

Chapter 2

Interpreter

The Numerical Control Kernel (NCK) unit is the key component of a CNC system and consists of a variety of modules that are sequentially executed in a synchronized schedule. In this chapter the code interpreter will be addressed. This is responsible for converting the part program and machine instructions into internal commands for NC. In order to understand the code interpreter the first thing is to understand the part program that is the input to the interpreter. After this, the structure and the functions of the code interpreter will be addressed in detail.

2.1 Introduction

The code interpreter is a software module, which translates the part program into internal commands for moving tools and executing auxiliary functions in a CNC system.

Figure 2.1 depicts the internal behavior of the CNC system and shows the functions of the Man-Machine Control (MMC), Numerical Control Kernel (NCK), and Drives (DRV). The part program that a programmer generates based on the shape of the part, cutting conditions, and tools is entered into CNC via the MMC and the NCK subsequently generates the control commands for the drivers from the part program through various stages; calculating the movement path by interpreting the part program, generating velocity profile and displacement for each axis by interpolation, smoothing the movement by acceleration/deceleration (acc/dec) control, and generating position control command.

Among these stages, the interpreter could be considered as a simple task for the conversion of G/M codes to the CNC-understandable internal data structures. However, the design and implementation of the interpreter is a large and comprehensive task because programming rules or grammar described in a programming manual and an operating concept shown in an operation manual should be considered when developing the interpreter. Therefore, the interpreter is the representative indicator that shows the design concept and the functional aspect of a CNC and is a big part of

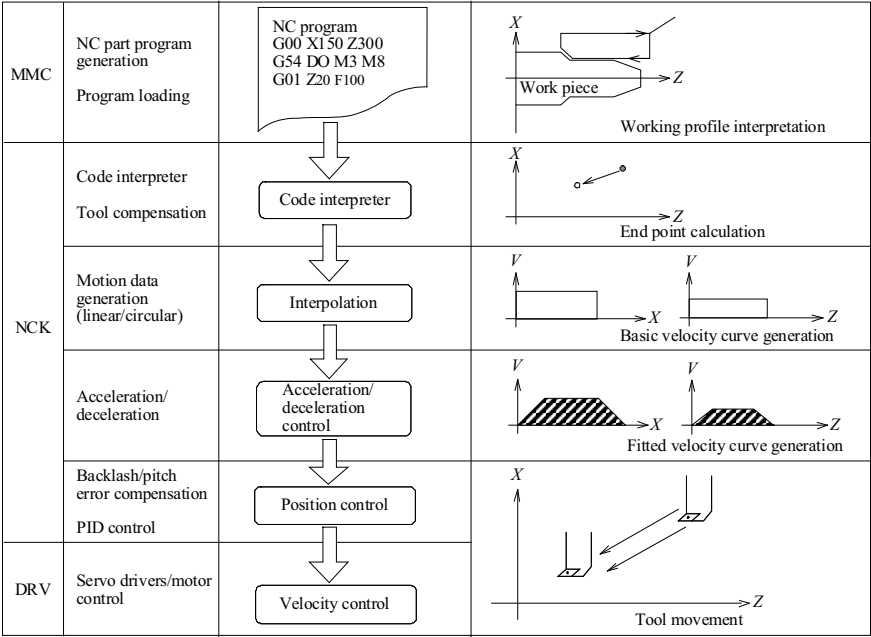


Fig. 2.1 Internal behavior of the CNC system (MMC, NCK, DRV functions)

CNC as it generally spends more than 50% of the total development time to develop the interpreter.

In this chapter, the format and function of CNC part programs will be briefly addressed and the architecture, such as the structure of an interpreter, execution procedure, and memory structure, will be addressed. However, because the detailed function of the CNC and the part program are slightly different for each CNC maker, the program manual should be referenced to find out the detailed functions of any particular CNC.

2.2 Part Program

Although the standard exists for generating a CNC part program, sequentially listing the commands for executing CNC, each CNC maker has, in practice, their own code system including their own commands. In this section, the common concept will be described based on the standard code.

2.2.1 Program Structure

A part program contains the commands, called blocks, for machining a part and each block can be defined using the following commands.

- NC commands such as G, M, S, T, H, D, F code and related address
- Call of sub program and displaying message
- Setting variable and conditional program calls

In a part program, the English alphabet, Arabic numbers, and symbols are used and Fig. 2.2 illustrates the format and elements of a part program.

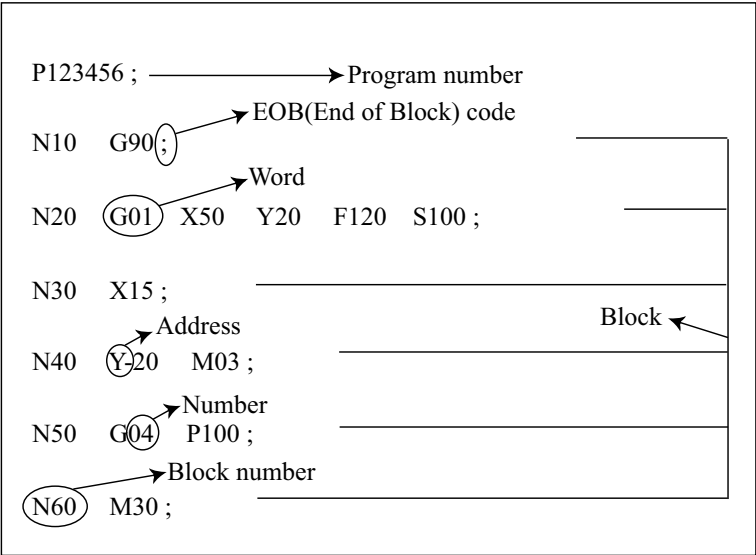
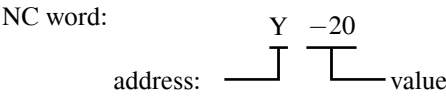


Fig. 2.2 Program format and construction elements

A part program consists of a sequence of NC blocks, each block consists of several words, and a word is composed of an address and number. The program number is a number for identifying the particular part program on CNC, where more than one part program is executed, and is written using a particular address and number in the heading of a part program. In this book, address P is used but O or # is also used by some specific CNC makers.

A block consists of one block number, at least one word, and the EOB, meaning the End Of Block. The word is the set of characters in a specific order. The word is the minimum unit for internal processing and commanding the machine tools to perform a particular behavior. The word consists of an address and a subsequent number, as below:



The address is constructed from one of the alphabetic characters (A Z) or a combination of alphabetic characters. The subsequent number provides the data that is required to execute the behavior related with the address. Table 2.1 summarizes the addresses that have typically been used and the function that is related with the address.

Table 2.1 Typical addresses and associated functions

| Function | Address | Meaning (Example) | Unit |
|---|----------------------|---|---------------------------------|
| Program number | P | Program identity no. <i>e.g.</i> P123456 | |
| Block number | N | NC seq.no. N100 | |
| Preparatory function | G | Mode command G01 | |
| Coordinate (command for translational axis) | X, Y, Z / U, V, W | Axis / dir. X100 W20 | mm, inch |
| Coordinate (command for rotary axis) | A, B, C | Axis A30 | deg |
| Feedrate | F | Feedrate per min. F200 | mm/min, inch/min, deg/min |
| | | Feedrate per rev. F1 | mm/rev, inch/rev, deg/rev |
| Spindle speed | S | S3000 | rpm |
| Tool | T | Tool number T12 | |
| Auxiliary function | M | Machine command M06 | |
| Offset Number | H, D | Offset register no. H10 | |
| Number of repetitions of subprogram | L | Iteration no. L5 | |
| Radius of circle or arc | R | Arc rad., corner rad. R3 | mm, inch |
| Chamfer | C | Chamfer amount C2 | mm, inch |
| Center position of circle | I, J, K | Circle center coords. | mm, inch |

Among the addresses described in Table 2.1, the G address, for preparatory function, and the M address, for auxiliary function, are largely related to the performance of CNC system. G addresses denote commands for tool movement by moving the

translational axes or the rotary axes along the specified path. M addresses denote commands for controlling the on/off functions in machine tools. G-codes are classified into two types: one is a modal code and the other is non-modal code. A modal-type code is effective throughout the following blocks until the modal cancel command is used. On the other hand, a non-modal-type code is effective within the commanded block and automatically canceled by the next block. Modal-type codes are classified into several groups, called modal groups, with respect to the similarity of function. In one block, it is prohibited to use more than one G-code that is included in the same modal group.

The address groups based on the functions of CNC system are summarized in Table 2.2. As the standard for editing a part program based on these addresses, ISO 6983 has been widely used. However, each CNC maker has their own G&M code system where maker-specific functions have been added to ISO 6983. Accordingly, current G&M code systems for generating part programs depend on the CNC system. If the CNC system is changed, it is almost impossible to reuse the existing part program. Therefore, in order to create a part program manually, it is necessary to refer to the programming manual of the particular CNC maker.

According to the level of CNC system, the number of feasible addresses varies from several tens to several hundreds. This means that the more feasible addresses a CNC system has, so the more advanced the equipment category to which the CNC system belongs. Further, according to the machine type, the applicable addresses are defined in different ways. The G-code list and the modal group for a milling machine and a turning machine are summarized in Appendix A. In the following, the interpreter is the module that has the function of interpreting the various addresses, words, and grammar.

Table 2.2 Summary of address groups based on CNC system functions

| Functions | Description |
|-------------------------------|---|
| Preparatory function (G-code) | <p>Codes are used for controlling axis and the CNC system prepares the execution of the particular function.</p> <p>(1) movement: command the relative movement between tool and workpiece.</p> <p>(2) Setting local coordinate system: specify the origin and orientation of local coordinate system including orthogonal coordinate system, the polar coordinate system, and rotational coordinate system.</p> <p>(3) Interpolation: command machining various profiles such as line, arc, helical, spline, etc.</p> <p>(4) Miscellaneous: commands for tool compensation, safety check, and skip.</p> |
| Feed function (F-code) | Command the relative speed between tool and workpiece for interpolation command. |
| Spindle functions (S code) | Command the spindle speed (RPM). |
| Tool function (T-code) | Command tool change and specify the tool compensation. |
| Auxiliary function (M-code) | <p>Function commands for the simple control of a machine tool including relay/switch on and off as well as axis control.</p> <p>(1) Coolant: command coolant on or off.</p> <p>(2) Start/End of a main and sub program: inform the beginning and end of a part program.</p> <p>(3) Spindle: command a spindle to rotate in CCW or CW direction and specify the spindle speed, limit speed and spindle gear change.</p> <p>(4) Miscellaneous: Command the change of tool, workpiece, pallet and rotation of the table/loader.</p> |
| Utility functions | <p>(1) Subprogram: To simplify the main program, subsidiary program</p> <p>(2) MACRO register a series of commands as a particular command and execute various functions by using the registered command. In a macro, the use of variables is possible and arithmetic operations between variables is possible.</p> <p>(3) Fixed cycle</p> <p>(i) Turning cycle Define a series of commands to perform the turning operations as one command. Outer turning, facing, grooving, and threading cycle are defined.</p> <p>(ii) Drilling cycle Define a series of commands to perform drilling, tapping, boring, reaming and peck drilling as drilling cycle.</p> |

Table 2.2 (continued)

| | |
|--|---|
| | <div>(iii) Milling cycle Pocket milling, Slot milling. Define a series of commands to machine profiles that are frequently machined during pocket milling and slotting as milling cycles.</div> <div>(iv) Touch Probe cycle This command enables the modification of a program via on-machine inspection before or after machining. This enables compensation of finishing with inspection of the machining accuracy.</div> |
|--|---|

It is possible to edit the description by inserting words in parentheses within a block, as below.

N20 G01X0Y0 (MOVE TO ZERO POINT);

comment

The description comment has no influence on the execution of a part program. Because the description can be shown on the display of the CNC system together with the block during editing or executing a part program, it is very useful for managing part programs.

The end of a part program is signalled by the command M02 or M03. By inserting M02 or M03 at the end of a part program, all modal values are initialized and reset. Since the commands M02 and M03 are executed last, they can be located anywhere within the last block.

2.2.2 Main Programs and Subprograms

2.2.2.1 Main program

A part program is classified into a main program and subprograms. Typically, the CNC system executes a main program. If a main program includes the command that is used for calling subprograms, the CNC system executes the subprogram indicated. If, during execution of the subprogram, the command for returning to the main program is called, the main program is then resumed at the block after the command that called the subprogram, as shown in Fig. 2.3.

2.2.2.2 Subprogram

In the case that there are fixed routine blocks or iterated operation patterns in a part program, part programming can be made easier if they are stored as a subprogram in the internal memory of CNC system. It is possible to call the subprogram from a

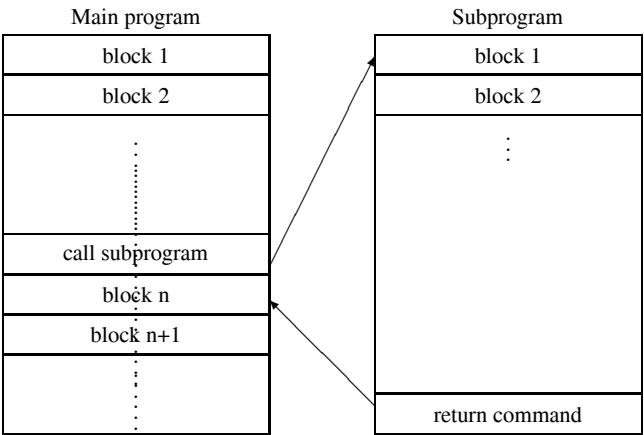


Fig. 2.3 Main program and subprogram execution

main program during auto mode of a CNC system. It is also possible to call another subprogram from within a subprogram.

2.3 Main CNC System Functions

The main functions of a CNC system can be classified into a variety of groups such as coordinating functions, interpolation functions, compensation functions, safety functions, and auxiliary utility functions. These will be described in the following sections.

2.3.1 Coordinate Systems

In CNC systems, a machine coordinate system, a workpiece coordinate system, and local coordinate systems are defined for convenience when editing a part program and handling machine tools.

A machine coordinate system is defined by setting a particular point of the machine tool as the origin of a coordinate system. A workpiece coordinate system is defined by setting a particular point on the workpiece as the origin so as to make editing a part program easier. That is, when editing a part program using one particular workpiece coordinate system, we can edit the part program by defining another coordinate system based on the workpiece coordinate system. We call this secondary coordinate system a “local coordinate system”. A workpiece coordinate system is set by commanding particular G-codes (G54 to G59) and a local coordinate system is defined by setting an offset (IP) that denotes the displacement of the local coordinate

system from the origin of the workpiece coordinate system. Based on the origin of the machine coordinate system, the relationship between each workpiece coordinate and local coordinate system is illustrated by Fig. 2.4.

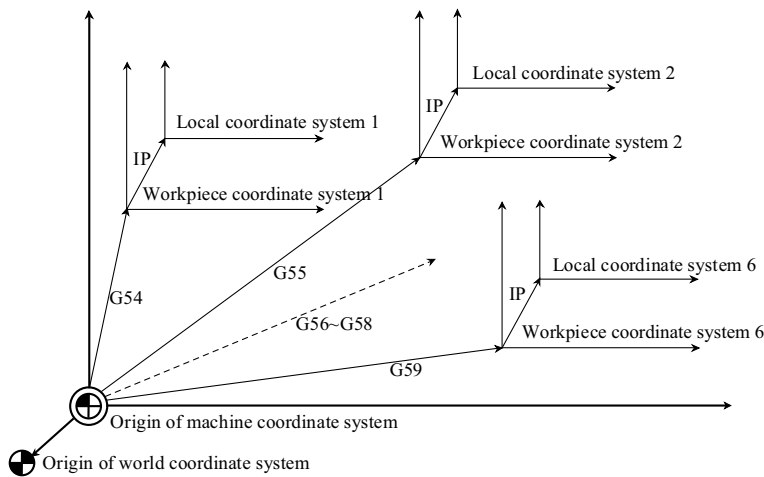


Fig. 2.4 Machine, workpiece and local coordinate systems

As methods to command displacements of each axis based on the specified coordinate system, there are two modes, absolute programming mode (G90) and incremental programming mode (G91). When absolute programming mode is used, the end position of each axis is programmed. When incremental programming mode is used, the relative displacement of each axis is programmed.

Besides orthogonal coordinate systems, it is also possible to use polar coordinate systems (G15) where a radius component and angle components are used. Figure 2.5 shows a part program using the polar coordinate system and the path that is commanded by the part program. To use the polar coordinate system, first a work plane is selected and then a polar coordinate system is invoked by issuing the command G15. Thereafter, when using address *X* and address *Y*, a radius and an angle, respectively, are commanded.

For part programming, it is possible to use the scaling function and the rotation function based on the specific coordinate system. The scaling function is used for scaling down or up the programmed workpiece shape. To command the scaling function you use the G51 *X_ Y_ Z_ P_* format in a block, wherein the *X*, *Y*, *Z* address denotes the center position for scaling and is given as an absolute value. The *P* address is used for the magnitude of the scaling. As G51 is a modal G-code, the toolpaths in the following blocks are scaled *P* times up or down with respect to the point determined by the values above *X*, *Y* and *Z*.

To rotate the specific shape in a part program, the G68 α β *R* format is utilized wherein α and β denote the center position for rotation and *R* means a rotational angle (+*R* denotes CCW and -*R* denotes CW). Accordingly, after declaring G68 in

```
N0100 G17 G90 G15 ; XY plane, absolute coordinate, polar coordinate start
N0200 G00 X100 Y30 ; rad. 100mm, ang. 30deg
N0201 X100 Y150 ; rad. 100mm, ang. 150deg
N0202 X100 Y270 ; rad. 100mm, ang. 270deg
N0203 G16 ; Polar coordinates cancel
```

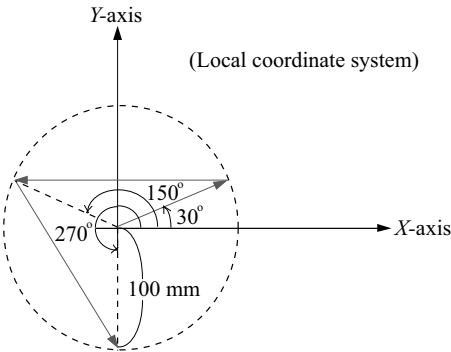


Fig. 2.5 Polar coordinate system programming

the block, the toolpaths in subsequent blocks are rotated by the angle R with respect to the point α, β .

If a workpiece is symmetric with respect to a specific axis, only part of the work-piece need be programmed, the other parts are created using the G51.1 address that utilizes a mirror image function. Figure 2.6 shows an example of usage of the mirror function. The subprogram below is for the path in the upper right side of Fig. 2.6 and the main program below commands the whole path with mirroring of the sub-program.

The subprogram makes the shape on the upper right. This is invoked in the original coordinate system in line N20 of the main program. The following command, on line N30 invokes the mirror function about the symmetry axis X=50. Line N40 then makes the symmetric shape on the top left. Following this, on line N50 the mirror function is again invoked to make the Y=50 symmetric axis. The next line, N60, then calls the subprogram to make the shape on the bottom left. On line N70 the X symmetry axis is revoked using the G50.1 command and the call of the subprogram on line N80 makes the shape at the bottom right. Finally, on line N90 the Y symmetry command is revoked.

2.3.2 Interpolation Functions

There are various interpolation functions that enable machine tools to move the axes along the specific path for multi-axis machine tools. A CNC system provides rapid movement, linear interpolation, circular interpolation, helical interpolation, and spline interpolation functions as interpolation functions.

Main Program

```

P000001;
N10 G00 G90;
N30 G51.1 X50; → sym. X=50
N40 M98 P100001;

N50 G51.1 Y50; → sym. X=50, Y=50
N60 M98 P100001;

```

Sub program

```

P100001;
N210 G00 G90 X60 Y60;
N230 Y100;
N240 X60 Y60;
N250 M99;

```

```

N70 G50.1 X0; → reset X
N80 M98 P100001;
N90 G50.1 Y0; → reset Y
N100 M30;

```

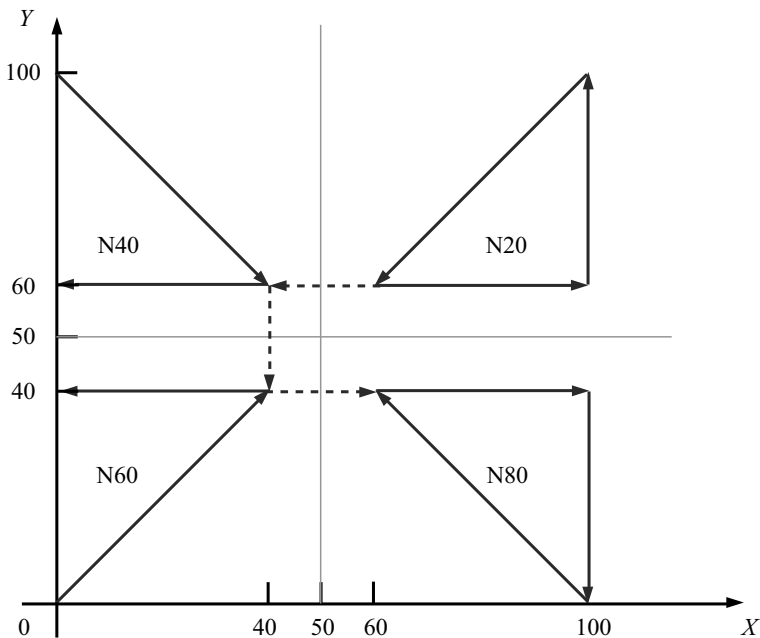


Fig. 2.6 Example of usage of mirror function

The rapid movement function (G00) is used for commanding the specific axes to move rapidly to the programmed position. In the case of an absolute programming mode (G90), this function makes the axes move to the commanded position from the current position. In the case of an incremental programming mode (G91), this function makes the axes move with the commanded incremental value and each axis moves with the specific feedrate defined in the CNC system. Therefore, it is not necessary to set an additional feedrate in G00.

The linear interpolation function (G01) is used for commanding the axes to move the tool along a line with the programmed feedrate, as shown in Fig. 2.7. G01 is a modal G-code and the commanded feedrate is effective until a new feedrate is commanded. Here, the feedrate means the joint speed of the axes.

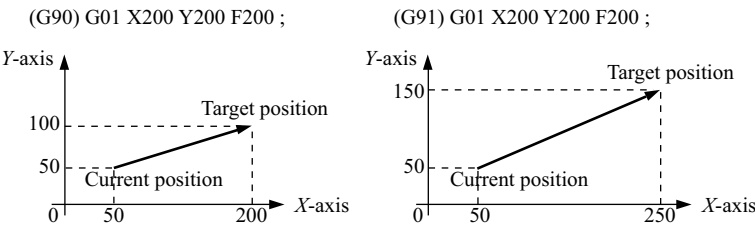


Fig. 2.7 Absolute (G90) and relative (G91) displacements

The circular interpolation function is used to command tool movement along a circle. G02 and G03 can be used for the circular interpolation function. G02 is for commanding circular interpolation in the clockwise direction and G03 is for commanding circular interpolation in a counter-clockwise direction. In order to command this function, the information summarized in Table 2.3 should be provided.

Table 2.3 Circular interpolation information summary

| No. | Information | | Command | Meaning |
|-----|---|----------|----------------------------|--|
| 1 | Plane | | G17 | Specification of arc on <i>XY</i> plane |
| | | | G18 | Specification of arc on <i>ZX</i> plane |
| | | | G19 | Specification of arc on <i>YZ</i> plane |
| 2 | Rotation direction | | G02 | Clockwise (CW) arc |
| | | | G03 | Counterclockwise (CCW) arc |
| 3 | End pos. | G90 Mode | Two in <i>X, Y, Z</i> axes | End position in workpiece coord. sys. |
| | | G91 Mode | Two in <i>X, Y, Z</i> axes | Distance from start to end |
| 4 | Distance between start point and the arc center | | Two in <i>I, J, K</i> axes | Distance from start to arc center (Sign value) |
| | Arc radius | | R | Radius of the arc |
| 5 | Feedrate along the arc | | F | Feedrate along the arc |

Normally, the rotation direction is defined based on the right-hand coordinate system. That is, if the programmed plane is the *XY* plane, then the CW or CCW directions are defined based on when the *XY* plane is viewed in the positive-to-negative direction of the *Z*-axis. Figure 2.8 shows the individual rotation directions in the cases where the programmed planes are the *XY*, *ZX*, and *YZ* planes.

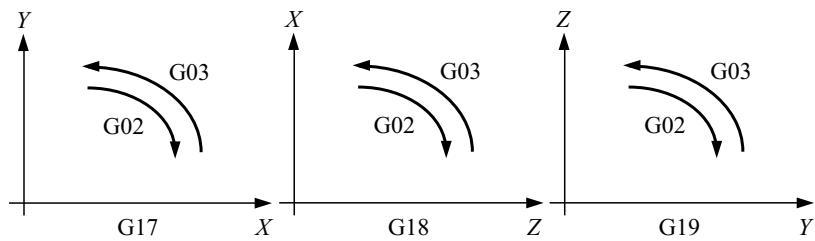


Fig. 2.8 CW and CCW directions for the XY, ZX and YZ planes

The end point of an arc is specified by the address X, Y, and Z, and is expressed as an absolute or incremental value according to whether G90 or G91 mode is current. For an incremental value, the distance of the end point that is viewed from the start point of the arc is specified by the sign value. The arc center is specified by addresses I, J, and K for the X, Y, and Z axes, respectively as shown in Fig. 2.9. The numerical value following I, J, or K, however, is a vector component in which the arc center is seen from the start point, and is always specified as an incremental value irrespective of G90 and G91 as shown below. I, J, and K must be signed according to the direction of the arc.

The arc center can also be specified by using radius R instead of addresses I, J, and K. In this case, there are two possibilities, where the arc is less than 180 degrees, or where it is more than 180 degrees. When an arc exceeding 180 degrees is commanded, the radius must be specified with a negative value. The feedrate in circular interpolation is equal to the feedrate specified by the F-code, and the feedrate along the arc (the tangential feedrate of the arc) is controlled by the specified feedrate.

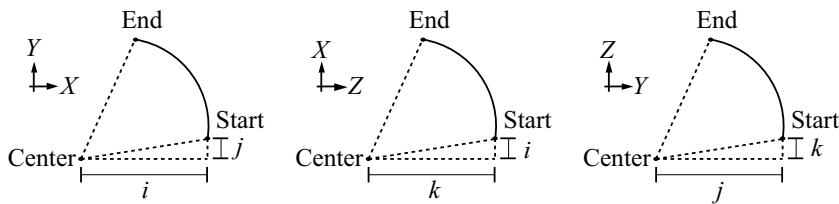


Fig. 2.9 Arc centers for the XY, ZX, and YZ planes

Figure 2.10 shows an actual programming example of circular interpolation in the case of G90 mode and G91 mode, respectively.

Helical interpolation is enabled by specifying up to two other axes that move synchronously with the circular interpolation by circular commands. The tangential feedrate of the arc is specified by an F-code and the feedrate of the linear axis to which circular interpolation is not applied is defined as follows:

$$\text{Feedrate of linear axis} = F * (\text{Length of linear axis})/(\text{Length of circular arc})$$

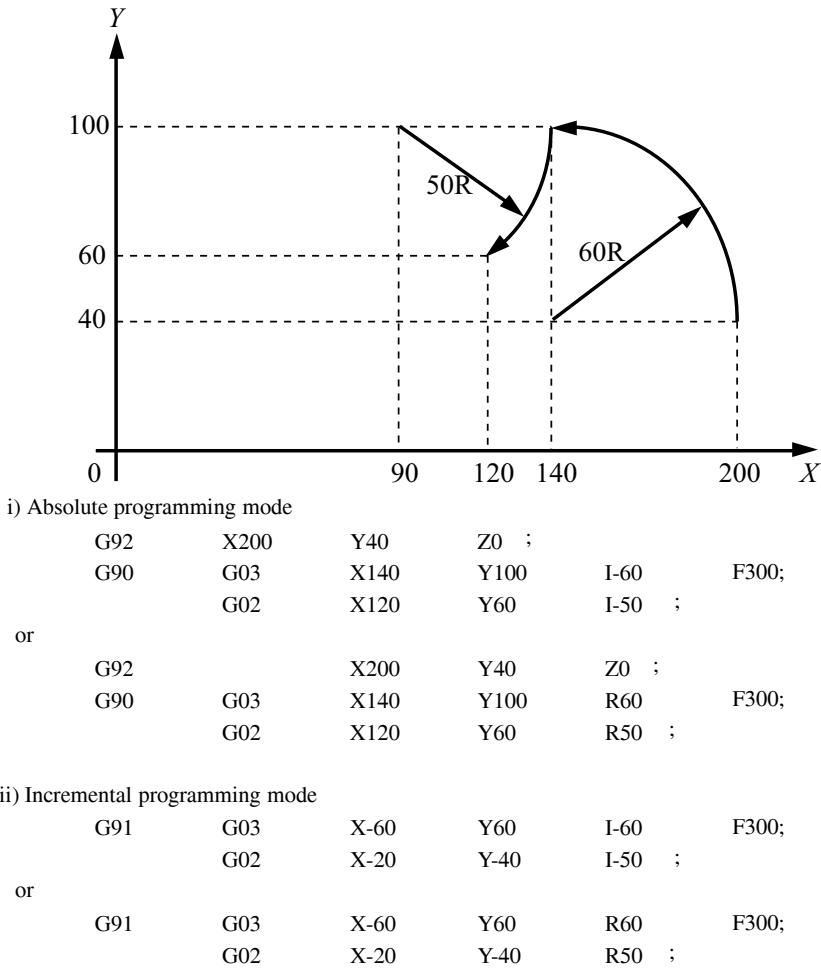


Fig. 2.10 Absolute and incremental circular interpolation

Cylindrical interpolation, which is useful for slotting and CAM machining on a cylinder, is a function where the amount of movement of the rotary axis, specified by an angle, is converted to the amount of movement on the circumference to allow linear interpolation and circular interpolation with another axis. For cylindrical interpolation, the development surface of the cylinder is regarded as a 2D shape and is programmed as shown in Fig. 2.12.

When machining the specified shape on a cylindrical surface, the use of the 2D developed surface on the C-axis and the Z-axis and cylindrical interpolation makes part programming easy.

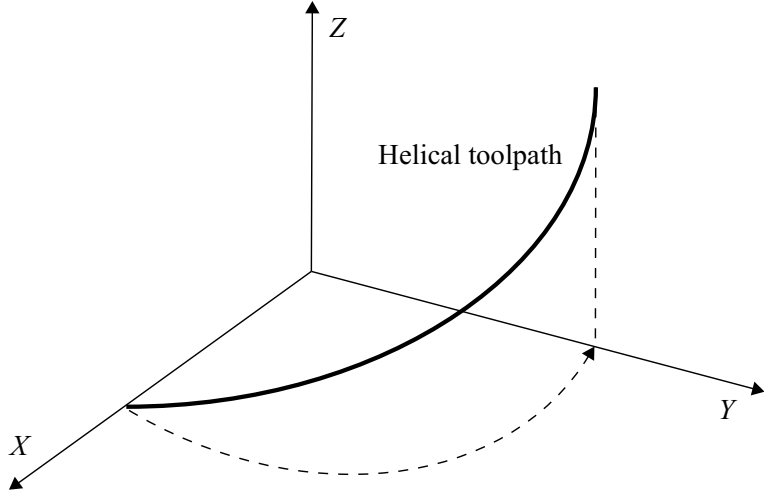


Fig. 2.11 Helical toolpath

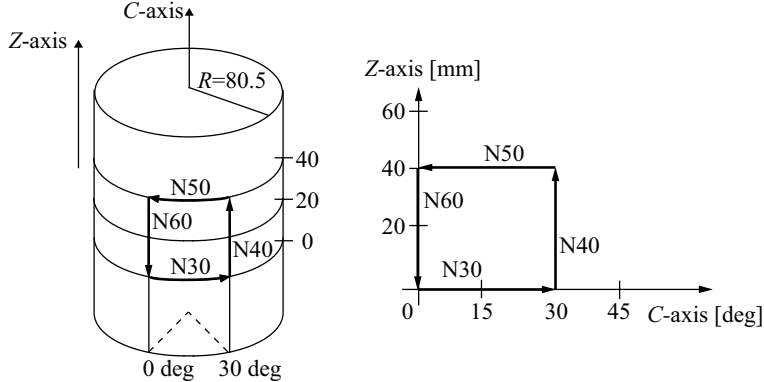


Fig. 2.12 Cylindrical interpolation

Spline interpolation (G06.1) is used for machining free-form curves or surfaces and enables the tool to be moved along the interpolated curve that passes through the specified points, as shown in Fig. 2.13. Spline interpolation is canceled by commanding another G-Code (*e.g.* G00, G01, G02, G03) that belongs to the same G-code group.

The typical type of spline interpolation is NURBS (Non Uniform Rational B-Spline) interpolation and the details of NURBS interpolation will be described in Section 3.5.

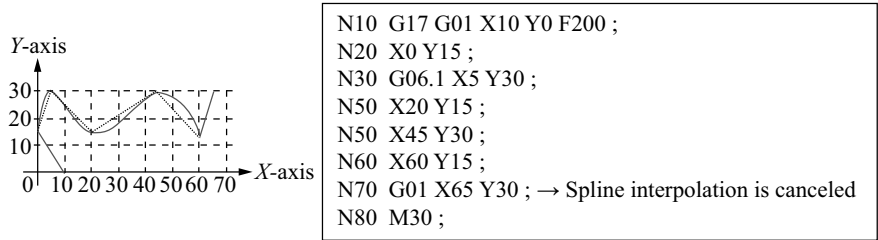


Fig. 2.13 Spline interpolation

2.3.3 Feed Function

The feed function is used for controlling the feedrate of axes and rapid movement, machining movement, path control mode (*e.g.* exact stop mode and continuous mode), and dwell function belong to this function. The feedrate, specified by the F-code, can be programmed as feed per min (mm/min or inch/min) or feed per revolution (mm/rev or inch/rev).

The rapid traverse function is used for moving the tool quickly to the commanded position and the feedrate for rapid movement is specified in the CNC system. Machining feedrate means the feedrate specified for linear interpolation or circular interpolation.

To prevent a mechanical shock, acceleration/deceleration is automatically applied when the tool starts and ends its movement. Furthermore, when the movement direction is changed between a specified block and the next block during cutting feed, the toolpath may be curved due to the relationship between the time constant of a servo system and the commanded feedrate. In the CNC system, linear, exponential, and S-shape acceleration/deceleration profiles, shown in Fig. 2.14, have been typically used. Each profile provides its specific characteristics in its own way. In general, the linear acceleration/deceleration profile has been widely used and enables the axis to reach at the commanded feedrate rapidly, in a simple way. Note, though, that the S-shape profile makes the axis movement smooth and has been widely used for high-speed machining.

Automatic acceleration/deceleration is very useful for preventing mechanical shock. However, it results in a servo delay due to the shift of speed profile by the acceleration/deceleration time constant and, finally, causes machining error. In particular, due to the machining error caused by automatic acceleration/deceleration of circular interpolation, the radius of the machined circular path comes to be smaller than that of the programmed circular path. The machining error is in inverse proportion to the radius of the circle being interpolated and in proportion to the square of the commanded feedrate.

As the command method to control the speed at the corner between the specific block and the next block, Exact Stop (G09), Exact Stop Mode(G61), and Cutting Mode (G64) can be used.

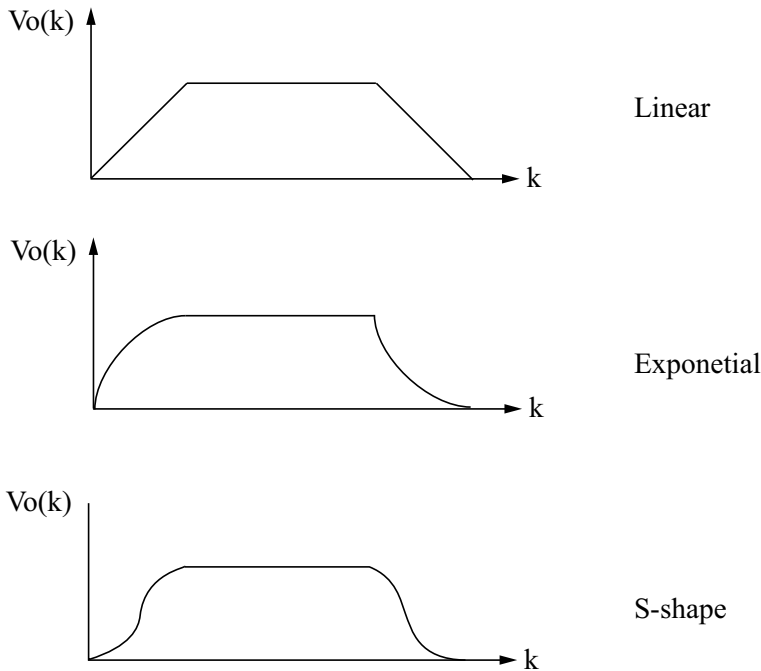


Fig. 2.14 Acceleration/deceleration profiles

In the block where Exact Stop(G09) is valid, the tool is decelerated at the end point of the block, then an in-position check is made. Under the rapid traverse movement, the tool is decelerated at the end point and an in-position check is made regardless of whether or not the command Exact Stop has been issued. When the Exact Stop Mode is specified, the tool is decelerated at the end point of a block, then an in-position check is made. This mode is valid until G62, G63, or G64 is specified and is used for making a right angle at the corner of a toolpath.

However, after Cutting Mode (G64) is specified, an in-position check is not made at the end point of the next blocks. In modern CNC systems, a Look-Ahead function is executed under Cutting Mode and this function is useful for increasing the actual machining feedrate during execution of the part program which consists of small line segments. The details of Look-Ahead function will be described in Chapter 3.

The dwell function (G04) is used for delaying the next execution block for the specified time interval. As this code is a one-shot G-code, it is valid only during the block where the function is commanded.

The threading function (G33) is used for the machining tapered threads and threads with a constant lead. When single screw threads are machined, the threading tool moves several times along the same path from roughing to the finishing process. For this thread cutting, the thread tool is started after detecting one revolution signal from the position coder attached to the spindle. Therefore, the start position

of threading is always identical in spite of repeating machining. In this way, it is possible to machine a single thread.

When multi-screw threads are machined, the start angle of threading is changed. If the angle is changed by 180 degrees, a double screw thread can be machined. If the change angle is 120 degrees, a triple screw thread can be machined. To machine multi-screw threads, the spindle speed is read from the position coder and the speed read is converted into the feed per minute value. The tool is moved based on the converted feedrate and the feedrate is identical during threading. However, if the feedrate calculated from the detected spindle speed exceeds the maximum allowable feedrate, the actual feedrate becomes smaller than the required feedrate and it becomes impossible to machine the thread with the required lead.

2.3.4 Tools and Tool Functions

The tool function (T-code) is used for selecting the machining tool with the specified tool number. The specified tool is effective until another tool is selected.

The tool life management function is used for managing the usage time and wear amount of each tool and the number of the part that is machined by each tool. This provides functions to replace the particular tool with a specified spare tool in the case when the usage time of the particular tool exceeds the pre-specified time or the number of parts machined by the particular tool exceeds the predefined number.

The tool radius compensation (G40, G41 and G42) functions are used for generating a path that is offset from the programmed path by the radius of tool. As shown in Fig. 2.15, the path followed by the tool center should be the path indicated by B, which is separated from A by the value R , in the case when a part, indicated by A, is machined by a tool with radius R .

Typically, the distance by which the tool is separated from the programmed path is called the “offset” and B in Fig. 2.15 is an offset path. The code G41 commands tool-radius compensation to the left of the tool movement direction, G42 commands tool-radius compensation to the right of the tool movement direction, and G40 commands cancelation of tool radius compensation. Tool compensation codes such as G41 and G42 are used with a D address that stores the tool offset value and the tool offset value is pre-specified by user.

Tool-radius compensation is applied differently according to the following modes.

1. *Cancel mode:* After power is turned on, the CNC system is reset, or M02/M30 is executed, the status of the CNC system turns into Cancel mode. In this mode, tool compensation mode is canceled and the path of the tool center point is the same as the programmed path.
2. *Start-Up mode:* If G41 or G42 is commanded in Cancel mode, the CNC system turns into Offset mode. (Figure 2.16).
3. *Offset mode:* This means the CNC operating period between the first block after declaring the tool radius compensation to the last block before canceling the tool-

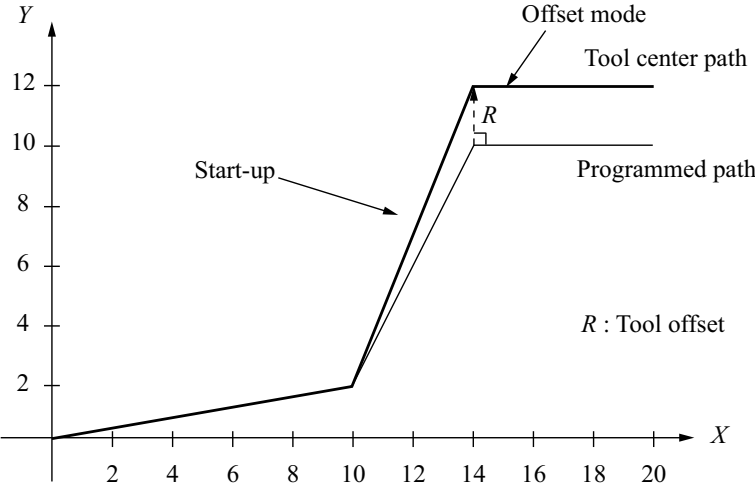


Fig. 2.16 Interpolation for start-up and offset modes

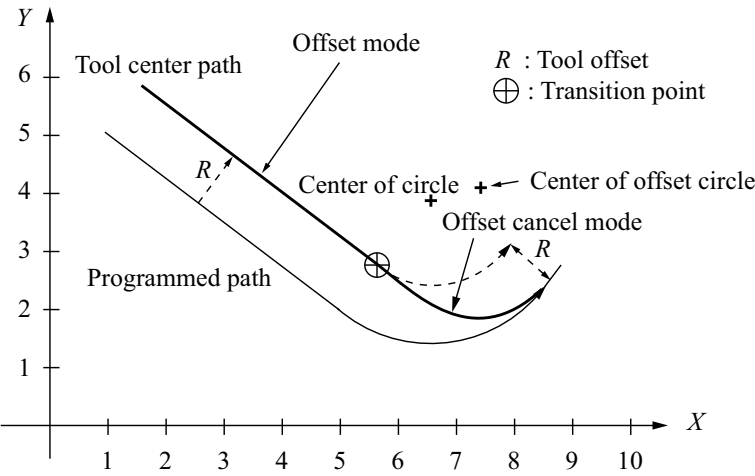


Fig. 2.17 Interpolation for offset and offset cancel modes

2.3.5 Spindle Functions

The spindle function (S-code) is for specifying the spindle speed and the spindle speed is restricted by the maximum spindle speed specified by user. The S-code is modal code and, therefore, the spindle speed specified by the S-code is effective until another spindle speed is specified. The spindle speed specified by an S-code is canceled after power on, or when the system is reset or when M30 is commanded. During execution of a part program, change of spindle speed is limited to being less than or equal to the specified maximum spindle speed.

The constant surface speed control function is used for rotating the spindle with constant surface speed regardless of the position of the tool. This function is applied for turning and the surface speed for this function is specified by the S-address. For this function the axis along which constant surface speed control is applied should be specified. To command constant surface speed control, the G96 command is used, and to cancel the constant surface speed control the G97 command is used.

Typically, the spindle connected to the spindle motor is rotated at a certain speed to rotate the workpiece mounted on the spindle. This spindle control status is referred to as spindle rotation mode. In addition to spindle rotation mode, the spindle position function, which turns the spindle through a certain angle, can be used to position the workpiece mounted on the spindle at a certain angle. Also, as the spindle orientation function is one of the spindle position functions, the spindle orientation function can be used to make the spindle stop at a pre-determined position. By specifying the particular angle using the S-code, it is possible to stop spindle at a particular angle. An example of the use of the spindle orientation function is given below.

```
N20 M03 S1000 ;  
N30 M19 ; → spindle stops at 0.  
N40 M19 S270 ; → spindle stops at 270  
N50 M03 ; → spindle begins rotating in 1000 rpm in clockwise direction.  
...
```

2.3.6 Fixed-cycle Function

The fixed-cycle function is used for executing specific machining for which more than one block is necessary using only one block. This is useful for simplifying a part program and the fixed-cycle code has been defined for a variety of machining in drilling, turning, and milling as shown in Table 2.4. The usage example of this function will be explained by using fine boring that is one of the cycle codes for drilling.

As shown in Fig. 2.18, the fine boring cycle command, G76, moves a tool to the reference position and stops. This command rotates the tool to a reference angle by commanding the spindle orientation function, moves the tool by a specified amount

Table 2.4 Operations and fixed cycle function codes

| Operation | | G-code | Operation | | G-code |
|-----------|------------------------------|--------|-----------|--------------------------|--------|
| Drilling | Peck Drilling | G73 | Turning | Roughing | G90 |
| | Reverse Tapping | G74 | | Threading | G92 |
| | Fine Boring | G76 | | Face roughing | G94 |
| | Cycle Cancel | G80 | | Finishing | G70 |
| | Drilling Cycle, Spot Boring | G81 | | Roughing | G71 |
| | Drilling Cycle, Count Boring | G82 | | Face roughing | G72 |
| | Peck Drilling | G83 | | Copying | G73 |
| | Tapping | G84 | | Grooving | G74 |
| | Rigid Tapping | G84.2 | | Face grooving | G75 |
| | Reverse Rigid Tapping | G84.3 | | Multiple threading | G76 |
| | Boring | G85 | Milling | Circular Elongated Holes | |
| | Boring | G86 | | Circular | |
| | Back Boring | G87 | | Circumferential Slot | |
| | Boring | G88 | | Facing | |
| | Boring | G89 | | Circular Pocket | |

to the opposite side of the tool cutter, and finally retracts the tool upwards to avoid damage to the machined surface.

The detailed procedure for the fine boring cycle function is given below.

1. The tool is moved to the cut start position.
2. The tool is moved rapidly to the R position.
3. With the tool movement to the Z position, boring is carried out.
4. If G76 is commanded with P address, the dwell function is executed.
5. The spindle orientation function (M19) is executed.
6. The tool is moved rapidly by the amount specified with the Q address along the direction specified by the parameter. (In this example, it is assumed that the XY plane is selected as the machining plane and, therefore, the tool can be moved along the X-axis or Y-axis.)
7. The tool is rapidly retracted to the specified position. If the G99 code is effective, tool movement position becomes the R position and if G98 code is effective, the tool movement position becomes the cut start position.
8. The tool is moved rapidly by the length specified with the Q address to the opposite direction pre-defined by parameter.
9. Tool rotation starts again.

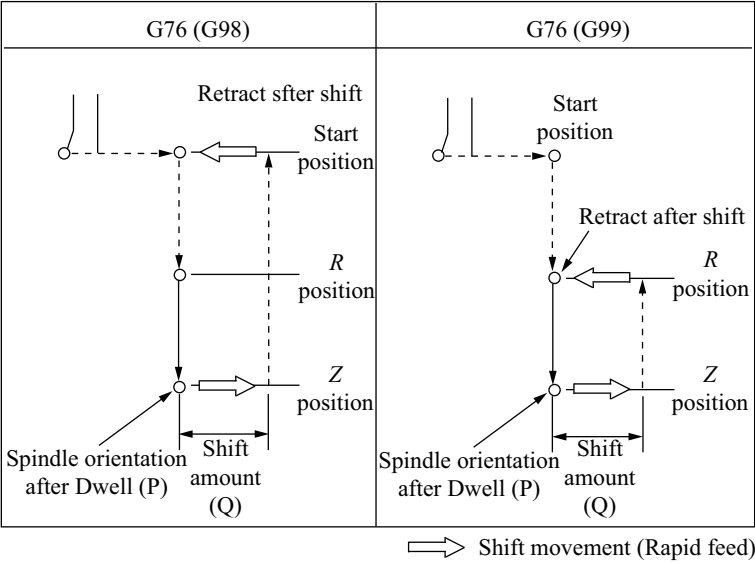


Fig. 2.18 Fine boring cycle movements

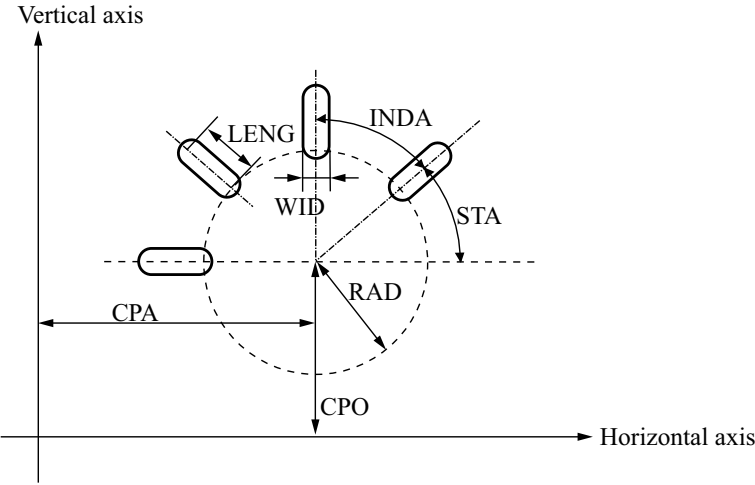


Fig. 2.19 Circular slot cycle - circular pattern of slots

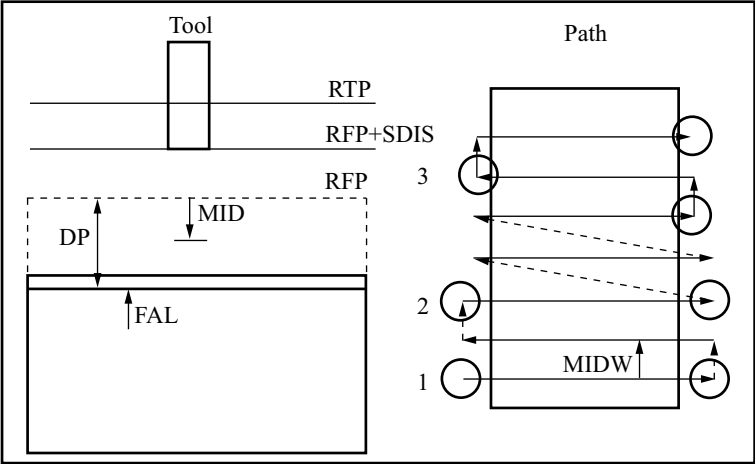


Fig. 2.20 Face milling pattern and parameters

2.3.7 Skip Function

During the execution of the skip function (G31), if an external skip signal is input, execution of the command is interrupted and the next block is executed. The skip function is commanded with linear interpolation such as G01. The skip function is used when the end of machining is not programmed but specified by a signal from the machine, for example, in grinding. It is used also for measuring the dimensions of a workpiece.

Figure 2.21 shows an example of the actual toolpath after the skip signal is detected in the case when absolute command mode is effective and the programmed path is on the XZ plane. As soon as a skip signal is detected, the tool (in this case a touch probe is generally used) is moved to the end point of the next block regardless of whether or not the tool reaches the end point of the current block. The feedrate of the linear path commanded by the skip function is specified by the F-address or a certain parameter and this feedrate is effective only on the linear path commanded by the skip function.

2.3.8 Program Verification

The part program edited by the machine tool operator is likely to include grammatical errors, logic errors, and numerical errors, such as incorrect computation of tool position, wrong tool-offset value, and invalid feedrate and spindle speed. Therefore, it is necessary to test the part program before executing it and the CNC system generally provides the functions listed below for immediate validation.

G90 G31 X200.0 F100 ;
 X300.0 Z100.0 ; → The tool is moved to the point of the next block.

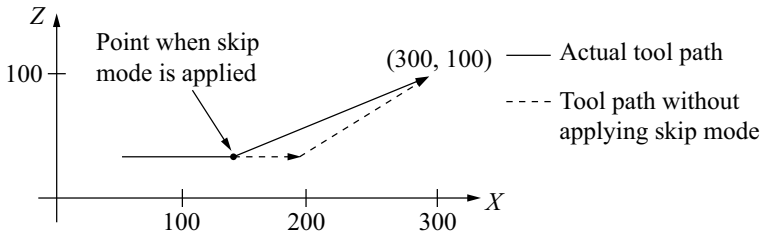


Fig. 2.21 Skip function action

1. **Dry Run:** During dry run mode, the tool is moved at the feedrate specified by a parameter regardless of the feedrate specified in the program. This function is used for checking the movement of the tool in the case where the workpiece is removed from the table. The tool moves at the feedrate specified by the parameter. The feed override switch can also be used for changing the feedrate during this mode but during automatic mode, dry run is not allowed to begin.
2. **Pressing the single-block switch starts single-block mode.** When the cycle start button is pressed in single-block mode, the tool stops after a single block in the program is executed. This function is used for checking the program block-by-block and can be used with the dry run function and machine lock function.
3. **Machine lock** is used to display the change in position without moving the tool and there are two types of machine lock: all-axis machine lock, which stops movement along all axes, and specified-axis machine lock, which stops movement along specified axes only.

2.3.9 Advanced Functions

Recently, CNC machine tools have become more accurate and faster and the functionality has become more complicated. To satisfy these requirements, advanced functions for high-speed and high-accuracy machining have been developed and applied in addition to the functions mentioned in the previous sections. The next sections describe typical advanced functions built into highly functional CNC systems.

2.3.9.1 Look Ahead

Generally, the part program for surface machining (die and mold is a typical example of surface machining) consists of a sequential linear path with short length and fast feedrate. In this case, if each block is executed line by line, the actual feedrate

becomes less than the programmed feedrate and the feedrate at the corners between one specific block and the next becomes discontinuous.

Therefore, the quality of the machined surface is degraded due to frequent acceleration/deceleration and the discontinuity of the feedrate and, after completing machining, grinding becomes essential. To solve this problem, the Look-Ahead function was developed. The look-ahead function looks ahead a hundred blocks and calculates an adequate feedrate for each axis within the maximum allowable feedrate and acceleration/deceleration.

With this function, it is possible to machine the free-form surfaces and contours of a complicated shape without stopping tool movement between successive blocks at high speed. The concept of the look-ahead function can be easily understood by comparing it with car driving. At night, it is difficult for the driver to see for long distances and, therefore, it is difficult to drive at the maximum allowable speed. However, during the day, a driver can see longer distances and, therefore, it is possible to examine the road status, predict maximum feasible driving speed, and, finally, to drive faster.

The look-ahead function calculates the maximum feasible feedrate of the specified block based on the interpreted result of the blocks that will be executed. This function requires much computing power. Recently, with the advance of CPU power, the number of the blocks that can be used for look ahead has grown to a thousand. Figure 2.22 shows the feedrate profiles when the look-ahead function is applied and when it is not and Fig. 2.22 also shows that the look-ahead function can increase the actual feedrate.

When the look-ahead function is applied, the feedrate at the end of the starting block (N1) is not decelerated and the programmed feedrate is kept to the programmed feedrate. To stop at the end position of the last block (N12), the deceleration of the feedrate starts in the preceding blocks. Therefore, the look-ahead function enables high-speed machining compared to exact stop mode where acceleration and deceleration is done at the start point and the end point of each block. Accordingly, with this function, reduction of machining time becomes possible.

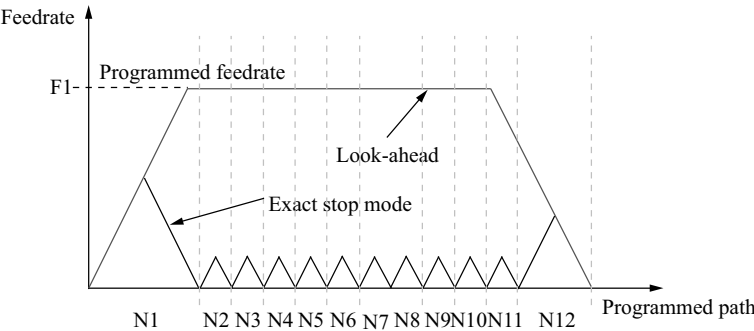


Fig. 2.22 Look-ahead mode and Exact stop profiles

2.3.9.2 Feedforward

The conventional position control method essentially has the following error and it is proportional to the square of the feedrate during high speed machining.

The cause of the control error is mainly based on the servo delay. In order to reduce the machining error, it is necessary to increase the position control loop gain. However, increasing the position control loop gain is likely to result in machine vibration and make the servo system and the machine unstable. Accordingly, as the feedforward control method plays the role of making the servo system stable and increasing the position control loop gain, it makes it possible to reduce the machining error and achieve high-speed and high-accuracy machining.

Figure 2.23 shows the actual feedrate profiles and path traces when the feedforward control method is applied and when it is not applied. From Fig. 2.23, we can see that when the feedforward control method is applied, the following error decreases and the machining error obviously also decreases.

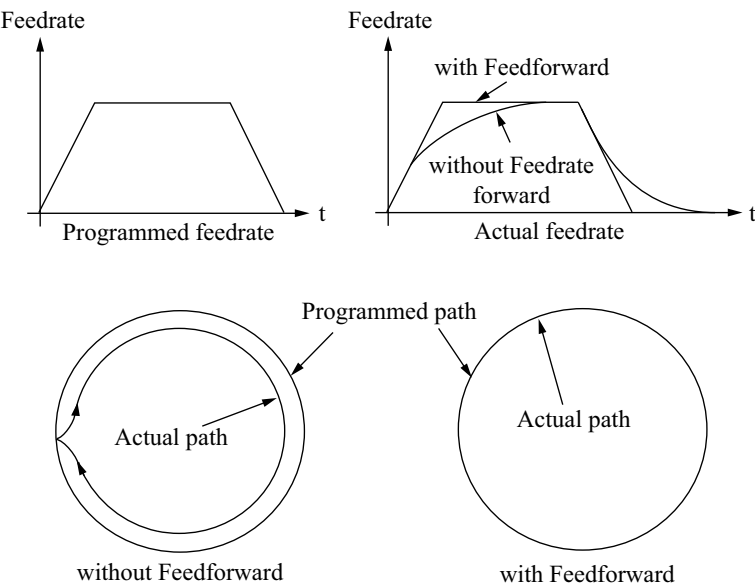


Fig. 2.23 Feedrate profiles and path traces with and without feedforward control

2.3.9.3 NURBS Interpolation

As high-speed machining and high-accuracy machining come to be generally used, the requirements for advanced functions to support them is growing. In particular, when conventional CNC systems (where free curve is defined by sequential small

line segments or arcs) are used for machining free-form surfaces, the tool moves in a discontinuous manner and this makes the quality of the machined surface poor. Also in this case, because a lot of program blocks are required, the size of the part program is large. Because the size of the internal memory of CNC system is limited, DNC (Direct Numerical Control) mode has typically been used to machine free-form surfaces. Since the baud rate of DNC communication is restricted it becomes impossible to raise the machining speed over a restricted specific value when a conventional CNC system is used. To overcome this problem NURBS interpolation was developed. In this section, the necessity of NURBS interpolation will be described and the details will be given in Chapter 3.

In NURBS interpolation, NURBS curve data (*e.g.* control points, weights, and knot vector) are directly input to the CNC system instead of the small line segment data that are defined by the G01 command. As the CNC system generates interpolation points based on the NURBS curve data, the programmed feedrate and the tolerance, it makes it possible to perform high-speed and high-accuracy machining.

Figure 2.24 shows the difference between the interpolation methods based on line-segment approximation and a NURBS curve. When offline CAM software is used the free-form curve geometry is approximated to within a pre-defined tolerance by a set of line segments. These in turn are then subdivided into a set of shorter line segments to give the desired feedrate. With direct NURBS interpolation in the CNC the interpolation the feedrate and tolerance are used to determine the step length along the curve directly to give the required speed profile.

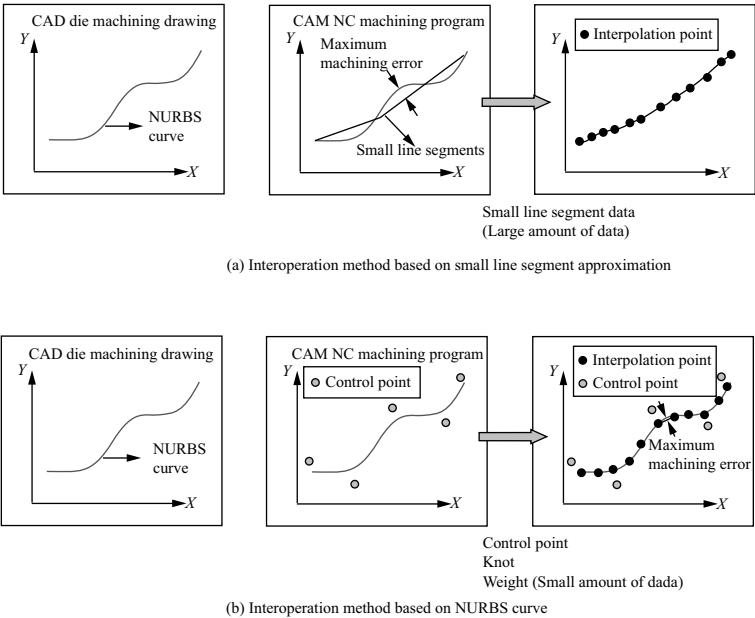


Fig. 2.24 Indirect and direct NURBS interpolation

2.3.9.4 NURBS Surface Machining

For free-form surface machining, recent CAD/CAM systems include a function to transmit the free surface data into the CNC system using the NURBS surface form. The G-code format to specify the NURBS surface data is different according to the CNC maker. However, despite the differences in the G-code representation method, the data elements for representing a NURBS surface are the same. Table 2.5 summarizes the status of development of NURBS interpolation functions for different CNC makers.

Table 2.5 Controller NURBS development summary

| | FANUC | SIEMENS | OKUMA | Mitsubishi | Toshiba Machine |
|-----------|---|------------------------------|--------------------------|----------------|-----------------|
| CNC Model | 15 Series, 16 Series, 18 Series, 30 Series | 840D | OSP700M (spar H1 CNC) | M700 | TOSNUC 888 |
| CPU | 64bit RISC | RISC | RISC | | |
| G-code | G06.2 | Language Type: BSPLINE | G132 | G70.0 G70.1 | |

Figure 2.25 shows the G-code format for representing a NURBS curve and Fig. 2.26 shows an example of a part program to machine a NURBS curve profile.

G06.2

P_X_Y_Z_R_K_F ;

G06.2 : NURBS interoperation

P : NURBS curve order

X, Y, Z : Control point

R : Weight

K : Knot

F : Feedrate

Fig. 2.25 FANUC system NURBS G-code format

In Fig. 2.26, in the block whose line index is 110, the degree of the NURBS curve and the feedrate are specified. From the block whose line index is 110 to the block whose line index is 350, the control points and knot vector are specified. In the case of the NURBS curve defined in Fig. 2.26, the degree of the curve is 4, the feedrate is 10 mm/min, and the weight of all control points is 1.

2.4 G&M-code Interpreter

As mentioned in the previous section, the interpreter of the CNC system is the software module of the NCK unit that interprets the part program consisting of G&M-code commands and related addresses such as S, T, and F. The interpreter consists of a parser, an executor, a path generator, a macro executor, and an error handler. The parser consists of a lexical analyzer, a calculator, and a sentence interpreter (YACC). As for the software modules connected with the interpreter, there are the program access module, which reads a program file, and the interpolator module, which generates the interpolated points of the programmed path based on the interpreted data. Figure 2.27 shows these modules in graphic form.

```
N100  G05 P10000
N110  G06.2 P4 K0. X-1.6953
      Y-.75 Z-.2358 F10
N120  K0. X-1.6544 Z-.2313
N130  K0. X-1.5752 Z-.2225
N140  K0. X-1.4053 Z-.2067
N150  K.0313 X-1.3031 Z-.1982
N160  K.0781 X-1.1215 Z-.1847
      ...
N300  K.9063 X1.7085 Z-.2373
N310  K.9688 X1.75 Z-.2421
N320  K1.
N330  K1.
N340  K1.
N350  K1.
N360  G01 Y-.7188 Z-.238
```

Fig. 2.26 NURBS-profile G-code part program

The functionality of each module is given below.

1. *Parser*: this module interprets the part program block by block. The lexical interpreter of this module reads the block character by character and makes meaningful words from the characters. The calculator carries out numerical operations within the part program. The sentence interpreter retrieves the command and the related data such as G-code, M-code, S-code, T-code, conditional branching, and iteration loop based on the words from the lexical interpreter.
2. *Executor*: this module executes the functions related with the interpreted sentence and stores the execution result in the internal memory. In addition, this module generates the data required for executing the modal code.
3. *Path Generator*: This module generates the position data based on the programmed coordinates. In this module, the computation for mapping from work-

- piece coordinates to machine coordinates, tool compensation, and the axis limit is carried out.
- 4. *Macro Executor*: this module interprets and executes macro commands included in an NC part program. As the macro is user-defined code, the user can make specific functions that are not provided by the CNC maker by using a macro language, which is similar to the BASIC language.
 - 5. *Error Handler*: if there is an error in a part program, the error should be noticed and the user notified. This module is responsible for this.

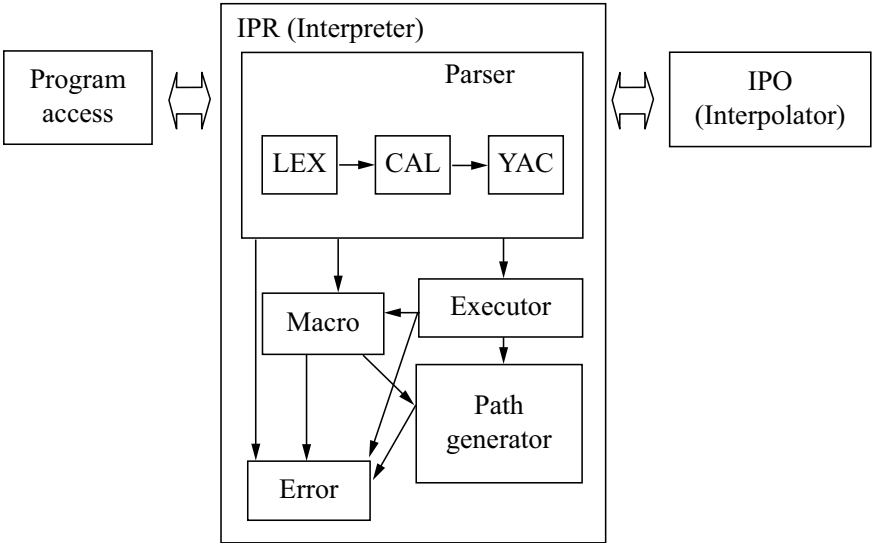


Fig. 2.27 Code interpreter modules

The workflow that should be executed by the interpreter with the various S/W modules from interpreting the part program to generating the tool position is shown in Fig. 2.28.

First, once the Cycle Start button on the MMI panel has been pushed, the interpreter starts pre-processing tasks such as reading the part program into the internal memory of the CNC system block by block, interpreting the block, and storing the interpreted data in the internal memory. During the pre-processing tasks, a cycle code is converted into blocks that consist of G01, G02, and G03. These converted blocks replace the cycle code and the interpreter reads and interprets the converted blocks. The internal block memory can be defined as in Table 2.6. The block memory shown in Table 2.6 shows the memory contents when the block whose line index is N300 has been read and where the block skip is specified and interpreted in subprogram P9000. The interpreter reads the addresses and the following numbers specified in the N300 block and stores the interpreted values in the related internal block mem-

ory. In addition, the SLASH variable that denotes whether a block skip command is on or not is set to true.

Next, the interpreter reads and performs the data on internal block memory. If the codes M02 or M30 are executed, the part program is 'rewound'. If the interpreter executes M98, the interpreter calls the subprogram whose number is the value following the P address. If M98 is commanded together with the L address, the interpreter executes the subprogram repeatedly as many times as the number following the L address.

Further, a macro call is monitored and, if it is called, the specified macro program is called. To make macro execution fast, it is more efficient to pre-interpret the program called by the macro, store the program as intermediate code, and execute this. As this method is one of the high-speed interpreting solutions, the definition of the virtual intermediate code is necessary and an additional execution module is required. How to call a macro program is similar to how to call a subprogram. In a macro program call, though, it is possible to specify arguments and use the global variables in contrast to execution of a subprogram call.

The next step is to execute the G-code. If the current block is the first block, the interpreter initializes the local variables and controls the optional block skip command. It also performs the G-code processing such as setting the G-code type, G-code group, modal data, and non-modal data.

The interpreter performs F-code processing where the interpreter reads the F-code data from the internal block memory and specifies the related variable. During F-code processing, the feedrate definition method such as feed-per-minute (mm/min, inch/min) or feed-per-revolution and the unit of feed-override and dwell time are set.

After completion of the above-mentioned stages, the end position of a block is computed as the final step of the interpretation. Generally, as stated earlier, various coordinate systems are used to make editing of the part program easy for machine tools with CNC systems.

There are three kinds of coordinate system used for machine tools; a machine coordinate system that is the reference coordinate system of machine tool itself and is defined with zero return, a workpiece coordinate system that is used for editing of a part program, and a local coordinate system that is used as a reference for machining and is specified based on the workpiece coordinate system. The machine coordinate system is specified by G53, a workpiece coordinate system is specified by G54 to G59, and a local coordinate system is specified by G52.

Accordingly, based on the above coordinate systems, for computation of the output position of the block, first, if necessary, values in inches are converted into values in millimeters. Incremental values are converted into absolute coordinate values. If rotation of the coordinate system is commanded, the programmed coordinate is rotated. If mirroring of the coordinate system is specified, then the mirror function is executed. If scaling of the coordinate system is specified, then the scaling function is performed.

The next stage is the path-modification stage where tool-length compensation and tool-radius compensation are executed. Finally, the data used for the check limit

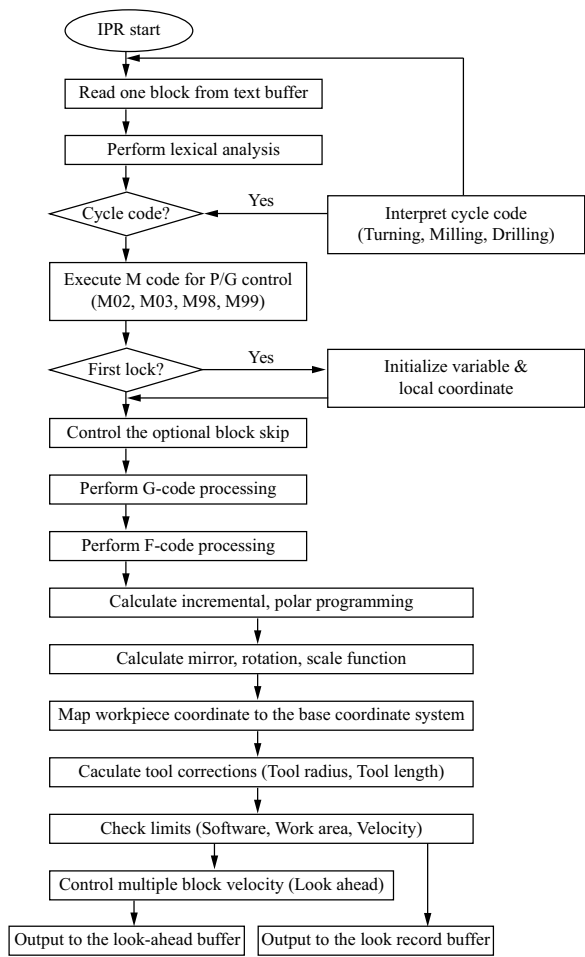


Fig. 2.28 Flowchart for interpreter function

value such as velocity, software, and work area are transmitted into the interpolator. Table 2.7 shows an example of the implementation of the block record memory.

The block record memory stores not only the interpreted data of a specific block but also the data specified in the CNC system. If the look-ahead function is used, the feedrate of the block is recalculated using the look-ahead function and the modified feedrate is stored in the additional memory for look-ahead function. The details of the look-ahead function that is used for preventing reduction of the feedrate for sequential small line segments will be described in Chapter 3.

Table 2.6 Memory map for the interpreter

| Field | Type | Value | State | Type | Value |
|---|--------|---------|---------------|----------|-------|
| A | double | | SLASH | unsigned | 1 |
| B | double | | PERCENT | unsigned | |
| C | double | | NUMBER | unsigned | |
| D | double | | ID | unsigned | |
| E | double | | CHANGE[M_ADD] | unsigned | |
| F | double | 1000 | GFLAG[MAX-G] | unsigned | |
| G | double | 01 | MFLAG[MAX-G] | unsigned | |
| H | double | G_MODAL | int | | |
| I | double | | | | |
| J | double | | | | |
| K | double | | | | |
| L | double | | | | |
| M | double | | | | |
| N | double | 300 | | | |
| O | double | | | | |
| P | double | | | | |
| Q | double | | | | |
| R | double | | | | |
| S | double | 5000 | | | |
| T | double | | | | |
| U | double | | | | |
| V | double | | | | |
| W | double | | | | |
| X | double | 150 | | | |
| Y | double | 20 | | | |
| Z | double | 30 | | | |
| P9000 N100 G92 G96; N200 G00 X100; N300 M03; /N300 G01 X150 Y20 Z30 F1000 S5000; N400 G1 Z120; | | | | | |

2.5 Summary

The interpreter plays the role of converting a user-edited part program into the internal data format for execution. In order to understand the structure and the internal behavior of the interpreter, it is necessary to understand the structure of a part program and the commands used therein.

In a CNC system, various coordinate systems, such as the machine coordinate system, workpiece coordinate system, and local coordinate system, are supported for the convenience of editing a part program and setting up the machine. Also, rotation, mirroring, and scaling of a coordinate system are provided and by using these functions it is possible to easily edit the part program.

Table 2.7 Block record memory

| Entity | Type | Comments |
|----------------------|-------------------|-------------------------------|
| End_point | Real Array[8] | Block end point |
| Start_point | Real Array[8] | Block start point |
| Prog_F_value | Real | Feedrate |
| Prog_S_value | Real | Spindle speed |
| F_limit | Real | Feedrate limit |
| S_limit | Real | Spindle speed limit |
| Scaling_factor | Real | Scale |
| G_code_group | Integer Array[20] | G-code group |
| Exact_stop | Integer | Exact stop mark |
| Block_number | Integer | Block number |
| Sub_PG_index | Integer | Subprogram index |
| Act_Number_PG_repeat | Integer | Number of program repetitions |
| Gear_box | Real | Parameter for gear box |
| Correction_active | Integer | Tool compensation start |
| Tool_correction | Real | Tool compensation |
| IPO_type | Integer | Interoperation type |
| G_Code | Integer Array[8] | G-code in block |
| Thread_flag | Integer | Thread start |
| Handwheel_flag | Integer | Handwheel start |
| Handwheel_direction | Integer | Handwheel rotation direction |
| Block_length | Real | Block length |
| Block_pointer | Pointer Array[4] | Block pointer |

To control tool movement along a line, an arc, a helical, or a spline path, interpolation functions such as G01, G02, or G03 code, F-code for specifying the feedrate, and S-code for specifying the spindle speed are used. To carry out the part program where the tool shape and assembly are not considered, the tool-radius compensation function and tool-length compensation function are provided. Furthermore, the macro function, the so-called “cycle function” is provided for convenience of editing a part program and simplification of the part program. Recently, to fulfill requirements for high-speed machining and high-accuracy machining, various advanced functions such as the look-ahead function, feedforward control function, and NURBS interpolation function have been applied.

Finally, the interpreter, which performs the above-mentioned functions, consists of a parser, an executor, a generator, a macro executor, and an error handler. The interpreter converts the block data read from the text memory into the internal data structure. Based on the interpreted data, the position of a block is computed by executing various mathematical operations such as the coordinate rotation and tool compensation and is stored in the block record memory.

Theory and Design of CNC Systems

Suh, S.-H.; Kang, S.K.; Chung, D.-H.; Stroud, I.

2008, XX, 456 p., Hardcover

ISBN: 978-1-84800-335-4