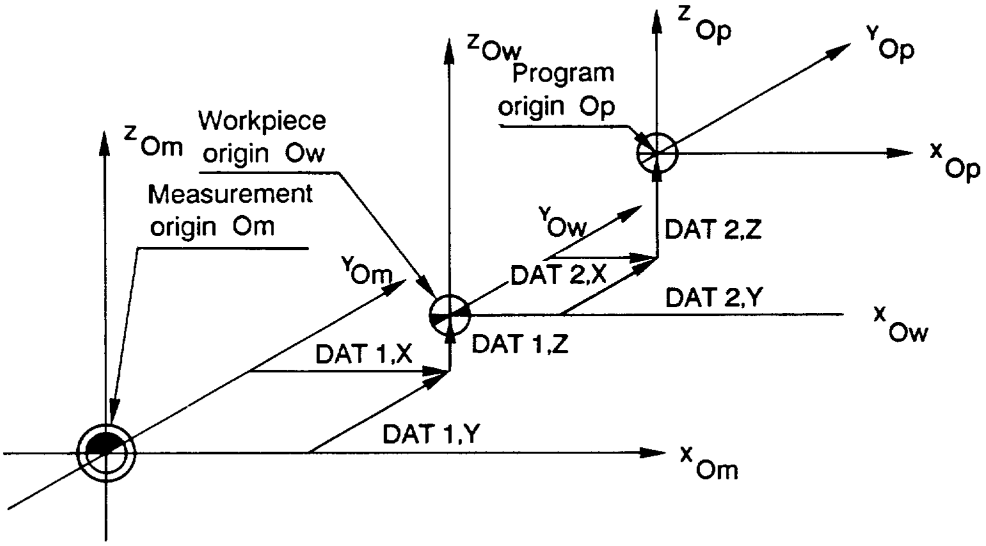
- *The first program movement along each axis MUST be programmed in absolute mode (G90) Otherwise, G91 would be taken as a G90 for this movement. No error will be reported!*

- *All programmed dimensions, modified by the various offsets and tool dimensions, are compared with the axes limits as defined by the builder. If these dimensions are exceeded, the system generates an error message.*

3.2.1 – DAT 1 and DAT 2

As previously described, DAT 1 is set with respect to the m/c measurement origin by positioning the tool to a datum point on the workpiece (Ow) and entering that position in DAT 1 mode.

DAT 2 represents any distance between the point at which DAT 1 was set by the operator and the program origin (Op) as required by the programmer. It may not be necessary to enter any value in DAT 2, in which case Ow and Op are coincident.



Once DAT 1 (and DAT 2) have been entered, the program origin (Op) will be in accordance with the workpiece program tape.

NOTE :

- *The DAT 1 and DAT 2 values are enabled by programming G54. They are disabled by G53.*

3.2.2 – Offsets programmed by G59

Program origin offsets must not exceed the system measurement capacity.

An offset programmed with G59 is modal.

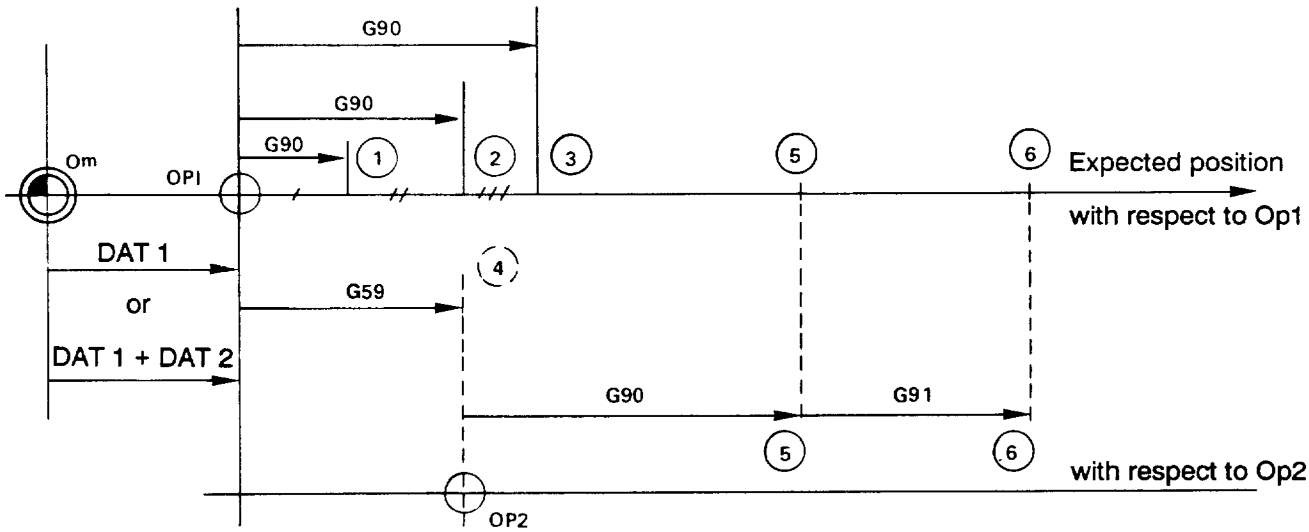
G59 X $\pm$ 5.3 Y $\pm$ 5.3 Z $\pm$ 5.3

There are not axis movements with G59, simply a relocation of the origin.

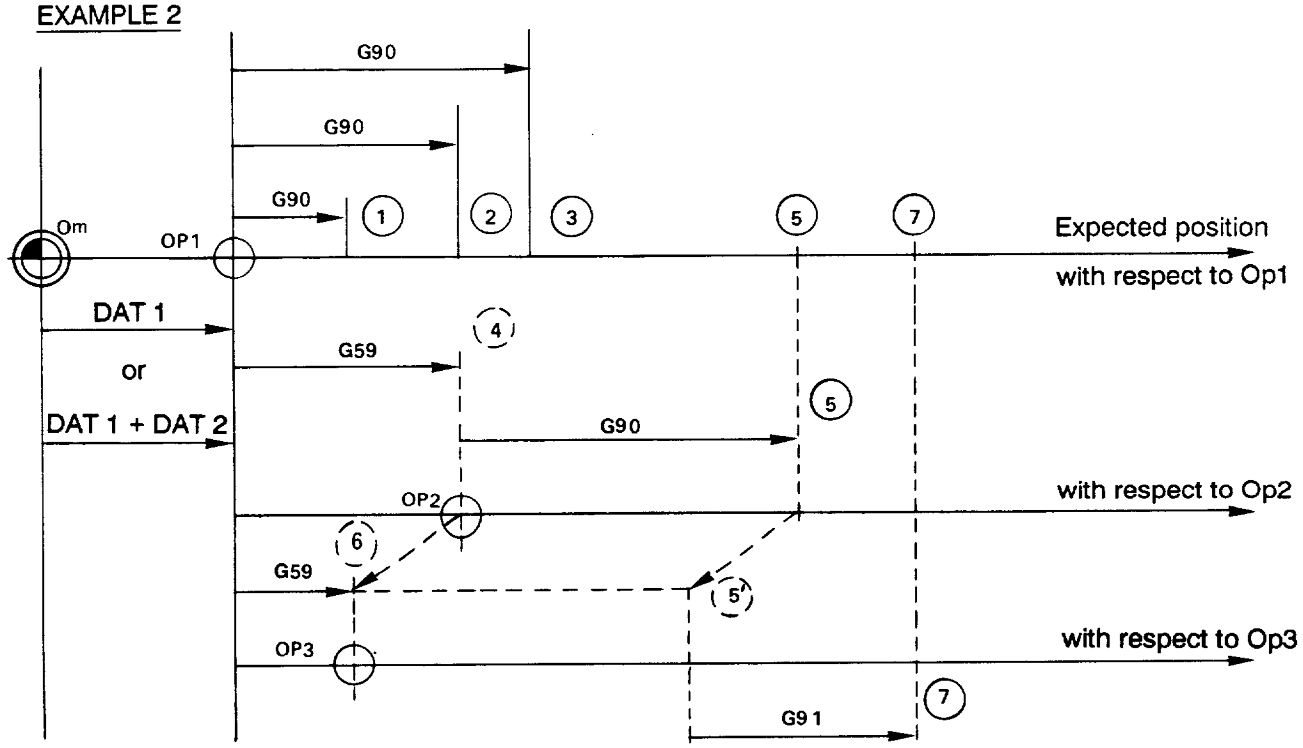
Absolute programming (G90 – G59)

EXAMPLES

EXAMPLE 1



EXAMPLE 2



When the system is programmed in absolute mode (G90), a G59 offset is added to the program origin as defined by DAT 1 + DAT 2.

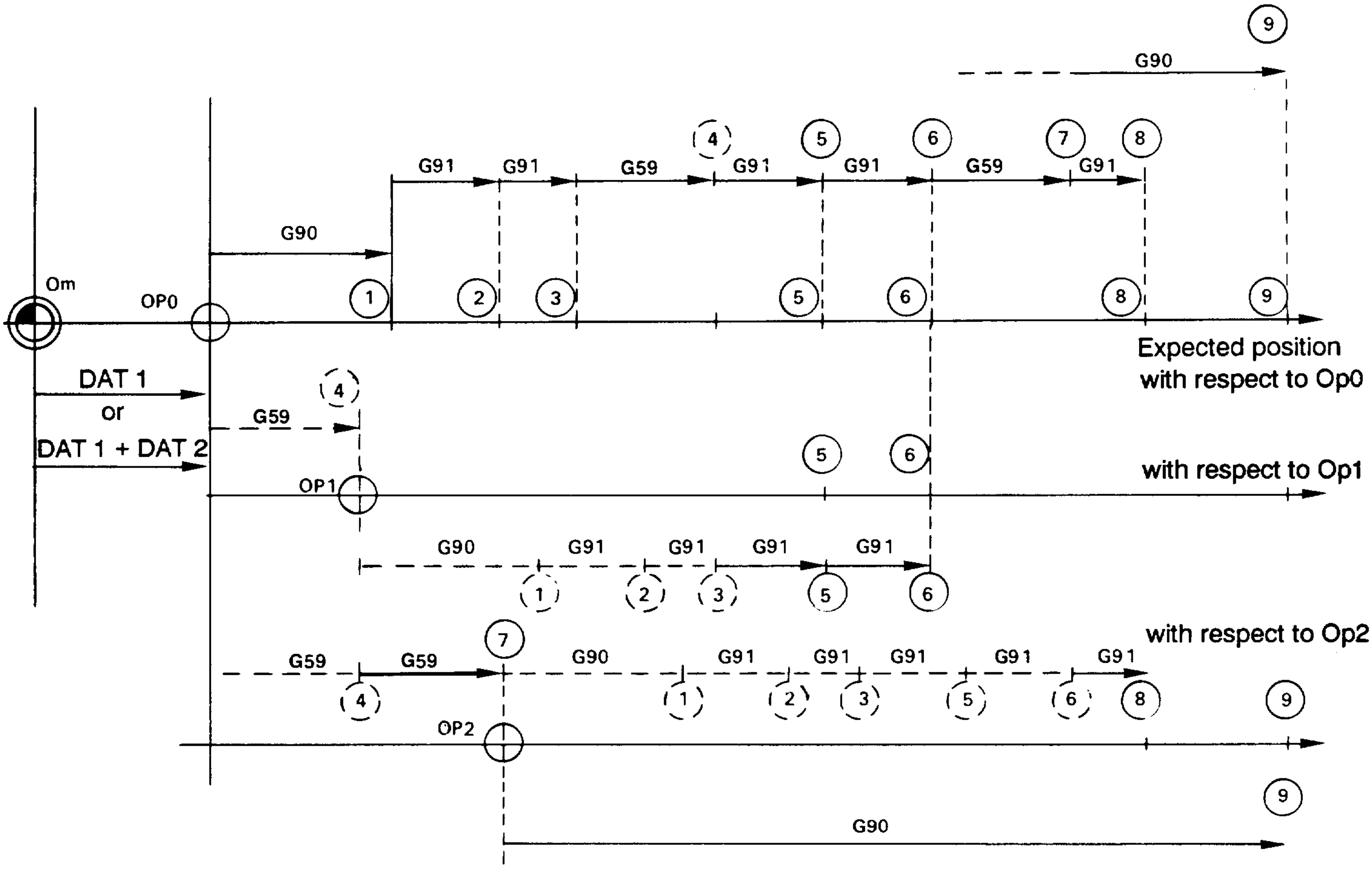
EXAMPLE 1 :

The absolute dimensions programmed after G59 are translated by the value programmed with G59 (point 5). Any incremental values programmed after a G90 (point 6) are with respect to the previous point.

EXAMPLE 2 :

The reasoning used in the first example applies up to point 5. A second G59 then "translates" the coordinate system without axis rotation.
Point 7 will therefore be defined with respect to the new point 5' and not the old point 5.
Programmed origin offsets can be removed by programming a zero offset on the translated axes, eg : G90 G59 X0 Y0 Z0.

Incremental programming (G91 – G59)



When the system is programmed in incremental mode (G91), the first dimension programmed after G59, (point 5), is translated by the value of the offset. The absolute origin op1 is determined from op0 using the same value.

If a second G59 origin offset is programmed in the same way, the new absolute origin op2 is determined from op0 by a translation equal in value to the sum of the two offsets.

On returning to absolute (G90) programming, (point 9), the absolute dimension will now be defined with respect to op2.

Incrementally programmed origin offsets may be removed by programming :

- either the offset corresponding to the sum of all the offsets carried out but with opposite sign, if still in incremental mode,
- or a zero origin offset, after switching to absolute programming : G90 G59 X0 Y0 Z0.

NOTE :

- *A manual reset or MO2 reset returns the system to its initial status (op0).*
- *To simplify and understand a part program, it is strongly recommended to be in G90 status before programming an origin offset (G59). Simply programming G59 X0 Y0 Z0 will then return to the initial reference system.*
- *Programming in m/c measurement origin mode (G52) is allowed only in G40 mode. Otherwise, the system will report error 27.*

3.2.3 – Pre-selection of program origin : G92

The dimensions programmed after the non-modal function G92, define the current position of the slide with respect to the program origin. DAT 1 is automatically updated as follows:

DAT 1 = machine dimension – programmed dimension – tool length compensation.

Note that DAT 2 is not affected by the above equation.

G92 has the effect of closing down the program "look ahead" decode, so preventing its use in radius compensation mode (G41 or G42), or any other mode which requires "look ahead" to be active.

NOTE :

- *G92 is not executed in TEST or SEARCH modes, so do NOT search to a block following a G92.*
- *The new DAT 1 is retained at the end of the program and so can affect the next part program, (or even the following part using the same program), unless deliberate steps are taken to prevent adverse repercussions.*

3.2.4 – Rotary table offset : DAT 3 (optional)

The rotary table offset applies only to rotary axis A, B or C.

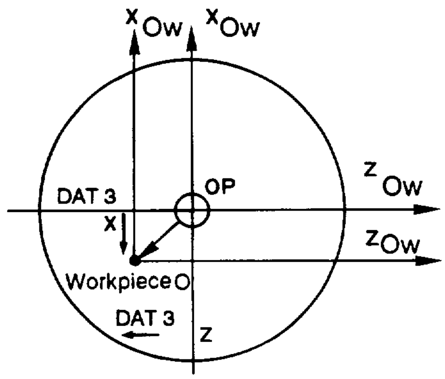
The relevant axis address and value are input, in DAT 3 :

Input Z... X... : compensation on the Z and X axes with respect to B.
Input X... Y... : compensation on the X and Y axes with respect to C.
Input Y... Z... : compensation on the Y and Z axes with respect to A.

The loading process can also be carried out using parameter E6n004 = ... (n = axis number), either from program or MDI.

The explanations below are provided for A DAT 3 rotary table offset for the B rotary axis. They are identical for the DAT 3 for A or C.

The rotary table offset makes allowance, during positioning, for a discrepancy between the theoretical axis of rotation of the workpiece and the true axis of rotation of the table, irrespective of the angle of the table.



. Op : Program origin of table
. Ow : Workpiece origin
. The offset applied is from Op to Ow

Application of this offset pre–supposes that :

- the program origin is set to the theoretical axis of rotation of the workpiece,
- the system is not in G52 or G53,
- the values DAT 3 X and DAT 3 Z are measured with the table positioned on the program zero. ie B = 0.

The part program dimension are converted as follows :

$Xm = Xp + DAT\ 1 + DAT\ 2 + PROG.\ OFFSET + \Delta X$
$Zm = Zp + DAT\ 1 + DAT\ 2 + PROG.\ OFFSET + \Delta Z$
Where :
$\Delta X = (DAT\ 3\ X\ .\ Cos\ B) - (DAT\ 3\ Z\ .\ Sin\ B)$
$\Delta Z = (DAT\ 3\ Z\ .\ Cos\ B) + (DAT\ 3\ X\ .\ Sin\ B)$
B = programmed angular position ($0 \le B < 360°$).
DAT 3 X and DAT 3 Z refer to the values entered, (for the programmed position B = 0) of the theoretical axis of rotation of the workpiece with respect to the real axis of rotation of the table.

NOTE :

- *The values entered with DAT 3 are modified as above and applied to the programmed dimensions X and/or Z.*
- *Angle B defines the position of the tool with respect to the workpiece (B + = anti–clockwise)*
- *The X and Z dimensions MUST be reprogrammed after a rotation, even if the values of the dimensions have not changed, so that the new offset can be applied to them.*

3.2.5 – Mirror function : G51

The mirror function can be applied to any axis of the machine. When applied to axes X, Y and Z, it can allow the system to carry out symmetrical machining using a program which only defines a half or a quarter of the workpiece.

The mirror function modifies :

- the programmed dimensions, which are inverted,
- the offsets programmed in G59,
- the compensation of the tool radius,
- the direction of rotation in circular interpolation mode.

The mirror function does NOT odify :

- DAT 1 – the workpiece origin,
- DAT 2 – the distance between the workpiece origin and the program origin,
- DAT 3 – the rotary table offset,
- the tool dimensions,
- any G52 – measurement origin programming.

The mirror function is enabled by G51, followed by the address of the axis to be mirrored, and by the "–" sign.
G51 X–, mirror on X axis
G51 X+, cancel mirror on X axis.

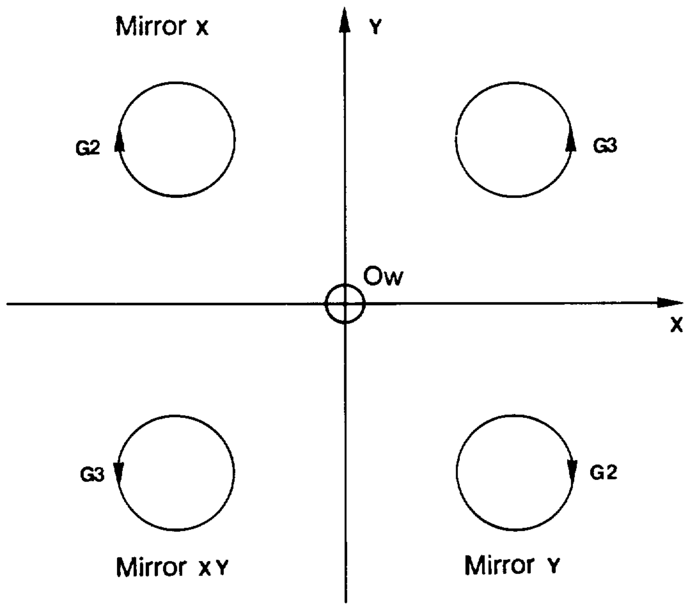
This function must be programmed in a single block. Several axes can be enabled or disabled in the same block.

Special features :

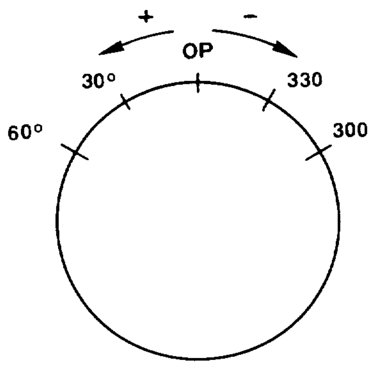
- When a mirror is requested on the axis corresponding to the tool direction, the tool direction must be reprogrammed (G16) to take the mirror into account.

EXAMPLE : G16 P–
X
Y
G51 X– (Mirror on X axis)
G16 P+

- In circular interpolation mode, the direction of G2 or G3 is reversed in mirror X or mirror Y.



- Rotary axes. The mirror function on a rotary axis reverses the direction of rotation and produces an angle equal to 360° minus the dimension.



EXAMPLE :

B30 programmed
When mirrored, go to 330° in negative direction

B300 programmed
When mirrored, go to + 60° in positive direction

NOTE :

- *The mirrored axis or axes are highlighted on the IN PROGRESS page.*

- *Any mirror conditions are removed when the system is switched on, or after a reset or when an M02 is executed.*
The axes return to their original status.

3.2.6. – Scaling Factor, SC/1000

This function is used to machine a part scaled to the programmed part.
The centre of scaling is the programme origin and the scaling factor is between 0.001 and 9.999.

Scaling affects:

- the programmed dimensions of the orthogonal axes X, Y, Z,
- the offsets programmed in G59.

Scaling does not affect:

- the dimensions of the rotary axes A, B or C,
- the part origin DAT1,
- the offset between the part origin and the programme origin DAT2,
- the table off–centering DAT3,
- the tool dimensions,
- programming with respect to the measurement origin G52,
- the release dimension at the bottom of the hole in G86 and the guard dimension in G83 (these dimensions are defined in machine parameters by the manufacturer).

The scaling factor is entered from the numerical control panel in DAT1 mode and displayed on the OFFSET page.

The scaling factor can be enabled (G74) or cancelled (G73) at any time except when the system is in radius correction (error 143), circular interpolation or on blocks not entirely defined in themselves (programming in PGP).
Functions G74 and G73 can be followed by dimensions in a given block.

NOTES:

- *The scaling factor can be loaded by external parameter E69000. This parameter is available for read and write.*

- *The value entered must be greater than 0 and less than 10000 (9999) and must be an integer: for instance E69000 = 465.*

- *The scaling factor can be tested to make a jump in the programme: G79 E69000 = xxxx Nxxx.*

NOTES

4 - MOVEMENTS OF THE MAIN AXES X,Y,Z.

The system can control 4 servo-controlled axes (3 main axes and 1 optional rotary axis).

The system can control the three linear axes simultaneously in linear interpolation mode, or two of the three linear axes in circular interpolation mode. The axes move at the feed rate expressed in mm/min or V/L.

4.1 - DIMENSION PROGRAMMING

4.1.1 - Linear Interpolation

The maximum dimension or linear interpolation is ± 99999.999 mm.

Preparatory function : G1 linear interpolation at program feedrate (A modal function).

This is the default function which is automatically set when the system is switched on, or after a reset. It is mutually exclusive with functions G0, G2 and G3.

Addresses :

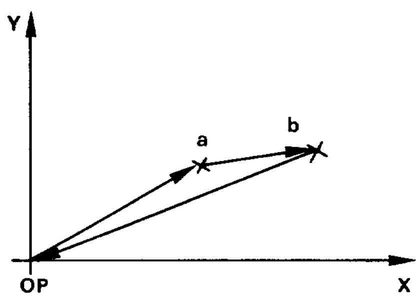
X053 Y053 Z053

The dimensions are expressed in absolute programming mode (+ 99999.999) with respect to the program origin OP in G90 (see 3.2) ; or in incremental programming mode with respect to the previous end point in G91 ; or in absolute programming mode with respect to the measurement origin in G52.

Preparatory function : G0 linear interpolation at rapid traverse rate. (A modal function)

EXAMPLE :

N10 G0	XOP	YOP	ZOP	(rapid to Op)
N20 G1	Xa	Ya	F	(feed to a)
N30 G91	Xab	Yab		(feed to b)
N40 G0 G90	XOP	YOP		(rapid to Op)

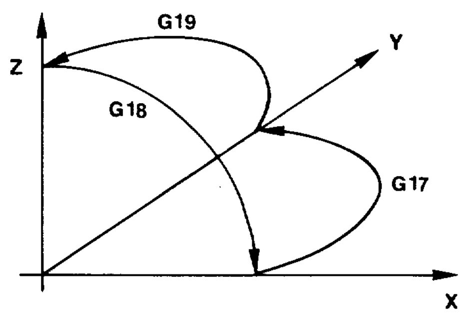


4.1.2 – Circular Interpolation

This interpolation can be selected from the three main planes (ie. Between the axes pairs XY, YZ or ZX). The interpolation of a complete circle can be programmed in a single block.

Maximum radius R 99999.999 mm.

Choice of interpolation plane

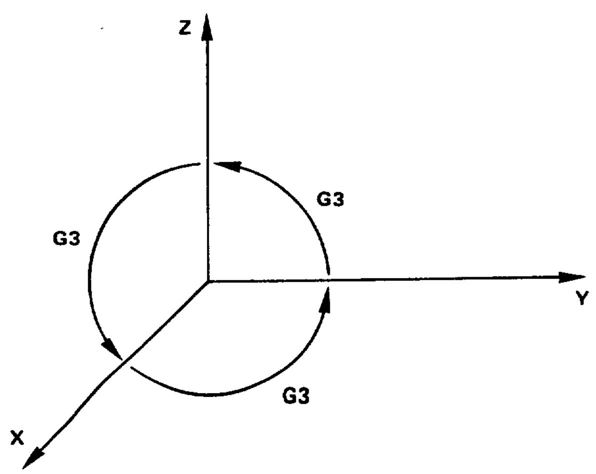


The default interpolation plane is G17, ie. the XY plane. Any change of interpolation plane should be specified before the circular move is programmed.

Preparatory function

G17 to G19	G2	G3	G23
↓	↓	↓	↓
Choice of interpolation plane	Clockwise circular interpolation	Anti-clockwise circular interpolation	Circular interpolation in a direction depending on the intermediate point

(All four functions controlling the type of movement, G0, G1, G2, G3 and G23 are mutually exclusive. ie. they cancel each other).



Addresses with function G2 or G3

X+053	Y+053	I+053	J+053		
Y+053	Z+053	J+053	K+053	or	R053
Z+053	X+053	K+053	I+053		radius of the circle

Finishing point of the circle

Coordinates of the centre of the circle relative to
– the program origin in G90
– the starting point in G91

The end point dimensions are also expressed either in absolute mode with respect to the program origin (Op), or in incremental mode with respect to the circle start point.

The addresses X, Y, Z, I, J, K and R MUST be programmed in the same block, even if I, J and K are zero or, in the case of X, Y, Z or R, are unchanged.

NOTE :

A circle programmed using its end point and its radius must not exceed 180°.

Addresses with function G23 : programming of a circle in the plane by three points, with the start point known.

EX+053	EY+053	X+053	Y+053
EX+053	EZ+053	X+053	Z+053
EY+053	EZ+053	Y+053	Z+053

Intermediate point — End point

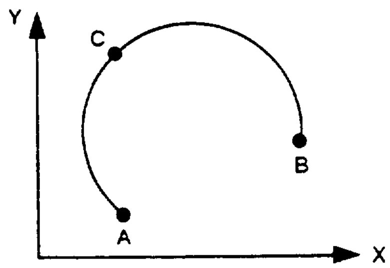
The direction of interpolation depends on the position of the intermediate point with respect to the start point and end point : to the left of line AB, the interpolation is clockwise and to the right anti-clockwise.

CAUTION :

Programming must always be in absolute dimensions.

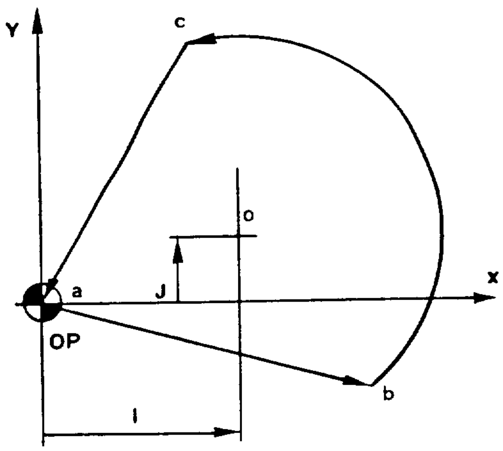
N10 X_A Y_A Circle start point
N20 G23 EX_C EY_C X_B Y_B

(clockwise circular interpolation)



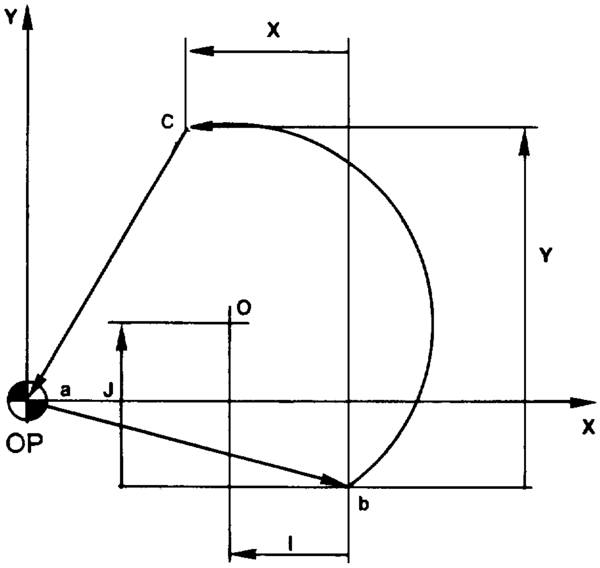
EXAMPLES :

Programming of a circle in absolute mode



N10 G90 G17	G0	Xa	Ya			
N20	G1	Xb	Yb			F
N30	G3	Xc	Yc	Io	Jo	
N40	G1	Xa	Ya			

Programming a circle in incremental mode

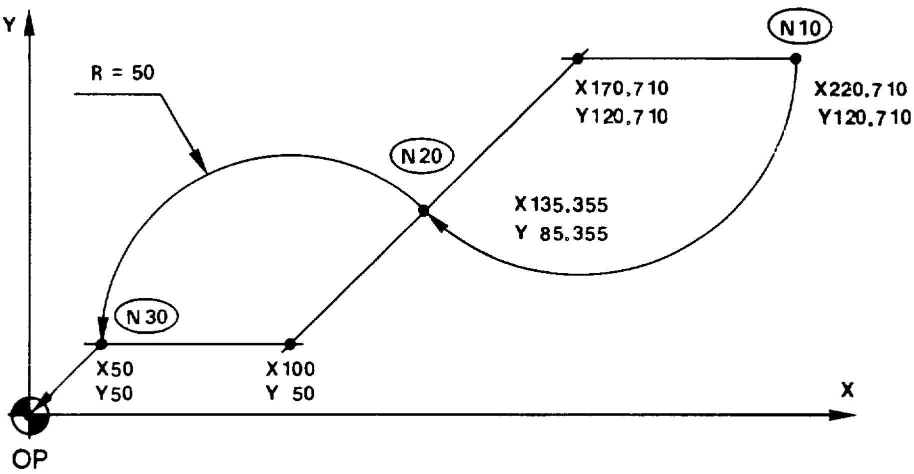


N10 G90 G17	G0	Xa	Ya			
N20	G1	Xb	Yb			F
N30 G91	G3	X	Y	I	J	
N40 G90	G1	Xa	Ya			

In both these examples :

N10 : Rapids to point, "a"
N20 : Feeds to the start point of the circle "b"
N30 : Feeds around the circle, to point "c"
N40 : Feeds back to point "a".

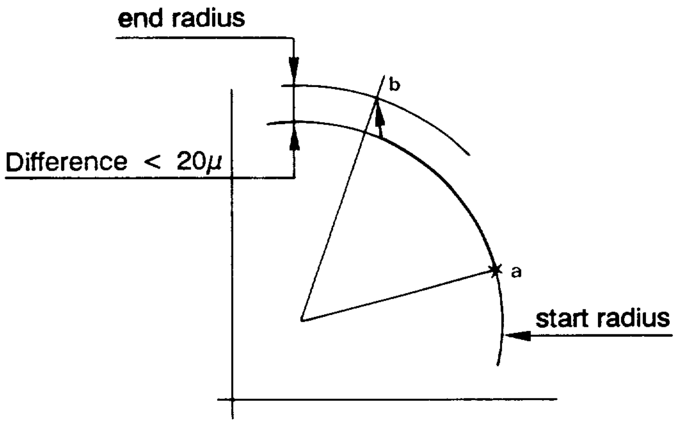
Absolute programming followed by incremental programming

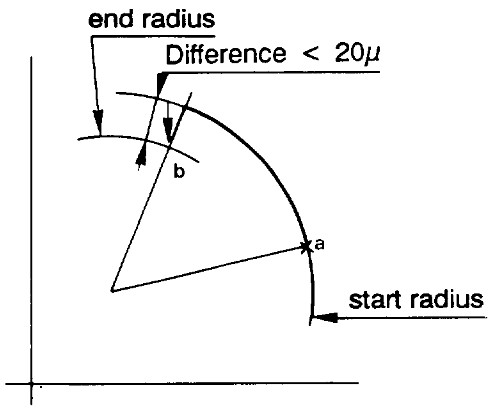


N10 G90	G0	X 220.710	Y 120.710			
N20 G17	G2	X 135.355	Y +85.355	I 170.710	J 120.71	F500
N30 G91	G3	X-85.355	Y -35.355	I-35.355	J-35.355	
N40 G1		X-50.	Y -50.			

NOTE :

- *If the difference between the start radius and the end radius is greater than 20µ, the system will generate an error message (error 107).*
- *If the radius of the circle is less than 20µ, the trajectory followed will be linear.*





4.1.3 – Helical Interpolation

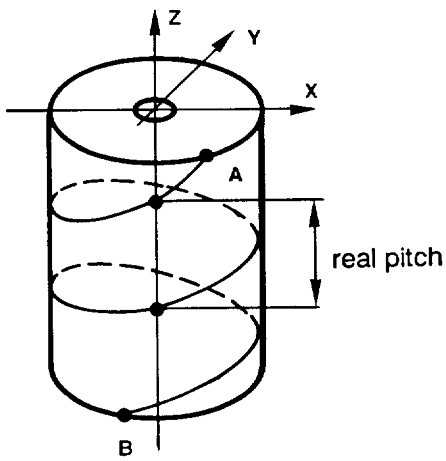
Helical interpolation is used to combine circular and linear machining modes, so as to move the centre of the tool in a constant pitch helix at the programmed feedrate.

Helical interpolation is possible in all three planes of the coordinate system.

The axis of the helix is defined by the data used in circular interpolation mode, and is independent of the orientation of the tool defined by function G16. In other words, a helix may be machined with a tool normal to the axis of the helix.

4.1.3.1 – Programming

EXAMPLE : Helical interpolation in the XY plane G17.



N10 G17 G90 XA YA ZA

N20 G2 XB YB $\left\{\begin{matrix} I & J \\ & R \end{matrix}\right\}$ ZB K F

A : coordinates of starting point in absolute or incremental programming

B : coordinates of end point in absolute or incremental programming

I–J : coordinates of the circle centre in the X Y plane.

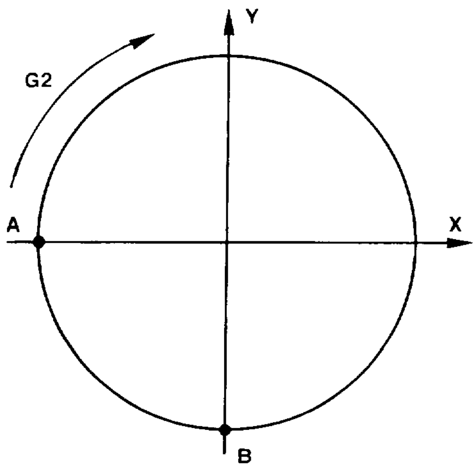
R : circle radius in the X Y plane

K : approximate pitch of helix in the Z direction. An unsigned value. Used by system to determine the number of revs to execute so that the real pitch is as close as possible to the programmed pitch (taking the start point and the end point into account).

F : Tangential speed in mm/min.

	G17	G19	G18
Coordinates of centre of circle	IJ	JK	KI
Pitch	K	I	J
Plane	XY	YZ	ZX
Radius R	R	R	R

EXAMPLE : Helical interpolation, G2 (clockwise) rotation



Let the distance, in Z direction, between A and B, Z = 20.

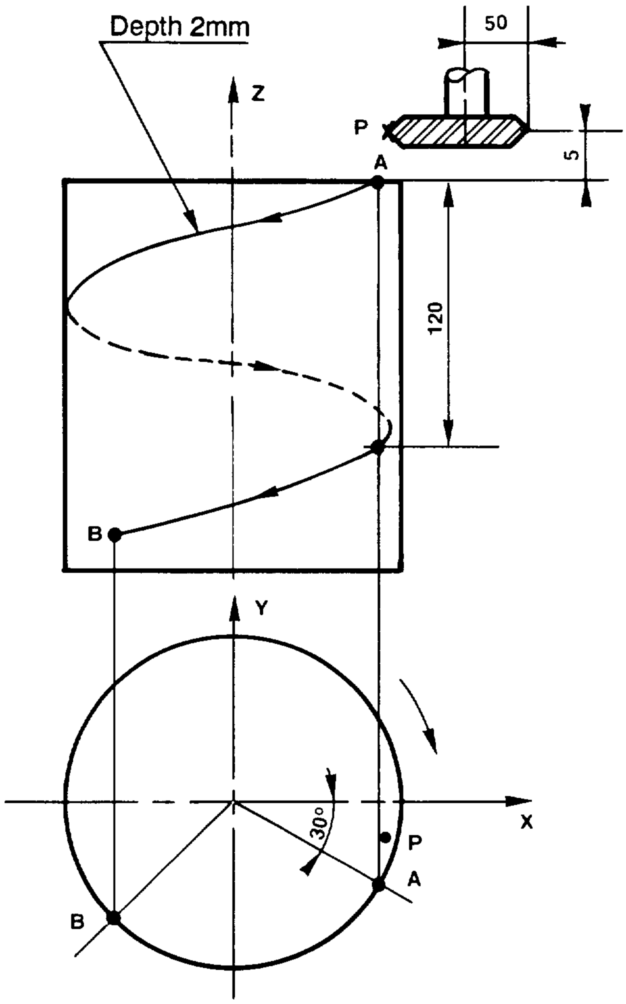
Z/K =	approx. number of revs to be done
K = 40	machine from A to B in .75 turn.
K = 20	machine from A to B in .75 turns.
K = 10	machine from A to B in 1.75 turns. Real pitch of helix : 11.4 mm
K = 5	machine from A to B in 3.75 turns Real pitch of helix : 5.33 mm
K = 3.8	machine from A to B in 5.75 turns. Real pitch of helix : 3.48 mm

In all the above cases, the system will deduce from the given start and end points that 0.75 of a turn must figure in the calculation of the true pitch. It uses the given values of Z and K, to derive the number of complete turns that are additionally required.

4.1.3.2 – Tool radius compensation

Tool radius compensation is allowed in helical interpolation mode (see tool radius compensation).

EXAMPLE : Machining a helix on a 100 mm radius cylinder



Depth of groove, 2 mm

Real pitch 120 mm

Positioning plane in Z = +5 mm

Point A : –30° at the Z0 plane

End of helix plane Z = –155 mm

To find point B with respect to A :

$\frac{360° \times 155}{120} = 465°$, i.e. 1 rev and 105°

Point B = –105° + (–30°) = –135°

To find the start point P with respect to A :

$\frac{360° \times 5}{120} = 15°$

Point P = –30° + 15° = –15°

N 10	G		X 100	Y 100	Z 100						
N 20									T1	D1	M6
N 30									S800	M40	M3
N 40	G41		X 94.661	Y 25.364	Z 5						
N 50	G2		X–69.296	Y–69.296	Z–155	IO	JO	K120	F 300		
N 60	G1	G40	X–110	Y–110							M5
N 70	G				Z 100						
N 80			X 100								M2

4.2 - PROFILE GEOMETRY PROGRAMMING (PGP)

The NUM 720F allows the workpiece program to be written directly from the drawing dimensions. It calculates inter-connections and tangent contacts or intersection points which are often left undefined between two path elements on the workpiece (line-line, line-circle, circle-circle). This programming method is only possible in absolute mode (G90).

The classic ISO programming method (see 4.1) remains valid and can be used in conjunction with PGP.

Programming is carried out in blocks, each block comprising a geometric element (straight line, arc of a circle).

A geometric element can be defined completely in a block (eg. the end point of a straight line, or the end point of an arc with the coordinates of the centre), or the element may be only partially defined in a block.

If the element is only partially defined, information MUST be provided either in the following block, (or two blocks), which will enable the complete entity, (sequence of blocks), to be defined.

PGP is applied in the current interpolation plane in accordance with function G17 – G18 or G19. If the interpolation plane is to be switched, however, it MUST only be switched at a point which has been completely defined.

4.2.1 - List of the functions characterizing a geometric element

- X..Y..Z.. : coordinates of the end point of a straight line
- EA.. : angle of a straight line relative to
  the X axis in G17
  the Y axis in G19
  the Z axis in G18

REMARK :

The minimum angle programmable between two lines is equal to 1/84 radian.

- X..Y..Z.. : coordinates of the end point of a circle
- I..J..K.. : coordinates of the centre of a circle
- R.. : radius of a circle
- EB+.. : blend radius :
  The block in which this function is programmed is connected to the following block by a blend radius of the given value.
- EB– .. : chamfer :
  The block in which this function is programmed is connected to the following block by a chamfer of the given value.
- ET : Tangential element
  The block in which this function is programmed forms a tangent to the following block.

  The programming of ET is compulsory when it is the only function of the block which characterizes the geometric element. ie. a straight line with known starting point, tangential to the following circle (Fig. 10) or a straight line tangential to two circles (Fig. 14).

  In all other cases, programming of ET is optional.

- ES : Secant element

  The block in which this function is programmed will form a secant to the following block. When the intersection point is not explicity programmed, the ES function MUST be programmed in the prior block of the two.

- $E\left\{\begin{matrix}+\\-\end{matrix}\right\}$: Discriminant

  Discriminant E+ or E− is used to choose between two mathematical solutions possible.

  The discriminant may be combined with functions ET and ES :
  EXAMPLE : ES− is equivalent to ES E−
  ET+ is equivalent to ET E+

  For a line–circle or a circle–circle intersection, there are only two possible solutions and the discriminant MUST be programmed.

  For tangential elements, there are several possible solutions : to limit these, the system executes only "continuous" tangents (ie. not turning back.) thus reducing the maximum number of solutions to two.

When two solutions are possible, one involves the creation of an arc of less than 180° and the other, an arc greater than 180°. Programming of the discriminant is optional. By default, the system will chooses the solution which gives the smaller arc.

The only exception is a circle whose centre lies inside the following circle, and which is characterized only by the coordinates of this centre and by the fact that it is tangential to the following circle (Fig. 5b, 12b, 23b). See paragraph 4.2.3.2.

4.2.2. – Entity programming – Choosing the discriminant

The group of blocks required by the system to calculate the coordinates of a geometric element (end point and/or centre of the circle) constitute a "geometric entity" originating from the starting point of its first block. This point can be :

- either fully defined in the preceding block,
- or already calculated by the system, given that the first block of a geometric entity may be the last block of the preceding entity.

Where a discriminant is used to define an element of a geometric entity, it must be programmed in the first block of this entity. The + and − signs will specify the positions of a characteristic point (intersection, tangent, circle centre) of the desired solution, with respect to a straight line (D) of known gradient.

NOTE :

The straight line (D) is :

- *the straight line, as defined by its angle EA..., (if one of the elements of the geometric entity is defined in this way,*
- *the straight line connecting a known point of the first element to a known point of the last element of the geometric entity (orientated from the first towards the last). This known point is normally the centre of a circle programmed by I..K.., or, by default, another programmed point (the start point of the first element or end point of the last element).*

There are two possibilities :

- *the points characterizing the two possible solutions are situated on the straight line (D) :*
- . *E+ defines the point closest to + ∞ on the straight line.*
- . *E– defines the point closest to – ∞ on the straight line.*

- *the points characterizing the two possible solutions are situated on either side of the straight line (D) :*
- . *E+ defines the point to the left of (D)*
- . *E– defines the point to the right of (D)*

4.2.3 – Block format authorized in programming a profile

Symbols used in the following examples :

{ }	Choice of several functions or items of data
[· ·]	Optional function
/	Tangency symbol
≠	Intersection symbol

The following examples refer to the XY axes (G17).

4.2.3.1 – A geometric element completely defined in one block

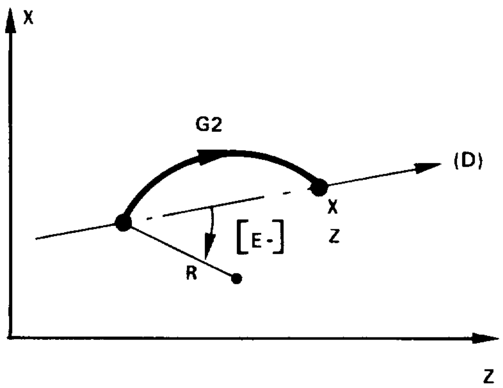
Straight line

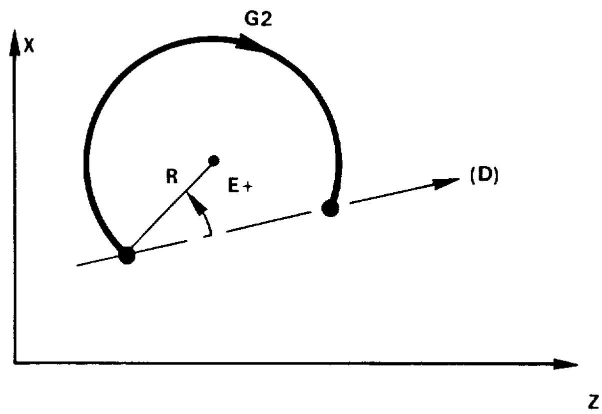
$$G01 \quad \left\{ \begin{array}{ll} X.. & \\ Y.. & \\ X.. & Y.. \\ EA.. & X.. \\ EA.. & Y.. \end{array} \right\}$$

Circle

$$G \left\{ \begin{array}{c} 2 \\ 3 \end{array} \right\} \quad X.. \quad Y.. \quad I.. \quad J..$$

$$G \left\{ \begin{array}{c} 2 \\ 3 \end{array} \right\} \quad X.. \quad Y.. \quad R.. \quad \left\{ E \pm \right\}$$





4.2.3.2 – A geometric element defined over more than one block.

- The first block is a straight line.
- The start point is completely defined.

Ist BLOCK	2nd BLOCK	3rd BLOCK	TYPE OF PROFILE	FIGURES
G1 EA.. ES	EA.. X.. Z		line ≠ line	FIG. 1
	EA..	$G\begin{Bmatrix}2\\3\end{Bmatrix} I..\ K.. \begin{Bmatrix}R..\\X..\ Z..\end{Bmatrix}$	line ≠ line/circ	FIG. 2
$G1\ EA..\ ES \begin{Bmatrix}+\\-\end{Bmatrix}$	$G\begin{Bmatrix}2\\3\end{Bmatrix} I..\ K.. \begin{Bmatrix}R..\\X..\ Z\end{Bmatrix}$		line ≠ circ	FIG. 3
	$G\begin{Bmatrix}2\\3\end{Bmatrix} I..\ K..$	G1 EA.. X.. Z..	line ≠ circ/line	FIG. 4
	$G\begin{Bmatrix}2\\3\end{Bmatrix} I..\ K.. \left[ET \begin{Bmatrix}+\\-\end{Bmatrix}\right]$	$G\begin{Bmatrix}2\\3\end{Bmatrix} I..\ K.. \begin{Bmatrix}R..\\X..\ Z..\end{Bmatrix}$	line ≠ circ/circ	FIG. 5
$G1\ EA..\ [ET]$	$G\begin{Bmatrix}2\\3\end{Bmatrix} I..\ K..$		line/circ	FIG. 6
	$G\begin{Bmatrix}2\\3\end{Bmatrix} R..$	G1 EA.. X.. Z..	line/circ/line	FIG. 7
$G1\ EA.. \left[ET \begin{Bmatrix}+\\-\end{Bmatrix}\right]$	$G\begin{Bmatrix}2\\3\end{Bmatrix} R..\ X..\ Z$		line/circ	FIG. 8
	$G\begin{Bmatrix}2\\3\end{Bmatrix} R..$	$G\begin{Bmatrix}2\\3\end{Bmatrix} I..\ K.. \begin{Bmatrix}R..\\X..\ Z..\end{Bmatrix}$	line/circ/circ	FIG. 9
G1 ET	$G\begin{Bmatrix}2\\3\end{Bmatrix} I..\ K.. \begin{Bmatrix}R..\\X..\ Z..\end{Bmatrix}$		line/circ	FIG. 10
	$G\begin{Bmatrix}2\\3\end{Bmatrix} I..\ K..$	G1 EA.. X.. Z..	line/circ/line	FIG. 11
	$G\begin{Bmatrix}2\\3\end{Bmatrix} I..\ K.. \left[ET \begin{Bmatrix}+\\-\end{Bmatrix}\right]$	$G\begin{Bmatrix}2\\3\end{Bmatrix} I..\ K.. \begin{Bmatrix}R..\\X..\ Z..\end{Bmatrix}$	line/circ/circ	FIG. 12

- The first block is a circle.
- The start point is completely defined.

Ist BLOCK	2nd BLOCK	3rd BLOCK	TYPE OF PROFILE	FIGURES
$G\begin{Bmatrix}2\\3\end{Bmatrix}$ I.. K.. [ET]	$G1\begin{Bmatrix}EA..\\X..\\Y..\\EA..\ X..\\EA..\ Z..\\X..\ Z..\end{Bmatrix}$		circ / line	FIG. 13
	G1 ET	$G\begin{Bmatrix}2\\3\end{Bmatrix}$ I.. K.. $\begin{Bmatrix}R..\\X..\ Z..\end{Bmatrix}$	circ / line / circ	FIG. 14
$G\begin{Bmatrix}2\\3\end{Bmatrix}$ I.. K.. $\left[ET\begin{Bmatrix}+\\-\end{Bmatrix}\right]$	$G\begin{Bmatrix}2\\3\end{Bmatrix}$ I.. K..		circ / circ	FIG. 15
	$G\begin{Bmatrix}2\\3\end{Bmatrix}$ R.. X.. Z..		circ / circ	FIG. 16
	$G\begin{Bmatrix}2\\3\end{Bmatrix}$ R..	G1 EA.. X.. Z..	circ / circ / line	FIG. 17
		$G\begin{Bmatrix}2\\3\end{Bmatrix}$ I.. K.. $\begin{Bmatrix}R..\\X..\ Z..\end{Bmatrix}$	circ / circ / circ	FIG. 18
$G\begin{Bmatrix}2\\3\end{Bmatrix}$ I.. K.. ES $\begin{Bmatrix}+\\-\end{Bmatrix}$	G1 EA.. X.. Z..		circ ≠ line	FI.G 19
	G1 EA..	$G\begin{Bmatrix}2\\3\end{Bmatrix}$ I.. K.. $\begin{Bmatrix}R..\\X..\ Z..\end{Bmatrix}$	circ ≠ line / circ	FIG. 20
	$G\begin{Bmatrix}2\\3\end{Bmatrix}$ I.. K.. $\begin{Bmatrix}R..\\X..\ Z..\end{Bmatrix}$		circ ≠ circ	FIG. 21
	$G\begin{Bmatrix}2\\3\end{Bmatrix}$ I.. K..	G1 EA.. X.. Z..	circ ≠ circ / line	FIG. 22
	$G\begin{Bmatrix}2\\3\end{Bmatrix}$ I.. K.. $\left[ET\begin{Bmatrix}+\\-\end{Bmatrix}\right]$	$G\begin{Bmatrix}2\\3\end{Bmatrix}$ I.. K.. $\begin{Bmatrix}R..\\X..\ Z..\end{Bmatrix}$	circ ≠ circ / circ	FIG. 23
$G\begin{Bmatrix}2\\3\end{Bmatrix}$ R.. $\left[ET\begin{Bmatrix}+\\-\end{Bmatrix}\right]$	G1 EA.. X.. Z..		circ / line	FIG. 24
	G1 EA..	$G\begin{Bmatrix}2\\3\end{Bmatrix}$ I.. K.. $\begin{Bmatrix}R..\\X..\ Z..\end{Bmatrix}$	circ / line / circ	FIG. 25
	$G\begin{Bmatrix}2\\3\end{Bmatrix}$ I.. K.. $\begin{Bmatrix}R..\\X..\ Z..\end{Bmatrix}$		circ / circ	FIG. 26
	$G\begin{Bmatrix}2\\3\end{Bmatrix}$ I.. K..	G1 EA.. X.. Z..	circ / circ / line	FIG. 27
	$G\begin{Bmatrix}2\\3\end{Bmatrix}$ I.. K.. $\left[ET\begin{Bmatrix}+\\-\end{Bmatrix}\right]$	$G\begin{Bmatrix}2\\3\end{Bmatrix}$ I.. K.. $\begin{Bmatrix}R..\\X..\ Z..\end{Bmatrix}$	circ / circ / circ	FIG. 28

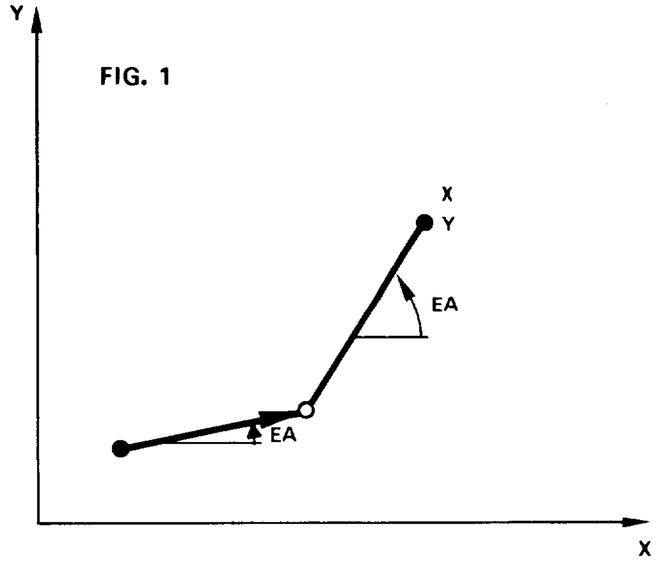


FIG. 1

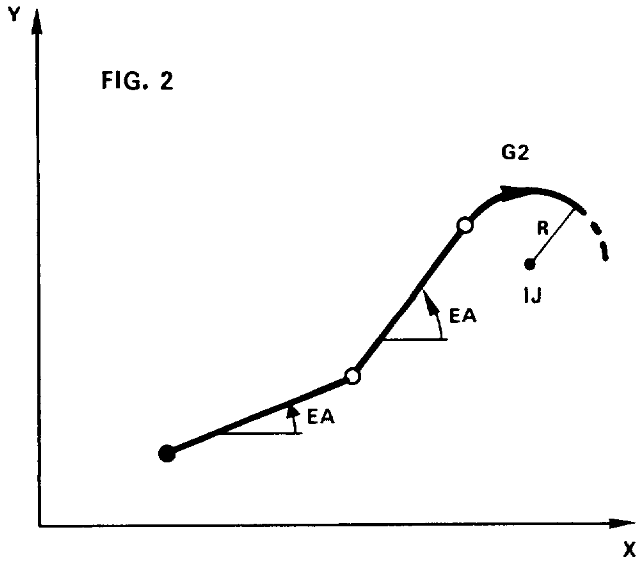


FIG. 2

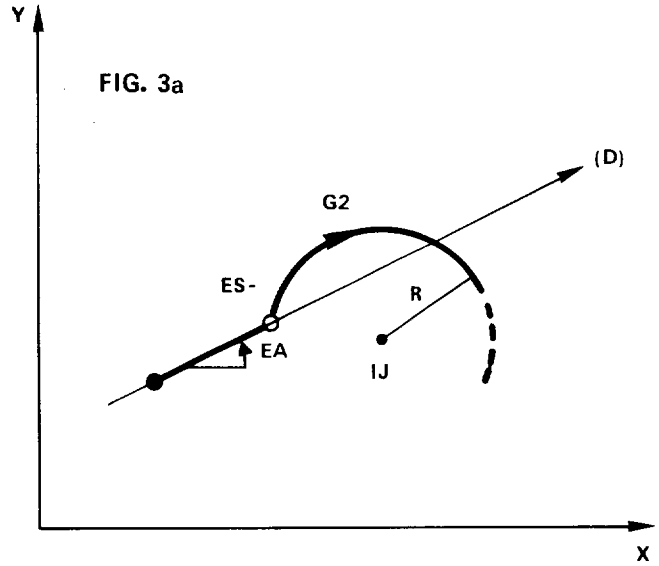


FIG. 3a

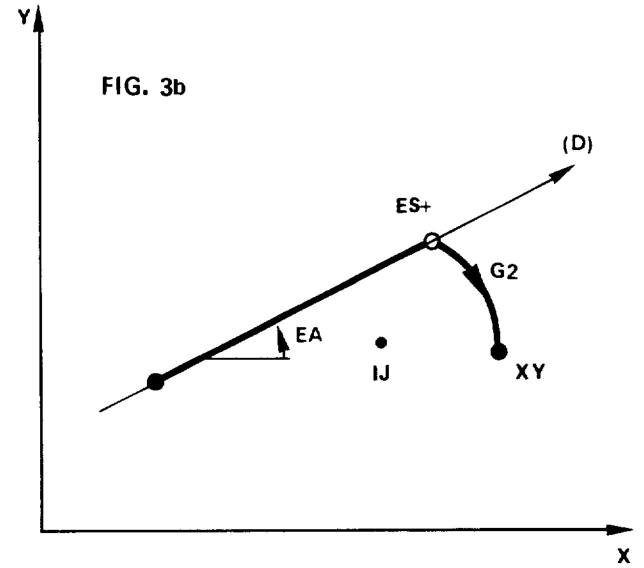


FIG. 3b

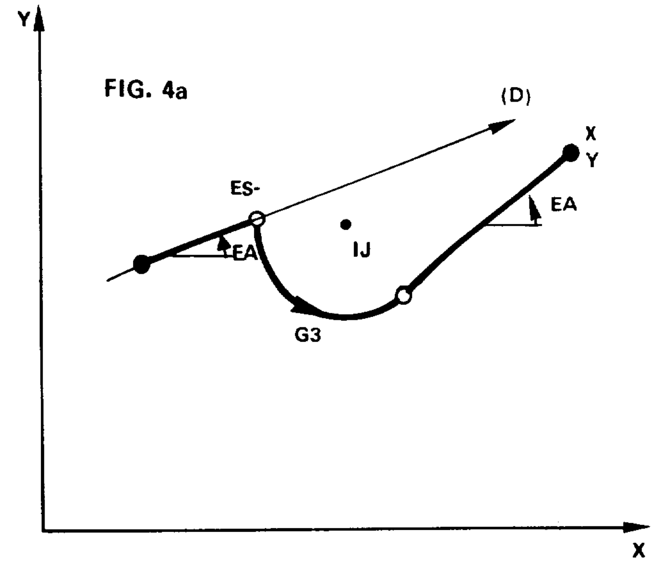


FIG. 4a

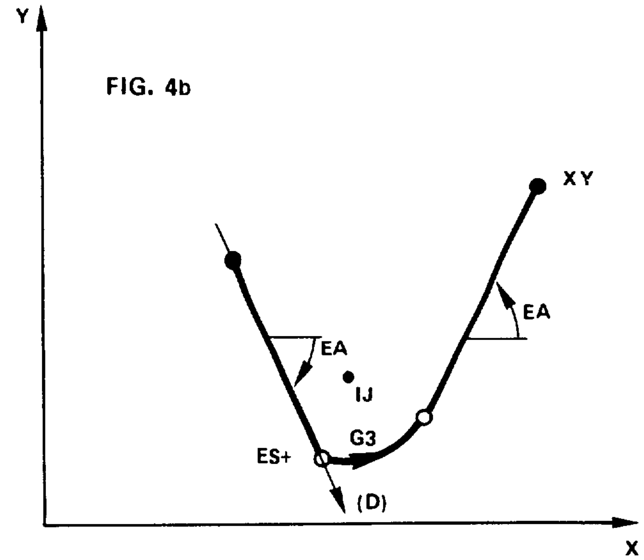


FIG. 4b

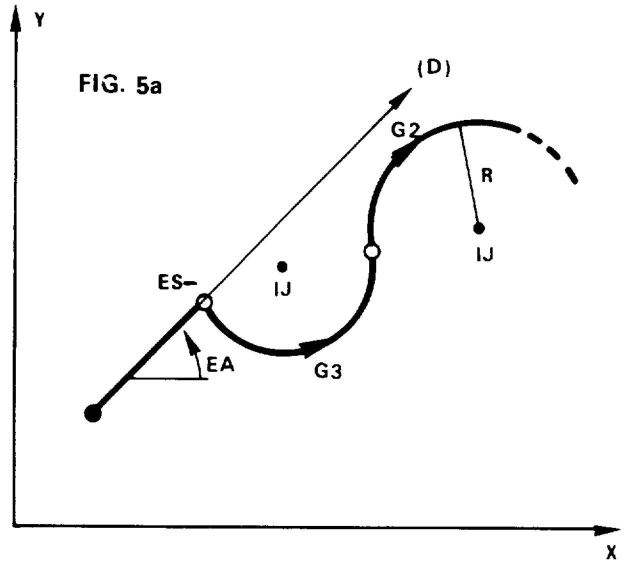


FIG. 5a

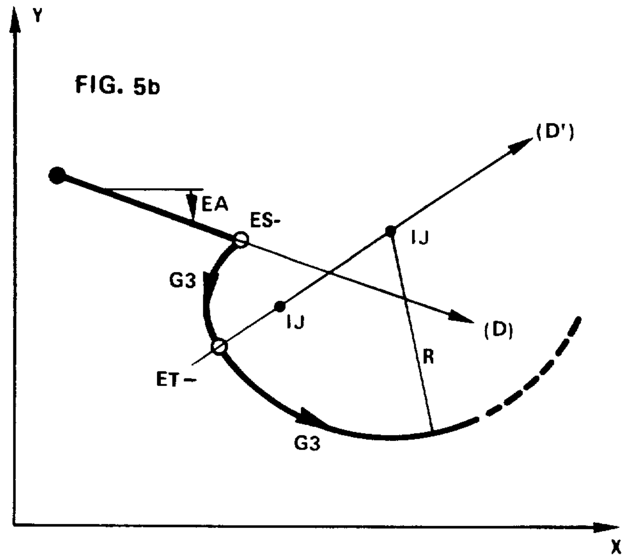


FIG. 5b

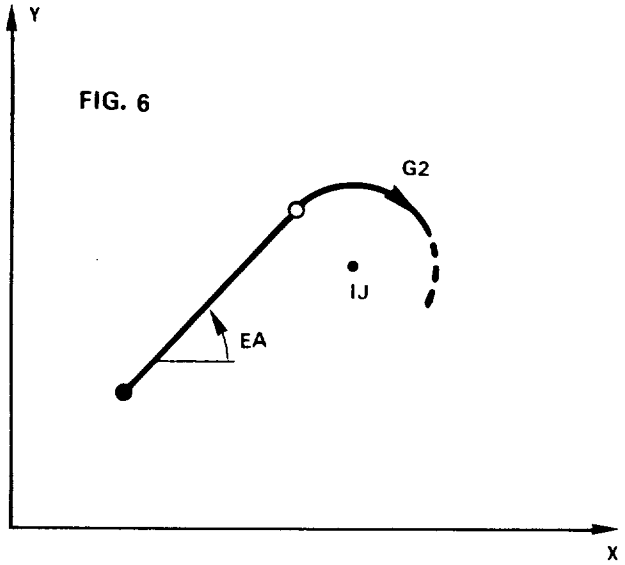


FIG. 6

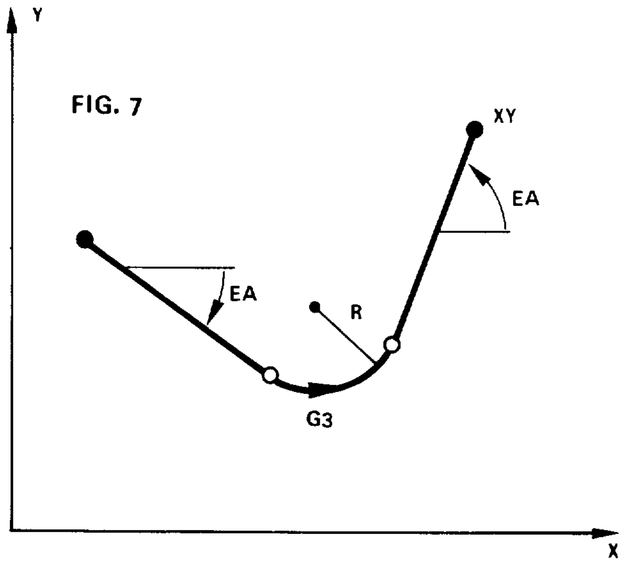


FIG. 7

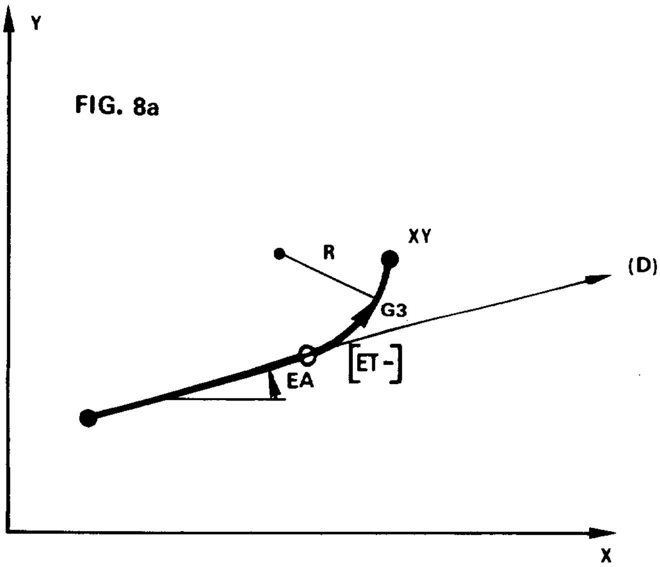


FIG. 8a

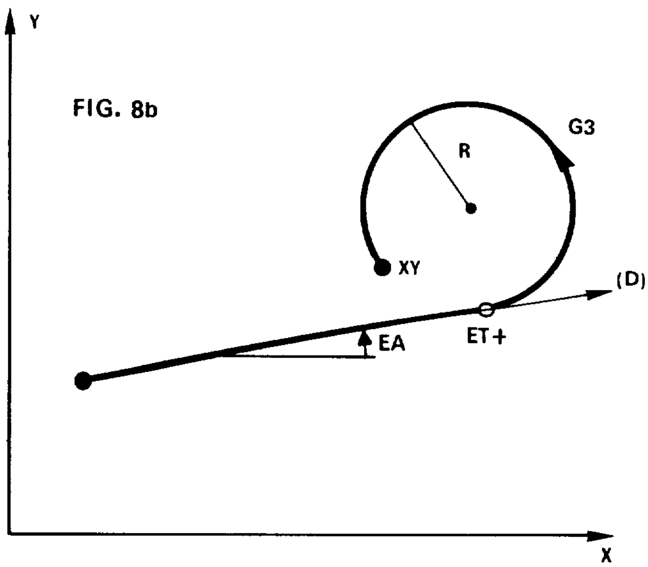


FIG. 8b

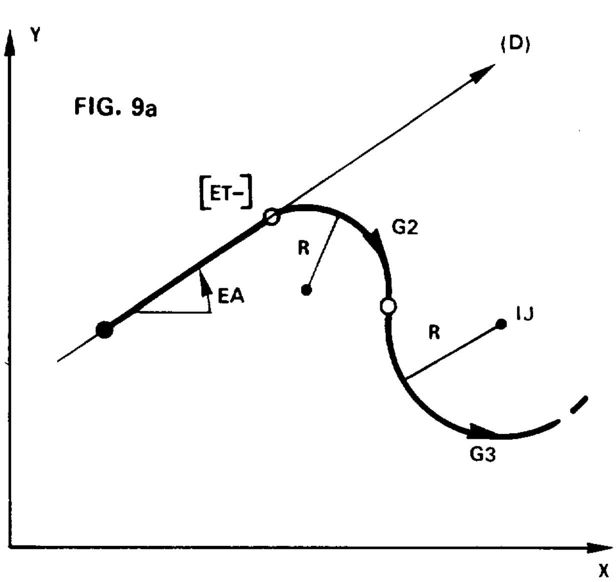


FIG. 9a

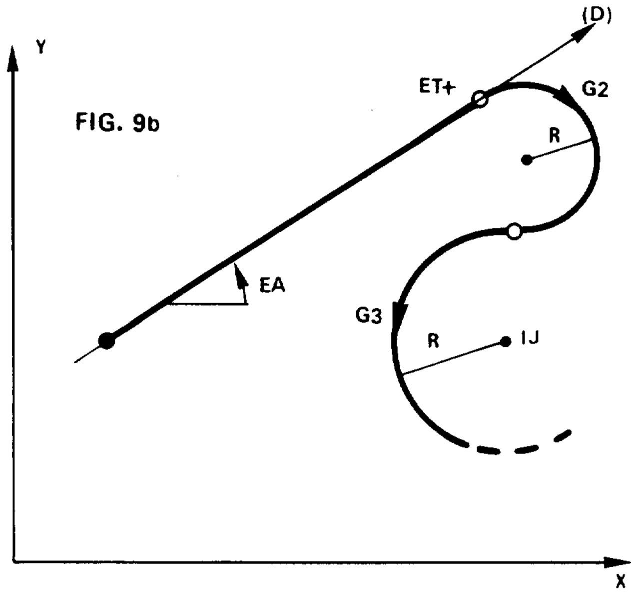


FIG. 9b

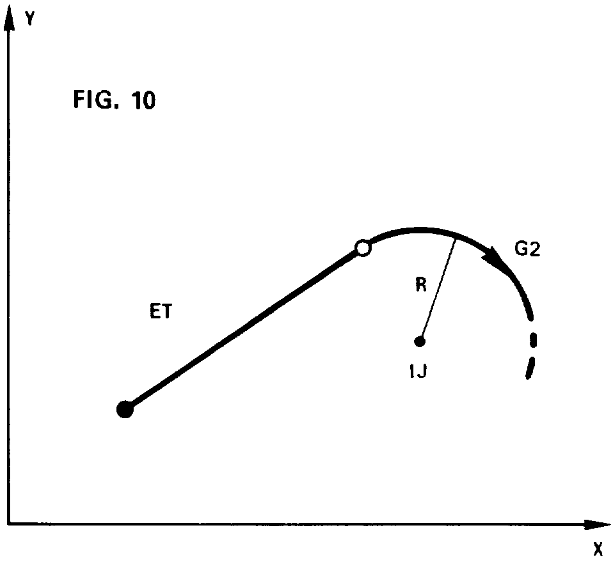


FIG. 10

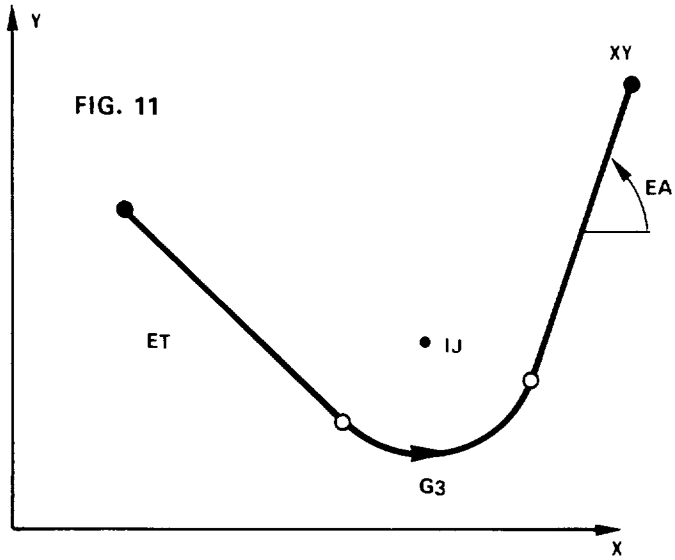


FIG. 11

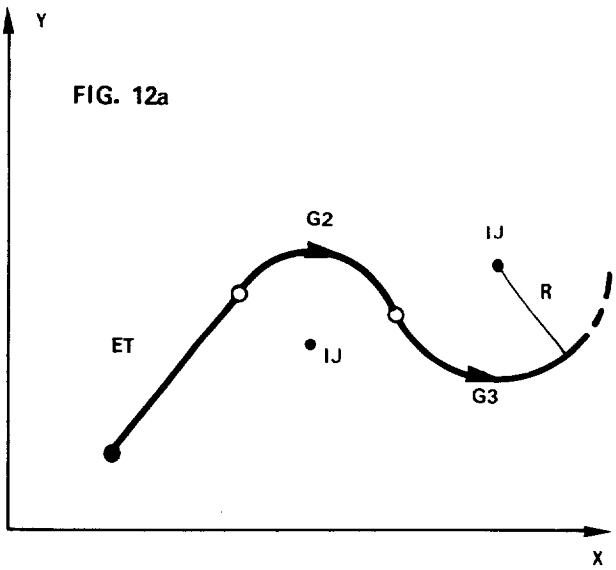


FIG. 12a

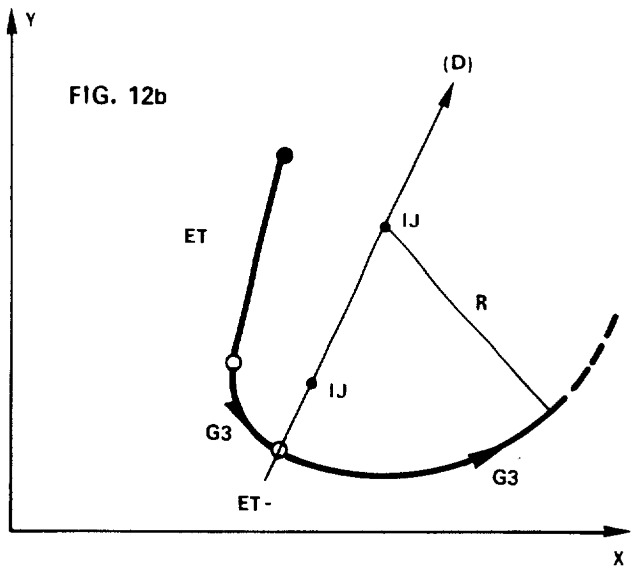


FIG. 12b

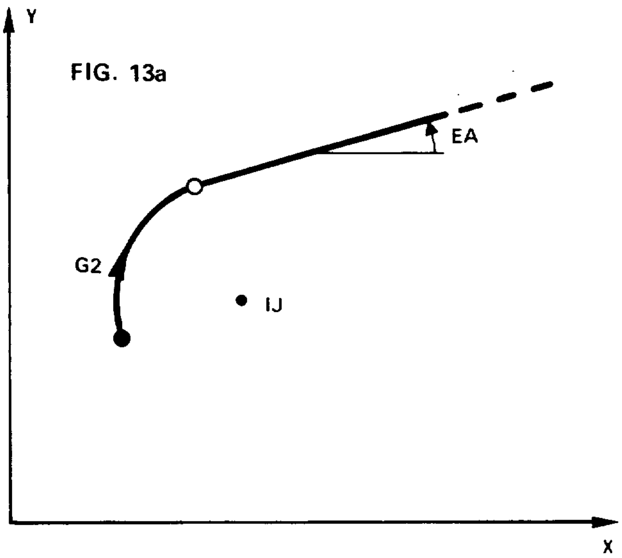


FIG. 13a

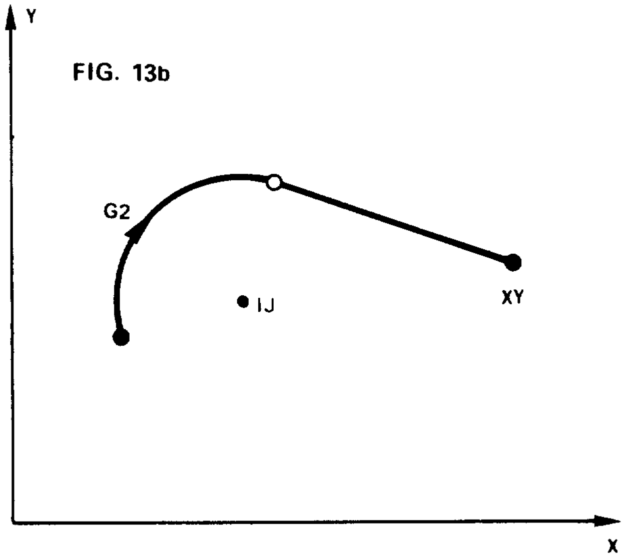


FIG. 13b

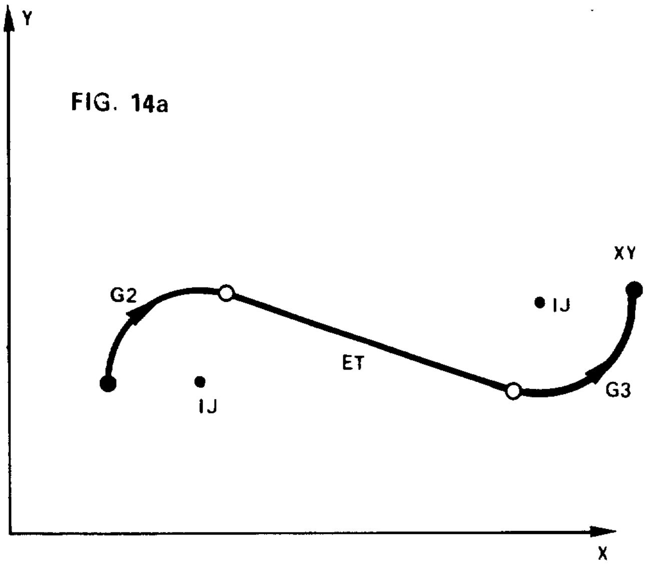


FIG. 14a

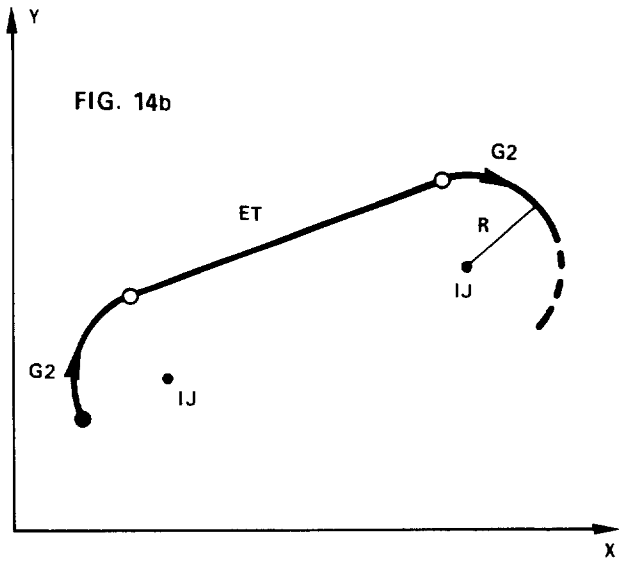


FIG. 14b

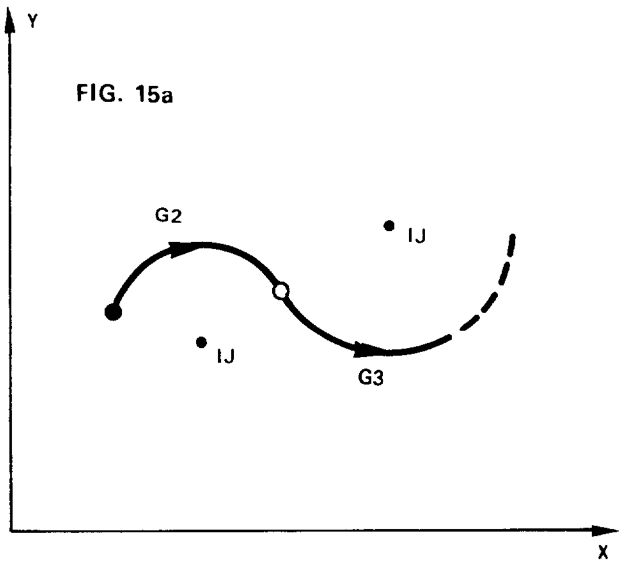


FIG. 15a

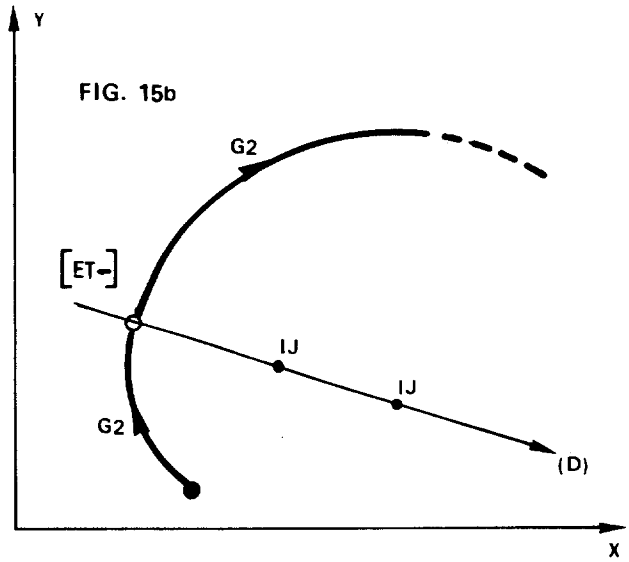


FIG. 15b

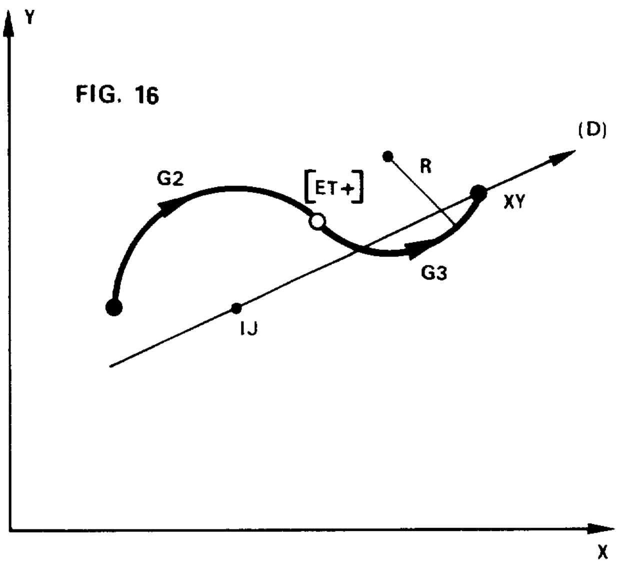


FIG. 16

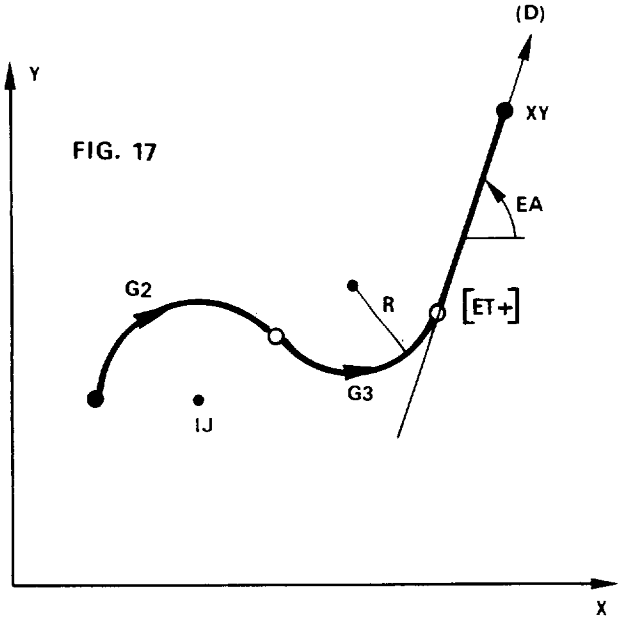


FIG. 17

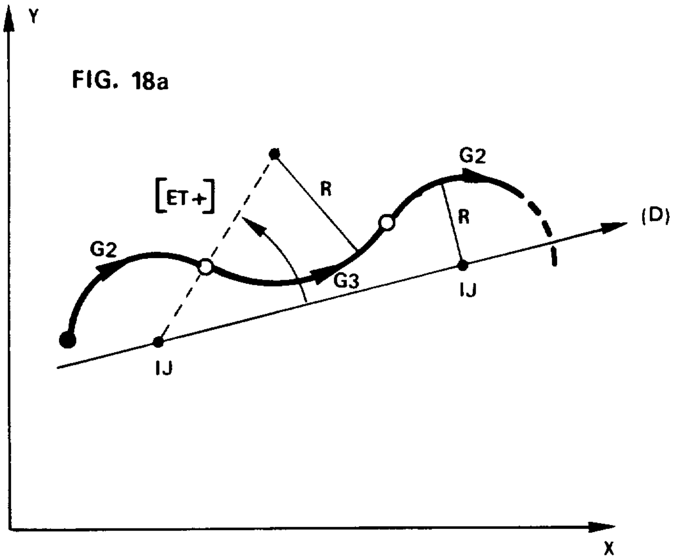


FIG. 18a

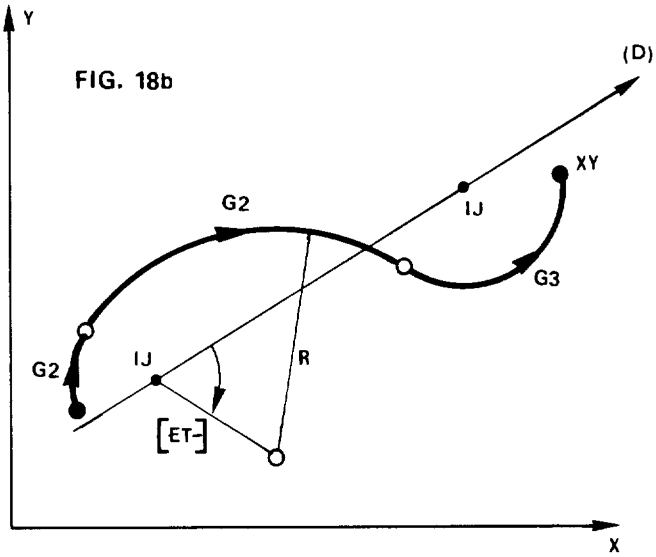


FIG. 18b

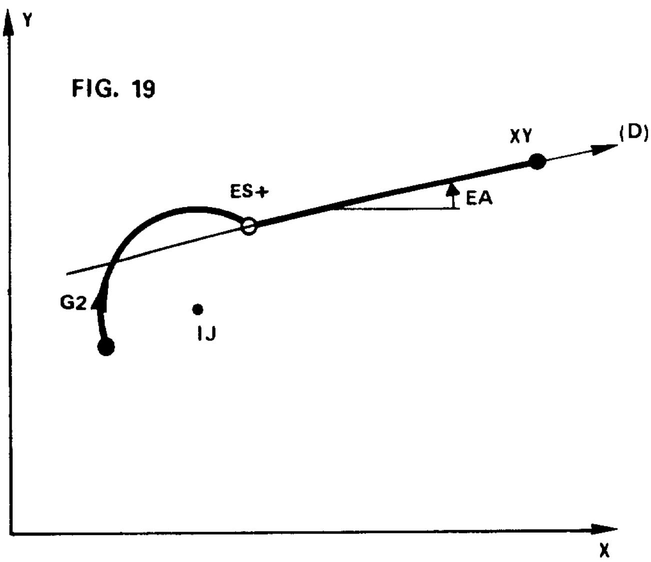


FIG. 19

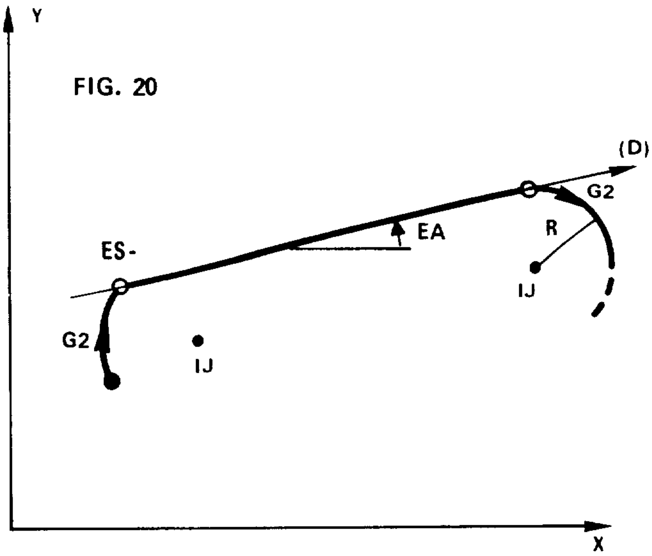


FIG. 20

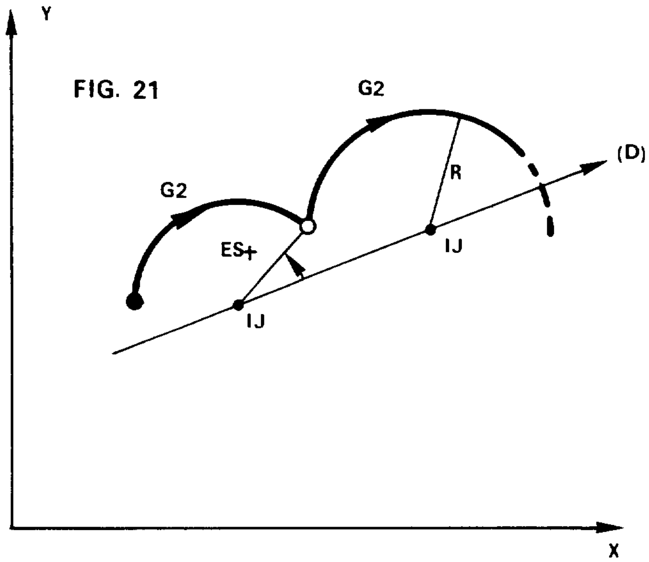


FIG. 21

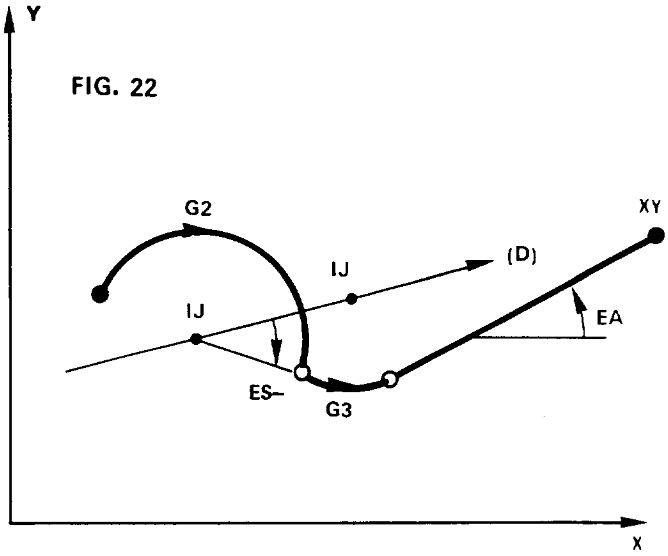


FIG. 22

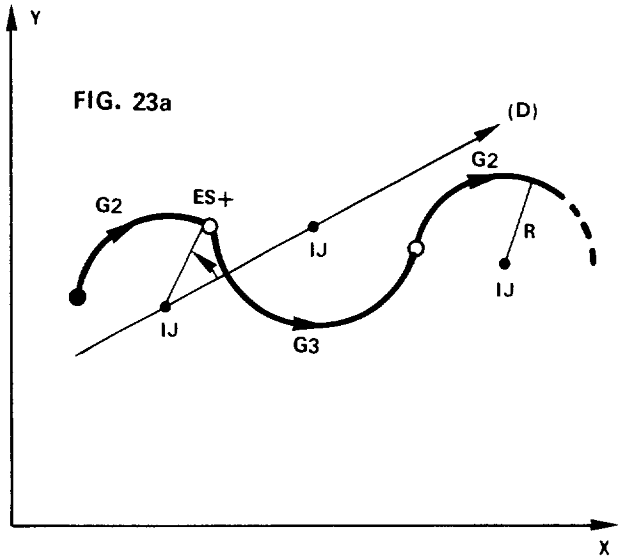


FIG. 23a

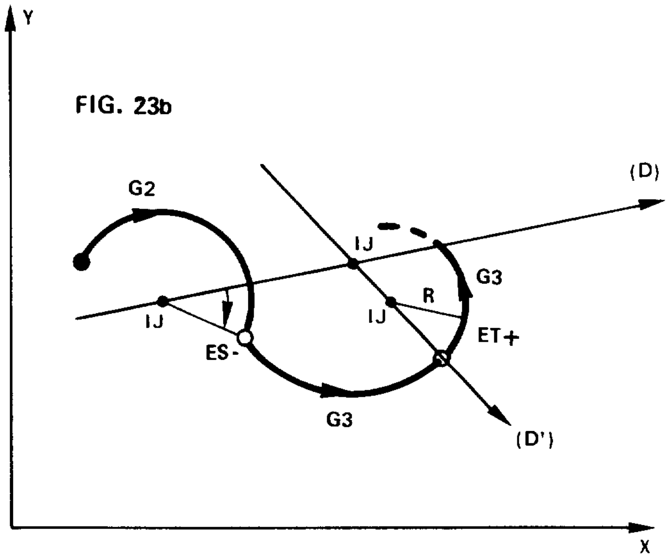


FIG. 23b

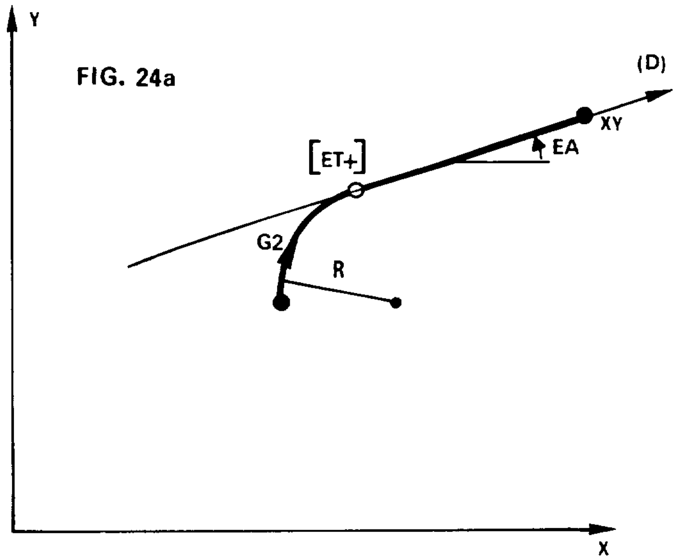


FIG. 24a

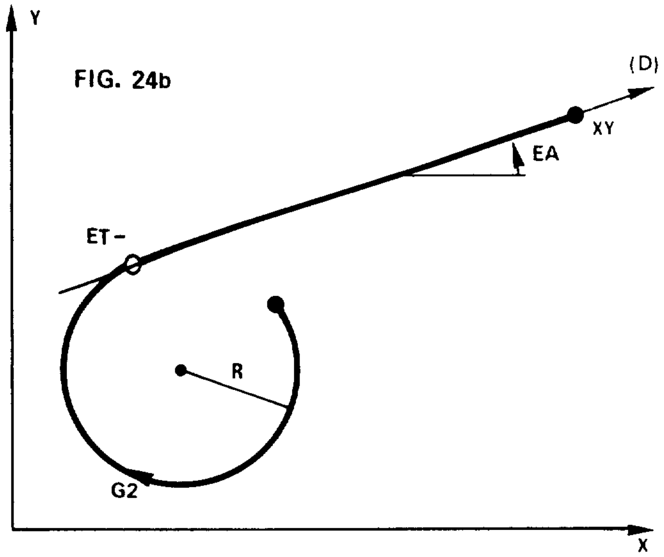


FIG. 24b

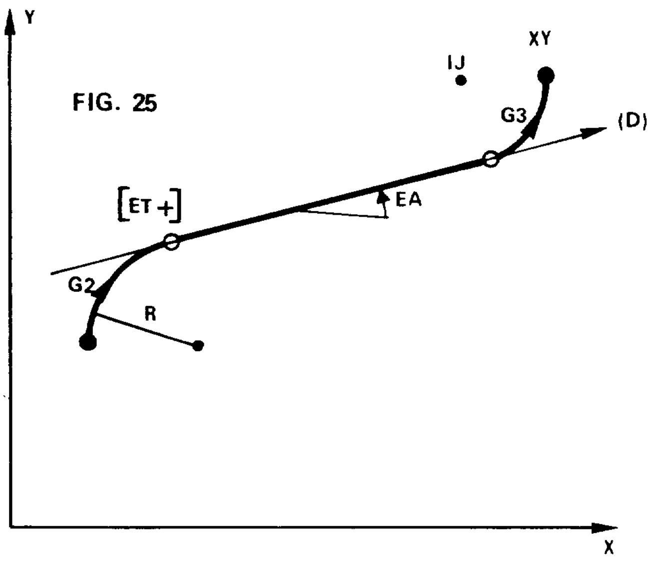


FIG. 25

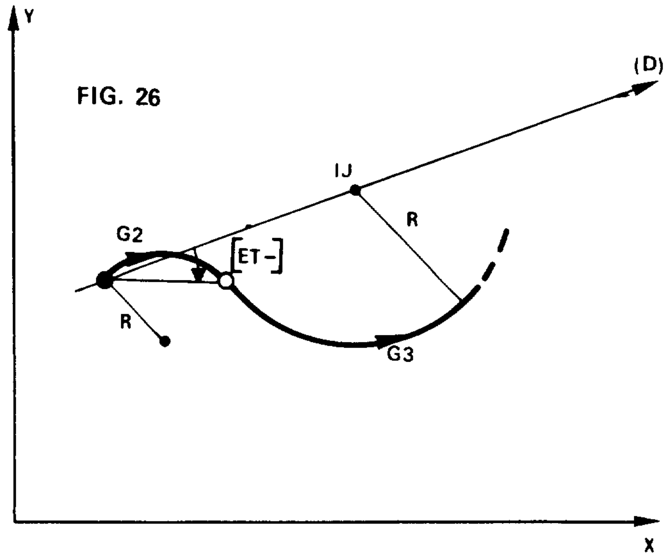


FIG. 26

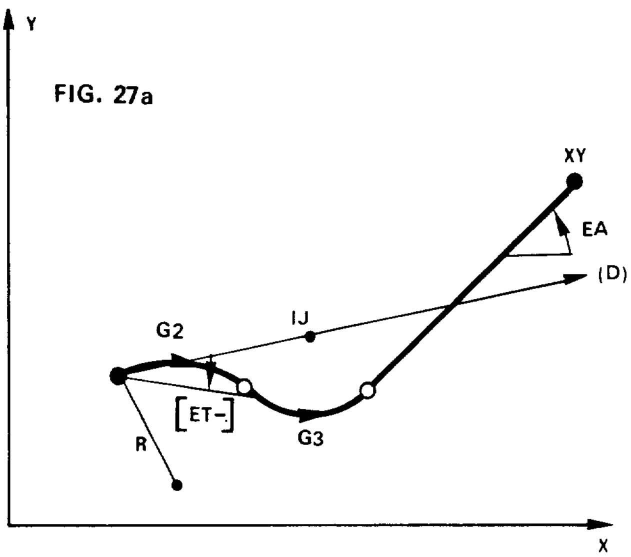


FIG. 27a

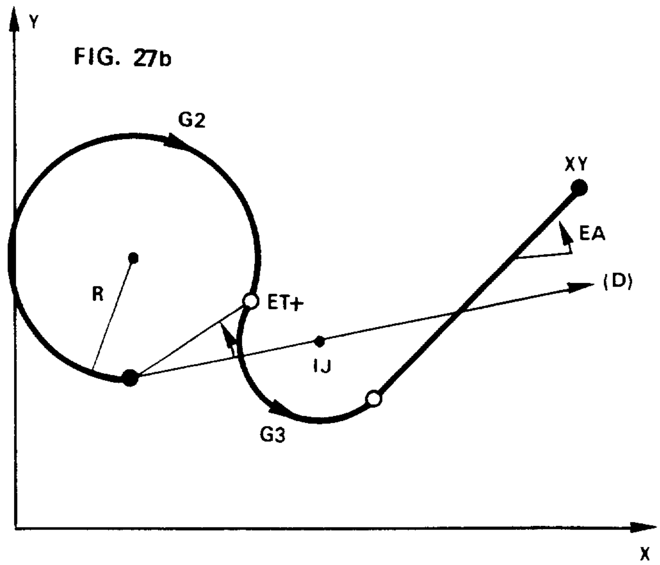


FIG. 27b

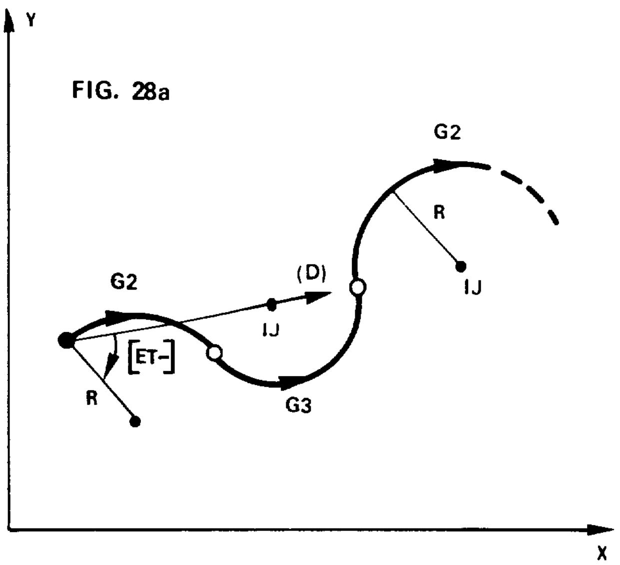


FIG. 28a

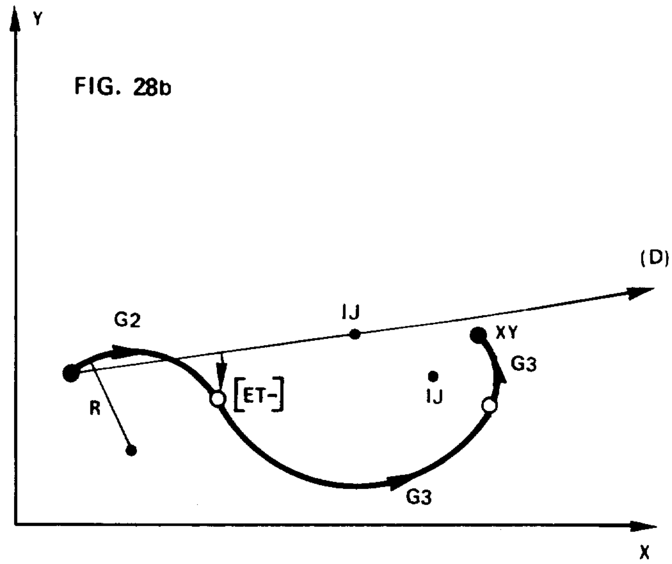


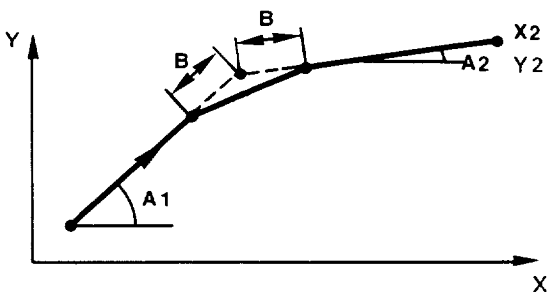
FIG. 28b

4.2.3.3 - Programming chamfers and blends

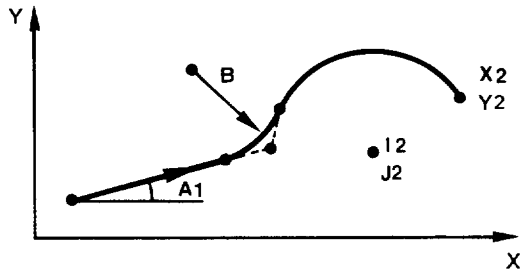
A chamfer can be inserted between two successive straight lines. A blend radius may be inserted between any two elements, line or circle.

The chamfer is programmed by code EB- and the value. The blend radius is programmed by code EB+ and the value.

chamfer



blend radius

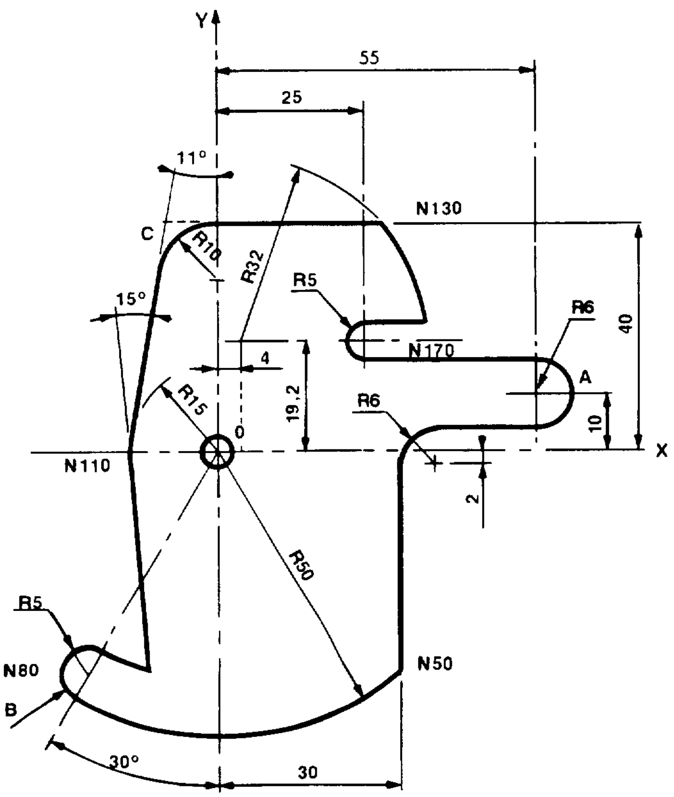


G1 EA1 ES EB-
G1 EA2 X2 Y2

G1 EA1 ES- EB+
G2 I2 J2 X2 Y2

4.3 - PROGRAMMING EXAMPLE

Profile definition from A to B then C and A



N10 X61 Y10 (Point A)	
N20 G2 I55 J10	
N30 G1 ET	Fig.14
N40 G3 I36 J-2 R6	
N50 G1 EA-90 ES+	Fig.3
N60 G2 I J R50	
N70 L1 = S30*45	
L2=L1*L1 L2=45*45-L2	
L2 = RL2	
N80 G2 I-L1 J-L2	
N90 G3 I J ES-	Fig.15
N100 G1 EA94	
N110 G2 I J R15	Fig.20
N120 G1 EA79 Y40 EB10	
N130 EA ES+	
N140 G2 I4 J19.2 R32 ES-	Fig.3
N150 G1 EA 180	
N160 G3 I25 J21 R5	
N170 G1 ET	Fig.10
N180 G2 I55 J10 X61 Y10 (Point A)	

Block N70 : computation of centre B using program variables.
L1 = X value of centre of circle B
L2 = Y value of centre of circle B

Block N120 : blend radius of 10 at point C.

5 - MOVEMENT OF A ROTARY AXIS (OPTIONAL)

5.1 - "MODULO 360 DEGREES" SERVO-CONTROLLED AXIS

The position loop is closed by the numerical control. The angular dimension can be programmed in either absolute or incremental, and is expressed in degrees.

A033 } Value of angle of rotation (0° to 360°)
B033
C033

Maximum movement in a single block is 1 revolution. The direction of rotation is determined by the sign.

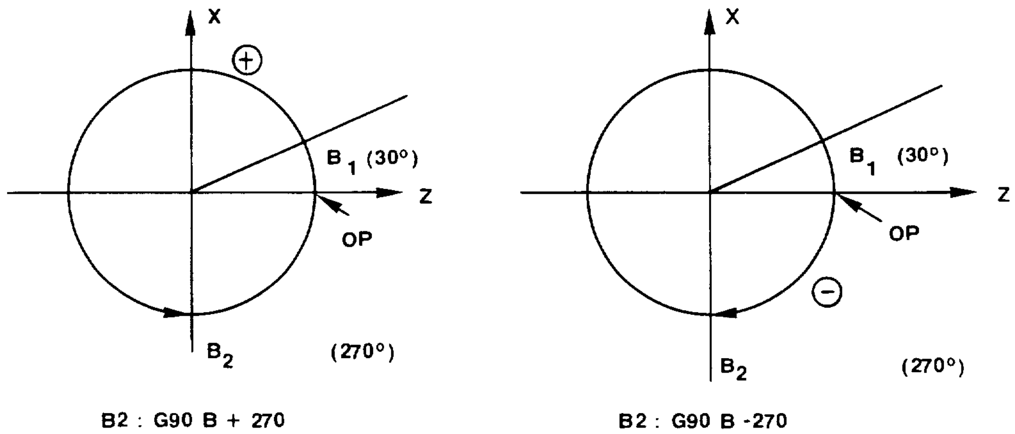
DAT 1, DAT 2 can be used to define the program origin.

DAT 3 for axis A, B or C.

Absolute programming

The dimension defined with respect to the program origin (Op) is expressed in absolute mode.

The sign indicates the direction of rotation of the tool with respect to the workpiece.



Incremental programming

The dimension programmed indicates the amplitude of tool rotation relative to the starting position. The sign indicates the direction of rotation. (See figure above).

ptB_2 G91 B + 240   ptB_2 G91 B − 120

- Table rotation can be simultaneous with movements of the linear axes in G0.
- Moves at feedrates can be programmed in V/L with G1 and G93, or in °/min. in G94.

NOTE :

The presence or absence of a sign immediately following the letter address will dictate the direction of rotation. Thus the mathematical value of a parameter which is substituted for the dimension will not be taken into account.
ie. AL0 will cause a positive direction of rotation, even if the value of L0 is negative.
Similarly, A−L0 will cause a negative direction of rotation even if L0 is positive.

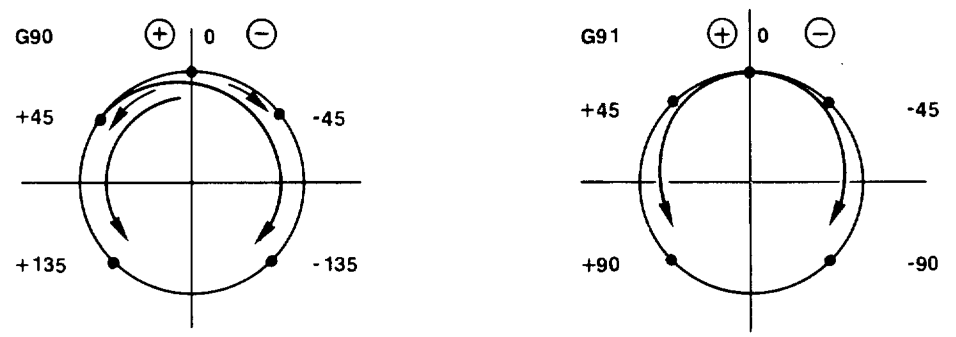
5.2 - SERVO-CONTROLLED ROTARY AXES WITH RESTRICTED MOVEMENT

This type of rotary axis is set up as a linear servo-controlled axis. It is thus subject to the same programming rules, but with the angle expressed in degrees.

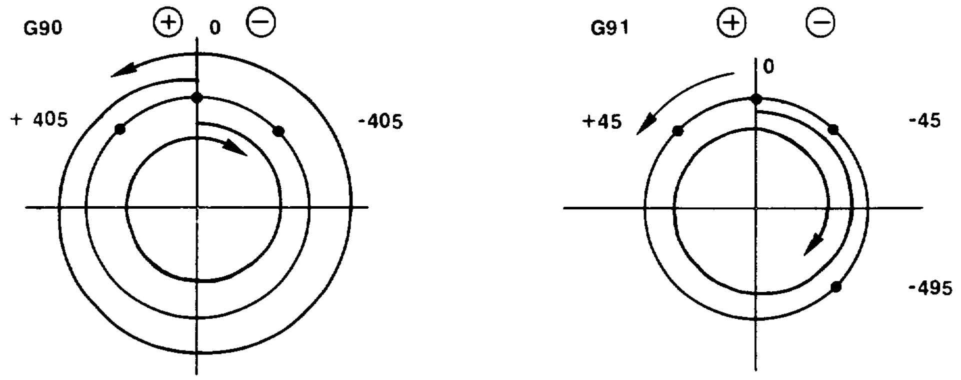
This definition of a rotary axis could be used for example for twist and tilt head applications :

- with movement of less than 360° relative to a specific position,
- which are to be controlled for more than one revolution.

EXAMPLE : less than 1 revolution



EXAMPLE : more than one revolution



6 – TOOL PROGRAMMING

The tool number is programmed with the T address (3 digits). Its' compensation number is specified with the D address (2 digits).

Any compensator can be assigned to any tool.

6.1 – PROGRAMMING THE TOOL NUMBER

T... the tool number is modal. (Max. value 255)
This code is normally used in conjunction with M6, to carry out a manual or automatic tool change.

6.2 – TOOL COMPENSATIONS

The system has 32 compensation triplets (L, R, @) which can be used to compensate for tool length and tool radius.

6.2.1 – Length compensation

Programming D.. ENABLES the associated length compensation along the tool axis. (Normally Z, but could be X or Y, depending on the tool orientation, see 6.3). No axis movements are generated by changing the D compensation. The length compensation will ONLY be taken into account during the next programmed move of the tool axis.

Maximum compensation value L : $\pm$ 9999.999 mm

The length compensation is cancelled by programming compensator D0, whose contents are always zero. Length compensations are not taken into account in G52 blocks.

Address D and its value are modal.

EXAMPLE :

N10	X1	Y1	Z1		
N20	X2	Y2		D5	Enabling compensator 5
N30	X3	Y3	Z3		Length compensation 5 effective on Z3
N40	X4	Y4	Z4	D8	Enabling compensator 8. Length compensation 8 effective on Z4
N50	X5	Y5	Z5	D0	Cancellation of length compensation on Z5

The length compensation applies along the tool axis as defined by G16 P+, Q+, R+). Tool LENGTH variations (ie. a change of compensation number or wear compensation or change in tool orientation) are applied :

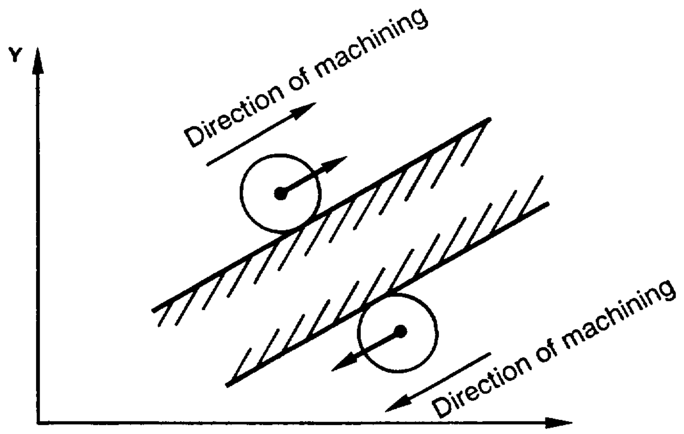
- if the tool axis is programmed,
- and if the tool axis is perpendicular to the interpolation plane.

6.2.2 – Radius compensation in the plane

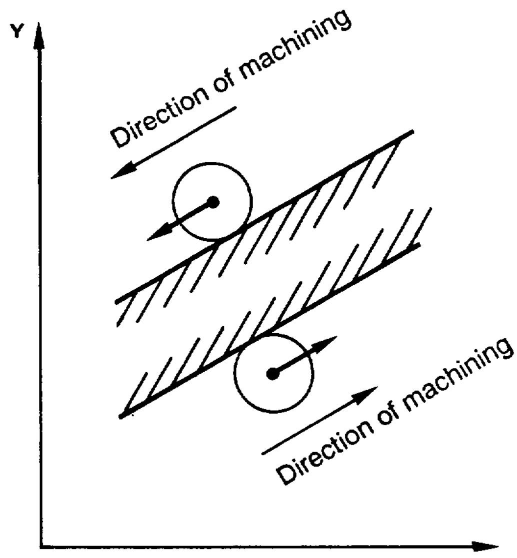
Radius compensation is applied in one of the planes of the coordinate system as defined by functions G17, G18 or G19.

Having specified a D.. compensator, the radius compensation is enabled either by G41 (for compensation to the left of the profile to be machined), or by G42 (for compensation to the right). It is cancelled by G40.

The maximum value of radius compensation is 9999.999 mm.



G41



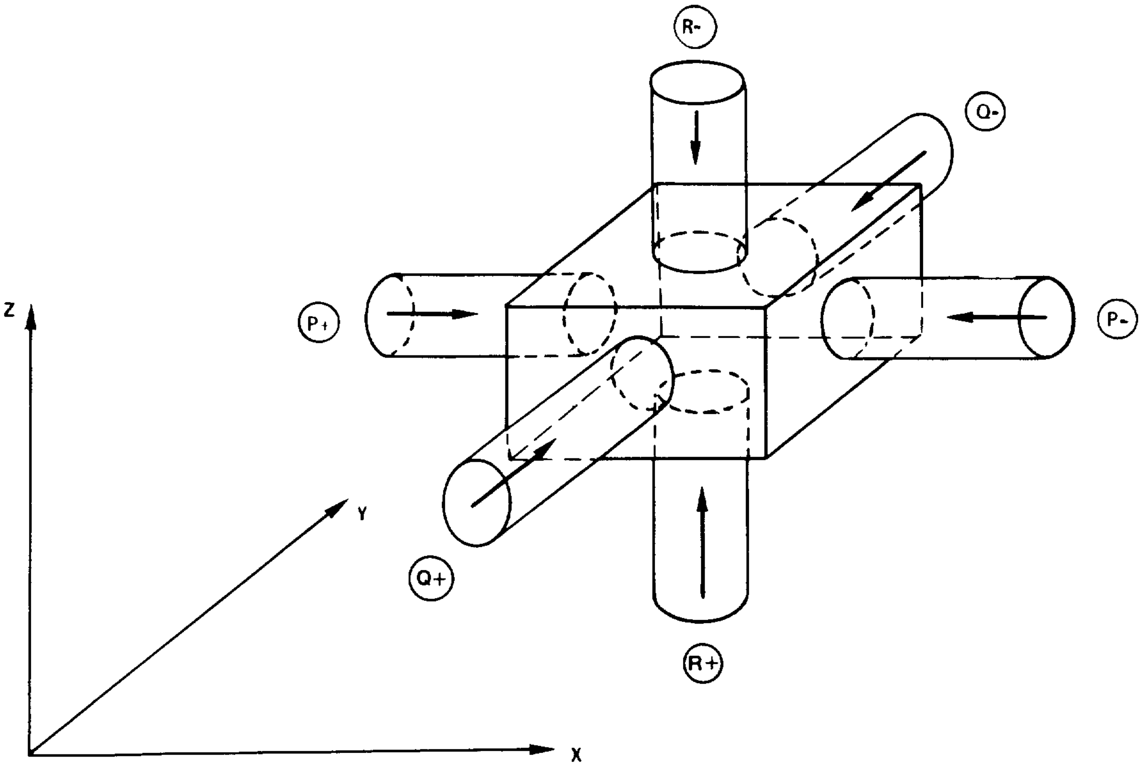
G42

NOTE :

If the compensation number (D..) is changed while radius compensation mode, (G41/42 active), or a new length or wear compensation value is input, only the new length compensation value will be taken into account (as per the conditions listed in paragraph 6.2.1). The radius compensation will stay at its initial value. The new value will only be taken up, after exiting with G40 and re–entering with G41 or G42.

6.3 – ORIENTATION OF THE TOOL AXIS

Machines with interchangeable heads or right–angle attachments allow the tool to assume between 2 and 6 different positions. In these 6 cases, the tool axis is always parallel to one of the axes of the coordinate system.



The tool vector is orientated from the tip towards its reference. Its sign is given by the reference coordinate system.

Function G16, associated with addresses P, Q and R, is used to define the tool axis.

Format :

G16 ↓ Tool axis orientation	P ±	P+	orientation	X+
		P–	orientation	X–
	Q ±	Q+	orientation	Y+
		Q–	orientation	Y–
	R ±	R+	orientation	Z+
		R–	orientation	Z–

It is recommended to be in G40 and G80 modes when defining the tool axis.

- By default, the system initialize the tool axis in the Z + direction,
- The block defining the tool axis may also include axis movements, auxiliary M functions and S and T.
- When the tool axis is changed, there is no movements of the axes concerned until they are reprogrammed.

NOTES

7 - USING TOOL COMPENSATION

Tool radius compensation is effected in the planes of the coordinate system. The compensation value can be set equal to the tool radius. If this is so, then it is possible to program the real profile of the workpiece.

The compensation value may also be set equal to the difference in radii between the real tool and a theoretical tool, as is often the case in computer assisted programming.

A compensation can be used to negate the effects of cutter deflection or similar machining problems. It is of course, particularly useful in automatically computing cutter paths and intersections between gradients, irrespective of whether the tool is larger or smaller than nominal size.

7.1 - COMPENSATION IN THE INTERPOLATION PLANE

The following data must be supplied to the system :

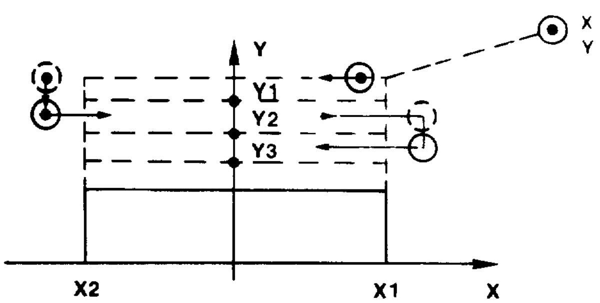
Format :

1	G17	X	Y	Compensation in the X Y plane
	G18	Z	X	Compensation in the Z X plane
	G19	Y	Z	Compensation in the Y Z plane
			D..	Compensation number
	G41			Compensation to the left
	G42			Compensation to the right
2	G40	XY	(YZ) (ZX)	Compensation cancellation

Functions G41 or G42 need not necessarily be accompanied in the same block by the two axes in the compensation plane ; one axis is sufficient, the other being compensated automatically.

The compensation sense may be changed (left-right, G41 - G42) without leaving compensation mode (G40).

Example of pendulum milling

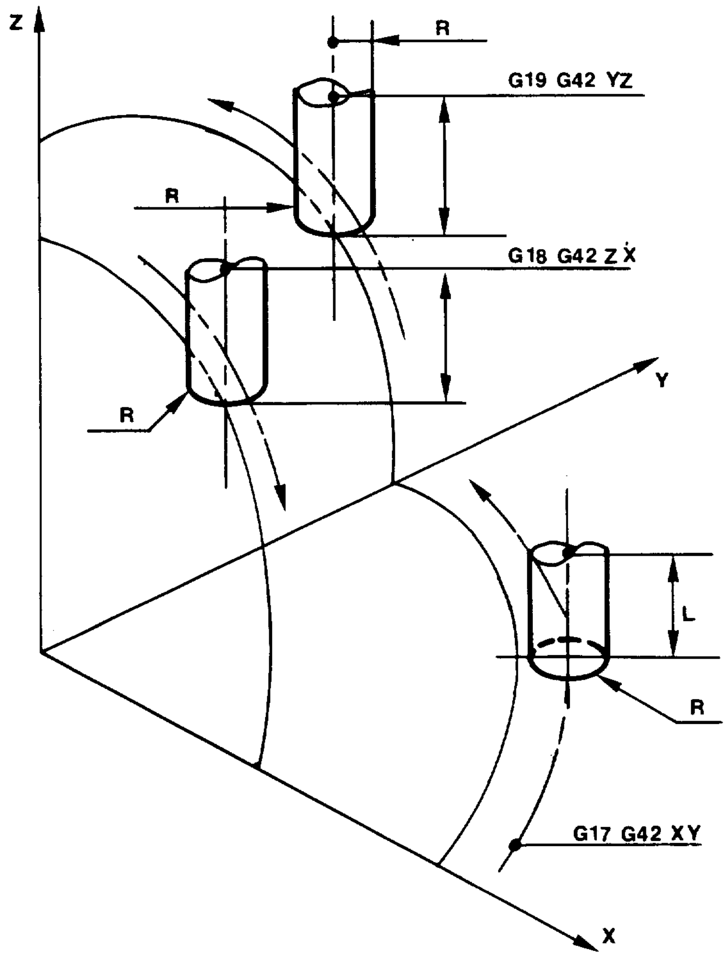



```
N10   G0    X     Y
N20   G1    G42   X1    Y1
N30   X2
N40   Y2
N50   G41   X1
N60   Y3
N70   G42   X2
.
.
.
```

The tool axis can be parallel or normal to the interpolation plane

Radius compensation is independent of the tool axis, and as such, is applied only with respect to the interpolation plane.

However, if the interpolation plane is defined to be parallel to the tool axis, then, in G41 or G42 (machining with ball-ended cutter), the length compensation, as applied in the tool axis direction, is equal to the displayed length minus the sphere radius (in G19 and G18).



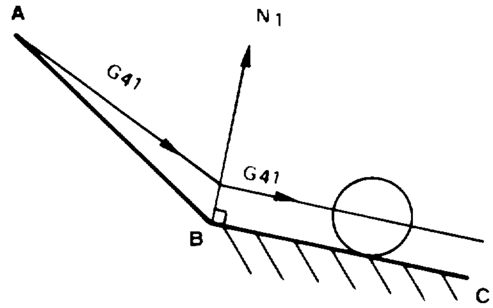
NOTE :

Radius compensation can be used with helical interpolation. It will be applied in the circular interpolation plane.

7.2 - ENTRY

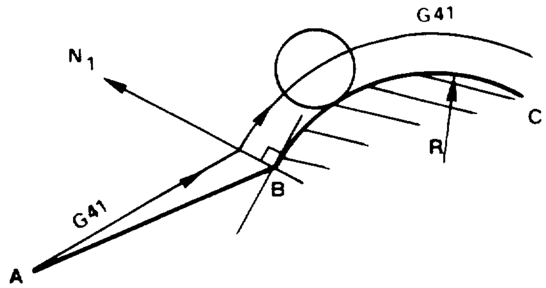
On entry into radius compensation mode, (G41 or G42), the programmed end point is modified so as to present the tool normal to the start point of the following move. The entry move MUST be linear.

Entry on a straight line



N10	G1	Xa	Ya	F
N20	G41	Xb	Yb	D10
N30		Xc	Yc	

Entry on a circle

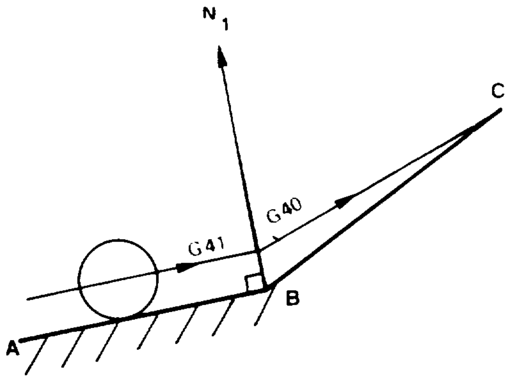


N10	G1	Xa	Ya	F
N20	G41	Xb	Yb	D20
N30	G2	Xc	Yc	R

7.3 - EXIT

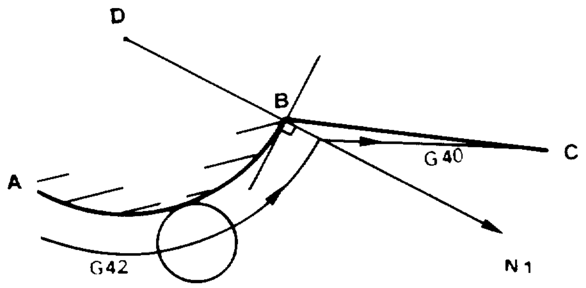
On exit from compensation mode, the last compensated move will finish at the normal to the programmed end point. Then the radius compensation will be removed during the last move, up to the G40 code. Once again, like the entry move, the exit move MUST be linear.

Exit from a straight line



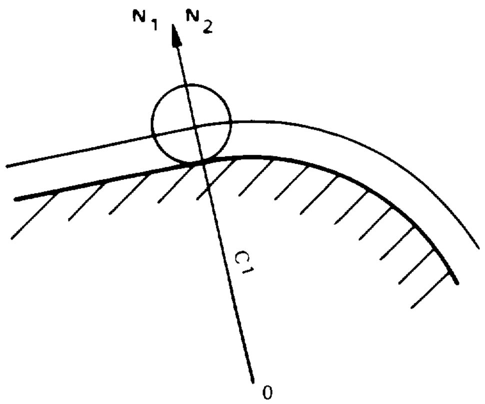
N10		X	Y	F
N20				D15
.				
.				
.				
N100	G41	Xa	Ya	
N110		Xb	Yb	
N120	G40	Xc	Yc	

Exit from a circle

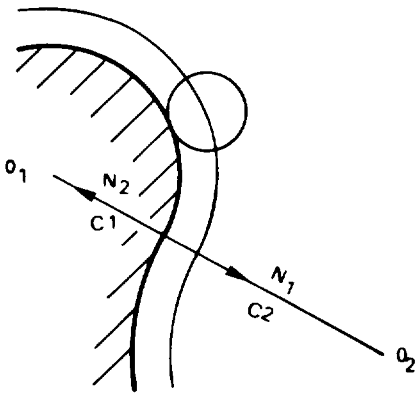


N10			X	Y	F		
N20						D16	
.							
.							
.							
N100	G42		Xa	Ya			
N110		G3	Xb	Yb	Id		Jd
N120	G40	G1	Xc	Yc			

7.4 - TANGENTIAL CONSECUTIVE PATHS



- Line–circle tangent intersection
  Compensation can be internal or external, since the elements are tangential.



- Circle–circle tangent intersection
  Compensation can be internal or external, since the elements are tangential.

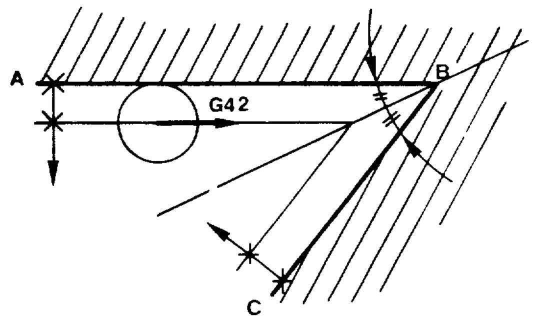
NOTE :

In both cases, the normals are merged and the compensation is applied along the normal common to both paths.

If two elements are practically tangential, i.e. the normals form an angle of less than 1/256 radians, the compensation is applied to the normal to the first path.

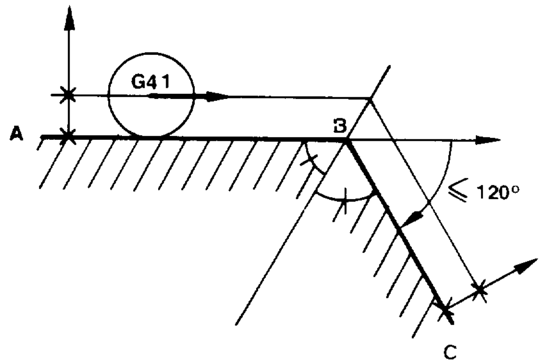
7.5 - INTERSECTION OF TWO STRAIGHT LINES

The compensation may be applied in various ways, depending on the nature of the paths, the angle they form with each other, and the nature of the profile (convex or concave) at the point at which a new block begins.



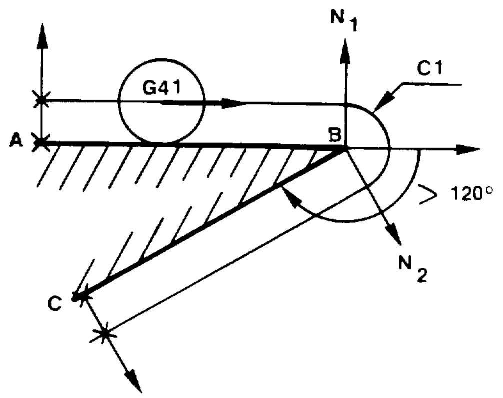
- Line-line intersection, Inside corner..

  The compensation is always applied on the bisector of the two straight lines, AB and BC



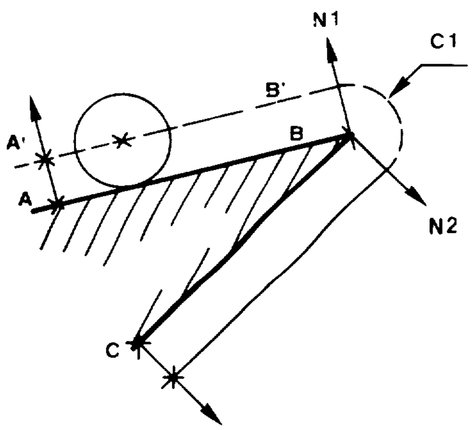
- Line-line Intersection. Outside corner ≤ 120°.

  If the angle is less than or equal to 120°, the compensation is applied on the bisector of the two straight lines, AB and BC.



- Line-line intersection, Outside corner > 120°.

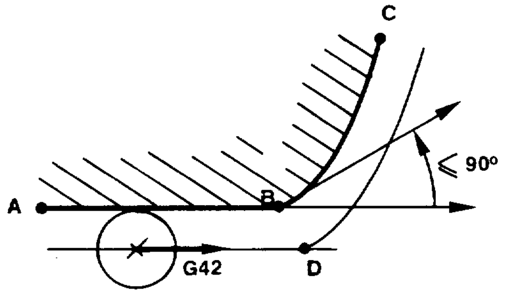
  If the angle between the straight lines is greater than 120°, the system automatically generates a connecting circular arc C1 between the two normals, N1 and N2, constructed from the intersection point B. The path offset in G41 will be, the segment of the straight line parallel to AB, the circular arc C1 and the straight line segment parallel to BC.



- The additional connecting circle

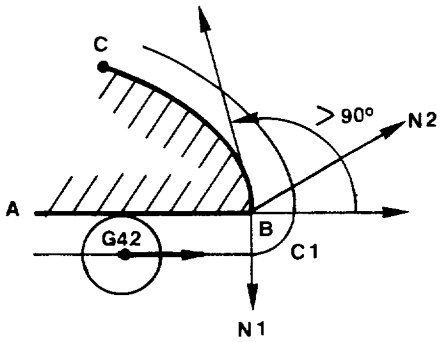
  This circle is considered to be appended to the offset path A'B', so that in block by block mode, the slides will execute path A'B' and C1, without stopping on normal N1.

7.6 - LINE-CIRCLE OR CIRCLE-LINE INTERPOLATION



- Line-circle intersection. Outside corner ≤ 90°.

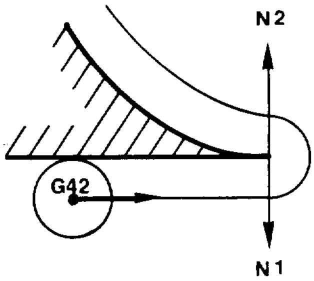
  If the angle of the tangent to the circle at the point of intersection with the straight line is less than 90°, the system applies the compensation along the offset paths to the intersection D, as shown.



- Line-circle intersection. Outside corner > 90°.

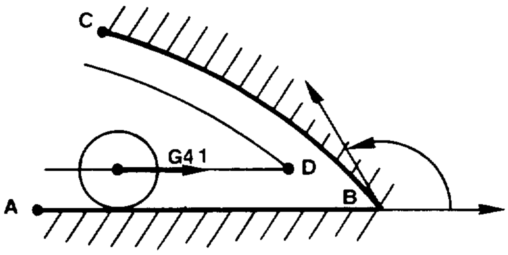
  If the angle of the tangent to the circle at the point of intersection with the straight line is greater than 90°, the system automatically generates a circular arc, C1, between the two normals N1 and N2, constructed from the point of intersection B.

  The compensation is applied in the same way, right upto 180°, at which point the circle tangent becomes coincident with the line.



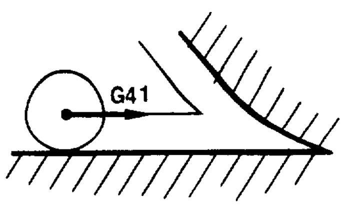
- Line-circle intersection, Inside corner.

  Whatever the value of the angle between the tangent to the circle at the intersection point and the straight line, the compensation is applied on the bisector D along the offset paths. Even if the angle is zero, the compensation is applied in the same way.

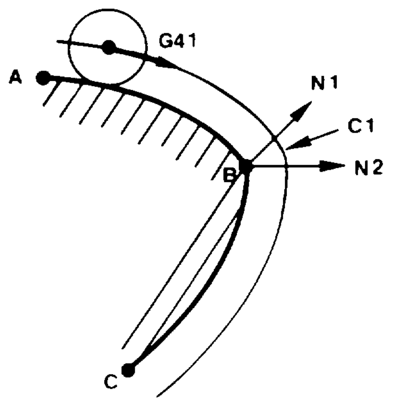


- Circle-Line intersection.

  The compensation is applied under the same conditions as above

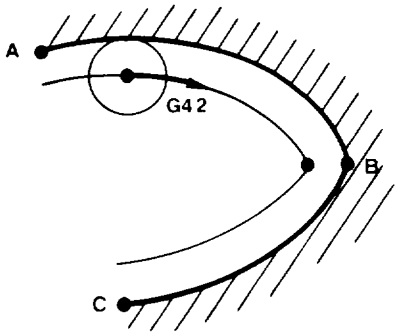


7.7 - INTERSECTION OF TWO CIRCLES



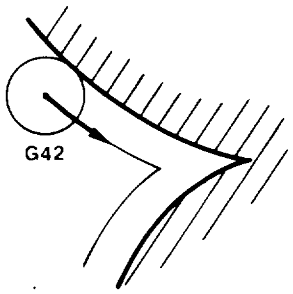
- Circle–circle intersection, Outside corner.

  The system automatically generates a circular arc C1, between the two normals N1 and N2, constructed from the point of intersection B.
  If the 2 circles are tangential, the path continues directly from the normal at the end of the first block.



- Circle–Circle intersection, Inside corner.

  Whatever the angle between the two tangents to the circles at point B, the compensation is applied on the bisector D, along the offset paths.

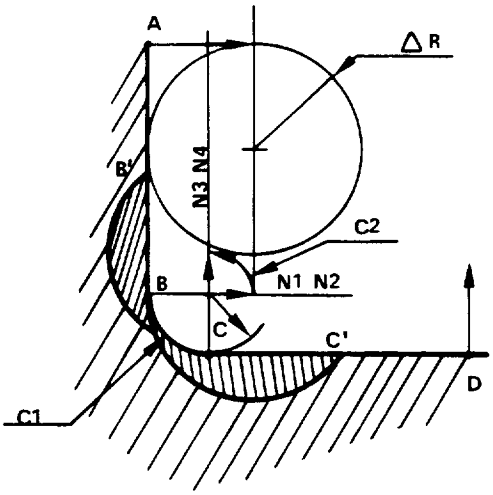


7.8 - LIMITS OF USE OF PLANE CORRECTION

Plane correction requires entering a tool radius less than or equal to the smallest programmed curve radius. The two examples below illustrate two cases of programming ignoring the relationship between part profile and real tool diameter. The system detects this impossibility and displays error 149 if the error is not inhibited by setting bit 4 in word 1 of machine parameter P7.

. Δ R greater than the programmed circle

Use of a tool nose radius value greater than the radius of the smallest programmed circle. The programmed path is A,B,C,D.



The system first calculates a path parallel to AB, and at point B, constructs the normal (N1–N2) to create a tangential entry condition as required by the programmed circle, BC.

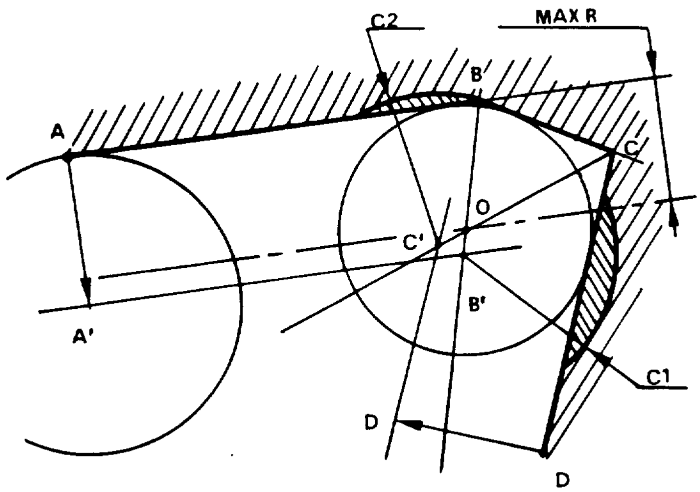
Similarly, the normal N3–N4 is constructed to the following linear block, CD, to give a tangential exit condition.

As true mathematical conditions for tangency have been created, so the system will generate the connecting circle, C2, in the counter clockwise direction, as programmed.

The figure clearly shows that this was unintentional, and how care should be taken to avoid having Δ R > radius C1.

. Δ R too big for small linear moves

Use of a Δ R value which is too large in terms of the small moves to be carried out. Again, the programmed path is A,B,C,D.



In considering the programmed path ABC, the system will calculate the required move as A'B'. The centre of the tool will be positioned to B', which is located on bisector $\widehat{ABC}$.

A similar consideration of the path BCD, will produce the move B'C'.The tool will be positioned to C', located on bisector $\widehat{BCD}$.

The successive positions will thus be A'B'C'D'.

The figure clearly shows that when the tool is first moved to B', it will already have unintentionally undercut the job. The following position, C', compounds the error, before exiting along the correct profile.

The shaded areas at C1 and C2 are wrong. In fact, the maximum Δ R tool nose radius compensation permitted can be calculated from the value of the projection of bisector BO on AA'. (O is the point of convergence of bisectors BB' and CC').

NOTES

8 - FIXED CYCLES

8.1 - CYCLES G81 TO G89

8.1.1 - General

The G80 series of machining cycles, (from G81 to G89), are modal and are cancelled by G80.

Modal functions which are present before the cycle is declared are valid in the cycle, just as most new modal functions defined within the cycle are retained after the cycle.

The machining cycle blocks are used to carry out :

- the machining cycles itself,
- plus the linear or circular positioning moves between operations.

A block may contain :

- positioning moves in the non-machining plane,
- a positioning move in the machining axis,
- the machining sequence of moves.

If all three are programmed in the same block, then a rapid move will occur, (in linear or circular interpolation mode), in the non-machining plane, followed by a rapid linear move along the machining axis to the withdrawal plane, followed by the particular machining cycle at the programmed in feed rate.

If the machining dimension and the feed rate have already been defined, it is not necessary to repeat them, because they are modal.

A block which contains only a positioning move in the machining axis, will NOT be followed by the machining cycle when the withdrawal plane is reached.

NOTE :

- *Circular interpolation during a cycle is possible only in the non-machining plane.*

Plane defined	Tool orientation	Machining axis	Possible circular interpolation plane
G19	G16 P +	X	YZ
G18	G16 Q +	Y	ZX
G17	G16 R +	Z	XY

8.1.2 – Table of fixed cycles (G81 to G89)

FIXED CYCLES ALONG THE TOOL AXIS

MOVEMENTS		G81	G82	G83	G84	G85	G86	G87	G88	G89
		Drilling Centering	Drilling Counter-boring	Woodpeck	Tapping	Reaming	Boring with orientated spindle stop	Chip-breaking	Boring and hand-facing	Fine boring up to a shoulder
Descent		Feedrate	Feedrate	Rapid, then feed with n successive pecks (PQ)	Feedrate	Feedrate	Feedrate	Feedrate with n successive pecks (PQ)	Feedrate	Feedrate
Lift after each peck				Rapid						
Dwell at each peck								Programmed by EF		
BOTTOM OF HOLE	Dwell		Programmed by EF		Fixed by m/c parameter or prog'd by EF				No limit (ARUS)	Programmed by EF
	Spindle				Reversal		Orientated stop			
	Relief						Fixed by m/c parameter			
Withdrawl		Rapid	Rapid	Rapid	Feedrate	Feedrate	Rapid	Rapid	Rapid after operator re-start	Feedrate
After withdrawl					Spindle reversal		Rapid return X and/or Y. Spindle re-start.			

IMPORTANT : If the system is equipped with the rigid tapping option, do not use programming of a normal tapping cycle.

8.1.3 – Address definitions

X,Y,Z,(B), : Axis demands on the main and secondary axes. (Modal)

- Dimension of the bottom of the hole, or
- Positioning dimensions, between holes.

ER : Withdrawal plane. (Modal)
This can be defined in any cycle block. (It refers to the machining axis.)
The machining axis must have been programmed at least once before ER can be used, however, this can be in the same block.

Whilst machining with fixed cycles, the programming of ER alone will simply provoke a re–positioning move of the machining axis. This is used to step over obstructions without repeating the fixed cycle.

I,J,K : Dimensions of the centre of the circle in circular positioning mode. (Non modal).

R : Radius of the circle in circular positioning mode. (Non modal).

F : Feed rate expressed in mm/min. (Modal).

EF : Dwell. (Modal).
EF is expressed in seconds, maximum 99.99 secs.
Only used in G82, G84, G87 or G89. (Else "error 87").

P : Depth of the first cut. (Modal).
Used in G83 and G87.

Q : Depth of the last cut. (Modal).
Used in G83 and G87.

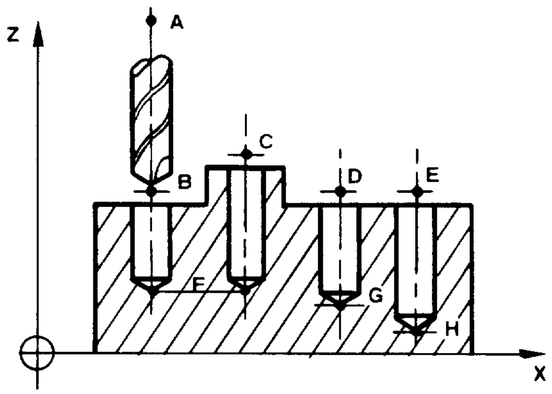
NOTE :

- *Woodpeck (G83) and chip–break (G87) cycles are performed on cut depths which decrease from P to Q.*
  If P is not programmed, (even if Q is), the depth of the first penetration will be to the full hole depth.
- *The G86 cycle MUST include the positioning dimensions in the call block, even if they are repeated, as they will be saved for future use during the cycle.*
- *The G86 cycle also makes use of the spindle orientation function, (M19), which is processed by the PLC. Thus orientation of the spindle is normally to a fixed position, as determined by the builder.*

8.1.4 - General example of programming

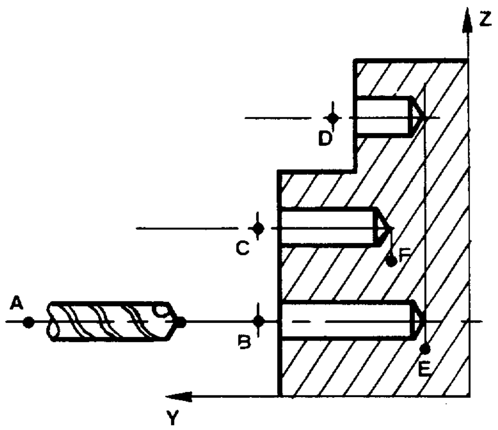
N10 G16 R+ M3 GO X Y Z	Machining axis Z (G16 R), tool direction R+.
N20 G82 X10 Y10 Z−10 F100 EF1	First machining cycle: Rapid to X10, Y10. Drill to Z−10 (from Z0) at F100. 1 sec. dwell at the bottom of the hole. Rapid withdrawal to Z0.
N30 X20 Y20	Rapid to X20, Y20. Drill to Z−10 at F100 with 1 sec. dwell at the bottom of the hole. Rapid withdrawal to Z0.
N40 ER 50	Rapid to Z+50. (NO drilling cycle).
N50 X30 Y30 ER 30 Z10	Rapid to X30, Y30. Rapid to Z30. Drill to Z10 at F100 with 1 sec. dwell at the bottom of the hole. Rapid withdrawal to Z30.
N60 G2 X60 Y60 I60 J30 ER 20	Rapid circular positioning to X60, Y60. Rapid to Z20. Drill to Z10 at F100 with 1 sec. dwell at the bottom of the hole. Rapid withdrawal to Z20.
N70 X90 Y30 R30	Rapid circular positioning to X90, Y30. Drill to Z10 at F100 with 1 sec. dwell. Rapid withdrawal to Z20.
N80 G0 X100 Y100 ER0 Z−10 F200	Rapid to X100, Y100. Rapid to Z0. Drill to Z−10 at F200 with 1 s dwell. Rapid withdrawal to Z0.
N90 ER 100	Rapid withdrawal to Z100 (NO drilling cycle).
N100 G16 Q+ X30 Z20 ER20 Y−10 F50	Change the machining axis to Y (G16 Q+). Rapid to X30, Z20. Rapid to Y20 from Y100, (back in block 80). Drill to Y−10 at F50 with 1 sec. dwell. Rapid withdrawal to Y20.
N110 G80 X150 Y120 Z150	Cancel the machining cycle. Rapid withdrawal to X150, Y120, Z150.

Example of drilling in Z direction



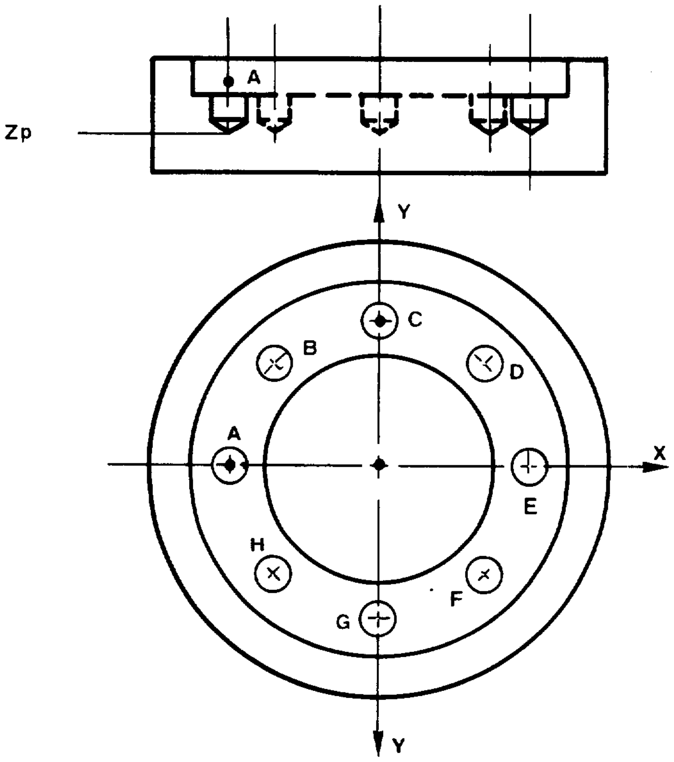
N10 G0 Xa Ya Za
N20 T1 D1 M6
N30 S800 M40 M3
N40 G16 R+ Xb Yb Zb
N50 G81 Zf F200
N60 ERc
N70 Xc Yc
N80 Xd Yd ERd Zg
N90 Xe Ye Zh M5 M9
N100 G80 Xa Ya Za D0

Example of drilling in Y direction



N10 G Xa Ya Za
N20 T1 D1 M6
N30 S800 M40 M3
N40 G16 Q+ Xb Yb Zb
N50 G82 Ye F200 EF1
N60 Xc Zc Yf
N70 Xd Zd ERd Ye
N80 G80 Xa Ya Za D0

Example of drilling in a circular groove



N10 G X Y Z
N20 T1 D1 M6
N30 S800 M40 M3
N40 G16 R+ G17
N50 G81 Xa Ya ERa Zp F500
N60 G2 Xb Yb I J
N70 Xc Yc I J
.
.
.
N120 Xh Yh I J
N130 G80 G0 Z D0

8.2 - RECTANGULAR AND OBLONG POCKET MACHINING CYCLE : FUNCTION G45

The pocket machining cycle is used to machine pockets with the following simple shapes :

- circular,
- oblong,
- rectangular,
- rectangular with corner radii.

The user can choose to program :

- the roughing cuts only,
- the roughing and finishing cuts on the bottom and/or sides of the pocket, (but only if the same tool is used for both roughing and finishing),
- finishing cuts only (on the bottom or sides of the pocket).

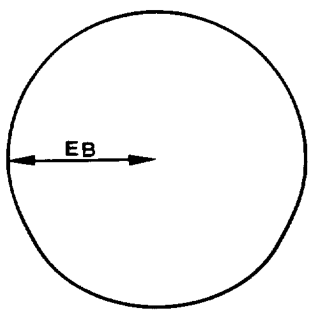
The programmer chooses between these various options by including or omiting certain parameters.

8.2.1 - Programming

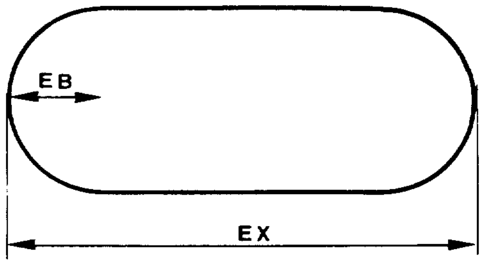
- G45 is a non-modal function.
- X, Y, Z, : co-ordinates that define the centre of the pocket, at final depth.
  They are programmed in absolute mode (G90).
- EB, EX, EY, EZ : incremental dimensions defining the outlines of the finished pocket.

  EB represents a radius
  EX width of pocket, in the X axis
  EY breadth of pocket, in the Y axis
  EZ depth of pocket, in the Z axis

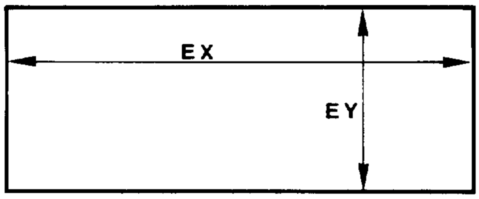
* EB programmed on its own will thus correspond to a circular pocket.



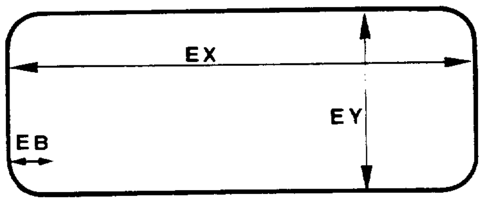
* one width and one radius corresponds to an oblong pocket.



* both width and breadth, corresponds to a rectangular OR square pocket.



* width, breadth and radius, corresponds to a rectangular or square pocket with corner radii.



Other configurations generate an error.

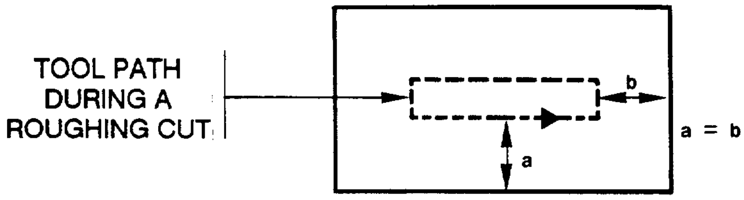
– P, Q, I, J : define the cut setting for roughing, and the stock allowances for finishing.

  P axial (depth of cut) setting
  Q lateral (width of cut) setting for roughing out

  I finishing stock allowance, on depth
  J finishing stock allowance, on sides

  These values are ALWAYS positive.

Oblong, square or rectangular pockets are roughed out in such a way that the material still to be machined on the two axes are equal.



- The programmed width of cut, Q, is automatically optimized as follows :

  (let R = the tool radius, and L = the width or breadth, whichever is the smaller)

  a) if Q > 2R, then error 64 will be reported,

  b) if Q ≤ R, then, if the result obtained by dividing half L by the width of cut is not an integer, the system calculates a new cut setting Q' thus.

  The number of passes, n = (L/2 – R)/Q (rounded up to the next whole number)

  The calculation of the new cut value, Q' = (L/2 – R)/n

  The value of Q' is rounded up to the nearest micron if the result of the division is not an exact value in microns. By doing this, the value of the last cut will then be :
  (L/2) – R – Q'(n–1)

  c) If R < Q < 2R, then, (except for circular pockets), the value of the first cut will be R/2 and the value of the other cuts will be optimized thus :

  Q' = (L/2 – R)/n

  For the depth of cut setting, (P), if the division of the total depth by the depth of cut does not produce an integer, then the system calculates a new cut, (P'), using the same formula as for the calculation of Q'

- EP, EQ, EI, EJ : machining feeds

  EP axial roughing feed
  EQ lateral roughing feed
  EI axial finishing feed
  EJ lateral finishing feed

  These values are always positive.

Four different feedrates can be programmed, but they are not compulsory.

EXAMPLE :

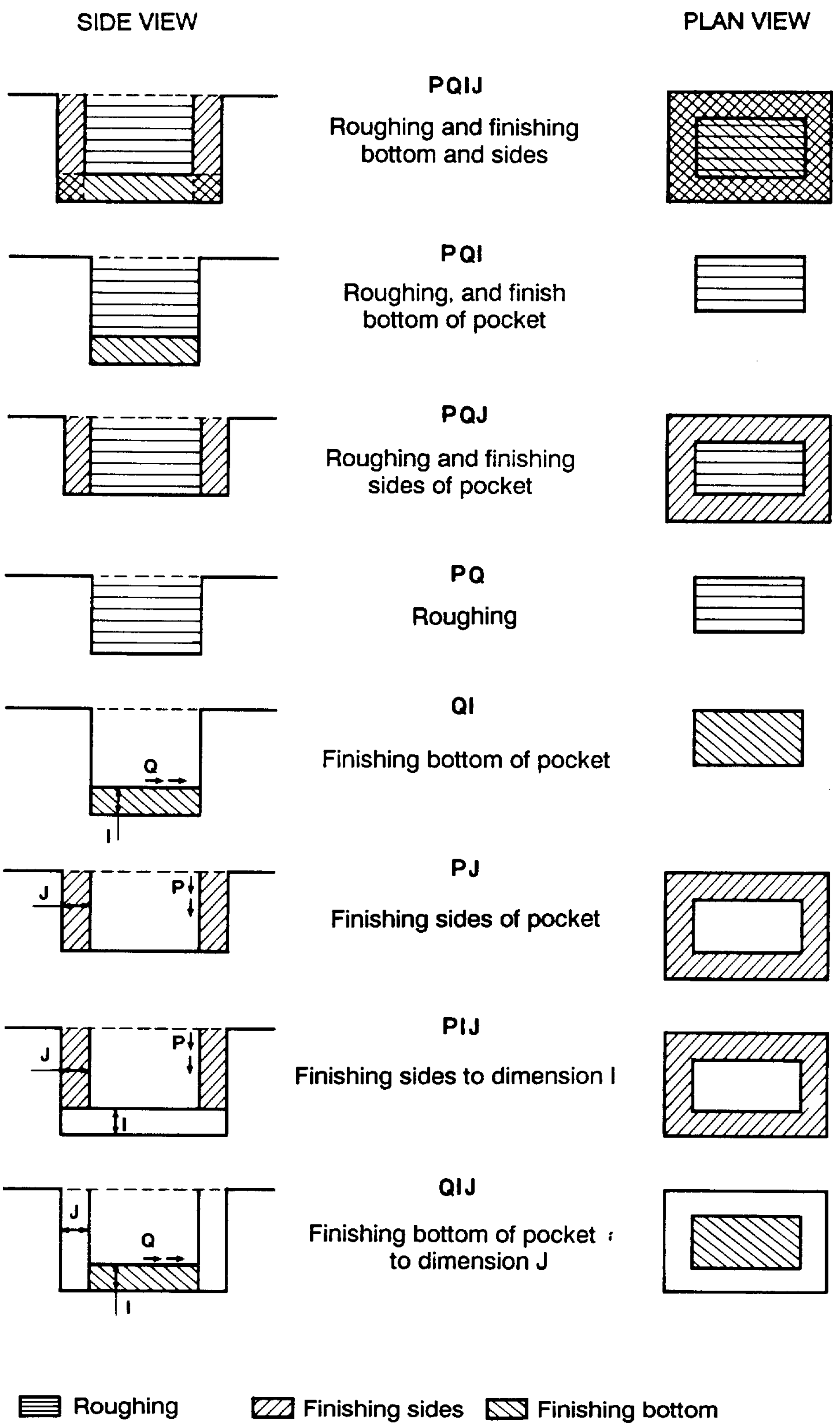
ie. If the same infeed will suffice for both roughing and finishing, then either EP or EI may be specified. Similarly, if only one lateral feedrate is given (EQ or EJ), then it will be applied to both roughing and finishing stages. However, at least one axial and one lateral feedrate MUST be given, or an error will be reported.

- ER : the optional withdrawal plane ; can be programmed in absolute or in incremental mode

NOTES :

- *Parameters X, Y, Z, I, J, P, Q, EB, EX, EY and EZ are not modal.*
- *Parameters ER, EI, EJ, EP and EQ are retained and can be used again in a later pocket machining cycle without being reprogrammed. (Note : EB and ER in the pocket machining cycles are stored separately from the EB in Geometric Profile Programming and the ER in the drilling cycles).*
- *Cycle blocks cannot be interrupted by a change of mode to single block.*
  Thus, modifications can only be made once the cycle has completed.

PROGRAMMED CUT SETTINGS



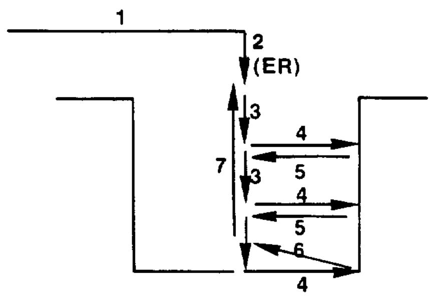
8.2.2 - Description of machining

8.2.2.1 - Roughing only

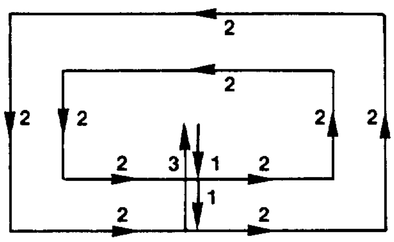
The tool is positioned to the centre of the pocket by a rapid move in the lateral plane. It will then move, still at rapid, to the withdrawal plane, if the optional withdrawal plane has been programmed (ER).

It then executes a move towards the bottom of the pocket, at cutting feedrate. This movement is equivalent to the axial, depth of cut, setting. The tool then moves towards the smaller side, a distance equal to the value of the lateral, width of cut setting, at the specified lateral feedrate. The tool then follows the path corresponding to the contour of the pocket.

Once the tool has returned to its starting point, it executes a new lateral cut and again follows the contour path, progressively enlarging the pocket. On finally finishing the machining in this plane, the tool is repositioned, at rapid, back to the centre of the pocket where it executes a new axial move, then repeats the preceding cycle. Re-centering is made with a "horizontal" move for all cuts except the last. The path of the last re-centering includes a tool withdrawal of 1 mm at the centre of the pocket to avoid scuffing.



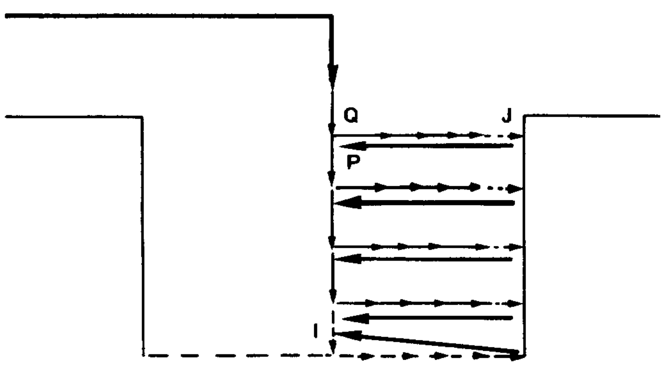
1 : rapid tool to pocket centre
2 : ER ---> change of starting point
3 : depth of cut, P, into the material
4 : machining in the plane
5 : re-centering
6 : last re-centering
7 : rapid withdrawal



1 : width of cut, (Q)
2 : following the pocket contour
3 : re-centering

8.2.2.2 - Roughing and finishing

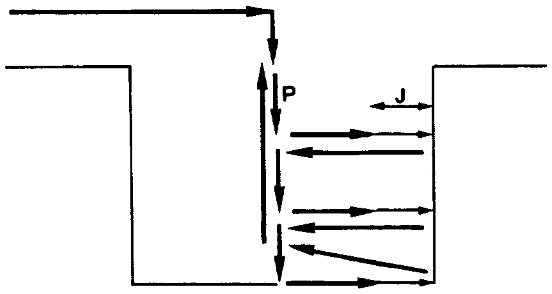
The two operations are carried out using the same tool.



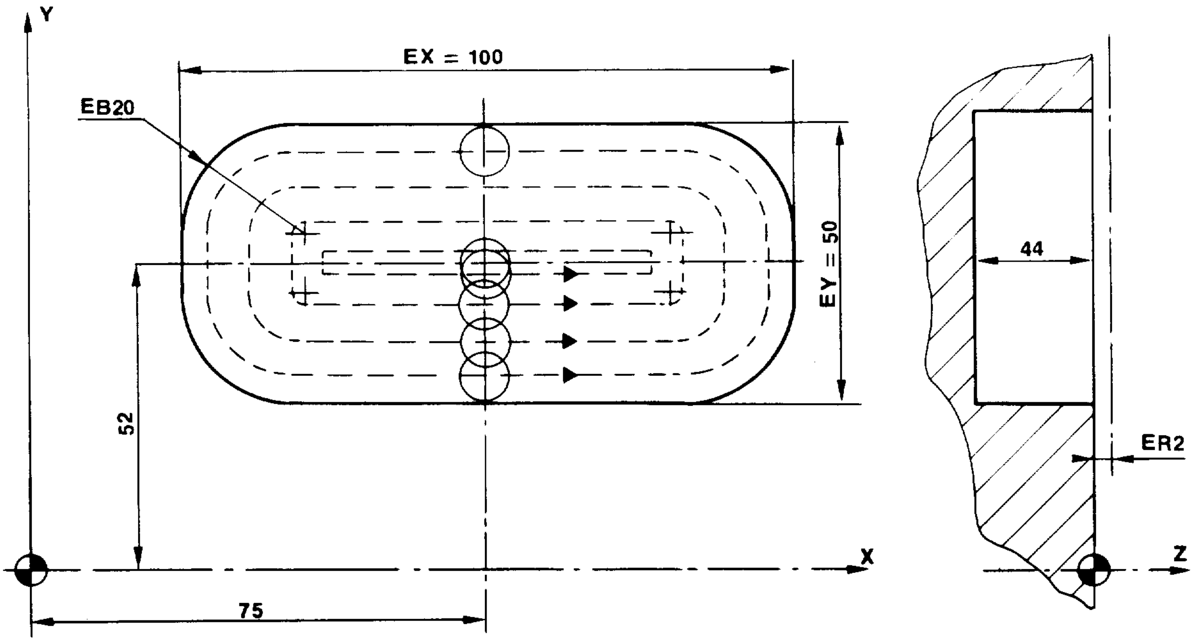
— ROUGHING
--- FINISHING
— RAPID MOVES

8.2.2.3 – Finishing only (Example : PJ)

Allows for a separate finishing tool.



8.2.3 – Programming example



Tool radius D1 = 4mm

1 – Without finishing cut

N50 G X–100 Y–20 Z50 D1 M3 S3000
N60 G45 X75 Y52 Z–44 ER2 EX100 EY50 EB20 P10 Q7 EP100 EQ500
N70 X–100 Y–20 Z50 M2

2 – Or with finishing cut of 0.5 at 300 mm/min at N60 becomes :

N60 G45 X75 Y52 Z–44 ER2 EX100 EY50 EB20 P10 Q7 I.5 J.5 EP100 EQ500 EI70 EJ300

8.2.3.1 – Computing the lateral cut (Example 1)

In this example, R < Q < 2R.

So, the first cut value will be R/2

The following cuts (except the last)

Number of cuts : n = (L/2 – R)/Q = (50/2 – 4)/7 = 3

Since there is no remainder, n = 3. Thus, the programmed width of cut Q
is the same as Q'. ie (L/2 – R)/n = (50/2 – 4)/3 = 7

The last cut = (L/2 – R) – Q' (n – 1) = 7 in the example.

8.2.3.2 – Computing the axial cut (Example 1)

Processing is identical to the calculation of Q', except that the value of the first cut is P'.

Number of cuts : n = Z/P = 46/10 (then rounded up, from 4.6 to 5), ie. 5 cuts

Then the optimized depth of cut : P' = 46/5 = 9.2

8.2.3.3 – Breakdown of cuts (Example 1)

Machining is described relative to the tool centre.

- Rapid position to X75, Y52.
- Rapid position in Z to ER = 2.
- Sink at feedrate by Z – 9.2.

*

- Mill to Y50, ie pocket centre, (52) – 1/2 radius, (2). Given that R < Q < 2R, the first cut is equal to R/2.

- Contour the pocket keping a constant distance from the edges, (L/2 – R/2) = 23
- Mill in Y to 52 – 7 (pocket centre – Q') : ΔY = 7.
- Contour at L/2 – Q' = 18 from the edges of the pocket corner radius = EB – 18 = 2.
- Mill in Y = 52 – 14 (pocket centre – 2 Q') : ΔY = 7.
- Contour at L/2 – 2 Q' = 11 from the edges of the pocket corner radius = EB – 11 = 9.
- Mill in Y = 52 – 21 (pocket centre – 3 Q') : ΔY = 7.
- Contour at L/2 – 3 Q' = 4 from the edges of the pocket corner radius = EB – 4 = 16.
- Rapid return to the centre of the pocket in the X Y direction.
- Descend at feedrate another Z – 9.2, then machine laterally as before.

  The preceding steps are carried out four times, starting from *.
- Rapid return, (with lift), to the centre of the pocket in X, Y, Z (+1 mm).
- Rapid return in the Z direction to Z2.

8.2.3.4 – Computing the cuts (Example 2)

- Lateral width of cut

  n = (L/2 – J – R)/Q, or 20.5/7 ie. 2.928 rounded up to 3

  Q' = (L/2 – J – R)/n, or 20.5/3 ie. 6.8333 rounded up to 6.834
  Last cut : (L/2 – J – R) – (n – 1) Q' = 6.832.

- Axial depth cut :

  n = (Z – I)/P, or (46 – 0.5)/10 ie. 4.55 rounded up to 5

  P' = (Z – I)/n, or 45.5/5 ie. 9.1

8.3 – POCKET MACHINING AND FACING WITH ANY CONTOUR : FUNCTION G46

8.3.1 – General

Pocket machining and facing cycles with any contour are used to machine one or more pockets of varied shapes with or without islands.

This type of machining is programmed with one definition command and three machining commands.

8.3.1.1 – Geometric definition command

This command is used to define :

- The geometry of the contours (pockets, islands, facings, recesses).
- The scan geometry (width and depth of cut, machining allowance, tool diameter).

It occupies several blocks and begins with a header block specifying the geometric characteristics of the scan. It ends with an end block.

Between the header block and the end block, the contours are defined in ISO or PGP. These contours are separated by segmenting blocks.

NOTE :

It is recommended to place the contour pocket or surface definition command before any machining command whatsoever, even relative to this pocket or surface.

8.3.1.2 – Machining commands

Three machining commands are available and are each part of the same block :

- Initial drilling command.
- Pocket roughing command.
- Pocket finishing and/or semi–finishing command.

The operations are divided up into several phases to allow tool changes between phases.

8.3.1.3 – Syntax

The blocks specific to the pocket machining or facing cycle are all identified by function G46, supplemented by the NU address specifying the type of block.

It is mandatory to program G46 and NU on each of the blocks. Function G46 must be programmed at the beginning of the block.

The values of NU are :

- G46 NU0 Definition header block
- G46 NU1 Definition segmenting block introducing a pocket
- G46 NU2 Definition segmenting block introducing an island
- G46 NU3 Definition segmenting block introducing a facing operation
- G46 NU4 Definition segmenting block introducing a recess
- G46 NU5 Definition segmenting block introducing a facing (contour with wall)
- G46 NU6 Definition segmenting block introducing a wall on a facing
- G46 NU9 Definition end block
- G46 NU10 Initial drilling command
- G46 NU15 Pocket roughing command
- G46 NU20 Pocket finishing (or semi-finishing) command

8.3.1.4 – Programming constraints

The roughing command leaves a variable amount of additional machining allowance added to the machining allowance programmed in the definition command. This additional machining allowance varies from one point to another of the roughed profile but remains below the value 3/2 Q, where Q is the width of cut. This means that a semi-finishing cycle is necessary to allow the real finishing cycle to be carried out on a constant machining allowance.

The semi-finishing cycle and the finishing cycle are carried out by the same machining instruction (G46 NU20) but a machining allowance must be specified in the instruction for semi-finishing. This allowance must be the same as that programmed with the definition header instruction (G46 NU0).

Constraints on the finishing and semi-finishing tools

The CNC performs semi-finishing cycles like standard contouring operations : G41, G42. The programmer must therefore make sure that the real finishing tool radius is sufficiently small to reach all the geometric parts of each of the contours programmed, pockets and islands, without colliding with the other geometric parts of the contour considered or with those of the other contours.

The same type of constraint applies to the semi-finishing tool used, but in this case the reference tool is a fictional tool whose radius is that of the real tool plus the machining allowance. The constraint is therefore more severe.

It should be noted that to remove all the material remaining after the roughing phase, the semi-finishing tool diameter (DS) must satisfy the following equation :

DS >= 3/2 Q i.e. Q <= 0.66 DS

Q is the width of cut.

Roughing constraint

The roughing path is constructed in compliance with a scale relationship between the width of cut (Q) and roughing tool diameter (ED) so as not to leave any material in the roughed area. This equation is approximately the same as that between the width of cut and the semi-finishing tool. The following equation is used :

Q <= 0.66 ED

This equation is tested when analysing the definition instruction.
An error is indicated if it is not verified.

It is recommended to use a roughing tool with the same size as the semi-finishing tool whose dimensions are subjected to collision avoidance constraints with the programmed profile parts as mentioned above.

Constraint on the pocket dimensions

The maximum pocket dimension on either of the axes of the plane must be less than 40 times the width of cut Q for reasons of memory space occupied.

This equation is tested when analysing the definition instruction. An error is indicated if the equation is not satisfied.

Narrowed areas

The narrowed areas are generally located between an island and the edge of the pocket but can also exist on a profile or between two islands.

There can be narrowed areas which, although greater in diameter than the roughing cutter, cannot be roughed. This case occurs when the roughing tool cannot pass because of the type of path used during this phase.

This means that after the semi-finishing phase, a region of unremoved material can remain in the vicinity of the median line of the narrowed area. The thickness of this residual material is always less than the diameter of the roughing tool and can be removed "manually" by a simple "milling cutter" pass.

The 3D graphic display can help detect areas of residual materials.

Summary of constraints

- Constraints on the finishing and semi-finishing tools : They are due to difficulties in passing along the contours. It is advantageous to choose the largest possible semi-finishing tool because of the constraints mentioned above. The manufacturing allowance must be added to the tool radius to evaluate any passing problems.

- Choice of the roughing tool : It is recommended to use the same size as the semi-finishing tool.

- Constraint on the width of cut : Must be less than or equal to 0.66 times the roughing cutter diameter.

- Constraint on the maximum pocket diameter : It must be less than or equal to 40 times the width of cut.

8.3.2 – Geometric definition instruction

8.3.2.1 – G46 NU0 (Definition header)

G46 NU0 NP(5) ED(5.3) Q(5.3) J(5.3) NR+/– R03/04 LX (5.3) LY (5.3)
EX(5.3) EY(5.3)

NP : Pocket number
ED : Roughing cutter diameter
Q : Roughing width of cut
J : Machining allowance (default zero)

J represents the minimum machining allowance after roughing if a constant finishing allowance is desired on the entire profile and roughing is to be followed by semi–finishing with the same allowance J.

NR+/– : Type of work (NR+ : inward, NR– : outward) (default NR+)

R03/04 : Desired tool rotation direction (default R03)

LX LY : Coordinates of the drilling point and the roughing start point in the plane of the pocket (by default, this point is calculated by the software)

Depending on the interpolation plane, it is necessary to program :

In G17 : LX, LY,
In G19 : LY, LZ,
In G18 : LZ, LX.

The order in which the parameters are programmed is important : for instance in G18, LZ, LX and not LX, LZ.

EX EY : Coordinates of the roughing end point in the plane of the pocket (by default, this point is calculated by the software)

Depending on the interpolation plane, it is necessary to program :

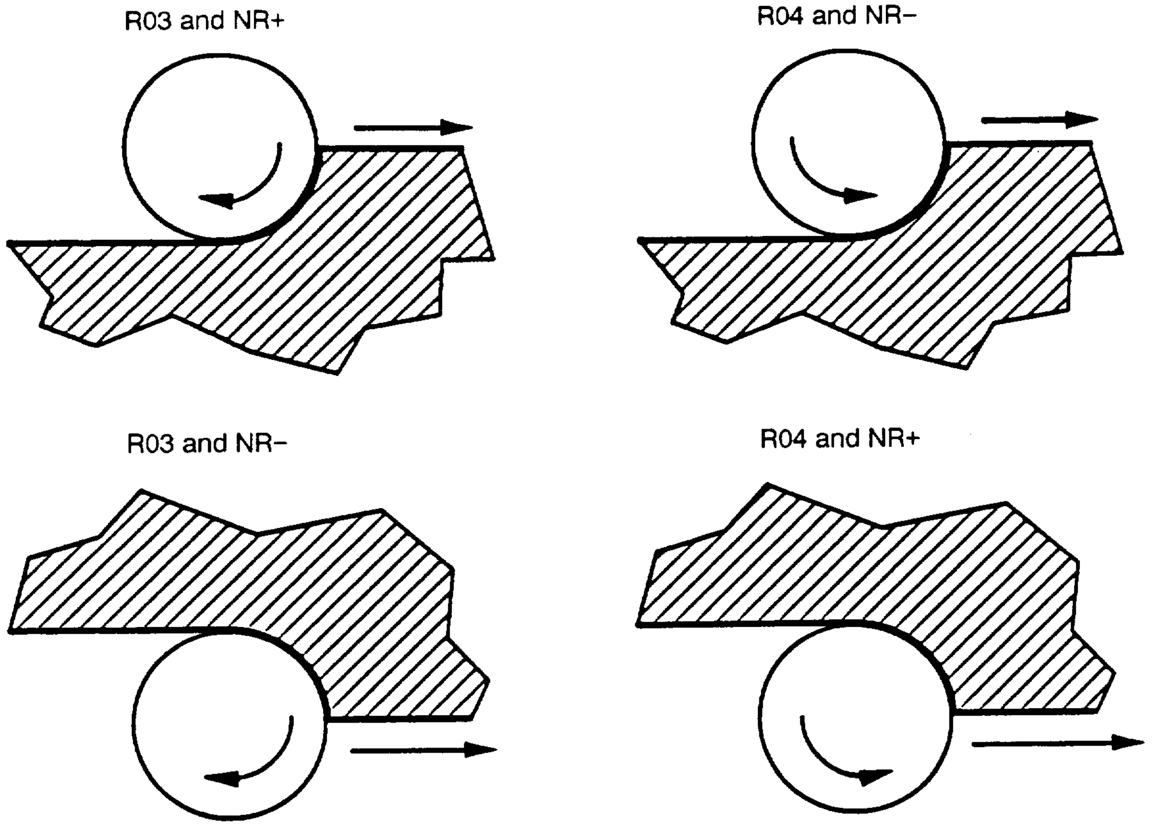
In G17 : EX, EY,
In G19 : EY, EZ,
In G18 : EZ, EX.

The order in which the parameters are programmed is important : for instance in G18, EZ, EX and not EX, EZ.

The CNC checks that Q and ED satisfy the equation :

Q <= 0.66 ED

ILLUSTRATION OF THE COMBINATIONS OF R AND NR



8.3.2.2 – G46 NU1 or NU2 (Start of pocket and island contours respectively)

The above blocks have the same syntax and differ only by the NU number.

G46 NU1 LX(5.3) LY(5.3)
G46 NU2 LX(5.3) LY(5.3)

LX LY : Coordinates of the contour finishing start point in the plane of the pocket (by default, the first point describing the contour).

Depending on the modal interpolation plane, it is necessary to program :

LX/LY, LY/LZ, LZ/LX.

Block G46 NU1 or G46 NU2 is followed by a sequence of ISO blocks with or without PGP. The first block after it contains the dimensions of the start point. If LX and LY are not programmed, this block must be separate from block G46.

If the finishing start point is located in an acute angle, the system generates an error (even if the G46 NU20 finishing instruction is not included in the program).

Block G46 NU2 must be associated with a G46 NU1 pocket milling, G46 NU3 facing or G46 NU5 and G46 NU6 open profile contour definition. Otherwise, the system generates an error (error 283).

ISO blocks offer possibilities for programming standard profiles :

- PGP programming.
- Parametric programming and possibility of sub-program calls (G77 N N).
- The PGP blocks can contain dimensions programmed on the axis perpendicular to the interpolation plane. This plane is the modal interpolation plane when programming the definition header block. These dimensions are ignored (e.g. : presence of Z dimension for an XY interpolation plane).
- The contours can be programmed in any order for roughing, but the order must correspond to the finishing order.
- The first PGP contour block can be in either G0 or G1.
- The presence of G0 in a profile description other than in the first block generates an error message.

Example :

	Programming wrong	Programming right
	G46 NU1	G46 NU1
	G0 X10 Y10	G0 X10 Y10
	X50	G1 X50
	.	.
	.	.
	.	.

- A sequence of two identical G1 blocks (same X and Y dimensions if the interpolation plane is XY) generates an error message.
- The use of external write parameters is prohibited in such blocks. The use of external read parameters is allowed but must be carried out with care.

RESTRICTIONS : In profile definitions, the following functions are ignored :

- angular shifts (ED),
- programmed shifts (G59),
- mirror functions (G51),
- scaling factor (G74),
- programming in inches (G70),
- turn table off-centering (DAT3).

The angular shift, mirror function and scaling factor can be used in machining instructions. They then affect all the profiles of the pocket to be machined.

8.3.2.3 - G46 NU3 and NU4 (Facing and recessing contour start points)

The above blocks have the same syntax and differ only by the NU number.
Block G46 NU3 or G46 NU4 is followed by a sequence of ISO blocks with or without PGP. The rules for writing these blocks are the same as described above in Sec. 8.3.2.2.

NOTE :

Contrary to blocks G46 NU1 and NU2, it is not possible to program definition of the start points since no finishing is carried out on such contours.
Block G46 NU4 must be associated with definition of a pocket contour by G46 NU1, facing G46 NU3 or open profiles G46 NU5 and G46 NU6.
Otherwise system generates an error (error 283).

8.3.2.4 – G46 NU5 and NU6 (Facing and wall contour start point)

The above blocks have the same syntax and differ only by the NU number.

Block G46 NU5 or G46 NU6 is followed by a group of ISO blocks with or without PGP. The rules for writing these blocks are the same as in Section 8.3.2.2. These two blocks define an open profile.

The sequence of blocks programmed must mandatorily form a closed profile.

The number of NU5 and NU6 blocks is unlimited for defining a closed profile.

The end point coordinates of an open profile must be the start point coordinates of the following open profile.

Blocks NU5 and NU6 describing a contour are inseparable. They must therefore not be separated by NU blocks of another type.

For programming contour looping, the end point coordinates of the last open profile must be the start point coordinates of the first open profile written.

NOTE :

For instruction G46 NU6, it is not possible to program the start point since the profile is open. Finishing therefore necessarily begins at one end of the profile (chosen according to the type of work).

Instruction G46 NU5 has the same syntax as G46 NU3.

A wall can be described by several NU6 blocks provided they are consecutive.

8.3.2.5 – G46 NU9 (End of definition)

Blocks G46 NU1 to NU9 do not contain the pocket number since they are associated with block G46 NU0.

8.3.3 – Machining instructions

8.3.3.1 – Initial drilling instruction NU10

There are three possible types of drilling :

- Simple drilling :
  G46 NU10 NP(5) G81 Z(5.3) [ER(5.3)] [F(5.2)]

- Peck drilling :
  G46 NU10 NP(5) G83 Z(5.3) P(5.3) [Q(5.3)] [ER(5.3)] [F(5.2)]

- Drilling with chip breaking :
  G46 NU10 NP(5) G87 Z(5.3) P(5.3) [Q(5.3)] [ER(5.3)] [F(5.2)] [EF(2.2)]

  . NP : Pocket milling or facing number

  . G81 : Simple drilling

  . G83 : Peck drilling

  . G87 : Drilling with chip breaking

  . Z : (Z or Y or X depending on the contouring plane) : hole bottom dimension

  . P : First penetration for G83 or G87

  . Q : Last penetration for G83 or G87

  . F : Drilling axial speed (in mm per min)

  . EF : Hole bottom dwell for G87

  . ER : Clearance plane dimension (hole top). This dimension is interpreted as being in the same axis as the hole bottom dimension.

    - Axial travel from dimension ER to hole bottom dimension Z is at the work feedrate.

    - Retraction is at high speed up to dimension ER.

    - If ER is not programmed, axial travel is at the work rate from the initial speed and dimension to the hole bottom Z dimension.
      (see Sec. 8.3.4).

The drilling point is given by the L.. dimensions programmed in NU0. Lacking them, the software performs the calculation.

RESTRICTION :

This cycle can be omitted in certain cases. Drilling is then carried out manually by the programmer in the required location. In this case, the programmer must include the hole position as roughing start point in the definition instruction.

This procedure cannot be applied if the pocket and/or island configuration is such that it requires several drilling points. In this case, the drilling cycle is required.

NOTE :

This cycle is used for facing if the roughing tool cannot access all the areas without retracting. It is therefore necessary to check on the graphic display whether or not this instruction is required.

8.3.3.2 – Roughing instruction : NU15

G46 NU15 NP(5) Z(5.3) P(5.3) [ER(5.3)] [EH(5.3)] [EP(5.2)] [EQ(5.2)]

- NP : Pocket milling or facing number
- Z : Z or X or Y depending on the contouring plane, specifies the machining bottom dimension.
- P : Axial width of cut. Roughing is carried out in (Z/P+1) consecutive axial passes. Each axial pass is in the location of the initial drilling.
- ER : Clearance plane dimension (see drilling cycle).
- EH : Material plane dimension (see Sec. 8.3.4).
- EP : Axial feed (the modal value of F is taken as default).
- EQ : Lateral machining feed (the modal value of F is taken as default).

The roughing start point is given by the L.. dimensions programmed in NU0. Lacking them, the software calculates the point.

When executing this cycle, the CNC checks that the tool radius R associated with the modal D corrector is compatible with the machining parameter ED specified during definition.

R = ED/2 (ED = cutting tool diameter).

8.3.3.3 – Semi-finishing or finishing instruction : NU20

G46 NU20 NP(5) Z(5.3) P(5.3) [ER(5.3)] [EH(5.3)] [EI(5.2)] [EJ(5.2)] [J(5.3)]

1. NP, Z, P, ER, EH : Same as for the roughing instruction
2. EI : Axial finishing feed
3. EJ : Lateral finishing feed
4. J : Finishing allowance (default zero)

The value of J must be less than or equal to the value of J programmed in NU0. Otherwise, the pass may be carried out without material removal.

The modal value of F is taken as default for EI and EJ.

During execution of the cycle, the CNC checks that the tool radius R associated with the modal corrector D is less than or equal to the radius (ED/2) of the cutting tool specified during definition.

NOTE :

This cycle is unnecessary when the contours are only type NU3 and NU4. However, if it is present, there are no problems, it is simply ignored.

Engagement, Clearance

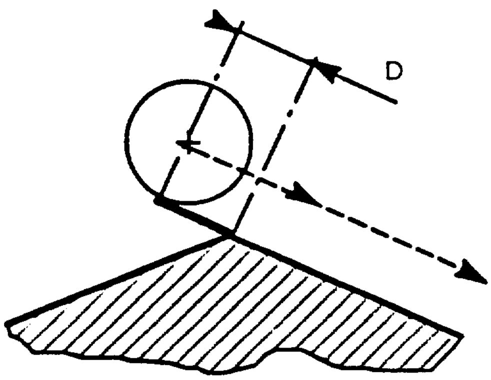
- At the apex of an obtuse angle

  Engagement and clearance are linear and are tangent to the start and end segments.

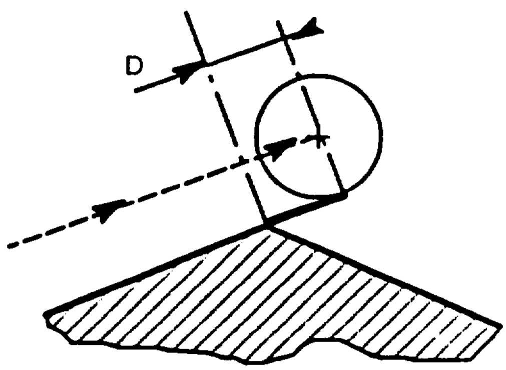
  The distance D is such that :

  D = 5/4 (Re + J)

  Re : cutting tool radius
  J : machining allowance programmed in NU0



Engagement



Clearance

- At the apex of an acute angle

  Engagement and clearance at an acute angle are impossible. In this case, the system generates an error message (error 288).

- On a flat angle

  General rule :

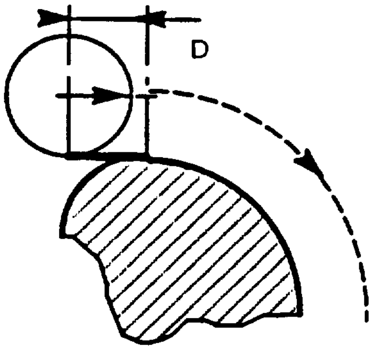
  If linear engagement is possible, it is carried out (case 1)
  Otherwise, if circular engagement with radius D is possible, it is carried out (cases 2, 3, 4)
  Otherwise, engagement is circular with radius D'

The values D and D' are such that :

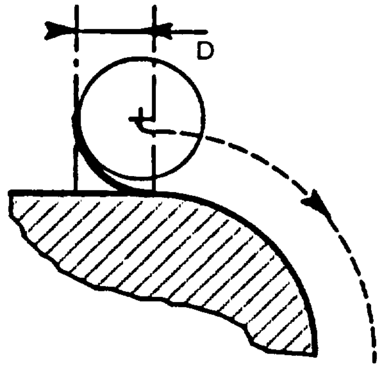
D = 5/4 (Re + J)
D' = 0.9 Rp

Re : cutting tool radius
J : machining allowance programmed in NU0
Rp : profile radius

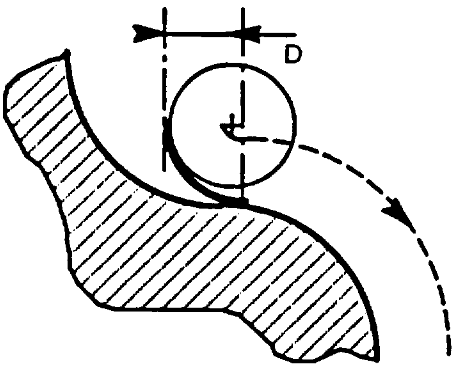
The same rule applies to clearance :



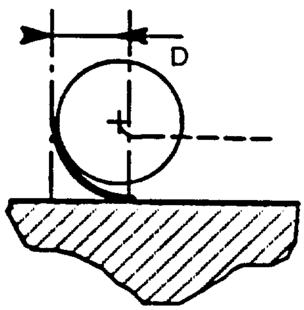
Case 1



Case 2



Case 3



Case 4

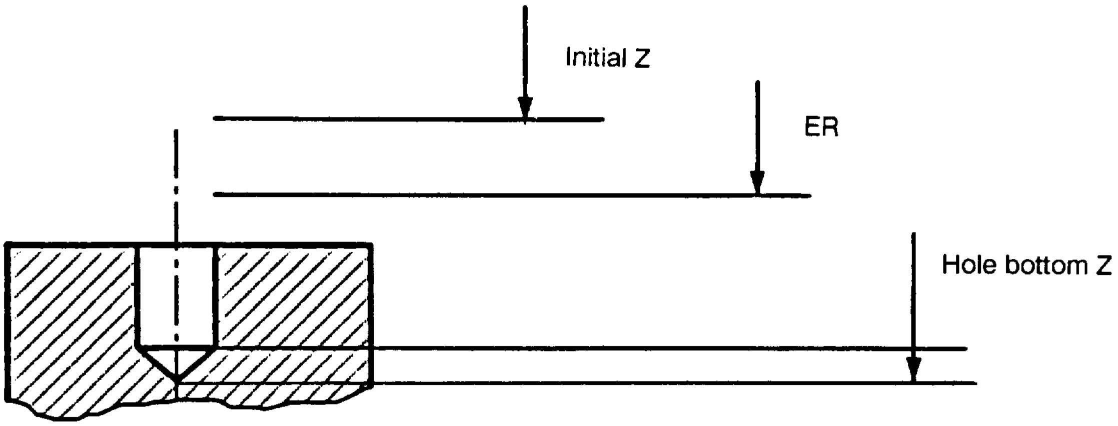
Engagement other than on the end of a segment is treated in the same way as for a flat angle.

8.3.4 – Interpretation of dimensions ER and EH

8.3.4.1 – Interpretation of guard plane ER

This dimension is interpreted by :

- The drilling cycle
- The roughing cycle
- The finishing cycle



The initial Z represents the current dimension on the axis perpendicular to the interpolation plane when the cycle is called.

- Axial travel from the initial Z to the ER dimension is at high speed. If ER is not programmed, axial travel is at the work rate from the initial Z dimension to the hole bottom Z dimension.

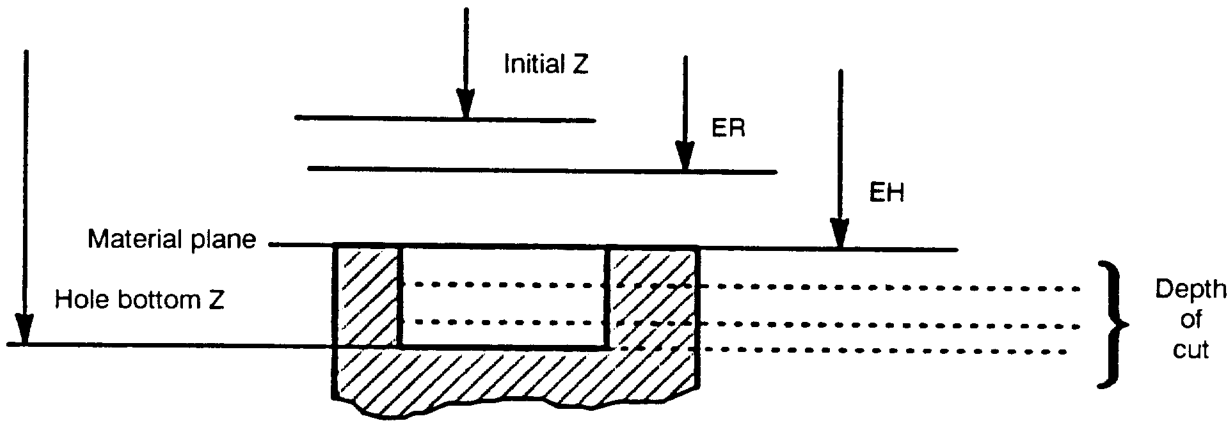
- Axial travel from the ER dimension to the hole bottom Z dimension is at the work rate.

- Retraction is at high speed up to dimension ER.

8.3.4.2 – Interpretation of the material plane dimension EH

This dimension is interpreted by :

- The roughing cycle.
- The semi–finishing and finishing cycles.



EH is used to cut several consecutive depths from the material plane dimension.

- Axial travel from dimension ER to dimension EH is at the work rate.
- Retraction is at high speed up to ER after each pass.
- If EH is not programmed, the consecutive cuts are made from dimension ER.
- If ER is not programmed, axial travel from the initial Z to dimension EH is at the work rate.

If dimension ER is between dimension EH and the hole bottom dimension Z, the system generates an error (error 276).

8.3.5 – Pocket data file

- Processing of pocket dimensions requires a large amount of computation time.

  With a concern for performance, the computations are conducted once and for all the first time the pocket is executed and the results are stored for future reuse.

  These preliminary calculations are only possible however in the following modes :

  . Test mode,

  . Graphic trace (direct graphic or graphic called by the conversational mode).

They are only carried out the first time the cycle is executed in test or graphic mode following :

. Program load,

. Modification of a pocket definition parameter (contour, width of cut, etc.).

Direct execution in CONT, STEP, RAPID or SEQNO mode immediately after a load or a pocket definition modification generates an error (error 263).

After the pocket definition computations have been completed, a file is associated with the current part program is created. This file contains :

- the pocket definition data,
- the results of the definition calculations.

If a part program contains several pocket occurrences, the data for the pockets are stored in the same data file.

The pocket data file is managed by the CNC like a part program file. It has the same number as the part program from which it results, but this number is followed by the suffix 9. For instance, if a pocket is programmed in part program %100, the pocket data file associated with the program is number %100.9.

. This file is displayed on the LIST page or MODIF page as an empty file. Only one line corresponding to this number (%) is displayed.

. This file is remanent. It is not deleted at the end of the program. It can however be deleted by the operator like any part program. It is automatically deleted if the part program is deleted.

However, the pocket data file cannot be transferred via the DNC link or printed out on tape.

8.3.6 – Sequence number search

The blocks generated internally from machining commands are numbered and can therefore be called in SEQNO mode.

The syntax is as follows :

If N100 is the sequence number of a roughing instruction, N100–5 represents the fifth block created by the sequencing instruction. This number is given on the current page. The operator can search for this block in SEQNO mode by command N 100 ; –5. The minus sign (–) before the 5 is mandatory.

8.3.7 – Performance

- Memory space used :

  A pocket data file associated with a part program occupies a memory space of at most 28 bytes per elementary block created.

  Immediately after load or editing, an additional 40 kbyte temporary workspace is required (although less memory space will be used).

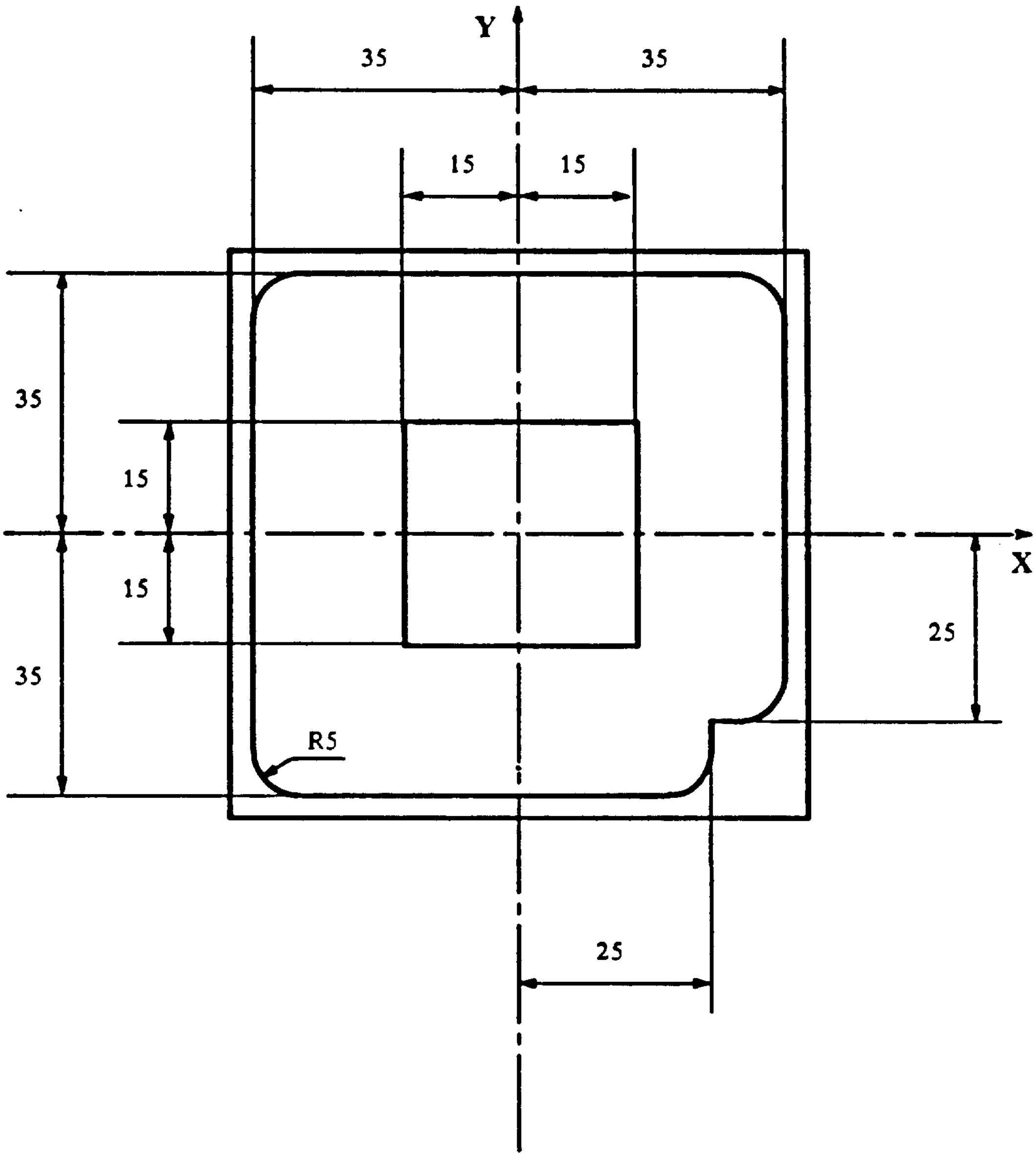
- Limits :

  For a given pocket, a maximum number of contours is 127.

  The maximum X or Y dimension of a pocket must be less than 40 times the width of cut.

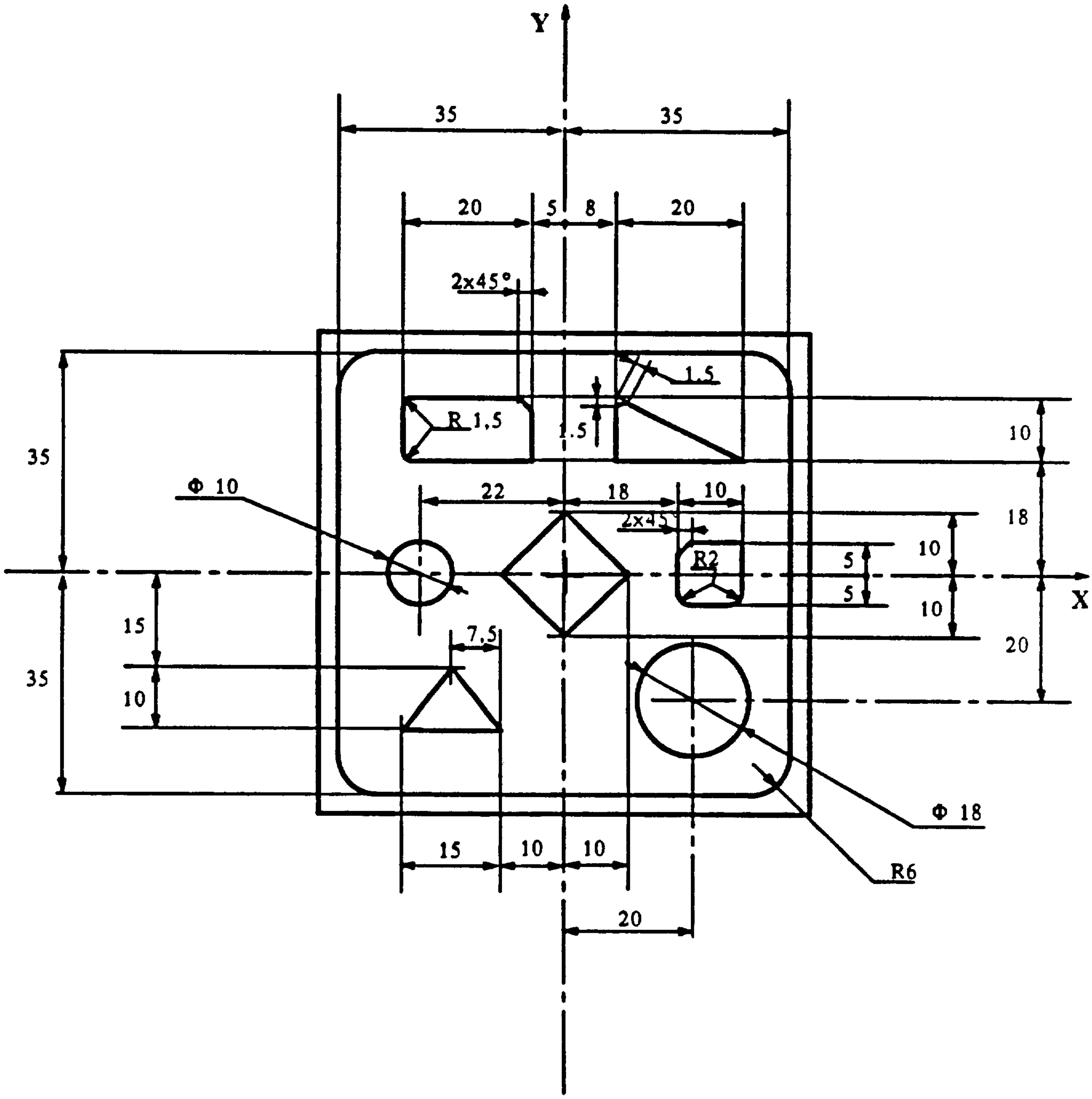
8.3.8 - Example

8.3.8.1 - Pocket with Island



```
% 105 (POCKET. P105. POC)

EM-X-40 Y-40 Z-10 EM+ X40 Y40 Z2                 Dimension of the parallelepiped
N5 G G52 X Y Z M3 S2500 D7 E52007=4000           for 3D display

N10 G46 NU0 NP1 ED8 Q5 J.5 NR+ R03 LX25 LY25 EX-25 EY-25 Geometric definition header

N100 G46 NU1                                     Pocket contour definition
N110 G1 X35 Y
N120 Y-25 EB5
N130 X25
N140 Y-35 EB5
N150 X-35 EB5
N160 Y35 EB5
N170 X35 EB5
N180 Y

N200 G46 NU2                                     Island contour definition
N210 G1 X15 Y15
N220 Y-15
N230 X-15
N240 Y15
N250 X15

N300 G46 NU9                                     End of geometric definitions

N350 G G52 XYZ
N360 T1 M6 D7 (PERC. DIA 8)
N400 G46 NU10 NP1 G83 Z-4 P2 Q1 ER2 F300         Peck drilling instruction
                                                 (drilling point given by LX and LY of N10)

N450 G G52 XYZ
N460 T2 M6 D7 (EB. DIA 8)
N500 G46 NU15 NP1 Z-5 P3 ER2 EP250 EQ350         Roughing instruction
                                                 (start and end points given by N10)

N550 G G52 XYZ
N560 T3 M6 D7 (DEMI FINI DIA 8)
N600 G46 NU20 NP1 Z-5 P4 ER1 EI200 EJ300 J.2     Semi-finishing instruction
                                                 (allowance 0.2 mm)
N650 G G52 XYZ
N660 T4 M6 D7 (FINI DIA 8)
N700 G46 NU20 NP1 P7 ER2 Z-5                     Finishing instruction

G G52 Z
G52 XY
M2
```

8.3.8.2 – Pocket with several islands



%100 (POCKET. P100. POC)

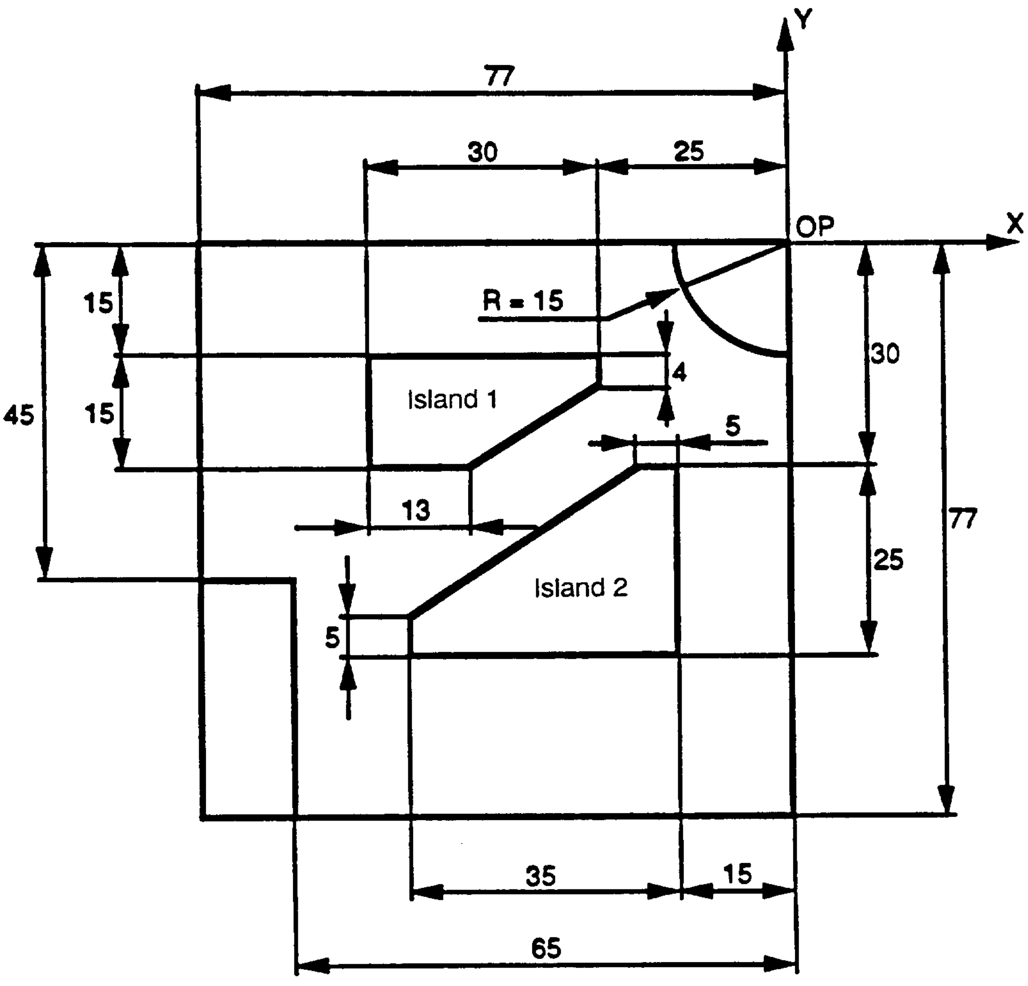
EM-X-40 Y-40 Z-10 EM+ X40 Y40 Z2 G G52 Z M3 S2500	Dimension of the parallelepiped for 3D display

N10 G46 NU0 NP1 ED5 Q3 J.2 NR+ R03 LX10 LY10 EX-10 EY-10 Geometric definition header

N100 G46 NU1 LX0 LY-35 N105 G1 X35 Y10 N100 Y-35 EB6 N120 X-35 EB6 N130 Y35 EB6 N140 X35 EB6 N150 Y10	Definition of the pocket contour : LX and LY : start point for finishing this contour
N200 G46 NU2 N205 G1 X Y10 N210 X10 Y N220 Y-10 X N230 X-10 Y N240 X Y10	Definition of the first island (central square)
N250 G46 NU2 LX11 LY-20 N260 G1 X29 Y-20 N270 G2 X29 Y-20 I20 J-20	Definition of the second island (circle in bottom right-hand corner) : LX and LY, start points for finishing this contour
N280 G46 NU2 N290 G1 X28 Y5 N300 Y-5 EB2 N310 X18 EB2 N320 Y5 EB-2 N330 X28	Definition of the third island (axial square on the right)
N340 G46 NU2 N350 G1 X8 Y18 N352 X28 N354 X8 Y28 EB-1.5 N356 Y18	Definition of the fourth island (triangle in the top right-hand corner)
N358 G46 NU2 LX-15 LY18 N360 G1 X-5 Y18 N365 Y28 EB-2 N370 X-25 EB1.5 N380 Y18 EB1.5 N390 X-5	Definition of the fifth island (rectangle in the top left-hand corner) : LX and LY, start point for finishing this contour
N400 G46 NU2 N410 G1 X-17 Y N420 G3 X-17 Y0 I-22 J	Definition of the sixth island (axial circle on the left)
N430 G46 NU2 LX-15 LY-25 N440 G1 X-10 Y-25 N450 X-25 N460 X-17.5 Y-15 N470 X-10 Y-25	Definition of the seventh island (triangle in the bottom left-hand corner) : LX and LY, start point for finishing this contour
N500 G46 NU9	End of geometric definitions

N502 G G52 XYZ	
N504 T1 D3 M6 (PERC DIA 5) E52003=2500 M00	
M3	
N600 G46 NU10 NP1 G83 Z-5 P2 Q1 ER2 F300	Peck drilling instruction. Drilling point given by LX and LY of N10
N602 G G52 XYZ	
N604 T2 D3 M6 (EB DIA 5) E52003=2500 M00	
M3	
N700 G46 NU15 NP1 Z-5 P3 ER2 EH0 EP250 EQ350	Roughing instruction
N702 G G52 XYZ	
N704 T3 D3 M6 (DEMI FINI DIA 5) E52003=2500 M00	
M3	
N800 G46 NU20 NP1 Z-5 P4 ER1 EI200 EJ300 J.15	Semi-finishing instruction
N802 G G52 XYZ	
N804 T4 D3 M6 (FINI DIA 5) E52003=2500 M00	
M3	
N900 G46 NU20 NP1 ER1 P5 Z-5 EH0	Finishing instruction
G G52 Z	
G52 X Y	
M2	

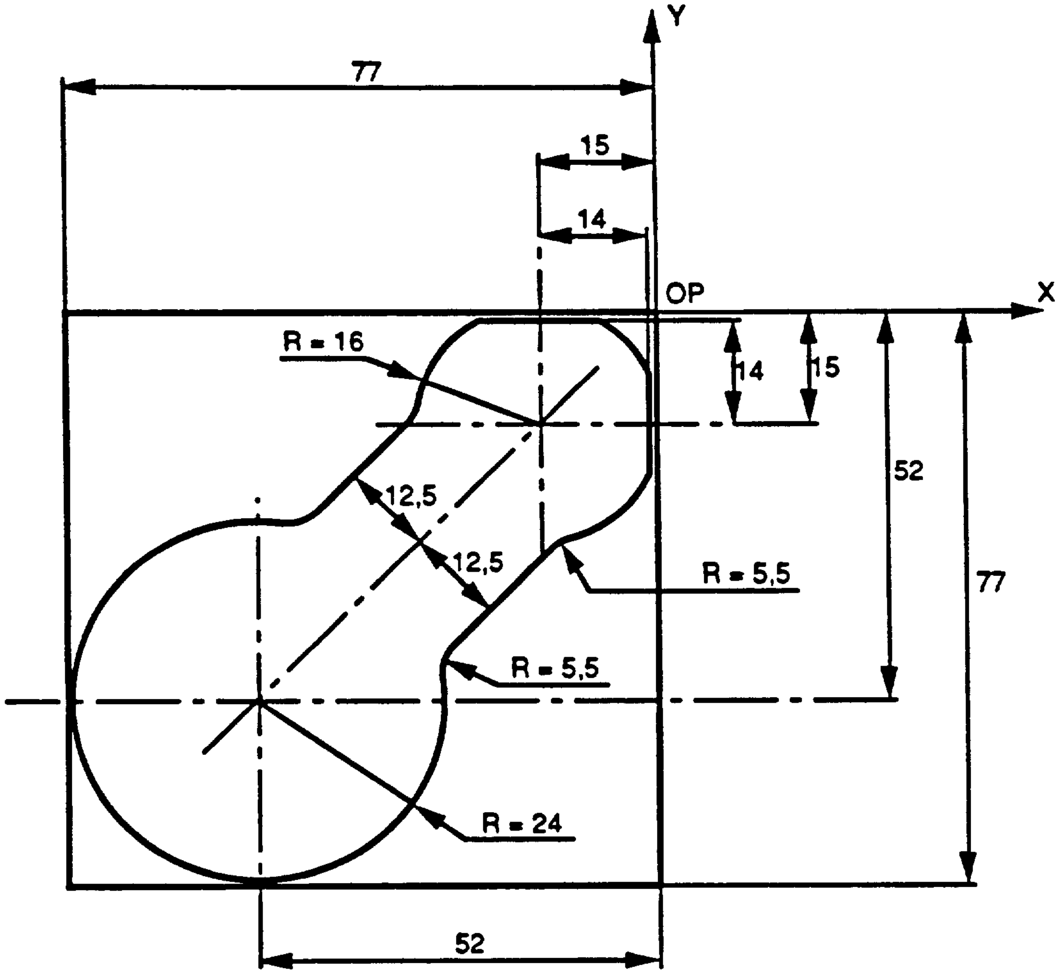
8.3.8.3 – Facing with walls and islands



%450	
EM-X-77Y-77Z-10 EM+ XYZ	Dimensions of the parallelepiped for 3D display
E52001 = 5000	
S2500	
N10 G G52 X Y Z40 M3 D1	
N20 G46 NU0 NP1 ED 10 Q6 J.2	Start of definition
N30 G46 NU5	Description of the external profile
N40 G1 XY-15	
N50 Y-77	
N60 X-65	
N62 G46 NU6	
N64 X-65 Y-77	
N70 Y-45	
N80 X-77	
N82 G46 NU5	
N84 X-77 Y-45	
N90 Y	
N100 X-15	
N102 G46 NU6	
N104 G1 X-15 Y	
N110 G3 XY-15 IJ	
N120 G46 NU2	Island 1 profile
N130 G1 X-55 Y-15	
N140 X-25	
N150 Y-19	
N160 X-42 Y-30	
N170 X-55	
N180 Y-15	
N190 G46 NU2	Island 2 profile

N200 G1 X−15 Y−30	
N210 Y−55	
N220 X−50	
N230 Y−50	
N240 X−20 Y−30	
N250 X−15	
N260 G46 NU9	End of definition
N275 G G52 X Y Z40	
N280 G46 NU 15 NP 1 Z−5 P3 ER2 EH0 EP250 EQ400	Roughing
N285 G G52 X Y Z40	
N290 G46 NU20 NP 1 Z−5 P5 ER2 EH0 EI230 EJ350 J.1	Semi−finishing contouring
N295 G G52 X Y Z40	
N300 G46 NU20 NP 1 Z−5 P5 ER2 EH0 EI200 EJ320	Finishing contouring
M2	

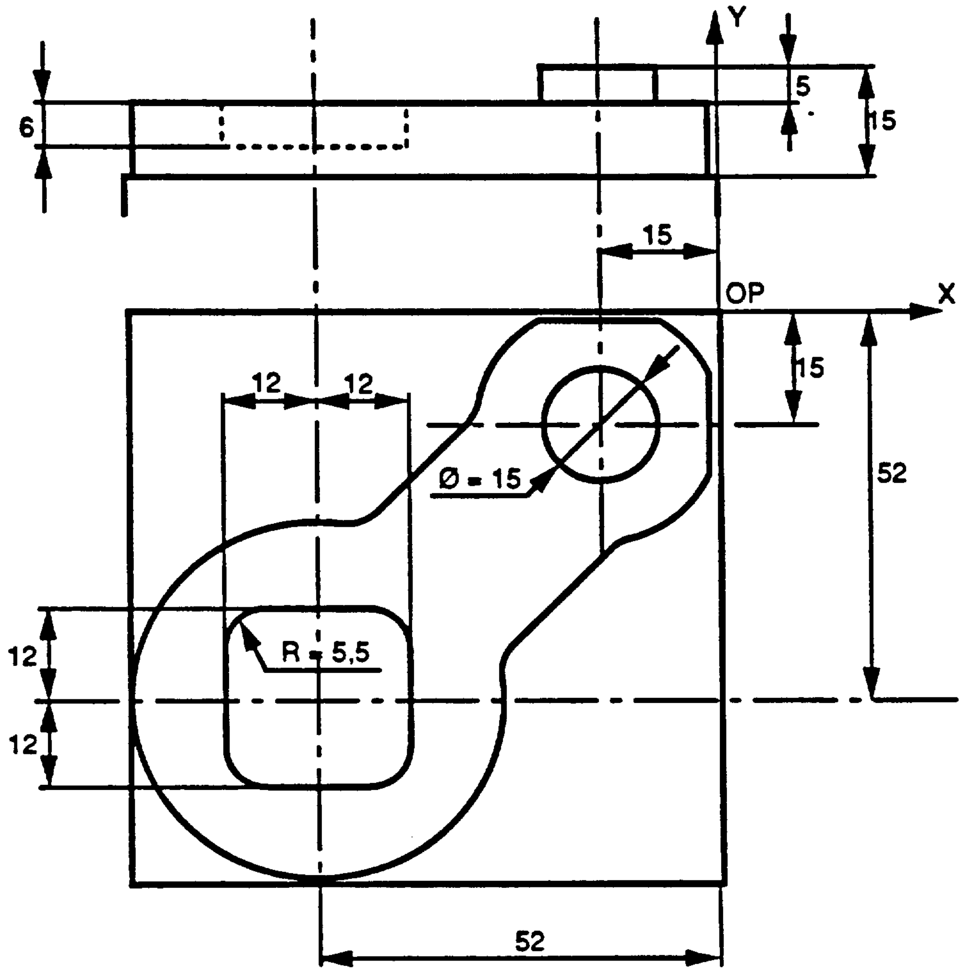
8.3.8.4 – Facing with island



%451 (Facing with island) – with %452 sub-program	
EM-X-77Y-77Z-20 EM+ XYZ	Dimensions of the parallelepiped for 3D display
E52001 = 5000	
S2500	
N10 G G52 X Y Z40 M3 D1	
N20 G46 NU0 NP1 ED 10 Q4 J.2	Start of definition
N30 G46 NU3	Part limit
N40 G1 XY	
N50 Y-77	
N60 X-77	
N70 Y	
N80 X	
N90 G46 NU2	Island profile
N100 G77 H452	
N110 G46 NU9	
N120 G G52 X Y Z40	
N130 G46 NU 15NP1Z-15.5 P10 ER2 EH0 EP250 EQ400	Roughing
N135 G G52 X Y Z40	
N140 G46 NU20 NP1Z-15.5 P15 ER2 EH0 EI230 EJ350 J.1	Semi-finishing contouring
N145 G G52 X Y Z40	
N150 G46 NU20 NP1Z-15.5 P15 ER2 EH0 EI200 EJ230	Finishing contouring
N160 G G52 X Y Z40	
N550 M2	

```
%452 (Sub-program - Description of the island profile)
N10 G1 X-1Y-15
N20 EA-90 ES+
N30 G2 I-15 J-15 R16 ES+ EB5.5
N40 G1EA225 X-19.445 Y-37.123
N50 EA225 ES-EB5.5
N60 G2 I-52 J-52 R24 ES+ EB5.5
N70 G1 EA45 X-37.123 Y-19.445
N80 EA45 ES-EB5.5
N90 G2 I-15 J-15 R16 ES-
N100 G1 EA0 X-15 Y-1
N110 EA0 ES+
N120 G2 I-15 J-15 R16 ES-
N130 G1 EA-90 X-1 Y-15
```

8.3.8.5 – Facing with island and recess



%455 (Facing with island and recess) – With %451 and %452 sub–programs

EM– X–77 Y–77 Z–20 EM+ XYZ	Dimensions of the parallelepiped for 3D display
S2500	
E52001 = 5000	
N10 G G52 X Y Z40 M3 D1	
N20 G46 NU0 NP1 ED10 Q4 J.2	Start of definition
N30 G46 NU3	Facing limit
N40 G77 H452	
N50 G46 NU2	Island profile
N60 G1 X–7.5 Y–15	
N70 G3 X–7.5 Y–15 I–15 J–15	
N150 G46 NU9	
N170 G G52 X Y Z40	
M00	
M3	
N180 G46 NU15 NP1 Z–5 P3 ER2 EH0 EP250 EQ400	Roughing
N190 G G52 X Y Z40	
N200 G46 NU20 NP1 Z–5 P5 ER2 EH0 EI230 EJ350 J.1	Semi–finishing contouring
N210 G G52 X Y Z40	
N220 G46 NU20 NP1 Z–5 P5 ER2 EH0 EI200 EJ320	Finishing contouring
N230 G G52 X Y Z40	
M00	
M3 D1	
N240 G46 NU0 NP2 ED10 Q4 J.2	Start of definition
N250 G46 NU1	Pocket contour limits
N260 G1 X–40 Y–52	
N270 Y–64 EB5.5	
N280 X–64 EB5.5	
N290 Y–40 EB5.5	

N300 X-40 EB5.5	
N310 Y-52	
N311 G46 NU4	Recess limit
N312 G1 X-50 Y-52	
N313 Y-54 EB5.5	
N314 X-54 EB5.5	
N315 Y-50 EB5.5	
N316 X-50 EB5.5	
N317 Y-52	
N320 G46 NU9	
N330 G G52 X Y Z40	
N334 G46 NU10 NP2 G83 Z-11 P2 Q1 ER2	Peck drilling instruction
N338 G G52 X Y Z40	
N340 G46 NU15 NP2 Z-11 P3 ER2 EH0 EP250 EQ400	Roughing
N350 G G52 X Y Z40	
N360 G46 NU20 NP2 Z-11 P5 ER2 EH0 EI230 EJ350 J.1	Semi-finishing contouring
N370 G G52 X Y Z40	
N380 G46 NU20 NP2 Z-11 P5 ER2 EH0 EI200 EJ320	Finishing contouring
N390 G G52 X Y Z40	
M00	
M3 D1	
N400 G46 NU0 NP3 ED10 Q4 J.2	
N410 G46 NU3	Facing limit with island
N420 G77 H451	
M2	

8.4 – RIGID TAPPING CYCLE : FUNCTION G84

With rigid tapping, the spindle speed controls the tap feed.

IMPORTANT : If the system includes this option, do not use the normal G84 tapping cycle.

At the start of axial travel, the tool axis is coupled with spindle rotation : the spindle starts and stops during tapping according to the deceleration distance. During this cycle, the tool axis following error is cancelled. In the hole bottom, the spindle does not completely stop rotating to prevent blocking in the bottom of the hole. The tool entry and exit feedrate is constant except in the deceleration region.

8.4.1 – Programming

The rigid tapping cycle is called by modal function G84 and cancelled by the functions of series G80–G83 and G85–G89.

General Format

G84 X+053 Y+053 Z+053 K053 EK053

This programming corresponds to the XY plane (G17)

X, Y : tool positioning dimensions
Z : hole bottom dimension
K : tap pitch in mm (K is programmed to specify rigid tapping)
EK : spindle in/out speed ratio. If EK is not included in the block, a default ratio of 1 is applied.

Rigid tapping can be carried out along any one of the three main axes.

Plane defined	Tool position	Threading axis	Thread pitch	Circular interpolation plane for positioning
G19	G16 P ±	X	K	YZ
G18	G16 Q ±	Y	K	ZX
G17	G16 R ±	Z	K	XY

Rigid tapping can be carried out over several penetrations, but this requires programming several blocks :

```
G84  M3  M41  S400  X...  Y...  Z1  K...  EK...
     Z2
     Z3
```

CAUTION :

- If the tapping inhibit input (I0 of PLC W4) is disabled, action on FEED STOP causes premature retraction of the tap before reaching the hole bottom. If input I0 is enabled, tapping continues normally.

8.4.2 – Example of programming

%800	
N10 G16 R+ D1 S200 M3	Z machining axis
N20 G0 X Y Z	
N30 G84 X10 Y5 Z–20 K2 EK2	Rigid tapping set to X10 Y5 in plane G17, depth 20, pitch 2.
N40 ER60	
N50 X35	Rigid tapping in X35
N60 G80 XYZ	
N70 G16 R– X20 Z10	Z machining axis machining coordinate X20 Y0 Z10
N80 G84 Z60 K2 EK2	Rigid tapping in plane G17, depth 50, pitch 2.
N90 X55	Rigid tapping in X55.
N100 G80 XY	
N110 G16 P– X5 YZ	X machining axis
N120 G84 Y10 Z10 X40 K2 EK2	Rigid tapping, set to Y10 Z10 in plane G19, depth 40, pitch 2.
N130 G80 XYZ	

NOTES

9 - PARAMETRIC PROGRAMMING

Programmed parameters are functions which can be assigned to all addresses instead of numerical values. They can also be used by the programmer as special functions.

There are two categories of parameters :

- program variables L,
- program parameters E.

9.1 - PROGRAM VARIABLES L

Two types of program variable are available :

- L0 to L19 variables
- L100 to L199 and L900 to L939 variables.

These variables are identical in terms of format and use, but can differ in the way they affect the running of the workpiece program.

9.1.1 - Variables L0 to L19

They are initialized, (set to 0) ; - when the system is switched on, at the end of the execution of the workpiece program (MO2), or by pressing the "RESET" button.

These variables can be a fixed value or they can be product of the following operations : - addition (+), subtraction (-), multiplication (*), division (/), square root (R), sine (S), cosine (C), truncation (T), arctangent (A), logical AND (&), logical OR (!).

- Their use does not restrict program writing in any way.
- The values are "floating point". (ie they can have a decimal part).
- AND and OR operations are carried out on truncated values. (i.e. that part of the value which follows the decimal point is automatically removed. Thus, the value is NOT rounded to the nearest whole number).
- A maximum of 8 digits can be programmed, excluding the sign. The position of the decimal point is irrelevant.

L8 =	18	Assigning an L parameter L to an NC address
XL8 =	X18 mm	involves equating the unit of L with the unit of
FL8 =	F18 mm/mn	the corresponding address.

Program variables can be assigned to all programmable addresses, dimensions and functions. They can be used with G79, to carry out conditional program jumps.

Example of using a parameter

L2 = 5
L1 = L2 + 5.3 * 3 * S30 (Simple left to right execution gives, L1 = 15.45)
XL1 (The value of L1 is applied to the X dimension. ie. X15.45 mm)

9.1.2 – Variables L100 to L199 and L900 to L939

These variables are identical in format and use to variables L0–L19. Using them, however, can have an effect on the running of the workpiece program.
Writing to one of the L100 to L199 variables will delay the preparation of that block until the PRECEDING block has been completed. (This is not so with the L0 to L19 variables). This means that a block whose execution takes into account the contents of the following block, (eg : geometric programming of the profile over 2 or 3 blocks, or tool radius compensation) cannot come just before a block which reads from or writes to one of variables, L100 to L199.

- However, by controlling access to PROGRAM EDIT, MDI modes and sub-program calls by the PLC, from the part program,

  the variables L100 to L199 and L900 to L939 may be used in the same way as variables L0 to L19. This uses M function M999.

  This M function prevents the operator or the PLC from intervening in the progress of a series of blocks.

  EXAMPLE :

```
%1
.
.
.
.
N 90  X Y ...
N 100 L110 = 1 + L110 ....
N 110 X L110 .......
N 120 L120 = L110 + L10
N 130 Y L120
.
.
.
.
```

```
%2
.
.
.
.
N 80 M999
N 90 X Y ...
N 100     L110 = 1 + L110 ...
N 110     XL110
N 120     L120 = L110 + L10
N 130     Y L 120
N 140     M998
.
.
```

  In program 1, block 100 will not be prepared and carried out until block N90 is finished.

  In program 2, block 100 will be prepared before execution of N80 and there will be no "stop" at the end of block N90.

ADDITIONAL ASPECTS OF USING PROGRAM VARIABLES :

The variables L900 to L925 correspond respectively to the letters of the alphabet, A to Z, if these letters are immediately followed by an equals sign. For example, programming C = 12.34, is the same as programming L908 = 12.34. This feature is most useful in developing custom macros.

As already explained, when function M999 is programmed, the operator cannot call EDIT and MDI modes, nor can the PLC call a sub-program until the M998 function has been encountered in the program.

However, manual movements, (JOG or RECALL), the overspeed function, (G12), and the premature block termination function, (G10), are allowed during M999.

When the system is switched on, or after a RESET or MO2, the initialized state is M998. (The M999 function is cancelled by M998 or M997). Although it is possible to search to a block within a sequence covered by M999, it is not possible to enter search mode from a series of "masked" blocks.

- Function M997 :

  The M997 function is used to force block continuation. If the system is in block by block mode when M997 is read, the following blocks will be run as if the system were in continuous mode until the program meets a cancelling function (M998 or M999) or M02 (reference to para. II. II).

9.2 - PROGRAM PARAMETERS

External parameters are defined by address E, followed by 5 digits. The most significant digit specifies the type of parameter.

9.2.1 - The different types of program parameters

Type 2 : E200XX : Information on bits, written by the interface and read by NC.

XX : Bit No. from 0 to 31.

E21XXX : Information on bits, read by NC : software option present (=1) or absent (=0).

XXX : Bit No. from 0 to 127

Type 3 : E31000 : These two read/write parameters are used to choose the type of graphic

E31001 trace of the part.

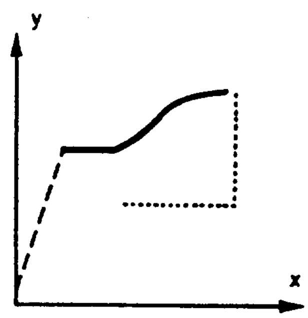
E31000 is relative to G0.
E31001 is relative to G1, G2 or G3.

The trace types are :
0 : solid line
1 : dotted line
2 : dashed line
3 : combined line
4 : no line (stylus raised)

These parameters are set to zero by graphic initialization of the system. Setting a value less than 0 or greater than 4 results in error 92.

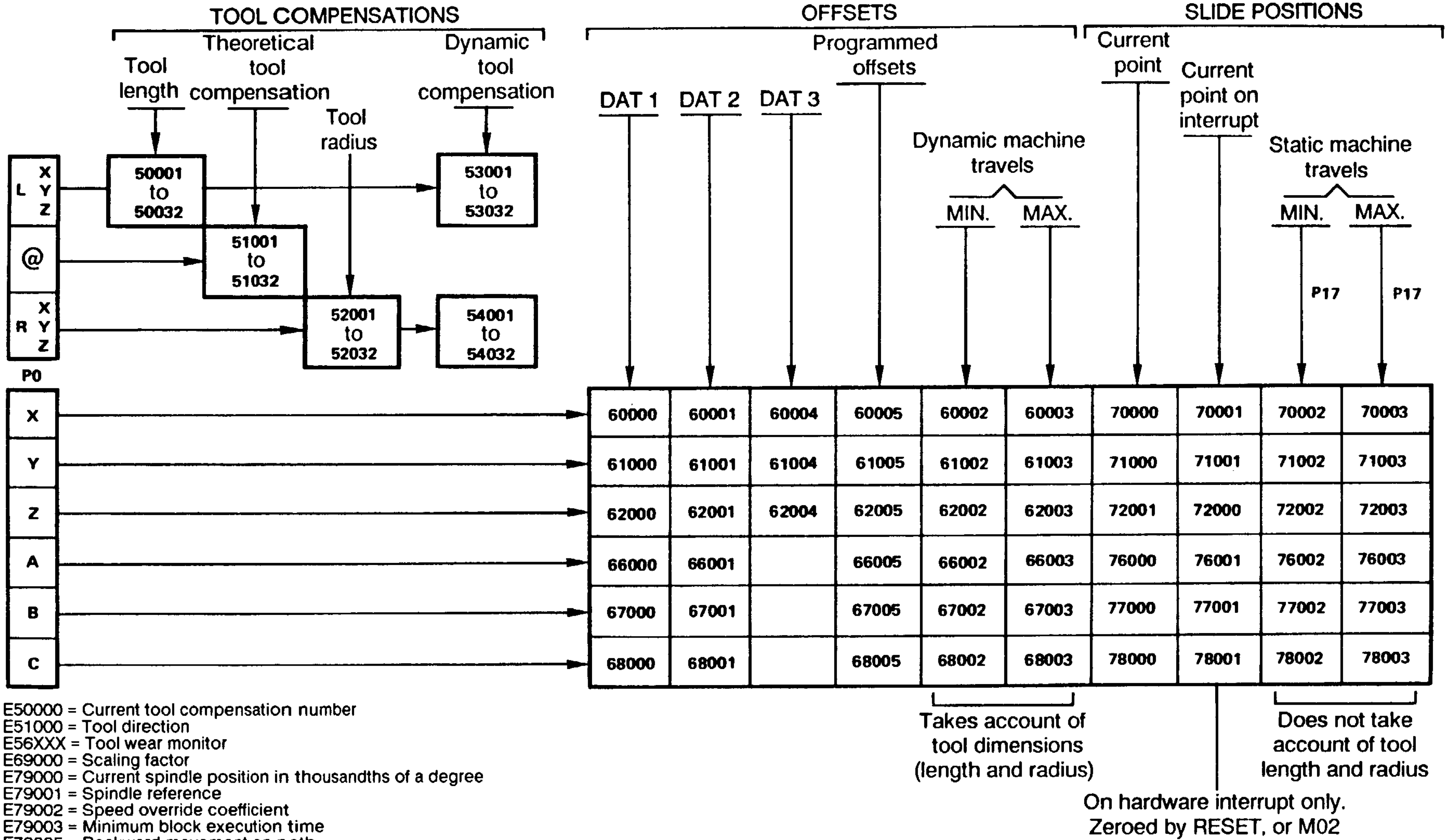
Example :

```
        %15
N10 E31000=4 XYZ
N20 G0 X50 Y50                        No line (stylus raised)
N30 E31001=1 G1 X100                  Dotted line
N40 Y100
N50 E31001=0 G3 I100 J50 R50 ES+ EB20 Solid line
N60 G1 EA180 X25 Y75
N70 E31000=2 G0 XY M2                 Dashed line
```



Type 4 :	E40000	:	Pure binary-coded data, by 32 bit word, written by the interface and read by NC. This parameter encompasses parameters E20000 to E20031.
	E41000	:	Current mode. Read-only parameter.
	E41001	:	Axis group number. Read only parameter.
	E41002	:	Job reference. Read-only parameter.
	E41003	:	Graphic simulation status. Read-only parameter = 0 no simulation = 2 3D graphic display
Type 8 :	E80XXX	:	Local NC read/write data.
			Data No. from 0 to 49.
Type 9 :			For these parameters the least-significant digit defines the order of the machine axis in decimals (0 to 3).
	E9000X	:	Axis measurement.
	E9010X	:	Datum switch position in machine dimensions. This parameter is the image of machine parameter P16. Read-only parameter. It can be used to create an automatic MOS programme by calculating the distance between the encoder zero pulse and the datum switch edge.
	E9100X	:	Status of the axes : servo-controlled or not
	E9200X	:	Enables or disables sensing of the m/c origin datum switches.
	E9300X	:	State of m/c origin datum switch.
	E9110X	:	State of the MOS on an axis.
	E91113	:	Spindle origin measurement forced. If = 1, stops the spindle and requires origin measurement. Is automatically reset after origin measurement.

NOTES



P0	DAT 1	DAT 2	DAT 3	Programmed offsets	Dynamic MIN.	Dynamic MAX.	Current point	Current point on interrupt	Static MIN.	Static MAX.
X	60000	60001	60004	60005	60002	60003	70000	70001	70002	70003
Y	61000	61001	61004	61005	61002	61003	71000	71001	71002	71003
Z	62000	62001	62004	62005	62002	62003	72001	72000	72002	72003
A	66000	66001		66005	66002	66003	76000	76001	76002	76003
B	67000	67001		67005	67002	67003	77000	77001	77002	77003
C	68000	68001		68005	68002	68003	78000	78001	78002	78003

E50000 = Current tool compensation number
E51000 = Tool direction
E56XXX = Tool wear monitor
E69000 = Scaling factor
E79000 = Current spindle position in thousandths of a degree
E79001 = Spindle reference
E79002 = Speed override coefficient
E79003 = Minimum block execution time
E79005 = Backward movement on path
E7X004 = Direction of movement
E7X005 = Axis assignment

9.2.2 – Recap of program parameters

NC REGISTERS		WORKPIECE PROGRAM	PLC REGISTERS
Tool compensations	E50001 to E50032 E51001 to E51032 E52001 to E52032 E53001 to E53032 E54001 to E54032		
Current tool compensation number	E50000		
Tool direction	E51000		
Tool wear monitoring	E56001 to E56032		E20000 to E20031 – Information on bits read by NC – 32 bits if used in one single parameter correspond to E40000
DAT 1	E60000 to E68000		
DAT 2	E60001 to E68001		
DAT 3	E60004 to E62004		
Minimum dynamic machine travel	E60002 to E68002		
Maximum dynamic machine travel	E60003 to E68003		
Programmed offsets	E60005 to E68005		
Current point	E70000 to E78000		
Current point upon interrupt	E70001 to E78001		
Minimum static machine travel	E70002 to E78002		
Maximum static machine travel	E70003 to E78003		
Allocation of axes	E70005 to E78005		
Local data 50 x 32 bits	E80000 to E80049		
Axes and datum switches	E90000 to E90003 E91000 to E91003 E92000 to E92003 E93000 to E93003 E91100 to E91103		

See table on preceding page

9.2.3 – Operating precautions

The program parameters are never reset to zero by the system.

Unlike the program variables, which are parameters linked solely to the workpiece programs, the use of program parameters is subject to several restrictions :

- Type 7 parameters are "read only".
- Any operation, (read or write), carried out on a program parameter stops preparation of the program until the end of the preceding block. Therefore, a block containing a program parameter cannot be preceded by a block whose execution requires "knowledge" of the contents of the following block or blocks (geometric programming, radius compensation, etc ...)
- The value of a program parameter is ALWAYS an integer. (Unlike the program variables, L)
- When an external parameter is assigned to an NC address, it equates the parameter unit with the decimal part of the corresponding function :

  If E80000 = 18000
  XE80000 ---> X18000 ---> X = 18 mm
  FE80000 ---> F18000 ---> F = 180 mm/min

- An E parameter can itself be parameterized
  If L0 = 80003, then EL0 corresponds to E80003, so EL0 = – 123.49 will put a value of – 123 into E80003. (Remember that the system will put an integer value into the E parameter, and will automatically round up or down to the nearest whole number).

9.2.4 – Using type 9 program parameters

Remember that an operation on one of these parameters stops movement at the end of the block preceding the block in which they are programmed.

9.2.4.1 – Axis measurement : E900xx

The absolute position of all axes, (with respect to m/c origin), can be read, irrespective of whether the axes are measuring only or fully servo–controlled.
However, only axes which are measuring only, and not servo–controlled, can be written to.

9.2.4.2 – Status of the axes : servo–controlled or not servo–controlled : E910xx

This parameter is used to modify the axis status as originally defined by machine parameter P3, (P3 retains is original value). The status of all machine axes can be read. In write mode, all the axes can be assigned servo–controlled or not servo–controlled.
When the value of the parameter is "1", the axis is servo–controlled. "0" means that it is not servo–controlled.

NOTE :

When the system is re–initialized from power up or reset or M02, the initial status of the axes, as defined by P3, is restored.

9.2.4.3 – Taking notice of the m/c origin datum switches : E920xx

This command status can be read from all axes of the machine which have position feedback.
This command can be written to measuring only axes, (which are NOT servo–controlled).
When the value of the parameter is 1, the datum switch is enabled, (0 = disabled).

NOTE :

Disabling is automatic on contacting the reference switch, (during m/c datum setting), or when a reset or an M02 occurs. In m/c referencing mode, it takes a value of 1, as soon as the axis jog pushbutton is pressed, and changes to 0 when the pushbutton is released, or when the datum switch is met.

9.2.4.4 – Origin datum switch status : E930xx

This is a read only parameter which is accessible on all the measured axes of the machine.

A value of "1" indicates that the datum switch is made, "0" indicates that it is not.

9.2.4.5 – Acknowledgement of the MOS on an axis : E911XX

This parameter is used to locate an axis which has been previously referenced.

If the value of the parameter is zero, the axis has been referenced. If the value is 1, the MOS has not been set.

This is a read/write parameter. (If an axis movement is demanded of an axis which has not previously been referenced, ie E911xx is a 1, the system will report an error).

9.2.5 – Special type 5 parameters

- E50000 : Current tool compensation number. (Dxx). Read only parameter.
- E51000 : Tool direction. This parameter gives the physical address of the axis parallel to the direction of the tool. If the tool direction is negative, the value of the axis address is increased by 100. This is a read only parameter.

  EG E51000 = 102 (negative direction on Z axis)
  E51000 = 2 (positive direction on Z axis).

- E560XX : additional element (H) in the table of dynamic tool compensations, which can be used to monitor tool wear or other data (max. 8 figures). This is a read/write parameter.

  The dynamic tool compensation page is presented as :

D1	DL + 0.123	DR + 0	H + 12345678
D2	DL + 12.456	DR + 1.325	H + 1200

  The format of DL and DR is 2.3, with a maximum of 99.999.

  The values of H may be entered manually. The appropriate syntax is : Dxx Hxxx (LF).

  This table may be input or output by tape or via the DNC 1 link.

9.2.6 – Special type 7 parameters

- E7n004 : shift direction, n program axis.
  In linear interpolation, this parameter contains the relative dimension along axis n of the current block.

  In circular interpolation, it contains the relative component along axis n of the current radius offset by + or – 90 degrees depending on whether function G3 or G2 is in use. This parameter is read only.

- E7n005 : assigning axes.

  This parameter is the part program equivalent of machine parameter P9. It is used to transfer a program axis from one machine address to another.

  eg. E7n005 = m where n = program axis, (P0 order). m = machine axis, (P9 order).

  If E7n005 = –1, the program axis (n) is uncoupled, ie feed.

  A program axis must be free, before being assigned to another machine axis.

  This is a read/write parameter.

  The original axes assigned by parameter P9 will be restored following a reset or M2 .

- E79005 : Return along trajectory.

  If E79005 = 1, the system is moving backward along the path.

  If E79005 = 0, operation is normal.

  This parameter can be read by the dynamic operator and displayed on the input/output page.

9.2.7 – Using parameters E41000 and E41001

When a workpiece program is run in "test" mode, the following functions are inoperative :

- datum switch stop,
- priority interrupts,
- sub–program calls, by the PLC.

Similarly, in graphic mode, the following operations are not effected :

- writing to the E program parameters,
- sub–program calls, by M function,
- message transmission to the screen, by $0.

As a result, if the program contains conditional return jumps, the program may unintentionally be sent into a permanent loop.

Such problems may be resolved by testing parameters E41000 and E41001.

- E41000 : current mode. (Read only parameter).
  If the system is in test mode, E41000 = 6. If in search mode, E41000 = 4, etc.
- E41001 : axes group number. (Read only parameter). This parameter is equal to 1 in graphic mode, otherwise its value is zero.

9.3 – OPERATIONS ON PARAMETERS

List of operators

Symbol	Operation	Links together the various terms in an expression (1)	Qualifies the following term
+	Addition	yes	yes
–	Subtraction	yes	yes
*	Multiplication	yes	no
/	Division	yes	no
R	Square root	no	yes
S	Sine (2)	no	yes
C	Cosine (2)	no	yes
T	Truncation (3)	no	yes
A	Arctangent (4)	no	yes
&	AND (5)	yes	no
!	OR (5)	yes	no

(1) The operation binds the term following the symbol to the result of the preceding operations (no priority, simply left to right).
(2) The following term is expressed in degrees, or 1/1000ths degree, if an E parameter.
(3) Forms an integer value by removing the decimal part of the term following the symbol.
(4) The result of the operation is expressed in 1/1000ths of a degree.
(5) These logical operations are carried out on automatically truncated values. (i.e. that part of the value which follows the decimal point is removed).

- Conditional jumps

General case : – G79 (Parameter), (Test), (Expression), N...

The system will jump to block N.. if the test is "true", otherwise it will proceed to the next block.

Comparison symbols : = equal
> greater than
< less than

Any TWO symbols may be used together, to create any particular test.

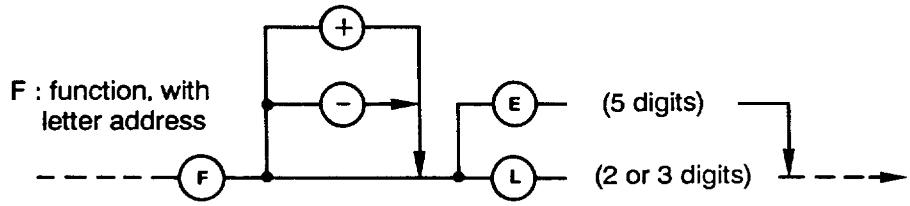
NOTE :

Programming : G79N... (without a test) will produce an unconditional jump.

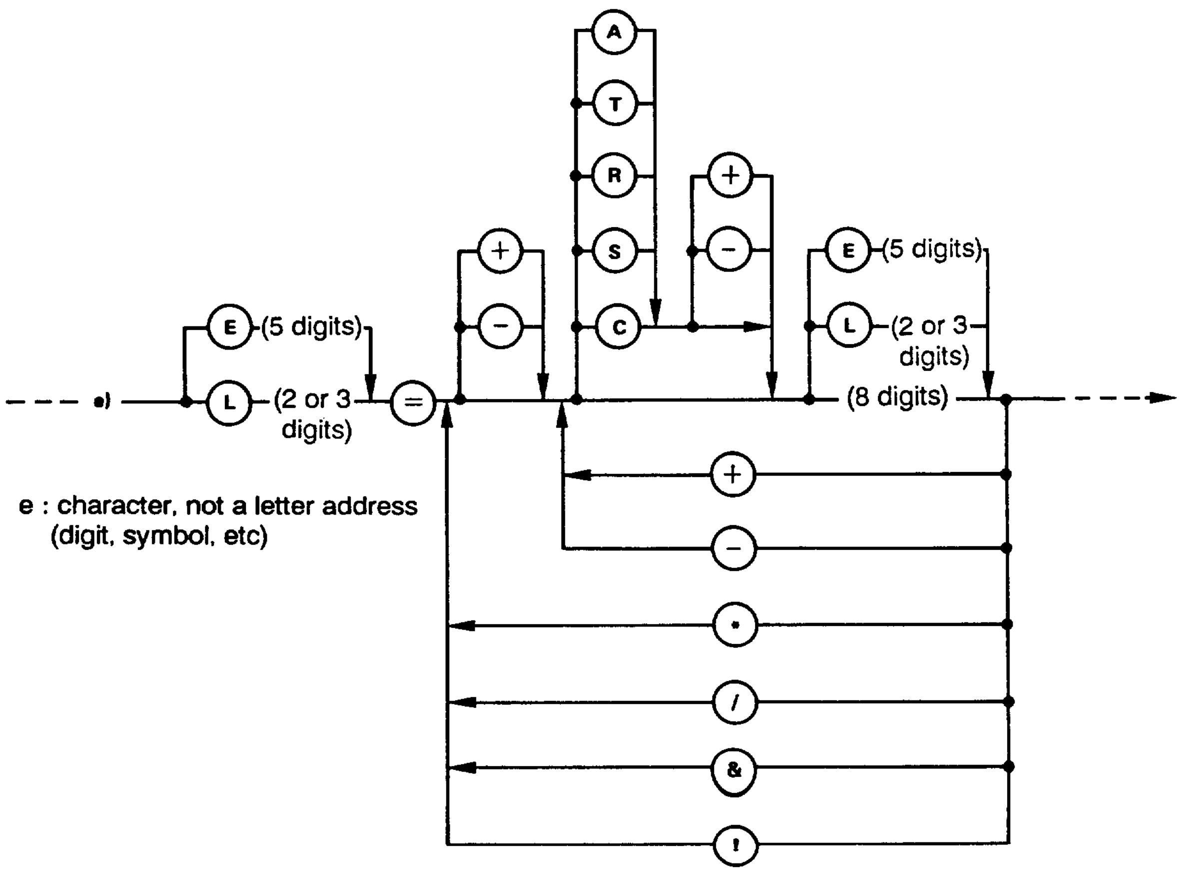
9.4 - SYNTAX OF PARAMETRIC PROGRAMMING

L (2 or 3 digits)	: program variable.
E (5 digits)	: program parameter

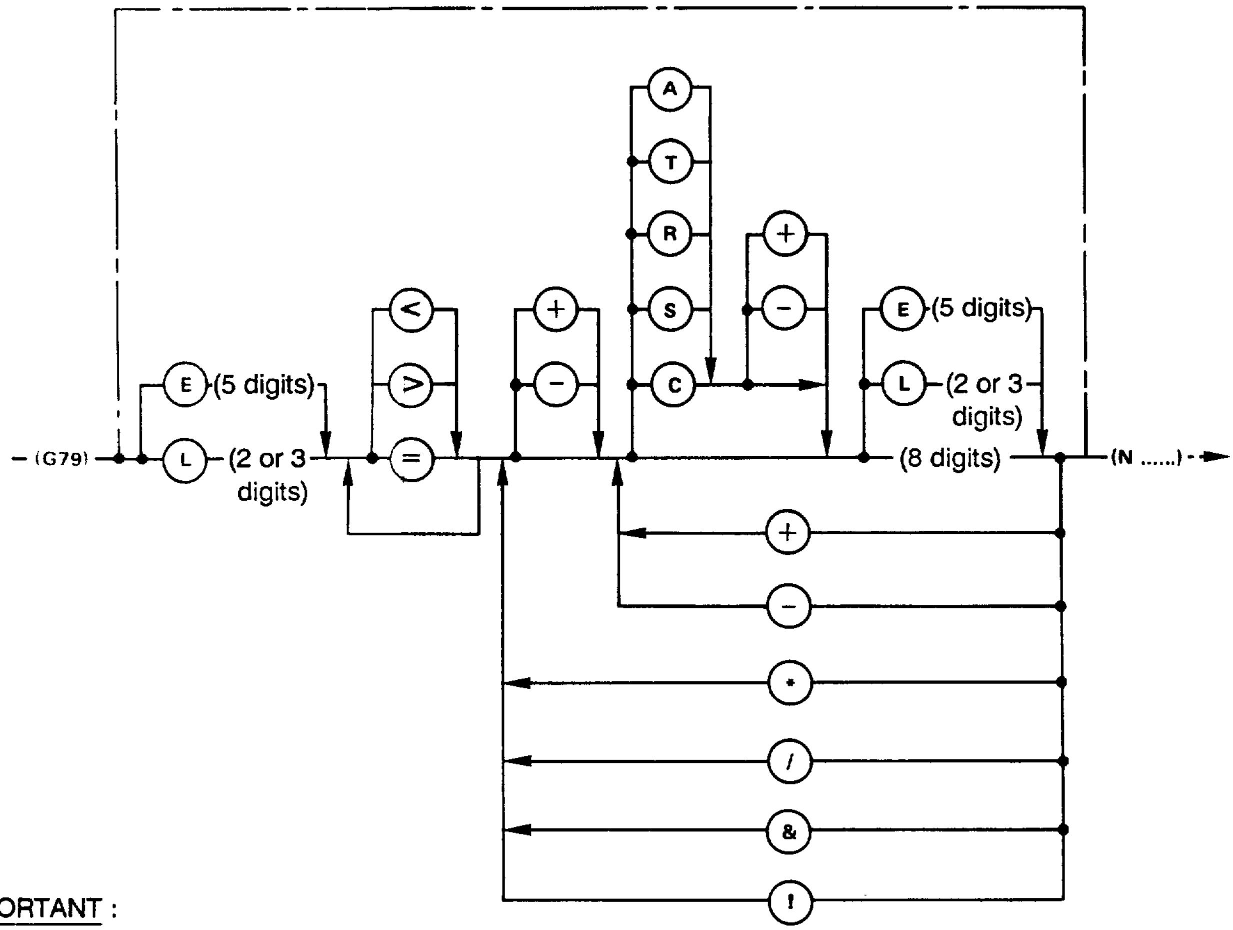
9.4.1 - Assigning a parameter to a function



9.4.2 - Declaring a parameter



9.4.3 – Testing a parameter (comparison with an expression for a conditional jump)



IMPORTANT :

When fractional values are used to increment program variables (L), the result of a test may be unexpected, due to rounding, even though the parameters' displayed value may look correct. Take care over this, particularly when planning a loop.

EG :

L1 = 0
N10 .....
.
.
.
L1 = L1 + 0.6 G79 L1 < 6 N10

After 10 loops, L1 shows 6 on the display. The true result given by the computation, however, is slightly less (perhaps 5.9999999), given that 0.6, held as the computer holds the number, (in binary), is not completely exact. Thus an additional unexpected pass through the loop may be made.

In practice, when incrementing in fractional values, make the test on a deliberately "safe" value :

Eg : for 0.6 with 10 loops, test for 5.9
for 0.1 with 10 loops, test for 0.96
for 0.01 with 10 loops, test for 0.096

9.5 - EXAMPLES USING E PARAMETERS

9.5.1 - Operations with E parameters : Division

The result of the division of an E parameter by another E parameter can be fully stored in an L program variable, because, as E parameters have no decimal part, any results stored in them can only be integers.

EXAMPLE :

E80002 = 3150
E80016 = 2400
L1 = E80002/E80016 (= 1.3125)

If E80005 had been written instead of L1, the stored value would have been 1.

If the result has to be transferred to an E parameter, it will be rounded to the nearest whole number.

If, in the example above, only 3 digits are to be retained after the decimal point, the following will be written :

E80005 = L1 * 1000 then, E80005 will contain 1313

The following could also have been written :

E80005 = E80002/E80016 * 1000. The result is still 1313.

9.5.2 - Using parameter E7(n)001

These parameters are used in conjunction with G10, to save all of the slide positions following an NC priority interrupt, (eg. from a probe or a datum switch stop). The stored values are retained in the E7n001 parameters until a new G10 is encountered, or until M02 or reset.

When a G10 is read, the parameters are automatically pre-loaded with a value of 99999999. This allows a test to be carried out to determine if an interrupt occured or not during the G10 block.

EXAMPLE : Modification of DAT 1 Z (E62000) by probe sensing

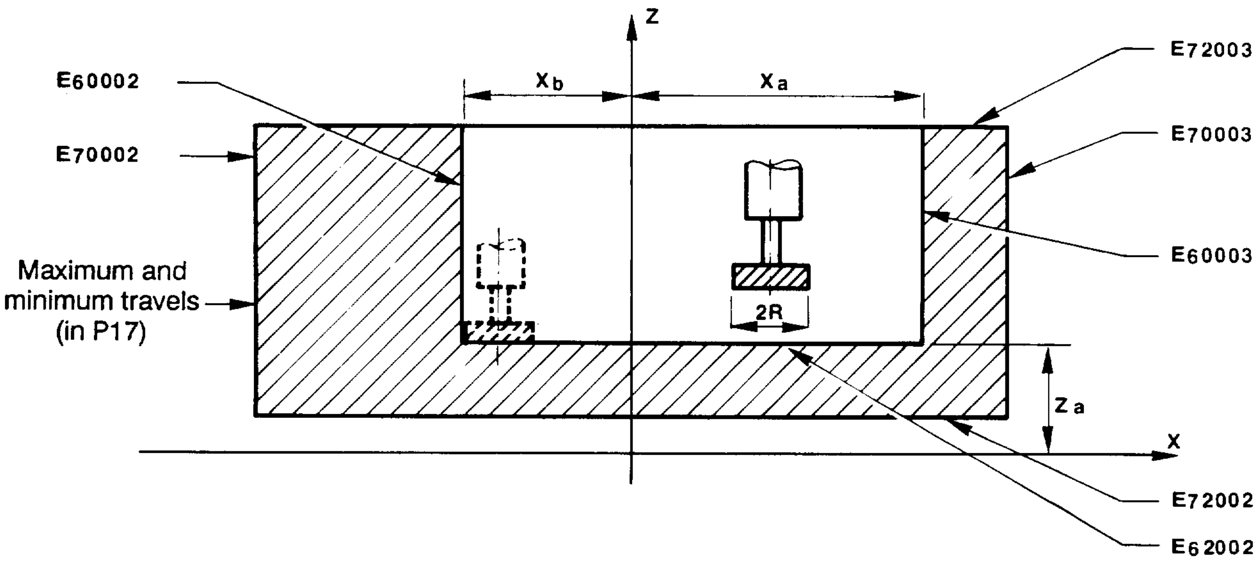
G X0 Y0 Z0	
G Z 85	Rapid approach
G1 G10 Z100 F500	Probing move towards Z100 at 500 mm/mn
G79 E72001 > 90000000 N100	Test whether the interrupt occured or not. (Jump to N100 if it did not).
E62000 = E72001	Load DAT 1 Z with the Z interrupt value.
N100 Continuation of the program	Depending on the test, the program continues with the old DAT 1 value, or with a new one.

9.5.3 – Using parameters E6(n)002 and E6(n)003

In normal operation, the system checks all axes demands against software limits. These limits, (contained in P17), are pre-set by the m/c builder, and define the maximum working envelope.

To prevent collisions, these limits may be temporarily modified to suit the conditions of a particular workpiece to be machined. To do this, the E6(n)002 and E6(n)003 parameters are used to re-define the limits, but always within those defined by P17.

When the system is switched on, or when there is RESET or M02, these parameters are initialized to a value of +/– 90000000, (depending on whether the parameter defines maximum or minimum travel).



To modify the limits, the programmer should write :

E60002 = Xb
E60003 = Xa
E62002 = Za

Whilst machining, the system takes account of the tool compensations so as not to exceed the defined limits. Thus in maximum X travel, the axis cannot exceed Xa – R.

9.5.4 – Programming using type 9 parameters and axis equivalence

Automatic machine origin setting on four axes : X (axis 0), Y (axis 1), Z (axis 2) and B (axis 7, modulo 360°, MOS zero).

%7001	
G79 N100	
N10	@X : axis name, L0 : axis number
L6 = 91100+L0	MOS recognised
L1 = 90000+L0	Measurement value
L2 = 91000+L0	Servo–controlled/not servo–controlled
L3 = 92000+L0	Datum switch enabling
L4 = 93000+L0	Datum switch status
EL6 = 0	MOS done on an axis
EL2=0 EL1=–1000 EL2=1 L5=EL1/1000	Initialize measurement to –1. Or, if rotary axis, L5 will be 359°
G79 L5 > 0 N18	Jump to N18 if axis is rotary
(linear axis)	
N11 G79 EL4=0 N12	Jump to N12 if not on the datum switch
G52 G0 L5=L5–1 @XL5 G79 N11	Else move off switch in 1 mm steps, depends on MOS direction
N12 EL2=0 EL1=–50000000 EL3=1 EL2=1	Initialize measurement at 50 m, the sign depending on the direction of the MOS
G52 G1 G10 @X0 @L0 > 0 N50	Move towards measurement origin till switch is detected.
(Rotary axis)	
N18 G79 EL4 + 0 N20	Jump if axis is not on the datum switch
L5 = L5–1 G G52 @X–L5 G79 N18	1° else clear the switch, sign of move depends on MOS direction
N20 EL2 = 0 EL3 = 1 EL1 = 100000 EL2 = 1	Initialize measurement
G52 G1 G93 G10 F.1 L5 = EL1/1000 + 180	Move 180° in V/L
@XL5 @L0 < 50 N50	relative to measurement origin until value > 0°
G79 N20	If datum switch not met, another 180° movement
N50 G94 F1000	Program returned to mm/min after a MOS on the rotary axis
N100 @X=X L0=0 G77 N10 N50 @X=Y L0=1 G77 N10 N50 @X=Z L0=2 G77 N10 N50 @X=B L0=3 G77 N10 N50	Definition of the axes equivalences

9.6 - NEW FUNCTIONS IN PARAMETRIC PROGRAMMING

These new functions require NC software at index D or above.

9.6.1 - Reading current block modal functions

The symbols for these functions preceded by character "." (dot) are declared between square brackets. These functions are accessible by parametric programming, but for read only.

- Boolean functions : functions G and M

  The symbol for these functions is their number preceded by the character "B".

  Example : [.BG01] ; [.BG70] ; [.BM05]

  . List of G functions :

  [.BG00] [.BG01] [.BG02] [.BG03] [.BG17] [.BG18] [.BG19] [.BG23]
  [.BG90] [.BG91] [.BG40] [.BG41] [.BG42]
  [.BG93] [.BG94] [.BG95] [.BG96] [.BG97]
  [.BG80] [.BG81] [.BG82] [.BG83] [.BG84] [.BG85] [.BG86]
  [.BG87] [.BG88] [.BG89] [.BG70] [.BG71] [.BG75]

  . List of M functions (decoded) :

  [.BM03] [.BM04] [.BM05] [.BM07] [.BM08] [.BM09] [.BM10] [.BM11]
  [.BM19] [.BM48] [.BM49] [.BM997] [.BM998] [.BM999]
  [.BM40] [.BM41] [.BM42]

- Auxiliary Functions with Encoded Values :

  . [.RF] feedrate - Format 5.2 in mm/min

  . [.RT] tool No. - Format 5.0

  . [.RS] spindle speed

  Examples (see Chapter 13 for the type of programming)

- Programming in metric units, then in inches, then return to metric units.

```
% 3
N1 G0 X Y <--         positioning in metric units if the system is initialized in metric units (G71).
VAR [INC MIC]
ENDV
[INC MIC] = 71* [.BG71] + [INC MIC]
N10 G70 X12 Y32 Z0  }
'                   }  Programming in inches
'                   }
G [INC MIC] X5 Y2   }
'                   }
'                   }  Programming in metric units
M2                  }
```

- Restoring the current plane after a change of plane

```
% 4
.
.
G18
.
.
.
L0 = [.BG17] * 17  L0 = [.BG18]  * 18 + L0
 [G PLAN] = [.BG19] * 19 + L0
G17
.
.
G [GPLAN] <--- return to plane G18
```

- Spindle orientating control sub-program.

```
% 8000
L0 = [.BM19]
G79 L0 = 1 N2         Jump if already orientated
L1 = [.BM05]
G79 L1 = 0 N1         Jump if spindle rotating
S100 M3               Start spindle rotation
N1 M19                Orientation
N2 ...                Start of the operation requiring spindle orientation.
```

9.6.2 – Acquisition of a value entered by the operator from the alphanumeric keyboard

The value entered by the operator is one of the terms of the right-hand member of a parametric equation.

- This value is symbolized in the equation by the character $.

  Example : L0 = L1 + $

- The value entered by the operator can be a number with a maximum of eight digits plus sign and decimal point. This value must be confirmed by pressing LF.

  When the system reads the equation, it stops to allow the operator to enter the value before continuing.

- When the first character entered by the operator is alphabetic, the system encodes it as a value between 1 and 26 (A = 1, B = 2, ..., Z = 26) and confirmation of this value is immediate.
- The characters entered are echoed on the CURRENT, CURRENT POINT and DYNAMIC TRACE pages like messages programmed after $0 or in addition to such a message if already programmed.
- Aside from the LF key, the only active function key is the <=== (backspace) key used to delete characters entered before confirmation.

9.6.3 – Display of a value after a programmed message

When the symbol $ is the operand of a parametric expression, the value resulting from this expression can be displayed after a previous message programmed with $0.

It is the presence of the character = after the symbol $ which causes display of the result.

Example : FOR L0 = 1 TO 20 DO $ PHASE NO : (LF) (this type of programming is explained in Chapter 12)

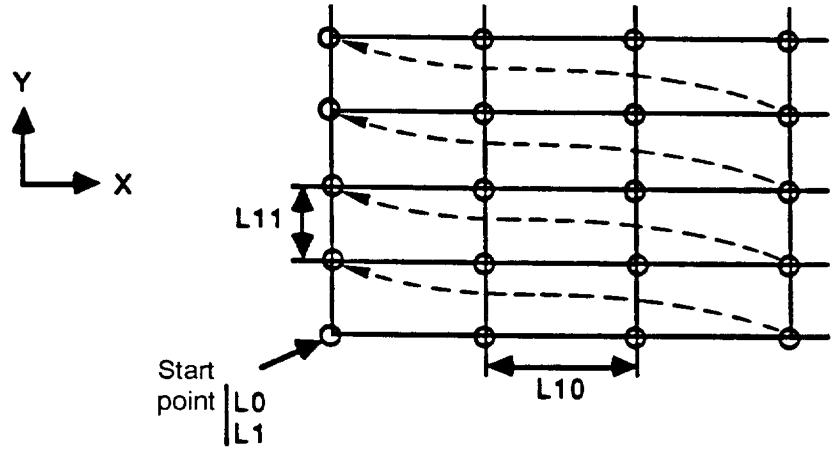
```
$ = L0
        .
        . processing
        .
ENDF
```

When the system reads the sequence $ = L0, the message PHASE NO : (value of L0) is displayed.

CAUTION :

The function $ = ... located in a block after function G79 may not be used as a jump condition.

Example : drilling to a pattern
(see Chapter 12, Structured Programming, for writing the program).



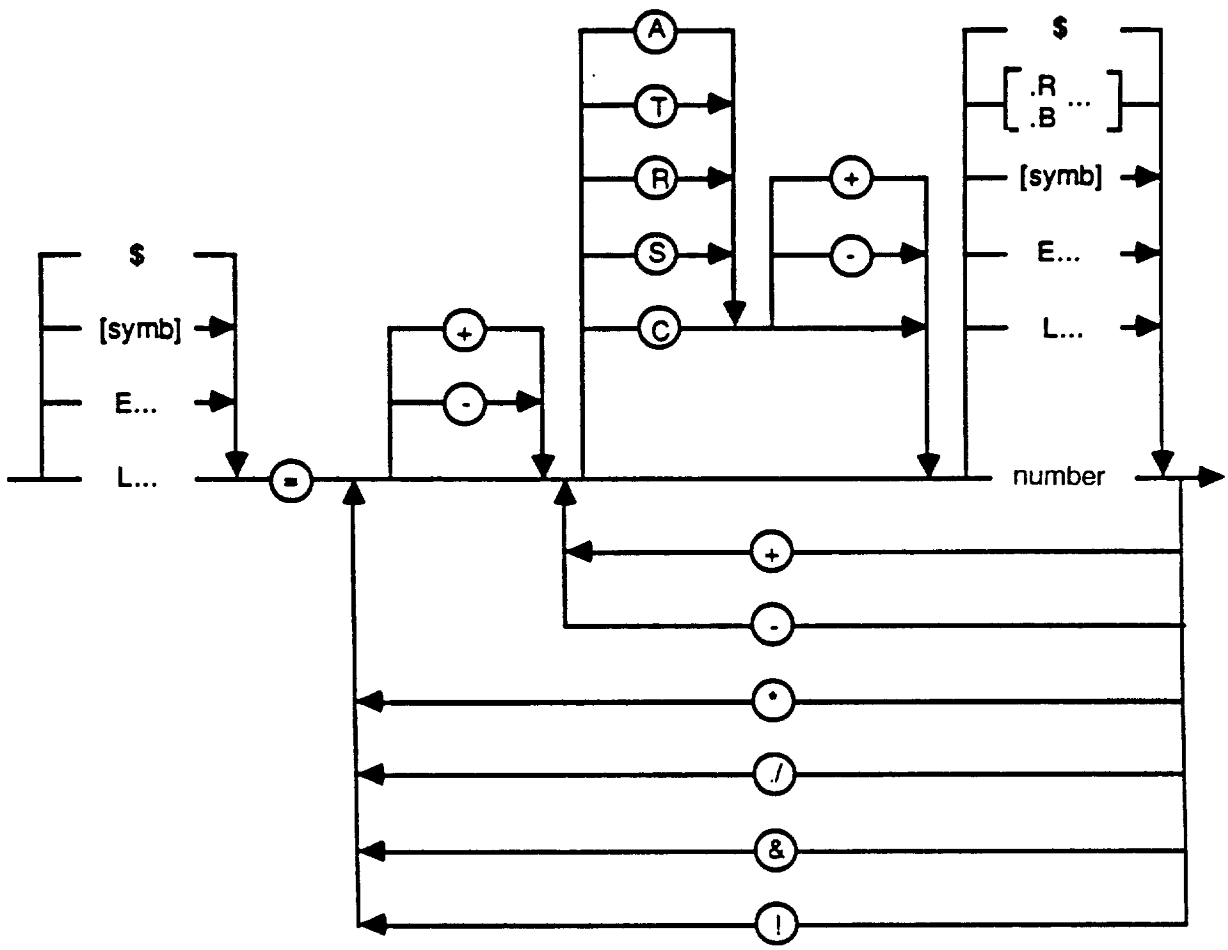
L2 = Number of points along X

L3 = Number of points along Y

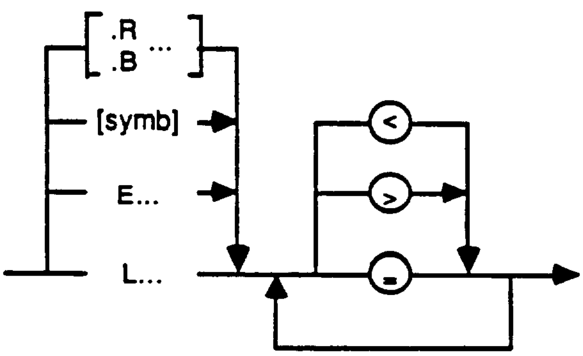
```
% 20
$ X start :
L0=$
$ Y start :
L1=$
$ delta X :
L10=$
$ delta Y :
L11=$ L4=0
WHILE L4 <> 15 DO
  REPEAT $ number of points on X :
     L2=$
  UNTIL L2>0
  REPEAT $ number of points on Y :
     L3=$
  UNTIL L3>0
  REPEAT $ pattern OK (Y/N) ? :
    $=L2 $=-L3 L4=$ $ =-L3   "-" sign used as separator between the number of points on X and
                             the number of points on Y
  UNTIL L4 = 14 OR L4 = 25   Wait for answer N or Y.
        If answer N (14)     return to REPEAT ù number of points on X :
        If answer Y (25)     continue program.
ENDW
  $     OK to clear display of pattern message (Y/N) ? :
        value L2-L3
FOR L5=1 TO L3 DO XL0 L6=L5-1*L11+L1YL6
  FOR L4=1 TO L2 DO $ point on pattern :
    $=L4 $=-L5 L6=L4-1*L10+L0 XL6   $=-L5 "-" sign used as separator between
    Z=10                            the number of points drilled on X
    Z                               along dimension Y.
  ENDF
ENDF                                L4 number of points drilled on X along
M2                                  the column in Y given by L5.
```

9.6.4 – New syntax graph of parametric programming

- Loading of a parametric expression :



- Comparison for conditional branch

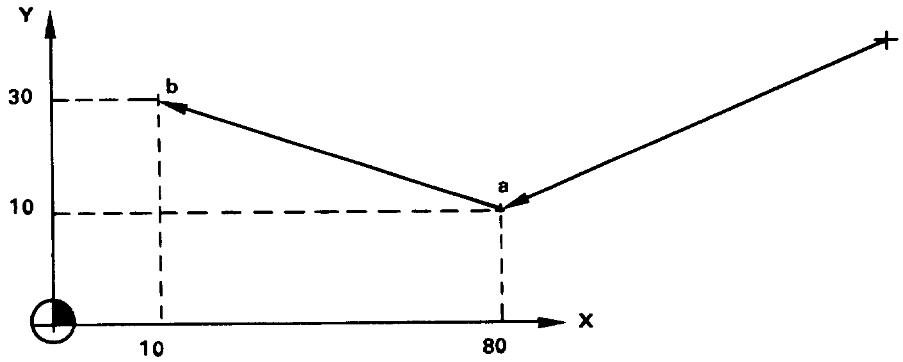


(Second part of the expression same as that loaded)

NOTES

10 - FEED RATE - DWELL - SPINDLE SPEED

10.1 - FEED PROGRAMMING



10.1.1 - Programming in mm/min

G94 Preparatory code which defines feedrates in mm/min.

F052 Feed value (min. value 0.01 mm)

Both G94 and the value of F are modal.

G0		X80	Y10	
G1	G94	X10	Y30	F500

Path ab is executed at 500 mm/min.

- Programming in mm/min is associated with the preparatory functions G1, G2, G3 and G23.
- Function G0 is modal and suspends the effect of F. Movements are executed at rapid.

The feed rate may be programmed in mm/min. when the movements are executed on the main axes X, Y and Z.

Programming a rotary axis in mm/min. or in degrees/min

A rotary axis may be programmed with G94, in either mm/min. or degrees/min. as follows :

- eg1 Rotary axis moves, programmed alone :

  G91 G1 G94 B30 F40 ; the feedrate of the B axis is expressed in °/min. (ie 40°/min.) : thus, time of execution = 30/40 minutes, or 45 seconds.

- eg2 Rotary axis moves programmed with a linear axis :

  G91 G1 G94 X10 B30 F100 : the X axis feedrate is expressed in mm/min. ; the B axis move is completed in the time taken to execute the linear path of X.

  t = X/F = 10/100 = 1/10 min. ie. 6 seconds

The speed of the B axis is thus 30°/6sec = 5°sec.

Similarly, if two linear axes X and Y are programmed with B, the time t will be calculated on the vector move :

$$\frac{\sqrt{\Delta X^2 + \Delta Y^2}}{F}$$

NOTE :

In radius compensation mode (G41–G42), the programmed feed rate (F) is applied to the path of the tool centre (Fig. 1).

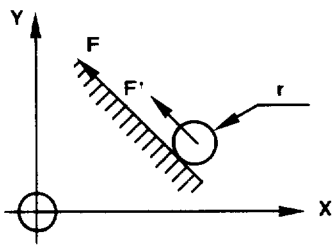


FIG. 1

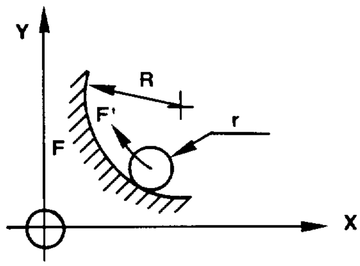


FIG. 2

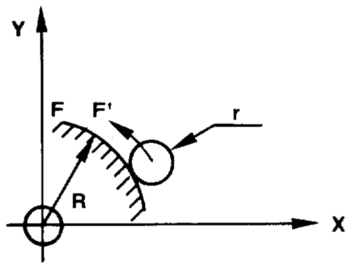


FIG. 3

In circular interpolation mode, the tool's feedrate at the point of contact may be higher or lower depending on whether the path is concave or convex.
To maintain the feed rate over the workpiece profile, a feed rate F' should be programmed with the value :

$$F' = \frac{F \text{ required at the point of contact } x\ (R \pm r)}{R}$$

Figure 2 : concave profile F' = F(R – r)/R

Figure 3 : convex profile F' = F(R + r)/R

This modification is justified only if the result of the operation 1 ± r/R differs significantly from 1.

10.1.2 - Programming in V/L = 1/time (in mins)

G93	Feed rate in inverse time.

F 4.2 The feedrate value is calculated by dividing the required velocity, (V) in mm/min by the length of the move, (L) in mm.

The format of the "feed rate number" F is : F4.2.
The theoretical limit values are :

F = 0.01	Machining time 100 min.
F = 9999.99	Machining time 0.006 sec.

In practise however, the maximum value of F will be limited in accordance with the time required to prepare a block, as defined by machine parameter P51.

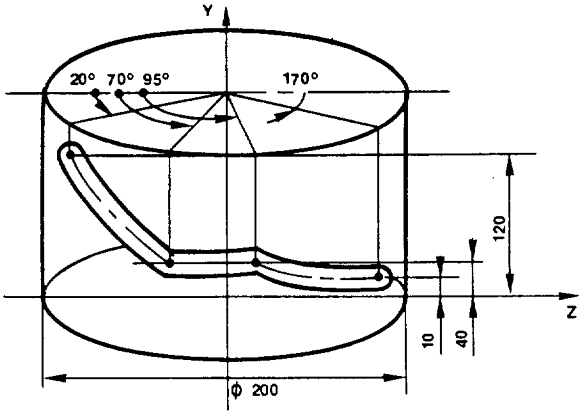
The value of F is, 1/duration in min.

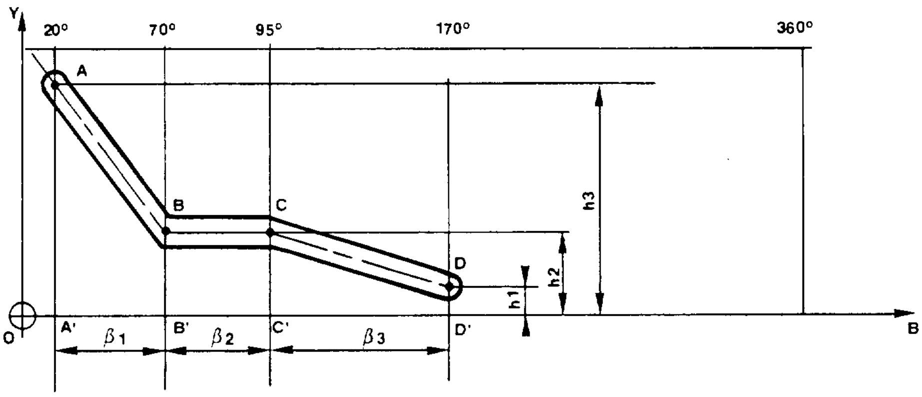
NOTE :

Programming in V/L is not allowed in circular or helical interpolation. There is no error message !

EXAMPLE :

5 mm deep groove on a 200 mm dia. cylinder, at a feedrate of 150 mm/min.





	Length of planned arc β in degrees	Length of real arc L	Feedrate in V/L
	$\beta = \sqrt{\dfrac{\pi \times D \times \alpha}{360}}$	$L = \sqrt{\beta^2 + h^2}$	F = V/L
β1,L1	87.266	118.386	1.26
β2,L2	43.633	43.633	3.43
β3,L3	130.899	134.293	1.12

N10	G0	X100	Y200	Z150	B					
N20						T1	D1	M6		
N30						S900		M41	M3	M8
N40		X	Y120	Z102	B20					
N50	G1			Z95		F150				
N60	G93		.Y40		B70	F1.26				
N70					B95	F3.43				
N80			Y10		B170	F1.12				
N90	G			Z102			M5	M9		
N100		X100	Y200	Z150	B	D0			M2	

10.1.3 – Deceleration function G9

The axes servo following error, ϵp, is directly proportional to the programmed feedrate. Its "SMOOTHING" effect (at a constant feedrate), increases as the angle α becomes more acute.

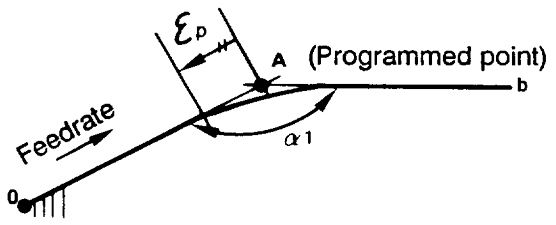


FIG. 1

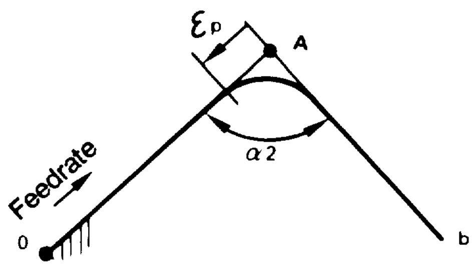


FIG. 2

Bold line :	The real path without G9, skids past "a"
Fine line :	The real path with G9, actually goes through "a".

- G9 is used to absorb the following error ϵp at the end of a given movement before continuing to the next block. In doing this the axes drop down to zero speed.

- The programmed path is followed exactly.

The G9 function allows the next block movements to continue, provided that the current block dimensions have reached an "in position" window, (as pre-defined by the machine builder in parameter P22).

N1	G1	X0	Y0	(Fig.1)
N2	G9	Xa	Ya	
N3		Xb	Yb	

The slide will be decelerated on "0A" at a distance ϵp from "a" and will go through point "a".

N1	G1	X0	Y0	(Fig.2)
N2		Xa	Ya	
N3		Xb	Yb	

The slide will not be decelerated and will not pass through point "a". The curve described will be proportional to the feed rates along 0a and ab, and the angle of these vectors.

10.1.4 – Limit rates

The limit feedrates depend on the machine and are set up when the system is switched on. In linear mode, the minimum time for execution of a block is 0.08 seconds, irrespective of the rate programmed. In circular interpolation mode, the maximum tangential feed rate is proportional to the radius of the circle as described by the centre of the cutter. It is expressed by the formula : Maximum feedrate (in mm/min) = 300 * Radius (in mm). This is derived from the system imposed limit, of the maximum feedrate being 1/20 of the Radius per hardware update, (10 ms).

For example : a 3 mm radius circle will be described at a MAXIMUM feedrate of 900 mm/min, irrespective of any faster programmed feedrate.

10.1.5 - Overspeed : Function G12

If the machine-tool is fitted with a handwheel, the latter allows movements obtained by rotating the handwheel to be superimposed on the normal demand as calculated by the system. This can be of use to reduce "air time" during the approach to an un-machined surface.

Movement acceleration applied in this manner is effective over the programmed path (paraxial, linear or circular).

The G12 function enables overspeed in the block concerned. This function is non-modal. The system continues with the following block when the programmed dimension is reached.

The value of the handwheel increment is defined in the 3rd and 4rth words of machine parameter P13.

10.1.6 - PROGRAMMING OF TANGENTIAL FEED : FUNCTION G92R

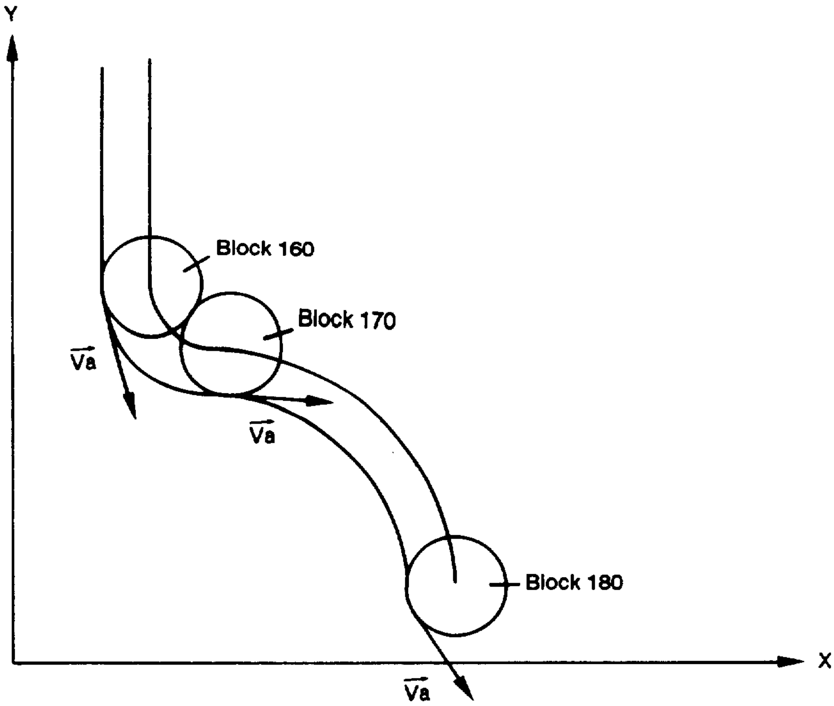
This function is used to apply the feed rate F.. to the programmed curve instead of to the tool center path.

It is used for tool radius compensation (G41, G42) and for programmed curves whose radius is higher than the value of the tool radius defined by R.

Function G92 R0 cancels programmed tangential feed.

EXAMPLE :

%	
N100 T1 D1 M6	Compensation D1, R = 10
N110 S800 M40 M3	
N120 G0 G41 X40 Y100	
N130 Z-10	
N140 G1 Y60 F200	
N150 G92 R10	(Tangential feed rate programmed for curves whose radius exceeds tool radius R10)
N160 G3 X60 Y40 R20	
N170 G2 X90 Y10 I60 J10	(Curves whose radius R is higher than the value of tool radius R
N180 G3 X100 Y R10	specified in block N150 : tangential feed is enabled)
N190 G92 R0	(Programmed tangential feed cancelled).
.	
.	
.	
.	
M2	



NOTES :

- *When a connection circle is created automatically, the tool center feed on the circle remains the same as that of the previous block.*
- *Programming block G92 R.. must not include any movements.*
- *Tangential feed is cancelled by a reset.*

10.1.7 - Programmed acceleration override : Function EG

This function is used to override the maximum acceleration tolerated on programmed movements by 1 to 100 percent. The maximum acceleration depends on the tool path and the accelerations assigned to the axes by machine parameters.

This function is modal and is forced to 100 by M02 or a reset or is modified by entry of a new value.

At power on, it is forced to 100.

- Programming : It is programmed by function EG followed by an absolute integer value between 1 and 100.

  The default value is 100.

  . if EG0 is programmed, the system converts it to EG1,
  . if EG200 is programmed, the system converts it to EG100.

  EG1 to EG99 are displayed on the current block page.
  EG100 is not displayed.

- Interpolation : in programmed movements, the acceleration override is always used by the interpolators except after action on FEED STOP or trip of the feed safety device.

  In JOG, MOS and RECALL modes, acceleration override is inoperative.

10.1.8 - Special feedrate on a blend or chamfer

A feedrate different from the machining feedrate can be programmed with a blend or chamfer programmed by EB+ or EB-.

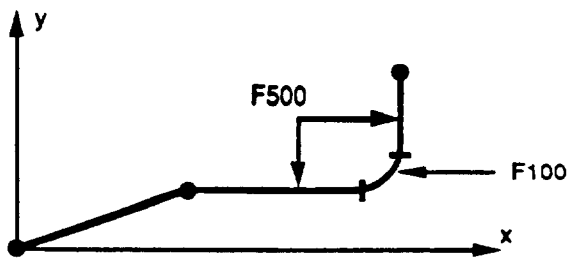
The feedrate is defined by address EF which is not modal.

It must be programmed after function EB. Otherwise error 86 is displayed.

EXAMPLE :

N1 G0 X0Y0
N2 G1 X350 Y50 F500
N3 X500 EB10 EF100
N4 Y100

The blend is carried out at a feedrate of 100.



10.2 – PROGRAM DWELLS

G4	F022
↓	↓
Dwell function	Value of dwell 1/100th sec. to 99.99 sec.

Function G4 and the dwell value are not modal.

G4 – F022 programmed in a block containing movements is active at the end of the block. The presence of F022 does not cancel a previously programmed feedrate.

If G4 is followed by two values of F, only the last one programmed is recognised.

EXAMPLE :

N10		X1	Y1	F500	500 mm/min feed
N20	G4			F1.5	1.5 sec. dwell
N30		X2	Y2		500 mm/min feed
N40	G0	X3	Y3		Rapid feed

10.3 – PROGRAMMING SPINDLE SPEED

The maximum programmable speed of rotation for an analog spindle is 32500 revs/min.

The value of S is modal.

If appropriate, three speed ranges may be defined, (M40/41/42), in association with the S address. If left unspecified, an automatic range selection will be made simply by programming S.

10.4 – SPINDLE ORIENTATION

The spindle can be orientated by programming M19. It is orientated in the position programmed by address EC.

11 - MISCELLANEOUS FUNCTIONS

11.1 - OPTIONAL BLOCK "/"

Blocks preceded by a slash code, (eg. /N...) can be omitted at the operators' discretion, by pressing the "optional block" key.

The key-press operation is only effective in an M00 or M01 or M02 status.

11.2 - PREMATURE BLOCK TERMINATION

11.2.1 - Premature block termination, via an external interrupt

If G10 is programmed in a movement block, the system is primed to expect an interrupt, which, if it occurs, will prematurely terminate that move and save, (for later analysis), ALL the axis positions at the moment of the interruption. Such interrupts are usually the result of a probe input. If the expected interruption occurs, a jump may be made to a designated block.

The G10 function is only allowed in G40 mode. Axis movements can be executed in G0, G1, G2 or G3. Dwells, in G4, can be interrupted.

eg. G10 X 12345 Nxx. If Nxx is omitted of if the interrupt fails to occur, the following block is executed.

11.2.2 - Premature block termination, via detection of a measured threshold

As well as probe type interrupts, the system can be made to generate its own interrupt by comparing the measurement of an axis to a programmed threshold.

The block syntax is :

$$\text{G10 X .....}\quad \underbrace{\text{@xx}}_{\text{measurement address}}\quad \underbrace{\left\{\begin{matrix}>\\<\end{matrix}\right\}}_{\text{comparison symbol}}\quad \underbrace{\text{xxx.xxx}}_{\text{programmed threshold}}\quad \text{(Nxx)}$$

- The measurement address in the axis number connected to the measurement sensor, (P9). If X is to be measured and the axis number of X is 0, @00 will be programmed. All axes of the system can be used, whether they are measuring only or fully servo-controlled.

- Comparison symbol : defines the interrupt condition.

  $\geq$ greater than or equal to
  $\leq$ less than or equal to

  The designated axis is compared to the programmed threshold every 10 ms. When the condition is true, the block is interrupted.

- Programmed threshold : a signed value with a decimal point expressed in mm or in inches.

- Example :
  G10 X ... @ 2 < 50.32 N100

  Jump to sequence 100 when the measurement of axis 2 is less than or equal to 50.32 mm.

11.3 – SUB-PROGRAMS CALLS AND PROGRAM BRANCHING

11.3.1 – Sub-program calls with return : Function G77

The call is unconditional.

The sub-program (or machining macro) can be defined in the main program. Each sub-program can be repeated up to 9999 times (S4), and upto eight levels of sub-program nestings are possible.

An INTERNAL sub-program is defined within the main program, and is identified by its start and end block numbers. It can be situated between the % code and the M02 or even beyond the M02, but before the XOFF character. Any series of blocks in a program may be defined as a sub-program.
An EXTERNAL sub-program is identified by the letter address H, followed by its program number, followed by the start and end block numbers.

EXAMPLE :

Main program

```
% 123
N10
N20
.
.
N50 G77 H252
N60
.
.
N100
N110
.
.
N180 G77 N20 N100 S3
N190
.
N210
.
N260
.
N300 G77 N500 N550
N310
.
.
N390 M02
N500
.
.
N550

X OFF
```

Sub-program 1

```
% 252
N10
N20
.
.
N80 G77 H5263 S3
N90
.
.
N160
X OFF
```

Sub-program 2

```
% 5263
N600
N610
.
.
N650
N660
.
.
N700 G77 N610 N650
N710
.
.
N920 G77 H123 N210 N260
N930
.
.
.
.
N1290
X OFF
```

- Sub-program 5263 and N20/N100 from program 123 are run 3 times (S3).
- Sub-programs 1 and 2 must not include function M02.

A sub-program defined by two sequence numbers can only be executed in the order of execution of the program.

EXAMPLE :

```
% 1
N10
N20
.
.
.
N200
.
.
.
N350 G77 N10 N200
.
.
.
```

```
% 1
N10
N20
.
.
.
N200
.
.
.
N350 G77 N200 N10
.
.
.
```

These two formats are identical.

NOTE :

If a sub-program in the workpiece program area of RAM memory has the same number as a resident sub-program, (in non-volatile EEPROM), the RAM version will take priority and always be accessed, effectively shielding the resident sub-program.

11.3.2 – Block jump without return : Function G79

A block jump will go directly to a designated block. The block called must be in the same program.

- Unconditional jump

  eg. G79 N280 : jump to block N280, which can be before or after the current block.
  Caution : beware of creating an endless loop with backward jumps.

- Conditional jump

  Use of a program variable or a program parameter in the jump block, and of the condition >or = or<, or two simultaneous conditions.

  eg. G79 L2>=3 N280. If L2 is greater than or equal to 3, jump to block N280, otherwise continue to the following block.

  (See paragraph 9.4.3 for more details of possible tests).

11.4 – EMERGENCY WITHDRAWAL : FUNCTION G75

Function G75 must be followed by a sequence number. This sequence number defines the beginning of the withdrawal program. The withdrawal program ends with a programmed stop M00 or an end of program M02.

A withdrawal program definition is operative from the time it is defined until it is superceded by a new definition from another G75.

An emergency withdrawal demand, is transmitted by the PLC, via the input signal DEBURG. When the status of this signal goes to 1 and the system is in CONT, SINGLE BLOCK, MDI or RAPID mode, the block in progress is interrupted, in mid-block, and the system jumps to the last declared withdrawal program.

Whatever the mode in use, the withdrawal program runs in CONTINUOUS mode.

If a withdrawal program terminates at a programmed stop M00, this M00 must be followed by an operator restart.

EXAMPLE :

N10... N20 N30 N40 G75 N1000	
N50 N60 " " N120	If a withdrawal is called in this field, jump to N1000
N130 G75 N2000	
N140 N150 " " " N290	If a withdrawal is called in this field, jump to N2000
N300 M02	
N1000 N1010 " " " N1050 M02	Withdrawal program for blocks 50 to 120, then," end of program"
N2000 N2010 " " N2080 M00	Withdrawal program for blocks 140 to 290, then, "programmed stop". The operator may choose to reset or to continue the cycle.
N2090 N2100 " " N2120 G79 N130 X OFF	Cycle restart sequence and return to block 130.

NOTE :

- *If an emergency withdrawal is demanded when no withdrawal program has been defined, the signal has the same effect as the FEED STOP button.*

- *If G75 N0 is programmed, an emergency withdrawal requested in the following blocks will result in a FEED STOP.*

11.5 - THE SUB-PROGRAM CALL FROM THE PLC

The PLC can call a specific sub-program with the number %9999.

To call this sub-program, the PLC sets the APP SSP data in the decoded exchange data to the NC.
If %9999 has not been created, Error 25 will be reported.

11.5.1 - Pre-conditions for call acknowledgenert

The system must be in CONT, SEQ or RAPID mode.

When a workpiece program is being run, the call cannot be executed until the END of an interruptible block. A non interruptible block is a block internally created by the system during a fixed machining cycle (eg. G31, G45...), or a block which uses values in the following blocks for its processing. (eg. in PGP).

While the sub-program is being run, PLC demands for that same sub-program (ie. the state of APP SSP), will be ignored.

As no direct acknowledgement is sent by the system to the PLC, it is usually that the sub-program itself send an entry acknowledgement to the PLC via an M function.

11.5.2 - Sub-program branching

Modal data in the last block executed prior to the %9999 call, is saved

Functions G01 and G40 are forced, on entry into %9999.

11.5.3 - On return from the sub-program to the main program

The modal state of G40, G41 or G42, which was active before the call, is restored, as are the modal functions (M.. and S..) and the program variables (L0 to L19). The compensator number (D...) is restored only if the tool has not been changed. However, the variables L100 to L199 are not restored.

The position of axes before the %9999 sub-program call will be re-established on exit. The type of axis move, (G0 or G1), to re-establish those positions must be defined in the sub-program.

If the system was in G41 or G42 status before the %9999 call, repositioning will be made normal to the next block of the workpiece program.

All modal G codes are restored and the workpiece program continues.

NOTE :

If there was no workpiece program in progress when %9999 was called, the cycle ends on completion of the sub-program.

11.5.4 – Structure of the sub-program

It is normal for %9999 to execute a single process, but if multiple functions are to be processed in a pre-defined order, it will be necessary to specify which function is required. This can be done using a program parameter, E40000 for example.

- 1st method :

  Each function is written in a different sub-program (%a, %b, %c, etc.). In this case, sub-program %9999 can consist of a single block which acts simply to re-direct the actual execution.

  EXAMPLE :
  ```
  %9999
  G77 H E40000 Mxx
  X OFF
  ```

  E40000 contains the number of the sub-program which is to be called (a, b, c, etc.). The Mxx function acts as an acknowledgement to the PLC.

  The disadvantage of this method is that it creates an additional level of sub-program nesting.

- 2nd method :

  All the functions are written in sub-program %9999, whose first block is a jump to a sequence number contained in parameter E40000.

  EXAMPLE :
  ```
                  %9999
  G79 N E40000    Mxx
  Na E40000=b     processing 1st function

  G79 Nz
  Nb E80000=c     processing 2nd function

  G79 Nz
  NcE8000=a       processing 3rd function

  Nz              End of sub-program .
  X OFF
  ```

NOTE :

The sub-program normally ends with X OFF, unless it is required to end the cycle in progress at the end of the %9999 sub-program, in which case an M02 is programmed.

11.6 – ANGULAR OFFSET : ED

The geometry of a complex workpiece may be such that it can be obtained from a simple geometric figure via angular rotation (eg. drilling on a PCD, toothed profiles, etc.). If the angular offset is used in such cases, only the basic figure required for machining need be programmed.

11.6.1 – Programming

Address : ED
Format : +033 (degrees and 1/1000th of a degree)

The value is modal. It is initialized to zero by a reset or by M02.

This function may be programmed either in incremental mode (G91) or in absolute mode (G90).

It may be parameterized : eg. if L3 = 30°, EDL3 is equal to ED30.

The angular offset can be applied in all planes, G17, G18 or G19.

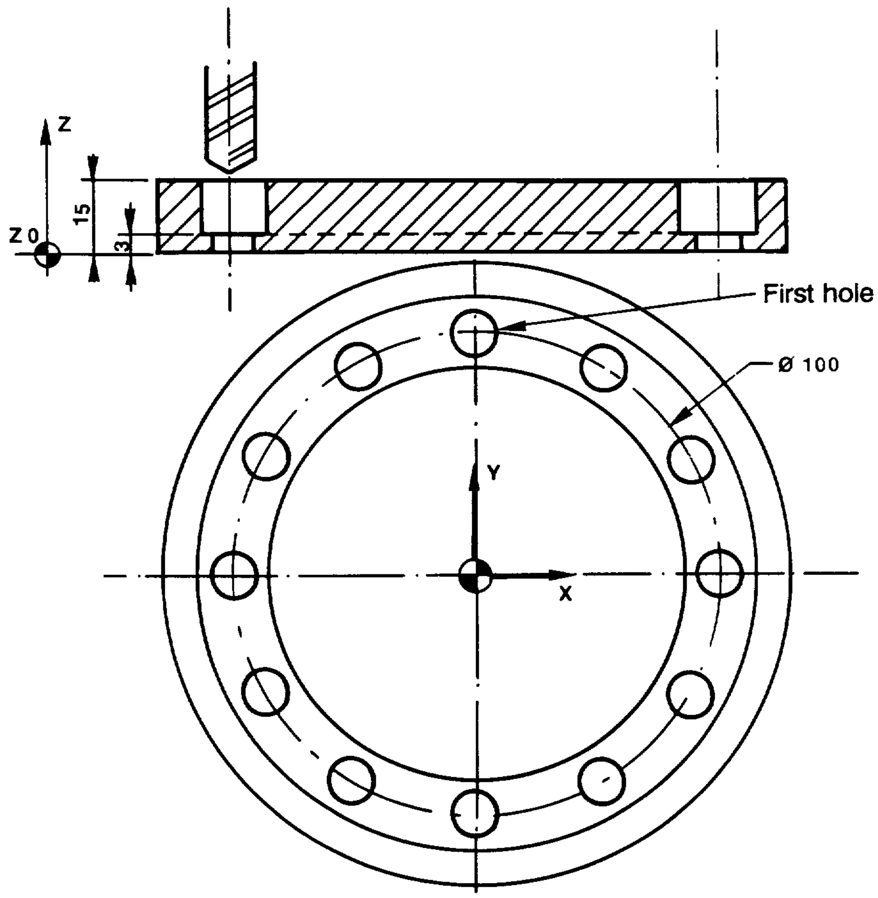
11.6.2 – Special features

- Tool radius compensations and geometric profile programming are modified by the angular offset.
- The ED function must not be used between two blocks which have not been completely defined (GPP).
- The G10 premature block termination may cause the system to jump to the following block or to another block. Either will be modified by an active angular offset.
- If an angular offset has been programmed in one plane, it will NOT be cancelled if the system switches to another plane. The dimensions of the new plane will be modified.

11.6.3 - Examples

11.6.3.1 - Simple machining cycle on a PCD

Drill 12 holes, with circular positioning between each hole.

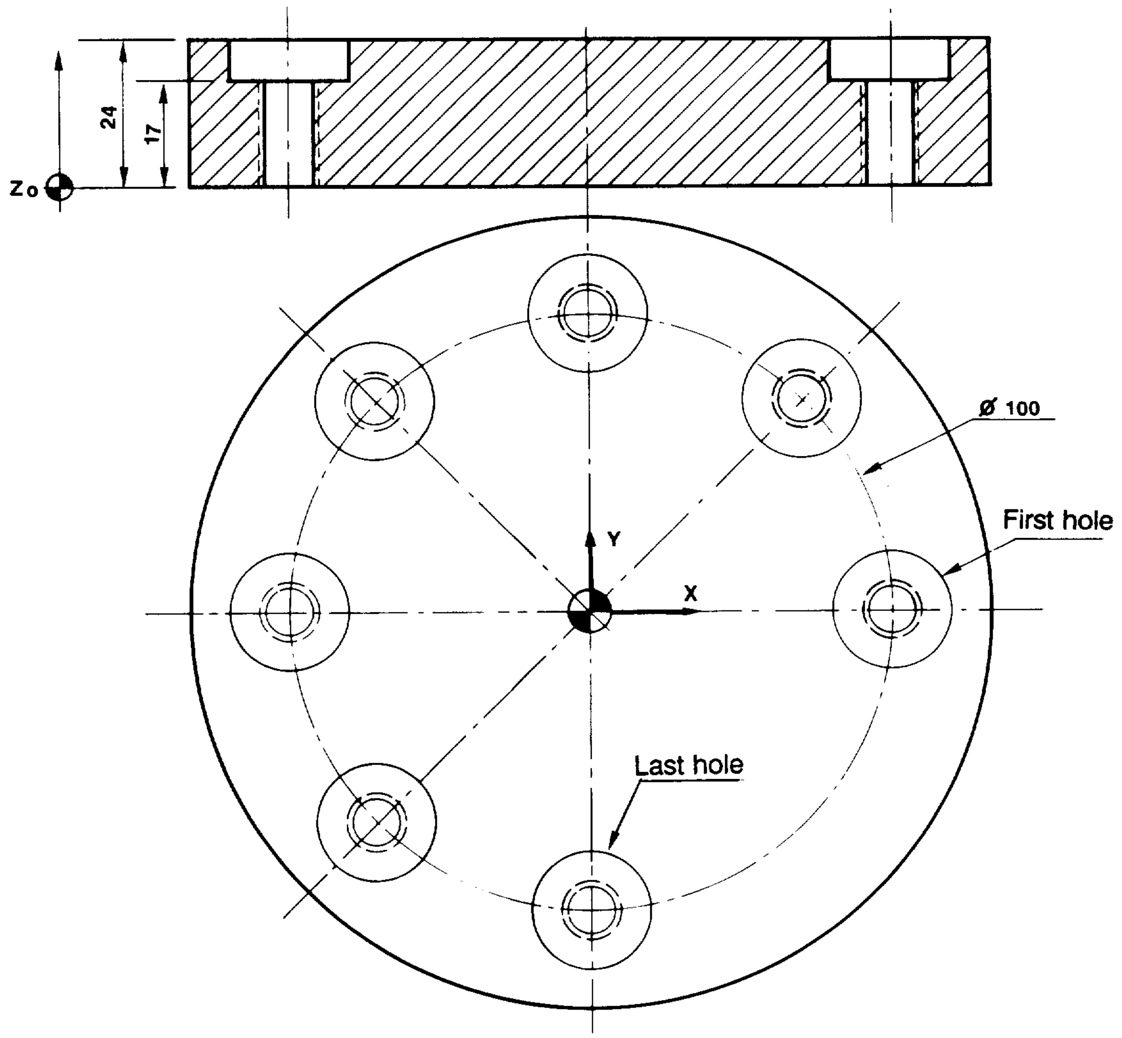



```
%9116
N10 G X100 Y100 Z100
N20(T1) D1(M6)
N30(M41 S1200)M3
N40 X Y50
N50 Z5
N55 G79 N90
N60 G1 G81 Z-5 F300
N70 G80 G91 ED30           } Angular offset and move in circular
N80 G90 G3 X Y50 I J F6000 } interpolation mode to next point.
N90 G77 N60 N80 S12
N95 G Z100
N100 G G52 X-50 Y-50 M2
```

11.6.3.2 – Multiple machining cycles on PCD, with ED programmed in incremental mode

Drilling, spot facing and tapping, 7 holes at 45°.

The movements between holes are linear.



Main program

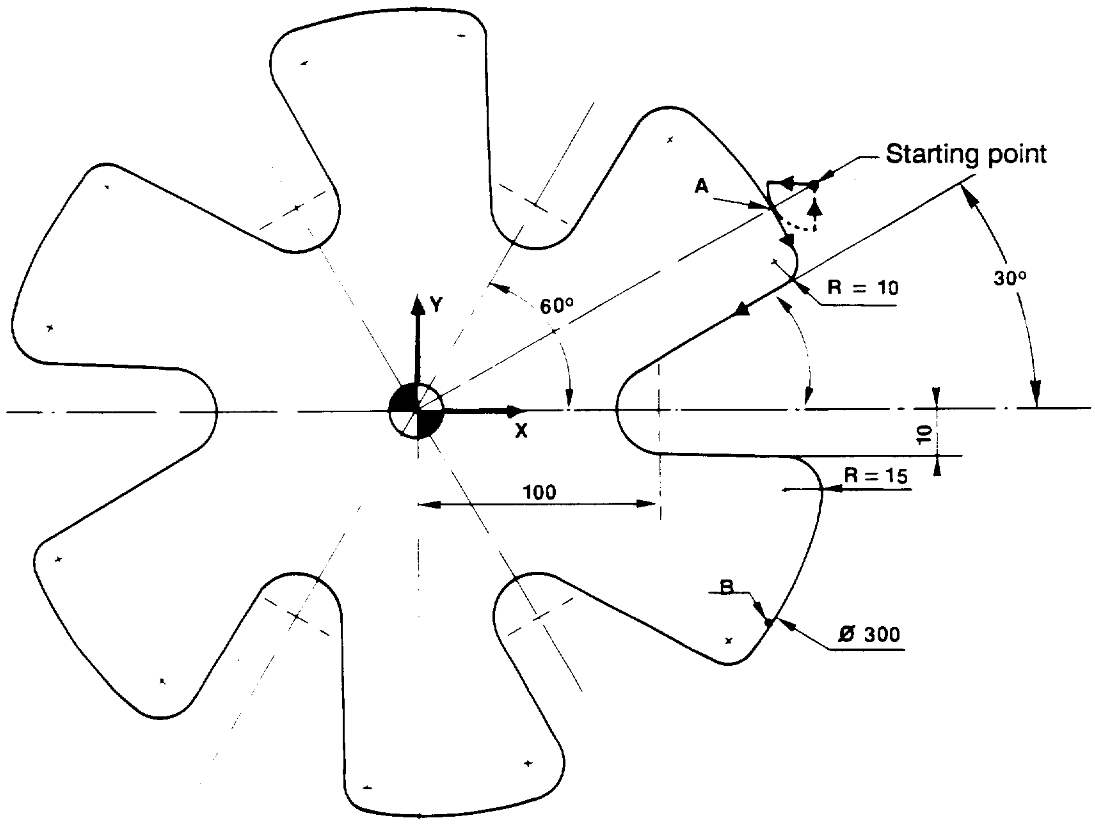
```
%1
N10 G XYZ26 M3
N20 G77 H10 S7
N30 G90 ED0
N40 G77 H20 S7
N50 G90 ED0
N60 G77 H30 S7
N70 G90 XY M2
```

Sub-program

```
%10 Drilling
N10 G90 G81 X50 YZ-8
N20 G80
N30 G91 ED45

%20 Spot facing
N10 G90 G82 X50 YZ17 EF10
N20 G80
N30 G91 ED45

%30 Tapping
N10 G90 G84 X50 YZ-5
N20 G80
N30 G91 ED45
```

11.6.3.3 – Rotation of a figure through 360° in steps of 60°

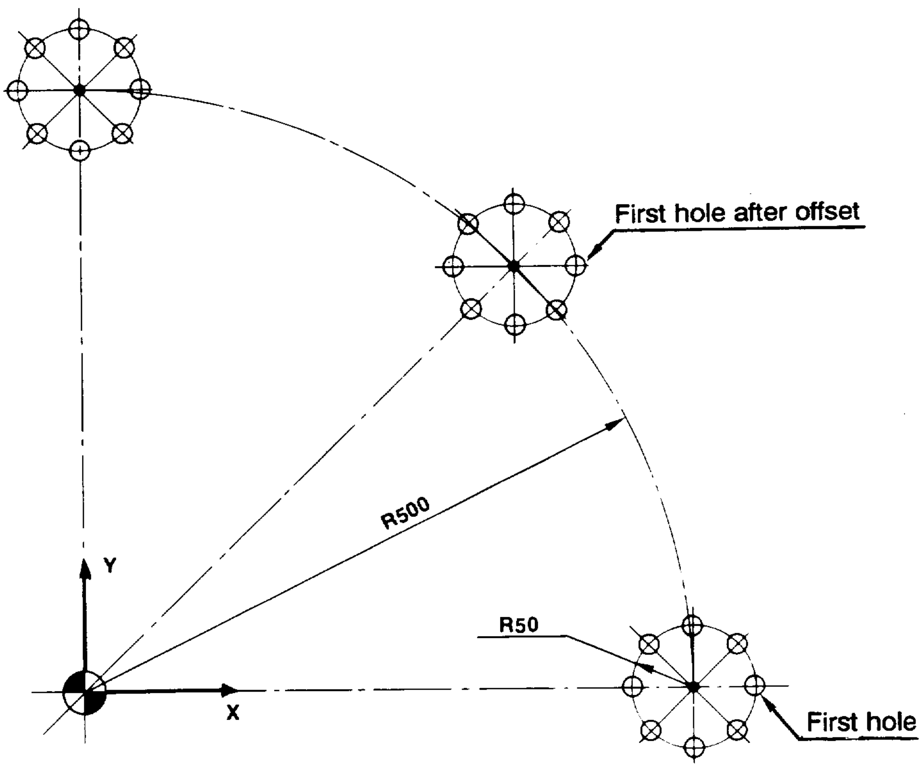
Machining a toothed wheel




```
%9121
M3 M41 S3500
N5 L0=170*C30 L1=170*S30
N10 L4=C30*150 L5=S30*150
N20 L6=C330*150 L7=S330*150
N30 D1 G XL0 YL1                    Start point
N35 Z-10
N40 G1 G41 EA180 ES+ F2000          Cutter entry
N50 G3 IL0 JL1 XL4 YL5
N60 G2 I J ES- EB10
N70 G1 EA210
N80 G3 I100 J R20                   Profile described from A to B
N90 G1 EA ES+ EB15
N100 G2 I J XL6 YL7
(-ANGULAR OFFSET-)
N110 L3=L3-60 EDL3                  Angular offset increment, test for
N120 G79 L3>-330 N60                end at the 6th angular offset
N130 G3 IL0 JL1 R20 ES-             Cutter withdrawal at end of
N140 G G40 EA90 XL0 YL1             machining and return to start point.
N150 G52 Z
M2
```

11.6.3.4 – Double angular offsets of a figure with programmed offset

- Rotation to obtain the figure by progression of ED.
- Rotation of the figure obtained by parameterized programmed origin offset (G59).
- Drilling eight sets of eight holes.



%9117 Main program : N5 G74 G90 G59 X Y N10 G X Y Z100 N20 L1=500*CL3 L2=500*SL3 N30 G59 XL1 YL2 L0=0 N40 G77 H9118 N50 L3=L3+45 N60 G79 L3 < 360 N20 N70 G X Y M2	45° offset of the figure Programmed origin offset
%9118 N5 M3 S1500 N10 G X Y N15 Z2 N20 G1 X50 N30 G81 Z-10 F400 N40 L0=L0+45 G80 EDL0 N50 G79 L0 < 360 N20 N60 G X Y Z100	Sub-program for drilling the 8 holes Angular offset

11.7 – TRANSFERING A MESSAGE FROM THE PROGRAM TO THE SCREEN

Strings of characters, (for example, messages or commands relating to the machining process), which are to be transmitted to the display screen, are included in the program but are not interpreted in any way by the system.

These messages begin with the character, "$0" and end in the LF character. (The zero is optional).

eg. N100 G X Y $0 ...message text... LF

Messages transferred to the screen can be read on the CURRENT MACHINE POSITION or IN PROGRESS display pages.

Only one message at a time may be displayed. A new message erases the previous one. Should a programming error be detected by the system, while a $0 message is being displayed, the error number will be displayed on the second line.

The number of characters in a message should not exceed 39. If the message is longer, only the first 39 characters will be displayed. No error will be reported.

$0 LF or an M02 or a RESET will erase a displayed message.

It is possible to add to an existing message to be displayed.

The syntax is : $ + added message

Example :

$axe X = $= E90000/1000 $+ axe Y = $= E90001/1000	X axis = 123.456 Y axis = 654.321 where E90000 = 123456 and E90001 = 654321

11.8 – PROGRAMMING IN INCHES : FUNCTION G70

This option allows dimensions to be entered directly in inches.

Inch programming is selected with the G70 code. This function is modal and is cancelled by G71 (metric programming).

It is possible to configures the system to default to either imperial or metric at switch on or RESET, by the appropriate setting of bit 3 in machine parameter P7.

- in Inch mode, the dimension and feed rate format is as follows :

X, Y, Z, I, J, K, P, Q, R, ER	+ 044
F in G94	043
F in G95	014

- The format of the other addresses do not change.
- The format of offsets and tool dimensions is as follows :

DAT 1, DAT 2, DAT 3 : + 044

Tool dimensions : L : 034
R : 024
@ : 024

Program variables (L) and program parameters (E) are simply numerical values. The particular use to which they are put, will dictate the units applied.

G70	
L1 = 10	
XLl	(X = 10 inches)
E80000 = 100000	
XE80000	(X = 10 inches)
L10 = E50001/25400	(tool length offset 1. converted to inches).
L2 = 100 + L10	(Internally the values are saved in microns).
XL2	(X = 100 inches + tool length offset 1 in inches).

It is the programmers' responsibility to adapt operations on dimensions and E program parameters, to match the appropriate units, as required.

11.9 – SUB-PROGRAM CALLS, VIA M FUNCTION

A sub-program may be called directly by an M function.

Upto eight different M functions may be pre-defined to call a sub-program. Each M function/sub-program pair is entered, by the builder, in machine parameter P35.

For example, if machine parameter P35 defines that M55 corresponds to sub-program %200, then, Nxx M55 in the workpiece program will act as Nxx G77 H200.

The return conditions are the same as for G77.

CAUTION

Take care with the S address when using sub-programs called via M codes. The results obtained will vary according to the position of S, in relation to the M function.

Example :
Nxx S1000 M55 will demand a spindle speed of 1000 revs/min
and will call %200, the sub-program defined by M55, once.

Nxx M55 S1000 corresponds to 1000 consecutive calls to sub-program %200.

NOTE :

In a sub-program called by an M function or by the PLC, it is not possible to call another sub-program using an M function or the PLC.

11.10 – AUTOMATIC COMPUTATION OF CUTTING TIME

When a workpiece program is stored in the memory, the sum of all the individual axis movement times may be computed. This indication of time, takes no account of m/c synthetics. (ie. dwells caused by machine actions, such as tool lock/unlock, or times for the axes and the spindle to run up to speed, etc.).

The overall time is calculated by running the program in TEST mode. The result is expressed in minutes and seconds (max. 546 minutes). It can be viewed on the program LIST page.

eg. %1 6532 BYTES 86' 37"

11.11 – "FORCED BLOCK CONTINUATION" MODE : FUNCTION M997

The blocks programmed after this function are executed continuously until the program meets M998, M999 or M02.
For example, if the system is in block by block mode when M997 is read, the following blocks will be run as if the system was in continuous mode until the program meets a concelling function. Similarly, if a sub-program is called by an M function in MDI mode and M997 is programmed, the sub-program will be run in continuous mode until it meets a concelling function.

Another application consists of creating programmed cycles (eg. tapping cycles) with non-interruptible linked blocks. In the tapping application. The external FEED STOP can be taken to trigger a cycle withdrawal by using G10.

11.12 – READER OR DNC1 PASS MODE

This mode is used to machine while the program is either being read on the reader or transmitted by the DNC1.

The mode is used with long programs which cannot be stored, or for machining a single part, for instance.

Program characteristics :

- Calls to sub-programs in memory (G77 H...) are allowed.
- Any reference to block numbers in the pass program is <u>NOT</u> allowed :
  . jumps to blocks G79 or G10 N...
  . calls to a sequence of blocks G77 N... N
  . declarations of emergency withdrawal blocks G75 N...

NOTE :

The pass mode is indicated on the IN PROGRESS and LIST display pages, by PPR for reader pass mode and PPL for DNC1 pass mode.

11.13 - TEACH-IN FUNCTION

A program can be created or modified in teach-in mode. This function is carried out in "EDIT" mode, and the machine origins must have been set.

The "EDIT" mode provides access to the axis handlers, enabling positionig of the axes.

The character "!" automatically allows recall of the current point co-ordinates.

11.13.1 - Block modification

The co-ordinates contained in a block can be modified by the methods described in the "EDIT" mode, or by taking the current point values.

With the program selected and in "EDIT" mode, search for the block to be modified. Use the handlers to position the slide.

Enter # ! X Y Z LF — The X, Y and Z co-ordinates are modified in the target block if they exist.

or

! LF — All co-ordinates presented in the block are modified (if X alone is progammed, it will be modified).

or

! X LF — X alone is modified, if present.

Where an axis has to be added to the block, the block must be brought into dialogue-line, and the axis preceded by "!" should be added.

Example: >N100 X20
#LF
In dialogue-line : #N100 X20 !Z LF (!Z being additional characters)

11.13.2 - Adding blocks

In the "EDIT" mode, blocks may be added in an existing program.

Select the block behind which the new block is to be inserted, position the axis or axes at the point, and then, given that the current points of the axis names to the right of "!" are automatically entered, proceed as follows:

In dialogue-line : + N80 ! LF (all co-ordinates are introduced).

+ N90 F100 ! XLF

becomes

N90 F100 Xxxxx

+ N100 X150 ! Z LF

becomes

N100 X150 Zzzzz

11.13.3 – Creation of a program

In the "CHARGE" mode, key in a program. For example :

%2
M2
X OFF

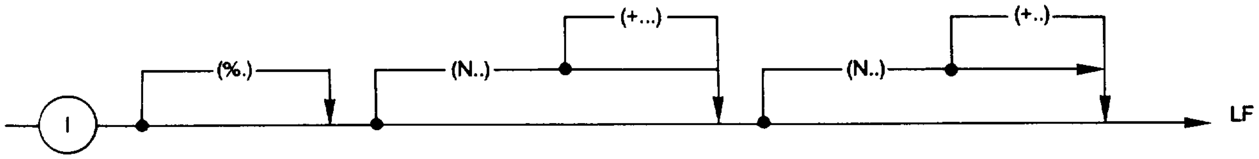
Then access the "EDIT" mode, and proceed as for "Adding blocks".

11.14 – INSERTION OF ALL OR PART OF A PROGRAM IN "EDIT" MODE

With this procedure you can duplicate a program, or insert parts of a program into another program.

Blocks are inserted in the "EDIT" mode after the block pointed by the cursor, using the character "I". The block number to be inserted must be less than 32767.

Access the "EDIT" mode, and select the program and sequence prior to insertion, then key in following the syntax below :



Any deviation from this syntax will cause the command to be denied, and the (dialogue) cursor will return to the beginning of the line, under the character "I".

When insertion is impossible due to shortage of storage capacity, the message "MODIFICATION IMPOSSIBLE" appears.

COMMENTS :

- When the program number %... is not specified, the program being edited is addressed.
- The pair of sequence numbers constitutes the boundaries of the insertion zone. Both boundaries are included in the insertion.
- When a command defines a single sequence number N.. the number addresses the start of the insertion zone, as the end boundary is also the end of the program itself.
- A boundary can be specified by a sequence number plus a value. For example, if N10 + 3 is programmed, the boundary is the 3rd block after the N10 sequence.

- EDIT Mode

```
    %2                          %4
    N10...                     _N10...
    N20...                    | N20...
    N30...                    | N30...
    N40...          ----------| N40...
    N50...         |          | BLOCK 1 (NOT ADDRESSED)
>   N60...  <------           |_BLOCK 2 (NOT ADDRESSED)
    N70...                      BLOCK 3 (NOT ADDRESSED)
                                BLOCK 4 (NOT ADDRESSED)
                                N50...
                                N60...
```

Dialog line : I %4 N20 N40 + 2 : Sequences N20 to BLOCK 2 (NOT ADDRESSED) of program 4 will be inserted after sequence 60 of %2.

The block pointed to, "Np", must not be :

- inside the limits of the part of the program to be inserted,
- one of the limits of the part of the program to be inserted.

Example :

```
    %1
    N10
    '
    '
>   Np
    '
    '
    N50
    '
    '
    N100
```

I Np N50 is impossible. The message "MODIFICATION IMPOSSIBLE" is displayed.

11.15 - SETTING-UP OF "RECALL" MODE : FUNCTION M12

Function M12 is an M-function which is decoded "after". it sets up the "RECALL" mode, illuminating the FEED STOP and CYCLE indicators, and hands over to the axis handlers or manual manipulation crank, in unrestricted mode only, as the system remains in the program run "CONT SEQ" mode.

In TEST or RNS mode, this function is not available. When the operator unables FEED STOP this cancels function M12 and causes the program run to resume from the new slide position. There is NO axis recall.

Function M12 is present in the PLC on the W1.11 bit as long as the FEED STOP indicator is illuminated.

11.16 - POSITIONING AT A PROGRAMMED DISTANCE FROM A POINT

This function is active only with linear interpolation G0 or G1. The system positions the axis on the programmed path at a distance equal to the tool radius from point X Y Z specified in the block, before the point if the sign following R is "-" or after if the sign is "+". This offset is applied only to the axes of the interpolation plane.

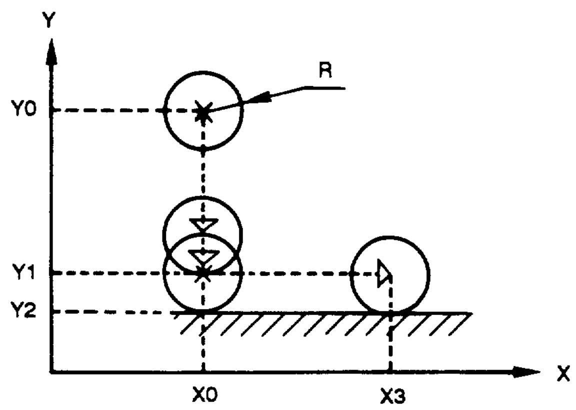
This block must not include any PGP, blends or chamfers.

Block syntax :

G1 (or G0) R+ (or -) X 053 Y 053 Z 053

Example of Programming

Axis positioned near the point where radius correction is to be entered : X3 Y2 Z0



N10 X0 Y0 Z0
N20 G0 R- Y1 D..
N30 G1 G41 Y2
N40 X3
.
.
.

11.17 – TRANSFER OF THE CURRENT PARAMETER VALUES INTO THE PART PROGRAM : FUNCTION G76

Function G76 allows update of an E and L parameter file with the new contents of the corresponding active data.

This file is a sequence of blocks that can be stored in the part program itself or a dedicated sub-program.

The block syntax is :

G76	Hp	Editing of the parameters of program %p.
	Nn1 Nn2	Editing of the parameters defined between Nn1 and Nn2 of the current program.
	Hp Nn1 Nn2	Editing of the parameters defined between Nn1 and Nn2 of program %p.

Editing of the parameters of a block continues until a function other than declaration of an E or L parameter is encountered in the block.

EXAMPLE :

```
G76  N100  N120
N100 ........................................................ (LF)
L101 = ........... E  80001         = ............... (LF)
L4    = ........... G4F6E52002  = ............... (LF)
L6    = ...........                                    (LF)
G1  X100  L3 = ........................................ (LF)
N120 ........................................................
```

Only parameters L101, E80001, L4 and L6 can be edited. The field where the new value of the parameter is loaded (after the symbol "=") includes at least 10 characters with only spaces, decimal digits and sign and decimal point where required.

Failure to comply with this format results in error 97.

EXAMPLE OF PROGRAMMING :

```
% 125
N10 G77 H200 N50 N80        Data file transfer
     Execution of the program
     with editing of the
     parameters.
N600 G76 H200 N50 N80       Data file update
M02

% 200
N50
L1 = ______________         E52002 = ______________
E80004 = ______________     minimum 10 characters or spaces.
N80
```

When the blocks between N50 and N80 of program 200 are written, an initialization value is written after the symbol "=", supplemented with spaces to obtain at least 10 characters or, if nothing is written, at least 10 spaces.

The data of program 200 become active at the beginning of program 125. Program 200 is updated at the end of program 125.

11.18 – SPLINE CURVES

The spline curves processed by the system are curves whose axes are interpolated by functions whose second derivative is zero or whose first derivative is known at either or both ends.

On a given curve, certain axes can have a second derivative equal to zero and others a first derivative known at one end.

Programming of the points connected by a spline curve precedes the curve execution instruction.

11.18.1 – Definition of the points on the SPLINE curve

The curve definition blocks are delimited by a header block and an end block.

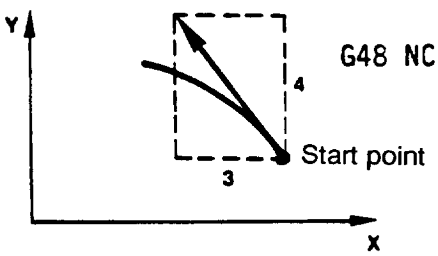
11.18.1.1 – Header block

This block begins with function G48 followed by function NC.. defining the curve number.

The axes whose first derivative is known in the start point are also included in this block. The values that accompany them are the relative dimensions whose result defines the tangent to the curve in its start point.

The axes whose second derivative is zero in the start point are not included in this block.

EXAMPLE : The projections of the tangent to the start point along X and Y have 3 and 4 as relative values. Programming of the header block gives :



G48 NC1 X-3 Y4

11.18.1.2 – Curve definition blocks

After the header block are programmed the blocks defining the coordinates of the points connected by a spline curve.

Each block must include all the interpolated axes.

A spline curve cannot be defined by only two points (at least three) or error 255 is displayed.

11.18.1.3 – End block

The last curve definition block is followed by an end block including function NC0. The axes whose derivative is known in the end point are also included in this block. The values accompanying them are the relative dimensions whose result defines the tangent to the curve in the end point.

For the other axes, the second derivative is zero in this point.

11.18.2 – Execution of a SPLINE curve

The execution instruction is given by function G06 followed by function NC.. defining the curve number.

Function G06 is nonmodal and forces modal state G01.

If the system is in state G41 or G42, the tool is offset normal to the curve in the interpolation plane (G17, G18 or G19).

- Interpolations

  The system cuts the curves into equally spaced line segments. Segmenting is carried out over time and the segment length depends on the programmed feedrate.

  The second word of machine parameter P51 gives the curve segment period.

  IMPORTANT :

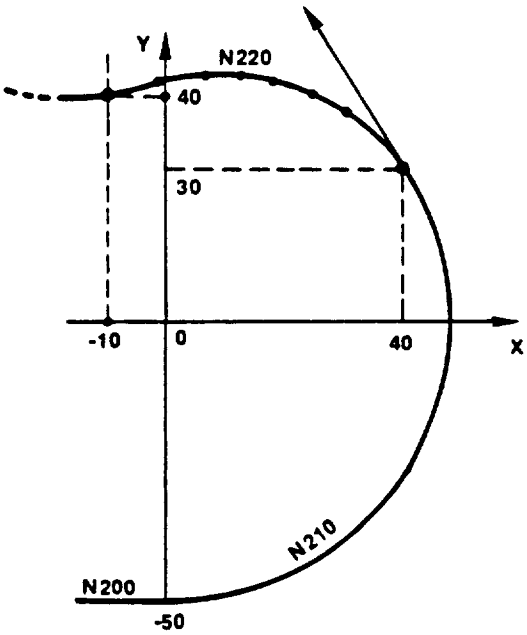
  In order for the first segment to be correctly interpolated, the first point of the curve must be programmed in the block preceding the declaration of G06.

  If a circle defined by its radius is programmed after the spline curve, the circle start point must be reprogrammed. Otherwise error 107 is displayed.

  EXAMPLE :

```
N5 G48 NC1 X-3 Y4
N10 X40 Y30
N15 X-10 Y40
:
:
N90 NC0
:
:
N200 G1 X0 Y-50
N210 G3 X40 Y30 I0 J0
N220 G06 NC1
:
```

The SPLINE curve start point (N10) is also programmed in block N210



11.18.3 – Programs including several SPLINE curves

During read of a spline curve definition (function G48), the system computes all the functions used to perform interpolation later when reading G06.

These values are stored in a buffer whose size is defined by the second word of machine parameter P58.

The size allocated may be insufficient to contain all the curves. However, the space occupied by curves already executed can be released in order to be able to define other curves.

Release is carried out by nonmodal function G48 followed by the number of the curve to be deleted (NC..).

NOTE :

Functions G48 and G49 must be programmed with radius offset cancel (G40) as their execution stops motion.

<u>EXAMPLE</u> :

```
:
:
G48  NC1  X..  Y..   }
X..  Y..  Z..        }    First spline curve definition
X..  Y..  Z..        }
:         :
:         :
NCO
:
G48  NC2             }
X..  Y..             }    Second spline curve definition
X..  Y..             }
:
NCO
:
G06  NC1                  First spline curve execution
:
G06  NC2                  Second spline curve execution
:
G49  NC1                  First spline curve deletion
:
G49  NC2                  Second spline curve deletion
:
```

11.18.4 – Restrictions

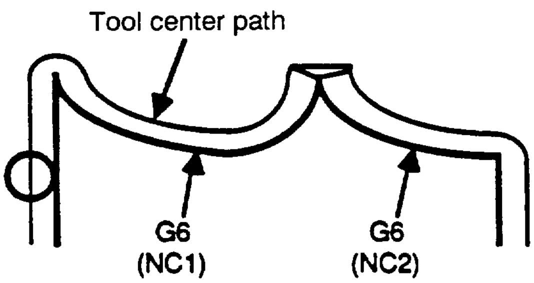
- Modulo programming of the axes is not recommended (problem of the sign and no movement).
- The mirror, angular offset and scaling factor functions are ignored.

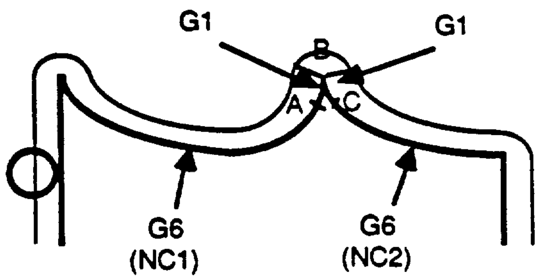
NOTES :

- *The spline curve programming error numbers are 250 to 256.*
- *Sequencing of two intersecting spline curves with radius correction.*

  When sequencing the two curves with radius correction, truncation may occur depending on the angle between the two curves since the correction stops on the normal to the end of the first curve and is resumed on the normal to the beginning of the second curve. The two points are connected by a line during machining.

  To cater for this drawback, a linear portion is programmed at the end of the first curve and the beginning of the second curve.

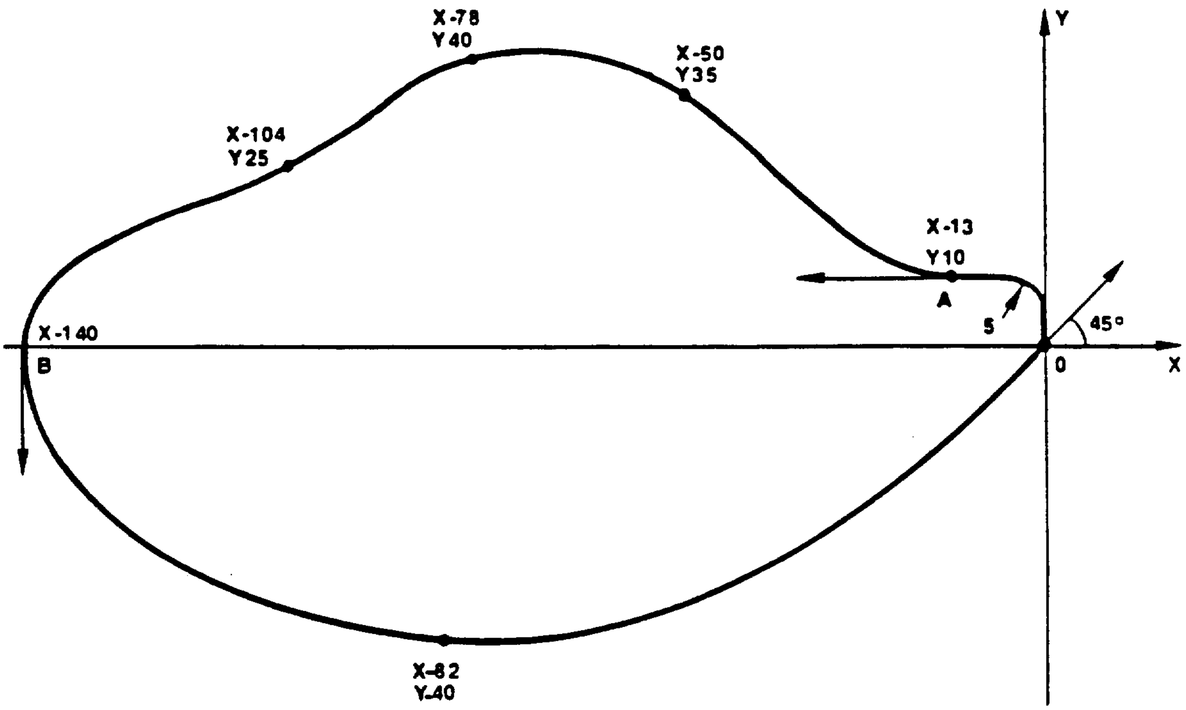





```
%----
G41
!
!
G06 NC1
G06 NC2
```

```
%----
G41
!
!
G06 NC1
G1   AB
G1   BC
G06 NC2
```

11.18.5 – Example




```
N10  G48 NC1 X-1 Y0      }
N15  X-13 Y10            }
N20  X-50 Y35            }
N30  X-78 Y40            }  Definition of first spline curve from A to B
N40  X-104 Y25           }
N45  X-140 Y0            }
N50  NC0 X0 Y-1          }
N60  G48 NC2 X0 Y-1      }
N70  X-140 Y0            }
N80  X-82 Y-40           }  Definition of second spline curve from B to 0
N90  X0 Y0               }
N100  NC0 X1 Y1          }
N110  G0 X20 Y0 D1
N120  G1 G42 X0 Y0       }
N130  EA90 EB5 ES-       }  Execution of 0 A
N140  EA180 X-13 Y10     }
N150  G06 NC1               Execution of A B
N160  G06 NC2               Execution of B 0
N170  G0 G40 X20 Y0
```

NOTES

12 - STRUCTURED PROGRAMMING

The system provides the possibility of programming structured branches and loops, making the programmes easier to read and simplifying programming for complex part programmes.

12.1 - GENERAL

A structured sequence always begins with one of the keywords IF, REPEAT, WHILE, FOR. It ends with ENDI for IF, UNTIL for REPEAT, ENDW for WHILE and ENDF for FOR. IF corresponds to a branch and REPEAT, WHILE and FOR to a loop.

- ELSE can be interposed between IF and ENDI.
- Fifteen structured nesting levels are possible, independently of G77.

	First nesting level	Second nesting level	Third nesting level
IF			
	IF		
		REPEAT	
		.	
		.	
		.	
		UNTIL	
	ENDI		
ENDI			

- Programming a conditional or unconditional branch with G79 is allowed in a structured section but must branch to the lowest nesting level of the current programme or subroutine.

EXAMPLE :

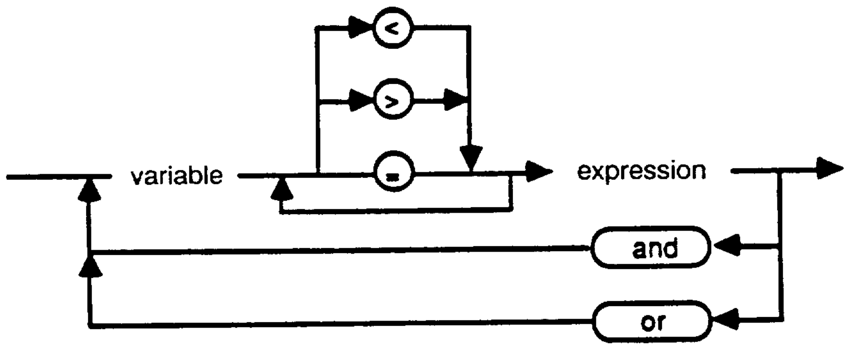
```
%1
IF
   WHILE
      G79 N100 OK
      G77 H2
      G79 N50 error
      G79 N30 error
N30
      ENDW
N50
   ENDI
N100
M2
```

```
%2
G79 N100 OK
REPEAT
     IF
        G79 N100 OK
        G79 N50 error
     ENDI
N50
     UNTIL
N100
```

12.2 - LIST OF COMMANDS

12.2.1 - Definition of terms

- Condition :



- Variable : All the L variables, symbolic variables and E parameters used in parametric programming.
- Expression : Sequence of parameters and immediate values connected by symbols +, -, *, /, !, &.
  The computations are made in sequence.

12.2.2 - IF

IF condition THEN (1)

(ELSE)

ENDI

ELSE is optional.

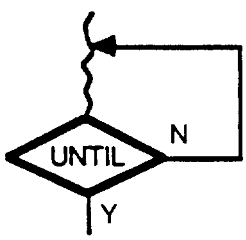
Condition true   Condition false

(1) THEN can also be programmed at the beginning of the next block and followed in the same block by functions to be executed :

Example : IF L4 < 8
THEN L6 = L2 + 1 X L6

12.2.3 - REPEAT : equivalent to an upward loop

REPEAT
UNTIL condition

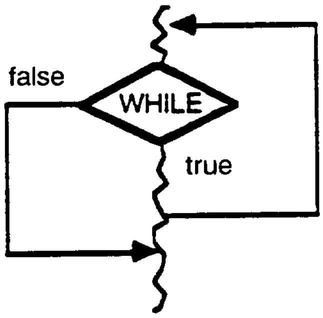


12.2.4 - WHILE

WHILE condition DO
ENDW

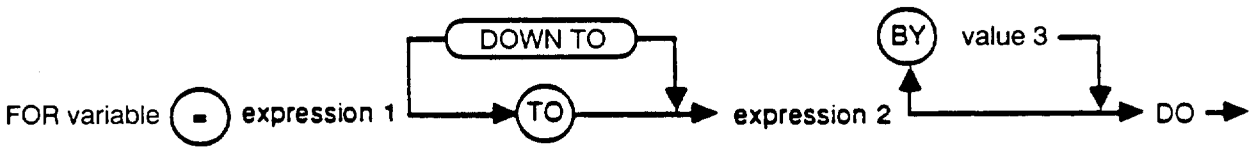
DO can also be programmed at the beginning of the next block and followed in the same block by functions to be executed.

Example : WHILE condition
DO X L6



If DO is omitted after WHILE, the system generates an error message (E191).

12.2.5 - FOR



- The variable can be an L function, a symbolic variable or an E80000, E81000 or E82000 parameter (be careful about stopping computations with L100 to L199 and E...).

  The expressions are positive or negative integer values. The system rounds them off (down to 0.5 and up above 0.5).

- With the TO function, the loop is no longer executed when the current variable value becomes higher than the final value (variable increment).

- With the DOWN TO function, the loop is no longer executed when the current value falls below the final value (variable decrement).

- The default value of BY is +1 to increment or decrement the variable each iteration. The value of BY must always be positive. If it is negative, TO and DOWN TO are reversed.

12.2.6 - EXIT

EXIT is used to exit from the current iteration loop to go down to the next lower nesting level (FOR, WHILE or REPEAT loop), bypassing the IF instructions in this loop.

EXIT must be included in an IF–THEN or ELSE– condition.

EXAMPLES :

```
REPEAT --                                  WHILE
______                                        REPEAT
IF ______          condition THEN                IF
______                                              IF
______                                                IF
ELSE                                                     EXIT
EXIT                                                     ENDI
ENDI                                                   ENDI
______                                              ENDI
______                                           UNTIL
UNTIL              condition                  ENDW
```

12.3 - SYNTAX

- The words IF, REPEAT, WHILE, FOR, ELSE, ENDI, UNTIL, EXIT, ENDW and ENDF must be the first in a block. (They must not be preceded by a sequence number).
- The condition must follow words IF, WHILE and UNTIL in the same block. Similarly, the limits and increment must follow FOR.
- DO and THEN must immediately follow the condition. If they are not in the same block as IF, WHILE or FOR, they must be the first words in the next block.
- DO, THEN and ELSE can be followed in the same block by ISO functions.

  Example :
  ```
  WHILE L1 < 3  DO G91 X 12
  or
  WHILE L1 < 3
  DO G91 X 12
  ```

  The following programming is refused :
  ```
  WHILE L0 < 3 G91 X 10
  DO
                 prohibited in the condition
  ```

- Blocks beginning with ENDI, ENDW, ENDF, EXIT or UNTIL must not include ISO functions.
- The space is a mandatory separator after IF, REPEAT, THEN, ELSE, UNTIL, WHILE, DO and DOWN TO.

  Example : WHILEL0 < 3 is not recognized by the system.

- ISO blocks with sequence numbers are allowed in loops (see programming example, page 9–19).

13 - PROGRAM STACK - SYMBOLIC VARIABLES

Symbolic variables are symbolic names assigned to the variables used in parametric programming. The number of variables can be extended by using symbolic names.

Each axis group (graphic axes included) has a stack located in the bottom of the memory for saving L variables and assigning symbolic variables.

13.1 - STACK ALLOCATION

The stack is allocated in MDI mode by declaring the stack size in bytes after % between square brackets.

Example : % [2000] (LF) (cycle)

Allocation of 2000 bytes for each stack. For the milling machine, there is one stack for the axes and one for the graphic display.

The stack size is remanent. It is only modified by allocating a new size. A reset deletes the contents of the stack but not the area allocated.

13.2 - SAVING AND RESTORING L VARIABLES

PUSH Lm - Ln

Variables Lm to Ln (inclusive) are saved in the stack, where m and n are numbers within one of the following ranges : 0-19 or 100-199 or 900-939.

PUSH L1-L110 is prohibited. It is necessary to program PUSH L1-L19
PUSH L100-L110

PULL LM - Ln

Restore Lm to Ln with the values of the last PUSH made using the same set of variables Lm–Ln.

```
%1                                          %1
.                                           .
.                                           .
L0 = 1                                      L0 = 1
.                                           .
.                                           .
PUSH L0-L10                                 PUSH L0-L10
.                                           .
.                                           .
L0 = 2                                      L0 = 2
.                                           .
.                                           .
PUSH L0-L10                                 PUSH L0-L11
.                                           .
.                                           .
PULL L0-L10      } Recover the              PULL L0-L11 => L0 = 2
.                } last PUSH                .
.                } L0 = 2 in                .
PULL L0-L10      } both cases               PULL L0-L10 => L0 = 1
```

PULL can only be used to restore variables saved in the current program or sub-program or those from lower nesting levels. To be accepted, a PULL must always be preceded by a PUSH.

```
%1                      %2
.                       .
.                       .
PUSH L0-L10             PULL L0-L10
.                       .
.                       .
G77  H2                 G77  N1  N2      }
.                       .                }
.                       .                }
                        PULL L10-L12     }
                        .                }
                        .                }
                        N1               } Prohibited
                        .                }
                        .                }
                        PUSH L10-L12     }
                        .                }
                        .                }
                        N2               }
```

PUSH and PULL must be the first words in a block and must be followed by a space (no sequence number starting the block).

13.3 - SYMBOLIC VARIABLES

13.3.1 - Variable declarations

Variables are declared between the words VAR and ENDV.
VAR and ENDV must be the first words in a block. VAR must always be followed by a space.

Preparation and execution of the blocks are synchronized except in M999. The symbolic variables are declared between square brackets : [...]

The symbols are from 1 to 8 alphanumeric characters, with the first character always alphabetic.

```
Example :  VAR    [INDEX]     [TOTO]
                  [PHASE]
           ENDV               (only word in the block)
```

To define an array of variables, the array dimensions are included after the last character between round brackets.

A table can have up to 4 dimensions. The symbol "," or ";" separates the values of the different dimensions.

Example : [TABL1 (10)] [TABLE2 (2,5,3)]

For table 2, 2 x 5 x 3 = 30, i.e. 5 groups of 2 then 3 groups of 10 elements are reserved.

The value of a dimension must be between 1 and 65535 for a one-dimensional table and 1 and 255 for tables with 2, 3 or 4 dimensions.

Dimensions must be declared as immediate values or symbolic variables already initialised. (L variables and E parameters cannot be used to define a dimension).

```
Example :  [VAR1] = 10
           [VAR  ] = [TAB1 (VAR1,5)] is equivalent to [TAB1 (10,5)]
```

Symbolic variables have real values (i.e. non-integer values).

13.3.2 - Static initialisation of variables and arrays

The default value is zero.

Variables and arrays can be initialised with values by declaring the initial values separated by "," after an equals character.

The initial values can be declared over several blocks. In this case, the character "=" must be declared before the values in the new block.

```
Example :  VAR [NBT] = 4 [TABLE (2, NBT)]  = 3, 6 (LF)
                                           = 10, 1, 8, 2, 6, 6 (LF)
           ENDV
                                        If L0 = [TABLE (2,3)]
                                          L0 = 2 (6 digits)
```

13.3.3 - Use of symbolic variables in an ISO programme

Symbolic variables can be assigned to all ISO functions and can be used in parametric expressions. They can also be associated with L variables and E parameters. They can be used in structured branch and loop commands.

The array indexes must be immediate values or symbolic variables (L and E parameters are prohibited). If [TAB (L0,3)] is programmed, it is not paramater L0 but variable [L0] that is searched for.

Array indexes can also be the results of additions and/or subtractions of values or symbolic variables.

Example :
```
VAR     [IX] [COSX] [SINX] [NBT] = 4
        [TABL (2, NBT)] = 0, 0, 10, 5, 20, 8, 30, -2
ENDV
FOR     [IX]= 1 TO [NBT] - 1 D0
        [COSX] = [TABL (1,IX+1)] - [TABL (1, IX)]
        [SINX] = [TABL (2,IX+1)] - [TABL (2, IX)]
        L0 = [COSX]*[COSX] LO = [SINX]*[SINX]+ L0
        [COSX] = [COSX]/ RLO   [SINX]=[SINX]/ RLO

        X[TABL(1,IX+1)] Y[TABL(2,IX+1)]
ENDF
```

- Access is only possible to the variables declared in the current programme or subroutine and those of lower nesting levels.

13.3.4 - Structure of arrays with several dimensions

The entries of the first dimension are given at the beginning of the table and are multiplied by the entries of the second dimension, the result being multiplied by the number of entries of the third dimension, etc.

13.4 - CREATION OF AN ARRAY TO STORE A PROFILE-BUILD FUNCTION

The system provides the possibility of storing a profile written in ISO or PGP in an array on the stack. The array is built as the profile blocks are read. The executable blocks are stored in a two-dimensional array. The first dimension is the set of functions to be saved in a block and the second is the number of blocks in the profile.

Parametric programming then gives access to the data in this array.

13.4.1 – Data in a block that can be stored on the stack

- G Function :

  Only one G function can be stored per block. For a change of interpolation plane in a block, the new function G17, G18 or G19 is stored. Otherwise, modal function G0, G1, G2 or G3 is stored in the array.

- Axis Dimensions :

  According to machine parameter P0 :

  . X, Y, Z, U, V, W, A, B, C for a milling machine

    It is the modal values of these functions that are stored in the array. The write format for these dimensions is 5.3 (Inches or tenths of a micron are not taken into account).

- Other Dimensions :

  . I, J, K, P, Q, R

    These values are not stored in the array unless they are programmed in the block. If they are not programmed, these entries are initialised with zero. The write format for these dimensions is 5.3.

- Feed Rate – F Function :

  The modal value of the F function is stored in the array. The write format for the value is 5.2 for mm/min or 2.3 for mm per revolution.

- T Function :

  The T function is not stored in the array unless it is programmed in the block. Otherwise, it is initialised with zero.

  The T function has an absolute value of five decimal digits at most.

- S Function :

  The S function is not stored in the array unless it is programmed in the block. Otherwise, it is initialised with zero. The S function has an absolute value of five decades at most or format 2.2, depending on the spindle characteristics (see machine parameter P7).

NOTE :

Error 1 is generated if the wrong format is used for the variables, e.g. F [FEEDRATE] with [FEEDRATE] = 123456.

13.4.2 – Programming

```
                                       H......
BUILD [TAB  (G/X/Z/I/K,NB)]            N.. N.. N..
                                       N.. N..
                                       N.. +n  N.. +m
```

Limits of the profile

Variable name containing the number of blocks

Specification of the functions included in the array

Array name on the stack

Call to the array build function for a profile

The two-dimensional array TAB and variable NB are automatically created by this programming.

- The word BUILD followed by the array declaration must be programmed at the beginning of a block and separated by at least one space.
- Additional entries, not initialised by block data, can be allocated in the first array dimension. They are declared by digits "0" and separated by "/".

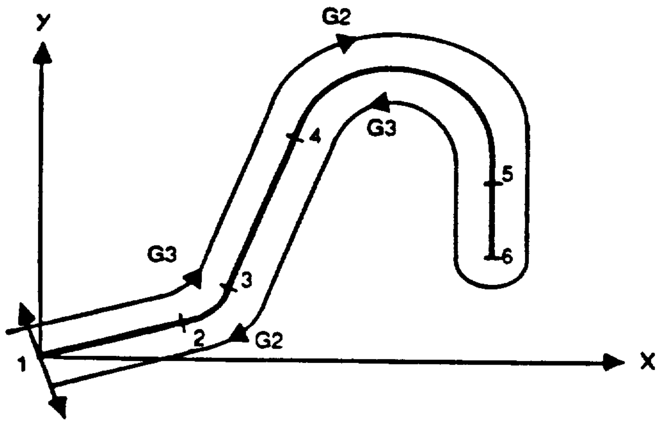
Example : [PROF1 (G/X/Y/Z/O/O, NBP)]
The first dimension of PROF1 has 6 entries.

- A maximum of 16 entries can be declared in the first dimension.
- A maximum of 255 blocks can be stored in the array (limit when there are several dimensions).

13.4.3 – Programming errors

E.195 : Program stack saturated
E.196 : Table dimensions out of limits
E.198 : Error in array or variable symbol declaration (the first character of a symbol must be alphanumeric)
E.199 : Variable and table declaration syntax error (square or round brackets, commas)
E.82 : First or last block of the profile incomplete (PGP)

13.4.4 – Example : path of the curve from points 1 to 6 and return to 1 with radius correction



Pts

[NB]-1

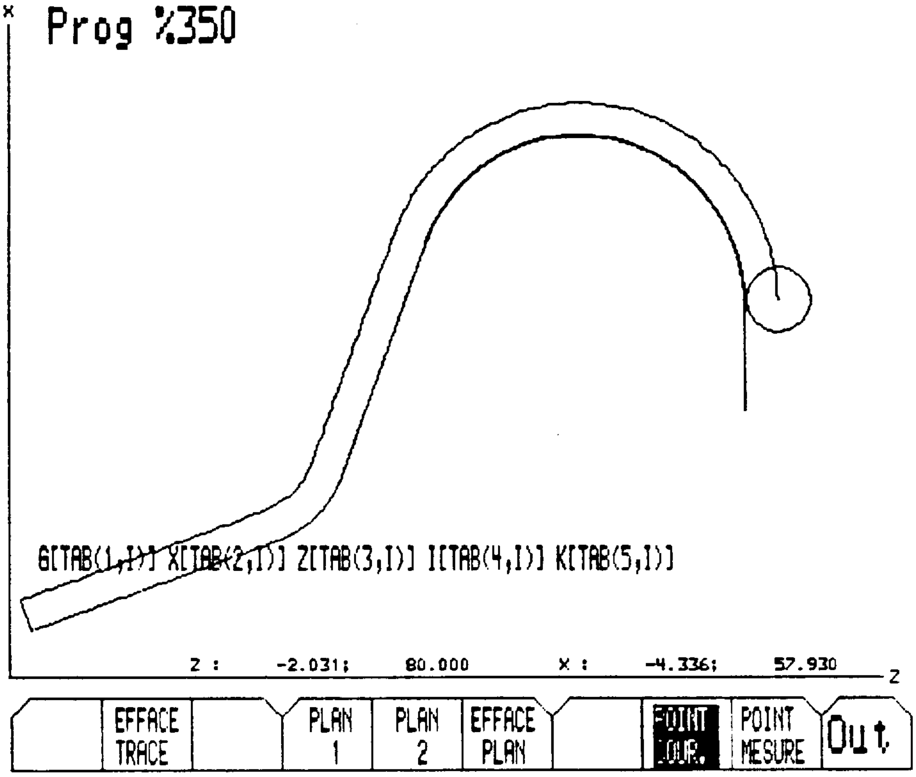
Pts	G	X	Y	I	J
1	1	0	0	0	0
2	1	22.266	8.104	0	0
3	3	28.243	14.081	18.846	17.501
4	1	35.905	35.13	0	0
5	2	65	30	50	30
6	1	65	20	0	0

G [TAB (1,I)]

X [TAB (2,I)]

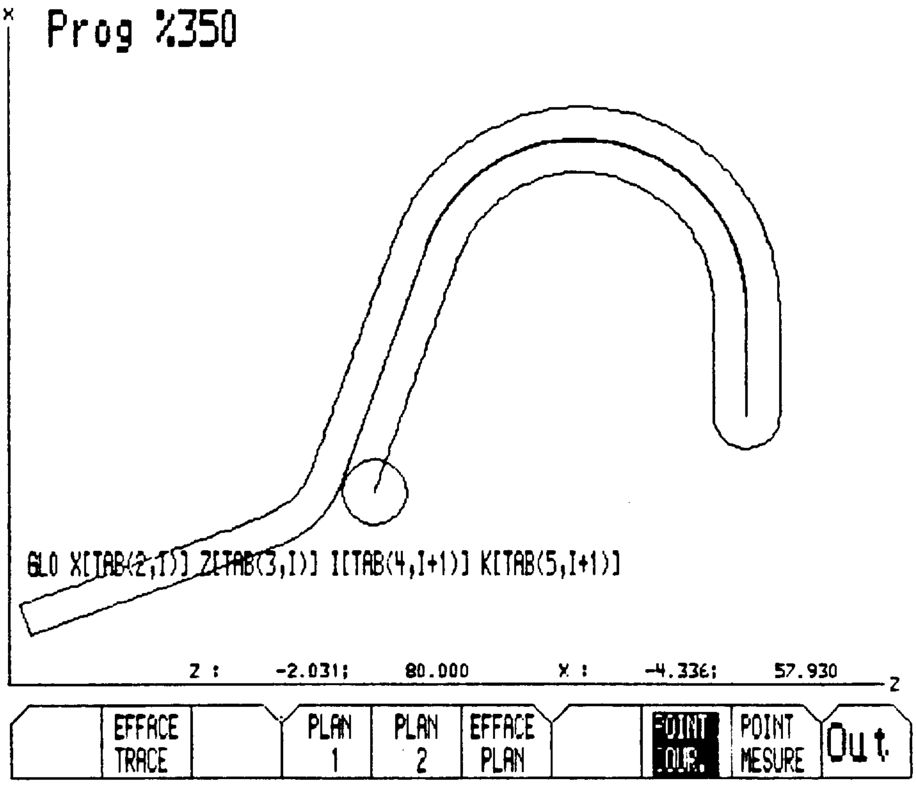
Value of the array after execution of BUILD

```
% 8
Z
M999  G79  N100
N10    G1  X  Y
       EA20  ES  EB10
       EA70                          Curve definition (points 1 to 6)
       G2  I50  J30  R15
N40  G1  Y20
N100
BUILD [TAB (G/X/Y/I/J,NB)] N10  N40
VAR  [I]
ENDV
FOR  [I]  =  1  TO  [NB]  DO  G41  D1      (path along the curve from 1 to 6)
G[TAB (1,I)]  X  [TAB(2,I)]  Y[TAB(3,I)]  I[TAB(4,I)]  J[TAB(5,I)]
ENDF
FOR  [I]  =  [NB]  -1  DOWNTO  1  DO    (path along the curve from 6 to 1)
LO  =  [TAB(1,  I+1)]
   IF  LO>1  THEN LO  =  LO*3+1&3      reversal of G2 and G3 for the path from 6 to 1
   ENDI
GLO  X  [TAB(2,I)]  Y  [TAB(3,I)]  I[TAB(4,I+1)]  J[TAB(5,I+1)]
ENDF
G40  X  Y
M2
```



Prog %350
G[TAB(1,I)] X[TAB(2,I)] Z[TAB(3,I)] I[TAB(4,I)] K[TAB(5,I)]
Z : -2.031; 80.000 X : -4.336; 57.930
EFFACE TRACE
PLAN 1
PLAN 2
EFFACE PLAN
POINT COUR.
POINT MESURE
Out

HC 1



Prog %350
G10 X[TAB(2,I)] Z[TAB(3,I)] I[TAB(4,I+1)] K[TAB(5,I+1)]
Z : -2.031; 80.000 X : -4.336; 57.930
EFFACE TRACE
PLAN 1
PLAN 2
EFFACE PLAN
POINT COUR.
POINT MESURE
Out

HC 2

13.5 - DELETION OF VARIABLES AND ARRAYS ON THE STACK

Deletion is not carried out until the block preceding the deletion command has been executed.

13.5.1 - Automatic variable deletion

- The return from a subroutine deletes all the variables declared in the subroutine and deallocates the space they occupied on the stack.
- The end of programme (M02) or a reset by the operator deletes all the variables and clears the stack.

13.5.2 - Programmed variable deletion

Deletion is programmed by the DELETE function followed by the list of variables and arrays in square brackets, separated by the character "/".

Example : DELETE [IX] / [TAB1] / [PROF1]

- The list of variables to be deleted must be separated by at least one space from the word DELETE. However, there must be no spaces in the list of variables.
- DELETE does not delete the variables of the calling programme.
- DELETE must be the first word in the block.

NOTE :

The system recognises only the first four letters of the keywords.

For instance, command :

DELE [IX] / [TAB1] / [PROF1] is accepted.

NOTES

APPENDIX 1

LIST OF ERRORS
This full list of errors may not apply to all installations

* E.1 : UNACCEPTABLE CHARACTER
* AXIS NOT RECOGNIZED BY THE SYSTEM ?
* TOO MANY DIGITS AFTER A FUNCTION ?
* A SIGN WITH A FUNCTION WHICH SHOULDN'T HAVE ONE ?
* INCORRECT FORMAT ?
SYMBOLIC VARIABLE FORMAT ERROR. e.g. F[FEEDRATE] WITH [FEEDRATE] = 123456
In G84, RIGID TAPPING, PITCH < 0 OR > 320000 MICRONS

* E.2 : G FUNCTION NOT RECOGNIZED BY THE SYSTEM
*
*
*

* E.3 :
*
*
*

* E.4 : PARAMETRIC PROGRAMMING OPTION NOT ENABLED
* LETTER O PROGRAMMED IN PLACE OF ZERO
* RIGID TAPPING OPTION NOT PRESENT
*

* E.5 : GEOMETRY PROGRAMMING OPTION NOT ENABLED
*
*
*

* E.6 :
*
*
*

* E.7 :
*
*
*

* E.8 : TOOL COMPENSATION NO. TOO BIG
*
*
*

* E.9 : TOO MANY CONSECUTIVE NON WORKING BLOCKS
* ENDLESS CALCULATION LOOP ?
*
*

* E.10 :
*
*
*

* E.11 :
*
*
*

* E.12 :
*
*
*

* E.13 :
*
*
*

* E.14 :
*
*
*

* E.15 :
*
*
*

* E.16 :
*
*
*

* E.17 : MISSING CLOSE BRACKET
*
*

* E.18 :
*
*
*

* E.19 :
*
*
*

* E.20 : M02 MISSING AT END OF PROGRAM
*
*
*

* E.21 : BLANK DEFINITION INCOHERENCE (IN 3D)
*
*

* E.22 :
*
*
*

* E.23 :
*
*
*

* E.24 :
*
*
*

* E.25 : UNDEFINED SUB-PROGRAM OR SEQUENCE NUMBER
*
*
*

* E.26 : TOO MANY SUB-PROGRAM NESTINGS LEVELS
*
*
*

* E.27 : RADIUS COMPENSATION
* – IN MACHINE ORIGIN PROGRAMMING (G52)
* – OR DURING AN INTERRUPTIBLE MOVE (G10)
* – OR WITH M00 OR M01 IN THE BLOCK
* – OR IN A DRILLING CYCLE

* E.28 : NO RANGE COMPATIBLE WITH S
*
*
*

* E.29 :
*
*
*

*---		Operational faults (E30 to E33 - E36 - E40 to E43)

* E.30	:	READER UNCONNECTED OR FAULTY
*		OR DNC LINE FAULT
*		OR PLOTTER FAULT
*
*

* E.31	:	DATA EXCHANGE WITH OPERATOR PANEL FAULT
*
*
*

* E.32	:	AXIS REFERENCING ERROR (SLIDE ALREADY ON ITS SWITCH)
*
*
*

* E.33	:	FAULT IN FIBRE - OPTIC LINK
*
*

* E 34	:
*
*

* E.35	:	SEARCHED FOR SEQUENCE NO. NOT FOUND
*
*
*

* E.36	:	WORKPIECE PROGRAM MEMORY FULL
*		MEMORY AREA INSUFFICIENT FOR PASS MODE
*
*

* E.37	:
*
*
*

* E.38	:	SPINDLE SPEED TOO HIGH FOR RIGID TAPPING
*		(DECELERATION DISTANCE GREATER THAN 128000 MICRONS)
*
*

* E.39	:
*
*
*

* E.40 : EXCESSIVE SERVO FOLLOWING ERROR – AXIS 0
*
*
*

* E.41 : EXCESSIVE SERVO FOLLOWING ERROR – AXIS 1
*
*
*

* E.42 : EXCESSIVE SERVO FOLLOWING ERROR – AXIS 2
*
*
*

* E.43 : EXCESSIVE SERVO FOLLOWING ERROR – AXIS 3
*
*
*

* E.44 :
*
*
*

* E.45 :
*
*
*

* E.46 :
*
*
*

* E.47 :
*
*
*

* E.48 :
*
*
*

* E.49 :
*
*
*

*---

* E.50 :
*
*
*

* E.51 :
*
*
*

* E.52 :
*
*
*

* E.53 :
*
*
*

* E.54 :
*
*
*

* E.55 :
*
*
*

* E.56 :
*
*
*

* E.57 :
*
*
*

* E.58 :
*
*
*

* E.59 :
*
*
*

*---   Errors generated from the machining cycles

* E.60 : MACHINING AXIS MISSING IN G45
*
*
*

* E.61 : INTERPOLATION PLANE INCORRECTLY DEFINED IN G45
*
*
*

* E.62 : PROGRAMMING ERROR. CHECK IF ANY OF THE FOLLOWING ARE MISSING :
* - A WIDTH/DEPTH OF CUT SETTING?
* - A FEED RATE?
* - A DIMENSION (EB, EX, EY, EZ)
* A SPINDLE SPEED? OR DIRECTION?

* E.63 : PARAMETER WITH A ZERO OR NEGATIVE VALUE
* - DIMENSION?
* - A WIDTH/DEPTH OF CUT SETTING?
* - A FEED RATE?

* E.64 : A PARAMETER IS INCOMPATIBLE WITH THE TOOL RADIUS
*
*
*

* E.65 : POCKET DIMENSIONS INCOMPATIBLE WITH THE
* INTERPOLATION PLANE
*
*

* E.66 : POCKET DIMENSIONS INCOMPATIBLE WITH DEMANDED
* WIDTH/DEPTH OF CUT SETTINGS
*
*

* E.67
*
*
*

* E.68 :
*
*
*

* E.69 :
*
*
*

* E.70 :
*
*
*

* E.71 : INCOMPLETE DATA, RELATIVE TO THE PREVIOUS BLOCK
*
*
*

* E.72 :
*
*
*

* E.73 :
*
*
*

* E.74 :
*
*
*

* E.75 :
*
*
*

* E.76 :
*
*
*

* E.77 :
*
*
*

* E.78 :
*
*
*

* E.79 :
*
*
*

* ---

* E.80 :
*
*
*

* E.81 :
*
*
*

* E.82 :
*
*
*

* E.83 :
*
*
*

* E.84 :
*
*
*

* E.85 :
*
*
*

* E.86 :
*
*
*

* E.87 : SYNTAX ERROR IN THE DEFINITION OF A G83 OR G87 CYCLE :
* – M3 OR M4 MISSING ?
* – P AND/OR Q, ZERO OR MISSING ?
* – P INCOMPATIBLE WITH THE CYCLE ? ; DWELL INCOMPATIBLE WITH THE CYCLE?
WITH RIGID TAPPING, HOLE BOTTOM DIMENSION INCONSISTENT WITH CLEARANCE DIMENSION

* E.88 : – SYNTAX ERROR IN A FIXED CYCLE DEFINITION
* – THE INTERPOLATION PLANE AND TOOL DIRECTION DO NOT CORRESPOND?
* – ER PROGRAMMED BEFORE THE MACHINING AXIS ?
* – POSITIONING AXIS MISSING IN THE FIRST BLOCK IN G86 ?
– G52 PROGRAMMED IN A BLOCK OF CYCLE G86

* E.89 :
*
*
*

* E.91 : UNRECOGNISED PARAMETER/VARIABLE
*
*
*

* E.92 : – ERROR IN ASSIGNING A PARAMETER/VARIABLE
* – AN UNSIGNED FUNCTION ASSIGNED A NEGATIVE PARAMETER?
* THE PARAMETER VALUE EXCEEDS THE ASSIGNED FUNCTION LIMIT?
*

* E.93 : ERROR IN PARAMETER DEFINITION OR IN THE EXPRESSION OF A TEST
* – RELATIONAL SYMBOLS "=, <, >" FOLLOWING AN L VARIABLE MISSING?
* – ONE OF THE CHARACTERS "–, +, *, /" UNRELATED TO ITS FUNCTION?
* – UNRECOGNISED PARAMETER?

* E.94 : IMPOSSIBLE OPERATION IN AN EXPRESSION :
* – SQUARE ROOT OF A NEGATIVE NUMBER?
* – DIVISION BY 0?
*

* E.95 : ATTEMPT TO WRITE TO A READ ONLY PARAMETER
*
*
*

* E.96 : AN EXTERNAL PARAMETER DEFINITION (WHICH SUSPENDS BLOCK PROCESSING)
* – CONFLICTS WITH THE LOOK AHEAD NEEDS OF THE PRECEDING BLOCK (S)
*
*

* E.97 : PARAMETER EDITING IMPOSSIBLE IN G76 :
* – NO "=" SYMBOL AFTER THE PARAMETER NUMBER
* – LESS THAN 10 CHARACTERS ALLOCATED FOR WRITING A VALUE
*

* E.98 :
*
*
*

* E.99 :
*
*
*

* ---   Errors generated from PGP profile definitions : -

* ---   From blocks in which the end point should be directly determined

* E101 : INSUFFICIENT DATA IN THE PROGRAMMING OF A CIRCLE
*
*
*

* E102 : WHEN PROGRAMMING A LINE USING ITS ANGLE (EA) AND ONE
* COORDINATE (X, Y OR Z) THE MISSING COORDINATE IS
* IMPOSSIBLE TO CALCULATE
*

* E103 :
*
*
*

* E104 :
*
*
*

* E105 :
*
*
*

* E106 : IN G2–G3, THIRD AXIS PROGRAMMED WITHOUT HELICAL OPTION
*
*
*

* E107 : – PROGRAMMING A CIRCLE USING ITS RADIUS AND END POINT, IN
* WHICH THE DISTANCE BETWEEN THE START AND END POINTS
* IS MORE THAN TWICE THE RADIUS.
* – OR, PROGRAMMING A CIRCLE USING X, Z, I, K, IN WHICH THE
* TRAJECTORY MISSES THE END POINT BY MORE THAN 20 MICRONS.
* – OR, A MISSING AXIS DIMENSION IN HELICAL INTERPOLATION

* E108 :
*
*
*

* E109 :
*
*
*

* --- Errors from profiles whose end points should be resolved over two blocks.

* E110 : SYNTAX ERROR IN THE FIRST OF THE TWO BLOCKS
*
*
*

* E111 : SYNTAX ERROR IN THE SECOND BLOCK
*
*
*

* E112 : LINE-LINE INTERSECTION IN WHICH :
* - START POINT OF 1ST BLOCK IS COINCIDENT WITH END
* POINT OF 2ND BLOCK
* - OR ANGLE OF 1ST LINE = ANGLE OF 2ND LINE

* E113 : THE INTERSECTION OR TANGENCY POINT CANNOT BE CALCULATED
* USING THE VALUES PROGRAMMED IN THE TWO BLOCKS
*
*

* E114 : THE POINT OF INTERSECTION OR TANGENCY IS NOT
* DEFINED BY ET+, ET-, ES+ OR ES-
*
*

* E115 :
*
*
*

* E116 :
*
*
*

* E117 :
*
*
*

* E118 :
*
*
*

* E119 :
*
*
*

* ---		Errors from profiles whose end points should be resolved over three blocks

* E121	:	SYNTAX ERROR IN LAST 0F 3 BLOCKS
*
*
*

* E122	:	1ST TWO BLOCKS ARE NON-SECANT STRAIGHT LINES
*
*
*

* E123	:	THE POINTS OF TANGENCY CANNOT BE CALCULATED FROM THE DATA
*		IN THE 3 BLOCKS
*
*

* E124	:	THE POINT OF TANGENCY BETWEEN THE 2ND AND 3RD BLOCKS
*		IS NOT DEFINED BY ET+ OR ET-
*
*

* E125	:
*
*
*

* E126	:
*
*
*

* E127	:
*
*
*

* E128	:
*
*
*

* E129	:
*
*
*

* ---	Errors in the definition of a blend radius or chamfer.

* E130 : NO AXIS MOVEMENT IN THE FIRST OF THE TWO BLOCKS
TO BE CONNECTED BY A FILLET RADIUS OR A CHAMFER
*
*

* E131 : A FILLET OR CHAMFER MUST NOT BE PROGRAMMED
* IN A BLOCK WITH M00 or M01 or M02
* - OR THE END POINT CANNOT BE CALCULATED FROM THE GIVEN DATA BLOCKS

* E132 :
*
*
*

* E133 :
*
*
*

* E134 :
*
*
*

* E135 : A CHAMFER CAN ONLY CONNECT TWO STRAIGHT LINES
*
*
*

* E136 : MORE THAN TWO NON-MOVEMENT BLOCKS BETWEEN TWO GEOMETRIC
* ELEMENTS WHOSE POINT OF INTERSECTION OR TANGENCY IS TO BE
* CALCULATED
*

* E137 : A CHANGE OF INTERPOLATION PLANE IS NOT ALLOWED HERE
*
*
*

* E138 : A CHANGE OF INTERPOLATION PLANE MUST BE IN G40
*
*
*

* E139 :
*
*
*

* E140 : ERROR IN RADIUS COMPENSATION MODE :
* – TOO MANY BLOCKS BETWEEN TWO COMPENSATED MOVES?
* – THE FOLLOWING FUNCTIONS ARE NOT ALLOWED IN G41/2 :
* M00, M01, M02 ; ACCESS TO EXTERNAL PARAMETERS ; WRITING
* TO E8xxxx OR L1xx PARAMETERS
* – MOVE DEMANDED BEYOND MACHINE TRAVEL LIMITS?

* E141 : .
*
*
*

* E142 :
*
*
*

* E143 : SCALING FACTOR ENABLED OR DISABLED DURING RADIUS CORRECTION
*
*
*

* E144 :
*
*
*

* E145 :
*
*
*

* E146 :
*
*
*

* E147 :
*
*
*

* E148 : MORE THAN TWO PROGRAM AXES WITH OPTION 51 TO 1
*
*
*

* E149 : TOOL CORRECTION EXCESSIVE WITH RESPECT TO THE PROGRAMMED PATH
*
*
*

* ---   Move demanded beyond a slide capacity

* E150 : DEMAND EXCEEDS TRAVEL LIMIT ON AXIS X
*
*
*

* E151 : DEMAND EXCEEDS TRAVEL LIMIT ON AXIS Y
*
*
*

* E152 : DEMAND EXCEEDS TRAVEL LIMIT ON AXIS Z
*
*
*

* E153 : DEMAND EXCEEDS TRAVEL LIMIT ON AXIS U
*
*
*

* E154 : DEMAND EXCEEDS TRAVEL LIMIT ON AXIS V
*
*
*

* E155 : DEMAND EXCEEDS TRAVEL LIMIT ON AXIS W
*
*
*

* E156 : DEMAND EXCEEDS TRAVEL LIMIT ON AXIS A
*
*
*

* E157 : DEMAND EXCEEDS TRAVEL LIMIT ON AXIS B
*
*
*

* E158 : DEMAND EXCEEDS TRAVEL LIMIT ON AXIS C
*
*
*

* E159 : MOVE DEMANDED ON A NON-REFERENCED AXIS
*
*
*

* ---	Structured programming errors

* E190 : TOO MANY BRANCH OR LOOP NESTING LEVELS (MAXIMUM 15)
*
*
*

* E191 : NON-COMPLIANCE WITH THE SYNTAX IN STRUCTURED PROGRAMMING

* STRUCTURED PROGRAMMING PROHIBITED IN MDI MODE

* THE INDEX OF A FOR LOOP MUST BE : AN L VARIABLE, A SYMBOLIC VARIABLE
* OR A PARAMETER E80000, E81000 OR E82000
* NON-COMPLIANCE WITH THE SYNTAX OF PUSH AND PULL INSTRUCTIONS POINT
PROGRAMMING OF IF, THEN, ELSE IN MDI
*
*

* E192 : KEYWORD NOT RECOGNISED OR PROHIBITED IN THE CONTEXT
*
*
*

* E193 : STRUCTURE ERROR
*
*
*

* E195 : PROGRAMME STACK SATURATED
*
* TOO MANY CONSTANTS DEFINED FOR THE SPACE ALLOCATED
*

* E196 : ERROR IN ARRAY INDEX DECLARATION
*
*
*

* E197 : USE OF A SYMBOL NOT DECLARED AS VAR
*
* PULL WITHOUT PUSH
*

* E198 : SYNTAX ERROR IN VARIABLE SYMBOL DECLARATION
*
*
*

* E199 : INCORRECT VARIABLE DECLARATION SYNTAX
*
*
*

* ---   Poor or faulty encoder signals

* E210 : POOR OR FAULTY POSITION SIGNALS ON AXIS 0
*
*
*

* E211 : POOR OR FAULTY POSITION SIGNALS ON AXIS 1
*
*
*

* E212 : POOR OR FAULTY POSITION SIGNALS ON AXIS 2
*
*
*

* E213 : POOR OR FAULTY POSITION SIGNALS ON AXIS 3
*
*
*

* E214 : POOR OR FAULTY POSITION SIGNALS ON AXIS 4
*
*
*

* E215 : POOR OR FAULTY POSITION SIGNALS ON AXIS 5
*
*
*

* E216 : POOR OR FAULTY POSITION SIGNALS ON AXIS 6
*
*
*

* E217 : POOR OR FAULTY POSITION SIGNALS ON AXIS 7
*
*
*

* E218 : POOR OR FAULTY POSITION SIGNALS ON AXIS 8
*
*
*

* E219 :
*
*
*

* ---   Spline curve programming errors

* E250 : SYNTAX ERROR
* IN G48 AND G49, PROGRAMMING OF OTHER PREPARATORY FUNCTIONS
* OR MISCELLANEOUS FUNCTIONS PROHIBITED
*

* E251 : CHARACTER NOT RECOGNIZED
*
*
*

* E252 : FORMAT ERROR. THE FORMAT FOR DIMENSIONS IS 5.3 OR 4.4 DEPENDING ON WHICH
* OF FUNCTIONS G70 OR G71 IS ACTIVE
*
*

* E253 : PROGRAMMING INCOHERENT
* ALL THE INTERPOLATED AXES ON THE CURVE MUST BE INCLUDED
* IN ALL THE DEFINITION BLOCKS
*

* E254 : DEFINITION MEMORY SPACE SATURATED
*
*
*

* E255 : PROGRAMMING OF TWO IDENTICAL CONSECUTIVE POINTS
* (ZERO MOVEMENT BETWEEN POINTS)
* SPLINE CURVES MUST BE DEFINED BY AT LEAST THREE POINTS
*

* E256 : CURVE NOT FOUND IN G06
*
*
*

* E257 :
*
*
*

* E258 :
*
*
*

* E259 :
*
*
*

* ---   Errors in pocket cycles (E260 to E292)

* E.260   :   WORKING MEMORY OCCUPIED
*
*
*

* E.261   :   PROGRAM NUMBER TOO HIGH
*
*
*

* E.262   :   NU NUMBER NOT INCLUDED AMONG THOSE AUTHORIZED
*
*
*

* E.263   :   EXECUTION IMPOSSIBLE. TEST OR GRAPHIC MODE MANDATORY AFTER THE
*   FIRST LOAD OR AFTER EDITING
*
*

* E.264   :   NO DIMENSION PROGRAMMED IN THE CONTOURING PLANE OR DIMENSION
*   OUTSIDE THE PLANE
*
*

* E.265   :   FIRST POSITIONING BLOCK MISSING. THE CONTOUR DEFINITION MUST
*   BEGIN WITH G0 OR G1
*
*

* E.266   :   NOT ENOUGH MEMORY SPACE
*
*
*

* E.267   :   CHARACTER NOT ALLOWED IN POCKET SYNTAX
*
*
*

* E.268   :   POCKET PROGRAMMING BLOCK INCOMPLETE
*   POCKET PROGRAMMING BLOCK CONTAINING
*   PROHIBITED DATA
*

* E.269   :   CONTOUR BLOCK INCOMPLETE
*   POSITIONING BLOCK MISSING BEFORE POCKET DEFINITION
*
*

* E.270 : POCKET DEFINITION PARTIALLY OR COMPLETELY MISSING
*
*
*

* E.271 : TOOL DIRECTION NOT PERPENDICULAR TO THE CONTOURING PLANE
*
*
*

* E.272 : REAL TOOL NOT COMPATIBLE WITH POCKET ENGINEERING DATA
*
*
*

* E.273 : CHANGE OF CONTOURING PLANE BETWEEN POCKET DEFINITION AND
* MACHINING
*
*

* E.274 : TWO NESTED POCKET DEFINITIONS
*
*
*

* E.275 : NU0 PROGRAMMED WITH G59
*
*
*

* E.276 : ZERO POCKET DEPTH
* ERROR IN PROGRAMMING DIMENSION ER
*
*

* E.277 : START POINT OR END POINT COORDINATES
* INCOMPLETE IN THE POCKET DEFINITION
*
*

* E.278 : SPINDLE ROTATION DIRECTION INCOMPATIBLE WITH THAT REQUIRED
* BY POCKET DEFINITION
*
*

* E.279 : G FUNCTION NOT ALLOWED IN A POCKET PROGRAMMING BLOCK
*
*
*

* E.280 : FIRST CONTOUR BLOCK INCOMPLETE
*
*
*

* E.281 : DISCONTINUITY IN ONE OF THE PROFILES DESCRIBED
*
*
*

* E.282 : POCKET DEFINITION PARAMETER ERROR
*
*
*

* E.283 : THE EXTERNAL PROFILE MUST BE UNIQUE AND MUST EXIST
*
*
*

* E.284 : PROFILE DEFINITION ERROR
*
*
*

* E.285 : TOO MANY CONTOURS
*
*
*

* E.286 : WIDTH OF CUT EXCESSIVE WITH RESPECT TO TOOL DIAMETER
*
*
*

* E.287 : WIDTH OF CUT INSUFFICIENT WITH RESPECT TO DIMENSIONS
*
*
*

* E.288 : FINISHING STARTED IN AN ACUTE ANGLE
* FINISHING STARTED IN AN UNROUGHED REGION
* (CHANGE THE START POINT)
*

* E.289 : TOOL DIAMETER TOO LARGE
*
*
*

* E.290 : INTERNAL ERROR (SOFTWARE)
*
*
*

* E.291 : FINISHING STARTED OUTSIDE THE PROFILE
*
*
*

* E.292 : DOUBLE POSITIONING AT THE START OF THE PROFILE
*
*
*

* E.293 : PROGRAMMING OF A ROUGHING START OR END POINT FOR A FACING
*
*
*

* E.294 :
*
*
*

* E.295 :
*
*
*

* E.296 :
*
*
*

* E.297 :
*
*
*

* E.298 :
*
*
*

* E.299 :
*
*
*

NOTES

NUM in the Great Britain and in the United States

List updated 15-12-92

		Address	Telephone Telex (Tx)	Fax
Great Britain	□ ■	NUM SERVOMAC UK Ltd - Coventry	(203) 69.25.25 Tx : 317 208	(203) 69.30.34
United States	□ ■	NUM Corp. - Naperville (IL.)	(708) 505.77.22	(708) 505.77.54

NUM in the world

Head Office				
France	□ ■	NUM SA 21, Avenue du Maréchal Foch BP 68 - 95101 Argenteuil Cedex	(1) 34.23.66.66 Tx : 609 611	(1) 34.23.65.49
Subsidiaries				
Germany	□ ■	NUM-GÜTTINGER GmbH - Stuttgart/Nellingen	(711) 34.80.60 Tx : 7256 712	(711) 348.06.10
	□	NUM-GÜTTINGER - Verkaufsbüro West-Mechernich	(2443) 59.59	(2443) 62.43
	□	NUM-GÜTTINGER - Verkaufsbüro Nord-Löhne 2	(5732) 820.07	(5732) 826.43
	□ ■	NUM-GÜTTINGER - Technisches Büro-Ratingen	(2102) 432-0	(2102) 432 190
	□ ■	NUM-GÜTTINGER - Technisches Büro-Chemnitz	(722) 64.31.47 Tx : 777 61, 777 63	(722) 64.31.50
Italy	□ ■	NUM Spa - Bologna/Zola Predosa (head office)	(51) 75.41.18	(51) 75.40.82
	□ ■	NUM Spa - Milano	(2) 27.00.09.63	(2) 27.00.15.68
		Servo-drives	(2) 27.00.25.23	(2) 27.00.25.18
	□ ■	NUM Spa - Torino/Rivoli	(11) 955.03.59	(11) 955.03.62
	□ ■	NUM Spa - Padova/Este	(429) 60.06.90	operator
Spain	□ ■	TELENUM SA - Itziar - Deba (Guipuzcoa)	(43) 19.92.55	(43) 19.91.05
	□ ■	TELENUM SA - Barcelona/Castelldefels	(3) 636.13.52	(3) 636.29.77
	□	TELEMECANICA ELECTRICA ESPAÑOLA SA Madrid	(1) 695.71.00 Tx : 22702	(1) 682.08.74
Sweden	□ ■	NUM NORDEN AB - Västerås	(21) 13.11.31	(21) 13.12.30
Switzerland	□ ■	NUM-GÜTTINGER AG - Teufen	(71) 33.04.11 Tx : 883 975	(71) 33.35.87
	□ ■	NUM-GÜTTINGER SA - Bienne	(32) 23.53.73 Tx : 34 333	(32) 23.55.25
Representatives				
Argentina	□ ■	FASTER SA - Buenos Aires	(1) 766.22.26 (1) 765.63.42	(1) 763.63.92
Belgium Luxembourg	□ ■	CARON VECTOR SA - Bruxelles	(10) 23.13.11 Tx : 59 509	(10) 23.13.36
CIS	■	KHARKOV TECHNICAL MAINTENANCE CENTRE - Kharkov/Ukraine	(0572) 23.64.40 (0572) 23.64.41 Tx : 115 147	

□ - Distribution

■ - After-sales Service

NUM in the world

List updated 15-12-92

Partners or Representatives		Address	Telephone Telex (Tx)	Fax
China	■	TELEMECANIQUE - Beijing	(1) 506.69.28 Tx : 210 301	(1) 506.69.18
Colombia	■	TELEMECANIQUE DE COLOMBIA - Bogota	(1) 413.91.81 Tx : 43 191	(1) 413.90.12
Finland	❐	NUCOS OY - Helsinki	(0) 294.35.90	(0) 294.75.90
	❐	NUCOS OY - Tampere	(31) 379.09.29	(31) 379.08.37
Hong-Kong **Philippines**	■	TELEMECANIQUE ASIA PACIFIC Ltd - Hong-Kong	(5) 65.06.21 Tx : 865 08	(8) 11.10.29
India	■	TELEMECANIQUE INDIA LIAISON OFFICE New Delhi	(11) 646.76.58 (11) 644.10.22 Tx : 031 712 82	(11) 646.50.80
Japan	■	TELEMECANIQUE JAPAN Ltd - Tokyo	(3) 35.88.86.71	(3) 35.85.66.93
Korea	■	TELKO Ltd - Seoul	(2) 681.14.51	(2) 682.81.07
Malaysia **Singapore**	❐ ■	INDUSTRIAL MAINTENANCE ENGINEERING Selangor	(3) 775.24.86 (3) 774.94.61	(3) 777.20.95
Mexico	■	TELEMECANIQUE MEXICO SA - Naucalpan/Mexico	(5) 300.15.76 Tx : 176 13 79	(5) 300.08.71 operator
South Africa	■	MACHINE TOOL TECHNOLOGIES - Isando	(11) 823.17.55 (11) 823.17.56	(11) 823.17.05
Taiwan	❐ ■	NUM TELEMECANIQUE TAIWAN - Taichung	(4) 328.20.81 (4) 328.20.82	(4) 328.20.83
Tunisia	■	SIME - Tunis	(1) 78.57.13	(1) 78.95.88
Turkey	❐ ■	TELEMECANIQUE Elektrik - Istanbul	(1) 558.78.82	(1) 547.13.56

❐ - Distribution

■ - After-sales Service