

Chapter 8

Man–Machine Interface

The Man–Machine Interface (MMI) provides the interface that enables a user to operate a machine tool, edit a part program, perform the part program, set the parameters, and transmit data. In this chapter, the function and components of the MMI will be addressed, and programming methods such as CAPS (Conversational Automatic Programming System) will be described. In addition, for designing CAPS, the main functions and components of CAPS will be described.

8.1 MMI Function

In order for a user to operate a machine effectively and to use the function of the machine optimally, it is necessary to design the operation panel for usability according to the machine–tool characteristics. In other words, an operation panel should be designed from the point of view of ergonomics, operation error prevention, key grouping and key allocation for specific machine tools with regard to user convenience. Figure 8.1 shows a typical operation panel and, in general, the operation panel can be divided into four areas.

8.1.1 Area for Status Display

This area displays the machine status and NC parameters. It provides the graphical user interface (GUI) for interaction between the CNC and the user. Figure 8.2 shows a typical display of this area and the functions related to the numbers shown in Fig. 8.2 are as follows.

1. *Machining information*: Displaying information related to the current machine status including the coordinates of machine tools, current part program, cutting tools and machine parameters.

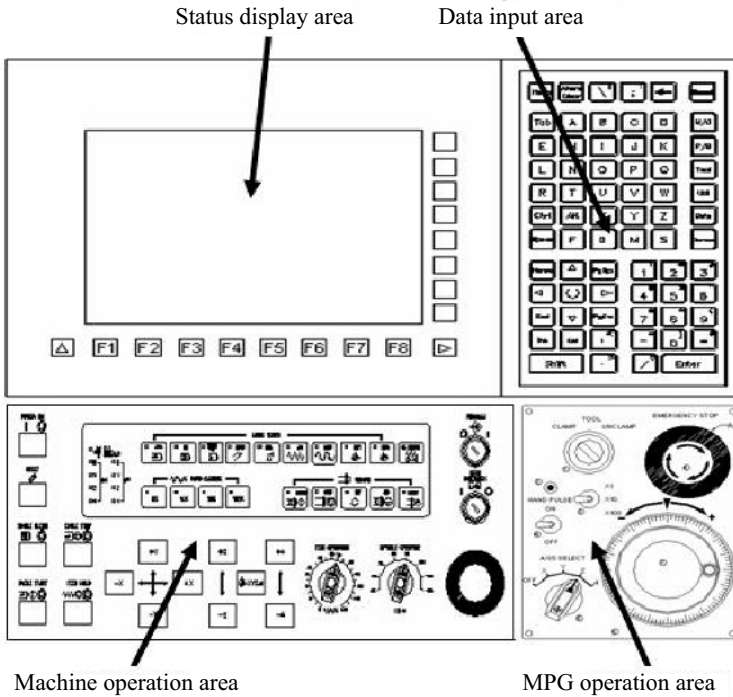


Fig. 8.1 Typical operation panel

2. *Operation Mode:* Displaying the operation modes of machine tools, such as zero position return mode, JOG mode, Automatic mode and MDI mode.
3. *Program name:* Displaying the name of the program that is currently loaded in the memory for machining.
4. *Alarm window:* Displaying the warning and alarm messages.
5. *Key input window:* Displaying the strings that are typed by a user.
6. *Window for displaying user interface relevant to operation mode and function:*
 - *Machining status (POS):* operation status such as axis position, spindle speed, feedrate, modal G-codes, and tool number is displayed by this function.
 - *Program (PROG):* the GUI for editing a part program, managing the program folders, graphical simulation, and CAPS is provided by this function.
 - *Tool management:* the GUI for managing tool compensation, tool life, and tool offset is provided by this function.
 - *Parameter and system:* the GUI for managing the NC parameters, system parameters for servo and spindle is provided.
 - *Auxiliary application:* the GUI for monitoring PLC, displaying alarms, performing DNC, and compensating pitch error is provided.

7. *Function keys*: these keys are horizontally placed in the bottom or vertically on the right-hand side of the display and are mapped to the particular functions. Therefore, to effectively design the menu structure, it is important to classify the functions into the appropriate group and enable the necessary keys to be displayed in one display. It is necessary to consider that the number of hierarchical layers increases if CNC functions are grouped and are designed as a hierarchical structure. Therefore, if the user wants to select a particular menu at the bottom of the hierarchical structure, the user has to select a sequence of menus from the top menu to the bottom menu. Also, the user has to remember the hierarchical structure and the menus located in each layer. This problem makes the user interface inefficient.

To overcome this problem, it is necessary to design a ring menu structure of menu trees where, by selecting the displayed menu tree, the user can carry out the desired task from the function keys displayed on one screen as much as possible and each function key is connected with the various modes. In this type of menu structure it is not necessary to remember the menu structure. However, the menu structure may be inconsistent and many function keys may be required.

8.1.2 Area for Data Input

As this area is the keyboard for inputting user data to the CNC system, it consists of alphanumeric input buttons and hot keys for executing the functions of CNC.

8.1.3 Area for MPG Handling

This area consists of the MPG (Manual Pulse Generator), the MPG handle ON/OFF switch and the feed ratio selection key that are used for the user to move each servo axis manually. In addition, the Chuck CLAMP/UNCLAMP key for manually loading and unloading tools to the spindle and the emergency stop button are located in this area.

8.1.4 Area for Machine Operation

This area consists of many kinds of switch and lamp that provide various functions as follows.

1. *Mode selection switch*: for selecting Auto mode, MDI mode, Teach-In mode, Return mode, JOG mode, Handle mode, Incremental Moving mode, and Rapid Moving mode.

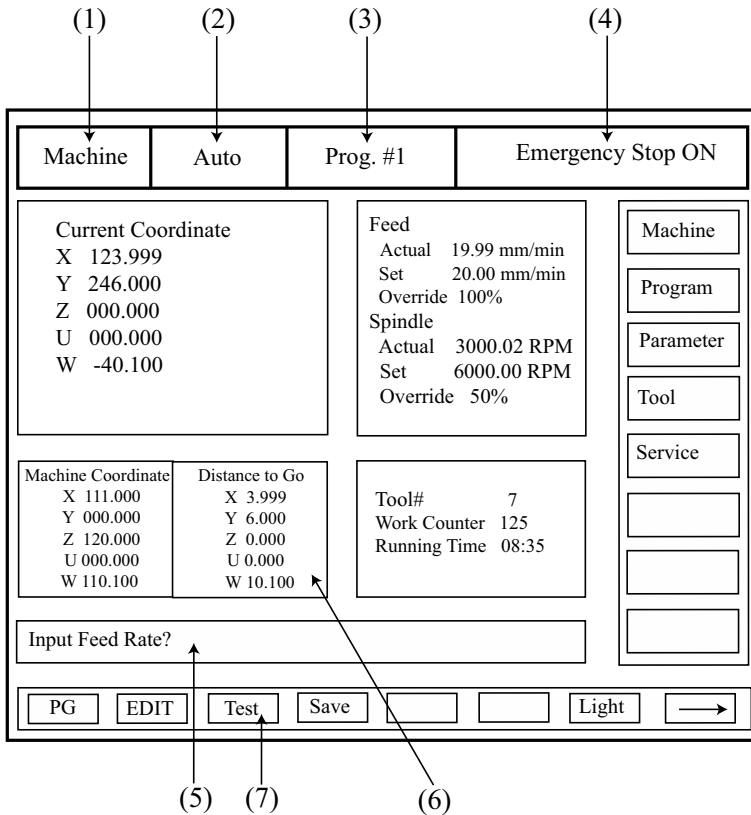


Fig. 8.2 Typical machine status and NC parameters display area

2. *Rapid Override button*: by using this button, rapid feed can be adjusted in scale to 10%, 50%, and 100%.
3. *Feed override switch*: by using this switch, the commanded feedrate can be adjusted from 10% to 150%.
4. *Spindle speed override switch*: using this switch, the commanded spindle speed can be adjusted from 50% to 150%.
5. *Spindle handling buttons*: these buttons consist of the spindle start button, the spindle stop button, rotation direction selection button, and the spindle orientation button, inverse. These buttons are used in MDI mode.
6. *Cycle Start button*: This button is used for starting the auto-execution or resuming the execution of a part program during feed hold state.
7. *Feed Hold button*: This button is used for temporarily stopping the axis movement in automatic machining. When the button is pushed, the spindle continues to rotate. If any axis of the machine tool is moving, that axis is stopped after deceleration.

8. *Single Block Button*: Single block execution means that in auto mode or MDI mode, the execution of a part program is stopped after the execution of one block has been completed and the next block begins only after the Cycle Start button has been pushed. The single block button turns on or off single block execution mode. If this button is ON during the execution of a part program, the CNC system goes into the idle state after completing the executed block. If this button is OFF, the remaining blocks are executed.
9. *Zero return button*: This button is used for making each axis return to the zero position. All axes can be returned to the zero position simultaneously. Feed override is validated during zero return.
10. *Emergency Stop button*: This button is used for stopping the machine in an abnormal state as soon as possible.
11. *Part program modification Lock/Unlock key*: This key is used for preventing an unauthorized user from modifying, editing, or deleting part programs or preventing unintended modification of a part program due to incorrect operation by a user.
12. *Door Interlock key*: In the case that this key is ON, if a door is opened while the spindle is rotating, the emergency stop is invoked.
13. In addition, there is an OT (Over Travel) cancel button that temporarily cancels safety mode when an axis moves beyond its set limit, a power switch, and a reset button that initializes the CNC system.

8.2 Structure of the MMI System

The ultimate design goal for the MMI system is to provide ease of operation and various functions for users. Following this trend, MMI has advanced to become PC-based MMI that is operated by an individual processor and allows various functions and advanced functions to be invoked from a single panel whereas traditional MMI only allows simple operations.

PC-based MMI allows the usage of a graphical user interface that replaces the earlier simple textual user interface. It also allows a CAM system to be used on the CNC system itself and enables the CNC system to communicate with external equipment. Furthermore, the user can use the various functions normally found on a PC. In recent times, the majority of PC-based MMIs use Windows OS from the Microsoft Corporation as an operating system, which makes third-party development and deployment of MMI applications relatively easy. Accordingly, the MMI system of PC-based systems are developed continuously to meet various user requirements. The details of PC-based systems will be addressed in Chapter 9.

As shown in Fig. 8.3, the structure of the MMI software can be divided into three layers; Application layer, Kernel layer, and OS layer.

The application layer is composed of the applications with which the user interacts. The following MMI functions belong in this layer and each application is made in stand-alone executable file format.

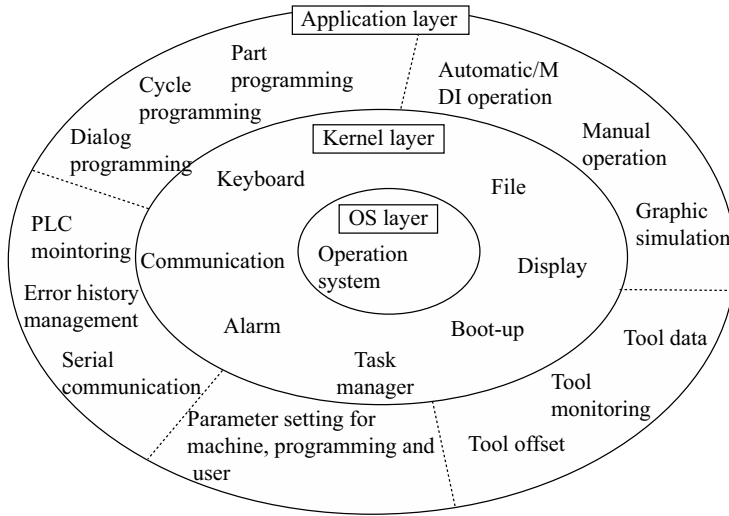


Fig. 8.3 MMI software structure

1. *Machine Manager*: This program monitors the machine status and displays the real-time tool path during machining in Auto mode or MDI operation mode.
2. *Parameter Manager*: The user can edit NC parameters and system parameters using this program.
3. *Program Manager*: This program provides the functions for editing G-code programs and managing part programs such as saving and deleting.
4. *Tool Manager*: This program is used for editing and managing the tool information, such as tool offset, tool life, and tool geometry.
5. *Utility*: Service functions of the CNC system such as alarm history management, PLC monitoring, DNC, and communication with external systems are provided.

The functions provided in the application layer may be added, deleted, or replaced according to the user's needs. Therefore, in order to make this possible, openness should be considered when the kernel layer is designed.

As the kernel layer is the core of the MMI software, it plays the role of linking the applications and the NCK. It sets environment variables during system boot-up, links application modules with key input and alarm/help file, and transfers files and parameters. The binary modules for executing the following functions are placed in the kernel layer. The modules are automatically linked with the applications while the CNC system is running.

1. *System boot-up*: This function initializes the variables of the operating system and system boot manager for setting the language type of MS Windows, machine parameters, etc.

2. *Communications interface*: This carries out communication and data exchange with the NCK and PLC. It manages the services for sending the data required by the user to the MMI for display.
3. *File management*: This provides the services for managing folders and files, such as copying, saving, deleting, and changing part programs and PLC programs.
4. *Alarm*: This displays alarm and error messages from the machine, PLC, and MMC in the alarm window. It manages the history and displays the help information.
5. *Key input*: This transmits the key input from soft keys, keyboard, and dialog boxes to the applications and the CNC system.
6. *Screen Display*: This handles the horizontal or vertical function key window that is shared by all applications and connects the function keys with particular applications. In addition, it provides the interface for handling MMI soft keys.
7. *Task manager*: This executes the programs registered in the application layer and provides the function for calling and switching them. It registers the applications as a program list in a text file format and executes the applications sequentially when the task manager begins. When the task manager is terminated, it terminates the applications in reverse order. The basic functions can be summarized as follows.
 - Registering/terminating applications
 - Defining the execution sequence for applications and initializing them while booting up.
 - Switching applications while they are executing.
 - Monitoring system resources.

An MMI system based on PC hardware typically uses a PC operating system as OS. MS Windows or Linux have both been used (recently, Windows embedded XP and Windows CE have become widely used) However, these operating systems cannot provide the real-time capabilities required by a CNC system. Generally, an MMI system requires a non-real-time environment, whereas an NCK system needs a real-time environment. Therefore, when the overall architecture of the CNC system is designed, techniques to overcome the non-real-time capabilities of the PC operating system must be considered. One simple solution is to use two operating systems, using a PC operating system (non-real-time OS) and a hard real time OS for the MMI and NCK systems, respectively. In this case, it is very important to regard the execution of the MMI system as one specific task in the NCK system.

In the MMI, various applications are executed based on the kernel and the user interface for editing a part program, which is one of the key applications in MMI. In general, the machine tool operator spends a lot of time learning how to generate a part program. So, from the MMI designer's point of view, the MMI should be designed for the MMI to be able to provide the most efficient method for generating a part program. In the following sections, the advantages and disadvantages of various programming methods will be discussed. The design of an efficient programming system will also be addressed.

8.3 CNC Programming

In order to machine the part in a drawing by using CNC machine tools, it is necessary to generate a series of instructions for activating those CNC machine tools. This task is called CNC programming.

8.3.1 *The Sequence of Part Programming*

Roughly, CNC programming is composed of the generation of a process plan from a part drawing and the generation of the part program. The detailed processes are as follows.

1. To analyze the part drawing.
2. To decide on the removal volume and to select the machine.
3. To decide on the jig and chuck.
4. To decide on the setups, machining sequences, cut start points, cut depths for roughing and finishing allowance.
5. To select tools and tool holders and to decide on the tool position.
6. To decide on the technology data such as spindle speed, feedrate, and coolant on/off.
7. To generate the part program (including post-processing).
8. To verify the part program.
9. To machine.

The tasks from stage 1 to stage 6 are included in the preparation stage where the part drawing is analyzed and the machining strategy is decided for creating a part program. These tasks are called “process planning”. Process planning is done by a programmer or a machine operator. Extensive knowledge about the machine tools, CNC equipment, tools, and cutting theory is required to generate fine process planning. However, in practice it is very difficult to find experts for these. Therefore, many studies on CAPP (Computer Aided Process Planning) for automatically executing process planning have been carried out.

After process planning, a part program (stage 7) for controlling CNC machine tools is generated. The generation of this part program can be done by the manual programming method or the automatic programming method. In the manual programming method, a programmer directly edits the part program in CNC-readable EIA/ISO code. In the Automatic programming method, a programmer edits the program in terms of graphical symbols or a high-level language via a computer. The CNC system then converts this program into machine-readable instructions and executes those instructions.

The automatic programming method can be classified into two types in terms of the editing method; the first is the language-type programming method where a high-level language is used for programming. The second type is the conversational

programming method where a programmer creates the program as he/she converses with the CNC system using graphical symbols. The various programming methods are depicted in Fig. 8.4. The key characteristics of each programming method will be described in detail in the following sections.

After completing the part program, the part program is verified by using simulation (stage 8). Through the simulation, errors can be detected and corrected. Also, if necessary, test cutting is carried out before real machining begins.

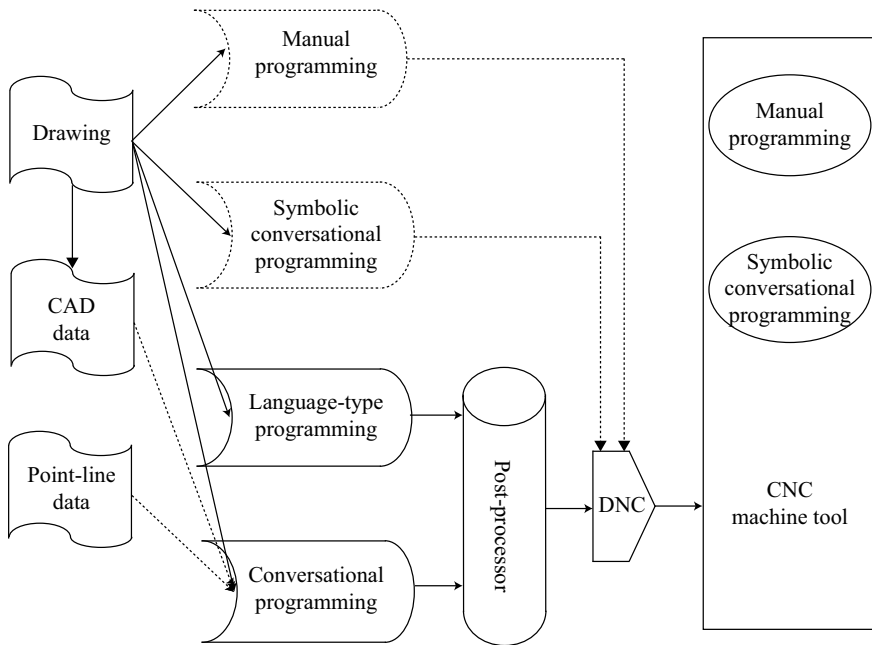


Fig. 8.4 Programming methods

8.3.2 Manual Part Programming

CNC equipment provides various instructions for the preparation functions, feed functions, spindle functions, tool functions, auxiliary functions, and other functions to meet the EIA/ISO standards. Direct editing of the program with the instructions (codes) provided by the CNC equipment is called manual programming. The part program generated by manual programming method can be executed not only within CNC equipment but also outside the CNC equipment.

Due to the differences in terms of function and design concept between CNC makers, each CNC system has a slightly different programming instruction set compared

with other CNC systems, although the EIA/ISO standard for programming instructions exists. This makes it difficult for a programmer to use a variety of CNC systems. Also, for the manual programming method, the efficiency and productivity of the part program depends on the programmer's ability. Therefore, knowledge about process planning, machining theory, G-code, and complex computations for tool-path generation are necessary for a good programmer and a long training time and much effort are also required. Further, because of the lack of compatibility between programming instructions (G-code), a programmer has to learn new programming instructions if the CNC system is changed. In addition, it is almost impossible to create a part program for machining 2.5D or 3D shape using the manual programming method. However, in the case of simple machining and repeated machining tasks, the manual programming method makes quick programming possible. It also makes it possible to generate a part program quickly by modifying an existing program and using macro programming. Moreover, depending on the programmer's ability, it is possible to generate a part program for unusual and specific shapes.

The automatic programming method, where a computer is used, was developed to overcome the above-mentioned problems with the manual programming method. The automatic programming method makes it easy to machine parts with complicated or 3D shapes. It also makes it possible to generate the large part programs in a short time. In addition, with computer simulation, it makes it possible to detect and modify machining errors before actual machining begins.

8.3.3 Automatic Part Programming

The automatic programming method can be classified into the language-type programming method and the conversational programming method. In the language-type programming method, the machining sequence, part shape, and tools are defined in a language that can be understood by humans. The human-understandable language is then converted into a series of CNC-understandable instructions. In the conversational programming method, the programmer inputs the data for the part shape interactively using a GUI (Graphical User Interface), selects machining sequences, and inputs the technology data for the machining operation. Finally, the CNC system generates the part program based on the programmer's input. Typically conversational programming can be carried out by an external CAM system and a symbolic conversational system that is located either inside the CNC system or in the external computer. In this book, the implementation of symbolic conversational programming systems embedded in the CNC will be addressed in detail.

8.3.3.1 Language-type Programming

Language-type programming is the method in which a programmer edits a part program using language-type instructions that the user can easily understand. As the

manual programming method is similar to assembly language programming, so the language-type programming is similar to programming in BASIC or FORTRAN. For language-type programming, APT, EXAPT, FAPT, KAPT, and COMPACT II have been widely used.

- *APT (Automatically Programmed Tool)*

APT, which was developed in the USA in the 1960s, is the most famous system for the language-type programming tool and has the greatest number of functions. APT allows representation of various geometries, such as line, circle, ellipse, sphere, cylinder, cone, tabulated cylinder, and general two-dimensional surfaces. By using APT, it is possible to generate programs for 3-axis, 4-axis, and 5-axis machining, including rotation control for spindles and machining tables.

Figure 8.5 shows the structure of a part program in APT. The part program consists basically of four parts; 1) the shape definition part where the shape for the machined part is specified, 2) the motion definition part where the tool paths are specified, 3) the post processor part where cutting conditions and the characteristics of the CNC system are specified, and 4) the Auxiliary part where auxiliary data such as tool size, workpiece number, and so on is specified.

- *EXAPT*

EXAPT was developed in Germany. There are three kinds of EXAPT; EXAPT I for position control and linear machining, EXAPT II for turning, and EXAPT III for milling such as two-dimensional contour machining and one-Dimensional linear machining. EXAPT is very similar to APT but without workshop technology. EXAPT decides automatically how many tools are needed by considering the material of the workpiece, required surface roughness, and the shape of the hole specified by the programmer. It calculates automatically the spindle speed and feedrate. In EXAPT II, with user specification of the shape of the blank material and machined part, all machining operations including the machining allowances are generated automatically. On the other hand, it is necessary to register the pre-specified data because appropriate spindle speed, feedrate, and cutting depth can be varied according to the machine and tools. Because EXAPT generates automatically not only the tool path but also machining operations and cutting conditions, it is easier to use than APT. However, the kinds of machineable part shape that can be handled are more limited than with APT.

- *FAPT*

FAPT was developed by FANUC and is similar to APT. FAPT can be used in carry-on exclusive programming equipment. By using particular programming software such as FAPT Turn, FAPT Mill, and FAPT DIE-II, part programs for turning, milling, and die and mold machining can be generated easily. The FAPT Turn/Mill system has the following characteristics.

FAPT turn is a software library for turning. For part programming, the coordinate system of the rotation axis of the workpiece is defined as the Z-axis and the radius direction of the workpiece is defined as the X-axis. It is possible to program based on both diameter and radius values of the X-axis. FAPT turn provides 1) roughing, 2) finishing, 3) grooving, and 4) threading as metal-removal operations. The

```

CLPRNT
LI82 =LINE/6.25,-1.0,2.0,0.25,-1.0,2.0
LI83 =LINE/0.25,-1.0,2.0,2.0,3.5,2.0
LI84 =LINE/2.0,3.5,2.0,6.7525,1.1319,2.0
CI58 =CIRCLE/6.2507,0.125,2.0,1.125
LI85 =LINE/6.2507,-1.0,2.0,6.25,-1.0,2.0

CUTTER/0.25,0.05,0.075,0.05,0.0,0.0,4.0
COOLNT/ON
SPINDL/1200
FEDRAT/1.0
OUTTOL/0.005
TLAXIS/0.0,0.0,1.0
    FROM/0.0,0.0,5.0
    RAPID
    GOTO/-0.1228,-1.255,3.0
THICK/0.0,0.13
    DNTCUT
    GOTO/-0.1228,-1.255,1.0
    GO/ON,LI82,TO,(PLANE/0.0,0.0,1.0,1.0),TO,LI83
    CUT
    INDIRV/0.3624454,0.932005,0.0
    TLLFT, GOFWD/LI83, PAST, LI84
    GORGT/LI84, TANTO, CI58
    GOFWD/CI58, TANTO, LI85
THICK/0.0,0.13,0.0
    GOFWD/LI85,ON,(LINE/(POINT/6.25,-1.255,1.0),PERPTO,(LINE/$
    6.2507,-1.255,1.0,6.25,-1.255,1.0))
THICK/0.0,0.13
    GOFWD/LI82,PAST,LI83
    GORGT/LI83,PAST,LI84
    GORGT/LI84,TANTO,CI58
    GOFWD/CI58,TANTO,LI85
THICK/0.0,0.13,0.0
    GOFWD/LI85,ON,(LINE/(POINT/6.25,-1.255,1.0),PERPTO,(LINE/$
    6.2507,-1.255,1.0,6.25,-1.255,1.0))
THICK/0.0,0.13
    GOFWD/LI82,PAST,LI83
    TLON,GORGT/(LINE/-0.1228,-1.255,1.0,2.0,1.0,1.0),ON,(LINE/$
    POINT/2.0,1.0,1.0),PERPTO,(LINE/-0.1228,-1.255,1.0,2.0,1.0,1.0))
FINI

```

Fig. 8.5 APT program structure

tool nose compensation such as leaving finish allowance based on the machining tolerance and tool radius is possible. In addition, the tool path can be displayed graphically.

FAPT Mill is the automatic programming system for generating a part program for milling. It supports drilling, 2.5D machining of shapes made from lines and arcs, 3D machining of shapes made from spheres, cylinders, cones, and slanted planes. Free-form curves made using discrete points and pattern drilling, which is a repetition along a pattern element such as a line, arc, or grid, are possible. During simulation, the tool path can be displayed on the *XY* plane, *YZ* plane, *ZX* plane, or on an arbitrary plane projected from an arbitrary direction. In FAPT Mill:

1. it is possible to define a variety of geometries based on point, line, arc, slant plane, cylinder, cone, and sphere.
2. it is not necessary to define extra geometries for generating desired shapes.
3. it is possible to specify tool movement with a descriptive geometry name.
4. Tool radius compensation (left/right) and subroutine calls are possible.
5. variables and a variety of mathematical functions, such as the four arithmetical operations and trigonometric functions, can be used.

Apart from these, other programming languages, such as COMPACT-II, have been developed. However, the basic concept of these is similar to that of APT.

8.3.3.2 Conversational Programming

In order to carry out manual programming or language-type programming, a programmer must know the program instructions, and this makes the generation of part programs difficult. To overcome this problem, creation of part programs without knowledge of detailed program instructions needs to be possible. Due to this requirement, conversational automatic programming systems were developed that enable a programmer to generate tool paths by selecting machining features and operations as well as inputting data and following the system's instructions. In general, the conversational programming system category includes systems executed outside the CNC system in order to generate part programs for two-dimensional contours and three-dimensional free-form surfaces, so-called CAM (Computer-Aided Manufacturing). There are various examples of this type of system, such as CATIA, MasterCAM, EdgeCAM, so on.

As the above-mentioned conversational programming system is an offline system, a part program is generated on an external computer rather than on the CNC system. Because of this, the part program has to be transferred to the CNC system via a DNC system. Therefore, the creator of the part program and the operator of the part program can be different and so, in practice, it can be difficult to apply data optimization to the part program. In addition, in the case of simple machining, the usage of a CAM system reduces productivity. Accordingly, with the improvement in CPU and graphic performance, the symbolic conversational programming method,

which enables programmers (including novices) to generate part programs quickly and accurately on the shop floor in order to overcome the disadvantages of CAM systems, has been widely used.

In general, the symbolic conversational programming method is called WOP (Workshop Oriented Programming) or SOP (Shopfloor Oriented Programming). As shown in Table 8.1, this has different characteristics compared with other programming methods. It has been widely used on the shop floor and has a good effect on productivity. In this text book, the design and development of Shopfloor Oriented Programming systems embedded in CNC systems and used on the shopfloor will be addressed in detail.

Table 8.1 Comparison between programming methods

	Advantage	Disadvantage
EIA/ ISO	Easy to apply to simple operations such as tapping, drilling Basic function of CNC equipment.	Full knowledge of G-code required. Knowledge of geometry/mathematics needed for calculating toolpaths.
CAM	Possible to specify complicated shape. Possible to generate programs for various machines with one package.	Very expensive and requires expert. Impossible to feed back programs optimized on shopfloor.
Symbolic	Experienced person can use easily. Easy to create part program. Possible to feed back program optimized on shopfloor.	Program can be used only on a particular machine. In order to apply program to different machine, re-programming required. Programming for complicated parts is restricted.

The shopfloor programming system in CNC can be widely used for generating a part program on a variety of machine tools. In particular, when this programming system is applied to machines that produce parts with simple 2D, 2.5D, and primitive 3D shapes, it is possible to improve productivity and flexibility.

Considering that an operator edits the part program at the front of a machine, off-line CAM systems are more appropriate than shopfloor programming system in the case of the milling, for which it takes a long time to specify the part shape. However, shopfloor programming systems can be applied to wire-EDM or turning where the part shape is simple. In particular, the usefulness of the shopfloor programming system can be maximized when it is applied to turning machines with milling functions. Figure 8.6a shows how a turning machine with milling function can machine a milling feature on the end of cylinder. Figure 8.6b shows how a turning machine can generate a groove on the surface of a cylinder. To carry out the machining shown in Fig. 8.6 it is necessary to make a part program whereby the rotation of the spindle and the movement of the turret or tool post are controlled simultaneously. In practice,

even experts have difficulty in creating part programs for turn-mill machining manually. However, if a programmer uses the shop floor programming system, he/she can generate a part program by merely entering the feature geometry data and cutting depth for the machining shown in Fig. 8.6a and by merely entering the data of the groove shape and cutting depth for the machining shown in Fig. 8.6b. Thereby, the productivity of novice operators can be drastically increased by using shopfloor programming systems.

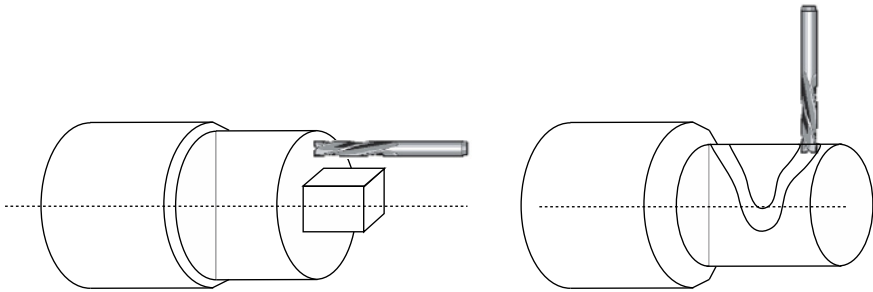


Fig. 8.6 Turning with milling

8.3.3.3 CAM Systems and Shopfloor Programming

Recently, with the use of PCs as MMI hardware, attempts have been made to embed PC-based CAM systems in the MMI and to replace shopfloor programming systems with online CAM systems. Because the ultimate goal is to edit the part program easily, they play similar roles. Each system consists of a graphical user interface, initialization module, contour module for specifying part profiles, machining cycle module for specifying machining operations and generating toolpaths, tool module for managing tools, simulation module for verifying the toolpath, and utility module for managing the part programs, as shown in Fig. 8.7.

The CAM system and Shopfloor programming system have slight differences in terms of function. The target machine of a shopfloor programming system is restricted to one machine or to machines of a similar type, but CAM systems can be applied to a variety of machines by providing a post-processing function. Therefore, in the case of a CAM system, a variety of machining conditions have to be considered. However, because only machine-specific functions are considered in the case of the shopfloor programming system, the function and architecture of the shopfloor programming system can be simpler than those of the CAM system.

However, there are too many problems caused by the difference between the design concepts to use CAM systems designed for offline usage on a CNC system. For example, a pointing device such as a mouse can be used for specifying the part profile and inputting the data to the CAM system. However, in the shopfloor pro-

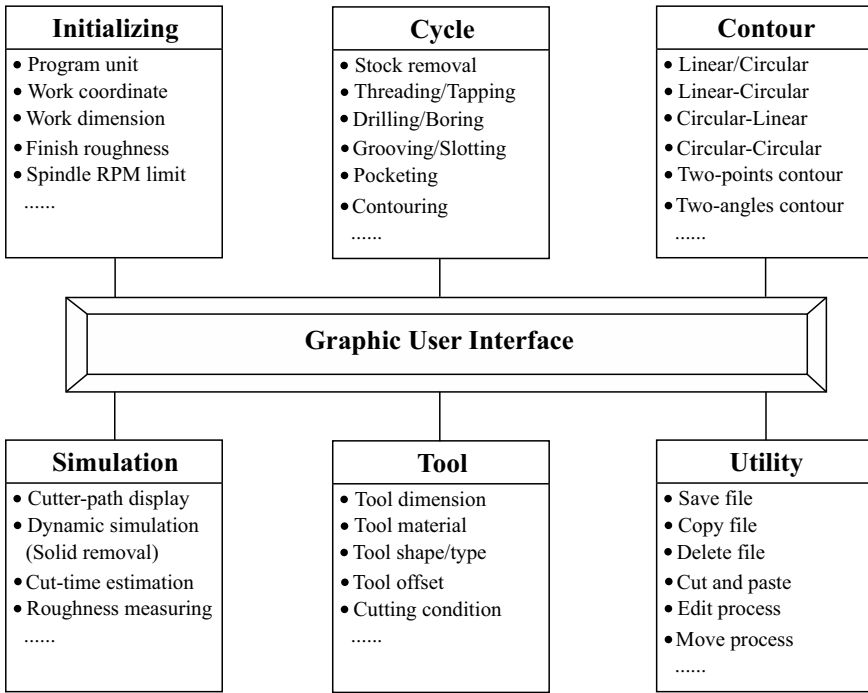


Fig. 8.7 CAM system structure

gramming system, a pointing device cannot really be used and only limited buttons are available. Also, because it is impossible for a user to edit a program at the front panel of a machine for a long time, quick and easy programming methods to specify part design and machining operations and enter key inputs are needed. In addition, it is necessary that the data modified is, after simulation, directly incorporated into the part program and saved.

Further, it is necessary to generate a machining cycle reflecting the parameters specified in the CNC system and it is also necessary to prevent programming that is outside the machine’s performance. Moreover, a way of helping a novice operator to decide input data (operation sequences, removal volumes (features), tools, and cutting conditions) or recommending input data values, is required.

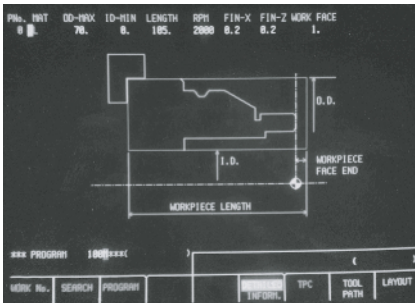
Therefore, the shopfloor programming system must provide a variety of methods to increase the program’s productivity by supporting the graphic user interface. The basic modules of the shopfloor programming system have the following properties and examples of modules are shown in Fig. 8.8.

1. Initialization module: In this module, the global variables, coordinate system (absolute/incremental), programming unit (inch/metric or diameter/radius), spindle data, feed unit (mm/rev or m/min), tool retract position, tool-retraction method, workpiece material, and machine data are specified, (see Fig. 8.8a).

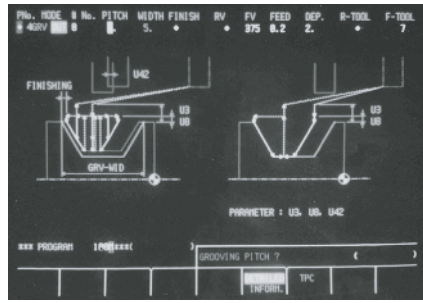
2. **Machining Cycle Module:** The specific machining scheme for milling, turning, and drilling is defined as one block. This block makes programming simple and efficient. The block is called a module. The machining operations such as roughing, finishing, drilling, slotting, and pocketing provide a variety of machining strategies. It is important to minimize the data that a programmer should input and select via the GUI during programming. This module is the core module of the conversational programming system (see Fig. 8.8b).
3. **Module for defining the part profile:** This module is used for defining the part shape. For this module, a different GUI is provided compared to that of a CAD system. This module provides the conversational contour programming GUI that consists of various graphic menus including line and arc geometries. In particular, in the case of finishing, individual surface finishes can be specified for each profile and feedrate can be computed automatically for each profile based on the surface finish. Of course, in the case of threading, slotting, and drilling, except for contour machining (profile machining), feature definition is carried out together with specification of machining cycles. The important thing for contour programming is that the dimensional data can be input easily without additional calculations during specification of the part profile. In addition, chamfer and round should be easily specified (see Fig. 8.8c).
4. **Tool module:** The tool module actually consists of two modules; the first is used for attaching the tool to the turret or tool magazine and the second is used for selecting the tool from the turret or tool magazine. One provides the GUI for specifying tool position, tool type, and tool geometry and the other provides the GUI for selecting the appropriate tool from the turret or tool magazine. Cutting conditions and spindle speed are automatically recommended by the system based on the tool, workpiece material and tool geometry. When the tool has been selected, a variety of data required for machining are automatically set using predefined values. If modification is needed, the programmer can modify these individually (see Fig. 8.8d).
5. **Toolpath verification module:** This module provides the functions for graphically simulating the toolpath of the program that was generated based on the programmer's input. By using this module, a programmer can verify the process from blank material to the final shape. Moreover, because this module displays the machining time (cutting time and non-cutting time) it can be used for optimization of the toolpath, (see Fig. 8.8e).
6. **Utility module:** this module provides the functions for copying, deleting, saving, and moving part programs, tool data files, and tool path files. It provides a text editor for modifying the file and moving, deleting, and editing operations for the generated programs, (see Fig. 8.8f)

The above-mentioned system can be summarized as a system that enables an operator to execute sequentially the steps of setting the program environment, setting the tool, selecting the machining cycle, and verifying the toolpath. The system provides a variety of graphical user interfaces for easily specifying the machining cycle

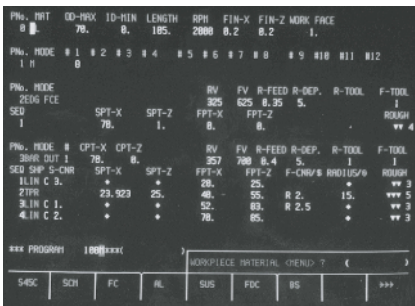
(a)



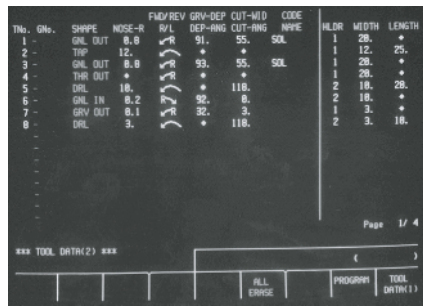
(b)



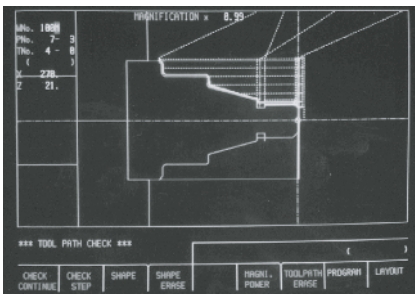
(c)



(d)



(e)



(f)

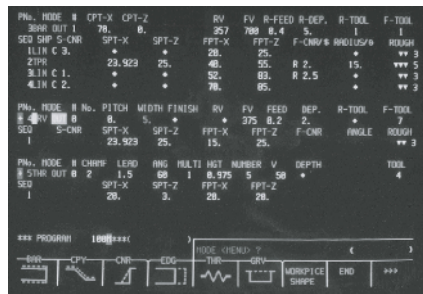


Fig. 8.8 CAM function displays (Courtesy of Mazatrol)

and machining feature, and generating a part program by interaction with the system without needing to memorize the programming method.

In order to help an operator generate, verify, and modify a part program quickly, CNC makers have developed and provided various shopfloor programming systems that can be operated only on their own CNC systems. For example, Siemens provided the Blue print programming system, the Support cycle programming system, and the WOP system. FANUC, Mazak, and Yasnac have provided the EZ-guide, the Mazatrol conversational programming system, and the Compact programming system, respectively.

These support various programming levels from low-level programming to high-level programming including complicated part machining. In the following section, using the Mazatrol system as an example, the characteristics of a shopfloor programming system will be addressed.

8.4 Mazatrol Conversational System

The Mazatrol Conversational Programming System is designed to enable a programmer to generate a part program quickly and verify it without needing either a manual or an assistant. It has been widely used in industry and provides various machining cycles that include the machinist's know-how. In addition, it provides a graphic interface (Fig. 8.8b) to enable programming without detailed programming knowledge.

8.4.1 Turning Conversational System

The machining cycles in terms of the machining mode and cutting mode, the key characteristic of the Mazatrol Turning Conversational Programming System, are summarized as follows.

1. *Feature Mode*: This denotes the machining cycles that are provided in conversational programming system. In this system, twelve machining cycles are provided as machining cycles, as shown in Fig. 8.9. As can be seen from the figure, the twelve cycles are as follows:

- **BAR**: This denotes the operation for machining a cylindrical part by turning. This cycle is used for rough machining of an arbitrary part.
- **CPY**: This is used for finish machining of a specified part with finishing allowance.
- **CNR**: After finish and rough machining, an undercut area can be left due to the tool's shape. This cycle is used for machining the undercut area.
- **EDG**: This cycle is used for machining the end face of the cylindrical part.
- **THR**: This cycle is used for threading.
- **GRV**: This cycle is used for machining a groove with arbitrary shape.

- MTR: This cycle is used for cutting in the part.
- DRL: This cycle is used for drilling a hole.
- TAP: This cycle is used for tapping.
- MNP: This is used for generating a part program in manual mode in order to machine special features that are not included specifically in this list.
- MES: This is used for measuring the machined part on the machine after machining has been completed.
- M: This is used for setting M-codes for controlling the machine behavior other than the servo motors.

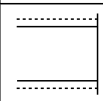


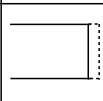

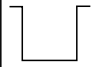

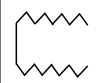
1. BAR	2. CPY	3. CNR	4. EDG	5. THR	6. CRV	7. MTR	8. DRV
							
9. TAP	10. MNP	11. MES	12. M				
	(Manual)	(Measurement)	(Auxiliary)				

Fig. 8.9 Machining cycles

2. *Cutting Feature:* After selecting the feature mode, the cutting method should be decided. For example, the rough machining mode feature (*i.e.* BAR) should be followed by inner contouring, outer contouring, facing, and back facing depending on the machined region. Therefore, the Cutting Feature is restricted by the type of Feature Mode. The relationship between Feature Mode and Cutting Feature is shown in Fig. 8.10. When BAR, CPY, CNR, EDG, THR, or GRV are selected, eight kinds of Mode Feature can be selected. In the case of MTR, only OUT (outer contouring) and IN (inner contouring) can be selected. In addition, because DRL and TAP can be applied in the face, selection of Mode Feature is not needed.
3. *Machining Strategy:* In order to execute the operation selected from Mode Feature, it is necessary to decide on the machining strategy. The machining strategies that can be applied according to the Feature Mode are shown in Fig. 8.11. For BAR and CNR, the tool retraction method has to be selected. When THR is selected, six kinds of machining strategy can be selected. In the case of GRV, various groove shapes can be selected. Since the conversational programming system guides the choice of appropriate strategies depending on the feature and operation, even non-expert programmers can select the appropriate machining strategy.
4. *Tool and cutting condition:* After Feature Mode, Cutting Feature, and machining strategy have been specified, it is necessary to select the appropriate tool and decide on the cutting conditions. The tool is selected from the tool database that

[1]. Mode Feature: BAR, CPY, CNR, EDG, THR, GRV

1. OUT	2. [OUT]	3. IN	4. [IN]	5. FCE	6. [FCE]	7. BAK	8. [BAK]

[3]. Mode Featurig: DRL, TAP-Set as “FCE”

[2]. Mode Feature: MTR

1. OUT	2. IN

Fig. 8.10 Relationship between Feature Mode and Cutting Feature

[1] Mode: BAR, CNR

1. #0	2. #1	3. #2

[2] Mode: THR

1. #0	2. #1	3. #2	4. [#0]	5. [#1]	6. [#2]
Standard	Constant depth	Constant width	Standard	Constant length	Constant area

[3] Mode: GVR

1. #0	2. #1	3. #2	4. #3	5. #4	6. #5

[4] Mode: DRL

1. #0	2. #1	3. #2	4. #3	5. #0	2. #1	3. #2
Stationary hole drilling	Deep hole drilling	High-speed drilling	Stationary hole reaming	Through hole drilling	Deep hole drilling	High-speed drilling

[5] Mode: TAP

1. #0	2. #1	3. #2	4. #3	5. #4	2. #5
M	UM	PT	PF	PS	Special

Fig. 8.11 Machining strategies to be applied according to Feature Mode

is pre-specified based on the tools loaded onto the machine. The cutting conditions can be recommended automatically by the system according to the tool and workpiece material or can be input directly by the programmer.

5. *Machining Geometry*: The last step to input the feature data is to specify the machining geometry. The geometry of the feature can be made up from lines, slanted lines, convex arcs, concave arcs, and circle centers, see Fig. 8.12. The programmer selects the geometric elements that compose the feature and inputs their positions to define them fully. For each segment, surface roughness can be specified. If the programmer does not specify this, a default value, defined by a global variable, is set.







1. LIN	2. TPR	3. 	4. 	5.
				CTR (Center)

Fig. 8.12 Feature geometric elements

8.4.2 Programming Procedure

The programming procedure in the Mazatrol system is as shown in Fig. 8.13. The procedure is composed of three parts; the first is the header part where the part program number is specified and global data in the initialization module are defined. The second is the body part where a variety of information for machining, such as feature data, machining operation data, and cutting condition data, are defined. The last is the end part where the data for the task to be carried out for completing the program are specified.

In the header part, the material, diameter, and length of the workpiece, maximum spindle speed, finish allowance, and surface finish are specified as global data.

In the body part, the data for defining machining features are specified. First, the machining mode (*e.g.*, BAR, CPY, DRL, and TAP), called “Mode Feature” in the Mazatrol system, is selected and the machining part (in Mazatrol called “cutting feature”) relevant to the selected Mode Feature (*e.g.*, internal, external, and face) is selected. After that, the machining strategic data are specified and the tool and cutting conditions are selected. The cutting conditions can be selected automatically from a pre-specified database or selected manually by the operator. Finally, if it is necessary to specify the part profile depending on the selected Mode Feature (for example, bar machining, copy machining, and grooving machining), the shapes of the blank material and finished part are specified by inputting segment features.

The end part can be used optionally. In this part, the tasks that must be executed before completing the part program are specified. For example, it is possible to spec-

ify whether the number of a finished part is counted or the M-code for activating automation equipment is called. In addition, it can be optionally specified how often the part program is repeated or what program will be invoked for subsequent execution.

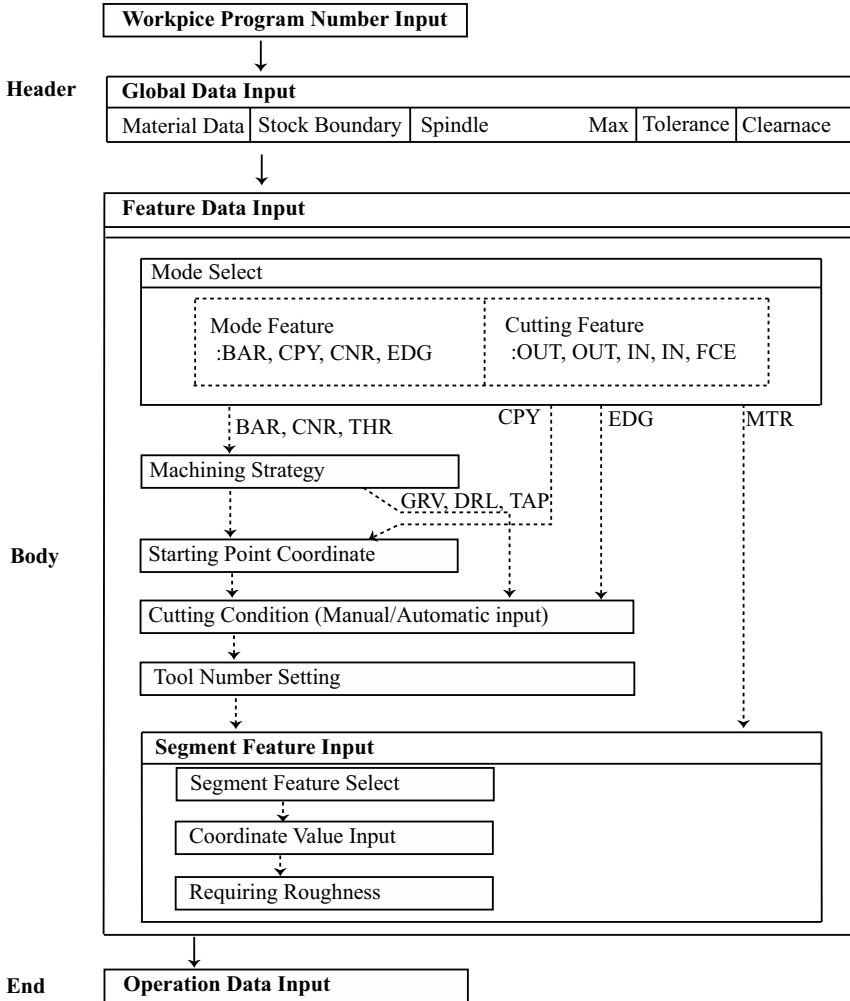


Fig. 8.13 Mazatrol programming procedure

8.5 Conversational Programming System Design

As the shapes of parts have become more complex and their accuracy has increased, so has part programming become more difficult. At the same time, on the shopfloor, the number of expert programmers has decreased. Because of this, a conversational programming system has become an essential function of an advanced CNC system. This conversational programming system must provide not only basic functions provided by the CNC user interface but also intelligent machining cycles based on a graphical user interface, strategy and cutting condition database based on expert know-how, tool path verification functions, and a variety of utilities.

A conversational programming system is designed as a system that:

1. can be used even by an inexperienced operator,
2. can generate a part program quickly with minimal key input,
3. can verify the generated part program in a short time,
4. can introduce the modified information easily into the generated part program, and
5. can be operated on the CNC system on the shopfloor.

8.5.1 Main Sequence for Design

The procedure for creating a part program in a conversational programming system can be summarized as shown in Fig. 8.14.

1. Start the conversational programming system by selecting the programming key in the MMI.
2. Set the initial data (global data) following the screen indications generated by the Conversational Programming System.
3. Select the particular operation and input the data via the GUI (graphical user interface) relevant to the selected operation. The shape of a part, machining strategy, tool, and cutting condition are given as input data.
4. After specifying all operations, generate the part program in standard G-Code or the manufacturer's own code by selecting the program generation key in the MMI.
5. Check the tool path or the finished part via the simulator.
6. If the verification result is not satisfactory, select the modification key and modify the data of the unsatisfactory operation.

Through the above-mentioned programming procedure, an operator inputs the data in steps 2 and 3 (these are represented by the gray boxes in Fig. 8.14) and the others are executed by the conversational program system. Editing the data in a parameter database is possible during programming and before programming. This database has the default values for the parameters for machining strategy, tool, and cutting conditions. In particular, the important thing is that the tool offset, which is

measured on a machine, is managed by the database that is connected to the conversational programming system. The value in the parameter database is used during selection of the machining feature. Figure 8.14 depicts the main components of the conversational programming system for turning. In the case of a milling system, the major components are the same and only the details about machining features, tool databases, and machining strategies are different.

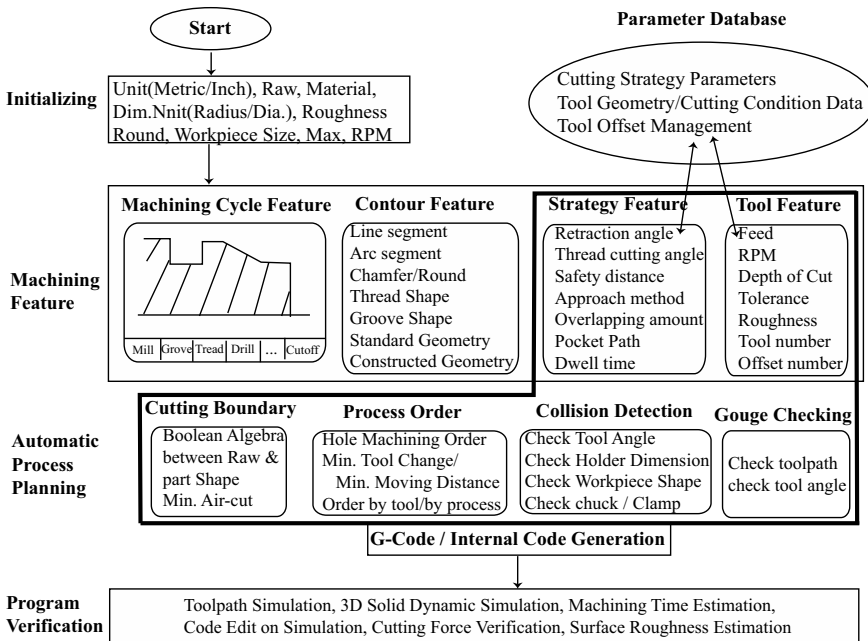


Fig. 8.14 Typical operation steps

Therefore, as mentioned above, the following are key points for designing a conversational programming system with an easy-to-use user interface.

1. For adequate design of the machining cycle feature that meets the machining characteristics of the machine, key functions are needed for minimizing user input by automatic recommendation of machining operations, helping non-experts to edit a program by automatic recommendation of cutting conditions, generation of the toolpath without tool interference, overcut, undercut, and aircut, and determining the operation sequence that minimizes the cutting time and tool change.
2. A unique method for specifying the part shape is needed. In order for an operator to generate a part program quickly at the machine, a simple and easy way of specifying the part shape is needed instead of an offline CAD system.

3. In addition, for realistic simulation, 3D graphics functions are needed. However, because this subject is outside the scope of this book, details of this are omitted.

In this book, the key design factors for a shopfloor conversational programming system will be summarized in terms of a milling system. The implementation of the machining cycle, which generates the toolpath from the machining feature, will also be addressed.

8.5.2 Key Design Factors

The initial setup, machining operation cycle, part profile drawing, tool management, utility, and operation management should be defined as key functions for a conversational programming system.

8.5.2.1 Initial Setup

In the initial setup module, not only global data that is used by the CNC system but also the coordinate system (absolute/incremental), programming units (inch/metric, diameter/radius), spindle data, feed unit (mm/rev, m/min), tool retract point, tool retract strategy, workpiece material and machine specification data are defined in this module. As shown in Fig. 8.15, workpiece geometry, start Z-point, Z safety plane, work coordinates, and clamp design are defined as well. The workpiece material is used together with a tool database for calculating the cutting conditions automatically. The workpiece geometry is used for displaying the initial material on the solid simulator. The working coordinates are used as the reference coordinates during real machining. Clamp design is used by the simulator for detecting tool collisions. Therefore, in order to use the advanced functions of the conversational programming system efficiently, it is essential to carry out an initial setup before specifying the machining operation cycle.

8.5.2.2 Machining Operation Cycle

Initially, how a tool approaches a workpiece, how it is retracted from the workpiece, and how an operation is terminated are specified regardless of the types of machining operation (*e.g.*, milling, turning, and drilling). The region to be machined can be arbitrarily divided and machined.

The machining operation cycle is the code block to command roughing, finishing, drilling, or grooving. It is used for making programming simple and efficient. The machining operation cycle supports various machining strategies and is designed for minimizing user input via interactive user interface. The majority of CAM systems have CAD functions as methods to specify the part shape and features. However,

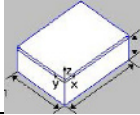
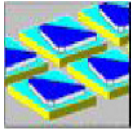
Categories	Contents
File type	Main Program / subprogram
Work material	Workpiece material is classified in terms of the hardness of material.
Work Size	Unit, Length, Width, Height, Face-off 
Initial point - Z	The Z position where tool and workpiece do not collide in the case of rapid traverse (G00).
Multi-mode	Pitch-X Pitch-Y or arbitrary points Xn, Yn  Number : $N \times M$
Work-Zero	Work Zero offset (G54, G55, G56, ...)

Fig. 8.15 Fundamental data for machining

with regard to shopfloor programming systems, CAD functions are inconvenient for an operator. So, for a shopfloor programming system, a different part specification method is used where machining features and operations are specified simultaneously used.

The machining operations for milling can be classified into three groups, each of which is composed of various machining operation cycles, as shown in Fig. 8.16.

Drilling cycle: As frequently used drilling operation cycles, drilling, boring, fine boring, back boring, tapping, reaming, step boring, and circular milling cycles are supported. In these cycles, it is possible for an operator to input the center of a hole and it is possible to generate automatically the pre-machining/post-machining sequence for a tool that has been selected by the automatic sequencing-tool module. Moreover, since a variety of patterns to specify the center of hole are provided for each cycle, patterned drilling cycles are possible with only one setting.

Profile machining cycle: This cycle is used for machining the specified profile by making a tool move along the specified profile. Linear-profile machining on a plane (top or side), chamfering of the specified profile, and engraving of characters on a plane are provided as profile machining cycles.

Milling cycle: As this cycle is most frequently-used in milling operation, facing, boss machining, pocketing and slotting are supported. As facing cycles, standard shapes

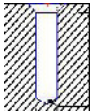
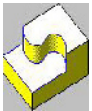
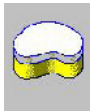
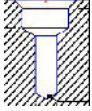
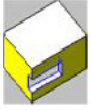

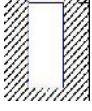
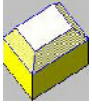
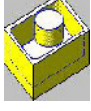
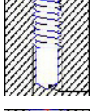

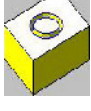
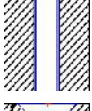
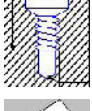
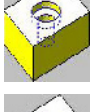
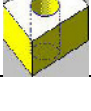
HOLE			PROFILE			MILL		
DRILL		Center deep hole	Line		R/L/C IN/OUT	Face		
Counter bore		F/B- Cbore	Side		R/L IN/OUT	Boss		Pocket
Bore		F/B- bore	Chamfer		R/L IN/OUT	Pocket		Pock Island
Tap			Engrave			Slot		Pock Valley
Ream								
Counter tap								
Step bore								
Circ-Mill								

Fig. 8.16 Classification of milling operation cycles

such as rectangular unidirectional, rectangular bidirectional, circular unidirectional, circular bidirectional, circular and rectangular island facing cycles are supported. The pocketing cycle is used for removing the interior of a particular profile. The pocketing cycles for standard shapes such as rectangular, circular, track, and slot profiles are provided. Furthermore, free pocketing cycles for complex profiles composed of linear profiles are also implemented. Contour (unidirectional) type or helical type (bidirectional) can be selected as toolpath generation strategies for optimal toolpaths.

8.5.2.3 Input Data List for Machining Operation Cycles

The user input needed to specify the cycles mentioned in the previous section is summarized in Tables 8.2, 8.3 and 8.4.

Table 8.2 Comparison between programming methods

1. Hole machining

Operation	type	Z-height	Diameter 1	Depth 1	Chamfer 1	Diameter (d2)	Depth (d2)	Chamfer 2	Pitch	Cutting Condition
drill	center drill dp.-hl. break	O	hole dia.	hole dep.	hole cham.	x	x	x	x	Ts,Td, Tch: Speed, Feed
count. bore	face back	O	c-bore Dia.	c-bore Dep.	c-bore Cham.	Drill Dia.	Drill Dep.	Bott. Cham.	x	Ts, Td Tch,Tm: S,F
Bore	face back	O	bore Dia.	bore Dep.	bore Cham.	Drill Dia.	Drill Dep.	x	x	Ts, Td, Tch ,Tb: S,F
Tap		O	Hoj.-ky.	Tap Dep.	Cham.	Drill Dia.	Drill Dep.	x	Pit.	Ts, Td, Tch ,Tt: S,F
Ream.		O	ream. Dia.	ream. Dep.	Cham.	Drill Dia.	Drill Dep.	x	x	Ts, Td, Tch ,Tr: S,F
Count. tap		O	Hoj.-ky.	Tap Dep.	Cham.	Cbore Dia.	Cbore Dep.	Bott. Cham.	Pit.	Ts, Td, Tch,Tm, Tt: S,F
Step bore		O	Bore Dia.	Bore Dep.	Cham.	Bore Dia.	Bore Dep.	Bott. Cham.	x	Ts, Td, Tch,Tb: S,F
Cir-Mill		O	Circ. Dia.	Circ. Dep.	Cham.	Drill Dia.	Drill Dep.	Bott. Cham.	x	Ts, Td, Tch ,Tm: S,F

* Ts: spot drill, Td: drill, Tch: chamfer, Tm: endmill, Tb: bore, Tt: thread

8.5.2.4 Machining Geometry Definition

In order to specify the shape and profile of a part, another user interface, different from and easier than that of a conventional CAD system, must be provided. For this, three kinds of method for part shape specification are provided, as shown in Fig. 8.17. The first is a method to design primitive geometric elements, rectangles, polygons,

Table 8.3 Comparison between programming methods

2. PROFILE

Operation	M-type	Start-Z	Cut-D	Cut-R	Chamfer	Fin-D	Fin-R	Cutting Condition
Line	R/L/C IN/OUT	O	O	O	O	O	O	Tr, Tf, Tch: S,F
Chamf.	R/L IN/OUT	O	kans. -D	kans. -R	x	x	x	Tch: S, F
Side Grv	R/L IN/OUT	O	O	O	O	x	O	Tr, Tf: S, F
Engr.	Font	O	Width	Height	x	Col. Span	Row Span	Tf: S, F

Table 8.4 Comparison between programming methods

3. Mill

Operation	M-type	Start-Z	Cut-D	Cut-R	Chamfer	Fin-D	Fin-R	Cutting Condition
FACE		O	O	x	O	O	x	Tr, Tf, Tch: S,F,Dd,Dr
Boss		O	O	O	O	O	O	Tr, Tf, Tch: S,F,Dd,Dr
Pocket	island valley	O	O	x	O	O	O	Tr, Tf, Tch: S,F,Dd,Dr
Slot		O	O	Slot width	O	O	O	Tr, Tf, Tch: S,F,Dd,

and ellipses, by specifying basic parameters. This is called the “standard geometry method”. The second is a method to design the contour profile by adding lines and/or arc profiles sequentially via a graphic menu (called the oriented geometry method). The third method, which is similar to that of a 2D CAD system, is a method to design profiles by cutting, connecting and copying points, lines and arcs (called the constructed geometry method).

In addition to the graphic-based conventional programming system, the software design should allow that connection elements, such as chamfers and rounds, to be specified when the profile (line or arc segment) is specified. To assign individual surface finish to each segment, it has to be possible to assign different feed rates to individual segments while profiles of a part are designed.

Method	Geometry	
Standard Geometry	Rectangle, Circle, Ellipse, Polygon	
Oriented Geometry	Line	Arc
	Line-Arc	Arc-Line
	Arc-Arc	2 Points
	2 Angles	
Constructed Geometry	Point (Cartesian/polar, circular, matrix, line) Line (parallel, perpendicular, tangent...) Circle (center + R, 3 line,...)	

Fig. 8.17 Part-shape specification methods

8.5.2.5 Tool/Technology Data

The Tool/Technology database recommends the appropriate cutting conditions in terms of the tool and workpiece material and the tool shape. Four kinds of database are supported for the tool used and tool sequence for hole machining.

1. *Tool database*: As shown in Fig. 8.18, this manages the data about tool shape and material. It provides the data for generating toolpaths and deciding cutting conditions.
2. *Cutting condition database*: this manages the data about cutting speed and feed according to tool type, material, and workpiece material.
3. *Tool sequence database*: This manages the machining sequence for efficient hole machining.
4. *Tool offset database*: This manages the data about tool offset.

8.5.2.6 Machining Strategy Data

To generate toolpaths using an operation cycle, it is necessary to decide the machining strategy once the machining features and tools have been decided. In Table 8.5, the data about the machining strategy that should be considered for milling operations are summarized. The usage of these for each operation cycle (hole/profile/milling or ‘H’, ‘P’, ‘M’ in Table 8.5) is checked.

Table 8.5 Machining strategy data

Parameters	Description	H	P	M	Remark
Return type	Tool retraction method (XY/Z, XYZ)	O	O	O	
CLD	Allowance amount when tool approaches	O	O	O	
RTD	Tool retraction distance	O	O	O	
Chamfer all. -R	Chamfer allowance along radial direction	O	O	O	
Chamfer all. -D	Chamfer allowance along depth direction	O	O	O	
Spot Depth	Depth of Spot Drill	O	x	x	
Through Hole	Selection of stop hole or through hole	O	x	x	
Through all.	Backside allowance of through hole	O	x	x	
Dwell	Dwell Time at bottom of hole	O	x	x	
Relief	Return relief amount in case of drilling and boring	O	x	x	Drill/Bore
Feed factor	Retract feed factor	O	x	x	Bore/Reamer
Mill type	Exact arc processing method	O	x	x	C bore/Cir-Mill
Tool end allowance	Tool end allowance in Boring	O	x	x	Bore
Tool end allowance 2	Tool end allowance in Back-Boring	O	x	x	Bore
Finishing allowance	Allowance of bottom in boring	O	x	x	Back C-Bore/Bore
Stop Hole allowance	Drilling depth allowance for threading, boring or reaming	O	x	x	Bore
Incomplete thread num.	Number of incomplete threads in tapping	O	x	x	
Interference	Checking the interference before machining (the number of blocks ahead that are examined)	x	O	O	
Corner	Machining method at edge. (Round/sharp/square/trang/fanuc)	x	O	O	
Cycle path	Straight/Zigzag with or without retract	x	x	O	Face
Pock path	para-contour/para-axial	x	x	O	
Cut type	Down/upward machining	x	x	x	
Direction	machining direction: any/CW/CCW	x	x	x	
Run-in	Approach: no/Arc/Parallel/Perpendicular	x	O	x	

Table 8.5 (continued)

Run-out	Approach: no/Arc / Parallel/Perpendicular	x	O	x	
Face R feed factor	Feed factor in radius direction.	x	x	O	Face
Face R allowance	Facing allowance in radius direction.	x	x	O	Face
Face allowance	Removal rate in radius direction	x	x	O	Face
Pock R feed factor	Feed factor in radius direction	x	x	O	Pocket
Boss outer Ffac	Feed factor for machining outside of Boss	x	x	O	Boss
Axial D-Ffac	Feed factor for machining in axial direction	x	x	O	Boss, Pocket
Pock R-Fac	Feed factor for full slot cutting in the case of pocketing.	x	x	O	Pocket
Overlap amount for closed shape	Overlap amount when approaching and retracting in closed shape	x	O	O	

8.5.2.7 Graphic Simulation for Verification

Graphic simulation is carried out by a path simulator for verifying tool paths and a solid simulator for verifying the machined shape. A path simulator displays toolpaths as a sequence of lines or arcs and is used for visual verification of the toolpath of a part program (see Fig. 8.19a). It provides the functions for checking for collisions between tools and clamps and editing a part program for correcting incorrect tool paths.

A solid simulator shows the change of part shape of a 3D solid model during machining. Also by using a solid simulator, it is possible to verify tool paths and analyze realistically the machined part (see Fig. 8.19b).

During the programming sequence, the screen displays complete operations. Therefore, if a verification result is different from the operator's expectations, it is possible to modify the operation and quickly correct the program during simulation.

The part shape is also displayed on the screen and regions that cannot be cut due to the tool geometry are checked and displayed. In particular, blank material and removal volumes are displayed simultaneously on the screen and whenever a particular operation is specified, the volume remaining after completion of the specified operation is displayed.

Tool interference due to the diameter of the specified tool is checked automatically. Because machining time (including cutting time and non-cutting time) is always displayed during simulation, the simulation function can be used as the tool for optimizing toolpaths.










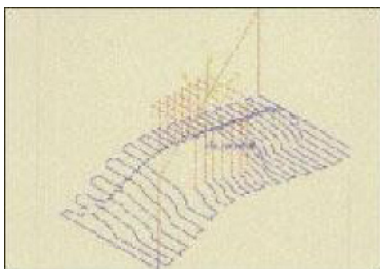
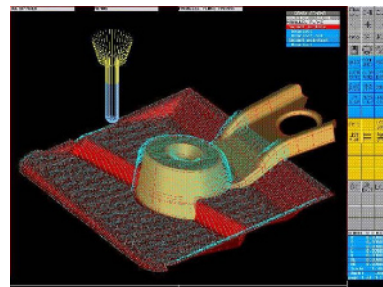
Tool	Feature	Tool data	Operation to which tool is applied
Center		Material, diameter, length, point, angle	Hole
Chamfer		Material, diameter, length, point, angle	Profile-chamfer
Drill		Material, diameter, length, point, angle	Hole
Bore		Material, diameter, length	Hole
Tap		Material, diameter, length	Hole
Reamer		Material, diameter, length	Hole
Face mill		Material, diameter, length, cutting teeth number	Mill-face
End mill		Material, diameter, len. flute num., tool type (flat, ball), ball radius	Hole, Profile, Mill
Side mill		Material, diameter, len. cutting teeth num., cutter length	Profile-side

Fig. 8.18 Milling tool database



(a)



(b)

Fig. 8.19 Graphical simulation

8.5.2.8 Operation Sequence Control

This module shows the specified operation cycles and enables an operator to modify and delete them while editing the operation cycle. It also enables addition of new operation cycles and operation sequence changes. It enables operators who are unfamiliar with process planning to generate consistent and efficient programs. Moreover, it is possible to store the generated program on memory or disk and use it whenever it is needed.

8.6 Development of the Machining Cycle

In this section, the implementation of turning manual G-code cycles and various machining cycles for conversational programming system will be described.

8.6.1 Turning Fixed Cycle

From the programmer's point of view, it is necessary that frequently used machining operations are defined in fixed format and used like subprograms when a part program is edited. A series of machining operations that are used repeatedly in NC machining are defined as one block that is called a "fixed cycle". The fixed cycle for turning can be classified into two types as shown in Table 8.6. Figure 8.20 shows G92, which is the simple fixed G-code for threading, and G76, which is the complex fixed G-code for threading. Compared with the tool path of the simple fixed cycle, that of the complex fixed cycle is complicated. However, it is relatively simple to generate the toolpath from the input data.

Table 8.6 Machining strategy data

Type	Code	Description	type	Code	Description
Single Fixed Cycle	G90	Turning (Cutting Cycle A)	Complex Fixed Cycle	G70	Finishing
	G92	Thread cutting		G71	Outer turning
	G94	Facing (Cutting Cycle B)		G72	Facing rough
				G73	Pattern repet.
				G74	Peck drilling in Z-axis
				G75	Grooving in X-axis
				G76	Thread cutting

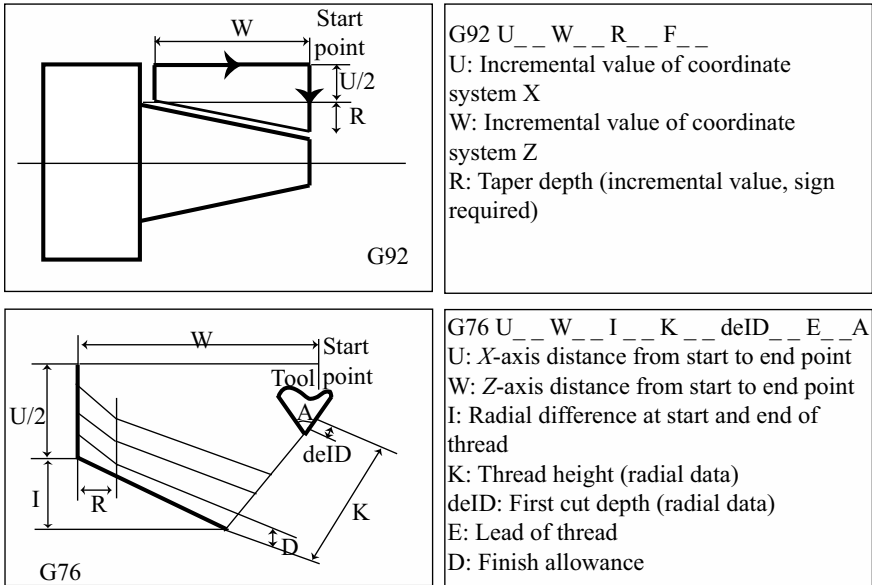


Fig. 8.20 Simple and complex G-codes for threading

8.6.2 Turning Cycle for Arbitrary Shape

8.6.2.1 Characteristics of Machining Cycles for Arbitrary Shapes

The G-code cycles mentioned in the above section are used for generating tool paths for a cylindrical part. In order to apply them successfully it should be assumed that the radius of a part increases or decreases consistently and that tool interference does not occur. However, a forged part or cast part typically has arbitrary shape and, in this section, the roughing cycle for these will be addressed. The roughing cycle generates the toolpath without tool interference by considering the geometry of the tool. It does not generate toolpaths for regions where material is absent in order to prevent cutting air. If there is a region where tool interference cannot be avoided, the region is not cut and remains to be cut in a subsequent operation.

For example, as shown in Fig. 8.21, the dotted line that represents the toolpath without air-cut is an appropriate tool path for obtaining the finished part from a cast workpiece.

8.6.2.2 Toolpath Algorithm

The cycle algorithm for generating an optimal toolpath is executed using eight steps. The steps are as follows.

In the first step, the workpiece shape and desired shape are specified (Fig. 8.22).

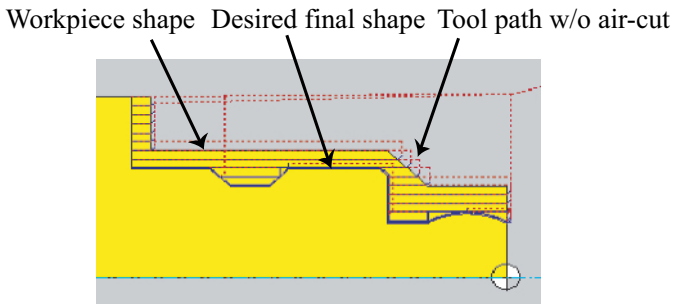


Fig. 8.21 Appropriate toolpath

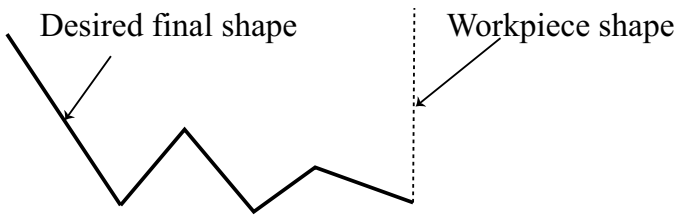


Fig. 8.22 Workpiece and desired shapes

In the second step, the collision-free region (machineable region) is calculated based on the cutting edge angle (side cutting edge angle and end cutting edge angle) of the tool, cutting angle, tool imaginary nose, tool type, tool holder's shape and workpiece shape, see Fig. 8.23.

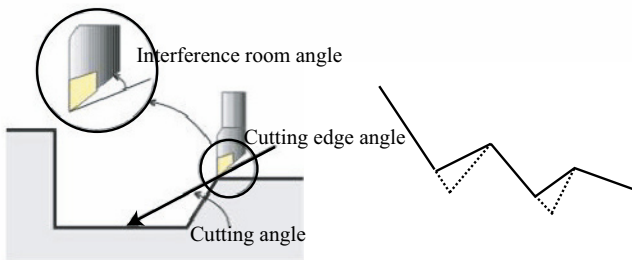


Fig. 8.23 Collision-free region calculation

The cutting angle is calculated as follows, based on the cutting edge angle and interference room angle.

$$\text{Cutting angle} = \text{cutting edge angle} - \text{interference space angle (in general, 3 or 5 degrees)}$$

Based on the computed cutting angle, the machineable region that prevents collisions between tool and workpiece and over-cut is calculated.

In the third step, the offset profile is generated by offsetting the machineable region from the second step within the finish allowance and with the tool nose radius.

In the fourth step, by combining the offset profile with the original profile of the part, a new profile is generated. In the case where the blank material is a cylinder, a new profile is generated by adding the linear profile of the cylinder, SA[], to the offset profile, S[], as shown in Fig. 8.24a. In the case where the blank material has an arbitrary shape, as with a cast part, the profile to be machined is created by combining the profile of the part, SA[], with the offset profile, S[], as shown in Fig. 8.24b.

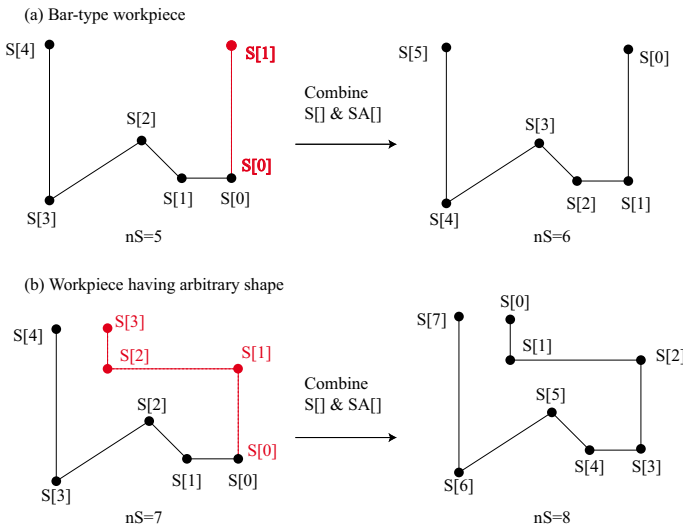


Fig. 8.24 Profile combination

In the fifth step, peak points and valley points are sought from the S[] obtained from the fourth step, and the total number of peak points is counted.

- Peak point (P_i): $\{ P_i | X_i \geq X_{i-1} \text{ and } X_i > X_{i+1}, \forall i \}$
 where, i is the index of a point on the profile.
 If $X_i = X_{i-1}$ and $X_i < X_{i-2}$, P_i is not peak point.
- Valley point (V_i): $\{ V_i | X_i \leq X_{i-1} \text{ and } X_i < X_i, \forall i \}$
 where, i is the index of a point of the profile.

In Fig. 8.24a, S[3] is a peak point and S[1] is a valley point. In Fig. 8.24b, S[5] is a peak point and S[4] is a valley point.

In the sixth step, the profile from the fourth step is divided into multiple profiles at the valley points and the divided profiles are stored in a buffer. In the case of the profile shown in Fig. 8.24a, the profile is divided at valley point S[1] as shown in Fig. 8.26.

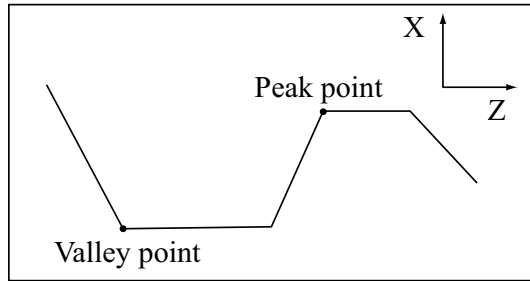


Fig. 8.25 Peak and valley points

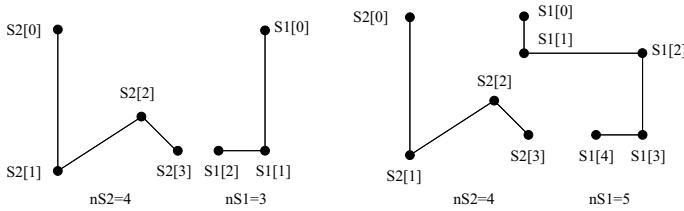


Fig. 8.26 Divided profiles

In the seventh step, the toolpath is generated based on the divided profiles and the specified cutting depth, as shown in Fig. 8.27a.

`Stock_removal_path () {`

- The peak points are sorted in terms of the X position and stored in `peak_array[]`.
 - The number of cutting layers is calculated ($num = (Xe - Xs) / feed + 2$).
 - From `S1[]` (first layer) and cutting depth are calculated the intersection point, `cross_S1[]` and `cross[]`.
- Where, `cross[]` is the path whose X position is constant and `cross_S1[]` is the intersection point between `cross[]` and `S1[]`.
- From `S2[]` and `cross[]`, `cross_S2[]` is computed.
 - The tool path is generated based on `cross_S1[]`, `cross_S2[]`, `S1[]`, and `S2[]`.

`}`

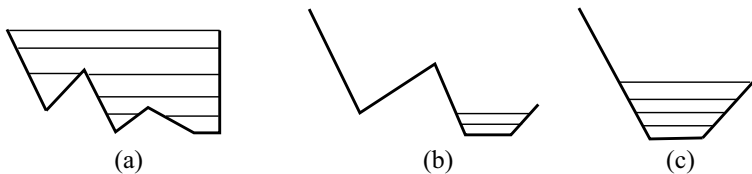


Fig. 8.27 Cutting toolpaths for turning

The eighth step, after the above steps have been completed, checks whether the current machined profile is the last peak profile. If yes, the cycle is terminated and, if not, the region to be machined is recalculated as shown in Figs. 8.27b and 8.27c.

Consequently, as the turning cycle for machining the part with arbitrary shape generates the toolpath automatically using the data on the workpiece shape, finished shape, and tools from the operator, it allows the unskilled operator to create the part program quickly.

8.6.3 Corner Machining Cycle

As mentioned in the previous section, an uncut area due to the tool shape can remain after completing the machining by the specified tool in the case of turning. This is why a toolpath is not generated in a region in which tool interference occurs. Therefore, in order to machine the uncut region after roughing, partial machining is executed after selecting a different tool. In the case of turning this is called “corner machining”.

The toolpath for corner machining is generated based on the intersection points (A and B) between the uncut area and the finished shape, cutting depth d , and finishing allowance k . The details of the algorithm can be summarized, with Fig. 8.28, as follows:

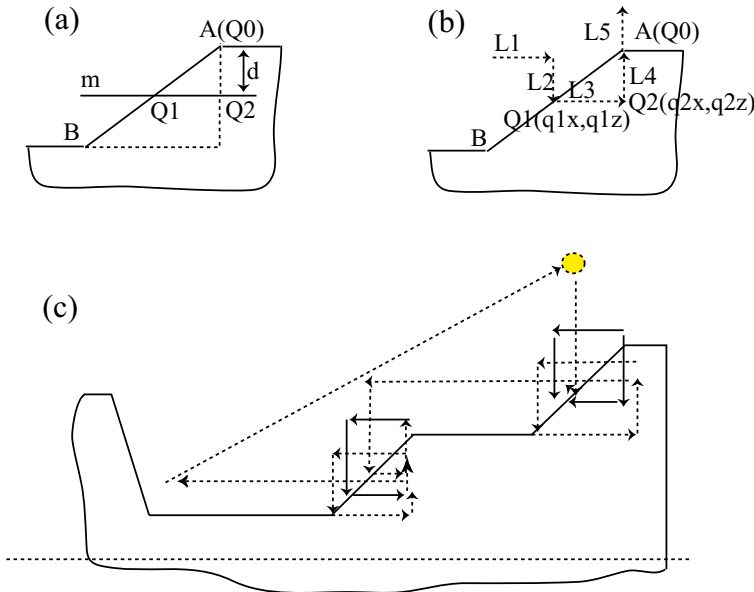


Fig. 8.28 Corner machining geometry

STEP 1. Cutting depth d and finish allowance k are input by the user and intersection points A and B are retrieved from the previous roughing cycle, (Fig. 8.28a).

STEP 2. The line m , which is at cutting depth d below the highest Z position of the corner, is defined. The intersection points $Q1$ and $Q2$ between the line m and the uncut area at the corner are calculated, (Fig. 8.28a).

STEP 3. The approach path for approaching the uncut area and the net-cut path for machining are generated. As shown in Fig. 8.28b, $L1$ and $L2$ are generated as the approach path for approaching and $L3$ and $L4$ for machining from $Q1$ to $Q0$ through $Q2$ are generated. In addition, $L5$ as a rapid path for retracting to the safety plane is generated.

STEP 4. The line m moves in steps of cutting depth d along the negative Z -axis. STEP 2 and STEP 3 are repeated until the Z position of the line m is smaller than the lowest Z position of the uncut area, (Fig. 8.28c).

STEP 5. If there is more than one uncut area, STEP 2, STEP 3, and STEP 4 are repeated for each uncut area. The join paths connecting the paths obtained from STEP 3 are generated and inserted. Finally, the rapid path for moving to the tool retract position is generated and inserted, (Fig. 8.28c).

This algorithm can be summarized as the procedure chart in Fig. 8.29.

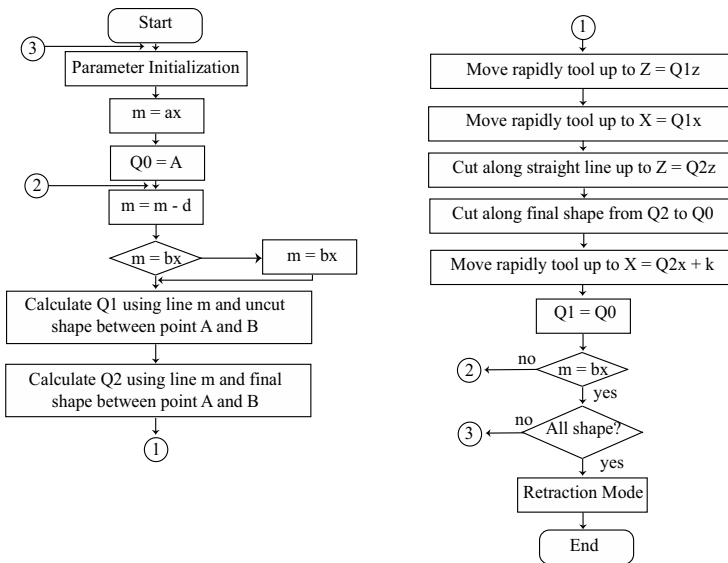


Fig. 8.29 Procedure chart for corner machining

8.6.4 Drilling Sequence

Typically, when the NC program for drilling multiple holes is generated, the programmer first classifies the holes into the groups depending on the hole shape and generates a program where the holes belonging to the same group are machined in a row. After completing machining of the holes in the group, holes belonging to another group are machined. In order to machine a hole, it is typical to use more than one tool. For example, in the case of tapping, center drilling, drilling, boring, and tapping should be executed one after the other. Therefore, if we consider the usage of tools related to drilling, the sequence of tools is considered as follows:

Group 1: $T_{11}, T_{12}, \dots, T_{1a}$

Group 2: $T_{21}, T_{22}, \dots, T_{2b}$

Group M: $T_{m1}, T_{m2}, \dots, T_{mm}$

where tools used in one particular group may be used in another group. Therefore, if the usage sequence of tools is well determined, it is possible to decrease the number of tool changes and hence the machining time.

However, if machining is executed group by group, the same tool may be used several times, which increases the tool change time and hence the total machining time. Therefore, it is necessary to make an NC program that reduces the number of tool changes.

Supposing that more than one hole can be classified into several groups and a particular tool can be used for holes belonging to different groups. In this case, if the tool is used for all the holes in a row, the number of tool changes is decreased as many as $M - 1$ times, where M is the number of groups where the tool is used.

The generation procedure of an NC program for drilling can be divided into three steps;

1. In the first step, the individual part program for each hole with different shape is generated.
2. In the second step, the usage sequence of the tools used in several groups is determined.
3. In the third step, the NC program for complete drilling is generated depending on the usage sequence of the tools.

In detail, the above steps are described as follows (Fig. 8.30). First, the usage sequence of tools is determined according to the shape of the holes. In the example, tools A, B, C, D, and E are used sequentially in the first part program P1. For the second part program P2, tools A, F, C, G, and E are used sequentially. For the third part program P3, tools H, I, C, J, and K are used sequentially.

After completing the first part program, the second, and the third, a check is made as to whether tools used for common operations exist. If these do exist, the tool that is used in the most operations is selected. In Fig. 8.30, this is tool C, which is used in three part programs.

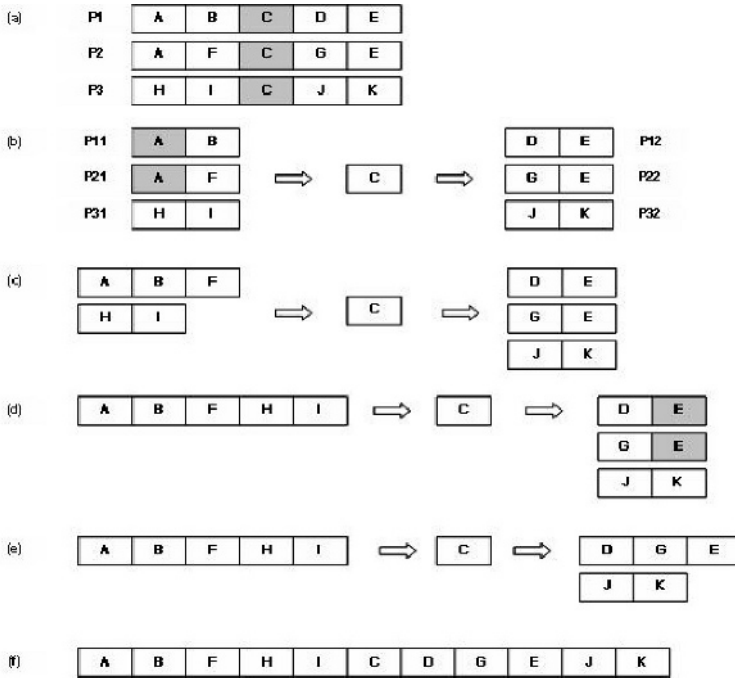


Fig. 8.30 Part programs to be executed

After selecting the common tool, each part program is divided into three parts; the part which should be executed before using the common tool, the part that is executed by the common tool, and the part that is executed after using the common tool. In addition, it is necessary to divide the tools in each group into 1) the tools used before the common tool, 2) the tools used after the common tool, and 3) the common tool.

In Fig. 8.30, P11, P21, and P31 are the part programs that should be executed before using the common tool. The tools used in P11, P21, and P31 are as follows.

- P11: A, B
- P21: A, F
- P31: H, I

In Fig. 8.30, P12, P22, and P32 are the part programs that should be executed after using the common tool and the tools used in P12, P22, and P32 are as follows.

- P12: D, E
- P22: G, E
- P32: J, K

After that, the above steps are repeated until no shared tool is found in P11, P21, P31, P12, P22, and P32,

The tool sequence in P11, P21, and P31 is determined as A'B'F'H'I.

The tool sequence in P12, P22, and P32 is determined as D'G'E'J'K.

Finally, the usage sequence of all the tools is determined as A' B' F' H' I' C' D' G' E' J' K. After determining the usage sequence of tools, the part programs are reassembled depending on the determined tool usage sequence.

8.7 Summary

The part that provides the user interface in the CNC system is the MMI unit. For the design of an MMI that allows a user to edit a part program, operate the machine, and monitor the machine status, it is necessary to consider ergonomics, design, and the user's aptitude as well as technological aspects.

In order to develop an MMI system, not only the user applications but also the design of the kernel layer for connecting to the NCK and the applications for monitoring the machine status, operating the NC, editing the program, and managing parameters are required.

The program-editing function, which is one of the key MMI functions, has advanced in a variety of versions developed by many CNC makers. Recently, conversational programming systems that can be used on the shopfloor have become a basic programming tool. The conversational programming system makes it possible for an unskilled user to generate a part program by allowing users to input data in an interactive way.

The core of the conversational programming system provides functions for generating interference-free toolpaths whose machining time is minimized and in which the number of tool changes is minimized, verifying tool paths and the finished part, and estimating machining time based on the minimum user input.