10 Series CNC
User Manual

Code: 45004452H
Rev. 15

PUBLICATION ISSUED BY:

OSAI S.p.A.
Via Torino, 14 - 10010 Barone Canavese (TO) – Italy

Tel. +39-0119899711

Web: www.osai.it

e-mail: sales@osai.it
service@osai.it

Copyright © 2001-2004 by OSAI
All rights reserved

Edition:November 2004

IMPORTANT USER INFORMATION

This document has been prepared in order to be used by OSAI. It describes the latest release of the product.
OSAI reserves the right to modify and improve the product described by this document at any time and without prior notice.
Actual application of this product is up to the user. In no event will OSAI be responsible or liable for indirect or consequential damages that may result from installation or use of the equipment described in this text.

SUMMARY OF CHANGES

General

This publication is issued with reference to Software Release 7.4 for the 10 Series systems (**E69**). The following modifications have been made to this revision.

PAGES	UPDATE TYPE
INDEX	Updated
Chapter 2	
page 10	Added new description: **PLUS SD000**
Chapter 3	
page 47	Added description of softkey: **SYSTEM SHUT DOWN**
Chapter 11	
page 8	Added new description: Execution of paramacros in MDI mode
Appendix A	
page 25	Added new codes: **SD214-SD215-SD216-SD217**

PREFACE

This manual describes how to operate a 10 Series CNC. It provides the end user with all the necessary information for using and programming the system.

REFERENCE

Consult:

- 10 Series CNC Product Specifications
- 10 Series CNC AMP - Software Characterisation Manual

For further information refer to:

- 10 Series CNC Programming Manual

SUMMARY

This manual describes how your 10 Series CNC works and explains how to operate the system. Each chapter deals with a separate topic.

1. **Features and specifications**

   This chapter lists the general hardware and software characteristics of 10 Series CNC's. It also describes the major features of the front panel.

2. **System start up**

   This chapter explains how to switch an and start up the system. It discusses the screens and messages displayed when the system runs the self diagnostic.

3. **User interface**

   This chapter describes the function of the softkeys, pushbuttons and controls available on the panel. An overview of all the softkeys levels available gives the user a reference guide of all the softkeys available. It provides a detailed description of the various video pages and discusses the optional operator console.

4. **Setting system variables and parameters**

   This chapter explains how to set the local and system variables used for program writing. It also describes how to set the machine operation parameters accessible with the MACHINE SET UP softkey.

5. **Establishing origins and homing the axes**

   This chapter describes how to home the axes and establish the machine origins. In addition, it provides a definition of the various reference and origin points managed by the control.

6. **Axes jogging and stop function**

   This chapter describes how to move the axes manually.

7. **Using tables**

   This chapter deals with the tables controlled by the system. It describes the Table Editor, a utility with which to compile and edit system tables.

8. **Table editor configurator**

   This chapter deals with the Table Editor, a utility for customising the tables visualised or printed by the system, the video pages and the softkey labels. It also discusses how to configure and handle customised tables.

9. **Tool management**

   This chapter explains how to define tool offsets and part origins an the spindle axis. In addition it describes how to manage the information stored in the Tool Data Base (tool mounting, transfer, etc.).

10. **Part program file manager**

    The chapter describes the part program management environment, the directories display, the copy, rename and edit of the part programs.

11. **Part program execution**

    This chapter explains how to set the machine for part program execution as well as the various execution modes.

12. **Machine Plot**

    This chapter introduces the MACHINE PLOT, a utility with which to visualise the tool path graphics in real time while the machine is executing the program.

13. **DOS Shell**

    This chapter describes the procedures for hard disk file management using the DOS operating system functions.

14. Security

This chapter describes a series of security functions that regulate the access to the system resources.

15. Peripherals

This chapter describes the programs used to configure and use the systems with communication ports connected to external devices (via a serial, parallel or Ethernet line).

16. Ethernet Communication

This chapter provides the most important information about Ethernet communication and describes how to configure the network and the network services.

17. Function tables

Lists the enabling tables.

A. Error messages

Lists the error codes and messages and the related recovery operations.

Warnings

For correct control operation, it is important to follow the information given in this manual. Take particular care with topics bearing one of the mentions: WARNING, CAUTION or IMPORTANT, which indicate the following types of information:

Draws attention to facts or circumstances that may cause damage to the control, to the machine or to operators.

Indicates information to be followed in order to avoid damage to equipment in general.

Indicates information that must be followed carefully in order to ensure full success of the application.

Terminology

Some terms appearing throughout the manual are explained below.

Control Refers to the 10 Series numerical control unit comprising front panel unit and basic unit.

Front Panel Is the interface module between machine and operator; it has a monitor on which messages are output and a keyboard to input the data. It is connected to the basic unit.

Basic Unit Is the hardware-software unit handling all the machine functions. It is connected to the front panel and to the machine tool.

END OF PREFACE

INDEX

FEATURES AND SPECIFICATIONS

INTRODUCTION 1-1
Processes 1-1
HARDWARE STRUCTURE 1-2
Central Unit 1-2
Operator Panel 1-4
WINMEDIA 1-7
General 1-7
Mass storage 1-14
FDU peripheral 1-14
CD-ROM peripheral 1-14
Keyboard connection 1-14
Mouse connection 1-14
Ethernet connection 1-14
Riser Card 1-14
External connections 1-14
Keyboard 1-15
Operator Console 1-16
SOFTWARE STRUCTURE 1-18
Utilities 1-20

SYSTEM START UP

SYSTEM POWER UP 2-1
DIAGNOSTIC SCREENS 2-1
Modules 2-4
EMERGENCY SYSTEM SWITCH-ON 2-5
DIAGNOSTIC MESSAGES 2-7
PASSED Tests 2-8
REPORTS about devices and actions in progress 2-9
Error messages 2-10

USER INTERFACE

KEYBOARD 3-1
Function keys 3-2
Alphanumeric Section 3-2

OPERATOR PANEL FUNCTION KEYS 3-4
Fixed Keys 3-4
Special keys 3-5
CONTROL BUTTONS 3-6
VIDEO 3-7
Main screen 3-9
Status information area 3-10
Axes data area 3-12
General data area 3-13
Part program data area 3-15
Large font display 3-16
Configurable screen pages 3-19
System status (SYS_STA) 3-20
Axes position (AXES_POS) 3-22
Program display (PRG_DISP) 3-23
Process status (PROC_STA) 3-24
Programmed code state (CODE_STA) 3-26
Axes offsets (AXIS_OFF) 3-27
Selecting a screen 3-27
Enlarging an elementary quadrant 3-28
Directory of Programs screen 3-29
Additional data windows 3-30
General rules about data entry windows 3-32
SOFTKEYS 3-35
Main menu 3-38
Auto 3-39
Manual 3-41
Part Program 3-42
Variables 3-43
OEM 3-43
Machine Set-Up 3-44
Tables 3-46
Diagnostics 3-47
Utility 3-48
OPERATOR CONSOLE 3-49
Operator Console Keys and Functions 3-49
Pilot Panel Selectors and Functions 3-52
Controls and functions of the WinMedia operator console 3-55

SETTING SYSTEM VARIABLES AND PARAMETERS

LOCAL AND SYSTEM VARIABLES 4-1
E variables 4-2
User variables 4-3
H variables 4-4
SN variables 4-5
SC variables 4-6
PLUS variables 4-9
Searching for pages and variable indexes 4-10
Using variables for calculations 4-11
SETTING MACHINE PARAMETERS 4-12
Dynamic Params 4-13
Dynamic limits 4-15
Program SET-UP 4-16
Block Retrace 4-16

Probe parameters 4-17
Axes Reference 4-18
Accuracy 4-19
Time/date 4-20

ESTABLISHING ORIGINS AND HOMING THE AXES

GENERAL 5-1
HOMING THE AXES 5-3
Manual Homing 5-3
Automatic Homing 5-3
DEFINING ORIGINS 5-5
DEFINING THE PART ZERO ON THE SPINDLE AXIS 5-8
Part zero on tool tip 5-8
Part zero on spindle nose 5-10
HOMING AND ORIGIN PRESETTING FOR DIAMETER AXES 5-11

AXES JOGGING AND STOP FUNCTIONS

JOG 6-1
Continuous Jog 6-2
Incremental Jog 6-2
DEFINING THE JOG INCREMENT 6-3
FEEDRATE OVERRIDE 6-4
ALTERING THE JOG INCREMENT 6-5
JOG RETURN TO THE PROFILE 6-6
SPINDLE SPEED OVERRIDE 6-7
STOP FUNCTIONS 6-8
Reset 6-8
Hold 6-8
EMERGENCY STOP 6-9
ACTIVE RESET 6-9
REMOVING THE TOOL AFTER AN EMERGENCY 6-12

USING TABLES

WHAT IS A TABLE? 7-2
USING THE TABLE EDITOR 7-3
Directory Window 7-5
Softkeys common to all Tables 7-6
Table Editor Keys 7-7
TABLE EDITOR FUNCTIONS 7-8
Opening a Table 7-8
Loading a Table 7-9
Altering a Table 7-10
Incremental Parameter Modification 7-11
Inserting a Record in a Table 7-12
Canceling a Record from a Table 7-13
Saving a Table 7-14
Printing a Table 7-14
Sorting Lines and Columns 7-15
Search for an Element 7-16
Changing the Measuring Unit 7-16

Table Backup 7-17
Restoring Tables 7-18
ORIGINS TABLE 7-19
TOOLS TABLE 7-21
TOOL OFFSETS TABLE 7-26
TOOL DATA BASE 7-29
MAGAZINE TABLE 7-32
USER TABLE 7-32

TABLE EDITOR CONFIGURATOR

GENERAL 8-1
CONFIGURATOR SCREENS 8-2
Main configurator screen 8-2
Configurator screen 8-3
SOFTKEYS 8-7
Softkeys in the main configurator screen 8-7
Configuration screen softkeys single tables 8-8
LOADING A CONFIGURATION FILE 8-10
SAVING A CONFIGURATION 8-11
RESTORING THE DEFAULT CONFIGURATION 8-12
AXES CONFIGURATION 8-13
CONFIGURING A TABLE 8-14
Configuring fields 8-17
LINKING TABLE FIELDS 8-20
UNLINKING TABLE FIELDS 8-22
ARRANGING THE TABLE FIELDS 8-23
CONFIGURING THE PRINTING OUTPUT 8-26
CONFIGURING HELP 8-28
CONFIGURING SOFTKEYS 8-29
QUITTING THE CONFIGURATOR ENVIRONMENT 8-30
PERSONALISING USER TABLES 8-31
User Table Manager screens and softkeys 8-32
Configuring a table 8-37
Configuring the table fields 8-39
Copying a table or the fields of a table 8-40
Copying a table 8-40
Copying table fields 8-41
Configuring HELP MESSAGES 8-42
Loading and saving a table on a file 8-43
Printing a user table 8-43

TOOL MANAGEMENT

GENERAL 9-1
TOOL PRESETTING 9-2
Tool presetting on an external system 9-3
Tool presetting on the machining part 9-4
DEFINING A MULTIPLE CUTTER TOOL 9-6
MANAGING THE TOOL DATA BASE 9-7
Transfering tool data between the Tool Data Base and the Tool Table 9-7
Entry of data fora new tool in the Tool Table (retrieved from Data Base) 9-7
Overwriting the data of a tool already in the Tool Table with the Data Base data. 9-9

PART PROGRAM FILE MANAGER

Main screen 10-1
SOFTKEYS 10-3
LINE EDITOR 10-4
Line Editor screen 10-4
Creating a new part program 10-7
Loading an existing Part Program 10-7
Loading a running Part Program 10-7
Characteristics of the Line Editor 10-8
Line Editor function keys 10-9
Writing new characters in a block 10-11
Opening a new line 10-11
Deleting lines 10-12
Retrieving lines 10-13
Configuration Editor 10-14
Numbering program blocks 10-16
Search for strings and block numbers 10-17
Cut & Paste 10-18
Inserting a part program 10-20
Saving a part program 10-20

PART PROGRAM EXECUTION

SETTING PART PROGRAM PARAMETERS 11-1
EXECUTION OF A PART PROGRAM 11-4
Selecting and activating a part program 11-4
Automatic execution 11-4
Semiauto (block-by-block) execution 11-5
Multiblock retrace 11-6
Executing blocks from keyboard 11-7
MDI execution without an active part program 11-7
MDI execution with an active part program 11-8
Executing part of a program 11-9
Modifying blocks in BLK/BLK mode 11-10
Restoring a part program 11-10
Search for a string 11-11
Overriding the programmed feedrate 11-12
Overriding the spindle speed 11-13
Overriding the rapid 11-14
Execution in *Dry Run* mode 11-14
MEMORY SEARCHING 11-15
Automatic searching 11-15
Searching for a pre-set block 11-16
Restarting a working cycle 11-16
Searching mode 11-17

MACHINE PLOT

MACHINE PLOT USED WITH AXES CONNECTED 12-1
User interface with axes connected 12-2
Machine Plot video page with axes connected 12-3
Softkeys active in axes connected mode 12-4
SETTING MACHINE PLOT PARAMETERS WITH AXES CONNECTED 12-5

DRAWING A PROFILE WITH CONNECTED AXES 12-9
CHECKING THE GRAPHICS 12-10
ZOOM 12-11
CAPTURING POINT COORDINATES 12-13
MACHINE PLOT WITH AXES DISCONNECTED (*DRY RUN* MODE) 12-15
Configuration and functionality in use with axes disconnected 12-15
Video page with axes disconnected (*Dry Run* Active) 12-16
Softkeys active in axes disconnected mode (dry run) 12-17
SETTING THE MACHINE PLOT PARAMETERS WITH AXES DISCONNECTED 12-18
VERIFY 12-21
CLEAR SCREEN 12-21
EXAME GRAPHIC 12-21

DOS SHELL

LAUNCHING DOS SHELL 13-1
DEFAULT PARAMETER VALUES 13-2
DISPLAYING A DIRECTORY 13-4
DOS SHELL COMMANDS 13-6
ABORT 13-7
BACKUP 13-7
COPY 13-8
DELETE 13-9
DIRECTORY 13-10
EDIT 13-11
EXIT 13-11
FORMAT 13-12
HELP 13-13
MKDIR 13-14
PRINT 13-15
RENAME 13-16
RESTORE 13-17
SHOW DRIVES 13-18
XCOPY 13-19

SECURITY

USER LEVELS 14-2
ABILITY TO USE SYSTEM FUNCTIONS 14-3
ABILITY TO INSTALL OPTIONS 14-4
ENABLING THE SECURITY ENVIRONMENT 14-5
SET LEVEL 14-6
SET PASSWORD 14-7
SET SECURITY 14-8
SHOW SECURITY 14-10
PRODUCT INFO 14-11
PRODUCT KEYS 14-11
INSTALL 14-12
SHOW OPTION and SHOW FIXUPS 14-13

PERIPHERALS

ACTIVATING PERIPHERALS 15-2

INTERLNK **15-3**
Configuration 15-3
InterInk Configuration for 10 Series System 15-3
InterInk Configuration Personal Computer 15-7
PRINT DEVICE **15-8**
REMOTE BOOT **15-9**

ETHERNET COMUNICATION

INTRODUCTION **16-1**
COMMENTS ON THE NETWORKS **16-3**
Definitions 16-3
Network topology 16-5
CONFIGURATION OF THE NETWORK **16-7**
General view 16-7
Network configurator 16-7
FUNCTIONS OFFERED BY THE NETWORK **16-8**
File Transfer 16-8
Server services 16-8
Access from a client 16-9
Exchanges of messages between tasks 16-9
USER INTERFACE **16-12**
Access to the configuration environment 16-12
Network configurator screen displays 16-13
Main screen 16-13
First screen display: *Advanced Level* 16-13
Second screen display 16-14
Third screen display (optional) 16-16
NETWORK CONFIGURATOR SOFTKEYS **16-18**
Level selection (LEVEL Softkey) 16-20
Channel configuration (CHAN GENERAL Softkey) 16-20
Configuration of server type services (SERVER Softkey) 16-22
Configuration of client type services (CLIENT Softkey) 16-24
CONFIGURATION OF TASK-TO-TASK TYPE SERVICES **16-26**
Enabling Task-To-Task communication 16-26
Defining a local environment as sharable in the network (SHARED ENV softkey) 16-26
Connection to a remote environment (REMOTE ENV. Softkey) 16-28
Immediate connection of remote environments (Softkey ENVIR. CONNECT) 16-30
Automatic connection of remote devices (Softkey SERVICES CONN.) 16-31
USE OF THE SERVICES IN THE NETWORK **16-32**
Network activation on the 10 Series system 16-32
Enabling the net on a Windows PC 16-32
ACCESS TO A SERVICE FROM DOS SHELL **16-33**
Display of the drives 16-33
ACCESS TO A PART PROGRAM **16-34**
DOS REAL TIME ENVIRONMENT **16-34**

FUNCTION TABLES

FUNCTION TABLES: BYTE 0 - Reserved 17-1
FUNCTION TABLES: BYTE 1 - Reserved 17-2
FUNCTION TABLES: BYTE 2 - Maintenance 17-2
FUNCTION TABLES: BYTE 3 - Directory Protection 17-3

FUNCTION TABLES: BYTE 4 - System Tables Protection 17-3
FUNCTION TABLES: BYTE 5 - Configuration 17-4
FUNCTION TABLES: BYTE 6 - PLUS Execution 17-4
FUNCTION TABLES: BYTE 7 - Part Program Development 17-5
FUNCTION TABLES: BYTE 8 - Execution of CNC Functions 17-5
FUNCTION TABLES: BYTE 9 - Reserved OEM 17-6

ERROR MESSAGES

USER INTERFACE ERRORS A-2
PART PROGRAM ERROR MESSAGES A-5
TABLE EDITOR ERRORS A-6
DOS SHELL ERROR MESSAGES A-11
SECURITY ERRORS A-13
EMERGENCY MESSAGES A-15
MESSAGES FOR D.S.I. ANOMALIES A-17
MESSAGES ON FASTWIRE ANOMALIES A-17
RESERVED MESSAGES A-17
HARDWARE DIAGNOSTICS ERRORS A-18
Base unit (main CPU) A-18
Other boards A-19
SOFTWARE DIAGNOSTICS ERRORS A-21
Operating system A-21
Software Configuration Errors A-23
APPLICATION ERRORS A-27
EDITOR ERRORS A-28
ERROR FROM PERIPHERALS A-31
ERRORS COMMON TO ALL OPERATOR PANELS A-32
CONFIGURATOR ERROR MESSAGES OF NETWORKS A-34
ERROR MESSAGES DURING NETWORK BOOTSTRAP A-36

END INDEX

FEATURES AND SPECIFICATIONS

INTRODUCTION

10 Series is a family of state-of-the-art controls capable of meeting a vast range of standard and non standard application requirements: milling machines, grinding machines, wood-, glass- and marble-working machines, oxygen cutters, etc. The upper segment in 10 Series can control up to 32 digital axes.

10 Series systems are the result of the most advanced technologies (32 bit microprocessor, surface mounting, etc.). The powerful multi-task, real time, event driven, operating system is enhanced by an open hardware and software architecture that permits MTB's to personalise the application by incorporating custom-specific packages. Access to the sophisticated NC functions is provided by an easy-to-use user interface that can be tailored to the MTB requirements.

Processes

The word "process" usually refers to the management of a machine tool: control of the machine axes, execution of machining programs, activation and control of the machine logic, etc. More in general, the notion of process may include any auxiliary task supporting the operation of the main machine: tool change, part loading/unloading, statistic calculations, report production, etc.

Multiprocess capabilities are among the major assets of 10 Series CNC, which can manage up to twenty processes. When synchronised, processes can be managed faster and more flexibly than with separate control systems.

HARDWARE STRUCTURE

The hardware structure of 10 Series systems is made up of different modules that can be combined to satisfy the application requeriments.

Two of these modules are present in every configuration:

- the operator panel
- the central unit

Central Unit

The central unit may be of the *monoboard* type, i.e all in one board, or consist of a rack that accomodates all of the electronic boards:

- the system CPU
- the axes control board(s)
- the I/O board(s)
- the PLC card

In addition, the system may include a series of optional modules picked up from the following list:

- one or more Pilot Panel operator consoles connected to the Central Unit
- a teach pendant connected to the central unit via the serial line
- up to two electronic handwheels connected to the encoder inputs available on the axes board
- local and remote I/O modules:
  - high density I/O (HD I/O)
  - medium density I/O (OSARING)
  - digital I/O
  - analog I/O
  - digital and analog I/O's for INTERBUS

All the I/O modules and the operator consoles are connected to the central unit over an optical fiber ring that protects the system against the electrical noise produced by the machine tool.
The I/O ring may include up to 1000 I/O points.

For more information about the system modules and how to connect them, please refer to the 10 Series Product Specifications and to the 10 Series Family Installation Guide.

From release sw 5.1.2, the INTERBUS connecting system may be used. Up to 2048 inputs and 2048 outputs may be managed on this system.

The figure that follows shows an example of how to connect the various 10 Series system modules.

Configuration of the Series 10 System

Operator Panel

The Operator Panel or front panel is the interface between the system and the operator.

There are 3 models of operator panel:

- BLink
- OPLink
- WinLink

In all versions, the Operator Panel comprises a screen with a liquid crystal display, a USA-ASCII standard keyboard separate from the monitor to increase the ergonomics of the system and 7 softkeys. It uses the latest Man-Machine interface techniques in a simple, friendly manner and its main functions are:

- key in operating commands by means of the softkeys;
- enter the start, stop and reset commands;
- enter data, in terms of commands, through the keyboard;
- enter data and software (part programs, software options, etc.), using diskettes;
- display system data and operating conditions;
- control the NC environment or integrated PC environment (on the WinLink model using the yellow key).

- ***BLink Operator Panel***

Is the basic version to be used with an external Operator Console, available with a 10.4” TFT colour screen.

BLink Operator Panel

- ***OPLink Operator Panel***

Is the version with a built-in Operator Console that provides CYCLE START; HOLD and RESET buttons; keys for selecting the AUTO and MANUAL modes, for changing the spindle working and rotating speed and 6 function keys with LED's which may be used by the manufacturer for machine logic customization. It is available with a 10.4” TFT colour screen.

OPLink Operator Panel

- ***WinLink Operator Panel***

The WinLink front panel, shown in the diagram below, consists of a 10.4" o 12.1” TFT monitor, alphanumeric keyboard, built-in mouse and built-in PC board (with its own HDU and 3.5" floppy disk drive).
It has no RESET, START or STOP key.
It has a button ("yellow key") which is used for switching the monitor and keyboard from the NC environment to the PC environment, and vice versa.
This front panel enables the monitor and keyboard to be shared by the NC and PC environments, which remain completely separate.
On the built-in PC, you can use Windows applications that communicate with the NC using the Mini DNC Ethernet option supplied with the dynamic library (DLL) developed for Windows.

WinLink type Operator Panel

WINMEDIA

General

The WinMedia Operator Panel is a front-end module for use by the operators of the CNC to which it is connected via the Ethernet line.
The panel can be supplemented with a keyboard connected directly to it and with an operator console connected to a bus line extraneous to the module (communication bus lines such as FastWire/CanOpen). The figure below shows a front view of the panel with the keyboard and the console.

Main characteristics of the panel

Anchoring compatible with 19” pitch rack

15” TFT display screen

Fdu and Cd-Rom accessed from the front

Power supply: 90~265Vac 50~60Hz (100W max) *
connector conforming to IEC/EN60320-1/C14 (INLET type, i.e., with male contacts)

* Connection to safety GND by means of the power supply cord

Characteristics of Cpu board (ProX-3650)

850 MHz Celeron processor

CRT controller/ 640x480, 800x600, 1024x768 FLAT PANEL

256/512 Mbyte DDR SDRAM

128/256 Kbyte cache

4 Mbyte Flash BIOS (I/O Setup and VGA)

Mini DIN connectors for keyboard and Mouse

IDE (Hdu) controller

Fdu controller

3 RS232 serial ports

1 RS232/422/485 serial port

1 SPP, ECP, EPP parallel port

RJ-45 connector for 10/100 Base-Tx Ethernet

2 USB 2.0 ports on back

1 2.0 USB port on front panel

1 dedicated 2.0 USB port for Touch Screen

1 PISA slot for Riser Card

PC panel with keyboard and console – front view

PC panel - rear view

Panel - Keyboard - Console drilling template

4 M5 threaded pins

ATTENTION: when the FDU and CD-ROM options are present, it is necessary to ensure an obstruction-free area, on the respective side, of 140 mm, to provide sufficient room to introduce the floppy/compact disks (zone marked with *).

On the right hand side, this area must also be left unobstructed if one wants to use the USB port for external units (this port is always present).

Display panel drilling template

Operator console drilling template

Keyboard drilling template

PC Panel – Details of connectors

PC Panel – Position of Riser Card

Mass storage

Mass storage by means of an HDU with a capacity of 20 and more Gbytes, mounted inside the module.

FDU peripheral

The module uses an integrated FDU peripheral, fitted to the left part of the panel (option).

CD-ROM peripheral

The module has an integrated CD-ROM peripheral, fitted to the right part of the panel (option).

Keyboard connection

The keyboard is connected by means of a special Mini Din connector on the back of the module.

Mouse connection

The mouse is connected by means of a special Mini Din connector on the back of the module.

Ethernet connection

The Ethernet connection of the module to an external unit or a network is by means of a special connector on the back. To connect the module to a PC or other similar units you need a crossover cable, to connect it to a HUB you need a straight cable (see the *Connectors/Cables* section).

Riser Card

A Riser Card fitted to the special PISA connector makes available a PCI/ISA expansion connector on the back.

External connections

Back

Com1, **Com3**, **Com4**		RS232
Com2		RS232/RS422/RS485 (configurable)
Parallel port		
LAN	(Ethernet)	
Keyboard	(6-pin Mini DIN)	
Mouse	(6-pin Mini DIN)	
Display screen	(15-pin)	Not used
USB, **USB**	(two ports)	
AC Pwr in		Module power supply

Front

USB	(one port)

Keyboard

The keyboard has a standard Windows PC configuration with 83 short-stroke keys of silicone rubber and a built-in mouse pad with right and left mouse keys. The two (independent) sections are connected to the WinMedia module by means of two cables ending with Mini DIN connectors. PS2 standard communication protocol.

WinMedia keyboard

Operator Console

Specification

The Operator Console is recognised by the NC as an I/O module managed through the FastWire or CanOpen line and it includes:

* Operators (trimmers): SPEED – FEED – JOG
* Selector buttons: CYCLE START – HOLD – RESET
  JOG INC – MANUAL – HOME
  MDI – AUTO – BLK/BLK
* Active functions LEDs
* 12 locations for 16 mm diameter custom operators
* 1 location for a 22 mm diameter custom operator

Front view of console

Console - Rear view of FastWire version

Console - Rear view of CanOpen version

SOFTWARE STRUCTURE

The figure that follows illustrates the software structure. Note that the Event-Driven Real-Time Multi-Task IntePprocess Communications software (IPC) overlaps the basic operating system that manages communications with the operator panel. IPC also supervises and synchronises communications between the five software partitions.

SOFTPIT.DRW

The functions of the five system partitions are as follows:

NC Process: Includes the Part Program interpreter. It interpolates the controlled axes and manages processes in machining centers.

Machine Logic Interface: Executes the OEM-developed interfaces between the NC and the MT.

Utilities: Includes a series of packages aimed at the end user, the MTB or the technical assistance engineer. Some are described later in the manual, others are optional (purchased separately).

Human Interface: The NC-operator interface manages visualisation, program editing procedures, softkey operation, help pages, etc.

Real-time DOS Interface: Is the 10 Series "aperture" to OEM for the building of personalised applications.

These applications can use all the 10 Series resources (axes, logic, human interface, etc.), as well as those of the Operating System (*realtime, multitask*, etc.).

Utilities

A utility is a standard or optional program addressed to various users levels (end user, MTB, technical assistance engineer, etc.). The 10 Series Utilities partition includes:

Table Editor

Permits to key in, display and edit the parameters that define the following features:

- origins
- tools
- tool offsets
- tool tables
- tool magazines
- tool data base

The Table Editor includes the Editor Configurator Table, which permits to personalise the contents and the layout of the tables displayed by the system. For details, refer to Chapters 7 and 8.

PLUS Editor

PLUS (Parallel Logic Universal System) is a complete universal programming language used for writing the machine logic that manages the interface between the 10 Series and the machine tool. PLUS collects information about the current process, the axes functions and the system variables and uses this data to make decisions about the machining process and other actions. For more information see the relative documentation, particularly the 10 Series CNC PLUS Library Manual.

Graphic Editor

This option permits to write programs using graphic menus. The Graphic Editor can be broken down into two utilities:

The **Geometric Editor** generates a profile from the definition of its elementary geometric constituents. The desired profile is obtained by linking these constituents.

The **Cycle Editor** permits to program canned machining or measuring cycles. The programmer is presented with a choice of graphic options and of parameters that define them.

For more information about this option, refer to the 10 Series CNC Graphic Editor Manual.

Digitizing

This option permits to program complex profiles (such as those including spline curves) using points captured with different methods. Digitizing methods include manual moves, probing cycles and keyboard entries. Points can also be uploaded from an external computer.

For more information about this option, refer to the 10 Series CNC DIGICAD User Manual.

AMP

The Adjustable Machine Parameters (AMP) permits the user to tailor the 10 Series control to the system applications.
The following is a list of configurable system parameters:

- number of axes and axes parameters (gain, velocity, electrical and mechanical pitch, etc.)
- operator interface or visualisation technique
- system variables
- hardware system configuration
- number of processes
- user-specific DOS packages
- auxiliary functions
- travel limits
- etc.

For more information about this option, refer to the 10 Series AMP Software Characterization Manual.

DOS Graphic Interface

This option permits the user to customise the machine by writing application programs in C language and using the DOS Graphic Interface libraries. For more information about this option, refer to the 10 Series CNC DOS Graphic Interface Manual.

MINI DNC

This option permits network connection of 10 Series systems and a PC for exchanging files (part programs, etc.) and for communication between processes ("task to task").
There are two types of connections:

- over a RS-232 serial line
- PC connected over an Ethernet link to various systems.

For more information about this option, refer to the MINI DNC Serial/Ethernet Manual.

In addition to the options listed and displayed in the figure, 10 Series systems offer other options now listed below:

Tool Management Manages the flow of tool information between the Tool Magazine and the shop floor, and the operations for moving the tools in the machine.

For more information about this option, refer to the 10 Series CNC Tool Magazine Option.

Electronic Cam This option permits to define a master axis and a slave using the positioning values stored in predifined tables. The master/slave pair sampling tick can be much smaller than that available with ordinary programming and calculation methods.

Oscilloscope This standard feature permits to visualise in a continuous graphic the feedrate or the following error of the axes, and to store the relevant data in a file.

System History This standard feature permits to store in a file all the diagnostic messages sent by the system when erroneous operations are performed or a mulfunctioning occurs. Messages are complete with date and time. This file is of considerable importance for troubleshooting and diagnostic purposes.

END OF CHAPTER

SYSTEM START UP

SYSTEM POWER UP

The system is powered up when power is supplied to the central unit and the front panel. To do this you must typically operate the machine main power switch.

Since most of the system software is stored in the hard disk, at power up the system automatically loads the NC management software in the system memory.

DIAGNOSTIC SCREENS

At power up the system also launches a diagnostics of the system hardware and software modules. The diagnostic sequence is as follows:

- hardware (CPU, ROM, RAM, keyboard and hard disk)
- additional boards and devices
- NC software

To each diagnostic module corresponds a screen similar to the one shown in the figure below.

On the intelligent front panel (in systems equipped with this model of front panel), the messages displayed on the screens may vary.
For a list of all error messages for this type of operator panel, see the relevant section of Appendix A.

A *"LED" Area*

B *Percentage of Loaded Software*

C *Message Area*

The diagnostic screen can be partitioned into three main areas:

A "*LED" Area:* this area displays 5 or 6 boxes, each one of which represents a diagnostic LED. The name of the label indicates the device tested by the diagnostic routine. The boxes are displayed in green when the modules pass the test. They are displayed in red when a failure occurs. In this case, a fixed label indicates a locking error, whereas a blinking label indicates a non-locking error. The error will be further described in the data logger display area.

B *Percentage of Loaded Software:* as the software is loaded in the system memory during initialisation, the color bar displayed in this area shows the loaded percentage.

C *Data Logger Display:* this area displays the test results. The layout of a typical message line is as follows:

The meaning of these columns is as follows:

Tested Device is the acronym of the device, typically the label displayed on the diagnostic "LED" area.

Message Code is the acronym of the diagnostic message. It is made up of two letters and three digits. In the standard front panel this acronym corresponds to the message code listed in Appendix A. Note that this appendix also provides the description of the failure and the remedial action.

Test Description provides a short description of the test (if passed) or an error message when an error has been detected.

Test Results this column may display one of these messages:

PASSED the device has passed the test.

WARNING the device has not passed the test but the error is not a locking one. When this occurs, the LED is displayed in red with a blinking label. The **WARNING** message is written in white letters on a red background.

FAILED the device has not passed the test and the relevant LED is red.

REPORT an optional device (mathematical coprocessor, software package, etc.) is present. This message also appears to indicate that the system is going through initialisation or is configuring the software.

In general only one **FAILED** message is displayed when a locking error occurs. On the contrary, when more than one **WARNING** must be displayed, the following string appears:

Press enter to continue

When this message appears, the test stops. To display all the messages that cannot be accomodated in the message area it is necessary to press the **[Arrow Up/Down]** arrow keys. To resume the test press **[Enter]**.

Modules

The LED labels and the messages displayed in the *Data Logger Display* vary with the type of tested module.

As a result various possible conditions may be displayed sequentially while the relevant module is tested:

	MAIN CPU HARDWARE DIAGNOSTIC				
Basic hardware (system CPU)	**CPU**	**ROM B02.03A**	**RAM**	**KEYB**	**DISK**

	CNC SOFTWARE DIAGNOSTICS				
Additional hardware (any boards installed)		**PLUS**	**AXIS**		

	CNC SOFTWARE DIAGNOSTICS				
NC software	**OS V03.00**	**PLUS V03.00**	**SERVO V03.00**	**PROCESS V03.00**	**HUMAN V03.00**

On the new 10/110 controls of the 10 Series, the initial diagnostic phase in which the basic hardware components are tested is displayed in a different format typical of the diagnostic messages displayed on standard personal computers.

On the other two models, the diagnostic screens are as described.

EMERGENCY SYSTEM SWITCH-ON

It is possible to perform a "minimum" bootstrap on 10 Series systems in which some operations only executed in this environment can be perfromed, while not activating all the system features.

IMPORTANT

The minimum bootstrap is activated by holding down the **[F1]** button when switching the system on.
The system will be brought up with the EMERGENCY DIAGNOSTIC screen:

IMPORTANT

If the front panel is WinLink type, first press the "*yellow button*" when switching on and then, after switching video and keyboard to NC environment, press **[F1]**.
The system will be brought up with the EMERGENCY DIAGNOSTIC screen:

In emergency, the following softkeys are available:

- **AMP**
  This softkey grants access to the AMP utility for system characterisation. The functions of this utility are described in the Characterisation Manual.

- **PLUS**
  Grants access to the PLUS environment for development of the machine logic programs. For further details, read the PLUS manuals.

- **VARIABLE RESET**
  With this utility, the dual port memory areas present in the system can be initialised, saved in a file and restored from a file. This topic is discussed in the Characterisation Manual.

- **SYSTEM HISTORY**
  This softkey grants access to the utility for display/printing/deleting of the history log of errors and system messages displayed during operation of the machine. This file has a fixed size, which means that beyond a certain number of messages, new incoming reports will involve the oldest reports being pushed out.

- **HELP**
  Displays the on-line help window.

- **DOS SHELL**
  This softkey grants access to the DOS SHELL utility.
  This utility permits execution of commands for control of the files in the drives acknowledged by the system (local or remote). Further information on this utility may be found in this manual.

- **SECURITY**
  Manages the system security levels at the different user levels. Further information on this utility may be found in this manual.

- **TABLE EDITOR**
  This utility grants management of the system tables; further information on this utility may be found in the 10 Series CNC user manual.

- **PERIPHERALS**
  Permits selection of the output port for the system printer or for use in the DOS environment.

- **LANGUAGES**
  This is a utility permitting personalization of the national language of the mesages displayed in the HELP, SOFTKEY, etc. Further information on this utility may be found in the Characterisation Manual.

- **SYSTEM SHUT DOWN**
  This is used to perform a bootstrap, as recommended after making changes to the software characterisation or the machine logic, and it is also used to shut down the system.

- **COMPILER**
  Allows compiling and linking of source files written in C or ASSEMBLER.

- **UTILITY**
  Permits execution of what is specified in file E:UTY\AUTOEMG.BAT or in file A:\AUTOEMG.BAT, present on a diskette.
  This is a utility permitting execution of programs for Hard Disk compacting, system scanning in search for "viruses", or other programs selected by the OEM.
  File E:\UTY\AUTOEMG.BAT (or a copy on diskette) must be modified for use of the utilities the OEM wishes to execute, the file itself presents an example of the typical structure to follow.

DIAGNOSTIC MESSAGES

As mentioned previously, the messages on the diagnostic screens vary according to the type of module being tested.

The messages about optional devices display the REPORT label in the result area, whereas actual test results may be indicated by PASSED, FAILED or WARNING..

IMPORTANT

In the WinLink front panel it must be recalled that on bootstrap of the full system (NC and WinLink), the default video is that of the integrated PC and the diagnostic will also be the PC diagnostic; the NC diagnostic will be visible only following a switch of operating environment on pressing the "*yellow button*".

An example of the sequence of PASSED or REPORT messages that may be displayed during the test carried out on the modules of a 10 Series system (except for the 10/110 model).

The BD acronym given in the table does not identify a single message but, rather, a tested group, i.e. CPU, RAM, etc.

PASSED Tests

Device	Acronym	Description	Result
CPU	**BD000**	**Processor** The CPU has passed the autodiagnostics.	**PASSED**
CPU	**BD000**	**Interrupt Controllers** The interrupt controllers have passed the test.	**PASSED**
CPU	**BD000**	**DMA Controllers** The direct memory access controllers have passed the test.	**PASSED**
CPU	**BD000**	**Pointing Device** The mouse hardware has passed the test.	**PASSED**
CPU	**BD000**	**Clock/Calendar** The real time clock registers work properly.	**PASSED**
CPU	**BD000**	**CPU Protected Mode** The protected CPU mode has passed the test.	**PASSED**
ROM	**BD001**	**PRODUCT KEY Checksum** The checksum of the product electronic key is correct.	**PASSED**
RAM	**BD002**	**CMOS Memory Battery Back Up** The back up battery of the CMOS has passed the test.	**PASSED**
RAM	**BD002**	**Parity Circuitry** The circuitry that generates the parity bit for RAM control has passed the test.	**PASSED**
RAM	**BD002**	**CMOS RAM** The CMOS RAM has passed the test.	**PASSED**
KEYBO	**BD003**	**Keyboard** The keyboard circuitry has passed the test.	**PASSED**
DISK	**BD004**	**Fixed Disk: 1 present** The hard disk unit is present and works properly.	**PASSED**
DISK	**BD004**	**Floppy Disk: 1 present** The floppy disk unit is present and works properly.	**PASSED**

REPORTS about devices and actions in progress

Device	Acronym	Description	Result
CPU	**BD000**	**Math Processor** The mathematic coprocessor is present.	**REPORT**
DISK	**BD039**	**Fixed Disk Present** The hard disk is present.	**REPORT**
DOS	**BD102**	**Primary Bootstrap** MS DOS is being initialised.	**REPORT**
DOS	**BD104**	**Remove the diskette and strike any key**	**REPORT**
DOS	**BD106**	**Insert release floppy n. 1 and reboot the control** This message appears while the system is loading the SW release from a diskette.	**REPORT**
OS	**OD001**	**Operating System Loading / Loaded** The NC operating system is being loaded or has been loaded in the system memory.	**REPORT**
OS	**OD002**	**File Server Loading** The NC operating system is being loaded in the system memory.	**REPORT**
OS	**OD003**	**Application Loading** The application program is being loaded in memory.	**REPORT**
OS	**OD100**	**RS232 Software Loading** The serial communications software is being loaded in memory.	**REPORT**
OS	**OD101**	**SK-NETBIOS Software Loading** The protocol of the SK-Netbios sofware for the MINI DNC Ethernet option is being loaded in memory.	**REPORT**
OS	**OD102**	**DEC-NETBIOS Software Loading** The protocol of the DEC-Netbios sofware for the MINI DNC Ethernet option is being loaded in memory.	**REPORT**
PLUS (*)	**HD061**	**Loaded Boot strap Program** The PLUS initialisation program has been loaded in memory.	**REPORT**
PLUS (*)	**HD062**	**Check RAM** The board has passed the RAM test.	**REPORT**

Device	Acronym	Description	Result
PLUS (*)	**HD063**	**Loaded Diagnostics** The board diagnostic program has been loaded in memory.	**REPORT**
PLUS	**SD000**	**PLUS Environment Creation** The creation of the PLUS environment has been completed.	**REPORT**
SERVO	**SD000**	**Servo Environment Creation** The creation of the SERVO environment has been completed.	**REPORT**
PROCESS	**SD000**	**Process 1 Environment Creation** The creation of the PROCESS 1 environment has been completed.	**REPORT**
PROCESS	**SD000**	**Process 2 Environment Creation** The creation of the PROCESS 2 environment has been completed.	**REPORT**
PROCESS	**SD000**	**Process 3 Environment Creation** The creation of the PROCESS 3 environment has been completed.	**REPORT**
PROCESS	**SD000**	**Process 4 Environment Creation** The creation of the PROCESS 4 environment has been completed.	**REPORT**
PLUS	**SD000**	**Creating I/O Environment** The creation of the I/O environment has started.	**REPORT**
HUMAN	**SD000**	**H-I Environment Creation** The creation of the User Interface environment has been completed.	**REPORT**
OEM	**SD000**	**OEM Environment Creation** The creation of the OEM environment has been completed.	**REPORT**

(*) Acronyms HD061-063 refer to a whole series of boards and devices. For example, in HD061 the message may refer to AXIS 1, AXIS 2, etc. rather than to PLUS.

Error messages

Locking (FAILED) and non-locking (WARNING) error messages and remedial actions are described in Appendix A.

END OF CHAPTER

USER INTERFACE

The user interface is the means by which the operator communicates with the system. The screen provides the operator with visual system information, while the operator uses the keyboard and the pushbuttons available on the front panel for entering parameters, instructions and commands.

This chapter describes 10 Series keyboard, video pages and softkeys.

KEYBOARD

The 10 Series keyboard includes:

- 3 function keys, **[F1]** - **[F3]** on the left-hand side
- 2 function keys, **[F11]** and **[F12]**, on the right-hand side
- an alphanumeric section

Function keys

[F1] - [F3] These function keys are not used; they are reserved for use by OEM operators.

[F11] - [F12] These function keys are not used; they are reserved for use by OEM operators (on the WinLink front panel).

Alphanumeric Section

This section of the keyboard consists of 89 keys which may be identified in three areas of the figure: the function keys, the typewriter section and the numeric keypad, which are used for entering data, moving the cursor and scrolling through video screens. The following keys perform a particular function in the alphanumeric section:

The **[Arrow Up/Down] [Arrow Left/Right]** keys move the cursor in the direction indicated by the arrow. They may be used in input boxes, utilities, editing operations and for selecting axes in manual movements.

The **[Return]** key is used for moving the cursor from one field to another in input boxes, tables, etc.

The **[Enter]** key is used for confirming input boxes and, more generally, data entered on the keyboard.

The **[Pg Up]** and **[Pg Dn]** keys are used for scrolling through help pages on the screen, program listings in the line editor and table data in the Table Editor. In general, they are used for scrolling through a list of items when the list is longer than the video screen on which it is displayed.

The **[Home]** key is used for moving the cursor to the start of the programs directory or, when pressed together with **[Ctrl]**, to the start of the program listing.

The **[End]** key is used for moving the cursor to the end of the programs directory or, when pressed together with **[Ctrl]**, to the end of the program listing.

The **[Esc]** key clears the input box without saving any changes made

The **[Ins]** key is used for switching from overwrite to insert typing mode or vice versa

The **[Back Space]** key deletes the first character to the left of the cursor.

The **[Delete]** key deletes the character at the cursor position.

The **[Clear Msg]** key removes the following types of messages from the screen::

- PLUS messages
- process (*) and emergency (*) messages

(*)for these environments, the function of the Clear Msg key may be configured (see the documentation of the "LANGUAGES" package).

The **[Clear Line]** key deletes the contents of the manual data entry line on which the cursor is positioned.

The **[Caps Lock]** key changes the typing mode from capital to small letters and vice versa.

The **[Shift]** key, when pressed together with a key, is used for performing the upper-case functions indicated at the top of the associated key.

The **[Alt], [Ctrl], [Pause] [Break], [Print]** keys are not used; they are reserved for use by OEM operators.

OPERATOR PANEL FUNCTION KEYS

Function keys **[F4]** to **[F10]** situated at the bottom of the operator panel are reserved for softkeys and fixed keys.

The five softkeys enable the user to send commands to the 10 Series CNC system. Their function is determined by the software in each individual case through a series of menus. The menus have a hierarchical structure and are displayed on the bottom three lines of the screen. The active line of a menu establishes the current functions of the softkeys.

A detailed description of the softkeys is provided later in this chapter.

Key **[F5]** SOFTKEY 1

Key **[F6]** SOFTKEY 2

Key **[F7]** SOFTKEY 3

Key **[F8]** SOFTKEY 4

Key **[F9]** SOFTKEY 5

Fixed Keys

The fixed keys are located on either side of the softkeys and have the following functions:

Key **[F4]**
The **left** fixed key always takes you back to the main menu and displays the main screen, thus facilitating access to the machining environment.

Key **[F10]**
The **right** fixed key is used for changing the active line of the softkeys by moving the blue bar vertically through the three lines of softkeys.

Special keys

yellow button This button on the WinLink front panel is used to switch video and keyboard from the integrated PC to the NC and vice versa, without affecting any ongoing operations on the two systems.

keys P1÷P6 These 6 function keys present on the WinLink and OPLink front panel are at the operator's disposal for customising the functions.

IMPORTANT

- On the WinLink front panel, these keys and their LED's may **ONLY** be controlled from the PC side.
- On the OPLink front panel, these keys and their LED's may be controlled by the machine logic (OEM SOFTKEY tasks)

CONTROL BUTTONS

The three control buttons perform the following functions:

The **RESET** button aborts the part program being executed and any active function.

The **CYCLE START** button starts executing the part program and enables the axis homing cycle and the execution of manual movements.

The **CYCLE STOP** button stops various functions depending on whether the system is in **Auto** or **Manual** mode. In particular:

Auto stops executing the part program with controlled deceleration of motion. To exit from the cycle stop condition, press **CYCLE STOP** again; the process enters HOLD RUN status. To resume execution, press **CYCLE START**.

Manual stops the automatic axis homing cycle, incremental JOG motion and automatic return from JOG. To exit from the cycle stop condition, press **CYCLE STOP** again. To resume execution, press **CYCLE START**.

More detailed information on each key will be provided in this manual as it is mentioned in the descriptions of the single operations.

VIDEO

10 Series systems use the video for visualising information about system operation and functions, requesting parameters or instructions, etc.

The video is a 12" color screen with 640 x 480 resolution. It can display up to 25 80-character lines. Screen lines are partitioned as follows:

- the first line always displays system status information
- the last three lines are always occupied by the softkey menu
- the remaning 21 lines are used for information display. The layout may vary enormously according to the context in which visualisation occurs. In some cases one of the lines in this area displays error messages.

Displayed information may occupy the whole data area of the screen or just a quadrant of it. Each quadrant is made up of 10 39-character lines. A quadrant may be enlarged to occupy the whole screen by means of the **ENLARGE** softkey, as shown in the figure below. The enlarging procedure will be described later in this chapter.

The table below lists the video pages managed by the system. The first three and the last six are standard; the others can be configured in the AMP environment from standard quadrants, special screens created by the logic, or customised screens written through the ASSET utility. Supplementary screens 1 through 5 are configurable, whereas Logic quadrants 1 through 4 can be customised and subsequently managed as configurable screens

Video type	Description
①	Main process screen
①	Main logic screen
①	Axes coordinates in large scale
①	Logic screen 1
①	Logic screen 2
①	Logic screen 3
①	Logic screen 4
①	Supplementary screen 1
①	Supplementary screen 2
①	Supplementary screen 3
①	Supplementary screen 4
①	Supplementary screen 5
¼	Logic quadrant 1
¼	Logic quadrant 2
¼	Logic quadrant 3
¼	Logic quadrant 4
¼	System status quadrant
¼	Axes coordinates quadrant
¼	Part program display quadrant
¼	Process status quadrant
¼	Programmed code status quadrant
¼	Axes offset parameters quadrant

① Full screen page

¼ Quadrant page

An example of AMP configurable screen could be a Supplementary screen made up of 4 quadrants: the upper left one displays the system status, the upper right one displays the process status, the lower left one displays the part program and the lower right one the axes coordinates. Further information about supplementary screens will be provided later in this chapter.

The screens displayed by the system may accomodate overlapping windows that provide system messages or additional information on request. These windows will be discussed later in this chapter.

The sections that follow describe the main system screens. Other screens, such as the Editor screens or the various Tables, will be discussed in the chapters dedicated to the relevant environments.

Main screen

This screen is normally displayed during cycle execution. It provides the end user with information about the machining environment. This information is partitioned into five areas, as shown in the figure below:

```
PROCESS CONTROLLED: 1      CAPS ON              SCREEN 1 OF 4      TIME: 9:35:00
PROC: 1 IDLE       AUTO                 OPT STOP                 RETRACE
     AXIS          WORK          [mm]          PROGRAMMED          ORIGIN
L    X             - 149.00000                    -149.23600          1
LM   Y               32.25603                      33.39406          2
     Z                0.00000                       0.00000          TI
     A                0.00000                       0.00000          0
     B                0.00000                       0.00000          0
                                                                     0

 S:          100.000                F:              1000.000
      RAP:          0.0000                                ACT :  Txx.xx
 112.5%       75.000                100.0%  900.000               100.0%  xxxxxx
                    NXT:    Txx.xx
 G:   00     80     99     40     27     90                    71     17     94
      97
      d:     0.000000
 M:   xx     xx     xx     xx
                    JOG:   0.00000                             Z:  0.000000

      PROGRAMS:     MAINPROG SUB1 SUB2
      N26     S2000 F200 T3.03
      N27     (UTO,1,X30,Y22)
      N28     (ROT,20)
      N29     G81 R-90 Z-110 M3
      N30     X25 Y25
      N31     X40 Y10
      N32     X25
POS DISPLAY   NEXT DISPLAY   SELECT PROCESS   ENLARGE   HELP
AUTO          MANUAL         PART PROGRAM     VARIABLES OEM SOFTKEY
MACHINE SETUP TABLES         DIAGNOSTICS      PERIPHERALS UTILITY
```

A *status information area*

B *axes parameters area*

C *general data area*

D *part program area*

E *softkey area*

Status information area

This area is made up the first two screen lines. It provides information about the control status.

PROCESS CONTROLLED This field visualises the number of the currently controlled process.

CAPS ON
caps off This field indicates whether the input characters will be upper cases (CAPS ON) or lower cases (caps off). To toggle between these two options press **[CAPS LOCK].**

SCREEN 1 OF 4 This field indicates the quadrant number in the sequence configured in AMP.

TIME This field specifies the current time in hours, minutes and seconds.

PROC: 1 IDLE This field visualises the status of the currently selected process. The status may vary according to the operation mode (MANUAL or AUTO): some statuses are only allowed in one mode, whereas others are compatible with both modes:

IDLE (auto/manual)
HRUN (auto/manual)
RUNH (auto/manual)
HOLD (manual)
CYCLE STOP (auto)
RUN (manual)
IN CYCLE (auto)
RESET (auto/manual)
WAIT (auto)
INPUT (auto) - with the ASSET option
ERROR (auto/manual)
EMERG (auto/manual)

For further information about this topic, refer to the "System set-up" section later in this chapter.

AUTO This field displays the current operation mode (AUTO or MANUAL). Each mode permits to select a series of sub-modes.

AUTO: BLK/BLK, MDI
MANUAL: HOME, JOG INCR, JOG RETURN, HANDWHEEL

The active options selected in the current mode are displayed in reverse. They may be as follows:

AUTO: OPT STOP, BLOCK DELETE, FEED BYPASS, RETRACE, RCM ON
MANUAL: RETRACE, JOG +, JOG -, RCM ON

The table that follows shows each mode with its relevant process statuses, sub-modes and options.

MODE	STATUS	SUB-MODE	OPTIONS
AUTO	RESET IDLE IN CYCLE CYCLE STOP HRUN RUNH WAIT INPUT (con opzione ASSET) ERROR EMERG	BLK/BLK MDI	OPT STOP BLOCK DELETE FEED BYPASS RETRACE RCM ON
MANUAL	RUN HRUN HOLD IDLE RUNH RESET ERROR EMERG	HOME JOG INCR JOG RETURN HANDWHEEL	RETRACE JOG + JOG - RCM ON

Axes data area

This area, which visualises the axes coordinates, is illustrated in the figure below. The axes in the example are the first two axes from the Main Screen figure on page 3-8. The area can display as many as 9 axes.

WORK This field specifies which axes parameters are displayed in real-time. The field name can be selected by pressing the **POS DISPLAY** softkey. The possible options are:

WORK	displays the calculated axis position with respect to the current origin
MACHINE	displays the distance from the current axis position to the current origin
TO GO	displays the distance to go from the current origin
ERROR	displays the following error, which is the variance between the WORK and MACHINE positions
ABSOLUTE	displays the distance from the current axis position to the absolute zero.

Coordinates Axes coordinates are displayed with as many as 10 digits in the format configured in the AMP (7.3, 6.4 or 5.5). Locked axes are marked by an **L**, whereas the axes that perform mirror machining are marked with an **M**.

[mm] Displays the current unit of measure for the selected process. It may be **mm** or **inches**.

PROGRAMMED Is the last programmed or input axis coordinate. It can have as many as 10 digits with the configured format.

ORIGIN Number and type of origin associated to the axes. This field can display the following words:

T for temporary origins **I** for incremental origins

If neither of these characters is present, the axes are associated to the absolute origin.

General data area

This area supplies general machining information such as spindle speed, feedrate, rapid, active G codes, etc. expressed in the current unit of measure (mm/inches), which is also displayed on the video screen, and are updated at each G70/G71 variation. The meaning of the sample fields illustrated on page 3-8 is as follows:

S:	**100.000**
112.5%	**75.000**

This area displays spindle speed information:

S: spindle data

(100.000) programmed speed or constant cutting rate in 6.3 format.
(112.5%) spindle speed override.
(75.000) current spindle speed (CSS) programmed in 6.3 format in case of open loop.

When the spindle speed override function is active, the field name is displayed in reverse.

F:	**1000.000**
112.5%	**900.000**

This area displays feedrate information:

S: feedrate data

(1000.000) programmed feedrate in 5.5 format.
(100.0%) feedrate override.
(900.000) current feedrate in 5.5 format.

When the feedrate override function is active, the field name is displayed in reverse.

RAP:	**0.00000**
100.0%	**xxxxxx**

Depending on the selected mode, this area displays rapid or manual rate information:

RAP: this label is displayed when the system is in AUTO mode. In manual mode this label will be MAN.
(0.0000) manual (MAN) or rapid (AUTO) rate in 5.5 format.
(100.00%) manual or rapid speed override.
(xxxxxx) current manual or rapid rate in 5.5. format.

When the rapid rate override function is active, the field name is displayed in reverse.

G:	**00 80 99 40 27** **90 71 17** **94 97**

Active G codes: this field displays an active G code for each of the configured modal groups. For further information about this topic refer to the Programming Manual.

M:	**xx xx xx xx**

Active M codes: this field displays an active M code for each of the configured modal groups. For further information about this topic refer to the Programming Manual.

JOG:	**0.00000**	Value of the increment defined for manual incremental moves.
ACT:	**Txx.xx**	Number of the tool mounted on the spindle and of the active offset associated to it. The T word, which identifies the tool, is followed by two numerical codes separated by a point. The code to left of the point is the tool number. The code to right of the point is the tool offset number.
NXT:	**Txx.xx**	Number of the next tool to be mounted on the spindle and of the active offset associated to it. The T word, which identifies the tool, is followed by two numerical codes separated by a point. The code to left of the point is the tool number. The code to right of the point is the tool offset number.
d:	**0.000000**	Tool diameter.
Z:	**0.000000**	This field displays the length offset applied to the active tool axis, which is typically the spindle axis. It is calculated by multiplying the value written in the tool offsets table by the configured direction with the opposite sign.

Part program data area

In this area are visualized the data relative to the running part program. On these rows it is not possible to modify the program.

Each part program block will occupy one or two rows according to its length.The sequence is the following: the part program block that has been run, the running one highlighted in reverse, then the block that hasn't been run yet.

The part program instructions will slide upwards as they will be run.

The following figure shows a possible example of a running part program visualization:

IMPORTANT

If the control manages more than 6 configured axes, 7-n lines will be reserved for displaying program blocks. n is any number of axes greater than 6.
In case of nine configured axes, for the part program run block only the first row is reserved, therefore the longest blocks of 80 characters are truncated.

The meaning of the parameters displayed in the program blocks is explained in the Programming Manual. The contents of the first display line are as follows:

PROGRAMS

The first name on the line is the main program name. Subsequent names are subroutine names. Though a program or subroutine name can be up to 48 characters long, only the first 12 characters will be displayed on this line.

When a subroutine becomes active, its name is displayed after the name of the calling program or subroutine. This means that the first displayed subroutine has the lowest nesting level.

Large font display

This screen displays some axes parameters using a large video font, which permits the operator to read the axes coordinates from a certain distance. It is a full screen that can be invoked by pressing the **NEXT DISPLAY** softkey. The layout of this screen is as follows:

Parameters are displayed in the current unit of measure (mm/inches), which depends on whether G70 or G71 is active.

Large font display

Fields present on left hand side of screen in large font display

The left hand side of the screen is made up of 6 lines, in which the same parameters shown on the first two columns of the axes data area of the main screen are displayed in a large font. The meaning of these parameters is discussed in the "Main Screen" section.

Left hand side of the screen detailed in enlarged view

Fields present on right hand side of screen in large font display

The right hand half of the screen displays the following information:

PROCESS Is the number of the selected process.

S Is the spindle speed or the constant surface speed (CSS) in 6.3 format. CSS is used when an open loop is programmed.

S% Percentage spindle speed override.

F Programmed feedrate in 5.5 format.

F% Percentage feedrate override.

RAP Depending on the selected mode (AUTO or MANUAL) this area displays the following information:

RAP this label indicates that the system is in AUTO mode. The field displays the **rapid** rate.

MAN this label indicates that the system is in MANUAL mode. The field displays the **manual** rate.

RAP % This field is related to the preceding one and its label varies accordingly.

When the system is in AUTO mode (**RAP%**) the field displays the percentage **rapid** override.

When the system is in MANUAL mode (**MAN%**) the field displays the percentage **manual** override.

JOG INC Programmed incremental jog for the selected axis.

NOTE:
S, **F** and **MAN/RAP** are displayed in reverse when the relevant function is active.

Configurable screen pages

10 Series offers the application designer the possibility to personalise screen pages to satisfy specific application needs. These screens can be created during system configuration by arranging the pre-defined system and logic elementary quadrants in non-standard layouts.

Each configurable screen has four 10x39 quadrants. Elementary quadrants may be enlarged to occupy the full screen by pressing the **ENLARGE** softkey (refer to the 'Enlarging an elementary quadrant' section). The figure below shows a typical four quadrant screen.

```
PROCESS CONTROLLED:  1          CAPS ON          SCREEN 1 OF 4      TIME: 9:35:00
PROC    MODE    STATE   EXECUTING                 AXIS    WORK          PROGRAMMED
                                                  X       10.00000      10.00000
  1     AUTO    IDLE    MAIN-PROC                 Y       20.00000      20.00000
                                                  Z        3.00000       3.00000

                                                  S RPM:  10.000        JOG INCR 0.00000

AUTO    :STEP   OPT STOP BLOCK DELETE             PROGRAM: MAIN-PROG          PROCESS: 1
          FEED BYPASS  RETRACE                    ACTIVE SUBPROGRAM: SUB1

FEED MMPM: 0.0000      87.5%   0.00000            N6X10Y20
RAP        : 0.0000    100.0%  4472.1361          N7G1Z-.2F2000
SPEED RPM : 100.00     112.5%  0.000              N8G1X340
                                                  N9GZ2
PROGRAM   :MAINPROG N6   2                        N10X324Y50
SUB-PROG  : SUB1                   N              N11G1Z-.2
                                                  N12X329

| POS DISPLAY | NEXT DISPLAY | SELECT PROCESS | ENLARGE | HELP |
|---|---|---|---|---|
| AUTO | MANUAL | PART PROGRAM | VARIABLES | OEM |
| MACHINE SETUP | TABLES | DIAGNOSTICS | | UTILITY |
```

Example of configurable screen page

NOTE:

Distances are displayed in the unit of measure --mm or inches-- selected for the process associated to the elementary quadrant. The unit of measure will remain visualised and will be updated each time the system shifts between G70 and G71.

The screens that visualise the Realtime data (axes position, feed…) are updated only if the information to visualise concern the selected process.

Example:

If in the small screen the axes of process 2 and the controlled process is number 1, the quote WORK (ABSOLUTE,MACHINE..) will not be updated in Realtime, but only when in process 2 some significative variations take place (es.end block execution…)

Here is a description of the pre-defined elementary quadrants that can be used for customising video screens.

System status (SYS_STA)

This quadrant displays information about the system status and indicates the active process. The meaning of its fields is as follows:

PROC	MODE	STATE	EXECUTING
1	AUTO	IDLE	MAIN-PROG

PROC Number of the selected process.

MODE Current operation mode (AUTO or MANUAL).

STATE Current process state. Some states are allowed only in AUTO mode; others occur in MANUAL mode or are allowed in either mode, as shown in the table below. For more information about them, refer to the description of operations that can be performed in each state later in this chapter.

STATE	AUT	MAN	MEANING
IDLE	●	●	The process is waiting for a command.
IN CYCLE	●		The process is executing a part program or an MDI command.
RUN		●	The process is executing a manual move.
CYCLE STOP	●		The execution of a part program or an MDI command has been interrupted and the process is in stand-by.
HOLD		●	The execution of the manual move has been interrupted with the **CYCLE STOP** pushbutton and the process is in stand-by.

STATE	AUT	MAN	MEANING
RUNH	●	●	The execution of the part program has been interrupted with the **CYCLE STOP** pushbutton. When switched to MANUAL mode the system shifts from CYCLE STOP to HOLD. If one or more axes are jogged, the message turns to RUNH and both the **CYCLE START** and the **CYCLE STOP** pushbuttons are on. RUNH tells the operator that a move is in progress.
HRUN	●	●	If the **CYCLE STOP** pushbutton is pressed with the system in HOLD or CYCLE STOP, the system state shifts to HRUN, which means that the process is ready to resume the interrupted process. The **CYCLE STOP** lamp turns off and the process remains in stand-by until you re-press **CYCLE START**.
RESET	●	●	Indicates that the **RESET** pushbutton has been pressed and that the process is being cleared.
WAIT	●	●	The process is waiting to be synchronised with another process.
INPUT	●	●	Indicates that the process is waiting for a manual data input, i.e. a data entry window is open.
ERROR	●	●	Indicates that the process has detected an error during execution. The error can be handled by the part program or lock the process.
EMERGENCY	●	●	The system has detected an error that requests an intervention of the operator. The error is described by an explanatory (refer to the list of Emergency Messages in Appendix A).

EXECUTING Process in execution. In AUTO mode this field displays the first 12 characters of the name of the part program in execution. In MANUAL mode this field may display the HOME, JOG INCR, JOG RETURN or HANDWHEEL submodes.

Axes position (AXES_POS)

This quadrant provides the current machine coordinates and the programmed coordinates calculated by the control.

```
                                [mm]        PROCESS: 1

      AXIS                WORK          PROGRAMMED

        X                   10.00000          10.00000
        Y                   20.00000          20.00000
        Z                    3.00000           3.00000


 S  RPM:    100.000      JOG INC:   0.00000
```

PROCESS Displays the process number associated to the quadrant.

AXIS Displays up to 6 axes. To make a manual move select the axis with the **[up/down arrow keys]**.Locked axes are marked by the L word, whereas mirroring axes are marked by the M word.

WORK Specifies the type of data displayed for each axis. The labels of this field, which can be selected by pressing the **POS DISPLAY** softkey, are: **WORK**, **MACHINE**, **TO GO**, **ERROR** or **ABSOLUTE**.

Axes coordinates are displayed in the 7.3, 6.4, 5.5 or 5.3 format, according to the AMP configuration. They are expressed in the current measuring unit of the process, which also appears on display.

PROGRAMMED Last programmed or input axis coordinate. It is a 10 digit number in the 7.3, 6.4 or 5.5 format.

S Spindle speed in 6.3 format. The field shows the unit of measure followed by the current spindle speed (or the constant surface speed --CSS-- programmed for an open loop control). The unit of measure may be:

RPM revolutions per minute
MPM meters per minute (for the tool cutter)
FPM feet per minute (for the tool cutter)

JOG INC Actual increment for jog moves expressed in 5.5 format.

Program display (PRG_DISP)

This quadrant displays information about the state of the part program in execution.

```
PROGRAM: MAIN-PROG                    PROCESS:  1

ACTIVE SUBPROGRAM:    SUB1

N6X10Y20
N7G1Z-.2F2000
N8G1X340
N9G72
N10X324Y50
N11G1Z-.2
N12X329
```

PROGRAM This field displays the first 12 characters of the name of the part program in execution (A program name can be up to 48 characters long).

PROCESS Displays the number of the process associated to this screen.

ACTIVE SUBPROGRAM Displays the name of the current subroutine, which is the one having the lowest nesting level. Although a program name can be up to 48 characters long, only the first 12 are displayed.

The displayed part program lines are as follows:

N6X10Y20	*executed block*
N7G1Z-.2F2000	*block in execution*
N8G1X340	*subsequent block*
N9G72	
N10X324Y50	*blocks to go*
N11G1Z-.2	
N12X329	

NOTE:
The part program blocks that are longer than 40 characters are truncated.

Process status (PROC_STA)

This quadrant provides information about the process.

```
AUTO        :   STEP                    [mm]    PROCESS: 1
OPT-STOP  BLOCK-DELETE  FEED-BYPASS   RETRACE

F MMPM      :  0.0000                    87.5  %   0.00000
RAP         :  0.0000                   100.0  %   0.00000
S RPM       :  0.000                    100.0  %   0.000

PROGRAM     :  XXXXXX                      N  :    XXX
RPT         :
SUB-PROG    :  XXXXXX                      N  :    XXX
RPT         :
```

PROCESS Displays the number of the process associated to the quadrant.

AUTO Current mode (AUTO or MANUAL). Displayed data vary with the selected mode:

AUTO BLK/BLK, MDI, OPT STOP, BLOCK DELETE, FEED BY PASS, RETRACE

MANUAL JOG RETURN, JOG INCR, HOME, JOG+, JOG-, RETRACE, HANDWHEEL.

F MMPM Displays the following information about the feedrate:

F	MMPM	:	0.0000	87.5	%	0.00000
	unit of measure		*programmed feedrate in 5.5 format*	*override percentage*		*current feedrate*

The units of measure for the feedrate are:

MMPM millimeters per minute

MMPR millimeters per revolution

IPM inches per minute

IPR inches per revolution

When the feedrate override function is active, the label of this field is displayed in reverse.

RAP

This field displays feedrate information in MANUAL or RAPID mode.

For **manual** moves the displayed data are as follows:

MAN	:	0.0000		100.0	%	0.00000
manual move		*manual rate in 5.5 format*		*override percentage*		*current manual rate*

For **rapid** moves the displayed data are as follows:

RAP	:	0.0000		100.0	%	0.00000
rapid move		*rapid rate in 5.5 format*		*override percentage*		*current rapid rate*

When the manual or rapid rate override function is active, the label of this field is displayed in reverse.

S RPM

This field displays the following spindle speed data:

S	RPM	:	100.000		100.0	%	0.00000
	unit of measure		*programmed spindle speed in 6.3 format*		*override percentage*		*current spindle speed*

The spindle speed units are as follows:

RPM revolutions per minute
MPM meters per minute (for the tool cutter)
FPM feet per minute (for the tool cutter)

When the spindle speed override function is active, the label of this field is displayed in reverse.

PROGRAM

This field displays the first 12 characters of the name of the part program in execution (A program name can be up to 48 characters long).

SUB-PROG

Displays the name of the current subroutine, which is the one having the lowest nesting level. Although a program name can be up to 48 characters long, only the first 12 are displayed.

RPT

Specifies the repeat level for the main program and the subroutines.

N

Current block number.

Programmed code state (CODE_STA)

This quadrant provides information about the T, G and M codes written in the part program.

```
                                                PROCESS: 1

T :     ACTIVE     001              OFFSET:     002
        NEXT       002              OFFSET:     028

G :     00     80     99     40     27     90
               71     17     94     97

M :     M00    M06    M19    M50
```

PROCESS Displays the number of the process associated to this screen.

T Information about T codes is arranged on two lines, one for the current tool and one for the subsequent tool, as shown in the figure below:

G Active G code for each of the 15 modal groups.

M Active M code for each modal group. Modal groups must be defined during process configuration.

Axes offsets (AXIS_OFF)

This quadrant provides information about the axes offsets.

```
                          [mm]     PROCESS: 1

 AXIS            ORIGIN            SCALING %

  X               1  T

  Y               2  I                 60.0

 ROTATION :

 XY-26.000
```

PROCESS Displays the number of the process associated to the quadrant.

AXIS Axis name. This field may display up to 6 axes associated to the process.

ORIGIN Displays the origin types associated to the axes. The words in this field may be:

- **T** for temporary origins
- **I** for incremental origins

SCALING % Percentage reduction of the axis scale (if active).

ROTATION Displays the axes of the interpolation plane and the rotation angle formed by them.

PLUS quadrants (LOGQUAD1,LOGQUAD2, LOGQUAD3, LOGQUAD4)

These elementary quadrants may be defined by the OEM operator and are handled by the machine logic. They can be customised through the PLUS logic or the ASSET language and then incorporated to configurable 4-quadrant screens similar to the ones described in the previous section.

Selecting a screen

When the **NEXT DISPLAY** softkey is pressed, the video screens are displayed in the sequence established by the OEM during system configuration. This sequence can be repeated on a cyclic basis.

Enlarging an elementary quadrant

Each of the quadrants in a configurable screen can be enlarged to full screen size. The procedure is as follows:

1. Select the configurable screen that includes the quadrant by pressing the **NEXT DISPLAY** softkey as many times as necessary.
2. Select the elementary quadrant with the **[PgUp]** or **[PgDn]** keys. The background of a selected quadrant becomes light. **[PgUp]** makes the light background move from quadrant to quadrant counterclockwise, whereas **[PgDn]** makes the light background move from quadrant to quadrant clockwise.
3. Press **ENLARGE**.

If **ENLARGE** is not pressed in a few seconds, the quadrant background turns dark again and the quadrant cannot be enlarged.

NOTES:

- When the quadrant is enlarged using the **ENLARGE** key all the lower cases will be converted to upper cases.
- The **ENLARGE** function is not available for PLUS quadrants.

Directory of Programs screen

By pressing the **PART PROGRAM** softkey it is possible to display the directory of programs, whose layout is as follows:

This screen permits to select a part program that can be used in the following operations:

- writing or editing (by means of the Line Editor or the Graphic Editor)
- program management
- activation/de-activation.

NOTE:
For a complete description, see the chapter 10 "Part Program File Manager".

Additional data windows

In addition to the data provided by the main screen and the configurable pages, the operator can request, display or receive further information on additional windows. These windows may temporarily overlap the current display page or open on a reserved area. In some cases they are used by the system for visualising messages for the operator.

10 Series permits to visualise as many as six additional windows. If opened simulataneously, the system applies a priority rule, i.e. the top priority window will cover all the others. System, part program and PLUS message windows are not assigned a priority level because they are displayed on a reserved area.

The table below lists the additional windows arranged on a priority basis.

1	**Small Help window**	This window may be opened with the **HELP** softkey when a data entry window is active. It provides information about the data entry window.
2	**Data entry window**	The system opens this window for the operator to write, edit or confirm parameters. Data entry windows may have differnt sizes, dending on the number of parameters.
3	**Full Help window**	This 20x40 window can be opened with the **HELP** softkey when no data entry window is active. (Otherwise, only a small Help window can be activated).
4	**Manual Data Input line**	This 128 line is used for writing program blocks on the keyboard.
5	**PLUS message window**	**This window is not subject to the priority rule.** It is a 39 character field used by the system for displaying PLUS messages.
6	**System and part program message window**	**This window is not subject to the priority rule.** It is a 39 character field used by the system for displaying system and part program messages.

Certain PLUS warnings and error messages are displayed on the PLUS window. Other PLUS logic messages appear on the LOGIC MAN screen (if configured).

Process or PLUS error messages may be displayed even when softkeys such as **EDIT** are active.

General rules about data entry windows

Data entry operations must be performed exclusively within a data entry window. These windows, which may be invoked by pressing a softkey, are the only tool the operator can use for entering data in the system.

The figure below illustrates a typical data entry window:

```
PROBE SET-UP  [mm]

APPROACH TOLERANCE  :     5.00000
SAFETY TOLERANCE    :     5.00000
MEASURING FEED      :   100.00001
HORIZONTAL SHIFT    :     0.00000
VERTICAL SHIFT      :     0.00000
```

Each window is made up of fields that contain parameters, file names, device names, etc. When filling in these fields you must follow a series of rules that apply to all data enty windows.

Entries must be written in the legal format.

In long real fields the allowed format is a number having up to 10 digts plus the sign and the decimal point.

This long real format is not the same as the long real format used in program blocks (refer to the Programming Manual). If the long real format is assigned to the data entry window by the program and overflows the window field, an E will be displayed to indicate the error condition.

NOTE:
Numerical parameter values are expressed in the default unit of measure configured for the process in AMP. This unit of measure is also displayed on the window.

When the window appears, the cursor is on the most right character of the field. Input characters are displayed as the cursor moves to the left. To make an entry you must use the following function keys:

KEY	FUNCTION
	Moves the cursor to the right or to the left inside the field.
	Moves the cursor from field to field.
PAGE↑	Displays the previous page of a data entry window having multiple pages.
PAGE↓	Displays the subsequent page of a data entry window having multiple pages.
DEL	Deletes the character on which the cursor is positioned.
	Deletes the character to the left of the cursor.
INS	Toggles between overwrite and insert modes.
	Moves the cursor to the subsequent field.
ESC	Aborts the current data entry window.
	Confirms the current data entry window.

A data entry window can be aborted at any moment with **[Esc]** or **[Enter]**.

TO ABORT A WINDOW . . .	PRESS . . .	DATA WILL BE. . .
without saving the alterations	**[Esc]**	lost
confirming the entries/alterations	**[Enter]**	saved

NOTE:
The contents of the data entry window can also be confirmed by pressing the softkey that invoked the window. Data entry windows having multiple pages may be confirmed by pressing **[PgUp]** or **[PgDn].**

SOFTKEYS

There are 5 function keys ([**F5**] ÷ [**F9**]) located at the bottom of the operator panel which are used for performing functions directly. The functions depend on the menus displayed at the bottom of the screen just above the softkeys (see the figures of the operator panels in the "Operator Panel" section of chapter 1).

Each menu is made up of three softkey lines. The functions associated to the softkeys at any one time are those of the active menu line. This line is highlighted by the cyan bar, which can be shifted cyclically from line to line by pressing the fixed key to the right of the softkeys. When a menu is displayed for the first time the active line is the middle one.

The function of the fixed keys is as follows:

This key permits to return to the previous softkey menu. Press it to fastly return to the machining environment and display the main screen. The current task is temporarily suspended but remains active.

This key permits to move the highlighting bar from one softkey menu line to the next.

The color of the softkey label varies according to the softkey status:

WHITE the softkey is available for activation

YELLOW the softkey has been pressed or is lit to indicate a certain mode.

BLACK the softkey is not available, i.e. cannot be activated.

Softkey-driven menus are organized hierarchically, i.e. a softkey may invoke a sub-menu, some of whose softkeys may invoke other sub-menus, and so on. The complete softkey tree is shown in the figure that follows:

At power up the system visualises the main softkey menu, which allows access to all of the system resources.

With the exception of the Diagnostics, Part Program and Utility menus discussed later in this chapter, the first softkey line in the machining environment remains unchanged even when other submenus are invoked. The first softkey line permits to select general visualisation features, as shown in the figure below:

POS DISPLAY	NEXT DISPLAY	SELECT PROCESS	ENLARGE	HELP
XXX	XXX	XXX	XXX	XXX
XXX	XXX	XXX	XXX	XXX

The sections that follow provide a short description of the functions available with the first two submenu levels. The third level is analysed in the context of the relevant operating procedures.

Main menu

POS DISPLAY	NEXT DISPLAY	SELECT PROCESS	ENLARGE	HELP
AUTO	MANUAL	PART PROGRAM	VARIABLES	OEM
MACHINE SET-UP	TABLES	DIAGNOSTICS		UTILITY

POS DISPLAY This softkey permits to select the type of visualised data for the axes configured in the process. Allowed options are: ABSOLUTE, MACHINE, WORK, TO GO, ERROR.

NEXT DISPLAY This softkey displays the next video page in the current sequence.

SELECT PROCESS Through an input window, this softkey allows to select one of the processes configured in AMP.

```
SELECT    PROCESS

Process number:
```

Press **[Ctrl] [P]** to select configured processes sequentially.

ENLARGE This softkey enlarges the quadrant to full screen size (21 lines x 80 columns). For more information about the procedure, please refer to the "Enlarging an elementary quadrant" section in this chapter.

HELP This softkey opens a context-sensitive help window. The visualised information varies with the active softkey and/or the open data entry window.

Auto

This menu, which can be invoked with the **AUTO** softkey, offers the following commands:

POS DISPLAY	NEXT DISPLAY	SELECT PROCESS	SEARCH MEMORY	HELP
MDI	BLK/BLK	BLOCK MODIFY	MACHINE PLOT	DRY RUN
XXXXXXXX+	XXXXXXXX-	OVERRIDE SEL	STRING SEARCH	EXEC FROM:TO

MDI Opens a data entry window for an MDI (Manual Data Input) operation such as writing a program block from the keyboard. Press **[Enter]** to confirm the input block. Press **CYCLE START** to execute the block if the process is in IDLE state.

BLK/BLK Selects block by block execution mode. To execute the selected block press **CYCLE START.**

BLOCK MODIFY Permits to edit the current program block. This softkey is enabled only after a part program has been selected and the MDI mode is active.

MACHINE PLOT This softkey enables an application that graphically visualises the tool path during the machining cycle. For more information about this application refer to chapter 12 “Machine Plot”.

SEARCH MEMORY To enable/disable memory searching function.

DRY RUN Enables/disable the **DRY RUN** feature on the active program.

XXXXXXXX + Applies the feedrate, spindle speed or rapid override percentage selected with the **OVERRIDE SEL.** softkey.

Each time you press this softkey the programmed **increment** is applied to the feedrate, spindle speed or rapid.

XXXXXXXX - Applies the feedrate, spindle speed or rapid override percentage selected with the **OVERRIDE SEL.** softkey.

Each time you press this softkey the programmed **decrement** is applied to the feedrate, spindle speed or rapid.

NOTE:
XXXXXXXX + and XXXXXXXX - change according to the name shown in the table below depending on the type of override desired. Changeover from one quantity to the next is performed cyclically each time the **OVERRIDE SEL** softkey is pressed:

XXXXXXXX +	XXXXXXXX -
FEEDRATE +	FEEDRATE -
SPND OVER +	SPND OVER -
RAPID OVER +	RAPID OVER -

OVERRIDE SEL. Selects the parameter to be overridden (**FEEDRATE, SPND OVER** or **RAPID OVER**) and alters the softkey label accordingly.

STRING SEARCH Searches for a string of characters in the current part program. When this softkey is pressed the system opens a data entry window for the operator to key in the string. This softkey is enabled only after a part program has been selected.

EXEC FROM:TO This softkey selects the start and end blocks for partial exceution of the program selected.

This softkey is enabled only after a part program has been selected.

Manual

This menu, which can be invoked with the **MANUAL** softkey, offers the following options:

POS DISPLAY	NEXT DISPLAY	SELECT PROCESS		HELP
HOME	JOG INCR	JOG RETURN	HPG	MANUAL SETUP
XXXXXXXX+	XXXXXXXX-	OVERRIDE SEL	JOG DIR	SET ORIG./TOOL

HOME Enables axes homing. To start homing the selected axis it is necessary to press the **CYCLE START** pushbutton.

JOG INCR Enables incremental manual moves. To move the selected axis by an increment press the **CYCLE START** pushbutton. See chapter 6 “Axes Jogging and Stop Functions” for more details.

JOG RETURN Enables return to the profile after a manual move. See chapter 6 “Axes Jogging and Stop Functions” for more details.

HPG Enables handwheel management by the logic.

MANUAL SETUP Opens a data entry window for defining the jog increment and the jog return mode. On special occasions it also permits to remove the tool from the part (see chapter 6 “Axes Jogging and Stop Functions”).

XXXXXXXX + Applies the feedrate, spindle speed or rapid override percentage selected with the **OVERRIDE SEL.** softkey.

Each time you press this softkey the programmed **increment** is applied to the feedrate, spindle speed or rapid.

XXXXXXXX - Applies the feedrate, spindle speed or rapid override percentage selected with the **OVERRIDE SEL.** softkey.

Each time you press this softkey the programmed **decrement** is applied to the feedrate, spindle speed or rapid.

NOTE:
XXXXXXXX + and XXXXXXXX - change according to the name shown in the table below depending on the type of override desired. Changeover from one quantity to the next is performed cyclically each time the **OVERRIDE SEL** softkey is pressed:

XXXXXXXX +	XXXXXXXX -
FEEDRATE +	FEEDRATE -
SPND OVER +	SPND OVER -
RAPID OVER +	RAPID OVER -

OVERRIDE SEL. Selects the parameter to be overridden (**FEEDRATE, SPND OVER** or **RAPID OVER**) and alters the softkey label accordingly.

JOG DIR Selects the jogging direction. When this softkey is pressed the + or - sign appear on the label and the axis jogging direction changes accordingly.

SET ORIG./TOOL This softkey has two functions: it sets the origins and presets the tool.

SET ORIG. function:

Opens a data entry window for defining the origins and setting the position of the specified axes to zero.

TOOL function:

Opens a data entry window for defining the axis name and entering the tool tip coordinate. This function applies a new tool length to the tool length offset of the current tool.

Part Program

This menu, which is enabled by the **PART PROGRAM** softkey, permits to access the part program environment and enables the relevant softkeys.

COPY	DELETE	RENAME		HELP
EDIT	SELECT DIR	DIRECTORY	DEACTIVATE	ACTIVATE

For further information, see Chapter 10 "Part Program File Manager".

Variables

This menu is enabled by the **VARIABLES** softkey. It offers softkeys dedicated to visualisation and editing of several classes of variables. For more information about system parameters and variables refer to the "Programming Manual".

POS DISPLAY	NEXT DISPLAY	SELECT PROCESS	ENLARGE	HELP
E VARIABLES	SN VARIABLES	USER VARIABLES	H VARIABLES	PLUS VARIABLES
FIND PAGE/IND		SUBSTITUTE	EVALUATE	SC VARIABLES

E VARIABLES Opens a data entry window for defining E variables.

SN VARIABLES Opens a data entry window for defining Long Real (SN) system variables.

USER VARIABLES Opens a data entry window for defining the user variables defined during the characterisation. These variables are listed in a single group even when different variable names have been defined. At the end of the list the system requests the name of the variable to be visualised.

H VARIABLES Opens the window of H variables for read only operations.

PLUS VARIABLES Opens the window for logic global variables.

FIND PAGE/IND Opens a data entry window for the page number and the index of the variable on which the cursor must be positioned.

SUBSTITUTE Writes in the selected field of the data entry window the value of the last expression calculated with **EVALUATE**.

EVALUATE Opens a data entry window to calculate an expression. In this calculation it is possible to use variables. By pressing the **SUBSTITUTE** softkey the result of the calculation can be written to the selected variable.

SC VARIABLES Opens a data entry window for Character (SC) system variables.

OEM

The menu activated by the **OEM** softkey permits to access and enable the softkeys defined by the application designer. The names of these softkeys may be configured in the AMP and therefore is application-specific.

POS DISPLAY	NEXT DISPLAY	SELECT PROCESS	ENLARGE	HELP
OEM 1-1	OEM 1-2	OEM 1-3	OEM 1-4	OEM 1-5
OEM 2-1	OEM 2-2	OEM 2-3	OEM 2-4	OEM 2-5

Machine Set-Up

The menu activted by the **MACHINE SET-UP** softkey permits to access a submenu of parameters and softkeys that define the machine functions.

POS DISPLAY	NEXT DISPLAY	SELECT PROCESS	ENLARGE	HELP
DYNAMIC PARAM	DYNAMIC LIMITS	PROGRAM SET-UP	BLOCK RETRACE	PROBE SET-UP
REF	SET ACCURACY			TIME/DATE SET

The **DYNAMIC PARAMS** softkey opens a data entry window similar to the one below:

```
DYNAMIC PARAMETERS

ENABLE VFF (Y/N)          :
G04 DWELL TIME            :
SPINDLE LIMIT (RMP)       :
% TAP RETRACT FEED (G84) :
```

Significance of the fields is as follows:

ENABLE VFF(Y/N) Enables/disables the servo algorithm with *velocity feed forward.*

When the algorithm is enabled, the tracking error will be *closed on zero* (consistent with the gain of the servo loop configured in AMP) to reduce the profile shaping error to the minimum.

When the algorithm is disabled, the axes are controlled with *tracking error* only.

G04 DWELL TIME Defines the end-of-block dwell time (G04).

SPINDLE LIMIT (RPM) Defines maximum spindle speed.

Used in CSS - Constant Surface Speed - mode when spindle speed depends on position of the *master* axis.

% TAP RETRACT FEED (G84)
Specifies the percentage of the feedrate applied to the retract phase of the tapping cycle.

DYNAMIC LIMITS Opens a data entry window for defining the dynamic limits of the machine in a contouring operation.

PROGRAM SET-UP

Opens a data entry window permitting configuration of some parameters conditioning the part program execution mode:

- Block delete
- Optional stop
- Feedrate bypass
- Rapid rate control
- Rotation
- Stock allowance

And for each axis configured:

- Stand still
- Mirror modes
- Scale
- Scale factor

BLOCK RETRACE

Permits definition, if configured in AMP, of the machine path to be performed backwards.

The softkey is enabled only if a program is selected and if the **CYCLE STOP** pushbutton is pressed during execution.

The sequence of operations is as follows:

System in CYCLE STOP

- BLOCK RETRACE enabled
- Press CYCLE START; the system moves the axes backwards towards the last point of the buffer.

  Movements will be executed in consistency with the mode selected (AUTO, BLK/BLK).
- To return to normal conditions, disable **BLOCK RETRACE**, exit from CYCLE STOP mode and press CYCLE START.

PROBE SET-UP

Opens a data entry window for setting all the probing parameters.

REF

Opens a data entry window to establish axes references.

SET ACCURACY

Defines accuracy parameters for circular interpolation.

TIME/DATE SET

Opens a data entry window for setting the time with hh/mm/ss format and the date with yy/mm/dd format.

Tables

The menu opened by the **TABLES** softkey permits to access the Table Editor and enables its softkeys. See chapter 7 “Using Tables” for more information about this option.

BACKUP MEMORY	RESTORE MEMORY	MAGAZINE	SETUP	HELP
TOOL	OFFSET	USER	ORIGIN	TOOL DATA BASE
SELECT PROCESS	SELECT MAGAZINE	SET INCR	USER TABLE	EXIT

BACKUP MEMORY Makes a backup copy on the hard disk of the tables resident in the dual port memory.

RESTORE MEMORY Restores the backup copy of the tables to the dual port memory.

MAGAZINE (Option) Permits to manage the Tool Magazine option. For more information about the Tool Magazine please refer to the specific documentation.

SETUP Allows configuration of the tables used by the *table editor*.

HELP Displays the Help page associated to the current data entry window or softkey menu.

TOOL Permits to access and edit the Tools Table.

OFFSET Permits to access and edit the Tool Offset Table.

USER Permits to access and edit the User Table.

ORIGIN Permits to access and treat the contents of the Origins Table.

TOOL DATA BASE Permits to access and edit the Tool Data Base Table, which can be enabled only with the Tool Magazine option.

SELECT PROCESS Permits to select the configured processes.

SELECT MAGAZINE Permits to select the Tool Magazine. For more information about this option refer to its documentation.

SET INCR Permits to set an increment value.

USER TABLE Permits to configure and treat the contents of the User Table. See chapter 8 “Table Editor Configurator” for details.

EXIT Exits from the Table Editor and returns to the main softkey menu.

Diagnostics

The menu enabled by the **DIAGNOSTICS** softkey permits access to the diagnostic functions.

				HELP
SECURITY	SYSTEM HISTORY	DSI SRV CHAN	SERVO MONITOR	
SYSTEM SHUT DOWN				

HELP Opens a Help page associated to the current data entry window or softkey menu.

SECURITY Permits to create a hierarchy of system users having different levels of access to the system resources. To each level corresponds a password that allows to perform specific operations. See chapter 13 “DOS Shell” for details.

SYSTEM HISTORY Enables the video screen the list of the last 200 system messages (errors and service messages) displayed by the system in the chronological sequence in which they occurred.

DSI SRV CHAN Enables the utility for the D.S.I axes parameter configuration.

SERVO MONITOR Permits to access the system Servo Monitor, a utility that can be broken down into two main functions:

Parameter change: permits to edit certain axes parameters during execution without using AMP or re-initialising the system.

Oscilloscope: monitors and graphically displays the axis path with the relevant following error.

SYSTEM SHUT DOWN This function is used to restart or shut down the system in a controlled manner, closing currently open files and applications.

Utility

This menu is enabled by the **UTILITY** softkey. It permits to access the OEM environment and enables the relevant softkeys.

AMP	PLUS	DOS SHELL	PERIPHERAL	HELP

AMP Permits to access the AMP environment, which is used for system configuration.

PLUS Permits to access the PLUS environment, which is used for developing the machine logic.

DOS SHELL Permits the OEM to handle files in the DOS environment.

PERIPHERAL Permits execution of the following operations:

- Configuration of the *output* d*evice* for the printer.
- Remote bootstrap configuration.
- Interlink configuration and activation to connect with a remote unit.
- If the MINI DNC option is present, activation of the network configurator.
- Immediate connection of all environments configured as remote.
- Immediate connection of all services configured as remote.

HELP Opens a Help page associated to the current data entry window or softkey menu.

OPERATOR CONSOLE

The Operator Console provides the operator with three control buttons for executing the CYCLE START, CYCLE STOP and RESET commands, and a group of buttons and/or selectors for changing the speeds of the axes and selecting the operating modes of the system.

If a system with a WinLink and/or BLink operator panel is used, an external Operator Console or **Pilot Panel** is essential while the OPLink operator panel has a built-in Operator Console.

If WinMedia is used as Operator panel, it is indispensable to use an external Operator Console or adopt the “WinMedia Operator Console” described in the following pages.

Nearly all the controls on the operator console duplicate functions performed by following specific operating procedures in which the softkeys are used.
If an external operator console is installed, it must be defined by means of the machine logic whether the commands executed on the pilot panel are to replace the equivalent system functions completely (most common case) or whether they may be carried out in parallel. In the latter case, measures must be taken to ensure that conflicting commands are not executed at the same time.
If the console built into the OPLink operator panel is used, the system automatically controls the commands in parallel.

Operator Console Keys and Functions

As illustrated in the figure, on the right-hand side of the OPLink operator panel there are operator console keys and the START, HOLD and RESET buttons, that is, the control buttons described in the "Control Buttons" section at the start of this chapter.
A description of the keys and functions of the operator console built into the OPLink operator panel is provided below.

Key for selecting the MANUAL MOVEMENT (JOG).
When this key or the **MANUAL** softkey on the main menu is pressed, the system automatically activates **continuous manual movement**. A description of the keys that enable the operator to execute the manual movement commands and the other related functions is provided below. For further details on manual movements, see chapter 6 "Axes Jogging and Stopping Functions" of this manual.

Jog increment

Enables incremental manual movement (**JOG INCR** softkey). When the **CYCLE START** button is pressed as defined in the MANUAL SET-UP input box, the incremental jog is performed on the selected axis.

Home

Enables the axis homing cycle (**HOME** softkey). When the **CYCLE START** button is pressed, the homing procedure is executed on the selected axis (see chapter 5 "Establishing Origins and Homing the Axes").

Speed change

Percentage change in the manual feed rate defined in the configuration (**FEEDMAN +** and **FEEDMAN -** softkeys). When the **[+]** and **[-]** keys are pressed alternately, the percentage change is **increased** or **decreased** by 12.5%.

Increment change

Change in the increment for incremental manual movement (**JOG INCR** + and **JOG INCR** - softkeys). When the **[+]** and **[-]** keys are pressed alternately, the value of the increment is **increased** or **decreased**.

Jog direction change

Selection of the direction of manual movement (**JOG DIR +/-** softkeys). When the **[+]** and **[-]** keys are pressed alternately, the jog direction changes from positive to negative or vice versa.

Key for selecting **AUTOMATIC** operating mode (**AUTO** softkey on the main menu).
When this key is pressed, the system automatically activates **automatic execution** of the selected part program: all the blocks in the part program are executed without interruption. A description of the keys for selecting the other execution modes and other functions of this environment is provided below. For further details on this environment, see chapter 11 "Part Program Execution" of this manual.

BLK/BLK

Block by block execution of the part program (**BLK/BLK** softkey).
When the **CYCLE START** button is pressed, the single selected block is executed.

MDI

Opens the MDI input box for insertion of a program block from the keyboard (**MDI** softkey). Execution of the block inserted starts when the **CYCLE START** button is pressed, having confirmed the block by pressing the Enter key.

Rapid traverse variation

Percentage change in traverse speed defined in the configuration (**RAPID OVER +** and **RAPID OVER -** softkeys). When the **[+]** and **[-]** keys are pressed alternately, the percentage change is **increased** or **decreased** by 12.5%.

Feed rate change

Percentage change in the feed rate set in the part program with the **[F]** operator (**FEEDRATE +** and **FEEDRATE -** softkeys). When the **[+]** and **[-]** keys are pressed alternately, the programmed feed rate value is **increased** or **decreased** by 12.5%.

Spindle speed change

Percentage change in the spindle speed set in the part program with the **[S]** operator (**SPND OVER +** and **SPND OVER -** softkeys). When the **[+]** and **[-]** keys are pressed alternately, the programmed spindle speed value is **increased** or **decreased** by 5%.

Pilot Panel Selectors and Functions

The figure below shows the external operator console or Pilot Panel.

The following buttons are present on the right-hand side of the pilot panel

CYCLE STOP button

CYCLE START button

RESET button

The functions performed by these three buttons are described in the "Control Buttons" section at the start of this chapter.

The commands associated with the pilot panel selectors are described below.

Speed

Controls the spindle speed override percentage and is equivalent to the **OVERRIDE SEL** and **SPND OVER+** / **SPND OVER-** sequence. Spindle speed override percentages range from 75% to 125% of the programmed value.

Feed

This selector controls the feedrate override percentage and is equivalent to the **OVERRIDE SEL** and **FEEDRATE+** / **FEEDRATE-** sequence. The variations posible range from 0 to 125% of the programmed feed rate.

Jog Increment

Selects an increment for each manual incremental move.

Jog

This selector determines speed and direction of the selected axis manual movements for the MANUAL JOG and JOG INCR functions.
If **RAPID OVERRIDE CONTROL** is enabled (in AMP environment or in PROGRAM SET-UP), the selector can be turned in the positive direction to limit rapid override percentage in steps of 100, 50, 20, 5, 1 and 0 percent.

Mode

This selector permits setting of certain operating modes which, when there is no operator console, are activated by the softkeys of the same name. These modes are:

- **MDI**
  Opens a data entry window for manual data inputs. This selection can be enabled in AUTO mode..
- **AUTO**
  Selects **Auto** mode.
- **BLK/BLK**
  Selects block by block program execution.
- **MANUAL JOG MODE**
  Enables axis jogging in the direction established with the Jog Direction selector. When you press **CYCLE START** the axis starts moving in the selected direction. The move lasts as long as **CYCLE START** is pressed.
- **JOG INCREMENT MODE**
  Enables incremental axis jogging in the direction established with the Jog Direction selector. If **[A]** and the increment value have been specified in the SELECT field of the MANUAL SETUP window, when you press **CYCLE START** the axis moves in the selected direction by the specified increment. If **[M]** has been specified in the MANUAL SETUP window it is necessary to press **CYCLE START** until the increment has been completed. (The axis will stop as soon as **CYCLE START** is released).
- **JOG RETURN MODE**
  Returns the axis to the profile after a manual move. For more details about this procedure refer to Chapter 6.
- **HOME**
  Enables the homing procedure. Press **CYCLE START** to start the homing sequence for the selected axis.
- **HPG**
  Enables the logic to handle the handwheel.

Controls and functions of the WinMedia operator console

The figure below illustrates WinMedia's external operator console:

On the right hand side of the console we find the following buttons:

HOLD and **CYCLE STOP** button

CYCLE START button

RESET button

The functions of these three buttons have been described in paragraph "Control buttons" at the beginning of this chapter.

To the left of the reset button we find six more controls for the selection of execution modes.

This is used to set the system on **AUTOMATIC** mode (**AUTO** softkey in the main menu).

When this key is pressed, the system activates automatically the **automatic execution** mode of the part program selected: all part program blocks are performed without breaks. Keys (BLK/BLK and MDI), for the selection of the other execution modes, and the other functions available in this environment are described below. For more detailed information on this environment, see chapter 11 of this Manual, “Part Program Execution”.

Block-by-block part program execution (**BLK/BLK** softkey).

Press the [**CYCLE START**] key to run the single block selected.

Opens the MDI input windows to enable the user to enter a program block from the keyboard (**MDI** softkey). The execution of the chosen block is started by pressing the [**CYCLE START**] key after confirming the command by pressing **Enter**

Key for system **MANUAL JOGGING** modality selection.

As in the case of the **MANUAL** softkey in the main menu, when this key is pressed the system activates the **continuous jogging** function automatically. Keys (JOG INCR and HOME) which are used to impart jogging commands and control other functions associated with jogging are described below. For more detailed information on jogging, see chapter 6 of this Manual, “Axes Jogging and Stop Functions”.

Enables incremental jogging (**INCR JOG** softkey). When the [**CYCLE START**] key is pressed according to the modalities described in the MANUAL CONF. input window, incremental jogging is performed for the axis selected.

Enables the axis homing cycle (**AXIS HOMING** softkey). Pressing the [**CYCLE START**] key starts the homing procedure for the axis selected (see Chapter 5 “Establishing Origins and Homing the Axes”).

We describe below the controls associated with the selectors equipping the WinMedia Operator Console.

Speed

This selector increases/decreases the % variation in spindle speed. It performs the same functions as the **SEL.OVERRIDE** and % **ROT MANDR+**/% **ROT MANDR-** softkey sequence. The admissible variation range is from 75% to 125% of the programmed value.

Feed

This selector increases/decreases the % variation in axis feed rate. It performs the same functions as the **SEL.OVERRIDE** and **FEEDRATE+**/**FEEDRATE-** softkey sequence.
The admissible variation range is from 0% to 125% of the feed rate.

Jog

This selector determines the jogging speed and direction of the selected axis for the JOG and JOG INCR **MANUAL** functions.
If the **RAPID OVERRIDE CONTROL** function is enabled (in either the AMP or the PROGRAM SET-UP environment) it also makes it possible to increase/decrease the RAPID rate in percentage terms.

END OF CHAPTER

SETTING SYSTEM VARIABLES AND PARAMETERS

The procedure for setting system variables and parameters may be broken down into two stages:

- **General configuration:** In this stage the system characteristics, i.e. number and type of axes, functions, programming codes, etc., are configured. These parameters must be set by the application designer in the AMP environment and in general cannot be altered from other environments. For a thorough description of the configuration procedure please refer to the Characterization Manual.
- **Configuration of specific variables and parameters:** In this stage the contents of the variables and the values of the parameters used in a given part program must be defined. The default values or the ranges of some variables, such as user and PLUS, may be set in the general configuration. The contents of any variable may be altered from keyboard or from part program, as illustrated later in this chapter.

LOCAL AND SYSTEM VARIABLES

A part program may include two types of variables:

Local variables are stored in a local memory area of the system and may be "seen" only by the part program in execution. These variables are of two types: **E variables** and **USER variables**

System variables these variables are stored in the system dual port memory and may be "seen" by the part program in execution and by any other active process. This class includes three types of variables: **System Numbers (SN), System Characters (SC)** and **PLUS variables.**

For more information on local and system variables and how to program them refer to the "Programming Manual".

To access the environment in which variables may be set, visualised or edited, press the **VARIABLES** softkey in the main menu.

E variables

E variables (# parameters) are local variables that may have predefined values when used in a part program. Default values for some E parameters are defined during system configuration and become effective at power up.

The **E VARIABLES** softkey permits to visualise and modify the values of an E variable to suit specific programming needs. Modified values are used by the system instead of default settings. When you press the **E VARIABLES** softkey, the following data entry window is displayed:

```
PROCESS CONTROLLED: 1        CAPS ON              SCREEN 1 OF 4      TIME: 9:35:00
 PROC: 1 IDLE          AUTO                                OPT STOP
       AXIS            WORK                [mm]            E VARIABLES    Page 1 of 7
 L     X                   - 149.00000
 LM    Y                     32.25603                      E 0000 = 0.00000
       Z                      0.00000                      E 0001 = 0.00000
       A                      0.00000                      E 0002 = 0.00000
       B                      0.00000                      E 0003 = 0.00000
                                                           E 0004 = 0.00000
  S:       100.000          F:      1000.000               E 0005 = 0.00000
  112.5%   75.000           100.0%  900.000                E 0006 = 0.00000
  G:   00 80 99 40 27 90          71 17 94 97              E 0007 = 0.00000
  M:   xx xx xx xx                                         E 0008 = 0.00000
                                                           E 0009 = 0.00000
   PROGRAMS: MAINPROG SUB1 SUB2                            E 0010 = 0.00000
   N26    S2000 F200 T3.03                                 E 0011 = 0.00000
   N27    (UTO,1,X30,Y22)                                  E 0012 = 0.00000
   N28    (ROT,20)                                         E 0013 = 0.00000
   N2/9   G81 R-90 Z-110 M3                                E 0014 = 0.00000
   N30    X25 Y25                                          E 0015 = 0.00000
   N31    X40 Y10
   N32    X25
POS DISPLAY   | NEXT DISPLAY   | SELECT PROCESS | ENLARGE        | HELP
E VARIABLES   | SN VARIABLES   | USER VARIABLES | H VARIABLES    | PLUS VARIABLES
FIND PAGE/IND |                | SUBSTITUTE     | EVALUATE       | SC VARIABLES
```

To modify an E variable declared in the configuration, move the cursor to the variable value, delete it with **[Backspace]** and key in the new value. After making all the necessary modifications, press **[Enter]** or **E VARIABLES** to confirm them.

IMPORTANT

If power is removed from the system the values assigned to E parameters in this window will be lost.

User variables

User variables may be defined by the user in the AMP environment. They are of two types:

- Long Reals (LR)
- Characters (CH)

If no user variable has been configured in AMP, the **USER VARIABLES** softkey in the VARIABLES menu will not be active. When you press the **USER VARIABLES** softkey the system displays the following data entry window listing the configured user variables with their types.

```
PROCESS CONTROLLED:  1        CAPS ON              SCREEN 1 OF 4      TIME: 9:35:00
 PROC: 1 IDLE          AUTO                                  OPT STOP
       AXIS            WORK                  [mm]
 L     X                    - 149.00000                 USER VARIABLES LIST
 LM    Y                      32.25603
       Z                       0.00000                 !VARIABLE1.LR
       A                       0.00000                 !VARIABLE2.CH
       B                       0.00000                 !VARIABLE3.LR

  S:       100.000            F:      1000.000
  112.5%   75.000             100.0%  900.000
  G:   00  80  99  40  27  90          71  17  94  97
  M:   xx  xx  xx  xx

   PROGRAMS: MAINPROG SUB1 SUB2
   N26    S2000 F200 T3.03
   N27    (UTO,1,X30,Y22)
   N28    (ROT,20)
   N29    G81 R-90 Z-110 M3
   N30    X25 Y25
   N31    X40 Y10                                      INSERT VARIABLE NAME:
   N32    X25
POS DISPLAY    NEXT DISPLAY    SELECT PROCESS    ENLARGE    HELP
E VARIABLES    SN VARIABLES    USER VARIABLES    H VARIABLES    PLUS VARIABLES
FIND PAGE/IND                  SUBSTITUTE        EVALUATE    SC VARIABLES
```

To modify the contents of a user variable use this procedure:

1. Key in the variable name and press **[Enter].**
2. The system visualises the contents of the selected variable.
3. Modify the contents of the variable and press **[Enter]** or the **USER VARIABLES** softkey to confirm them. To visualise the contents without modifying the variables, press **[Esc]**.

Long Real (LR) variables are managed with the same procedure as E parameters.

Character (CH) variables are managed with the same procedure as System Character (SC) variables. More information about SC variables is provided later in this chapter.

H variables

H variables are read-only local variables used by the system paramacros and cannot be modified manually. To visualise the contents of the H variables in a data entry window press **H VARIABLES**. To browse through the data entry window, press **[PgUp]** or **[PgDn]**. If no paramacro has been configured in AMP the **H VARIABLES** softkey will be disabled.

```
PROCESS CONTROLLED:  1        CAPS ON                SCREEN 1 OF 4      TIME: 9:35:00
 PROC: 1 IDLE          AUTO                          OPT STOP
       AXIS            WORK                [mm]        H VARIABLES    Page 1 of 7
 L     X                   - 149.00000
 LM    Y                     32.25603                 H 00 = 0.00000
       Z                      0.00000                 H 01 = 0.00000
       A                      0.00000                 H 02 = 0.00000
       B                      0.00000                 H 03 = 0.00000
                                                      H 04 = 0.00000
 S:       100.000          F:      1000.000           H 05 = 0.00000
 112.5%   75.000           100.0%  900.000            H 06 = 0.00000
 G:   00 80 99 40 27 90          71 17 94 97          H 07 = 0.00000
 M:   xx xx xx xx                                     H 08 = 0.00000
                                                      H 09 = 0.00000
  PROGRAMS: MAINPROG SUB1 SUB2                        H 10 = 0.00000
  N26    S2000 F200 T3.03                             H 11 = 0.00000
  N27    (UTO,1,X30,Y22)                              H 12 = 0.00000
  N28    (ROT,20)                                     H 13 = 0.00000
  N29    G81 R-90 Z-110 M3                            H 14 = 0.00000
  N30    X25 Y25                                      H 15 = 0.00000
  N31    X40 Y10
  N32    X25
POS DISPLAY   | NEXT DISPLAY   | SELECT PROCESS | ENLARGE        | HELP
E VARIABLES   | SN VARIABLES   | USER VARIABLES | H VARIABLES    | PLUS VARIABLES
FIND PAGE/IND |                | SUBSTITUTE     | EVALUATE       | SC VARIABLES
```

SN variables

System Number (SN) variables are system variables with predifined values that may be used in a part program. These values are stored in the dual port memory and remain unchanged until they are reassigned.

The **SN VARIABLES** softkey permits to visualise or reassign the current values of the SN variables according to specific program needs. When you press the **SN VARIABLES** softkey the following data entry window is displayed:

```
PROCESS CONTROLLED:  1        CAPS ON              SCREEN 1 OF 4      TIME: 9:35:00
 PROC: 1 IDLE          AUTO                                  OPT STOP
       AXIS            WORK                  [mm]             SN VARIABLES  Page 1 of 2
 L     X                    - 149.00000
 LM    Y                       32.25603                       SN 00 = 0.00000
       Z                        0.00000                       SN 01 = 0.00000
       A                        0.00000                       SN 02 = 0.00000
       B                        0.00000                       SN 03 = 0.00000
                                                              SN 04 = 0.00000
  S:       100.000            F:      1000.000                SN 05 = 0.00000
  112.5%   75.000             100.0%  900.000                 SN 06 = 0.00000
  G:   00  80  99  40  27  90          71  17  94  97         SN 07 = 0.00000
  M:   xx  xx  xx  xx                                         SN 08 = 0.00000
                                                              SN 09 = 0.00000
   PROGRAMS: MAINPROG SUB1 SUB2                               SN 10 = 0.00000
   N26    S2000 F200 T3.03                                    SN 11 = 0.00000
   N27    (UTO,1,X30,Y22)                                     SN 12 = 0.00000
   N28    (ROT,20)                                            SN 13 = 0.00000
   N29    G81 R-90 Z-110 M3                                   SN 14 = 0.00000
   N30    X25 Y25                                             SN 15 = 0.00000
   N31    X40 Y10
   N32    X25
POS DISPLAY    NEXT DISPLAY    SELECT PROCESS    ENLARGE    HELP
E VARIABLES    SN VARIABLES    USER VARIABLES    H VARIABLES    PLUS VARIABLES
FIND PAGE/IND                  SUBSTITUTE        EVALUATE   SC VARIABLES
```

SN may be modified with the same procedure described for E variables.

It is possible to define as many as 25 SN variables.

IMPORTANT

The values assigned to SN variables remain stored on a permanent basis even after the system has been switched off.

The value assigned to an SN variable during part program execution will supersede that declared in the data entry window.

SC variables

System Character (SC) variables are system variables with predifined values that may be used in a part program. These values are stored in the dual port memory and remain unchanged until they are reassigned.

The **SC VARIABLES** softkey permits to visualise or reassign the current SC variables to suit specific program needs. When you press the **SC VARIABLES** softkey the following data entry window is displayed:

```
PROCESS CONTROLLED: 1        CAPS ON              SCREEN 1 OF 4      TIME: 9:35:00
 PROC: 1 IDLE        AUTO                 OPT STOP                 RETRACE
      AXIS          WORK          [mm]          PROGRAMMED          ORIGIN
 L    X            - 149.00000                  -149.23600            1
 LM   Y             32.25603                    33.39406             2    TI
      Z             0.00000                     0.00000              0
      A             0.00000                     0.00000              0
      B             0.00000                     0.00000              0

  S:      100.000        F:     1000.000       RAP:     0.0000        ACT  : Txx.xx
  112.5%  75.000         100.0% 900.000        100.0%   xxxxxx        NXT:  Txx.xx
  G:  00 80 99 40 27 90        71 17 94 97                            d:  0.000000
                                   SC VARIABLES
 0000 000   000   000   000   000   000   000   000   000   000   ".............."
 0010 000   000   000   000   000   000   000   000   000   000   ".............."
 0020 000   000   000   000   000   000   000   000   000   000   ".............."
 0030 000   000   000   000   000   000   000   000   000   000   ".............."
 0040 000   000   000   000   000   000   000   000   000   000   ".............."
 0050 000   000   000   000   000   000   000   000   000   000   ".............."

 Address:   0000   Value:                                            Page  1  of  2

 POS DISPLAY    | NEXT DISPLAY   | SELECT PROCESS | ENLARGE        | HELP
 E VARIABLES    | SN VARIABLES   | USER VARIABLES | H VARIABLES    | PLUS VARIABLES
 FIND PAGE/IND  |                | SUBSTITUTE     | EVALUATE       | SC VARIABLES
```

This tables shows 100 3-digit character variables. Variables are divided into 10 strings, each of which occupies a line of the table. The first column specifies the start adress of the string. The last column displays a string of ASCII characters.

start address — *ASCII code of the character* — *ASCII string*

0000	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**". . . . . ."**

To assign an SC variable it is necessary to first define the start address and then specify the ASCII string. The procedure is as follows:

1. Key in the start address in the "*Address*" field.
2. Press **[Tab]** to move the cursor to the "*Value*" field (Press **[Shift Tab]** to move the cursor backwards in order to alter the address).
3. Either the character or its ASCII code may be written in the "Value" field. Press **[Tab]** to move the cursor to the subsequent character or **[Shift Tab]** to move it backwards by one characters. To enter only one character it is necessary to write it between quotes. For example, "A", "B" or "C".
4. Press **[Return]** to visualise the string on the most right column. Then press **[Enter]** to confirm the input data.

IMPORTANT

The values assigned to SC variables remain stored on a permanent basis even after the system has been switched off.

The value assigned to an SC variable during part program execution will supersede the one declared in the data entry window.

Example:
To assign the "COOLANT" value to the SC that starts from address 53 the procedure is as follows:

1. Press the **SC VARIABLES** softkey:

SC VARIABLES

0000000	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**". . . . . ."**
0010000	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**". . . . . ."**
0020000	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**". . . . . ."**
0030000	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**". . . . . ."**
0040000	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**". . . . . ."**
0050000	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**000**	**". . . . . ."**

Address: 0000 Value: **Page 1 of 2**

POS DISPLAY	NEXT DISPLAY	SELECT PROCESS	ENLARGE	HELP
E VARIABLES	SN VARIABLES	USER VARIABLES	H VARIABLES	PLUS VARIABLES
FIND PAGE/IND		SUBSTITUTE	EVALUATE	SC VARIABLES

2. Key in 53, press **[Tab]**, and then key in the following sequence (including the quotes):

"C" Tab "O" Tab "O" Tab "L" Tab "A" Tab "N" Tab "T"

```
                                   SC VARIABLES
0000 000   000   000   000   000   000   000   000   000   000   ".............."
0010 000   000   000   000   000   000   000   000   000   000   ".............."
0020 000   000   000   000   000   000   000   000   000   000   ".............."
0030 000   000   000   000   000   000   000   000   000   000   ".............."
0040 000   000   000   000   000   000   000   000   000   000   ".............."
0050 000   000   000   000   000   000   000   000   000   000   ".............."

Address:   0053  Value:   "C"   "O"   "O"   "L""A"   "N"   "T"   Page  1  of  2

| POS DISPLAY | NEXT DISPLAY | SELECT PROCESS | ENLARGE | HELP |
| E VARIABLES | SN VARIABLES | USER VARIABLES | H VARIABLES | PLUS VARIABLES |
| FIND PAGE/IND |  | SUBSTITUTE | EVALUATE | SC VARIABLES |
```

3. Press **[Return].** The following will appear on the screen:

```
                                   SC VARIABLES
0000 000   000   000   000   000   000   000   000   000   000   ".............."
0010 000   000   000   000   000   000   000   000   000   000   ".............."
0020 000   000   000   000   000   000   000   000   000   000   ".............."
0030 000   000   000   000   000   000   000   000   000   000   ".............."
0040 000   000   000   000   000   000   000   000   000   000   ".............."
0050 000   000   000   067   079   079   076   065   078   084   ". . . COOLANT  "

Address:   0053  Value:   "C"   "O"   "O"   "L""A"   "N"   "T"   Page  1  of  2

| POS DISPLAY | NEXT DISPLAY | SELECT PROCESS | ENLARGE | HELP |
| E VARIABLES | SN VARIABLES | USER VARIABLES | H VARIABLES | PLUS VARIABLES |
| FIND PAGE/IND |  | SUBSTITUTE | EVALUATE | SC VARIABLES |
```

Note that from address 53 onwards each column displays the numerical ASCII code of a character, whereas the string appears on the most right column. An alternative method for writing this sequence would have been entering the following values in step 2:

067 Tab 079 Tab 079 Tab 075 Tab 065 Tab 078 Tab 084

PLUS variables

PLUS variables are system variables defined by the user in the AMP environment that remain stored in the PLUS dual port memory. These variables can be used for reading or writing operations.

There are three types of PLUS variables:

- Long Real (LR)
- Boolean (BL)
- Short (SH)

If no PLUS variable has been configured, the **PLUS VARIABLES** softkey in the VARIABLES menu is disabled. When you press the **PLUS VARIABLES** softkey the following data entry window is displayed:

```
PROCESS CONTROLLED:  1        CAPS ON          SCREEN 1 OF 4      TIME: 9:35:00
 PROC: 1 IDLE          AUTO                                  OPT STOP
       AXIS            WORK                [mm]
 L     X                   - 149.00000                        PLUS VARIABLES   Page 1 of 1
 LM    Y                     32.25603
       Z                      0.00000                         @PROG1 SH = 0.00000
       A                      0.00000                         @PROG2 BL = 0.00000
       B                      0.00000                         @PROG3 LR = 0.00000

  S:       100.000          F:      1000.000
  112.5%   75.000           100.0%  900.000
  G:   00  80  99  40  27  90        71  17  94  97
  M:   xx  xx  xx  xx

   PROGRAMS: MAINPROG SUB1 SUB2
   N26    S2000 F200 T3.03
   N27    (UTO,1,X30,Y22)
   N28    (ROT,20)
   N29    G81 R-90 Z-110 M3
   N30    X25 Y25
   N31    X40 Y10
   N32    X25
POS DISPLAY   NEXT DISPLAY   SELECT PROCESS   ENLARGE   HELP
E VARIABLES   SN VARIABLES   USER VARIABLES   H VARIABLES   PLUS VARIABLES
FIND PAGE/IND                SUBSTITUTE       EVALUATE  SC VARIABLES
```

This window permits to read or write the value of a variable as illustrated in the previous section.

Searching for pages and variable indexes

The **FIND PAGE/IND** softkey facilitates the seach for the index of a variable or a page in a data entry window when it contains too many system variables to use **[PgUp]** and **[PgDn]**. When you press **FIND PAGE/IND** the following page appears:

The window permits to search for either a page number or an index.

1. Key in the page or index number.
2. Press **[Enter].**

Using variables for calculations

In some cases you must calculate the value of a variable before writing it in the data entry window. To make these calculations without closing the windows you can use the following softkeys: **EVALUATE** and **SUBSTITUTE**. The procedure is as follows:

1. Press **EVALUATE**. The following data entry window will be displayed:

2. Key in the mathematical expression to be calculated (algebric or trig functions are allowed). For more information about formulas and syntactic rules, refer to the "Programming Manual". Press **[Return]** to visualise and save the result. Press **[Esc]** to close the EVALUATE window.

IMPORTANT

If you press **EVALUATE** again at this point the result will be cleared.

The result of the calculation may be written in any type of variable with the following procedure:

1. Select the data entry window that corresponds to the type of variable.
2. Position the cursor on the desired variable and press **SUBSTITUTE.** The contents of the EVALUATE data entry window will be assigned to the variable replacing the previous value.

SETTING MACHINE PARAMETERS

Machine parameters are variables that affect system operation. The user can access and modify the following 10 Series machine parameters:

- VFF (Velocity Feed Forward) selects the bevel machining mode.
- Dynamic limits. These three parameters affect:
  - automatic deceleration on the bevels of the path (DLA)
  - maximum deviation angle (MDA) in G27
  - velocify factor (VEF) in G27 on the bevels of the path.
- Spindle speed limits.
- Percentage of return speed in G84.
- Probe parameters.
- Axes references.
- Circular interpolation accuracy.
- Dwell time for G04
- Time and date.

To access these parameters press the **MACHINE SET-UP** softkey, which displays the following submenu:

POS DISPLAY	NEXT DISPLAY	SELECT PROCESS	ENLARGE	HELP
DYNAMIC PARAM	DYNAMIC LIMITS	PROGRAM SET-UP	BLOCK RETRACE	PROBE SET-UP
REF	SET ACCURACY			TIME/DATE SET

The sections that follow describe each parameter with the procedure for setting or altering its value.

To open a data entry window it is necessary to press the corresponding softkey. To confirm a modification either re-press the softkey or press **[Enter]**. To close the window without saving the modifications press **[Esc]** .

Dynamic Params

The **DYNAMIC PARAMS** softkey opens a data entry window similar to the one below:

```
DYNAMIC PARAMETERS

ENABLE VFF (Y/N)          :
G04 DWELL TIME            :
SPINDLE LIMIT (RMP)       :
% TAP RETRACT FEED (G84) :
```

Significance of the fields is as follows:

ENABLE (Y/N) Can have these values:

[Y] Enables VFF in part program execution. When VFF is enabled and the axis velocity is constant the following error will be almost zero.

[N] Disables VFF in part program execution. When VFF is disabled, the following error is proportional to the axis velocity and the ring gain (which can be configured in AMP). Therefore, at constant velocity the following error will be constant.

When a part program is active and the axes move without deceleration on bevels (G28), the trajectory of the axes depends on whether VFF is enabled or disabled. Different shape errors may occur on the bevels:

- if VFF is disabled there is a trajectory error because the axes move beyond the angle;
- if VFF is enabled there is a trajectory error because the angle is truncated.

The figures that follow illustrate four applications of the VFF parameter:

ROUGH EXTERNAL WITHOUT VFF

ROUGH INTERNAL WITHOUT VFF

ROUGH EXTERNAL WITH VFF

ROUGH INTERNAL WITH VFF

G04 DWELL TIME

Defines the end-of-block dwell time (G04).
This time is used in G04 and in the following canned cycle blocks:

- In G94 dwell time is expressed in seconds.
- In G95 dwell time is expressed in revs per min.

For further information on this parameter, see the Programming Manual.

SPINDLE LIMIT (RPM)

Is the maximum spindle speed expressed in rpm when the speed of the tool cutter is programmed in mm/min or inch/min. The allowed range is from 0 to 99999,9999 rpm. The default value is 0 RPM.

% TAP RETRACT FEED

Tapping retract speed. Specifies the percentage of the feedrate applied to the retract phase of the tapping cycle. The allowed range is from 0 to 125%. The default value is 100%.

Dynamic limits

```
S:        100.000          F:       1000.000        RAP:      0.0000          ACT  : Txx.xx
112.5%   75.000           100.0%  900.000          100.0%    xxxxxx          NXT:   Txx.xx
G:   00  80  99  40  27  90          71  17  94  97                          d:  0.000000
M:   xx  xx  xx  xx                                JOG:  0.00000            Z:  0.000000

 PROGRAMS: MAINPROG SUB1 SUB2                          DYNAMIC LIMITS
 N26    S2000 F200 T3.03
 N27    (UTO,1,X30,Y22)                        ENABLE LOOK-AHEAD (Y/N)   :  N
 N28    (ROT,20)
 N29    G81 R-90 Z-110 M3                      MAXIMUM DEVIATION ANGLE   :  90.00000
 N30    X25 Y25
 N31    X40 Y10                                VELOCITY FACTOR           :  0.8
 N32    X25
POS DISPLAY   | NEXT DISPLAY   | SELECT PROCESS | ENLARGE        | HELP
DYNAMIC PARAM | DYNAMIC LIMITS | PROGRAM SET-UP | BLOCK RETRACE  | PROBE SET-UP
REF           | SET ACCURACY   |                |                | TIME/DATE SET
```

ENABLE LOOK-AHEAD (Y/N) Automatic deceleration on bevels. It may have the following values:

[Y] Enables automatic deceleration on bevels during part program execution.

[N] Enables automatic deceleration on bevels during part program execution.

If the selection is N, DLA instructions in the current part program will be ignored.

MAXIMUM DEVIATION ANGLE **(**MDA) Is the maximum angle deviation of the axis for which G27 is active. The selected angle is the operating limit for G27.

If the angle deviation is greater than this limit, the path will be executed in G29 or in point to point.

The allowed range is from 0 to 180 degrees. The default value is 90 degrees.

VELOCITY FACTOR (VEF) permits to regulate axes velocity on bevels in G27. Small VEF values permit to obtain considerable velocity reductions.

The allowed range is from 0 to 999999999999.0. The default value is .8.

To obtain further information about DLA, MDA and VEF refer to the 10 Series Programming Manual.

Program SET-UP

The **PROGRAM SET-UP** softkey opens a data entry window permitting configuration of some parameters conditioning the part program execution mode:

- Block delete
- Optional stop
- Feedrate bypass
- Rapid rate control
- Rotation
- Stock allowance

And for each axis configured, the following parameters may be configured:

- Stand still
- Mirror modes
- Scale
- Scale factor

For further information, see chapter 11 "Part program execution".

Block Retrace

Permits definition, if configured in AMP, of the machine path to be performed backwards.

The softkey is enabled only if a program is selected and if the **CYCLE STOP** pushbutton is pressed during execution.

The sequence of operations is as follows:

- System in CYCLE STOP
- BLOCK RETRACE enabled
- Press CYCLE START; the system moves the axes backwards towards the last point of the buffer.
- Movements will be executed in consistency with the mode selected (AUTO, BLK/BLK).
- To return to normal conditions, disable **BLOCK RETRACE**, exit from CYCLE STOP mode and press CYCLE START.

For further information, see chapter 11 "Part program execution".

Probe parameters

These parameters are used by the system during a probing cycle. When you press the **PROBE SET-UP** softkey, the following data entry window is displayed:

```
S:      100.000        F:      1000.000       RAP:     0.0000          ACT  : Txx.xx
112.5%  75.000         100.0%  900.000        100.0%   xxxxxx          NXT:  Txx.xx
G:  00  80  99  40  27  90        71  17  94  97                       d:  0.000000
M:  xx  xx  xx  xx                              JOG:  0.00000          Z:  0.000000

PROGRAMS: MAINPROG SUB1 SUB2          PROBE SET-UP  [mm]
N26   S2000 F200 T3.03
N27   (UTO,1,X30,Y22)                 APPROACH TOLERANCE     :  5.00000
N28   (ROT,20)                        SAFETY TOLERANCE       :  5.00000
N29   G81 R-90 Z-110 M3               MEASURING FEED         :  90.50000
N30   X25 Y25                         HORIZONTAL SHIFT       :  0.00000
N31   X40 Y10                         VERTICAL SHIFT         :  0.00000
N32   X25                             PROBING MODALITY       :  0

| POS DISPLAY | NEXT DISPLAY | SELECT PROCESS | ENLARGE | HELP |
|---|---|---|---|---|
| DYNAMIC PARAM | DYNAMIC LIMITS | PROGRAM SET-UP | BLOCK RETRACE | PROBE SET-UP |
| REF | SET ACCURACY | | | TIME/DATE SET |
```

The values displayed in the **PROBE SETUP** window are expressed in the default unit configured in AMP (mm or inch). This unit is shown in the window between square brackets.

APPROACH TOLERANCE Is the distance from the nominal probing point. The probe arrives at this distance moving at the rapid rate and then starts the probing cycle at the feedrate programmed in the MEASURING FEED field.

SAFETY TOLERANCE This value must be set in order to preserve the probe. The probe stops if it does not touch the machined part within this tolerance.

MEASURING FEED Is the velocity at which the probe moves.

HORIZONTAL SHIFT Is the requalification value applied to the abscissa of the probe axis expressed in mm.

VERTICAL SHIFT Is the requalification value applied to the ordinate of the probe axis expressed in mm.

PROBING MODALITY Enter the probing modality value: 0 for standard probing - 1 for probing without rapid return.

Axes Reference

```
S:      100.000        F:      1000.000       RAP:     0.0000          ACT  : Txx.xx
112.5%  75.000         100.0%  900.000        100.0%   xxxxxx          NXT:  Txx.xx
G:  00 80 99 40 27 90          71 17 94 97                             d:  0.000000
M:  xx xx xx xx                               JOG:  0.00000            Z:  0.000000

 PROGRAMS: MAINPROG SUB1 SUB2                  AXES HOMING
 N26    S2000 F200 T3.03
 N27    (UTO,1,X30,Y22)                  AXIS NAME        REFERRED (Y/N)
 N28    (ROT,20)                            X                 [Y]
 N29    G81 R-90 Z-110 M3                   Y                 [Y]
 N30    X25 Y25                             Z                 [Y]
 N31    X40 Y10
 N32    X25
POS DISPLAY   NEXT DISPLAY   SELECT PROCESS   ENLARGE   HELP
DYNAMIC PARAM DYNAMIC LIMITS PROGRAM SET-UP   BLOCK RETRACE PROBE SET-UP
REF           SET ACCURACY                                  TIME/DATE SET
```

With the **REF** softkey the axes may be referred to the current position. The procedure is as follows:

1. Select the axes to be referred in the data entry window with the **[Up/Down Arrow keys]**.
2. Key in **[Y]** and press **[Enter]**.

IMPORTANT

Unlike the HOME cycle, referring the axes with the **REF** softkey is a logical operation. Therefore, the NULL OFFSET and HOME POSITION parameters configured in AMP for each axis are ignored. Also, the operating limits are referred to the point on which the axes were when the **REF** procedure was closed with **[Y]**.

Accuracy

```
S:        100.000        F:      1000.000        RAP:      0.0000          ACT  : Txx.xx
112.5%   75.000          100.0%  900.000         100.0%    xxxxxx          NXT:   Txx.xx
G:   00  80  99  40  27  90        71  17  94  97                          d:  0.000000
M:   xx  xx  xx  xx                                JOG:  0.00000           Z:  0.000000

 PROGRAMS: MAINPROG SUB1 SUB2          SET ACCURACY    [mm]
 N26    S2000 F200 T3.03
 N27    (UTO,1,X30,Y22)          CIRCLE ENDPOINT TOLERANCE  :  0.01000
 N28    (ROT,20)
 N29    G81 R-90 Z-110 M3        FULL CIRCLE THRESHOLD      :  0.00100
 N30    X25 Y25
 N31    X40 Y10                  ARC MODALITY               :  0
 N32    X25
POS DISPLAY    NEXT DISPLAY    SELECT PROCESS    ENLARGE    HELP
DYNAMIC PARAM  DYNAMIC LIMITS  PROGRAM SET-UP    BLOCK RETRACE    PROBE SET-UP
REF            SET ACCURACY                                       TIME/DATE SET
```

CIRCLE ENDPOINT TOLERANCE **(CET)** Establishes a tolerance interval for the difference between the start and the final radius of a circle arc.

ULL CIRCLE THRESHOLD **(FCT)** Establishes a threshold for the distance between the first and the last point of a circle arc. If this distance is smaller than FCT, the circle arc is closed.

ARC MODALITY **(ARM)** Selects a normalisation mode for the arc programmed with the coordinates of the center (I, J) and the last point in order to make the arc geometrically congruent. Allowed values are 0, 1, 2, and 3.

Time/date

The **TIME/DATE** softkey permits the machine date and time to be set.

```
S:      100.000        F:      1000.000       RAP:     0.0000          ACT  : Txx.xx
112.5%   75.000        100.0%  900.000        100.0%   xxxxxx          NXT:   Txx.xx
G:   00  80  99  40  27  90        71  17  94  97                      d:   0.000000
M:   xx  xx  xx  xx                            JOG:   0.00000          Z:  0.000000
                                                   TIME DATE SET
 PROGRAMS: MAINPROG SUB1 SUB2
 N26    S2000 F200 T3.03                      HOURS        :  11
 N27    (UTO,1,X30,Y22)                       MINUTES         16
 N28    (ROT,20)                              SECONDS      :  57
 N29    G81 R-90 Z-110 M3
 N30    X25 Y25                               YEAR         :  1993
 N31    X40 Y10                               MONTH           11
 N32    X25                                   DAY          :  15
POS DISPLAY   NEXT DISPLAY   SELECT PROCESS   ENLARGE   HELP
DYNAMIC PARAM DYNAMIC LIMITS PROGRAM SET-UP   BLOCK RETRACE PROBE SET-UP
REF           SET ACCURACY                                  TIME/DATE SET
```

END OF CHAPTER

ESTABLISHING ORIGINS AND HOMING THE AXES

GENERAL

The reference points that permit to calculate the workpiece coordinates can be established on the machine tool. These points are illustrated in the figure below:

The ABSOLUTE ZERO of an axis coincides with its zero microswitch. The position of the zero microswitch is read by the control during the homing procedure described later in this chapter.

The NULL OFFSET position is defined during configuration with respect to the absolute zero. It is normally very near the zero microswitch and serves for correcting the zero microswitch position.

The HOME position is defined during configuration with respect to the NULL OFFSET. It is normally at the centre of the table and represents the MACHINE ZERO.

ABSOLUTE ORIGINS must be defined by the programmer and serve as reference points for workpiece and machining coordinates. ABSOLUTE ORIGINS are referenced to the MACHINE ZERO.

NOTE:
If the HOME position has not been defined, the MACHINE ZERO coincides with the NULL OFFSET or, if this is also missing, with the NULL OFFSET. NULL OFFSET and HOME POSITION values must be specified for each axis in AMP.

HOMING THE AXES

This is the first operation to be performed after system power up.

Homing an axis means making it touch the switch that marks the beginning of its travel.

An axis may be homed in Manual or in Auto mode. Both procedures are described in the sections that follows.

Manual Homing

1. Select the desired axis by moving the selection bar with the **[Up/Down Arrow keys]**.
2. Press the **JOG DIR (+/-)** softkey to select the direction in which the axis will move away from the current position.
3. Press the **HOME** softkey.
4. Press **CYCLE START** and keep it pressed until the axis has been homed. The axis will automatically move towards the zero switch that defines its absolute zero point.
5. Repeat steps 1 through 4 for all the other axes.

Automatic Homing

1. Select the desired axis by moving the selection bar with the **[Up/Down Arrow keys]**.
2. Press the **HOME** softkey.
3. Press **CYCLE START** and keep it pressed until the axis has been homed.
4. Repeat steps 1 through 3 for all the other axes.

In both cases the axis moves automatically towards the zero switch and from here either to the zero position or to the NULL OFFSET position (if the *null offset* parameter has been configured).

After an axis has been correctly homed, the following message is displayed:

NC 150 Axis homed
Distance home marker..

IMPORTANT

When the axis is homed in manual mode the **CYCLE START** pushbutton must not be released until the axis reaches the zero microswitch. If **CYCLE START** is released before homing has been completed, the axis will decelerate to a complete stop. To resume homing, press **CYCLE START** again and keep it pressed until the axis arrives to the zero microswitch.

NOTE:

- If **Manual** homing mode has been configured in AMP and the direction of motion selected in step 2 is different from that configured in AMP, the axis will not move and the following message will be displayed:

  NC108 Home and jog direction not congruent

  Select the correct direction and repeat the homing procedure.

- To abort a homing move press **RESET**.
- If **Automatic** homing has been configured in AMP, the system will automatically force the selected axis to move in the configured direction of motion.

DEFINING ORIGINS

Workpiece or machining coordinates can be referenced to a point located inside or outside the part. This point is known as origin or part zero. In some cases, however, part of the coordinates may be referenced to other points. The system is able to handle up to three different origins, thus eliminating the need to convert the coordinates to a single reference point.

Absolute origins (part zero) These origins are referenced to the machine zero (Home Position) and are stored in the Origins Table. During execution an absolute origin may be invoked at any time by writing its number in the program, after which all of the coordinates will be referenced to this new origin. The system permits to define up to 10 origins per process.

Temporary origins These origins are referenced to an absolute origin and may be defined and enabled from the program by means of the UTO instruction. They are not stored in the Origins Table and remain active until a new temporary origin is programmed (with an UTO), an absolute origin is invoked (with an UAO), or the control is reset. For more information about these origins, refer to the Programming Manual.

Incremental origins These origins are referenced to the currently active origin and may be defined and enabled directly from the program with an UIO instruction. An incremental origin remains active until a new incremental, absolute or temporary origin is invoked or until the control is reset. For more information about these origins, refer to the Programming Manual.

Prior to starting a new execution cycle on the machine it is necessary to establish the absolute origins. Each absolute origin is determined by writing in the Origins Table the distance from the origin point to the HOME POSITION. The procedure is as follows:

1. Home the axes using the procedure described earlier in this chapter.
2. Jog the axes to the point at which the absolute origin must be located (see Chapter 6 for Axes jogging).
3. Press the **SET ORIG./TOOL** softkey in the MANUAL menu. The following data entry window will be displayed:

4. Press **[O]** to select origin presetting. Then move the cursor to the next data entry field and key in the origin number. Press **[Enter]** to confirm the entries. The following data entry window will be displayed:

ORIGIN PRESET

ORIGIN NUMBER : 1

AXIS VALUE

X Y
Z

NOTE:
This data entry window lists the axis names configured in AMP for the current process. In the example it has been assumed that the configured axes are X, Y and Z.

5. Key in the absolute origin offsets for X and Y. If the absolute origin coincides with the point in which the axes have been positioned, key in a 0 offset for X and Y.

The control calculates the absolute origin coordinates referenced to the HOME POSITION (and, therefore, to the zero microswitch) and stores them in the origins table. To visualise this table, use the Table Editor tool described later in this chapter.

An origin becomes active when the control reads the relevant UAO program instruction.

Examples:
The figures below illustrate how to define absolute origins.

1) Absolute origin overlapping with the current axis position.

SET ORIG./TOOL data entry window

Key in X0, Y0.

2) Absolute origin offset by a given value, i.e. absolute origin moved from the current axes position.

Offsets	SET ORIG./TOOL data entry window
10 mm on X axis	
10 mm on Y axis	Key in X10, Y10.

To visualise, alter or cancel absolute origins, refer to Chapter 7.

DEFINING THE PART ZERO ON THE SPINDLE AXIS

Since the system must take into consideration the length of any tool mounted on the spindle, a special procedure must be used for defining the zero of the spindle axis.

To this purpose, a special set of offsets must be defined. For more information about these offsets, refer to Chapter 9 "Tool management".

According to the location of the reference point, there are two methods for defining the absolute origin of the spindle axis:

- reference point on the tip of the tool mounted on the spindle;
- reference point on the spindle nose (no tool mounted on the spindle).

Part zero on tool tip

To determine the axis zero when a tool is mounted on the spindle use this procedure:

1. Home the spindle axis with the procedure described earlier in this chapter.
2. Press the **AUTO** softkey and then press **MDI** to select MDI mode.
3. Enable the desired machining tool with the relevant offset by entering their codes (for example, T123.5M6), press **[Enter]** to confirm them and press **CYCLE START**.
4. Jog the spindle axis until the tool tip touches a known point.
5. Press the **SET ORIG./TOOL** softkey in the MANUAL menu. The following data entry window will appear:

6. Press **[O]** to select origin presetting. Then move the cursor to the next data entry field and key in the origin number. Press **[Enter]** to confirm the entries. The following data entry window will be displayed:

```
ORIGIN PRESET

ORIGIN NUMBER        :  1

AXIS VALUE

X                    Y
Z
```

Key in the distance of the tool tip from the part zero on the spindle axis (Z).

Part zero on spindle nose

To define the part zero when no tool is mounted on the spindle, use this procedure:

1. Take the spindle axis to the HOME POSITION with the procedure described earlier in this chapter.
2. Disable the current spindle axis offset. For example, key in T0M6 in MDI.
3. Measure the distance from the spindle nose to the reference point selected on the part.
4. Press the **SET ORIG./TOOL** softkey in the MANUAL menu. The following data entry window will be displayed:

```
ORIGIN/TOOL PRESET

SELECT (Origin/Tool)   :
NUMBER                 :  1

Push ENTER to access the data entry
```

5. Press **[O]** to select origin presetting. Then move the cursor to the next data entry field and key in the origin number. Press **[Enter]** to confirm the entries. The following data entry window will be displayed:

```
ORIGIN PRESET

ORIGIN NUMBER      :  1

AXIS VALUE

X                  Y
Z
```

6. Move the cursor to the Z axis and key in the measured distance with its sign.

HOMING AND ORIGIN PRESETTING FOR DIAMETER AXES

Diameter axes are programmed with diameter coordinates and are used for drilling and reaming holes. The control commands the moves of the tool towards and away from the spindle rotation axis. The figure shows an example of diameter axes:

Diameter axes must be homed before machining starts. The homing procedure is as follows:

1. Position the spindle axis on a hole to be reamed by inputting the hole coordinates.
2. Select the diameter axis (for example, U) by moving the selection bar with the **[Up/Down Arrow keys]**.
3. Home the diameter axis with the above described automatic or manual procedures (**HOME** and **CYCLE START**, or **JOG DIR +/-**, **HOME** and **CYCLE START**).
4. Key in an instruction block programming hole reaming with the hole coordinates. For example :

   G81 .. R .. Z .. F .. S .. M13

   Then press **CYCLE START**.
5. If the hole is executed properly, move the tool away from it without moving the U axis.
6. Measure the hole dimensions accurately. In the example, the diameter measures 98.73 mm.
7. Define the origin of the diameter axis by writing the measured value with a negative sign (in the example, U = -98.73) in the ORIGIN PRESET data entry window.

If during the machining cycle the tool is found to be too worn out for the configured tolerance, the diameter axis value (U in our example) may be updated in the ORIGIN PRESET data entry window by writing the exact diameter of the last machined hole with a negative sign. For example, if the diameter of the last machined hole is 119.95, key in the following: ***U -119.95***

END OF CHAPTER

AXES JOGGING AND STOP FUNCTIONS

JOG

There are two different types of manual or jog moves:

- continuous jog
- incremental jog

To perform a jog move, press the **MANUAL** softkey. The following menu will be displayed:

POS DISPLAY	NEXT DISPLAY	SELECT PROCESS		HELP
HOME	JOG INCR	JOG RETURN	HPG	MANUAL SETUP
XXXXXXXX+	XXXXXXXX-	OVERRIDE SEL	JOG DIR+	SET ORIG./TOOL

This menu includes all the softkeys that permit to carry out manual moves and related operations. In the sections that follow we assume that you have displayed this menu before starting a jogging move.

NOTE:
When the **MANUAL** softkey is pressed, the control defaults to continuous jog mode.

The selected axis for the motion will be emphasized by a selection bar; besides the axis name will be contained in square brackets.

Example:

[x]

Continuous Jog

This mode permits to jog the axes continuously in the selected direction at the selected rate. The procedure is as follows:

1. Select the desired axis by moving the selection bar with the arrow keys.
2. Select the jog direction by pressing the **Jog DIR** softkey. The softkey label will change according to the selection (**JOG DIR -**, **JOG DIR +**).
3. Select the jog rate and the override percentage as described in the "Feedrate Override" section of this chapter.
4. Press **CYCLE START** to start jogging the axis. The axis will continue to move at the selected rate in the desired direction as long as **CYCLE START** is kept pressed.

Incremental Jog

This mode permits to jog the axes by a given increment in the selected direction at the selected rate. The procedure is as follows:

1. Select the desired axis by moving the selection bar with the arrow keys.
2. Select the jog direction by pressing the **JOG DIR** softkey. The softkey label will change according to the selection (**JOG DIR -**, **JOG DIR +**).
3. Select the jog rate and the override percentage as described in the "Feedrate Override" section of this chapter.
4. Select the increment with the **MANUAL SET-UP** softkey (refer to the "Jog Increment" section later in this chapter). If this step is omitted, the system will apply the current increment, i.e. the latest selected one.
5. Press **JOG INCR** and then the **CYCLE START** softkey.

Depending on the JOG INCR selection in the MANUAL SET-UP data entry window, there are two possible incremental jog modes:

Manual: keep **CYCLE START** depressed until the move has been completed. If **CYCLE START** is released before the end of the move, the axis will decelerate to a stop. To complete the move, press **CYCLE START** again and keep it pressed until the move has been terminated.

Automatic: this mode permits to release **CYCLE START** before the move has been completed without stopping the axis.

DEFINING THE JOG INCREMENT

Prior to starting an incremental jog it is necessary to define the jog type and the increment. To make these selections, press the **MANUAL SET-UP** softkey.

The following data entry window will be displayed:

```
                    MANUAL SET-UP  [mm]

INCREMENT VALUE                  :  0.10000

JOG RETURN / JOG  INCR MODE
SELECT (Auto/Manual)             :  M

TOOL DIRECTION (Y/N)             :  N
```

INCREMENT VALUE key in the manual increment value in this field. This value will also be displayed in the JOG field of the main video screen.

MODE select the JOG RETURN or JOG INCR mode, which can be automatic **[A]** or manual **[M]**. For more details about JOG RETURN, see later in this chapter.

TOOL DIRECTION this parameter is used for removing the tool from the part in special cases (see "Removing the tool after an emergency" later in this chapter).

To confirm the entries press the **MANUAL SET UP** softkey or the **[Enter]** key.

FEEDRATE OVERRIDE

The maximum jog rate defined in the configuration may be altered with the following procedure:

1. Press **OVERRIDE SEL** until the first two softkeys on the third line display the following labels: **FEEDMAN +** and **FEEDMAN -**.

2. Press **FEEDMAN +** or **FEEDMAN -** to increment or decrement the configured rate. Each time one of these softkeys is depressed, a 12.5% increment or decrement will be applied to the configured value. Feedrate override percentages range from 0% through 100%.

Example:
If the maximum configured feedrate is 2000 MMPM (millimetres per minute) and a 50% feedrate decrement is applied, the actual jog rate will be 1000 MMPM.

ALTERING THE JOG INCREMENT

To alter the jog increment selected in the **MANUAL SETUP** window use this procedure:

1. Press **OVERRIDE SEL** until the first two softkeys on the third line display the following labels: **FEEDMAN +** and **FEEDMAN -**.

2. Press **JOG INCR +** or **JOG INCR -** to increment or decrement the jog increment value.

   The allowed JOG INCR values are shown in the table below:

mm	inch
0,0001	0,00001
0,001	0,0001
0,01	0,001
0,1	0,01
1,0	0,1
10,0	1

   Before **OVERRIDE SEL** is pressed, the JOG INCR value is the one written in the MANUAL SET UP window (see earlier in this Chapter). As the **OVERRIDE SEL** softkey is pressed, JOG INCR returns to the value it had the last time the **OVERRIDE SEL** softkey was pressed. This value will be one of the above listed increments and can therefore be very different from the one written in the MANUAL SET-UP window.

   Each time **JOG INCR+** or **JOG INCR-** is pressed, the increment/decrement listed in the table is applied to the JOG value.

JOG RETURN TO THE PROFILE

It is possible to interrupt machining during part program execution by pressing the **CYCLE STOP** pushbutton. This puts the system on hold and permits to jog the axes away from the profile.

Part program execution can be resumed according to one of the following three procedures:

1. Moving the axes back to the position on the profile they occupied before program execution was halted.

Proceed as follows:

a) Select the axis to be returned to the profile.

b) Select the axis feedrate percentage desired.

c) Press the **JOG RETURN** softkey and then the **[CYCLE START]** button.

- if JOG RETURN is in "Manual" mode, keep **[CYCLE START]** pressed until the "AXIS ON PROFILE" message appears; this tells you that the axis return move has been completed.
- if the JOG RETURN is in "AUTO" mode, just press **[CYCLE START]** once. The system will automatically move back to the profile ALL the axes previously displaced by following the same trajectory in space in the reverse direction, so as to avoid obstructions. To abort automatic return to the profile press [**CYCLE STOP]**.

2. Moving the axes back to the final point of the profile interruption entity

Proceed as follows:

a) From the **MANUAL SET-UP** Data Entry window activate the “SKIP JOG RETURN” option by entering “Y” and confirm with **[Enter]**.

b) Quit the HOLD status by pressing **[CYCLE STOP]**.

c) Resume program execution by pressing **[CYCLE START].** At this point, the axes will perform a linear movement from their current position to the final point of the movement interrupted by pressing **[CYCLE STOP]**. At the end of this positioning process, program execution will be resumed automatically.

This procedure cannot be performed in the following conditions:

- G61 on.
- Canned cycles and probe function on.
- TCP with mno/ijk programming or with rotary axes in "following" mode (interpolation with following mode).
- Program block in Hold without axis movements.

In such cases, the standard execution is forced and it proves necessary to move the axes back to the profile interruption point according to procedure **1.**

SPINDLE SPEED OVERRIDE

The programmed spindle speed can be altered during execution with the following procedure:

1. Press **OVERRIDE SEL** until the first two softkeys on the third line display the following labels: **SPND OVER +** and **SPND OVER -**.

2. Press **SPND OVER + or SPND OVER -** to increment or decrement the programmed spindle speed. Each time one of these softkeys is depressed, a 12.5% increment or decrement will be applied to the programmed value. Spindle speed override percentages range from 75% through 125%.

Example:
If the maximum programmed spindle speed is 1500 RPM and a 80% feedrate decrement is applied, the actual spindle speed will be 1200 RPM.

STOP FUNCTIONS

There are various ways of stopping program execution and machine operation: **Reset**, **Hold**, **Emergency Stop**, **Active Reset**.

Reset

The reset function permits to:

- Stop axes moves.
- Clear the program execution buffer.
- Restore the absolute origin (Home Position).
- Restore the configured G codes.
- Positions the part program at file start.

To perform a reset function press the **RESET** pushbutton.

To resume program execution press the **CYCLE START** pushbutton. Execution will restart from the beginning of the program.

IMPORTANT

Depending on how the machine logic has been programmed, it will give some of its typical stop commands (such as spindle stop, coolant off, etc.) when it receives the reset command.

Hold

The hold function permits to:

- Stop the axes with controlled deceleration.
- Temporarily stop program execution.

To put the axes on hold press **CYCLE STOP**.

To resume execution press the **CYCLE STOP** pushbutton again to exit the hold condition and then press **CYCLE START**. Execution will restart from the point at which the **CYCLE STOP** command was given.

If manual moves have been performed after pressing **CYCLE STOP**, it is necessary to re-position the axes on the profile before resuming automatic execution. To do this, use the above described procedure.

EMERGENCY STOP

The **EMERGENCY STOP** pushbutton stops power supply to the system. It is not available on the control panel but is included in the machine logic and in the MTB panel.

ACTIVE RESET

This command is available only in the PLUS logic program. It can be used after a cycle stop in order to clear either the remaining portion of the current motion block or the subsequent part program blocks.

To allow automatic active reset, with PLUS the ACTIVE RESET function is available on an external pushbutton or on the machine devices. For example, when the microphone detects the contact between the tool and the part.

During motion block execution, the ACTIVE RESET command is allowed if the system is on HOLD. In addition, the interrupt move must be point to point (G29) with linear (G1) or circular (G2, G3) interpolation. When an ACTIVE RESET is given, the system:

a) resets the interrupted move.

b) searches the subsequent program block for execution. This block will be calculated from the current position.

c) executes assignment blocks, three-letter code instructions, branching instructions, etc. during the search.

d) after the subsequent move has been terminated, the ACTIVE RESET function is reset and the system does not re-launch the part program.

When a part program is selected, the ACTIVE RESET command is allowed if the system is in IDLE status. In this case the system responds with steps b through d.

If an ACTIVE RESET command is given when the system is neither on HOLD nor in IDLE status, the following error message will be displayed: "Command not congruent with the system status".

ACTIVE RESET can be given in AUTO, MDI and BLK/BLK. If the control is in any other mode, the following message will appear: "Bad selected mode".

If MDI (Manual Data Input) mode is active, ACTIVE RESET will reset only the interrupted move. All the instructions that have not been transferred to PLUS will be lost.

If ACTIVE RESET is allowed but the interrupted block is not a point to point move with linear or circular interpolation, the "Active reset not permitted" message will be displayed. If the interrupted block has been stopped at interpolation end, the ACTIVE RESET command will be ignored.

The subsequent block must be a linear move (G1); if it is not, the "Active reset not permitted" error message will be displayed. If the subsequent block is a contouring move (G27/G28), the ACTIVE RESET function will be allowed but the new block will be executed as a point to point move.

All the M, T, S and logic functions that are programmed in the interrupted block or in subsequent blocks and have not been transferred to PLUS will be lost.

The new block calculated by the ACTIVE RESET command contains the coordinates of the axes programmed in the subsequent block.

If the ACTIVE RESET command has been executed successfully, the system switches from HOLD to HOLD RUN (HRUN). The red LED turns off and the system awaits a **CYCLE START** command.

If an ACTIVE RESET function generates an error (for example, if the cycle start coordinate coincides with the subsequent motion block) the system remains on HOLD (red LED ON) but part program execution does not resume when you give a **CYCLE START** command. All the subsequent ACTIVE RESET functions will be allowed and the error will be removed.

Since during an active reset the system reads and executes part program blocks, some of the errors displayed may be numerical control errors. However, the "End of file" message is not an error but a signal. When this message occurs, the system status does not shift to HRUN and **CYCLE START** permits to execute the block that has been restored after one or several ACTIVE RESET commands.

When the active reset function is not accepted by the system, further active request commands shall be taken to refer to the following blocks and, in the absence of errors, the active reset function for such blocks shall be accepted.

The block-end M functions programmed in the interrupted block will be "forgotten" after an active reset. Execution of the subsequent block will start when the **CYCLE START** command is given. If other ACTIVE RESET commands are given, the M codes programmed in these blocks will also be ignored.

G29 G01 X100 M05	Active reset to X=50 (M05 will be ignored)
G01 Y100 M06	Active reset (M06 will be ignored)
G01 M03 Y100 M05	Cycle start :
	1. M03
	2. Move to X50 Y100
	3. M05

An ACTIVE RESET function accepted by the system will enable the subsequent block. This can be observed on the display, where the enabled block will appear in reverse mode.

Example 1

X10 Y40

Y20 Active reset during execution of this block

X100 Y30 X100 at the end of this block

Example 2

X10 Y40

X50 Y10 Active reset during execution of this block

X100 Y30 Y30 X100 at the end of this block.

REMOVING THE TOOL AFTER AN EMERGENCY

In some special applications (such as a 4- or 5-axis machines performing inclined machining in which the moves in the direction of the tool axes are the result of interpolations by other axes), execution may be interrupted by an emergency stop (voltage drop, general emergency, etc.) or by a programmed stop followed by manual moves.

An example of this is shown in the picture below.

This machine has three normal axes (X, Y and Z) and two rotary axes that incline the spindle carrier head.

When an emergency stop occurs, it sometimes happens that the tool remains inside the part. To remove the tool from the part after the machine has been re-powered up, the system must offer you the possibility to move the spindle in the tool direction.

10 Series CNC generates this move automatically, by introducing a virtual axis assigned to a name written in AMP. The moves of this virtual axis cause linear axes to move, so that the tool moves in the linear axes direction, i.e. in the direction indicated by the position of the rotary axes.

For the system to generate the virtual axis you must set a parameter in the data entry window opened by the **MANUAL SETUP** softkey.

The procedure is as follows:

1. After the system has been re-powered up press **MANUAL** and **MANUAL SETUP**. The following data entry window will appear:

```
                    MANUAL SET-UP  [mm]

INCREMENT VALUE                  :  0.10000

JOG RETURN / JOG  INCR MODE
SELECT (Auto/Manual)             :  M

TOOL DIRECTION (Y/N)             :  N
```

2. Write **[Y]** in the TOOL DIRECTION field (in all other cases TOOL DIRECTION must be set to **[N]**). Press **[Enter]** to confirm the entry. There are three possible cases:

   a) *The rotary axes are not referenced and the interrupt occurred during a cycle that included a TCP instruction.*

   In this case the following data entry window will appear:

```
                 SET ROTARY AXIS POSITION

HORIZONTAL AXIS                  :  X.XXXXX

VERTICAL AXIS                    :  X.XXXXX
```

   The meaning of these parameters is as follows:

   Horizontal axis: horizontal rotary axis coordinate in degrees.
   Vertical axis: vertical rotary axis coordinate in degrees.

   These parameters show the rotary axes position before the interrupt. They can be modified, which permits to change the tool withdrawal direction.

   b) *The rotary axes are not referenced and the interrupt occurred during a cycle that did not include a TCP instruction.*

   In this case the SET ROTARY AXIS POSITION window appears but the displayed values are not valid and the axes coordinates must be keyed in by the operator.

   c) *The rotary axes are referenced and the interrupt occurred during a cycle that did not include a TCP instruction.*

   In this case, the SET ROTARY AXIS POSITION window does not appear because the tool direction is the same as the rotary axes direction.

3. In cases A and B press **[Enter]** or the **MANUAL SETUP** softkey.

Now the system will generate a virtual axis that has the same direction as the tool and can be jogged or moved with part program instructions.

NOTE:
If only one of the rotary axes is referenced, the system displays the non-referenced axis coordinate and responds as shown in the previous section.

See also Tool Center Point, mode 5, in the Programming Manual.

END OF CHAPTER

USING TABLES

10 Series CNC handles a great number of parameters. To facilitate operation, this information is organized in tables, which are resident in the dual port memory and can therefore be seen both by the system software and the machine logic (PLUS). These tables remain stored in the dual port memory after power has been removed from the system.

10 Series CNC tables include:

- Origins Table
- Tools Table
- Tool Offsets Table
- User Tables
- Tool Data Base
- Magazine (optional).

Each of these tables is described later in this chapter. The tables are managed by the Table Editor, a utility that also allows editing of the tables stored on the hard disk. Here are some of the editing operations available with the Table Editor:

- Table uploading, opening and saving
- Modification of table data
- Printing
- Incremental modifications in table fields
- Measure unit conversion (mm/inch)
- Tool data base compilation
- Table backup and restore.

The Tool Data Base resides in the hard disk and contains general tool information that can be used for making graphic verifications as well as for compiling the Tool Table.

10 Series tables are described in the sections that follow. All of them can be configured and customised by means of the User Configuration Interface. The video pages, the softkeys and the data entry windows of this interface are described in detail in Chapter 8, to which reference should be made.

WHAT IS A TABLE?

A table is a set of records, each of which consists of a series of fields. Records correspond to the table lines, whereas fields correspond to the table columns. Tables provide an economical way of visualising and accessing large amounts of information.

Because of space restrictions, the fields visualised on one table line are only part of the information stored in that table.

For example, each record of the Tools Table corresponds to a tool whereas the visualized fields are only part of the tool parameters handled by the table. To visualise all of the tool parameters it is necessary to select the desired tool and them press the **EDIT** softkey as shown in the figure below:

The rotary axes are not referenced and the interrupt occurred during a cycle that included a TCP instructionWhen you open a table (refer to the "Opening a Table" section), the first 10 records and the main fields defined with the Configurator will be displayed.

USING THE TABLE EDITOR

To enable the Table Editor press the **TABLES** softkey in the main menu. The main Editor screen will appear:

A *status line*
B *status window*

The status line provides the following information:

- Name of the utility (Table Editor)
- Current date and time.

The status window displays the following information:

Measurement unit: (mm or inches) is the current measurement unit.

Data set: is the selected storage area. It can be Memory (for the dual port memory) or the name of the file that contains the table).

Increment: is the current increment defined with the **SET INCR** softkey for the table environment.

Selected Table: is the table selected by means of the relevant softkey.

Sort selected page: is the selected page.

Current process: is the active process.

Current magazine: is the number of the active tool magazine (if present).

The meaning of the softkeys available with the main editor screen is shown in the table below:

SOFTKEY	FUNCTION
BACKUP MEMORY	makes a backup copy on the hard disk of all the tables stored in the dual port memory.
RESTORE MEMORY	restores the backup copy of all the tables stored in the dual port memory.
MAGAZINE	allows access to the Magazine Table for data entry or modification.
HELP	displays the on line help page that corresponds to the current data entry window or softkey menu.
TOOL	allows access to the Tools Table for data entry or modification.
OFFSET	allows access to the Tool Offsets Table for data entry or modification.
USER	allows access to the User Table for data entry or modification.
SETUP	allows to configure the tables used by the Table Editor.
ORIGIN	allows access to the Origins Table for data entry or modification.
USER TABLE	enables a full user-defined management of the user table.
TOOL DATA BASE	allows access to the Tool Data Base.
SELECT PROCESS	selects the process to which the origins apply. Origins tables are process-specific. All the other tables are shared by the various processes. In other words, each process has its own Origins Table, which is invoked when the process is selected.
SELECT MAGAZINE	selects the magazine to which the Tools Table applies (optional).
SET INCR.	selects the value of the increment to be applied to a parameter.
EXIT	allows to quit the Table Editor and return to the main softkey menu.

Directory Window

After selecting the Table Editor with the **TABLES** softkey and invoking the process with the **SELECT PROCESS** softkey, any the following softkeys permit to visualise the Directory Window of the Tools Table: **BACKUP MEMORY, RESTORE MEMORY, MAGAZINE, TOOL, OFFSET, USER, ORIGIN.**

The possible choices are the tables resident in the dual port memory (visualised as MEMORY) or a table resident in the hard disk (visualised with its name).

Softkeys common to all Tables

The following menu of sofkeys is common to all tables:

LOAD MEMORY	SAVE MEMORY	PRINT	TOOL DATA BASE	HELP
INSERT	EDIT	DELETE	FIND	SORT
MULTI-CUTTER	ADD INCR	SUB INCR	METRIC/INCH	EXIT

The meaning of these softkeys is illustrated in the table below:

SOFTKEY	FUNCTION
LOAD MEMORY	loads the specified table from the hard disk into the dual port memory.
SAVE MEMORY	saves the table resident in the dual port memory.
PRINT	prints the specified table.
TOOL DATA BASE	picks up data about the specified tool from the Tool Data Base (valid only for the Tools Table during the Editing phase).
HELP	visualises the on-line help page for the current data entry window or softkey menu.
INSERT	(active in the Tools Table and the Tool Data Base) permits to insert a new tool in the Tools Table or the Tool Data Base.
EDIT	opens a data entry window containing all the records in a table for editing purposes.
DELETE	cancels the selected record from the Tools Table or the Tool Data Base. In the Tool Offsets, Users or Origins tables it sets the fields of the selected records to zero.
FIND	searches for the element specified in the first column in the table. To shift to another type of element, press the **SORT** softkey.
SORT	permits to arrange the columns of the selected table in a given order.
MULTI CUTTER	(available only in the Tools Table and the Tool Data Base) permits to define multicutter tools with up to 3 cutters.
ADD INCR	each time this key is pressed the selected increment is added to the current field of the visualised table.
SUB INCR	each time this key is pressed the selected increment is subtracted from the current field of the visualised table.
METRIC/INCH	toggles between metric and inch measuring systems.

SOFTKEY	FUNCTION
EXIT	quits the current table and returns to the main Table Editor menu.

Table Editor Keys

The following keys are active in the Table Editor environment:

KEY	FUNCTION
PAGE↑	visualises the previous page of the current table.
PAGE↓	visualises the next page of the current table.
▼ ▲	moves the cursor upwards by one field.
◀ ▶	moves the cursor downwards by one field.
ESC	permits to quit the Edit mode and return to the current table without saving the alterations. Also permits to quit the Directory window and return to the main Editor page.
↵ ↵	confirms a data entry window.

TABLE EDITOR FUNCTIONS

Opening a Table

To open a table use this procedure:

1. Press the **TABLES** softkey from the main menu. The main Table Editor video page will be displayed.
2. Select the desired table (Tools Table, Tool Offsets Table, User Table, Origins Table) by pressing the relevant softkey. The Directory window will appear. Select a table by moving the selection bar with the cursor keys or key in the table name in the data entry field.

- To open a table resident in the dual port memory move the bar to MEMORY.
- To open a table resident in the hard disk select a table from the directory by moving the selection bar with the cursor keys or key in the table name in the data entry field. If the input table name does not exist a new table will be created.

3. Press **[Enter]** to confirm the selection.

Loading a Table

To load into the dual port memory a table that resides in the hard disk use this procedure:

1. Open the table by pressing the relevant softkey (**TOOL**, **ORIGIN**, etc.) and select MEMORY from the directory window.
2. Press **LOAD MEMORY**. The directory window will be displayed.
3. Select the name of the desired table from the directory by moving the selection bar with the cursor keys or key in the table name in the data entry field
4. Press **[Enter]**.
5. The system prompts for confirmation that the file selected is to be loaded; if the answer is **[N]**, the table is not loaded. For an affirmative answer, on the other hand, the system behaves differently depending on whether or not the *magazines* option is present:

   - If the *magazines* option is missing and the answer to the prompt is **[Y]** the table is loaded immediately.
   - If the *magazines* option is present and the answer to the prompt is **[Y]**, the data entry window **LOAD POCKETS** is opened, and the table is opened after the fields of this window are filled in:

LOAD POCKETS

Magazine on file :

Load Pockets (Y/N) :

The **MAGAZINE ON FILE** field represents the number of the active magazine, whereas **LOAD POCKETS** is for selection of whether or not the pockets recorded in the file are to be loaded.

Altering a Table

To alter the contents of a table resident in the dual port memory use this procedure:

1. Select MEMORY from the directory window of the table.
2. Position the selection bar on the record to be altered.
3. Press the **EDIT** softkey to visualise the fields of the selected record. These fields will appear in the window below:

Fill in and modify the window fields with the same procedures as any other data entry window. Then:

- Press **[Enter]** to confirm the input data and close the data entry window.
- Press **[Esc]** to close the data entry window without saving the alterations. Use this mode when the table is opened for visualization purposes only.

To modify a table resident in the hard disk, write the table name in Step 1.

Incremental Parameter Modification

The contents of a field can be modified by applying a positive or negative increment. The procedure is as follows:

1. To set the increment value, press the **SET INCR** softkey. The following data entry window will be displayed:

Enter increment value :	**0.00000**

2. Key in the increment and press **[Enter]**.
3. Open the table to be altered.
4. Select the record to be incremented.
5. Press the **EDIT** softkey. The record fields will be displayed in a data entry window and the **ADD INCR** and **SUB INCR** softkeys will be enabled.
6. Position the cursor on the field to be incremented/decremented.
7. Press the **ADD INCR** softkey to make an increment and **SUB INCR** to decrement the field value. Press these softkeys as many times as necessary until the desired value is reached.

Inserting a Record in a Table

To insert new data in the Origins, Tool Offsets, Users or Magazine tables it is necessary to select the desired record with the bar and then press the **EDIT** softkey to open the data entry window containing all the table parameters. See the individual tables later in this chapter for details on the parameters of each.

If the selected record has never been invoked before, most parameters in the window are set to zero. To alter the displayed value key in the new figures and press **[Enter]** to confirm them. For further information on how to make these alterations, refer to the description of the table later in this chapter.

The records in the Tools Table and the Tool Data Base are not numbered. They are identified by the code assigned to the tool. This means that it is not possible to select a record from one of these tables unless it has been previously associated to a tool.

To insert a new tool code in the Tools Table and the Tool Data Base, use this procedure:

1. Press the **TABLES** softkey and then press **TOOL** or **TOOL DATA BASE** to select the table.
2. Press **INSERT**. A data entry window will be displayed containing all the parameters handled by the table. All of the numerical values will be 0. For details, refer to the "Tools Table" and "Tool Data Base" sections later in this chapter.
3. Key in the desired tool code and the parameters associated to it and press **[Enter]**. The data entry window will be closed and the new code will be included in the table.

Canceling a Record from a Table

To cancel a record from the Origins, Tool Offsets, Users or Magazine tables means to set all the record parameters to zero. To cancel a record from the Tools Table and the Tool Data Base means to eliminate it from the table.

In both cases the procedure is as follows:

1. Open the table by pressing the **TABLES** softkey followed by the softkey of the table to be altered.
2. Position the selection bar on the record to be canceled.
3. Press **DELETE**. In the Tools Table or the Tool Data Base the record will be suppressed. In all the other tables the parameters will be set to zero.

Saving a Table

The following procedure permits to save on the hard disk a table resident in the dual port memory:

1. Open the table by pressing the relevant softkey (**TOOL**, **ORIGIN**, etc.) and select MEMORY from the directory window.
2. Press the **SAVE MEMORY** softkey. The directory data entry window will be displayed.
3. Key in a file name to create a new file or select a name from the directory window to overwrite an old file.
4. Press **[Enter]**.
5. If the file already exists, a request for confirmation will be displayed. Press Y to save the table and N to abort the procedure without saving the table.

This procedure can also be used for creating in the dual port memory a new table based on an old one.

The new file will be saved in the E:\TBL directory with the typical table extension (.TOL, .OFS, .USR, .ORG). From this directory it can be copied to a floppy disk with DOS SHELL.

Printing a Table

When a printer is connected to the control CPU it is possible to print the table.

NOTE:
This feature is available only when the table is open.

The procedure is as follows:

1. Check that the printer is on and connected to the control.
2. Open the table to be printed.
3. Press the **PRINT** softkey. The following data entry window will be displayed:

PRINT TABLE

Enter printer device :

4. Key in the name of the printing device: LPT1, LPT2, LPT3 or PRN.
5. Press **[Enter]**.

Sorting Lines and Columns

This option permits to alter the sequence in which the table columns and lines are displayed. Each table can be rearranged according to specific criteria and then personalised by means of the Table Editor Configurator (see Chapter 8). The table layout affects the **FIND** function, because the search for an element operates on the first column.

NOTE:
The SORT option is available only if the table is open.

To modify the table layout use this procedure:

1. Press the **SORT** softkey. The following data entry window will be displayed (the example refers to the Tool Table):

```
SORT CRITERIA

1 -   Tool & status criteria
2 -   Pocket & tool criteria
3 -   Tool & lifetype criteria
4 -   Tool & user variable
5 -
6 -

Enter sort criteria :  1
```

2. Key in the number of the desired option.
3. Press **[Enter]**.

Search for an Element

This option permits to seach for an element displayed on the first column of the table. If the desired element is not on the first column, use the **SORT** to alter the table layout.

NOTE:
This option is available only if the table is open.

To search for an element use this procedure:

1. Press **FIND**. The following will be displayed:

Search element :

1. Key in the index of the element.
2. Press **[Enter]**.

Changing the Measuring Unit

Table parameters may be expressed and displayed in mm or inches. The **METRIC/INCHES** softkey permits to toggle between both measuring systems.

Press **METRIC/INCHES** to automatically convert all the parameters into the alternative measuring unit. The current measuring system is displayed on the status window.

A mesuring system remains active until the alternative one is selected.

Table Backup

All the tables resident in the dual port memory, i.e. Origins, Tools, Tool Offsets, User, Tool Data Base and Magazine, can be backed up on the hard disk.

The back up procedure is as follows:

1. Press **BACKUP MEMORY**. The system displays the directory window with the existing backup copies.
2. The backup copy can overwrite an old file or be saved under a new name:
   - To overwrite an old backup copy, select the old name from the directory window. A prompt for confirmation will be displayed. Press **[Y]** to overwrite the old backup copy or **[N]** to abort the procedure without making the backup copy.
   - To make a new backup copy key in the new name in the data entry window. In this case no confirmation prompts are displayed.
3. Press **[Enter]**.

The backup copy can be used for tool management. For example, to tables stored in the dual port memory can be duplicated on the hard disk, modified and saved with a different name. This permits to have two different tool management routines on the same support.

Backup files are saved in the E:\TBL directory with the .IMA extension. From this directory they can be copied into a diskette by means of the DOS SHELL.

Restoring Tables

The following procedure permits to recopy into the dual port memory a backup file resident in the hard disk:

1. Press the **RESTORE MEMORY** softkey. The system displays the directory window listing all the backup copies..
2. Position the selection bar on the desired backup copy.
3. Press **[Enter]**.
4. The system displays a confirmation prompt. Press **[Y]** to restore the copy or **[N]** to abort the procedure.

WARNING

When a backup file is restored the tables currently residing in the dual port memory will be lost.

IMPORTANT

With this editor it is not possible to delete tables on disk.
This operation may be executed with the "Reset Tables" utility described in the "AMP software characterization" manual.

ORIGINS TABLE

The Origins Table is exclusive of each process, whereas the remaining tables are common to all the processes. For each configured process it is possible to define up to 10 origins.

To enable an origin from the part program write the UAO code followed by the number of the origin associated to that process. To disable an origin write the UAO code followed by 0. When you open the Origins Table (refer to the "Opening a Table" section in this chapter), the system visualises the following page:

TABLE EDITOR **DATE: 10/01/94** **TIME: 9:35:00**

Measurement unit	:	Mm
Data set	:	MEMORY
Increment	:	0.00000
Selected Table	:	ORIGIN
Sort selected page	:	1 of 1
Current process	:	1
Current magazine	:	1

Origin	Axis 1	Axis 2	Axis 3	Axis 4	Axis 5
000001	118.32000	93.67000	37.90000	0.00000	0.00000
000002	28.57000	327.00000	0.00000	0.00000	0.00000
000003	325.00000	975.00000	0.00000	0.00000	0.00000
000004	298.09000	980.00000	0.00000	0.00000	0.00000
000005	0.00000	0.00000	0.00000	0.00000	0.00000
000006	0.00000	0.00000	0.00000	0.00000	0.00000
000007	0.00000	0.00000	0.00000	0.00000	0.00000
000008	0.00000	0.00000	0.00000	0.00000	0.00000
000009	0.00000	0.00000	0.00000	0.00000	0.00000
000010	0.00000	0.00000	0.00000	0.00000	0.00000

LOAD MEMORY	SAVE MEMORY	PRINT	TOOL DATA BASE	HELP
	EDIT	DELETE	FIND	SORT
MULTI CUTTER	ADD INCR	SUB INCR	METRIC/INCH	EXIT

The highlighting bar is on the first origin, which displays the values of five different axes. To alter the coordinates of the axes that define an origin, select the origin with the bar and press the **EDIT** softkey. The system will open the following data entry window (see the "Modifying a Table" section in this chapter):

TABLE EDITOR **DATE: 10/01/94** **TIME: 9:35:00**

Axis X	**Origin value**	**:**	**118.32000**
Axis Y	**Origin value**	**:**	**093.67000**
Axis Z	**Origin value**	**:**	**037.90000**

Measurement unit	**:**	**Mm**
Data set	**:**	**MEMORY**
Increment	**:**	**0.00000**
Selected Table	**:**	**ORIGIN**
Sort selected page	**:**	**1 of 1**
Current process	**:**	**1**
Current magazine	**:**	**1**

Axis 3	Axis 4	Axis 5
37.90000	0.00000	0.00000
0.00000	0.00000	0.00000
0.00000	0.00000	0.00000
0.00000	0.00000	0.00000
0.00000	0.00000	0.00000
0.00000	0.00000	0.00000
0.00000	0.00000	0.00000
0.00000	0.00000	0.00000
0.00000	0.00000	0.00000
0.00000	0.00000	0.00000

LOAD MEMORY	SAVE MEMORY	PRINT	TOOL DATA BASE	HELP
INSERT	EDIT	DELETE	FIND	SORT
MULTI CUTTER	ADD INCR	SUB INCR	METRIC/INCH	EXIT

The data entry window visualises the origins that are associated to the axes defined for the selected process.

TOOLS TABLE

The Tools Table provides information about the tools mounted on the spindle, such as tool position, tool offset, etc.

A tools table can define up to 250 tools, which is also the maximum number of tools controlled by the system.

When you open a Tools Table (refer to the "Opening a Table" section in this chapter) the following screen is displayed:

TABLE EDITOR **DATE: 10/01/94** **TIME: 9:35:00**

Measurement unit	:	Mm
Data set	:	MEMORY
Increment	:	0.00000
Selected Table	:	TOOL
Sort selected page	:	1 of 1
Current process	:	1
Current magazine	:	1

Tool code	Pocket	Status	Position	Offset	Record
1	118	OUT	OUT	000002	000001
2	28	READY	MAGAZ	000001	000002
3	0	EXPIRED	OUT	000005	000004
4	5	BROKEN	OUT	000004	000003

LOAD MEMORY	SAVE MEMORY	PRINT	TOOL DATA BASE	HELP
INSERT	EDIT	DELETE	FIND	SORT
MULTI CUTTER	ADD INCR	SUB INCR	METRIC/INCH	EXIT

The selection bar is on the first table record. The table columns display only some of the tool parameters. To alter all the tool parameters, select the tool with the bar and press the **EDIT** softkey. The complete set of tool parameters will be displayed as follows:

TABLE EDITOR **DATE: 10/01/94** **TIME: 9:35:00**

Tool code	:	**2**
Tool pocket	:	**28**
Random class	:	**1**
Life type	:	**0**
Tool status	:	**1**
Tool position	:	**1**
Initial life	:	**0.00000**
Actual life	:	**0.00000**
User variable 1	:	**0.00000**
User variable 2	:	**0.00000**
User variable 3	:	**0.00000**
User variable 4	:	**0.00000**
Offset number	:	**1**
Length 1	:	**18.50000**
Max requalif length 1	:	**15.00000**
Curr requalif length 1	:	**7.50000**
Length 2	:	**33.65000**
Max requalif length 2	:	**10.00000**
Curr requalif length 2	:	**2.35000**
Diameter	:	**18.00000**
Curr requalif diameter	:	**12.00000**
Orientation	:	**0**

Measurement unit	:	**Mm**
Data set	:	**MEMORY**
Increment	:	**0.00000**
Selected Table	:	**TOOL**
Sort selected page	:	**1 of 1**
Current process	:	**1**
Current magazine	:	**1**

Position	Offset	Record
OUT	000002	000001
MAGAZ	000001	000002
OUT	000005	000004
OUT	000004	000003

LOAD MEMORY	SAVE MEMORY	PRINT	TOOL DATA BASE	HELP
INSERT	EDIT	DELETE	FIND	SORT
MULTI CUTTER	ADD INCR	SUB INCR	METRIC/INCH	EXIT

The meaning of these parameters is as follows:

Tool code is a 12 digit without sign that identifies the tool. Some of the digits may be used for identifying the tool family.

Tool pocket is a number between 1 and 255 that identifies the tool position in the magazine. 0 indicates that the tool is not in the magazine. If the magazine option is present, the system performs a series of verifications on this parameter (refer to the documentation of the magazine option). If the magazine is not present no verifications will be made.

Random class is a number that specifies the class of the tool in the magazine, i.e. whether it occupies a *random* or a *non-random* position, and the number of magazine positions occupied by the tool, i.e. 1, 2 or 3. Allowed values are as follows:

0 tool not in the magazine

1 tool occupying 1 *non-random* position

2 tool occupying 2 *non-random* positions (the one defined by the *tool pocket* field and the preceding one)

3 tool occupying 2 *non-random* positions (the one defined by the *tool pocket* field and the subsequent one)

4 tool occupying 3 *non-random* positions (the one defined by the *tool pocket* field plus the subsequent and the preceding ones)

5 tool occupying only one *random* position

6 tool occupying 2 *random* positions (the one defined by the *tool pocket* field and the preceding one)

7 tool occupying 2 *random* positions (the one defined by the *tool pocket* field and the subsequent one)

8 tool occupying 3 *random* positions (the one defined by the *tool pocket* field plus the subsequent and the preceding ones)

For more information about the magazine option refer to the specific documentation.

Life type is a number that specifies the unit of measure applied to the tool life. It can be:

0 tool life disabled

1 tool life measured in meters/feet

2 tool life measured in minutes

3 tool life measured in cycles

Tool status is a number that specifies the current tool status. It can be:

0 tool not ready

1 tool ready

2 damaged tool

3 end of tool life

Tool position is a number that specifies the current tool position. It can be:

0 tool not present

1 tool in the magazine

Initial life is a number with 5.5 format that specifies the initial tool life. It may be expressed in meters, feet, minutes or cycles, depending on the setting of the *Life type* parameter. If the tool life is not controlled, this parameter must be set to 0.

Actual life	is a number with 5.5 format that specifies the current tool life. It may be expressed in meters, feet, minutes or cycles, depending on the setting of the *Life type* parameter. This value is decremented by the Tool Magazine option while the tool is used in a machining cycle. When it is 0 the tool is considered out of life (value 3 of the *Tool status* field).
User variable 1	variable available to the use
User variable 2	variable available to the user.
User variable 3	variable available to the user.
User variable 4	variable available to the user.
Offset number	is the number of the record associated to the tool, which corresponds to a record in the relevant table. It is an integer between 1 and 300. If it is set to 0 the tool is not associated to any offset.
Length 1	is the nominal tool length value.
Max requalif length 1	is the maximum value for the *Curr. requalif. length 1* parameter. It is 0 when this limit does not exist.
Curr. requalif. length 1	is the current requalification value applied to the *Length 1* parameter. The sum of both values is the current tool length.
Length 2	has the same meaning as the *Length 1* parameter but is applied to another axis. It is normally not used in milling machines and machining centers, but may be necessary in lathes, where tool variation affects two axes, typically X and Z.
Max requalif length 2	has the same meaning as *Max requalif length 1* but is applied to the second axis.
Curr. requalif. length 2	has the same meaning as *Curr. requalif length 1* but is applied to the second axis.
Diameter	is the nominal value of the tool diameter . It requires that a tool *Orientation* value is defined.
Curr requalif diameter	is a value applied to the nominal *Diameter*.

Orientation is the orientation of the tool tip with respect to the interpolation plane. It can be a number from 0 to 8 as shown in the figure below:

For milling machines the allowed value is 0. For lathes and grinding machines values range from 1 to 8.

IMPORTANT

If a tool offset is altered in the Tools Table, it will be also automatically updated in the Tool Offsets Table.

TOOL OFFSETS TABLE

Tool offsets permit to vary the tool parameters without modifying the part program or the origins.

Tool offsets may be used in the following cases:

- when the tool coordinates defined before actually mounting the tool on the machine must be altered to suit variations or fixture changes.
- when the tool has worn out or has been sharpened.

The Tool Offsets Table includes all the parameters (length, diameter, etc.) of the tools mounted on the machine. Each Tool Offsets Table has 300 offsets, which can be associated to the tools is various ways.

When you open a Tools Table (refer to the "Opening a Table" section in this chapter), the following screen is displayed:

TABLE EDITOR **DATE: 10/01/94** **TIME: 9:35:00**

Measurement unit	:	Mm
Data set	:	MEMORY
Increment	:	0.00000
Selected Table	:	OFFSET
Sort selected page	:	1 of 30
Current process	:	1
Current magazine	:	1

Record	Curr. Length 1	Curr. Length 2	Curr. Diameter	Orient
000001	26.00000	36.00000	30.00000	000000
000002	18.00000	24.00000	9.00000	000001
000003	18.00000	26.00000	9.00000	000005
000004	0.00000	0.00000	0.00000	000000
000005	0.00000	0.00000	0.00000	000000
000006	0.00000	0.00000	0.00000	000000
000007	0.00000	0.00000	0.00000	000000
000008	0.00000	0.00000	0.00000	000000
000009	0.00000	0.00000	0.00000	000008
000010	0.00000	0.00000	0.00000	000000

LOAD MEMORY	SAVE MEMORY	PRINT	TOOL DATA BASE	HELP
INSERT	EDIT	DELETE	FIND	SORT
MULTI CUTTER	ADD INCR	SUB INCR	METRIC/INCH	EXIT

The selection bar is positioned on the first record. The columns of the table visualise the current lengths (nominal length + current requalification). To alter a tool offset parameter, select it with the bar and press the **EDIT** softkey. All the parameters managed by the table will be displayed.

```
TABLE EDITOR                                          DATE: 10/01/94      TIME: 9:35:00

Length 1                        :  18.50000        Measurement unit     :  Mm
Max requalif length 1           :  15.00000        Data set             :  MEMORY
Curr requalif length 1          :  7.50000         Increment            :  0.00000
Length 2                        :  33.65000        Selected Table       :  OFFSET
Max requalif length 2           :  10.00000        Sort selected page   :  1 of 30
Curr requalif length 2          :  2.35000         Current process      :  1
Diameter                        :  18.00000        Current magazine     :  1
Curr requalif diameter          :  12.00000
Orientation                     :  0               Curr. length 2   Curr  Diameter    Orient

                                                        36.00000        30.00000      00000
                                                        24.00000         9.00000      00001
                                                        26.00000         9.00000      00005
                                                         0.00000         0.00000    0.00000
                                                         0.00000         0.00000    0.00000
                                                         0.00000         0.00000    0.00000
                                                         0.00000         0.00000    0.00000
                                                         0.00000         0.00000    0.00000
                                                         0.00000         0.00000    0.00000
                                                         0.00000         0.00000    0.00000

LOAD MEMORY   SAVE MEMORY   PRINT   TOOL DATA BASE   HELP
INSERT        EDIT          DELETE  FIND             SORT
MULTI CUTTER  ADD INCR      SUB INCR METRIC/INCH     EXIT
```

The meaning of these fields is as follows:

Length 1 is the nominal tool length value.

Max requalif length 1 is the maximum limit for the offset specified by the *Curr. requalif. length 1* parameter. If it is set to 0 no limit exists.

Curr. requalif. length 1 is the current requalification value applied to the nominal *Length 1* parameter. The result of this sum is the current tool length.

Length 2 has the same meaning as *Length 1*, but is applied to another axis. It is not normally used for mills and machining centers but applies to lathes, in which the tool variation affects two axes (typically X and Z).

Max requalif length 2 has the same meaning as *Max requalif length 1* but applies to the second axis.

Curr. requalif. length 2 has the same meaning as *Curr. requalif length 1* but applies to the second axis.

Diameter is the nominal value of the tool diameter. It must be used in association with the tool orientation.

Curr requalif diameter is added to the nominal value specified by the *Diameter* parameter.

Orientation is the orientation of the tool tip with respect to the interpolation plane. It can vary from 0 to 8 as shown in the figure below:

Values from 1 to 8 are used for lathes and grinding machines. For milling machines the value is always 0.

Each tool must have its own tool offset but the same offset may be apllied to different tools.

The parameter that links a given offset to a tool is the *Offset number* in the Tools Table. This parameter specifies the number of the Offsets Table record in which the offset is stored. For example, if 114 is the *Offset number* that corresponds to tool code 145, it means that the length or diameter of tool 145 will vary according to the contents of record 114.

IMPORTANT

If the parameters stored in the Tool Offsets Table are modified, the new offset values will be automatically updated in the records of the Tools Table that are associated to the modified tools.

TOOL DATA BASE

The TOOL DATA BASE stores information about as many as 500 different types of tool families.

The TOOL DATA BASE can be used for storing the parameters that describe the tools mounted on the shop magazine. These parameters can be transfered to the Tools Table when the tools are used in a given machining cycle. For more information about this topic, refer to the 'Tool Magazine' documentation.

When the **TOOL DATA BASE** softkey is pressed, the following screen appears:

TABLE EDITOR DATE: 10/01/94 TIME: 9:35:00

Measurement unit	:	Mm
Data set	:	TOOLDBASE
Increment	:	0.00000
Selected Table	:	DATABASE
Sort selected page	:	1 of 1
Current process	:	1
Current magazine	:	1

Description	Tool code	Length 1	Length 2	Diameter
Tool 01	1	36.00000	54.00000	10.00000
Tool 02	2	24.00000	9.00000	25.00000
Tool 03	3	26.00000	45.00000	32.00000
Tool 04	4	57.00000	25.00000	18.00000
Tool 05	5	32.00000	12.00000	27.00000

LOAD MEMORY	SAVE MEMORY	PRINT	SHAPE	HELP
INSERT	EDIT	DELETE	FIND	SORT
	ADD INCR	SUB INCR	METRIC/INCH	EXIT

The selection bar is positioned on the first record. The columns of the table visualise the most important parameters. To alter a family of tools select it with the bar and press the **EDIT** softkey. To create a new family of tools press the **INSERT** softkey. All the parameters will be displayed in a specific data entry window (refer to the "Modifying a Table" section in this chapter).

```
TABLE EDITOR                                          DATE: 10/01/94      TIME: 9:35:00

Tool code                  :  1                  Measurement unit    :  Mm
Random class               :  1                  Data set            :  TOOLDBASE
Life type                  :  3                  Increment           :  0.00000
Initial life               :  10.00000           Selected Table      :  DATABASE
User variable 1            :  0.00000            Sort selected page  :  1 of 1
User variable 2            :  0.00000            Current process     :  1
User variable 3            :  0.00000            Current magazine    :  1
User variable 4            :  0.00000
Length 1                   :  36.00000           Length 1        Length 2        Diameter
Length 2                   :  54.00000
Diameter                   :  10.00000           36.00000        54.00000        10.00000
Graphic class              :  3                  24.00000         9.00000        25.00000
Description                :  Tool 01            26.00000        45.00000        32.00000
                                                 57.00000        25.00000        18.00000
                                                 32.00000        12.00000        27.00000

LOAD MEMORY   SAVE MEMORY   PRINT   TOOL DATA BASE   HELP
INSERT        EDIT          DELETE  FIND             SORT
MULTI CUTTER  ADD INCR      SUB INCR METRIC/INCH     EXIT
```

The meaning of these data entry fields is as follows:

Tool code is a 12 digit without sign that identifies the tool. Some of the digits may be used for identifying the tool family.

Random class is a number that specifies the class of the tool in the magazine, i.e. whether it occupies a *random* or a *non-random* position, and the number of magazine positions occupied by the tool, i.e. 1, 2 or 3. Allowed values are as follows:

0 tool not in the magazine
1 tool occupying 1 *non-random* position
2 tool occupying 2 *non-random* positions (the one defined by the *tool pocket* field and the preceding one)
3 tool occupying 2 *non-random* positions (the one defined by the *tool pocket* field and the subsequent one)
4 tool occupying 3 *non-random* positions (the one defined by the *tool pocket* field plus the subsequent and the preceding ones)
5 tool occupying only one *random* position
6 tool occupying 2 *random* positions (the one defined by the *tool pocket* field and the preceding one)
7 tool occupying 2 *random* positions (the one defined by the *tool pocket* field and the subsequent one)
8 tool occupying 3 *random* positions (the one defined by the *tool pocket* field plus the subsequent and the preceding ones)

NOTE:
For further details, see the magazine option documentation.

Life type is a number that specifies the unit of measure applied to the tool life. It can be:

0 tool life disabled
1 tool life measured in meters/feet
2 tool life measured in minutes
3 tool life measured in cycles

Initial life is a number with 5.5 format that specifies the initial tool life. It may be expressed in meters, feet, minutes or cycles, depending on the setting of the *Life type* parameter. If the tool life is not controlled, this parameter must be set to 0.

User variable 1 variable available to the user.

User variable 2 variable available to the user.

User variable 3 variable available to the user.

User variable 4 variable available to the user.

Length 1 is the nominal tool length value.

Length 2 has the same meaning as the *Length 1* parameter but is applied to another axis. It is normally not used in milling machines and machining centers, but may be necessary in lathes, where tool variation affects two axes, typically X and Z.

Diameter is the nominal value of the tool diameter and is normally used in milling machines and machining centers rather than in lathes. It requires that a tool *Orientation* value is defined.

Graphic class specifies the tool graphic class for the Graphic Verification option.
The allowed values are:

0 no graphic class
1 Slot/End milling
2 Drilling
3 Spot drilling
4 Boring bar
5 Facing
6 Dovetail
7 Concave milling
8 Probing
9 Convex milling

Description it is a string of max. 16 characters that describes the tool type.

IMPORTANT

When the part program Graphic Verification option is not present, the **SHAPE** softkey is not enabled.

MAGAZINE TABLE

The Magazine Table is similar to the Tools and Tool Data Base tables. Most of its parameters are also common to the other two. For more information about the parameters that are specific of the Magazine Table, refer to the documentation of the Magazine option.

USER TABLE

The User Table is made up of 100 application-specific records, each of which has 4 use variables. When you open a User Table (refer to the "Opening a Table" section in this chapter), the following page appears:

TABLE EDITOR DATE: 10/01/94 TIME: 9:35:00

Measurement unit	:	Mm
Data set	:	MEMORY
Increment	:	0.00000
Selected Table	:	USER
Sort selected page	:	1 of 10
Current process	:	1
Current magazine	:	1

Record	Variable 1	Variable 2	Variable 3	Variable 4
000001	26.00000	89.00000	13.00000	0.00000
000002	14.35500	21.57900	69.00000	0.00000
000003	18.00000	36.00000	95.00000	0.00000
000004	0.00000	0.00000	0.00000	0.00000
000005	0.00000	0.00000	0.00000	0.00000
000006	0.00000	0.00000	0.00000	0.00000
000007	0.00000	0.00000	0.00000	0.00000
000008	0.00000	0.00000	0.00000	0.00000
000009	0.00000	0.00000	0.00000	0.00008
000010	0.00000	0.00000	0.00000	0.00000

LOAD MEMORY	SAVE MEMORY	PRINT	TOOL DATA BASE	HELP
INSERT	EDIT	DELETE	FIND	SORT
MULTI CUTTER	ADD INCR	SUB INCR	METRIC/INCH	EXIT

The selection bar is positioned on the first record, which corresponds to the first set of user variables. To alter the value of a variable, select the record with the bar and press the **EDIT** softkey. Variables are displayed in a dedicated window. For more information about them, refer to the Programming Manual.

NOTE:

The contents of the User Table can be easily customised. While in most tables it is possible to change the softkeys, the help messages and the sequence in which fields are displayed and/or printed, in the User Table it is also possible to reconfigure the table layout, size, variable format and structure, number of pages, etc. For the details, see Chapter 8.

END OF CHAPTER

TABLE EDITOR CONFIGURATOR

GENERAL

The Table Editor described in chapter 7 can be customised to suit various application requirements. Although personalisation is typically carried out during system installation, it can be altered at any one moment.

Alterations to the Table Editor must be carried out by qualified personnel that is well acquainted with both the system features and the impact of these changes upon the system.

This chapter describes the procedures for configuring and customising the video screens, the softkeys and the data entry windows managed by the Table Editor.

The Table Editor configurator permits to customise the following tables:

- Tools table
- Tool Offsets table
- User table
- Origins table
- Magazines table
- Tool Data Base table

In addition, the configurator permits to set up a series of axis parameters for the version released with offline applications (execution on PC).

CONFIGURATOR SCREENS

The Table Editor configurator has two video pages: a main screen and a configuration screen. Each of them has its own set of softkeys.

The main screen consists of a set of softkeys, whereas the configuration screen varies with the type of table to be edited or the function to be performed.

Main configurator screen

To access the configuration environment it is necessary to press the **TABLE** softkey from the Table Editor main menu.

The following menu will be displayed:

BACKUP MEMORY	RESTORE MEMORY	MAGAZINE	SETUP	HELP
TOOL	OFFSET	USER	ORIGIN	TOOL DATA BASE
SELECT PROCESS	SELECT MACHINE	SET INCR		EXIT

Press the **SETUP** softkey to display the configurator main screen:

In the sections that follow it has been assumed that the system is displaying the above screen.

Configurator screen

To enable any configuration environment press the relevant softkey: **TOOL CFG**, **OFFSET CFG**, **USER CFG**, **ORIGIN CFG**, **DATA BASE CFG** or **MAGAZINE CFG**. The system will display the Configurator screen:

The window displayed in the data area of the screen varies according to the selected table. The above table, for example, is a User Table. Data entry windows in the Table Configurator are similar to those displayed in the Table Editor environment but display different parameters. For example, the User Table in the Table Editor displays only a column of variables (*Variables 1 through 4*) with the relevant numerical parameter, whereas the User Table in the Table Configurator displays the PLUS name of the variable (first column) and the format (D for double) but not the value.

The sections that follow show each of the tables as they appear in the Table Configurator environment. To see how the same table is displayed in the Table Editor environment refer to Chapter 9 of this manual.

Tools Table

TCODE	Tool code	(T)
TOOLPOS	Tool pocket	(S)
TOOLTYPE	Random class	(S)
TOOLTYPE	Life type	(S)
TSTATUS	Tool status	(S)
POCKET	Tool position	(S)
MAXLIFE	Initial life	(D)
REMLIFE	Actual life	(D)
TUSER1	User variable1	(D)
TUSER2	User variable2	(D)
TUSER3	User variable3	(D)
TUSER4	User variable4	(D)
TOLOFNR	Offset number	(S)
TCNTRL	Free	(S)
TACTL1	Length 1	(D)
TCMAXL1	Max requalif length 1	(D)
TCACTL1	Curr requalif length 1	(D)
TACTL2	Length 2	(D)
TCMAXL2	Max requalif length 2	(D)
TCACTL2	Curr requalif length 2	(D)
TDIAMET	Diameter	(D)
TCACDIA	Curr requalif diameter	(D)
TORIENT	Orientation	(S)

Tool Offsets Table

TACTL1	Length 1	(D)
TCMAXL1	Max requalif length 1	(D)
TCACTL1	Curr requalif length 1	(D)
TACTL2	Length 2	(D)
TCMAXL2	Max requalif length 2	(D)
TCACTL2	Curr requalif length 2	(D)
TDIAMET	Diameter	(D)
TCACDIA	Curr requalif diameter	(D)
TORIENT	Orientation	(S)

Origins Table

Axis	(C)	Origin value	(D)
Axis	(C)	Origin value	(D)
Axis	(C)	Origin value	(D)
Axis	(C)	Origin value	(D)
Axis	(C)	Origin value	(D)
Axis	(C)	Origin value	(D)
Axis	(C)	Origin value	(D)
Axis	(C)	Origin value	(D)
Axis	(C)	Origin value	(D)

Magazine Table

MAG_TYP	Magazine type	(S)
NUM PKT	Pocket number	(S)
ROW_PKT	Pocket per row	(S)
FIR_PKT	Index of first pocket	(S)
SHORT1	User short 1	(S)
SHORT2	User short 2	(S)
SHORT3	User short 3	(S)
SHORT4	User short 4	(S)
SHORT5	User short 5	(S)
SHORT6	User short 6	(S)
SHORT7	User short 7	(S)
SHORT8	User short 8	(S)
DOUBLE1	User double 1	(D)
DOUBLE2	User double 2	(D)
DOUBLE3	User double 3	(D)
DOUBLE4	User double 4	(D)
DOUBLE5	User double 5	(D)
DOUBLE6	User double 6	(D)
DOUBLE7	User double 7	(D)
DOUBLE8	User double 8	(D)

Tools Data Base Table

TCODE	**Tool code**	**(T)**
TFAMCOL	**Random class**	**(S)**
TOOLTYPE	**Life type**	**(S)**
MAXLIFE	**Initial life**	**(D)**
TUSER1	**User variable1**	**(D)**
TUSER2	**User variable2**	**(D)**
TUSER3	**User variable3**	**(D)**
TUSER4	**User variable4**	**(D)**
TACTL1	**Length 1**	**(D)**
TACTL2	**Length 2**	**(D)**
TDIAMET	**Diameter**	**(D)**
GRAPH	**Graphic class**	**(S)**
DESCR	**Description**	**(C)**

As we have explained before, most of the paremeters displayed in these data entry windows are the same as in the relevant Table Editor windows. Additional information provided by the configurator tables includes the name by which a variable or field is known in the PLUS environment (most left column) and the variable format.

Allowed variable formats are as follows:

(C) character variable (one or several characters)

(S) short number (1 word = 2 byte)

(D) double number (4 word = 8 byte)

(T) tool code, which is made up of a series of alphanumeric characters.

SOFTKEYS

Softkeys in the main configurator screen

LOAD	SAVE	MAGAZINE CFG		HELP
TOOL CFG	OFFSET CFG	USER CFG	ORIGIN CFG	DATA BASE CFG
RESET CONFIG	AXIS CFG	TOOL 2CFG		EXIT

The meaning of these softkeys is as follows:

SOFTKEY	FUNCTION
LOAD	Permits to load the configuration file. Opens a data entry window for writing data associated to this file.
SAVE	Permits to save a configuration file. Opens a data entry window for writing data associated to this file.
MAGAZINE CFG	Permits to open a submenu with the softkeys that configure the Magazine Table.
HELP	Visualises an on-line help page related to the active data entry window or to the general menu of the active video page.
TOOL CFG	Visualises a video screen with the softkeys for parameter setting and the data entry window with the current Tools Table configuration.
OFFSET CFG	Visualises a video screen with the softkeys for parameter setting and the data entry window with the current Tool Offsets Table configuration.
USER CFG	Visualises a video screen with the softkeys for parameter setting and the data entry window with the current User Table configuration.
ORIGIN CFG	Visualises a video screen with the softkeys for parameter setting and the data entry window with the current Origins Table configuration.
DATA BASE CFG	Visualises a video screen with the softkeys for parameter setting and the data entry window with the current Tools Data Base Table configuration.
AXIS CFG	Permits to set up axes parameters for the version released with the OFFLINE application that can be executed on a PC. In 10 Series systems the configuration is enabled with the data read by the control and this softkey is disabled. When you press this softkey on the PC the screen visualises a data entry window for writing the relevant data.
TOOL 2CFG	Used for modifying the additional tool table.
RESET CONFIG	Restores the default configuration. Opens a data entry window for confirming the RESET function.
EXIT	Permits to exit from the Configurator environment. A data entry window is displayed for the operator to confirm whether or not the altered configuration must be saved.

Configuration screen softkeys single tables

PRINT CFG	DB UNLINK	DB LINK		HELP
REMOVE	FIELD CFG	HELP CFG	SOFT KEY CFG	INSERT
SORT CFG	SORT EDIT			EXIT

The meaning of the softkeys in this menu is as follows:

SOFTKEY	FUNCTION
PRINT CFG	Permits to configure the printing grid and position the printing fields. Displays all the fields included in the current table highlighting those that are printable at that moment. Also displays the current printer configuration.
DB UNLINK	This softkey is active only when the Tools Table is linked to the Tools Data Base Table. It permits to unlink a Tools Table field from the Tools Data Base Table. Before pressing the **DB UNLINK** softkey position the cursor on the desired Tools Table field. As you press **DB UNLINK** the system visualises a Data Base Table in which the ACTUAL LINK section specifies the linked field. To confirm the selection press **[Enter]** or re-press **DB UNLINK**.
DB LINK	This softkey is active only when the Tools Table is linked to the Tools Data Base Table. It permits to link a Tools Table field to a field in the Tools Data Base Table. Before pressing the **DB LINK** softkey position the cursor on the desired Tools Table field. Then press **DB LINK** and move the cursor to the desired Data Base Table field. To confirm the link press **[Enter]** or re-press **DB LINK**.
HELP	Displays an on-line help page associated to the current data entry window, menu or video page. While the help page is active, printing configuration and sorting are not allowed, and no data can be written in the data entry windows.
REMOVE	Permits to remove a field from the printing and sorting layout. Position the cursor on the field with the arrow keys and press **[Return]** or **[Enter]** to remove it.
FIELD CFG	Permits to configure the fields of the selected table. Before pressing **FIELD CFG** position the cursor on the desired field. This opens a data entry window that lists all the configurable field parameters.
HELP CFG	Permits to define the help messages associated to the selected field. Before pressing **HELP CFG** position the cursor on the desired field. This opens a data entry window in which the message can be written.
SOFT KEY CFG	Permits to alter the label of the softkey associated to the selected table. If the label field is left empty, the relevant softkey will not appear in the Table Editor menu. When you press **SOFT KEY CFG** the system opens a data entry window for writing the label name.

SOFTKEY	FUNCTION
INSERT	Permits to insert a new field or move an existing field in the printing configuration or the table layout. By moving the cursor with the arrow keys it is possible to visualise the position of each field in the layout. Press **[Enter]** or **[Return]** to confirm the field to be inserted, which will be highlighted in yellow in the menu window. Use the arrow keys to move the field to the desired position and press **[Enter]** or **[Return]** to confirm it. The inserted field will be highlighted in white.
SORT CFG	Permits to define the sorting number (max 6) and the relevant label. When this softkey is pressed the screen displays a data entry window for writing the parameters that will be configured with the **SORT EDIT** softkey. The maximum sorting number for all of the tables is 15.
SORT EDIT	Permits to configure the parameters that have been previously selected with **SORT CFG**. All the fields handled by the selected table are displayed in the screen; the sorted fields and the column headers will be highlighted in white. Before pressing this softkey, press **SORT CFG** and select the desired sorting number.
EXIT	Closes the configurator screen and returns to the main configurator menu.

LOADING A CONFIGURATION FILE

The configuration of a table is normally stored in a file with the .CNF extension. Although the system may include several types of configuration, only one of them is active at any one time. The remaining configurations may be loaded with the following procedure:

Press the **LOAD** softkey in the main menu. The system visualises two windows: one lists the configuration files stored in the system memory; the other can be used for writing the name of the desired configuration file.

Name	Size	Date	Time
TAB_ONE	47026	04/05/1992	10:45:00
TAB_TWO	47026	12/10/1993	15:30:00
TAB_THR	47026	20/10/1993	14:15:00

LOAD CONFIGURATION

Enter filename: **TAB_ONE**

The meaning of these fields is as follows:

Name is the name of the configuration file.

Size is the size of the configuration file. It is a fixed value (47026 bytes).

Date is the date of creation of the configuration file.

Time is the time when the configuration is saved.

To load a configuration stored in the system hard disk, select the file name with the **[Up/Down Arrows]** on the most left window or key in the file name in the *Enter filename:* field of the LOAD CONFIGURATION data entry window.

Press **[Enter]** or the **LOAD** softkey to load the file or **[Esc]** to abort the operation. In the first case the system will prompt the operator for confirmation.

SAVING A CONFIGURATION

To save an altered or a new configuration press the **SAVE** softkey in the main menu.

When you press **SAVE** the system opens a data entry window that lists the configuration file stored in the system:

Name	Size	Date	Time
TAB_ONE	47026	04/05/1992	10:45:00
TAB_TWO	47026	12/10/1993	15:30:00
TAB_THR	47026	20/10/1993	14:15:00

SAVE CONFIGURATION

Enter filename: **TAB_ONE**

To save a configuration stored in the system hard disk, select the file name with the **[Up/Down Arrow keys]** on the most left window or key in the file name in the *Enter filename:* field of the SAVE CONFIGURATION data entry window.

Press **[Enter]** or the **SAVE** softkey to save the file or **[Esc]** to abort the operation. In the first case the system will prompt the operator for confirmation.

RESTORING THE DEFAULT CONFIGURATION

The **RESET CONFIG** permits to restore the default Table Editor configuration. When you press **RESET CONFIG** on the main menu the following data entry window appears:

RESET CONFIGURATION

Load default config. (Y/N) :

Key in **[Y]** and press **[Enter]** or **RESET CONFIG** to restore the default Table Editor.

To abort the procedure Key in **[N]** or press **[Esc]**.

IMPORTANT

In order not to lose the current configuration, make sure you save it before restoring the default one.

AXES CONFIGURATION

If the Table Editor is installed in a PC under an offline application and not on the 10 Series system, certain axes and process parameters can be configured on the PC.

On the 10 Series Configurator parameters are loaded directly by the system and the **AXIS CFG** softkey is not available.

To configure the axes press **AXIS CFG** from the main menu. The following data entry window will appear:

```
                AXIS CONFIGURATION

Process  :  x                    Type  :  x

Axis/Type

  x:x        x:x        x:x        :          :
   :          :          :         :
```

The meaning of these field is as follows:

Process: This field is used for writing the process name. The system can support as many as twenty-four simultaneous processes. To configure them you can browse through the relevant data entry windows with **[PgUp]** and **[PgDn]**.

Type: Identifies the type of process as follows:

0 undefined

1 milling

2 turning

3 grinding.

Axis/Type: These columns make it possible to define the axis name and type. Allowed values are as follows:

Axis (character to the left):

The axis name, as configured in the system.

Type (character to the right):

N Normal

R Rotary

S Spindle

CONFIGURING A TABLE

The procedure for table configuration is similar for all the tables managed by the system except for the Tools Table, in which the **TOOL CFG** softkey opens the following data entry window:

TABLE MANAGEMENT

Database access	**(Y/N)**	**:**
Offset access	**(Y/N)**	**:**
Magazine access	**(Y/N)**	**:**
Multicut management	**(Y/N)**	**:**
PLUS compatibility	**(Y/N)**	**:**

This window prompts the operator to define which tables or functions are to be accessed. If the answer is **[N]** the relevant softkey will be disabled from the Table Editor. For example, if you write an **[N]** in the *Magazine access* field, the Tool Magazine will not be linked to the Tools Table and the **SELECT MAGAZINE** softkey will be inactive (black).

The meaning of the fields displayed in this table is as follows:

Database access: From the Tools Table it is possible to access the Tools Data Base Table for loading preconfigured tool types.

Offset access: Permits to access the Tool Offsets Table.

Magazine access: Links the Tools Table to the Magazine Table.

Multicut management: Selects multicutter tool management.

PLUS compatibility: Defines acccess to fields in standard PLUS mode.

PLUS compatibility, Magazine access and *Multicut management* affect the position of the fields and the relevant PLUS addresses in the internal Tools Table, as shown in the figure below:

		Field bits				
N°	**Form**	**63 48**	**47 32**	**31 16**	**15 0**	**Symbolic Name**
1	(T)	*Tool code*				TCODE
2	(S)				*Magaz/pocket*	TOOLPOS
3	(S)				*Random class*	TFAMCOL
4	(S)				*Life type/rand.*	TOOLTYPE
5	(S)				*Tool status*	TSTATUS
6	(S)				*Tool position*	TCNTRL
7	(D)	*Initial life*				MAXLIFE
8	(D)	*Actual life*				REMLIFE
9	(D)	*User variable 1*				TUSER1
10	(D)	*User variable 2*				TUSER2
11	(D)	*User variable 3*				TUSER3
12	(D)	*User variable 4*				TUSER4
13	(S)				*Offset number*	TOLOFNR

In particular, the fields affected by these parameters are 2 (POCKET), 3 (TFAMCOL), 4 (TCLASS) and 6 (TCNTRL). In standard mode, PLUS accesses these fields as illustrated in the figure below, which shows each bit in the field:

	15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**	
2	***Tool pos.***				***Magazine N°***						***Pocket N°***						TOOLPOS
3																	TFAMCOL
4								***Life type***					***Random class***				TOOLTYPE
6	*Not used*																TCNTRL

If you write *PLUS compatibility* **[Y]** the fields will be addressed as indicated in the table regardless of the settings for *Magazine access* and *Multicut management*. The same addressing mode can be obtained with *PLUS compatibility* **[N]** and *Magazine access* **[Y]**.

If *PLUS compatibility* is **[N]** and *Magazine access* is **[N]**, the access mode depends on how *Multicut management* is set:

Multicut management: **[Y]**

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
2	***Pocket***																TOOLPOS
3																	TFAMCOL
4								***Life type***					***Random class***				TOOLTYPE
6	***Tool position***																TCNTRL

Multicut management: **[N]**

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
2	***Pocket***																TOOLPOS
3	***Random class***																TFAMCOL
4	***Life type***																TOOLTYPE
6	***Tool position***																TCNTRL

Key in **[Y]** or **[N]** in the Table Management data entry window and press **TOOL CFG** or **[Enter]** for confirmation.

As for the remaining tables, the configuration softkey permits direct access to the above described configuration screen. In this screen it is possible to alter the description of a variable by simply positioning the cursor on the desired variable with the arrow keys, deleting the existing text and writing the new description.

IMPORTANT

TABLE MANAGEMENT assignments are significant only if they are made during the initial configuration stage.

If previously configured table files are used, and the TABLE MANAGEMENT fields are subsequently assigned values that are different from the original ones, visualised or printed table data may acquire a different meaning. For example, if *Magazine access* is set to **[N]** and tables for various magazines are prepared, the tools in those magazines will be concentrated under a single item, i.e. the notion of magazine will be lost.

Configuring fields

Each table field may be personalised with a series of parameters that vary according to the field type (double or other). These parameters are listed in two different data entry windows, one for each field type, that are displayed when **FIELD CFG** is pressed.

The procedure is as follows:

1. With the main video screen on display press the desired table configuration softkey: **TOOL CFG**, **OFFSET CFG**, **USER CFG**, **MAGAZINE CFG**, **ORIGIN CFG** or **DATABASE CFG**. If **TOOL CFG** is selected, press it again to confirm the links.
2. Press the **FIELD CFG** softkey. Depending on the type of field, one of the following data entry windows will be displayed:

   Double fields

   Double fields are marked with a *(D)* in the data entry window:

```
Enable       (Y/N) :
Input field  (Y/N) :
Metric/Inch  (Y/N) :
Set Incr     (Y/N) :
Conversion   (Y/N) :
Signed       (Y/N) :
Integers           :
Decimals           :
Min Value          :
Max Value          :
```

Other field types

```
Enable       (Y/N) :
Input field  (Y/N) :
Conversion   (Y/N) :
Signed       (Y/N) :
Length             :
Min Value          :
Max Value          :
```

The meaning of the fields is as follows:

Enable (Y/N): Enables or disables the field.

Input field (Y/N: Enables/disables field modification.

Metric/Inch (Y/N): Enables or disables conversion between millimeters and inches.

With **[Y]** the value written in the table field will be converted from one unit of measure into the other each time the **METRIC/INCH** softkey is pressed. With **[N]** the displayed value will be expressed in the configured unit of measure and will not be converted into the alternative unit by pressing **METRIC/INCH**.

Set incr (Y/N): Permits to enable or disable the field increment/decrement function.

With **[Y]** the value written in the table field will be incremented or decremented by a given amount each time the **ADD INCR** or **SUB INCR** softkeys are pressed. With **[N]** these two softkeys do not affect the value written in the field.

Conversion (Y/N): The meaning of this field depends on the format. In addition, it is significant only for the fields listed in the tables below:

Short fields:

Defines whether mnemonic or numerical information will be displayed or printed.

Enter **[Y]** to display/print mnemonic information and **[N]** to display/print a numerical value.

For example, if the *Random* field has been set to 1, it may be displayed/ printed mnemonically as FIXED or numerically as 1.

The table that follows lists the mnemonic information that will be displayed/printed in each parameter according to the numerical value. For more information about this topic refer to the Tool Management Option Manual.

Numerical setting	***Random class***	***Life type***	***Tool status***	***Tool position***	***Magazine Type***
0	NO USE	NO LIFE	OUT	OUT	NOMAGA
1	FIXED	METERS	READY	MAGAZ	LINEAR
2	FIX-	MINUTE	BROKEN		ROT CW
3	FIX+	CYCLES	EXPIRD		ROT CCW
4	FIX3				ROTAT
5	RANDOM				PLAN
6	RND-				
7	RND+				
8	RND3				

Double fields:

Defines the type of displayed or printed value for tool lengths and diameters, as shown in the table below:

Conversion type	Length 1	Length 2	Diameter
Y	Length 1 + Curr. Requalif. Length 1	Length 2 + Curr. Requalif. Length 2	Diameter + Curr. Requalif. Diameter
N	Length 1	Length 2	Diameter

Signed [Y/N]: Specifies whether or not the field must include the sign.

Integers: Defines the number of integer digits to be written in the field value.

Decimals: Defines the number of decimal digits to be written in the field value.

IMPORTANT

If the number of decimals is less than the configured value, the remaining decimals can be used for accomodating integers as long as the total number of digits remains constant.

For example, if the configured format is 5.5, the following formats are also legal: 6.4, 7.3, 8.2, 9.1 or 10.0. On the contrary, formats like 4.6, etc. are illegal.

IMPORTANT

Modifications of *Integers* and *Decimals* parameters can generate interruption anomalies on data present into the system or on part program data already set for different values.

For example, if the system has been set for 5 integers and 5 decimals, and only 3 decimals are configured in this environment, the exceeding decimals are lost and the system accuracy can be affected. In the opposite case no problems occur.

Length: Defines the field length in number of digits. It does not apply to double fields.

Min Value: Specifies the minimal value that can be written in a field without generating an error. This value must conform to the format defined with the *Integers* and *Decimals* fields.

Max Value: Specifies the maximum value that can be written in a field without generating an error. This value must conform to the format defined with the *Integers* and *Decimals* fields.

LINKING TABLE FIELDS

As we have seen in a previous section, it is possible to import data from the Tools Data Base Table into the Tools Table. However, not all of the Tools Table fields are linked to the Tools Data Base field bearing the same name (refer to the "Configuring a Table" section).

The procedure that follows describes how to select the fields that can be imported to the Tools Table. Linked fields are marked by a >> to the right of the format (C, S, D, T). For further information on how to import data refer to Chapter 9, "Tool Management".

Field links can be established only between the Tools and Tools Data Base Tables. To enable a link, enter **[Y]** in the *Database access* field of the *TABLE MANAGEMENT* data entry window. Then follow these steps:

1. Press the **TOOL CFG** soktkey.
2. Check that the *Database access* field in the data entry window displays **[Y]** (default value) and re-press **TOOL CFG** .
3. Position the cursor to the desired field and press the **DB LINK** softkey. The data area of the screen will display a second window listing all the Data Base fields. The *ACTUAL LINK* area will show whether the link exists or not, in which case *"none"* will appear. The Table Editor Configurator is as follows:

4. With the arrow keys select the Tools Data Base fields to be linked to the Tools Table and press **[Enter]** or **DB LINK** to establish the link.

NOTE:
Linkable fields must have the same format: short (S), double (D), etc. Unless *User variable* fields are used, it is recommended to establish a link between fields bearing the same name.

UNLINKING TABLE FIELDS

The procedure for unlinking table fields is as follows:

1. Press the **TOOL CFG** twice to display the Tools Table screen.
2. Select the fields to be unlinked from the window that lists all the fields in the Tools Data Base Table. Linked fields are marked with a ">>" to the right of the field format. In addition, when a field is selected, the name of the linked field or "*none*" are displayed in the ACTUAL LINK window to indicate whether or not a link exists.

4. Press the **DB UNLINK** softkey or **[Enter]** to remove the link.

ARRANGING THE TABLE FIELDS

The layout with which table fields appear on the screen may be arranged to suit specific user needs. Sorting criteria can be established in the system configuration and subsequently altered or replaced by others. It is possible to define up to 6 criteria for a given table and up to 15 criteria for the whole set of tables.

The type of data to be displayed on a table must be selected directly in the Table Editor (refer to Chapter 7). The Configurator permits to establish the type of parameter that appears in each column and the sequence in which columns are displayed.

To modify or define a display sequence it is necessary to use the **SORT CFG** and **SORT EDIT** softkeys in the following procedure:

1. Select the table to be modified by pressing the relevant configuration softkey.
2. Press the **SORT CFG** softkey. The following data entry window will be displayed:

SORT CONFIGURATION

Sort 1 : Tool & status criteria
Sort 2 : Pocket & tool criteria
Sort 3 : Tool & lifetype criteria
Sort 4 : Tool & user variable
Sort 5 :
Sort 6 :

(This data entry window refers to the Tools Table).

4. To modify the items in a sorting criteria, select it with the arrow keys and edit the text. To define a new criteria move the cursor to one of the free fields and key in the new text. To delete a sorting criteria select it with the arrow keys and cancel the text.
5. To modify or add sorting criteria press the **SORT EDIT** softkey. The following screen will be displayed:

The upper window lists all the fields available in the selected table highlighting the active ones.

The lower window is divided into two lines: the upper line displays the field headers and the lower line displays the current field settings.

Table Editor Configurator **DATE: 00/00/00** **TIME 15:20:00**

Tool code	User variable 2	Max requalif length 2
Tool pocket	User variable 3	Curr requalif length 2
Random class	User variable 4	Diameter
Life type	Offset number	Curr requalif diameter
Tool status	Free	Orientation
Tool position	Length 1	Record number
Initial life	Max requalif length 1	
Actual life	Curr requalif length 1	
User variable 1	Length 2	

MODIFY

Tool code	Poket	Status	Position	Offset	Record
xxxxxxxx	xxxx	±xxxxxx	±xxxxxx	xxxxxx	xxxxxx

PRINT CFG	DB UNLINK	DB LINK		HELP
REMOVE	FIELD CFG	HELP CFG	SOFT KEY CFG	INSERT
SORT CFG	SORT EDIT			EXIT

To alter the name of a table header position the cursor on the relevant field. The alteration may be made in insert (*INSERT*) or overwrite (*MODIFY)* mode. To switch between *INSERT* and *MODIFY* press **[Ins.]** To delete a text press **[Delete]** or **[Backspace]**.

IMPORTANT

The following ASCII characters cannot be written in header names:

- Characters from ASCII decimal 0 to 31 (or between hexadecimal 0 and 1F)
- * (42 decimal, 2A hexadecimal)
- ; (59 decimal, 3B hexadecimal)
- @ (64 decimal, 40 hexadecimal)
- # (35 decimal, 23 hexadecimal).

IMPORTANT

Since there is no predefined link between headers and data, the user must check that parameters are displayed in the table under the correct header.

16To delete a field in the upper window use this procedure:

1. Press the **REMOVE** softkey.
2. Select the field from the upper window with the arrow keys. As you move the cursor through the list the relevant format will be highlighted in the data area.
3. Press **[Enter]** or **[Return]**.
4. To remove other fields repeat steps 2 and 3. To close the session press **REMOVE** again.

To insert values under a header use this procedure:

1. Press the**[Ins]** softkey.
2. Select the field from the upper window with the arrow keys. As you move the cursor through the list the relevant format will be highlighted in the current position. Press **[Enter]**. The field format (for example, ±*XXXX.XXXXX*) will be displayed in yellow in the data area.
3. Use the arrow keys to position the selected field under the relevant header and press **[Enter]**.
4. To add other fields (up to 6 fields per criteria), repeat steps 2 and 3. To close the session press **INSERT** again.

NOTES:
Each sorting criteria must include at least two fields. The table will be arranged starting from the most left field in the data area.

To substitute a field for another it is necessary to first delete the existing field with the **REMOVE** procedure and then add the new one with the **INSERT** procedure.

After making all the necessary alterations press **SORT EDIT** again to return to the screen that originally displayed with the **SORT CFG** softkey. To exit from this screen press the **SORT CFG** softkey again or press **[Enter]**.

CONFIGURING THE PRINTING OUTPUT

The procedure for configuring the printing output is similar to the one described in the previous section.

When you press **PRINT CFG** the following screen appears:

The upper window lists the fields available with the selected table and highlights the printable ones.

The lower window is split in two areas separated by the *HEADER/BODY* line: the header area and the data area. This window lists the current printing configuration, with the headers and the parameters that will be printed on each page.

The header area is made up of four lines. It displays the headers printed on top of each page, which identify the parameters printed on the page body.

To alter the header of a column position the cursor on the header and select the writing mode, *INSERT* or *MODIFY* (overwrite). To toggle between both modes press **[Ins]**. To cancel a portion of text press **[Delete]** or **[Backspace]**.

The data area of the lower window displays the values of the field selected and positioned by the user.

IMPORTANT

The following ASCII characters are not allowed in header names:

- haracters from ASCII decimal 0 to 31 (or between hexadecimal 0 and 1F)
- * (42 decimal, 2A hexadecimal)
- ; (59 decimal, 3B hexadecimal)
- @ (64 decimal, 40 hexadecimal)
- # (35 decimal, 23 hexadecimal).

IMPORTANT

Since there is no predefined link between headers and data, the user must check that parameters are printed in the table under the correct header.

To move the cursor between the *HEADER* and *BODY* areas use **[PgDn]** and **[PgUp].**

To delete a field in the data area use this procedure:

1. Press the **REMOVE** softkey.
2. Select the field from the upper window with the arrow keys. As you move the cursor through the list the relevant format will be highlighted in the data area.
3. Press **[Enter]** or **[Return]**.
4. To remove other fields repeat steps 2 and 3. To close the session press **REMOVE** again.

To insert values under a header use this procedure:

1. Press the **INSERT** softkey.
2. Select the field from the upper window with the arrow keys. As you move the cursor through the list the relevant format will be highlighted in the current position. Press **[Enter]**. The field format (for example, *±XXXX.XXXXX*) will be displayed in yellow in the data area.
3. Use the arrow keys to position the selected field under the relevant header and press **[Enter]**.
4. To add other fields (up to 6 fields per criteria), repeat steps 2 and 3. To close the session press **[INSERT]** again.

To substitute a field for another it is necessary to first delete the existing field with the **REMOVE** procedure and then add the new one with the **INSERT** procedure.

After making all the necessary alterations press **PRINT CFG** to return to the configuration screen.

CONFIGURING HELP

The Table Configurator permits to customise the help messages associated to each table field. The customised message can be added to the standard field description or be overwritten and replace it altogether.

To configure a help message follow these steps:

1. Press the configuration softkey to open the data entry window of the desired table, move the cursor to the field whose help must be edited and press **HELP CFG**. The following data entry window will appear:

2. Key in the new help message and press **HELP CFG** again to return to the table configuration screen.

CONFIGURING SOFTKEYS

With the Table Configurator it is possible to alter the labels of the softkeys displayed with the Editor tables. For example, to rename the *TOOL* softkey from the Tools Table follow these steps:

1. With the main configuration screen displayed press the **TOOL CFG** softkey twice. The tools data entry window will be displayed.
2. Press **SOFT KEY CFG**. The following data entry window will be displayed:

EXTRA CONFIGURATION

Softkey label : TOOL
Record number : 250
Initial position : 1
Avaible memory. : Y
File extension : TOL

3. The cursor is positioned on the first character. The new label may be overwritten on the old one.
4. After making all the necessary alterations, press **[Enter]** or re-press **SOFT KEY CFG** to confirm them and return to the configurator screen.

QUITTING THE CONFIGURATOR ENVIRONMENT

After completing or altering the configuration, press the **EXIT** softkey. The following data entry window will be displayed:

EXIT

Confirm new setup (Y/N)

Press **[Y]** to confirm the alterations. The system will display a series of messages that show the update in progress. When the update is completed the system returns to the initial Configurator screen.

Press **[N]** to return to the initial Configurator screen without saving the new configuration.

PERSONALISING USER TABLES

User Tables are general purpose tables that can display and handle miscellaneous information. Unlike other system tables, which contain parameters that may be configured within rigid ranges, User Tables provide more general information. Moreover, they can be arranged in up to 8 different tables, all of which are stored in the Dual Port memory and can be handled simultaneously.

Standard with 10 Series controls is a dedicated utility that permits to handle User Tables with higher customisation levels than the ordinary Table Editor.

To access this environment press the **USER TABLE** softkey in the Table Editor.

User Table Manager screens and softkeys

If user tables have already been configured, when you press **USER TABLE** the first user table will be displayed. Otherwise, the video data area will be void. The layout of the first user table is shown in the figure below:

This screen consists of 14 fields and an area dedicated to context-sensitive help. The screen layout permits to display the Help message associated to a field without covering the remaining fields. To display the Help message for a given field select the field and press the **HELP** softkey.

The meaning of the softkeys available in this environment is as follows:

SOFTKEY	FUNCTION
SETUP	Enables another screen with a submenu of softkeys that permit to configure the tables, fields, softkeys, and help messages described later in this section.
LOAD MEMORY	Permits to load the contents of a User Table from a file. It operates like the equivalent softkey described in the general User Table environment. **NOTE:** With this softkey it is possible to load the contents of the active table and not those of all the tables.
SAVE MEMORY	Permits to save the contents of a User Table in a file. It operates like the equivalent softkey described in the general User Table environment. **NOTE:** With this softkey it is possible to save the contents of the active table and not those of all the tables..
PRINT	Permits to print the contents of a User Table from a file. It operates like the equivalent softkey described in the general User Table environment.
HELP	Permits to open the context-sensitive help window associated to the selected field, data entry window or softkey.
TABLE X SFK	These eight softkeys can be customised. They permit to open the User Tables (max. 8) in which user data can be organised. When you press a softkey other than the one associated to the currently displayed table, the system closes the table confirming all the parameter modifications and opens the table associated to the pressed softkey.
EXIT	Closes the current table confirming all the modifications and returns to the video that occupied the screen before **USER TABLE** was pressed.

If no user table has been configured, when you press **USER TABLE** the system will display an empty screen and only **SETUP** and **EXIT** will be active.

The following keys are active in the User Table environment:

The **[UP/Down arrows] [Left/Right Arrows]** keys permit to move the cursor inside an field or from field to field.

The **[Return]** key permits to move the cursor from one field to the next inside the displayed table.

The **[Enter]** permits to confirm the modifications made to the current data enry window. If a table is made of several pages, **[Enter]** permits to move from one page to the next. An alternative method for displaying the next page is re-pressing the softkey associated to the table.

[PgUp] and **[PgDn]** permits to move from page to page within the current table. Since a screen can display as many as 14 fields, the table data will be subdivided and arranged in various pages.

When you press the **SETUP** softkey, a screen similar to the one on the following page appears.

This page is a new softkey menu and lists the eight possible user tables configurable with this utility. In fact, whereas in the normal Table Editor (see Chapter 7), the 400 user variables were located in a single table divided into 4 groups with 100 records, with this configurator they may be divided into a maximum of eight tables, access to which is provided by the softkeys seen above.

Significance of the columns of the table on this page is illustrated in the drawing following the figure. Further details on the information contained in this list will provided later.

The meaning of the softkeys displayed in this screen is as follows:

SOFTKEY	FUNCTION
LOAD	Permits to load a User Table configuration. It operates like the equivalent softkey available with the Table Editor.
SAVE	Permits to save a User Table configuration. It operates like the equivalent softkey available with the Table Editor.
EDIT	Permits to edit the table selected from the visualised directory. It opens a data entry window for the operator to key in the softkey label and the indexes of the first and last variable in that table.
DELETE	Deletes the selected table from the configuration. When you press this softkey a request for confirmation is displayed.
TABLE CONFIG	Permits to configure the fields of the selected table by modifying the default values. When you press this softkey the system displays a video screen (similar to the one displayed by the **USER TABLE** softkey and reproduced earlier in this section) whose fields can be selected and edited. In this environment the **FIELD CONFIG** and **HELP CONFIG** softkeys are active.
FIELD CONFIG	This softkey is active only in conjunction with the data entry window displayed with the **TABLE CONFIG** softkey. Before pressing **FIELD CONFIG** it is necessary to select the field to be configured. In turn, **FIELD CONFIG** opens another data entry window for the operator to define the characteristics of the field to be configured.
HELP CONFIG	This softkey is active only in conjunction with the data entry window displayed with the **TABLE CONFIG** softkey. Before pressing **HELP CONFIG** it is necessary to select the field to be associated to the help message. In turn, **HELP CONFIG** opens another data entry window for the operator to key in the help message (up to 9 36-character lines).
COPY	Copies the selected table into an empty one. When **COPY** is pressed the system visualises a data entry window for the operator to key in the destination table number. If the dimensions of the source and the destination tables are not the same, the copy will be carried out but the characterisation parameters will be truncated or completed. If the table configurator is active, i.e. if the **TABLE CONFIG** softkey has been pressed, **COPY** permits to copy the contents of a field into another.
EXIT	Pemits to exit from the SETUP environment saving the configuration.

Configuring a table

To configure a User Table follow these steps:

1. Press the **TABLES** and **USER TABLE** softkeys from the main screen. If user tables have already been configured, the first User Table Manager screen will be displayed. Otherwise, the video data area will be void.

2. Press the **SETUP** softkey. The following screen will be displayed, which lists the eight configurable tables with the selection bar positioned on the first one. If no tables have been configured, the columns *Softkey ID*, *Start* and *End* will be void.

Table	Softkey ID	Start	End
1	Table 1	xxx	xxx
2	Table 2	xxx	xxx
3	Table 3	xxx	xxx
4	Table 4	xxx	xxx
5	Table 5	xxx	xxx
6	Table 6	xxx	xxx
7	Table 7	xxx	xxx
8	Table 8	xxx	xxx

	LOAD	SAVE		HELP
EDIT	DELETE	TABLE CONFIG	FIELD CONFIG	HELP CONFIG
COPY				EXIT

3. Move the selection bar to the desired table with the arrow keys and press the **EDIT** softkey to select it. The following data entry window will appear:

EDIT TABLE

Softkey :
Start variable :
End Variable :
Reset (Y/N) : **Y**

The meaning of these fields is as follows:

Softkey This string may have as many as 14 characters. It is the label to be applied to the softkey associated to the selected table. This label will appear on the main User Table Manager screen (**TABLE X SFK**).

Start variable Is the index of the first variable associated to the table.

End variable Is the index of the last variable associated to the table.

Reset (Y/N) Enter **[Y]** to delete the existing configuration written in this table. Enter **[N]** to retain this configuration.

4. Key in the requested data. To move from one field to the next press **[Return]** or an arrow key. When the table is complete, press **[Enter]** to return to the previous video screen, whose columns will show the new data.

The above steps must be repeated for each table.

IMPORTANT

Tables cannot be overlapped, i.e. a variable may be displayed only in one table. Check that the *End variable* of one table is smaller than the *Start variable* of the next.

Configuring the table fields

Once the number of variables and the softkeys have been configured it is possible to define the table fields. (This step can be omitted if the default values are to be retained).

To configure a table field use this procedure:

1. Select the desired table by moving the selection bar with the arrows and press the **TABLE CONFIG** softkey.
2. Move the selection bar to the desired field. If the field does not appear on the first page, press **[PgUp]** or **[PgDn]** to display other pages. Overwrite the new field name on the old one.
3. Press the **FIELD CONFIG** softkey. The following data entry window will be displayed:

```
EDIT FIELD

Field Name      :
Sign            :
Integers        :
Decimals        :

Min value       :
Max value       :
```

The meaning of these fields is as follows:

Field name	This string appears on the *Field x* column and cannot be altered in this data entry window.
Sign	Enter **[Y]** to write a number with sign in this field. Enter **[N]** to write a number without sign.
Integers	It is a number from 1 through 5 that specifies the length of the integer part of the variable.
Decimals	It is a number from 0 through 5 that specifies the length of the integer part of the variable.
Min value	Is the minimum variable value.
Max value	Is the maximum variable value.

IMPORTANT

If the number of decimals is less than the configured value, the remaining decimals can be used for accomodating integers as long as the total number of digits remains constant.

For example, if the configured format is 5.5, the following formats are also legal: 6.4, 7.3, 8.2, 9.1 or 10.0. On the contrary, formats like 4.6, etc. are illegal.

4. Key in the requested data. To move from one field to the next press **[Return]** or an arrow key. When the table is complete, press **[Enter]** to return to the previous video screen.

Copying a table or the fields of a table

The COPY softkey permits to perform the following operations:

- Copy a configured table into another one.
- Copy the contents of a field into another field of the same table. This operation can be carried out only during table configuration.

Copying a table

To copy a table into another use this procedure:

1. Display the directory of source tables with the **SETUP** sofkey and move the cursor to the line of the desired sourrce table.
2. Press the **COPY** softkey. The directory of destination tables will appear to the right of the source tables, as shown in the figure below:

Table	Softkey ID	Start	End
1	Table 1	xxx	xxx
2	Table 2	xxx	xxx
3	Table 3	xxx	xxx
4	Table 4	xxx	xxx
5	Table 5	xxx	xxx
6	Table 6	xxx	xxx
7	Table 7	xxx	xxx
8	Table 8	xxx	xxx

Source table

Table	Softkey ID	Start	End
1	Table 1	xxx	xxx
2	Table 2	xxx	xxx
3	Table 3	xxx	xxx
4	Table 4	xxx	xxx
5	Table 5	xxx	xxx
6	Table 6	xxx	xxx
7	Table 7	xxx	xxx
8	Table 8	xxx	xxx

Destination table

	LOAD	SAVE		HELP
EDIT	DELETE	TABLE CONFIG	FIELD CONFIG	HELP CONFIG
COPY				EXIT

3. Use the **[Up/Down Arrow keys]** to move the selection bar to the name of the destination table and press **[Enter]** or **COPY**. After the contents of the source table will be copied into the destination directory the system will return to the previous screen.

If the number of variables in the source table is smaller than that of the destination table, overflowing variables will be truncated. If the number of variables in the source table is greater than that of the destination table, overflowing fields will remain unchanged or void.

Copying table fields

To copy the contents of a table field into another field of the same table the table must be in the configuration environment and the **TABLE CONFIG** softkey must have been pressed. The procedure is as follows:

1. Position the cursor on the destination field.
2. Press the **COPY** softkey. The following data entry window will be displayed:

VARIABLE COPY

Copy from variable number : 0

3. Key in the source variable number and press **[Enter]** or **COPY**. The data entry window will be closed and the contents of the destination variable will be identical to those of the source variable.

Configuring HELP MESSAGES

Each User Table field may be associated to a help message describing the characteristics and the allowed range of the variable. To create this message use this procedure:

1. Select the field to be associated to the message.
2. Press the **HELP CONFIG** softkey to open a data entry window (*Field x*) with 9 36-character lines:

3. Key in the desired message. When the line reaches the border of the window close it with **[Return]** or with the **[Down Arrow].** To create a new paragraph press **[Return]**, not **[Enter]**.
4. When the message is complete, press **[Enter]** or **HELP CONFIG** to confirm it and return to the previous screen.

Loading and saving a table on a file

To load a table configuration file and save it on disk follow the same procedure described for the Configuration utility earlier in this chapter. For more information about these procedures refer to the "Loading a configuration file" and "Saving a configuration file" sections.

Printing a user table

To print a user table configured in this environment it is necessary to use the same procedure as in other environments. The printer must be on and connected to the CPU:

1. Open the table by pressing the relevant softkey (**TABLE X SFK**).
2. Press the **PRINT** softkey. The following data entry window will be displayed:

PRINT TABLE

Enter printer device :

3. Key in the printing device name (LPT1, LPT2, LPT3 or PRN).
4. Press **[Enter]**.

END OF CHAPTER

TOOL MANAGEMENT

GENERAL

Tool Management is a series of procedures that permit to define the tool parameters stored in the Tools, Tool Offsets and Tool Data Base tables.

One of the fundamental parameters for the tools, Presetting, is described in this chapter which also describes how the system manages multiple cutter tools and transfers information from the Tool Data Base into the Tools and Tool Offsets table.

The procedures that permit to manage tool parameters listed in the offsets table are as follows (see Chapter 7):

- visualise and modify tool offsets
- cancel tool offsets
- apply incremental modifications to tool offsets
- add new tools to the Tools table
- visualise and modify tools
- cancel tools.

TOOL PRESETTING

Tool *presetting* is measuring the tool length and storing it in the Tool Offsets table.

Prior to presetting a tool it is necessary to reference the spindle axis.

NOTE:
The procedures described in this section are based on use of the Table Editor.
This chapter deals with *presetting;* for more information about the Table Editor, see Chapter 7.
Tool *presetting* may be performed in two ways:

1) Tool presetting on an external system:

   with this method the tool length is measured directly by a system external to the control.

2) Tool presetting on the machining part:

   with this method the tool is measured by the CNC, using an appropriate procedure.

Tool presetting on an external system

This method permits to measure the tool length using an external system and then write it on the Tool Offsets table.

The procedure is as follows:

1. Open the Tool Offsets table as shown in the "Opening a Table" section in Chapter 7.
2. Select the desired offset with the selction bar.
3. Press the **EDIT** softkey. The following screen will be displayed:

The parameters of the selected tool offset appear on the data entry window.

4. Write the measured tool length in the length 1 field.
5. Press **[Enter]** to confirm it.

Tool presetting on the machining part

This method permits to measure the length of the tool while it is mounted on the spindle, with respect to a fixed point. The length measured will be stored in the Tool Offsets table (see Chapter 7 - Table Editor).

Prior to presetting the tool, the following operations must be performed:

- Refer the spindle axis.
- Make sure there are no active spindle origins or offsets. The spindle must be referenced to the HOME POSITION.
- Measure the distance between the spindle "nose" (which coincides with the *home position*) and the reference point on the part.

1. Press the **AUTO** softkey and then **MDI** to select MDI mode.
2. Key in the following instruction T*n*.0M6 to enable the tool (*n* is the tool number) and press **[ ] [Enter]** to confirm it.
3. Press **CYCLE START** to execute the instruction.
4. Jog manually the spindle axis until it skims the tip of the reference tool.
5. Press **SET ORIG./TOOL** in the MANUAL menu. The system displays the following:

```
ORIGIN / TOOL   PRESET

SELECT (Origin / Tool)      :
NUMBER                      :  1
                            :

 Push ENTER to access the data entry
```

6. Press **[T]** to select TOOL PRESET, move the cursor to the next field of the data entry window and key in the tool name. Press **[Enter]** to confirm the entry. The following data entry window will be displayed:

```
                TOOL   PRESET

OFFSET NUMBER          :  1

REFERENCE AXIS         :

TOOLTIP POSITION
```

Key in the following data:

Offset number selected tool offset number (*offset number* field)

Reference axis spindle axis name, typically Z (*reference axis* field)

Tooltip position measured distance from the spindle "nose" to the fixed reference point (*Tool Tip position* field).

7. Press **[Enter]** to confirm the data entry window.
8. Repeat steps 2 through 7 for each tool to be used in the cycle.

The tool offset will be stored in the Tool Offsets table and can be displayed in the Table Editor environment (refer to Chapter 7).

DEFINING A MULTIPLE CUTTER TOOL

In 10 Series a multiple cutter tool is configured by defining each cutter as a separate tool and assigning the same tool number to all cutters. A multiple tool can have up to 3 cutters (one main cutter and two secondary ones).

To define a multiple tool use this procedure:

1. Define the tool by writing the characteristics of the main cutter as shown in chapter 7 ("Writing a Record in a Table" section).
2. Select the tool defined in the Tools Table with the selection bar and press the **MULTI CUTTER** softkey. The following data entry window will appear in the Tools Table:

Tool pocket, *Random class* and *Tool position* cannot be edited in this window because they belong to the main cutter. In the Tools table the multiple cutter tool is displayed on as many lines as are the cutters. All lines are assigned to the same tool code.

If the tool is multi cutter type and one of the cutters is deleted, the whole multiple cutter will be deleted.

MANAGING THE TOOL DATA BASE

The Tool Data Base is an archive (data base) that may be used to store the characteristics of all the tools available. The Tool Table, on the other hand, contains the characteristics of the tools in the tool magazine of the Numeric Control machine.

Transfering tool data between the Tool Data Base and the Tool Table

If the operator loads a new tool in the tool magazine of the CNC machine for machining requirements, complex measuring operations may be avoided simply by retrieving and loading the data for that tool from the Data Base. The operator can still choose to load or modify the data in question manually.

The paragraphs that follow illustrate how to enter the data for a tool not present in the magazine and also how to restore the original data (the data in the Data Base) of a tool already in the tool magazine.

Entry of data fora new tool in the Tool Table (retrieved from Data Base)

1. Open the tool table in which the tool data must be added. To do this:

- Press the **TABLES** softkey from the main menu.
- Press the **TOOL** softkey to display the directory of Tools tables.
- Select the table and press **[Enter]** or **TOOL** to display it.

2. Press the **INSERT** softkey to open a data entry window, all of whose fields will be either void or set to zero, except for *Random class* , whose default value is 1.

3. Key in the new tool code in the *Tool code* field. It can be the same code assigned to the tool in the Tool Data Base or another code chosen by the user.

4. Key in the the tool code number associated to the new tool in the *offset number* field. If you leave the default value (0), the following parameters will not be imported from the Tool Data Base: *length 1, length 2* and *diameter*.

IMPORTANT

If the *offset number* is not 0, the parameters associated to the offset, i.e. *length 1*, *length 2* and *diameter*, will replace those assigned to the same tool number in the Tool Offsets table.

The other parameters stored in the Tools Offset table, i.e. *Max requalif length 1, Curr requalif length 1, Max requalif length 2, Curr requalif length 2* and *Curr requalif diameter* retain the values they had before the new tool was imported.

5. Press the **TOOL DATA BASE** softkey. The following data entry window will be displayed. The tool identification number provided by the system is the *Tool Code* specified in Step 3.

```
IMPORT TOOL ID:        xxxxxx
```

6. Key in the identification number of the tool to be imported as was specified in Step 3 or, if it is the same as in the Tool Data Base, confirm it.
7. Press **[Enter]** or **TOOL DATA BASE**. The tool parameters will be recopied in the Tools Table. The new tool will be added on the first free record. If no free records are available, the new tool will be added after the last record of the Tools Table.

Overwriting the data of a tool already in the Tool Table with the Data Base data

1. Same as step 1 above.
2. Move the selection bar to the desired tool and press **EDIT**. The data entry window displaying all the stored parameters for that tool will appear. DO NOT modify the *Tool code* value that appears in this window.

IMPORTANT

The tool offset parameters associated to that *Tool Code* through the *Offset number* will also be altered. These parameters are: *length 1*, *length 2* e *diameter*.

To leave these parameters unchanged, modify again the *Offset number*. Write any offset number or 0 if no alteration is needed.

The other parameters stored in the Tool Offsets table, i.e. *Max requalif length 1, Curr requalif length 1, Max requalif length 2, Curr requalif length 2* and *Curr requalif diameter*, retain the values they had before the alteration.

3. Press the **TOOL DATA BASE** softkey. The following data entry window will be displayed. The tool identification number provided by the system is the *Tool Code* specified in Step 2.

IMPORT TOOL ID: xxxxxx

4. Key in the identification number stored in the Tool Data Base that corresponds to the characteristics to be altered in the selected tool.
5. Press **[Enter]** or **TOOL DATA BASE**. The tool parameters will be overwritten on the same record of the Tools Table.

END OF CHAPTER

PART PROGRAM FILE MANAGER

The 10 Series CNC has an environment for management of the *files* of *part programs,* organized in directories.

Main screen

When the **PART PROGRAM** *softkey* is pressed, the environment main screen is displayed:

The screen is divided into 5 functional areas (marked with the letters A÷E in the figure); the letters will be used later in this chapter to refer to these areas.

(A) Standard header of the 10 Series screen pages.
(B) Display of current directory.
(C) List of the last *files* used in work.
(D) *Data entry* containing the file name the softkeys will have effect on.
(E) *Softkey* area.

Standard 10 Series screen page header (A)
The features of this part of the screen page are described in detail in the other chapters of this manual.

Display of the current directory (B)
The name of the current directory is displayed in this area.

A directory is selected by pressing in sequence keys **[PgUp]** and **[PgDn]**.

The Part Program directories configured in AMP may be selected by pressing the **SELECT DIR** softkey; the files present in the directory, identified by the logic name defined in AMP, appear in the window (C).

By pressing the **SELECT DIR** softkey, all the Part Program directories configured in AMP ar selected cyclically.

Last of the part program files worked on (C)
Window (C) lists the names of the last *part programs* on which work was performed (max. 10). It permits rapid selection of the most frequently used files.

A desired program is selected by moving the cursor (light blue coloured bar) with **[Up\Down Arrow keys]** ; the file currently selected appears in window (D).

Keys **[Home]** and **[End]** respectively select the first and last program of the window.

NOTE:

Pressing **[Control] [D]** resets the contents of this list.

Data entry (D)
The data entry (D) contains the name of the *file* selected by the selection bar. Action of softkeys **EDIT**, **ACTIVATE**, **DEACTIVATE**, **COPY**, **DELETE**, **RENAME** will be on the file of this name.

The *filename* may be changed by keying in another name on the keyboard.

SOFTKEYS

The softkeys of the *part program file manager* screen page (E) are as follows:

COPY	DELETE	RENAME	PROG UTILITY	
EDIT	SELECT DIR	DIRECTORY	DEACTIVATE	ACTIVATE
		PP MANAGEMENT		

- The **COPY**, **DELETE**, **RENAME**, **EDIT**, **ACTIVATE** softkeys produce the following action on the file indicated in the data entry (D):

  COPY Copies the part program file into another file with a different name.

  RENAME Renames the file.

  DELETE Deletes the file.

  EDIT Activates the editor on the file. If the file specified in the input line does not exist, the system will create one with that name.

  ACTIVATE Permits activation of the part program file. Activation of one program results in de-activation of an already active one, where applicable.

  PP MANAGEMENT This softkey is only present if part programs with logic directories are used (see the SW characterisation manual - AMP).

- The **DEACTIVATE** softkey de-activates a currently active program.
- The **PROG UTILITY** softkey activates a user utility acting on part programs; it must be declared in AMP (see the SW characterisation manual - AMP).
- The **SELECT DIR** softkey displays in areas (B) and (C) a list of all the files in the current directory.
  Repeated pressing of this softkey automatically displays the files in the other directories on the system.
  To restore display of (B) and (C) press **[Esc]**.
- Softkey **DIRECTORY** displays in areas (B) and (C) a list of all the files in the directory specified in the data entry opened when this softkey is pressed.
  The name of the directory must be given in its standard DOS format, also including the device (examples: E:\FILE, E:\FILE*.PRO, E:\FILE\PIPPO?.*).
  To restore display of (B) and (C) press **[Esc]**.

NOTE:

The maximum number of files that a directory can hold is 300; if a DOS directory contains more than 300 files, only the first 300 will be displayed, subdirectories included.

LINE EDITOR

Line Editor is a utility that permits to write and edit part programs. You can use the Line Editor to write programs in standard ISO language (RS 274 code). You can also use the Line Editor in conjunction with the Profile Editor to also program profiles in geometric language.

This section describes the Line Editor screen and softkeys, as well as the operations that can be performed in this environment.

The Line Editor is activated by pressing the **EDIT** softkey in *part program file manager* environment.

Line Editor screen

```
A  LINES:00013      FILE:  MAINPROG              DATE: 15/10/90          TIME: 9:28:21
B  00010 G3X26I93J57F100

   G1X5Y20F1000
   Y75
C  X34.78967Y100
   X95
   Y80
   X50
   G3Y66.587I50J73F1000
   G1X95
   Y19.999
D  G3X26I93J57F100
   G1X150
   Y57.987

C

   CONFIG        CYCLE MODIFY   CUT & PASTE                   HELP
E  DEL-LINE      OLD-LINE       SEARCH        INCLUDE         SEQUENCE
   GEOMETRY      CYCLE                                        EXIT
```

(A) *status information area*

(B) *old block line*

(C) *part program area*

(D) *editing line*

(E) *softkey area*

The Line Editor screen is made up of the following areas:

Status line (A)
Is the first line and provides general information such as number of blocks, part program name, date and time.

Old block line (B)
Displays the current line number and an unmodified copy of the block that is being edited. If no alterations are made, the old block line will be identical to the editing line.

Part program area (C)
This area is reserved for displaying the part program.

Editing line (D)
This red line appears in the middle of the screen and is the only area in which writing is enabled. It is used for displaying the block to be edited. To take a block to the editing line press the **[Up/Down Arrow keys]**.

Softkey area (E)
The three lines at the bottom of the screen display the Line Editor softkeys.

The functions of the Line Editor softkeys are as follows:

SOFTKEY	FUNCTION
CONFIG	Modifies the Editor configuration (see the 10 Series CNC - Graphic Editor manual).
CYCLE MODIFY	Enables the Cycle Editor, a utility that permits to modify fixed cycle parameters. It is active only if the Graphic Editor option has been configured and the file is not profile type (without extension .DFP) - (see the 10 Series CNC - Graphic Editor manual).
CUT&PASTE	Selects a group of lines for copying, cutting, moving, pasting, etc.
HELP	Displays information about the active softkey menu or an open data entry window.
DEL-LINE	Deletes the block displayed on the editing line. The parameters of the deleted block remain on the old line but can be deleted by pressing **DEL-LINE** twice. After being deleted from the old line these parameters can no longer be retrieved with the **OLD-LINE** softkey.
OLD-LINE	Retrieves a block that has been cancelled from the editing line by pressing **DEL-LINE** once. The retrieved block is the original one, i.e. all the alterations will be lost.
SEARCH	Permits a search for strings or block numbers. The system displays a data entry window prompting for the desired block number or string.
INCLUDE	Inserts all the blocks of a file in a predefined point of the current file.
SEQUENCE	Numbers a list of part program blocks in the desired sequence. When **SEQUENCE** is pressed the system displays a data entry window prompting for the initial number and the numbering pitch (only enabled on files without extension .DFP).
GEOMETRY	Displays the Profile Editor video panel for definition of geometric profiles. This softkey is enabled only of the Profile Editor is enabled and the file has extension .DFP (see the 10 Series CNC - Graphic Editor manual).
CYCLE	Displays the video panel for canned cycle definition. Active only of the Cycle Editor is configured and if the file does not have extension .DFP (see the 10 Series CNC - Graphic Editor manual).
EXIT	Quits the Line Editor. It opens a data entry window with a prompt to save the file.

Creating a new part program

To create a new part program, key in a non-existing *program name* in the data entry (D) of the *part program file manager* screen (page 10-1) and then press **EDIT**.

Example:

PROGRAM NAME :	**NEW**

The editor screen page is now displayed (page 10-4), with areas (B), (C) and (D) empty.

Key in the program name, which is an ASCII string of up to 48 characters plus 3 characters for the extension. The extension is mandatory for profile files (DFP) and will be used by the Profile Editor.

Loading an existing Part Program

To access an existing part program, select the name in the C section of the video panel "Part ProgramFile Manager" (page 10-1), using the **[Up/Down Arrow]** to move the selection bar and press the EDIT softkey.

The part program is displayed in sections B C and D of the Line Editor video panel.

From this point on, the Editor operations described on the following pages are enabled.

Loading a running Part Program

It is possible to call the Editor on a part program already in execution (activated through softkey **ACTIVATE**).

In this case, display of the part program in the Editor video panel is preceded by the message:

File already open: all changese will be lost

Type <ESC> to exit or <ENTER> to continue.

The part program will be opened in read only and any editing perfromed on it will not be saved.

Characteristics of the Line Editor

Line overflow
A part program block can have as many as 126 characters, but the maximum length of the editing line field is 79 characters. To move the cursor inside the block use the **[Right/Left Arrow keys]**.

If you key in a block on the editing line and the cursor reaches the last position (column 80), it is always possible to add characters beyond this point. With each new character the block displayed on the editing line will move to the left by one position.

Displaying the original block
The old block line area displays the same block as is on the editing line. If this block is longer than one line, the second line will also be displayed on the old block line.

Cursor
The cursor is a blinking block that can be moved inside the editing line with the **[Left/Right Arrow keys]**. It remains on the first position to the right of the last inserted character and always operates in 'insert' mode.

The **[Up/Down Arrow keys]** permit scrolling of a file up or down on screen, whereas the editing line remains in a fixed position.

When the file scrolls on the display, the editing line displays the previous or next block depending on whether the file is scrolling down or up.

Line Editor function keys

The following function keys are active with the Line Editor:

FUNCTION KEY	MEANING
◀	moves the cursor to the left on the editing line by one character.
▶	moves the cursor to the right on the editing line by one character.
▲	scrolls the displayed file downwards; the editing line will display the previous part program block.
▼	scrolls the displayed file upwards; the editing line will display the subsequent part program block.
PAGE↑	moves the cursor upwards by nine lines.
PAGE↓	moves the cursor downwards by nine lines.
CTRL ▶	moves the cursor to the last block character.
CTRL ◀	moves the cursor to the first block character.
CTRL HOME	moves the cursor to the first program character and displays the first program block on the editing line.
CTRL END	moves the cursor to the last program character and displays the last program block on the editing line.

FUNCTION KEY	MEANING
←	deletes the first character to the left of the cursor.
DEL	deletes the character to which the cursor is positioned.
↵ ↵	opens a new line on the editing line moving the currently displayed block upwards. Confirms the data written in a data entry window.
END	splits the block displayed on the editing line from the current position of the cursor. All the characters to the right of the cursor will be transferred to the subsequent line. If the cursor is on the first character, this key opens a new line that precedes the block displayed on the editing line.

Writing new characters in a block

New characters can only be written on the editing line as the cursor is always positioned on it. An inserted block must be confirmed with **[Enter]** or **[Return]**, which causes the cursor to moves to a new editing line.

When a new line is selected with the **[Up/Down Arrow keys]**, the empty lines are automatically removed. Empty lines are also automatically removed when you quit the Editor.

Program blocks can be edited only when they are on the editing line and must therefore be moved to this line with the **[Up/Down Arrow keys]**, before being altered. The **[Left/Right Arrow keys]** permit to move the cursor within the editing line, whereas **[BackSpace]** deletes the characters to the left of the cursor (refer to the "Line Editor function keys" section in this chapter).

Opening a new line

There are two ways of opening a new line:

1. if the new line must be written after the block displayed on the editing line, press **[Return]** or **[Enter]** with the cursor on any position.
2. if the new line must be written before the block displayed on the editing line, press **[End]** with the cursor positioned on the first block character.

IMPORTANT

If the cursor is on any other position, when you press **[End]** the block will be split. To restore it, press **OLD-LINE** immediately afterwards.

Deleting lines

Part program blocks may be deleted with the **DEL-LINE** softkey, which has two different functions depending on the number of times it is pressed.

If you press **DEL-LINE** once, the block is deleted but can be retrieved with **OLD-LINE**. If you press **DEL-LINE** twice, the block is deleted irreversibly and cannot be retrieved. In both cases, **DEL-LINE** permits to delete the block displayed on the editing line.

The table that follows summarizes the functions of **DEL-LINE**.

PRESS	EFFECT
DEL-LINE once	the block displayed on the editing line is deleted and the editing line appears empty. To retrieve the deleted block, press **OLD-LINE**.
DEL-LINE twice	the block displayed on the editing line is deleted, the editing line displays the subsequent program block, and the empty line is removed. **OLD-LINE** does not permit to retrieve the deleted block.

IMPORTANT

Pressing **DEL-LINE** once and then moving the cursor with the **[Up/Down Arrow keys]** is equivalent to pressing **DEL-LINE** twice.

To cancel a line use this procedure:

1. Take the block to be deleted to the editing line with the **[Up/Down Arrow keys]**.
2. Press **DEL-LINE** twice.

IMPORTANT

If the block to be deleted is the last block in the file, press **DEL-LINE** once and then press **[Up Arrow]**.

Retrieving lines

The **OLD-LINE** softkey is used for:

- retrieving the original version of the block displayed and modified on the editing line.
- retrieving the block deleted from the editing line. This is possible only if **DEL-LINE** has been pressed only once.
- for reconstructing an orginal block that was split by pressing the **[End]** key with the cursor located inside the block.

To retrieve the original block displayed on the first line (or on the first two lines), use this procedure:

1. Do not move the cursor from the editing line.
2. Press **OLD-LINE**.

The contents of the old block line will be copied to the editing line.

Configuration Editor

The **CONFIG** softkey permits to customize parameters to specific requirements. Configuration is performed with the 3 data entry pages described below:

```
---------------- GLOBAL PARAMETERS -------------

Keystroke                :20

Cycle screen 1           :
Cycle screen 2           :
Cycle screen 3           :
Geometry type : GEOMIL

                                          1 of 3
```

Significance of the fields in the **GLOBAL PARAMETERS** data entry is as follows:

Keystroke Number of keys to press before saving in the "recovery" file.

Cycle screen 1÷3 Name of the screen files for the cycle editor.
see the 10 Series CNC - Graphic Editor manual)

Geometry type Name of the screen file for the geometry editor.
(see the 10 Series CNC - Graphic Editor manual)

```
---------------- AXIS CONFIGURATION ------------

         Name  Diam  Minval        Maxval

Axis     X     N     0.000         100.000
Axis     Y     N     0.000         100.000

-------------- GEOMETRY PARAMETERS ---------

Orientation              1
Tolerance                :0.00100
Incremental center       :N
Unit (0) mm     (1)inch :0
                                          2 of 3
```

Significance of the fields in the **AXIS CONFIGURATION** data entry is as follows (see the 10 Series CNC - Graphic Editor manual):

Axis name: X Name of the axis of abscissas. This name will replace the X axis in the translated part program.

Diam (diametral) Axis type:

N = normal
D = diametral
R = radial

Minval Lower limit of first axis for graphic scale declaration. This field may contain values between -999999.999 and +999999.999; the distance between the value set as **Minval** and the value set for **Maxval** must be ≥ 0.06.

Maxval Upper limit of first axis for graphic scale declaration. This field may contain values between -999999.999 and +999999.999; the distance between the value set as **Maxval** and the value set for **Minval** must be ≥ 0.06.

Axis name: Y Name of the axis of ordinates. This name will replace the Y axis in the translated part program.

Significance of the fields in the **GEOMETRY PARAMETERS** data entry is as follows:

Orientation Defines axis orientation according to the following scheme:

Tolerance Declares the tolerance to be used in calculating tangency between elements (0.0 < T < 0.05)

Incremental center If the declaration is **[Y]**, the ISO program is translated with the incremental circle center, otherwise (**[N]**) it is translated with absolute dimensions.

Unit (mm/inches) Unit of measurement 0=millimetres, 1=inches

```
--------------- CYCLE PARAMETERS ---------------

        Real name      Symbolic

Axis1   X              X
Axis2   Y              Y
Axis3   Z              Z
Axis4
Axis5
Axis6
Axis7
Axis8
                                        3 of 3
```

The fields in the **CYCLE PARAMETERS** data entry establish the symbolic names for the machine's physical axes (see the 10 Series CNC - Graphic Editor manual).

NOTE:
Valid names for axes are as follows: X,Y,Z,A,B,C,U,V,W,P,Q,D.

Numbering program blocks

Part program blocks may be numbererd with the **SEQUENCE** softkey.

The procedure is as follows:

1. Press the **SEQUENCE** softkey. The following will be displayed:

2. Key in the first block number. The maximum allowed value is 999000.
3. Key in the increment. The maximum allowed value is 200.
4. Press **[Enter]** or the **SEQUENCE** softkey to confirm these entries.

To move the cursor from one field to the next press **[Return].** If the increment is zero, the part program will not be numbered. If a zero increment is given to a numbered program, the block numbers will be removed.

Search for strings and block numbers

The **SEARCH** softkey permits to search for strings and/or block numbers.

The procedure is as follows:

1. Press **SEARCH**. The system displays the following data entry window for the operator to write the desired string or block number:

2. To move the cursor from the *Line number* field to the *Character string* field press **[Return]**.
3. Depending on the object of your search, you may key in:

   - a string up to 12 characters in length. Press **[Enter]** and then **[Up Arrow]** or **[Down Arrow]** to search for the string upwards or downwards. The searched string will be displayed on the editing line.
   - a line number having up to 6 digits. Press **[Enter]** to confirm it. The searched line will be displayed on the editing line.

   NOTE:
   By line number is meant the number displayed in the status line (top left on the line editor screen page)and not a number given with the SEQUENCE function.

In a line number search, if the number searched for is greater than the last program block number, the last block will be displayed in the editing line.

IMPORTANT

To quit the search after the item has been found, press the **SEARCH** softkey again.

If the desired string is not found, an error message will be displayed. If both a string and a block number have been written in the data entry window, the system will search for the block number only.

Cut & Paste

The CUT&PASTE softkey opens the following menu:

START SELECT	END SELECT	DESELECT		HELP
			SEARCH	
DELETE	COPY	MOVE		EXIT

Initial softkey status:

ENABLED	DISABLED
Start select	End select
Help	Deselect
Exit	Delete
Search	Copy
	Move

NOTE:
This video screen is similar to the one you open with the Line Editor.

The meaning of the softkeys is as follows:

SOFTKEY	FUNCTION
START SELECT	Marks the current line as the first line of the program section to be selected for cut&paste purposes. The section may be selected with **[Up Arrow], [Down Arrow], [PgUp]** and **[PgDn]**. The current line/s is/are highlighted in yellow. Once trhe selection has been made, the system disables **START SELECT** and enables **END SELECT** and **DESELECT**.
END SELECT	Marks the current line as the last line of the program section to be selected for cut&paste purposes. The section remains highlighted in yellow. Once the selection has been made, the system disables **END SELECT** and enables **DELETE, COPY** and **MOVE**.
DESELECT	Deselects highlighted line/s. After the section is deselected, this softkey returns to the initial status.
SEARCH	Softkey with the same functionality as the one in the Line Editor main video panel. It is always enabled and does not alter the status of other softkeys.
DELETE	Deletes the selection confirmed with the data entry. The deleted section cannot be restored. After the selection has been deleted, **DELETE** returns to the initial status
COPY	Copies the selected block to the current cursor position. After this operation the selection is lost and the block may not be used again. Status of the softkeys is not modified.

SOFTKEY	FUNCTION
MOVE	Moves the block selected to the current cursor position. After this operation the selection is lost and the block may not be used again. After the section is deselected, this softkey returns to the initial status.
HELP	Displays the HELP page(s) describing all the features available on this screen page.
EXIT	Returns to the Line Editor main screen. Any still active selection is lost.

Inserting a part program

The **INCLUDE** softkey permits to paste various modules into the current part program. The insert is always pasted after the block displayed on the editing line.

To insert a part program use this procedure:

1. Display on the editing line the block after which the program must be inserted.
2. Press the **INCLUDE** softkey. The following will be displayed:

FILE NAME :

3. Key in the name of the part program to be inserted and press **[Enter]**.

Saving a part program

To save a part program use this procedure:

1. Press the **EXIT** softkey. The following will be displayed:

SAVE FILE ? (Y/N) : Y

2. Key in

- **[Y]** and press **[Enter]** to save the part program on disk.
- **[N]** and press **[Enter]** to abort the procedure without saving the program on disk.

Both responses permit to quit the Editor and return to the directory of programs.

END OF CHAPTER

PART PROGRAM EXECUTION

SETTING PART PROGRAM PARAMETERS

Prior to executing a part program it is possible to set a series of parameters that affect the execution mode and enable/disable controls, visualisations and special operations.

These parameters are listed on a data entry window that can be displayed by pressing the **PROGRAM SET-UP** softkey in the menu associated with the **MACHINE SET-UP** softkey.

```
                    PROGRAM SETUP  [mm]

BLOCK DELETE                    (Y/N):     Y
OPTIONAL STOP - M001            (Y/N):     Y
FEEDRATE BYPASS                 (Y/N):     N
RAPID OVERRIDE CONTROL          (Y/N):     N
DISABLE PROGRAM SCROLL          (Y/N):     N
HORIZONTAL AXIS:  X             VERTICAL AXIS:  Y
ROTATION ANGLE        :    0.00000
STOCK ALLOWANCE       :    0.00000
 AXIS NAME    LOCKED (Y/N)   MIRROR (Y/N)   SCALE (Y/N)    SCALE FACTOR
      X            N              N              N              0.000
      Y            N              N              N              0.000
      Z            N              N              N              0.000
```

The meaning of these fields is as follows:

BLOCK DELETE (Y/N) enables/disables execution of slashed blocks. Enter **[Y]** to disable slashed block execution (This is equivalent to programming DSB = 1).

At power up this field displays the selection made during characterization in AMP. Resetting the system does not alter the previous characterization.

OPTIONAL STOP - M01 (Y/N)

enables/disables program stop.

At power up this field displays the selection made during characterization in AMP. Resetting the system does not alter the previous characterization.

FEEDRATE BYPASS (Y/N)

enables/disables the bypassing of the rapid rate with G1, G2 or G3.

At power up the default mode is **[N]**. Resetting the system changes this field to **[N]**.

RAPID OVERRIDE CONTROL (Y/N)

enables/disables rapid override control with the softkeys **RAPID OVER +** and **RAPID OVER -** from the AUTO menu.

At power up this field displays the selection made in AMP. Resetting the system does not alter the previous characterization.

DISABLE PROGRAM SCROLL (Y/N)

enables/disables program scrolling in AUTO mode. If scrolling is disabled, program execution is faster.

At power up this field displays the selection made in AMP. Resetting the system does not alter the previous characterization.

HORIZONTAL AXIS

define the machining plane. It is equivalent to programming the G16 code in a block.

VERTICAL AXIS

At power up or after a reset this field displays the selection made in AMP: G17, G18 or G19.

ROTATION ANGLE

defines a rotation angle expressed in degrees for the machining plane. A positive angle determines a CCW rotation. Unlike programming with the three-letter instruction (ROT, angle), this rotation angle is not canceled by a reset.

If a (ROT, angle) is programmed after a rotation angle has been assigned through this data entry window, the final rotation angle will be the sum of both angles.

If a (ROT, angle) is programmed with the three-letter instruction before setting a rotation angle through this data entry window, the final rotation angle will be the latter, i.e. the rotation angle programmed with (ROT, angle) will be lost.

STOCK ALLOWANCE

defines the stock allowance to be applied to the tool radius when calculating the tool path with tool offset (G41 - G42). It is equivalent to programming the MSA block.

At power up or after a reset this field is set to zero.

The following parameters refer to each configured axis:

LOCKED (Y/N) enables/disables physical axis motion. When you write **[Y]** the system executes the part program without physically moving the axes.

This setting is not disabled by a reset.

MIRROR (Y/N) enables/disables mirroring of the specified axis trajectory. It is equivalent to programming with the three-letter code (MIR, axis name).

This setting is disabled by a reset.

SCALE (Y/N) enables/disables the scale factor associated to the specified axis.

This setting is disabled by a reset.

SCALE FACTOR defines the scale factor applied to the specified axis. It is equivalent to programming the (SCF, axis name, scale factor) block.

This setting is disabled by a reset.

EXECUTION OF A PART PROGRAM

There are two part program execution modes:

- Auto
- Semiauto (block by block)

In order to execute a part program it is necessary to:

1. Select and activate the part program
2. Select the execution mode
3. Launch execution.

Selecting and activating a part program

These operations must be carried out before selecting the execution mode. The procedure is as follows:

1. Press the **PART PROGRAM** softkey in order to enter the *Part Program File Manager* environment.
2. Select a part program for execution as described in Chapter 10.
3. Press the **ACTIVATE** softkey. A **[Y]** will appear on the A column to the left of the selected program name.

Automatic execution

This mode permits to execute all the blocks in a program without interruptions. To execute the program in auto mode, use this procedure:

1. Select and activate the part program
2. Press the **AUTO** softkey from the main menu.
3. Press the **CYCLE START** pushbutton.

The selected part program will be executed from the first to the last block.

Semiauto (block-by-block) execution

This mode permits to execute one part program block at a time. The procedure is as follows:

1. Select and activate the part program.
2. Press the **AUTO** softkey from the main menu.
3. Press the **BLK/BLK** softkey to enable block-by-block execution (the softkey will be shown yellow in colour and will remain so until it is disabled).
4. Press **CYCLE START** each time you want to execute a block. Blocks will be executed in the same sequence in which they are programmed. After executing a block the system stops until the operator gives another **CYCLE START** command.

To alter the sequence of block execution or bypass one or several blocks, use the arrow keys to position the cursor to the desired block.

To close block-by-block execution, press **BLK/BLK** again. The disabled softkey is white.

IMPORTANT

If **BLK/BLK** is pressed during automatic execution, the system executes all the preanalysed blocks and then stops. To resume part program execution in block-by-block mode, press **CYCLE START**.

Multiblock retrace

This function permits to retrace a series of programmed blocks by moving the axes backwards along the executed profile. Depending on the parameter configured in AMP, the number of retraceable blocks varies from 1 through 64.

The procedure is as follows:

1. Press the **CYCLE STOP** pushbutton to stop the axes and **MACHINE SET-UP** from the main menu.
2. From **MACHINE SET-UP**, press the **BLOCK RETRACE** softkey, enter **[Y]** in the **ENABLE** field of the data entry window and confirm it with **Enter**.
3. Press the **CYCLE START** pushbutton. The axes will move backwards in the selected mode:
   - In BLK/BLK mode the profile will be retraced one block at a time each time the **CYCLE START** pushbutton is pressed.
   - In AUTO mode the system will retrace the last n blocks in the profile. n is the number of blocks configured in AMP.
4. When the retrace move is completed the following message appears:

   NC 153 End of block retrace

To interrupt a retrace move, press **CYCLE STOP**. At this point you can either resume the retrace move or invert the direction of motion. To resume retracing, press **CYCLE START**.

To invert the direction of motion use this procedure:

1. Press **BLOCK RETRACE**, enter **[N]** in the **ENABLE** field of the data entry window and enter **[Y]** in the **EMIT AUXIL FUNCT** field if you want the auxiliary functions to be emitted during the forward stage of the BLOCK RETRACE move, or enter [N] if you do not want the auxiliary functions to be emitted. Then confirm with **[ENTER]**.
2. Press the **CYCLE START** button. The axes will move in the mode selected at the previous step.

NOTES:

- During part program execution in AUTO mode the BLOCK RETRACE function can be performed in BLK/BLK mode. Part program execution will be resumed automatically when the profile has been retraced and the axes have returned to their stop point.
- When the BLOCK RETRACE function is active the axes move in point-to-point mode (G29) in the backwords retrace, while in the forwards trace they are executed according to the programmed modality (G27,G28,G29).
- In BLOCK RETRACE mode, the control unit performs the motion blocks only and, on request, the emission of auxiliary functions; assignment blocks are ignored.
- The emission of auxiliary functions during the BLOCK RETRACE forward move can also be enabled by assigning the **MBA** process variable.

  MBA = 1, emission is enabled
  MBA = 0, emission is disabled
- When the BLOCK RETRACE function is active the feedrate can be altered with the FEEDRATE OVERRIDE function (see later in this chapter).

Executing blocks from keyboard

The system permits to execute blocks entered from keyboard (MDI) both when a part program is active and when no part program is selected.

MDI execution without an active part program

The procedure is as follows:

1. Press the **AUTO** softkey from the main menu.
2. Press the **MDI** softkey to enable manual data input mode. The system displays a window for the operator to key in the desired block. When enabled, the **MDI** softkey is yellow.

3. Write the block to be executed in the data entry window.
4. Press **[Enter]** to confirm the input block. As the block is confirmed, the MDI line displayed in the data entry window will turn white. When the data entry window is opened, the displayed MDI line is always the last confirmed block. If no block has been confirmed, this line will be void. By pressing any key, the confirmed block will be disabled.
5. Press **CYCLE START** to execute the inserted block.

To input other blocks for execution repeat steps 3 and 4. To cancel the contents of the MDI line press the **[Clear line]** key. To disable MDI mode, press **MDI** again. This softkey will return white.

Using the **[Up Arrow]** and **[Down Arrow]** keys with the MDI input window active, the last MDI blocks set are reactivated and can be executed again by pressing the **CYCLE START** button.

The last 10 blocks set are stored.

MDI execution with an active part program

While the system is executing a part program it is possible to execute blocks in MDI mode. The input block does not affect the active part program and is not retained in the system memory.

To execute a block in MDI mode use this procedure:

1. Press the **MDI** softkey to enable manual data input mode. The system displays a window for the operator to key in the desired block. If the program has not been stopped with the **BLK/BLK** softkey, the system will stop after executing all the pre-analysed blocks.
2. Write the block to be executed in the data entry window and press **[Enter]** to confirm it.
3. Press **CYCLE START** to execute the confirmed block.

To input other blocks for execution repeat steps 3 and 4. To cancel the contents of the MDI line press the **CLEAR LINE** key. To disable MDI mode, press **MDI** again. This softkey will return white.

To restore automatic part program execution press **CYCLE START.** To resume block-by-block execution re-press **BLK/BLK** and then press **CYCLE START.**

IMPORTANT

If **MDI** is pressed during automatic execution, the system executes all the preanalysed blocks and then stops. After MDI execution, return to auto or block-by-block mode and press **CYCLE START**.

Execution of paramacros in MDI mode

This function makes it possible to run in MDI mode the paramacros called implicitly by the S,T,M, auxiliary functions, which must have been configured beforehand as paramacros.
For information on how to configure and program these functions as paramacros, see the AMP Manual and the Programming Manual.

To run a paramacro in MDI mode, proceed as follows:

1. Activate the auxiliary function in MDI mode, as described in points 1 and 2 of the previous paragraph. This displays the name of the corresponding paramacro in the list of active programs.
2. Deactivate the MDI mode by pressing the **MDI** softkey again.
3. Press the **[CYCLE START]** button to execute the paramacro.

At the end of the execution, the paramacro is closed and the system returns to MDI mode.

NOTE:
If you try to activate the same paramacro several times, the error message “programme already selected” (error 149) appears.

Executing part of a program

When part program blocks are numbered the system can execute the part of program delimited by two block numbers.

The procedure is as follows:

1. Select and activate the part program.
2. From the **AUTO** menu, press the **EXEC FROM:TO** softkey to display the data entry window below.

3. Key in the starting and end block numbers
4. Press **[Enter]** or the **EXEC FROM:TO** softkey
5. Press **CYCLE START** to start executing the part of the program delimited by the specified blocks.

Modifying blocks in BLK/BLK mode

Part program blocks can be altered and executed from the keyboard. These alterations do not affect the original programmed block and are not saved in the active program.

To modify a programmed block use this procedure:

1. Press the **AUTO** softkey in the main menu.
2. Press the **BLK/BLK** softkey.
3. Execute the program up to the block to be edited.
4. Press the **MDI** softkey to enable MDI mode.
5. Press the **BLOCK MODIFY** softkey to display the block on the data entry window.
6. Edit the block displayed in the data entry window and press **[Enter]** to confirm the modifications.
7. To execute the modified block press **CYCLE START**.
8. To quit the BLOCK MODIFY status press the **MDI** softkey, which closes the data entry window, and re-press **BLK/BLK.**

NOTE:
This procedure is not applicable when tool offset is active (G41, G42).

Restoring a part program

When the system is in block-by-block mode the **[Up/Down Arrow keys]** permit to scroll the part program listing backwards or forwards.

To return the cursor to the block programmed after the last executed block, press **HOME.** Execution may now be resumed from this point.

Search for a string

The following procedure permits to search for a string of characters in an active program.

1. Press the **STRING SEARCH** softkey in the AUTO menu. The system displays a data entry window for inputting the desired ASCII string.
2. Input the ASCII string.
3. Press **STRING SEARCH** or **[Enter]** to confirm the string.
4. Press the **[Up Arrow key]** to search for the string upwards or **[Down Arrow]** to start a search downwards.

When the search is completed, the string is displayed in reverse mode.

If the string is not found after searching in a given direction, one of the following error messages appears: "End of File" or "Beginning of File".

Overriding the programmed feedrate

The feedrate value to be used in the program is specified by the *F value* parameter. To alter this value without modifying the program use this procedure:

1. From the **AUTO** menu, press the **OVERRIDE SEL** softkey until the labels of first two softkeys on the third line are FEEDRATE + and FEEDRATE -.

2. Press the **FEEDRATE +** or **FEEDRATE -** softkeys to increment/decrement the programmed value. Each time one of these softkeys is pressed a 12,5% increment/decrement is applied. The allowed variation ranges from 0% to 125% of the configured value. If the configured value is the maximum feedrate, variation will range from 0% to 100%.

Example:
If the programmed feedrate value is 2000 MMPM (mm per minute) and a 50% variation is applied, the actual feedrate will be 1000 MMPM.

Overriding the spindle speed

The spindle speed value to be used in the program is specified by the *S value* parameter. To alter this value without modifying the program use this procedure:

1. From the **AUTO** menu, press the **OVERRIDE SEL** softkey until the labels of first two softkeys on the third line are SPND OVER + and SPND OVER -.

2. Press **SPND OVER +** or **SPND OVER -** to increment/decrement the programmed spindle speed. Each time one of these softkeys is pressed a 12,5% increment/decrement is applied. Allowed increments/decrements range from 50% to 150% of the configured value.

Example:
If the programmed spindle speed is 1500 RPM and a 50% variation is applied, the actual spindle speed will be 750 RPM.

Overriding the rapid

The rapid is configured in AMP. To alter this value use this procedure:

1. From the **AUTO** menu, press the **OVERRIDE SEL** softkey until the labels of first two softkeys on the third line are RAPID OVER + and RAPID OVER -.

2. Press **RAPID OVER +** or **RAPID OVER -** to increment/decrement the configured rapid. Each time one of these softkeys is pressed a 12,5% increment/decrement is applied. Allowed increments/decrements range from 0% to 125% of the configured value.

Example:
If the configured rapid is 3000 MMPM (millimeters per minute) and a 50% variation is applied, the actual rapid will be 1500 MMPM.

Execution in *Dry Run* mode

In *Dry Run* mode (with axes disconnected), it is possible to try to execute a part program without any command being sent to the drives of the axes or to the machine logic (I/O inhibited).

This feature is activated by pressing the **DRY RUN** softkey in the **AUTO** menu.

With **DRY RUN** active, it is also possible to verify program correctness graphically by way of the **MACHINE PLOT** utility.

For further details about use of **MACHINE PLOT**, see chapter 12.

MEMORY SEARCHING

Memory searching means searching for the interruption block and subsequently resume the interrupted cycle automatically or searching for a pre-set block where to start machining.
The parameters which identify univocal an executing cycle are stored and continuously updated during the execution of the program.
According to these parameters it is possible to carry out an automatic searching.

Memory searching can be accomplished in two ways:

- **Automatic memory searching for the interruption block** (due to switch-off or reset of the machine)
- **Memory searching for a pre-set block.**

Automatic searching

To search automatically for the block which interrupted the part program, proceed as follows:

1. Select the part program in which machining was interrupted.
2. Set **AUTO** mode (**AUTO** is displayed).
3. Press the **SEARCH MEMORY** softkey: the system enters the SEARCHING mode and the first line of the screen displays RCM ON reverse image.
4. Press **CYCLE START**: the system runs the searching and in the end the message **NC156 end of search in memory** and the programmed dimensions of the axes position for the cycle reset are displayed. The block of the part program which will be executed at the end of the search, when the program is restarted, is displayed on reverse image. By switching to **BLK/BLK** and pressing **CYCLE START**, the search can be continued to the desired block.
5. Press again the **SEARCH MEMORY** softkey to exit from MEMORY SEARCH mode.

The systems is now ready to restart the working cycle.

Searching for a pre-set block

To search for a **Nxxxx** block of a part program proceed as follows:

1. Select the part program.
2. Set **AUTO** mode (**AUTO** is displayed).
3. Press the **SEARCH MEMORY** softkey: the system enters the SEARCHING mode and the first line of the screen displays RCM ON reverse image.
4. Press the **EXEC FROM:TO** softkey and set the starting and the finishing blocks for the memory searching in the data entry. The starting block can also be omitted (by default part program starts with the first block); the finishing block stands for the previous block from which you want to restart the cycle. The searching blocks cannot be included in a subprogram.
5. Press **CYCLE START**: the system runs the searching and in the end the message **NC 156 End of search in memory** and the programmed dimensions for the position of the cycle reset are displayed. The block of the part program which will be executed at the end of the search, when the program is restarted, is displayed on reverse image. By switching to **BLK/BLK** and pressing **CYCLE START**, the search can be continued to the desired block.
6. Press again the **SEARCH MEMORY** softkey to exit from MEMORY SEARCH mode.

Before exiting the MEMORY SEARCH mode, it is possible to continue the searching up to another block resetting the number by means of the **EXEC FROM:TO** softkey and pressing **CYCLE START** again

Restarting a working cycle

After exiting the MEMORY SEARCH mode to restart the working cycle, proceed as follows:

1. Press **CYCLE START**: the system send auxiliary functions to the machine logic and enters **HOLD** status. Auxiliary functions are sent also in SEARCHING mode provided that the logic has been adequately pre-set.
2. Position axes on the programmed dimensions reversing on the profile (**JOG RETURN** softkey, **MANUAL** mode).
3. Select again **AUTOMATIC** OR **SEMIAUTOMATIC** mode.
4. Press **HOLD** to exit from **HOLD** status.
5. Press **CYCLE START** to restore the working cycle.

Searching mode

- When using multiprocess systems, memory searching can be accomplished on the first four processes only.
- When an interrupted cycle is restored automatically, the program is restarted from the beginning of the block where the interruption took place. When a cycle is restored from a pre-set block, it is restarted from the block next to the one searched.
- Automatically resuming the search, if the position of the start of the search is changed, for example, switching to **BLK/BLK** and using the arrow keys or setting the starting block only in the data entry **EXEC FROM:TO:**, the search starts from the new position and ends when the system has executed the number of blocks stored throughout the machining phase.
  The same thing happens when a search is carried out for a set block if the **TO** block is not specified.
- Automatic searching is only possible provided the relevant configuration (AMP) is abilitated. It can only be used when machining is executed at the beginning of the part program. Similarly, if during a working cycle MDI-blocks are executed, memory searching subsequently run on that part program would not work.
- When machine is switched off, modified data from **PROGRAM SET UP** or from other data entries are erased.
- When searching, variable allocation blocks are always executed.
- Searching is not operating properly when part program makes use of variables written or read from logic or from other environment.
- Following are listed the strings that can and can not be used within a part program in order to execute a memory searching:

  – **The following strings are accepted in searching mode:**

    DAN, IPB, UAO, UTO, UIO, SOL, DPA, PAE, PAD, MIR, ROT, SCF, AXO, LOA, RPT, ERP, CLS, PTH, EPP, EPB, GTO, IF, ELSE, ENDIF, GDV, RDV, UGS, CGS, DGS, DIS, DSB, GTA, ECM, PRO, PVS, GTP, CCP, SPA, SPF, SPP, CLP, TGL, OPN, WRT, REA, CLO, DEL, INS, PRO, RTP, ROP, GPS, PLS.

  – **The following string is accepted but non executed in searching mode:**

    DLY.

  – **The following strings are not accepted in searching mode:**

    RQO, RQT, RQP, TOU, DPP, UDA, SDA, UPR, UVP, UVC, TCP, REL, WOS, SND, WAI, EXE, DCC, FIL, SOP, GET, PUT, SCL, EPS, CON, COF, HON, HOF, RES, SMD, SAX, DIR, JOG, FHO,

Besides the following tracing cycles are not accepted:G72, G73, G74.

END OF CHAPTER

MACHINE PLOT

LAYOUT OF CHAPTER

Machine Plot is a utility permitting graphic display of the part profile; there are two different modes possible:

- With axes connected, during actual work on the part.
- With axes disconnected (Dry Run mode).

This chapter is divided into two parts; in the first part, operation of the utility is described in *axes connected mode*; the second part deals with operation with *axes disconnected* (Dry Run mode).

MACHINE PLOT USED WITH AXES CONNECTED

The utility may be activated manually with the softkeys, or by way of the manipulations described in this chapter. The tool path is displayed as the part program is executed.

It is also possible to activate *Machine Plot* directly from the part program using three-letter code UGS (Use of Graphic Scale).

When instruction UGS is programmed, the cartesian axes are displayed on the screen and the profile followed by the tool is visualised.

For details on this three-letter code and how to use it, see the 10 Series Programming Manual.

When the Machine Plot utility has already been activated through the softkeys, programming instruction UGS has no effect (the parameters set manually remain valid). On the other hand, if the utility is started from the part program, operation may be interrupted manually at any time and the parameters altered at will. In other words, the manual setting always has precedence.

User interface with axes connected

MACHINE PLOT provides a user interface that permits to configure and alter the tool path along the profile.

Different modes of display and enlargement are also offered:

- Two and three dimensional graphics.
- Orientation of visualised axes (for 2D graphics).
- Visualisation of the programmed and actual tool paths for an evaluattion of the variance between them.
- Visualisation of specific profile points on which the marker is positioned.
- Possibility to enlarge parts of the profile.
- Possibility to establish the sampling time (tick in milliseconds) in order to increase or lower the number of points visualised on the profile.

NOTE:
The smaller the tick, the higher the calculation time.

The user interface consists of a specific video page and a menu of softkeys, with which to configure and activate the features of the utility.

Machine Plot video page with axes connected

To access the MACHINE PLOT *axes connected* environment from the main menu, press the **AUTO** and then **MACHINE PLOT** softkeys in order. The data area of the MACHINE PLOT page is initially empty. When visualisation of the path is activated, the page layout will be as follows:

The data area provides general information that recurs on every video page; in addition, the following specific items are included:

Axes coordinates — While the system processes a given point this window displays the point coordinates for the axes configured in the data entry window opened with the **CONFIG** softkey.

Part program block — This line displays the part program block in execution.

Path display area — This area displays two cartesian axes whose orientation is configurable. They appear together with the following MACHINE PLOT configuration parameters:

- axes names
- minimum and maximum coordinates

Softkeys active in axes connected mode

The softkeys available in MACHINE PLOT environment, *axes connected* mode, have the following functions:

SOFTKEY	FUNCTION
CONFIG	Opens a data entry window that permits to set all the MACHINE PLOT parameters: visualisation mode, axes names, axes orientation, axes limits, sampling tick, etc.
PLOT	Starts graphic visualisation of the processed coordinates.
CHECK	Displays the last 16,000 points stored in the plotting buffer. If the configured visualisation mode is 2D graphics with programmed and actual tool path, the programmed path will be displayed in white and the actual path will be displayed in red.
CLEAR SCREEN	Clears the visualised path from the screen.
ZOOM	Permits to enlarge a desired portion of the path. The first time you press this key the system displays a window in which you can set the zoom scale. This window can be moved to the desired profile portion with the **[+]**and **[-]** keys. If you press **ZOOM** again the window will be enlarged to full screen size. The coordinates on the axes are the limits of the previous window.
DIMENSION	When you press this softkey, a cross shaped cursor appears in the centre of the screen and its coordinates are displayed in the top right corner. The arrow keys may be used to move the cross to any point of the profile. Each time an arrow key is pressed, there is a movement of one step; dimension of the step can be adjusted by pressing **[+]**and **[-]**.

SETTING MACHINE PLOT PARAMETERS WITH AXES CONNECTED

Before visualising a profile it is necessary to configure the MACHINE PLOT parameters. These parameters are listed in a 2 page data entry window accessible through the **CONFIG** softkey. The first page displays the following parameters:

MACHINE PLOT PARAMETERS (1 of 2)

- **Sampling tick (ms)** : **8**
- **Process number** : **1**
- **Horizontal axis name** : **X**
- **Vertical axis name** : **Y**
- **Third axis name** :
- **Axes orientation** : **1**
- **Plot type (1/2/3)** : **1**
  - **1 - 2D work position**
  - **2 - 2D work + machine position**
  - **3 - 3D work position**

CONFIG				HELP
PLOT	CHECK	CLEAR SCREEN	ZOOM	DIMENSION
				EXIT

The meaning of these fields is as follows:

Sampling tick

This is the sampling tick used by the system for reading profile points. It must be equal to or a multiple of the system tick and is expressed in milliseconds.

The allowed value is an integer from 2 to 65535.

If the selected value is too small (less than 8 milliseconds), the drawing time can be longer than the actual machining time.

Process number

Is the number of the visualised process. It may range from 1 and the highest number of process configurations.

Horizontal axis name

Is the name of the axis to be used as abscissa of the graphics. It normally coincides with the **X** axis but may be any other configured axis.

Allowed axes names are: **[X], [Y], [Z], [A], [B], [C], [D], [P], [Q], [W], [V], [U]** (capitals).

Vertical axis name

Is the name of the axis to be used as ordinate of the graphics. It normally coincides with the **Y** axis but may be any other configured axis.

Allowed axes names are: **[X], [Y], [Z], [A], [B], [C], [D], [P], [Q], [W], [V], [U]** (capitals).

Third axis name

Is the name of the third axis for 3D graphics. It normally coincides with the **Z** axis but may be any other configured axis.

Allowed axes names are: **[X], [Y], [Z], [A], [B], [C], [D], [P], [Q], [W], [V], [U]** (capitals).

Axes orientation

This parameter defines the orientation of the visualised axes. Allowed values are illustrated in the figure below:

Values from 1 to 4 select 2D path visualisation on the four quadrants of the plane. 5 selects 3D visualisation.

Plot type (1/2/3)

Defines the visualisation mode. Allowed options are as follows:

[1] Visualisation of the programmed path(*work*)

[2] Visualisation of the programmed path (*work*) and the actual path (*machine position*).

[3] 3 D visualisation.

If *Plot type* is **[2]**, both paths will be visualised only if the verification is enabled with the **CHECK** softkey. If plotting rather than verification has been enabled (by pressing **PLOT** rather than **CHECK**) only the programmed path will be visualised. Further information about graphics verification is provided later in this chapter.

With 3D visualisation, capture of point dimensions is not possible so the relative softkey (**ZOOM** and **DIMENSION**) is disabled.

IMPORTANT

When MACHINE PLOT is invoked from part program with the UGS code, *Plot Type* is always 1.

After all the parameters on this page have been set, press **[PgDn]** to display the second page.
The complete second page illustrated in the figure below appears in case of 3D visualisation. If 2D visualisation has been selected, angle parameters α and β will not be included in the list of parameters, and the icon showing a system of cartesian axes used for 3D axes orientation will not be displayed on the lower left corner of the screen.

Here is the complete set of MACHINE PLOT parameters displayed on page 2:

Min. abscissa (X)
This coordinate appears next to the axes origin. It is the minimum abscissa displayed in the graphic and is expressed in the unit configured for that process (mm or inches).
The allowed value ranges from -99999.99999 to +99999.99999.

Max. abscissa (X)
This coordinate appears on the right hand end of the abscissa. It is the maximum abscissa displayed in the graphic and is expressed in the unit configured for that process (mm or inches).
The allowed value ranges from -99999.99999 to +99999.99999.

Max. ordinate (Y)
This coordinate appears next to the axes origin. It is the ordinate abscissa displayed in the graphic and is expressed in the unit configured for that process (mm or inches).
The allowed value ranges from -99999.99999 to +99999.99999.

Max. ordinate (Y)	This coordinate appears on the upper end of the ordinate axis. It is the minimum ordinate displayed in the graphic and is expressed in the unit configured for that process (mm or inches). The allowed value ranges from -99999.99999 to +99999.99999.
Min. vertical coordinate (Z)	It is the minimum vertical coordinate displayed in the graphic and is expressed in the unit configured for that process (mm or inches). The allowed value ranges from -99999.99999 to +99999.99999.
Max. vertical coordinate (Z)	It is the maximum vertical coordinate displayed in the graphic and is expressed in the unit configured for that process (mm or inches). The allowed value ranges from -99999.99999 to +99999.99999.
α angle	It is the rotation to be applied to the horizontal plane in case of 3D visualisation. The horizontal plane is typically XY. The allowed value ranges from -359.99999 to +359.99999.
β angle	It is the rotation to be applied to the vertical plane in case of 3D visualisation. Vertical planes are typically XZ and YZ. The allowed value ranges from -359.99999 to +359.99999.

Prior to establishing maximum and minimum limits you must decide whether to visualise the whole of the profile or just part of it.

In the first case minimum and maximum limits must be greater than or equal to the minimum or maximum coordinates reached by the machining process.

After writing all of the parameters press **[Enter]** or **CONFIG** to close the window.

DRAWING A PROFILE WITH CONNECTED AXES

To draw the profile of a programmed machining process and currently in execution on the machine tool (axes connected mode), use this procedure:

1. Select and activate the desired program (see chapter 10).
2. Press **AUTO**.
3. If necessary, configure the MACHINE PLOT parameters as described in the above section.
4. Press **PLOT**.
5. Press **CYCLE START**: the system will start executing the program and drawing the graphic. The layout of the graphic screen is as shown in the figure on page 12-11:

The coordinates displayed on the upper right corner vary as the cycle proceeds. The line displayed above the ordinate axis shows the program block in execution.
To interrupt the drawing at any moment press **PLOT**. When the process stops the last executed part program block will be displayed.

The number of visualised points is inversely proportional to the *Sampling tick* configured in the MACHINE PLOT window: the smaller the tick the higher the number of points. The *Sampling tick* also affects the velocity at which the graphics is displayed. If the tick is tool small, the graphics may terminate after the cycle.
The ideal solution is to assign the same value to both cycle and MACHINE PLOT ticks. In certain simulations, however, it may be preferable to use different values.

Drawing a profile example

CHECKING THE GRAPHICS

When you press **PLOT** with *Plot type* 1 or 2 the system displays only the programmed path. To also visualise the actual machining path follow these steps:

1. Run an ordinary path visualisation with the procedure described in the previous section.

2. When the whole cycle or the part of the profile to be monitored is completed press **CHECK**.

The system will display the last 16,000 processed points stored in the plotting buffer or, if the actual number of processed points is less than 16,000, the complete profile that was executed before the interruption occurred.

With *Plot type* 2 the actual profile and the programmed profile will be displayed in different colours. However, both profiles tend to overlap, thus making it difficult to tell which points belong to each of them.

The programmed path will be displayed with white colour points while the actual profile will be displayed in red

Since a greater variance between both profiles can improve resolution but also generate a servo error, in this case it is advisable to use the Zoom option.

The *Sampling tick* affects the velocity and resolution of both graphics.

ZOOM

This extremely powerful option may be used to enlarge the selected portion of the profile, better illustrating the deviations between programmed and actual profiles. Successive zooms are also possible

The **ZOOM** softkey is enabled with *Plot type 1* and is used for graphic verifications with *Plot type 2.* When you press **ZOOM** the following window appears:

This window delimits the part of the profile that can be enlarged to full screen size. Press **[+]** or **[-]** to enlarge or reduce the window. **[+]** enlarges the window but reduces resolution, whereas **[-]** has the opposite effect.

To position the zoom window on other parts of the profile use the arrow keys.

After optimising the position and size of the window, press **ZOOM** again. The area delimited by the window will be enlarged to full screen size as illustrated in the figure below:

The coordinates displayed on each cartesian axis are the limits of the zoom window and permit calculation of size of the enlarged area.

Depending on the configured tick, the profile may be a continuous line or an array of points. The number of points in the profile also depends on the drawing velocity.

To further enlarge a portion of the zoomed profile, dimension and re-position the window and then press **ZOOM** again.

CAPTURING POINT COORDINATES

The visualised profile can be used for capturing the coordinates of its points. Greater accuracy can be obtained if points are captured from enlarged portions of the profile.

To capture a coordinate follow these steps:

1. Enlarge the profile area with **ZOOM.**
2. Press **DIMENSION**. A cross will appear that shows the position of the cursor; these coordinates are displayed in a window in the upper right corner of the screen. The layout of the page, in which the profile on the previous figure has been greatly zoomed, is as follows:

From the above figure, it may be noted that:

- the cross cursor and a window with the coordinates of its position are displayed.
- resolution is so high that the profile is displayed as a sequence of points rather than as a continuous line (the axes coordinates tell you that the enlarged area measures 1 mmx1.5 mm).
- deceleration on the bevel is shown by a greater concentration of points on the vertical line.
- we assume that the profile was zoomed for verification with *Plot type 2* mode and the points on the vertical line are displayed in two different colors.

To move the cross to any point in the profile press the arrow keys. The cross moves step by step. To increment and decrement the step press **[+]** and **[-]**. When you press **[+]** the step increases, and the marker moves at a faster pace but with less accuracy. By pressing **[-]** you can obtain the opposite effect.

MACHINE PLOT WITH AXES DISCONNECTED (*DRY RUN* MODE)

This second section of the chapter describes how to use Machine Plot with *axes disconnected* (Dry Run mode).

This feature is activated ny pressing softkey **DRY RUN** in the **AUTO** menu. With **DRY RUN** active, the part program is executed without any command being sent to the drives of the axes ('axes disconnected') and to the machine logic (I/O inhibited).

The graphic utility is activated in the same way, from both the part program (three-letter code UGS) and using the **MACHINE PLOT** softkey.

Configuration and functionality in use with axes disconnected

Even if used with **DRY RUN** active, some of the Machine Plot characteristics remain unaltered, others differ slightly, while others may be configured differently with the **CONFIG** softkey**.**

Main features of the utility used with *axes disconnected* are:

- Visualisation of movement entities rather than real points.
- Visualisation of the tool and offset in use.
- Diversification between movements executed in rapid (dashed line) and movements executed at the programmed rate (continuous line).
- Visualisation in 2D or 3D.
- Possibility of selecting orientation of the axes visualized.
- Visualisation of the coordinates of points visualised in the diagram by setting a graphic pointer on them.
- Possibility of performing zooms of sections of profiles.

Video page with axes disconnected (*Dry Run* Active)

To access the MACHINE PLOT *axes disconnected* environment from the system main menu, press the **AUTO**, **DRY RUN** and **MACHINE PLOT** softkeys in the order.

The **DRY RUN** softkey is a toggle key, the feature being active when the softkey appears in highlight colour.

The data area of the MACHINE PLOT page is initially empty. When visualisation of the path is activated, the page layout will be as follows:

The data area provides general information that recurs on every video page; in addition, the following specific items are included:

Axes coordinates — This window displays the point coordinates calculated in part program, referrred to the origin currently applied.

Part program block — This line displays the part program block in execution.

Path display area — This is the main area of the screen and displays two cartesian axes whose orientation is configurable.
The dimensions of the axes are referred to the origin which is applied at the time drawing the graph starts:

Softkeys active in axes disconnected mode (dry run)

The softkeys in the **MACHINE PLOT** environment when used in *dry run* mode perform the following functions:

SOFTKEY	FUNCTION
CONFIG	Opens an input window in which all the parameters available in *dry run* mode can be configured.
VERIFY	Permits visualisation of the path as the part program is being executed.
CLEAR SCREEN	Deletes the path drawn from the screen.
EXAME GRAPHIC	Opens a submenu, in which the retrace and analysis functions available for an executed path are available. The softkeys associated with the submenu are **REDRAW, ZOOM, CLEAR SCREEN, DIMENSION.**

SETTING THE MACHINE PLOT PARAMETERS WITH AXES DISCONNECTED

Before visualising a profile it is necessary to configure the MACHINE PLOT parameters. These parameters are listed in a 2 page data entry window accessible through the **CONFIG** softkey. The first page displays the following parameters:

Significance of the fields in this input window is as follows:

Horizontal axis name Is the name of the axis to be used as abscissa of the graphics. It normally coincides with the **X** axis but may be any other configured axis.

Allowed axes names are: **[X], [Y], [Z], [A], [B], [C], [D], [P], [Q], [W], [V], [U]** (capitals).

Vertical axis name Is the name of the axis to be used as ordinate of the graphics. It normally coincides with the **Y** axis but may be any other configured axis.

Allowed axes names are: **[X], [Y], [Z], [A], [B], [C], [D], [P], [Q], [W], [V], [U]** (capitals).

Third axis name Is the name of the third axis for 3D graphics. It normally coincides with the **Z** axis but may be any other configured axis.

Allowed axes names are: **[X], [Y], [Z], [A], [B], [C], [D], [P], [Q], [W], [V], [U]** (capitals).

Axes orientation This parameter defines the orientation of the visualised axes. Allowed values are illustrated in the figure below:

Block display (Y/N) This parameter enables (or disables) visualisation of the part program block currently being executed, in the window at the top left of the screen.

Rapid movement plot (Y/N) Enables/disables display in dashed lines of movements in G0 mode (rapid).
The default is **[Y]**.

Tool colour trace (Y/N) Enables/disables a trace colour change on each tool change. This functionality permits highlighting of the stretches of work performed with the same tool.
There are 14 different trace colours available, used in rotation on each tool change.
On programming of T0M6 (offload tool), the trace stays the same colour as before.

After all the parameters on this page have been set, press **[PgDn]** to display the second page.

The complete second page illustrated in the figure below appears in case of 3D visualisation. If 2D visualisation has been selected, angle parameters α and β will not be included in the list of parameters, and the icon showing a system of cartesian axes used for 3D axes orientation will not be displayed in the lower left corner of the screen.

NOTE:
This screen page is compiled in exactly the same way as described on page 12-5 in the section related to the *axes connected* mode.

VERIFY

The **VERIFY** softkey permits graphic verification of a profile during part program execution with axes disconnected.

This softkey is *toggle* type, each time it is pressed, it changes status, enabling/disabling drawing.

CLEAR SCREEN

Deletes the graphic drawn in the display area.

EXAME GRAPHIC

This softkey opens a submenu in which the executed path retrace and analysis functions are available.

The softkeys associated with this submenu are as follows:

		CLEAR SCREEN		HELP
REDRAW	ZOOM	DIMENSION		
				EXIT

REDRAW Redraws the profile graphic, in the original dimensions.

ZOOM Enlarges the selected portion of the graph; use of the **ZOOM** function is illustrated on page 12-11.

DIMENSION Makes possible capture of the coordinates of the single points of the graphic path; use of the **DIMENSION** function is illustrated on page 12-13.

CLEAR SCREEN Deletes the graph drawn in the display area.

END OF CHAPTER

DOS SHELL

DOS SHELL is a utility that permits to operate with the files resident in the hard disk, in the floppy disk or in a remote unit. Typical DOS SHELL operations are: file copy, file delete, creation of a directory, file print, etc.

DOS SHELL can manage any file recognised by the system. If the system has a *stand-alone* configuration, DOS SHELL operates only with local disks, i.e. disks present in the system. If the system is linked to a local area network, DOS SHELL operates both with local disks and with disks resident in the network host. DOS SHELL can access remote disk units and printers only if one of the following options is available: MINI-DNC, Ethernet or Serial.

DOS SHELL provides the user with a series of commands that permit to execute file management operations. These commands can be enabled by means of the softkeys in the DOS SHELL menu. They may request the user to input parameter values in specific data entry windows.

LAUNCHING DOS SHELL

To launch DOS SHELL press the **UTILITY** softkey from the main menu and then the **DOS SHELL** softkey. The DOS SHELL video page is shown in the figure below.

NOTE:
DOS SHELL can also be launched if the system has been started in emergency mode, i.e.by keeping F1 pressed during bootstrap.

DEFAULT PARAMETER VALUES

The data entry windows of the DOS SHELL commands are made up of fields, each of which shows the default value that corresponds to a given parameter. These values may be re-defined from the keyboard.

In some commands, such as **DELETE** or **MKDIR**, the parameter to be defined is a pathname. If no directory has been visualised with the **DIRECTORY** command, the default pathname is that of the last executed command of the same type. If no such command has been executed, there is no default pathname. If a directory has been visualised with the **DIRECTORY** command, the default pathname is that of the visualised directory and the default file is the one marked by the selection bar.

Some commands, such as **COPY** request to input two pathnames. If no directory has been visualised with the **DIRECTORY** command, the default pathnames will be those of the last executed command of the same type. If a directory has been visualised with the **DIRECTORY** command, the default source pathnames will be the last visualised ones.

The pathnames must be written in full. For example, if the following directory tree resides in the A: disk:

Correct pathnames will be:

A:\DIR 1\SUBD1|FIL1

A:\DIR1\SUBD2\FIL1

A:\DIR2\FILE

A:\FIL.DAT

The following pathnames are not allowed:

\DIR1\SUBD1\FIL1

A:DIR2\FILE

A:FIL.DAT

NOTE:
The notions of source and destination pathname are discussed in the **DIRECTORY** section.

DISPLAYING A DIRECTORY

The **DIRECTORY** command permits to visualise a page that provides information about the files stored in the specified directory as well as general information about the disk that accomodates them. Files are listed in alphabetical order and can be selected by moving the selection bar with the **[Up/Down Arrows]**. The selected file is also the current file and becomes the default source file for file management commands such as **COPY**. A typical DOS SHELL directory is shown in the picture below:

DOS SHELL **DATE: 15/01/94** **TIME: 9:35:00**

Disk has no label
Directory of e:\mydir

ATTR	NAME	EXT	LENGTH	DATE	TIME
			<DIR>	30/10/93	10:25
			<DIR>	30/10/93	14:05
_ _ _ _	AB	DOC	0002732	31/10/93	16:18
_ _ _ _	BACKG	DOC	0000805	31/10/93	16:21
_ _ _ _	CHKUSE	DOC	0003596	31/10/93	16:25
a _ _ _	DEC_CNFG	ASC	0011108	10/11/93	09:50
a _ _ _	DEC_CNFG	OTX	0008295	15/11/93	10:45
	DNC		<DIR>	17/11/93	11:17
a _ _ _	DOSSHELL	ASC	0032820	22/11/93	15:17
a _ _ _	DOSSHELL	DOC	0000536	02/12/93	10:32
a _ _ _	DOSSHELL	OTX	0024602	05/12/93	14:35
_ _ _ _	FTPN_FSP	ASC	0128126	06/12/93	11:34
_ _ _ _	FTPN_USE	ASC	0147361	06/12/93	16:12
a _ _ _	GLOSSARY	GLO	0006898	10/12/93	08:47
_ _ _ _	HUM_FUNE	DOC	0002730	10/12/93	10:22

Free space : 004354048 **Total files : 0023** **Total space : 026599424**

PRINT	COPY	RENAME	DELETE	HELP
MKDIR	DIRECTORY	XCOPY	EDIT	ABORT
FORMAT	SHOW DRIVE	BACKUP	RESTORE	EXIT

The file directory may occupy several pages. To visualise other pages, press these keys:

KEY	FUNCTION
PAGE↓	displays the subsequent page
PAGE↑	displays the previous page
HOME	displays the first page
END	displays the last page

NOTE:
If a help or data entry window is active, **[PgDn]** and **[PgUp]** do not permit to select other directory pages. You can use these keys for browsing through the help pages.

The meaning of the various fields and/or messages is as follows:

- General disk information

  Disk label is specifies the label associated to the disk.

  Disk has no label specifies that no label is associated to the selected disk.

  Directory of indicates the complete pathname of the visualised directory.

  Free space indicates the amount of free disk space in bytes.

  Total Files indicates the number of files or directories stored in the specified pathname.

  Total space indicates the total disk capacity in bytes.

- For each file the following informations are provided:

  ATTR file attributes:
  - **a** archive
  - **r** read only
  - **h** hidden
  - **s** system.

  NAME file name (max. 8 characters).

  EXT file extension (max 3 characters).

  LENGTH file length in bytes. If the name corresponds to a directory, the <DIR> string will be displayed in this field.

  DATE date of file creation or last alteration with the dd/mm/yy format.

  TIME time of file creation or last alteration with the hh:mm format.

DOS SHELL COMMANDS

This section describes the DOS SHELL commands in alphabetical order.

In the table that follows each command is listed under the relevant softkey name.

NAME	FUNCTION
ABORT	permits to abort a **COPY/PRINT** command.
BACKUP	permits to backup a file or a group of files on a diskette.
COPY	permits to duplicate the specified file.
DELETE	permits to delete a file or a directory.
DIRECTORY	visualises the contents of a directory.
EDIT	enables the line editor for files.
EXIT	permits to exit from DOS SHELL.
FORMAT	permits to format a diskette.
HELP	visualises help information in a window.
MKDIR	permits to create a directory.
PRINT	permits to print a file.
RENAME	permits to change the name of a file or a directory.
RESTORE	permits to restore a backup copy on the hard disk.
SHOW DRIVES	permits to visualise information about the configured disk drives.
XCOPY	permits to copy a tree of directories and subdirectories.

ABORT

The **ABORT** command forces the system to end the current **COPY/PRINT** session. The **ABORT** softkey is enabled when **COPY/PRINT** is active.

BACKUP

The **BACKUP** command permits to make a backup copy of a file or group of files on a diskette. This backup copy is fully compatible with the DOS 3.30a BACKUP format. **BACKUP** also permits to format the diskette during the copy.

BACKUP FILEs TO FLOPPY DISK

From: E:

Subdirectories ?: Y Format diskette ?: N

From specifies the pathname of the file/s to be backed up. Characters "*" and "?" are allowed.

Values: up to 65 alphanumeric characters.

Default: last pathname of which a **BACKUP** or a **RESTORE**. have been executed. No default appears for the first **BACKUP** command if it has not been preceded by a **RESTORE.**

Subdirectories request to make a backup copy of all the subdirectories in the specified directory.

Values: **[Y]** = Yes; **[N]** = No

Default: **[Y]**

Format diskette request to format the diskette during the backup session. The only format available with DOS SHELL is 1.44 MB HD..

Values: **[Y]** = Yes; **[N]** = No

Default: **[N]**

Characteristics:

If the pathname specified in the *From* field terminates by a directory and the *subdirectory* field is to **[Y]** this command will backup all the files in the directory plus all the files in all the subdirectories. If the pathname specified in the *From* field terminates by a file or a group of files and the *subdirectory* field is to **[Y]**, this command will backup all the specified files and all the files having similar characteristics stored in the subdirectories.

COPY

The **COPY** command permits to duplicate a file or group of files.

```
                    COPY FILE

From        : E:\MYDIR\DEC\CNFG.ASC
To          : a:\docum\*.*
```

From specifies the pathname of the file/s to be backed up. Characters "*" and "?" are allowed.

Values: up to 65 alphanumeric characters.

Default: last copied pathname or current file name. The current file is that selected during the last visualisation of the directory. No default name will be given during the first **COPY** unless it has been preceded by a **DIRECTORY** command.

To specifies the destination pathname to which the copy will be made. "*" and "?" are not allowed except in the "*.*" sequence.

Values: up to 65 alphanumeric characters.

Default: last copied pathname or current file name. The current file is that selected during the last visualisation of the directory. No default name will be given during the first **COPY** unless it has been preceded by a **DIRECTORY** command.

NOTE:
For further information about source and destination pathnames refer to the description of the **DIRECTORY** command.

Characteristics:

If the current file or the pathname specified in the *From* field is a subdirectory, all the files in the subdirectory will be duplicated.
When you press the **COPY** softkey, the **ABORT** softkey will also be enabled. The **ABORT** command permits to interrupt a copy at any point in the session. The interruption occurs when the **ABORT** request is given. Subsequent files will not be copied.

NOTE:
Do not use the **COPY** command to copy a part program between the floppy disk and the directory of part programs (E:\UPP). Such copy can be obtained with the **IMPORT/EXPORT** command available in the PP Management environment.

DELETE

The **DELETE** command permits to delete files or empty directories.

```
                    DELETE  FILE

Filename     :     E:\MYDIR\CHKUSE.DOC

File/Directory ?     :     F
Confirm delete ?     :     Y
```

Filename specifies the pathname of the file/files or the directory to be deleted. "*" and "?" are allowed only in filenames.

Values: up to 65 alphanumeric characters.

Default: name of the last deleted file/files or current file name. The currentl file is the file selected during the last visualisation of the directory. No default pathname is given for the first **DELETE** command unless it has been preceded by a **DIRECTORY** command.

File/Directory indicates whether a file or a directory is to be deleted. Select *File* to delete files in the current directory and *Directory* to delete the current directory (if empty).

Values: **[F]** = File; **[D]** = Directory

Default: **[F]**

Confirm delete confirmation command.

Values: **[Y]** = Yes; **[N]** = No

Default: **[N]**

Characteristics:

In the *File/Directory* field you must specify whether the pathname indicated in the *Filename* field refers to a file, a group of files or a directory.

Since only empty directories can be deleted with this command, make sure all the files and/or subdirectories have been deleted before making a request to delete a directory.

The current directory cannot be deleted even if it is empty.

DIRECTORY

The **DIRECTORY** command permits to visualise the contents of the specified directory.

```
                    DIRECTORY

Path1     :     E:\MYDIR
Path2     :
```

The data entry window for this command requests two pathnames. The first one (*path1*) is mandatory whereas the second (*path2*) can be omitted. When you exit this data entry window, the position of the cursor determines which directory will be displayed and which default pathname will be used in the fields of the subsequent commands.

path1 is the pathname of the directory, the file or the group of files to be displayed. "*" and "?" can be used for specifying a group of files. It is a mandatory field. If the cursor is on this field when you exit the data entry window, the relevant directory will be displayed and *path1* will become the default source pathname. Also, if the *path2* field is not empty, it will become the default destination pathname.

Values: max. 65 caratteri alphanumeric characters.

Default: last specified pathname. No default pathname will be displayed for the first **DIRECTORY** command.

path2 is the pathname of the directory, the file or the group of files to be displayed. "*" and "?" can be used for specifying a group of files. It is an optional field. If the cursor is on this field when you exit the data entry window, the relevant directory will be displayed and *path2* will become the default source pathname. Also, if the *path1* field is not empty, it will become the default destination pathname.

Values: max. 65 alphanumeric characters.

Default: last specified pathname. No default pathname will be displayed for the first **DIRECTORY** command.

EDIT

The **EDIT** command permits to invoke a subset of the part program Line Editor for writing or editing a text file.

LINE EDITOR

Filename : E:\MYDIR\BACKG.DOC

Filename specifies the pathname of the desired file. "*" and "?" are not allowed.

Value: max. 65 alpahnumeric characters.

Default: last opened file or current file. The current file is the file selected from the last directory visualisation. No default file is displayed for the first **EDIT** command unless it has been preceded by a **DIRECTORY** command.

NOTE:
For more information about the Line Editor, refer to the "Line Editor" chapter.

EXIT

The **EXIT** command permits to quit DOS SHELL.

FORMAT

The **FORMAT** command permits to format 720 Kb double density (DD) or 1.44 Mb high density (HD) 3.5" diskettes.

	FLOPPY DISK FORMAT	
System ? : N	**720K diskette ? : Y**	**Label ? : N**

System specifies whether the diskette to be formatted will be used for duplicating the DOS 3.30a system file.

Values: **[Y]** = Yes; **[N]** = No

Default: **[N]**

720K diskette specifies the diskette format (720 KB DD or 1.44 MB HD).

Values: **[Y]** = 720 KB; **[N]** = 1.44 MB

Default: **[N]**

Label permits to assign a label to the formatted diskette. The label will be requested by the system after formatting has been completed.

Values: **[Y]** = Yes; **[N]** = No

Default: **[N]**

HELP

The **HELP** command permits to display a window providing information about the current procedure. A Help window may be invoked or closed at any moment.

Help information may be general or context-sensitive. At the general level it describes the functions of the softkeys available in the current menu. Context-sensitive information provides the meaning and allowed values of the parameters listed in the current data entry window.

Characteristics:

Context-sensitive information always refers to the field on which the cursor is positioned. When the cursor is moved to another field, the contents of the help window are automatically updated.

By enabling/disabling the data entry window it is possible to automatically shift between the context-sensitive and the general level.

To browse through a help window made up of several pages press **[PgDn]** and **[PgUp]**. To disable the help window, press **HELP**.

MKDIR

The **MKDIR** command permits to create a directory or a subdirectory in the selected disk drive.

MAKE DIRECTORY

Path : e:\mydir\newdir

Path specifies the pathname of the directory or the subdirectory. "*" and "?" are not allowed. A subdirectory must necessarily be included in an existing directory.

Values: max. 65 alphanumeric characters.

Default: last specified pathname for directory creation. No default pathname is displayed for the first **MKDIR** command unless it has been preceded by a **DIRECTORY** command.

PRINT

The Print command permits to print text files using local or remote printers.

```
                    PRINT FILE
Filename    :   e:\mydir\newdir
Printer     :   LPT1
```

Filename specifies the file or the group of files to be printed. "*" and "?" are allowed.

Values: max. 65 alphanumeric characters.

Default: last printed file or current file. No default file will be displayed for the first **PRINT** command unless it has been preceded by a **DIRECTORY** command.

Printer specifies the configured output printer (refer to the SHOW DRIVES command in this chapter), which may be remote or local.

Characteristics:

If a remote print out has been requested but no remote printer is connected to the host, the system will give no error message.

RENAME

The **RENAME** command permits to alter the name of a file or a directory. A file can be renamed only in the directory to which it belongs.

RENAME FILE

Old name : e:\mydir\CHKUSE.DOC

New name : newfile.doc

Old name specifies the path of the file or the directory to be renamed. "*" and "?" are allowed.

Values: max. 65 alphanumeric characters.

Default: last renamed file or current file. The current file is the one selected from the last directory visualisation. No default name is displayed for the first **RENAME** command unless it has been preceded by a **DIRECTORY** command.

New name specifies the new name to be assigned to the specified file/directory. "*" and "?" are not allowed.

Values: max. 12 alphanumeric characters

Default none.

RESTORE

The **RESTORE** command permits to retrieved file that have been previously backed up.

```
RESTORE FILEs FROM FLOPPY DISK

To              :   e:\mydir\newdir
Subdirectories ?    :   Y   Confirmation for changed files ? :   N
```

To specifies the pathname of the directory in which the back up copy will be restored. "*" and "?" are not allowed.

Subdirectories permits to restore all the subroutine branches with their files.

Values: **[Y]** = Yes; **[N]** = No

Default: **[Y]**

Confirmation for changed files specifies whether the system will send a confirmation request before restoring those files that have been modified after the back up. Enter **[Y]** to display such a request.

Values: **[Y]** = Yes; **[N]**=No

Default: **[N]**

Characteristics:

The system prompts the operator to insert the diskettes in the correct sequence. If one of the diskettes is missing, the restore session cannot be completed.

NOTE:
This command can restore backup copies produced with the system Backup function or with a format compatible with DOS 3.30a **BACKUP**.

SHOW DRIVES

The **SHOW DRIVES** command permits to visualise information on the disk drives accessible to DOS SHELL. When a **SHOW DRIVES** command is given, the last visualised directory is cleared from the screen but all the selected files and source and destination pathnames will remain active.

```
DRIVE        HOST PATH

A          :  Local
E          :  Local
K          :  \\server\diskC
LPT1       :  Local
```

For each accessible drive **SHOW DRIVES** visualises a status window that provides the following information:

DRIVE is a letter from A to Z that identifies the drive name. It must be quoted in the pathnames associated to commands.

HOST PATH is a string that specifies whether the drive is local or remote. It may be:

local for a local disk

*nomehost**service* for a remote disk, where

nomehost is the logic name of the remote disk in the local network

service is the name of the shared remote disk drive or directory in the local network.

NOTES:

- A remote printer is that connected to the host.
- A remote disk is the one resident in the hard disk of the local network host. Within the network the host is identified through a logic name. Each drive shared by the systems connected over the local network is associated to access permits that establish which operations are allowed to each local system. If the command given by a networked local system is not allowed, (for example, 'copy to a read-only remote disk') an error message will be displayed.
- The current networked drives are highlighted on the screen. To connect new drives, use the MINI DNC, Ethernet or Serial option.

XCOPY

The **XCOPY** command permits to copy a directory tree complete with all its branches to any disk drive accessible to DOS SHELL.

```
                    COPY DIRECTORY TREE

From              : E:\MYDIR\*
To                : a:\docum\*.*
Subdirectories? :   Y   Empty subdirs?:  N      Confirmation for each file?: N
```

From specifies the pathname of the source directory tree. "*" and "?" are not allowed.

Values: max. 65 alphanumeric characters.

Default: last directory pathname or current file/pathname. The current file is that selected from the last directory visualisation. No default pathnames are given for the first **XCOPY** command unless it has been preceded by a **DIRECTORY** command.

To specifies the destination pathname for the copy. "*" and "?" are not allowed except for the "*.*" string.

Values: max. 65 alphanumeric characters.

Default: last destination pathname or destination pathname specified in the last **DIRECTORY** command. No default name will be given for the first **XCOPY** command unless it has been preceded by a **DIRECTORY** command.

Subdirectories specifies whether the subtree of the directory written in the *from* field or just the files stored in in must be copied.

Values: **[Y]** = Yes; **[N]** = No

Default: **[Y]**

Empty subdirs specifies whether the existing empty directories must be copied.

Values: **[Y]** = Yes; **[N]** = No

Default: **[N]**

Confirmation for each file specifies whether the system will send a confirmation request before copying a file. Enter Y to display such a request.

Values: **[Y]** = Yes; **[N]** = No

Default: **[N]**

Characteristics:

If the pathname specified in the *From* file ends by a directory, all the files in this directory will be copied. If the subdirectories that make up the tree do not exist in the source drive, **XCOPY** will create them.

NOTE:
Do not use the **XCOPY** command to copy a part program between the floppy disk and the directory of part programs (E:\UPP). Such copy can be obtained with the **IMPORT/EXPORT** command available in the PP Management environment.

END OF CHAPTER

SECURITY

10 Series CNC provides a set of security functions that restrict access to the system resources. These functions are assigned to system users organised hierarchically according to three main criteria:

- User levels
- Ability to use system functions
- Ability to install optional components

USER LEVELS

System users are distributed in seven hierarchical levels, each of which is assigned a specific set of functions. User levels and their functions are shown in the table below:

LEVEL	USER	FUNCTIONS
0	R&D	reserved
1	*Osai*	reserved
2	APPLICATIONS	application development and machine tool integration
3	OEM/SERVICE	alteration of machine parameters, customer support
4	END_USER	installation of options
5	END_USER_TEC	development of technological programs, tool management, etc.
6	OPERATOR	machine tool operation

Levels 0 and 1 are reserved for *Osai* and cannot be accessed by users not belonging to *Osai*. Users in levels 2 through 6 do not belong to *Osai* but their ability to access the system resources is restricted by a password.

A password is a string of 20 alphanumeric characters that can be defined and altered by users located at different hierarchical levels. By defining his own password a user bars access to the system by users at lower levels.

IMPORTANT

At power up the default user level is 6. No password is needed in order to access this level.

ABILITY TO USE SYSTEM FUNCTIONS

Each user has the ability to use the set of system functions typical of his level. The number and type of functions assigned to one level can be defined by the user who occupies the lower level and has greater hierarchical importance.

There is a specific procedure that permits to enable the user occupying one level to use a system function.

System functions are grouped in ten classes. Each class may include as many as eight functions:

0 reserved
1 reserved
2 maintenance
3 directory protection
4 system tables protection
5 configuration
6 PLUS execution
7 part program development
8 execution of CNC functions
9 reserved for the OEM

Each class of functions is described by a byte whose bits correspond to the functions in the class. A function is enabled when its bit is to 1 an is disabled when its bit is to 0. The ten bytes that describe the functional classes and the meaning of each bit are listed in Appendix B.

The ability to access the functions in a given level can be pictured as a bit map.

When a release is installed for the first time, all the levels from 3 to 6 have the same bit map, i.e. 10 enable bits:

0	00000000	reserved
1	00000000	reserved
2	01111111	maintenance
3	10001110	directory protection
4	00001111	system tables protection
5	11110111	configuration
6	00011111	PLUS execution
7	00001111	part program development
8	00000111	execution of CNC functions
9	11111111	reserved for the OEM

ABILITY TO INSTALL OPTIONS

Options purchased separately can be installed in the system at any time as long as the user is granted the ability to install them. At *Osai* there are two ways of granting users permission to install an option:

- providing the user with a control specific hardware key generated by *Osai* during system production. The 'system' is a unit made up of a control plus its optional components.
- providing the user with a product key that permits him to install an option purchased after the control.

ENABLING THE SECURITY ENVIRONMENT

The Security environment can be enabled by turning the control on in emergency status. It can also be enabled when other processes are running on the system.

When the system is powered on in emergency status, i.e. keeping **[F1]** pressed, the following submenu appears:

AMP	PLUS	VARIAB RESET	SYSTEM HISTORY	HELP
DOS SHELL	SECURITY	TABLE EDITOR	PERIPHERALS	LANGUAGES
REBOOT SYSTEM	COMPILER		UTILITY	SYSTEM

To enable the Security when a process is running on the system, press the **DIAGNOSTICS** softkey from the main softkey menu. In this case, the following submenu appears:

				HELP
SECURITY	SYSTEM HISTORY		SERVO MONITOR	

From either submenu press the **SECURITY** softkey. The following page is displayed:

SYSTEM SECURITY LEVEL : xx DATE: 10/01/94 TIME: 9:35:00

10 Series

Code number : xxxxxx

Manufactured : xx/xx/xxxx

				HELP
SET LEVEL	SET PASSWORD	SET SECURITY	SHOW SECURITY	PRODUCT INFO
PRODUCT KEY	INSTALL	SHOW OPTIONS	SHOW FIXUPS	EXIT

The sections that follow describe the softkeys available in the SECURITY environment.

SET LEVEL

This softkey permits to select a user level. When **SET LEVEL** is pressed the following data entry window is displayed:

```
SET SECURITY LEVEL

LEVEL          :

PASSWORD       :
```

level is the user level.

password is the password associated to the specified level. It is mandatory in order to access a user level lower than the current one. No password is necessary if the password for the selected level has not been defined (refer to the SET PASSWORD section).

The password does not appear on the display when user writes it on the field. If the password is not correct an error message will be displayed.

To abort the operation press **[Esc]**. The system will return to the previous user level.

The ability to use the functions associated to a given user level remains active until a higher user level is selected.

The user level is not altered if the system is reinitialised. This facilitates repetitive operations that require frequent reinitialisation, such as development of the system logic.

IMPORTANT

When a user selects a user level, he is enabled to use the functions associated to both his level and the subsequent ones.

SET PASSWORD

This softkey permits to alter both the current password and the passwords assigned to upper levels. It also permits to define a password for those levels that do not have any. When **SET PASSWORD** is pressed the following data entry window appears:

```
NEW PASSWORD

level            :
old password     :
new password     :
confirm          :
```

level is the user level whose password is to be altered. The default level is that selected with **SET LEVEL**. Allowed values must be greater than the current level.

old password is the password displayed on the *level* field. If no password has been defined for the selected level, press **RETURN** to skip this field.

new password is the new password to be assigned to the *level* field.

confirm key in the new password on this field for confirmation.

The assigned password is not displayed in the data entry window. To abort the procedure without altering the old password press **[Esc]**.

Before starting this procedure the user must declare his own level by means of the **SET LEVEL** softkey. After accessing his own level the user is in a position to alter the passwords of both his own and the higher levels.

SET SECURITY

This softkey permits to alter the ability to use system functions by users having a higher hierarchical level.

Before starting this procedure the user must declare his own level. When the **SET SECURITY** keyboard is pressed the following data entry window appears:

SET SECURITY PROTECTIONS

selected level:

selected level is the user level whose ability to use system functions must be modified. It must be higher than the current level.

Key in the level and press **[Enter]** to confirm it. The following bit map will be displayed:

SET SECURITY PROTECTIONS
Selected level: .....

byte:	default bit 7 ... 0	configured bit 7 ... 0
0:	xxxxxxxx	xxxxxxxx
1:	xxxxxxxx	xxxxxxxx
2:	xxxxxxxx	xxxxxxxx
3:	xxxxxxxx	xxxxxxxx
4:	xxxxxxxx	xxxxxxxx
5:	xxxxxxxx	xxxxxxxx
6:	xxxxxxxx	xxxxxxxx
7:	xxxxxxxx	xxxxxxxx
8:	xxxxxxxx	xxxxxxxx
9:	xxxxxxxx	xxxxxxxx

code is the code of the class of functions. It is also the number of the protection tables listed in Appendix B.

default this 8-bit column shows the enabled functions that may be accessed by the upper user level in each class. Bits can be written as 1 or 0.

configured this 8-bit column shows the enabled functions that may be accessed by the current user level in each class. Bits can be written as 1 or 0.

A bit to 1 marks an enabled function. A bit to 0 marks a disabled function. The user can grant access to the functions that are enabled in the *default* column.

The system filters the alterations made by the user to prevent him from making illegal modifications. In particular, the system:

- Forces to 0 the functions that are disabled in the *default* column.
- Forces to 0 on higher levels the functions that are disabled on the current level.

On-line help describing each bit (function) is provided for every class of functions.

To abort the procedure at any time without altering the current bit map press **[Esc].**

Example:
If you are a level 3 user and want to alter the access to the system by a user on level 4, follow these steps:

1. Press **SET LEVEL** to declare your user level. At the system prompt key in your level and password.
2. Press **SET SECURITY.** The system displays a data entry window with the user level (4) in the *selected level* field. When you confirm this level, the bit maps of levels 3 (current) and 4 will be displayed.
3. Let's assume that access to class 4 functions is to be modified and that the following bit map is displayed for class 4 when you press **SET SECURITY**:

   default configured
   00001111 00001110

4. Key in the modifications on the *configured* column:

   default configured
   00001111 00011011

   Press **[Enter]** to confirm them and close the data entry window.

5. To check whether the system has accepted the alterations press **SET SECURITY** again, key in your level (3) and confirm it. The following bit map will be displayed:

   default configured
   00001111 00001011 *(originally the altered map was 00011011)*

   Note that the system has filtered the following modifications:

   - The disabled function associated to bit 3 has also been disabled for level 4 (xxxxx0xx).
   - The function associated to bit 1 has been re-enabled (xxxxxxx1). This occurs because the bit is also enabled in the *default* map (level 3). However, the bit is not enabled on higher levels. To enable the bit also on higher levels, press **SET SECURITY.**
   - The request to enable the fifth bit (xxx1xxxx) has been denied because this bit is disabled in the *default* map.
   - The bit maps of the lower levels have not been modified.

SHOW SECURITY

Permits to display the bit map associated to the current level specified with **SET LEVEL**. When you press **SHOW SECURITY** the following window appears:

SECURITY PROTECTIONS STATUS

byte	default bit 7 ... 0	configured bit 7 ... 0
0:	xxxxxxxx	xxxxxxxx
1:	xxxxxxxx	xxxxxxxx
2:	xxxxxxxx	xxxxxxxx
3:	xxxxxxxx	xxxxxxxx
4:	xxxxxxxx	xxxxxxxx
5:	xxxxxxxx	xxxxxxxx
6:	xxxxxxxx	xxxxxxxx
7:	xxxxxxxx	xxxxxxxx
8:	xxxxxxxx	xxxxxxxx
9:	xxxxxxxx	xxxxxxxx

The *configured* column shows the bit map of the current level, whereas the *default* column shows the bit map of the higher level. These bit maps cannot be altered.

PRODUCT INFO

This softkey permits to visualise the configuration, release number and options of the 10 Series CNC. It is available only if accessible to the current user level.

PRODUCT KEYS

This softkey permits to enable a user to install optional components on the 10 Series CNC. It is available only if it is accessible to the current user level and the system is in emergency status.

When your press **PRODUCT KEY** the following data entry window is visualised:

```
PRODUCT AUTHORIZATION KEYS

Key 1        :
Key 2        :
Key 3        :
```

Key1 . . . key3 are three alphanumeric product keys. Each product key can be up to 20 characters long and is supplied in conjunction with the optional component.

Enter the product keys in the displayed sequence and press **[Enter]** to confirm it. If the product key is correct the system enables the user to install the option.

INSTALL

This softkey permits to install software options on the 10 Series CNC. It is available only if it is accessible to the current user level and the system is in emergency status.

To install optional software use this procedure:

1. Turn on the system in Emergency status (i.e. keeping **[F1]**. depressed). The following menu will be displayed:

AMP	PLUS	VARIAB RESET	SYSTEM HISTORY	HELP
DOS SHELL	SECURITY		PHERIPHERALS	LANGUAGES
REBOOT SYSTEM	COMPILER			SYSTEM

2. Press the **SECURITY** softkey.
3. Press the **SET LEVEL** softkey to define the user level enabled for software installation (refer to the description of the **SET LEVEL** softkey).
4. Press the **PRODUCT KEY** softkey and key in the product key (refer to the description of the **PRODUCT KEY** softkey).
5. Press the **INSTALL** softkey. The system displays a prompt to insert the first option diskette.
6. Insert the diskette and press **[Enter]**. The system prompts the user to select a language and displays general information about the option.
7. Follow the installation instructions supplied by the program. After installation has been completed the system returns to the main SECURITY menu.
8. Press the **EXIT** softkey to close the SECURITY environment.
9. Remove the diskette from the drive and reboot the system either by pressing **REBOOT** or by switching it off and on.

SHOW OPTION and SHOW FIXUPS

SHOW OPTION permits to visualise the list of files with the xxxxxxxx.0nn extension, which store information about the optional software installed in the system.
SHOW FIXUPS permits to visualise the list of files with the yyyyyyyy.Fmm extension, which store information about the optional software fixups (i.e. updates) installed in the system.
To display the information lines of a file in any of the lists position the cursor on the file name and press **[Enter].**

Example:

a. Press **SHOW OPTION** to display the software options installed in the system:

If the list is made up of several pages, use **[PgUp]** and **[PgDn]** to display previous or subsequent pages.

b. Position the cursor on the desired option and press **[Enter]**.

c. The information lines in the installed option (E390202.001) will be displayed. This option bears the following code: E39 Rel. 2.02.

If the list is made up of several pages, use **[PgUp]** and **[PgDn]** to display previous or subsequent pages.

d. Press **[Enter]** to close the second data entry window and **[Esc]** to close both of them.

e. Restart the procedure to visualise the contents of other files or press **[Enter]** again to close the 'INSTALLED OPTIONS' data entry window.

END OF CHAPTER

PERIPHERALS

PERIPHERALS is a utility program which allows to access the configuration of network-connected systems and relevant devices.

This environment allows the following procedures:

- configuring and enabling Interlnk;
- configuring the output device for the printer;
- configuring the remote switching-on;
- enabling the network configuration;
- connecting the remote environment and services (by means of network configuration);
- configure the Serial MiniDNC for Windows option

This chapter will deal with the first two procedures while for remote switching-on and network configuration please see the CHAPTER "Ethernet Comunication" in this manual.

ACTIVATING PERIPHERALS

To activate peripherals press the UTILITY softkey and subsequently the PERIPHERALS softkey on the viewed menu The following display will then come up.

NOTE:
PERIPHERALS can be activated even if the system is started in emergency mode (during bootstrap press **[F1]** key).

INTERLNK

INTERLNK is a utility program supplied with the MS-DOS 6.22 operating system which allows to connect the 10 Series system with another computer (or another 10 Series system) using a serial port.
From one system it is possible to access data owned by the other system without using floppy disks to copy files.

The system consists of a Client which sends commands out and a Server which is the unit connected to the Client.
The Client makes use of the Server's disks while the Server displays the connection status between the two systems and the comparison between local and remote disks.
When the systems are connected to Interlnk, Server's disks are displayed on the Client as additional disks.

Configuration

To be able to use Interlnk it is necessary to have:

- a free serial port for each system;
- a 3-wired serial cable.

Interlnk must be configured on the 10 Series system and on the unit to be connected (personal computer or 10 Series system).

Interlnk Configuration for 10 Series System

To activate Interlnk configurator, press the **INTERLNK CFG** softkey on the PERIPHERALS video panel. The following input window will be displayed:

INTERLNK CONFIGURATOR

ACTIVE PORT: **COM1**

SELECTED PORT: **COM1**

ACTIVE PORT is the serial port used at present by Interlnk.

SELECTED PORT is the name of the serial port to be activated and used by Interlnk every time the 10 Series System is operated

The serial ports to be selected are:

COM1 - serial port
COM2 - serial port

In case no serial port is to be used, enter **NULL**.

When the system will be subsequently operated, ACTIVE PORT will be equal to SELECTED PORT.

Make sure that the serial port used by Interlnk does not come into conflict with the serial ports defined in AMP (Hardware Configuration) or in other options (serial MINI DNC, Remote Diagnostic); should the same port be defined both in AMP and in Interlnk Cfg., when pressing the INTERLNK or the INTERSVR softkey to operate the system, a display indicates that the device is not available for use.

NOTE:
In case of remote access (read and/or write in) with Interlnk, the 10 Series System **must** be IDLE; besides it is advisable to overload the system as little as possible to avoid affecting the current communication.

INTERLNK (Client)

To start the Client, press the **INTERLNK** softkey on the PERIPHERALS menu. The status of the local disks (and also of the remote disks if the Server has been started on the other system) will be displayed giving the prompt for the DOS commands.

Should the Server not being activated on the other system, a message will be displayed to check whether the cable on the selected serial port has been correctly connected and Intersrv is already operating on the other system.
At the command prompt type in **Interlnk**, in order to check the status of the disks and of the selected serial port.
Example: when Intersvr has been loaded on the computer and Interlnk on the 10 Series System through the serial port 1, the following information will be displayed:

Port=Com(3F8)

This computer (Client)		Other Computer (Server)		
G:	equals	A:		
H:	equals	C:	(536Mb)	
I:	equals	D:	(6Mb)	MS_RAMDRIVE

The A, C e D remote disks of the computer Server can be accessed from 10 Series system at the command prompt supplied by Interlnk using the identifiers G, H and I.

NOTE:
Re-direction of these disks is NOT visible inside the DOS SHELL program.

Once the connection to the Server is established, it is possible to use the following MS-DOS commands:

CHKDKS	FORMAT	DEFRAG	MIRROR
DISCOMP	SYS	DISKCOPY	UNDELETE
FDISK	UNFORMAT	SCANDISK	

Interlnk Client can be used from the 10 Series System both when it is in emergency mode (pressing the **[F1]** key at the bootstrap) and when in running mode.
To exit Interlnk, it is necessary to type in **EXIT** at the DOS command prompt on the first Interlnk video page thus returning to the PERIPHERALS video page.

INTERSVR (Server)

Pressing the **INTERSVR** softkey from the PERIPHERALS menu, Interlnk displays the status of the local disks; a status bar at the bottom of the screen displays the connection status indicating the serial port in use, the transfer rate (baud rate) and whether the Client is reading or writing in.
In case Interlnk has already been started on the other system (Client), the re-direction of the remote disks, i.e. the ID to access the remote disks of the Server from the Client is displayed; when the Client accesses a disk from the Server, an asterisk (*) next to the relevant ID is shown while the status bar indicates the type of access (read or write in).

NOTE:

Intersvr could be activeted on 10 Series System only in Emergency mode (pressing **[F1]** at bootstrap) ; in other cases the intersvr function is disabled.

When executing Itersvr on the 10 Series System, the following video page will be displayed:

Microsoft Interlnk Version 1.00

THIS COMPUTER (SERVER)	OTHER COMPUTER (CLIENT)
A: C: (10Mb) D: (23Mb) E: (16Mb) F: (298Mb) LPT1:	

TRANSFER | **PORT =** **SPEED =** | **ALT+F4 = EXIT**

When connecting the other system (Client) to the 10 Series System through the serial port 1, the re-direction of the Server remote disks will be also displayed; the page will be modified as follows:

Microsoft InterInk Version 1.00

THIS COMPUTER (SERVER)	OTHER COMPUTER (CLIENT)
A:	Equals E:
C: (10Mb)	Equals F:
D: (23Mb)	Equals G:
E: (16Mb)	Equals H:
F: (298Mb)	Equals I:
LPT1:	Not connected

TRANSFER | **PORT =** **SPEED =** | **ALT+F4 = EXIT**

NOTE:
Intersvr can be activated on the 10 Series System only when this is in emergency mode (i.e. after pressing F1 at the bootstrap); when the system is in running mode, Intersvr is not available (the relevant softkey is disabled).

InterInk Configuration Personal Computer

InterInk can be activated both as a Client and a Server on a Personal Computer connected to the 10 Series System.

INTERLNK (Client Configuration)

To activate InterInk on a Personal Computer it is necessary:

- to ensure that **INTERLNK.EXE** file is on the PC hard disk;
- to type the following command line in the file CONFIG.SYS:

  device=c\dos\INTERLNK.EXE/drives=6
- Presuming that the INTERLNK.EXE file is in the C:\DOS directory and you want to redirect 6 instead of 3 default disks, proceed as follows:
- start the computer again;
- when started InterInk displays the status of the redirected disks and the port used; anyhow such information are displayed when typing in InterInk at the prompt of MS-DOS commands.

Once the Server has been activated on the 10 Series System, it is possible to access the remote disks of the Server.

NOTE:
InterInk can be executed from the DOS shell of Microsoft Windows 3.1 or 3.11. Initiating the Client from the DOS shell of Windows 95, the system indicates that it is exiting Windows and starting DOS; when the InterInk session is ended, the computer is restarted typing **[EXIT]** in.

INTERSVR (Server Configuration)

To activate the InterInk Server it is not necessary to modify the CONFIG.SYS file; just type in **INTERSVR** at the prompt of the DOS commands.
As in the 10 Series version, InterInk displays the status of the local disks; a status bar at the bottom of the screen displays the status of connection and the serial port used by the Server, the transfer rate (baud rate) and whether the Client is reading or writing in.
In case InterInk has already been activated on the Client, the re-direction of the remote disks is displayed, i.e. the ID's to access from the 10 Series to the remote disks of the Server; when the 10 Series access a disk from the Server, an asterisk (*) next to the relevant ID is displayed.

NOTE:
Intersvr can be executed from the MS-DOS shell of Microsoft Windows 3.1 or 3.11, however it is no longer possible to go from this shell to the tasks of Microsoft Windows **[ALT][TAB]** is disabled). Starting the Client from the DOS shell of Windows 95, the system indicates that it is exiting Windows and starting DOS; when the Intersvr session is ended, the computer is restarted pressing **[ALT][F4].**

For further information about InterInk.exe, InterInk and Intersvr refer to the MS-DOS guide or to the PC-Help Guide where the MS-DOS 6.22 or subsequent versions are installed.

PRINT DEVICE

PRINT DEVICE allows to define the correct device to print locally.

To define the port to be used for printing, press the **PRINT DEVICE** softkey from the PERIPHERALS video panel. The following input window will be displayed:

LOCAL PRINT DEVICE

Enter one of the following names:

COM1	**Serial Port 1**
COM2	**Serial Port 2**
LPT1	**Parallel Port 1**
NULL	**None**

The ports to be selected are as follows:

COM1 - serial port 1

COM2 - serial port 2

LPT1 - parallel port 1

In case no serial port is to be used, enter **NULL**.
When starting the system next, the configuration of the local port for the printer will be activated.

The herewith defined print port is used as local default port in some software packages having the **PRINT**.

REMOTE BOOT

This softkey allows to enable or disable the connection to a remote controller of the CNC bootstrap.

This function is available only when the CNC and the remote controller (Personal Computer) are connected through the MINI DNC option and when the Remote Development kit option is installed on the PC.
The softkey gives access to a data entry requiring two types of information:

REMOTE BOOTSTRAP

Remote Bootstrap	**[Y/N]:**	**Y**
Syncronized	**[Y/N]:**	**Y**

The first field allows to enable or disable the connection to the remote PC.

The second field indicates if during the bootstrap the PC has to wait for the connection to the remote PC (synchronized=**[Y]**) or whether it can perform the bootstrap even without such connection (synchronized=**[N]**)

For a more detailed description of this function, refer to the CNC 10 Series Remote Development kit manual.

END OF CHAPTER

ETHERNET COMUNICATION

INTRODUCTION

The communication channels of the CNC's have also followed the PC evolution. Today Ethernet communication is a de-facto standard in factories. As RS 232 Ethernet defines an electric communication standard but with the advantage that the transmission speeds are much higher: instead of the 9600 bps for RS 2323 we have 10 million bps with Ethernet.

Therefore the users have increasingly been asking to connect also the CNC's to the factory's Ethernet, in order to be able to easily exchange programs from the technical department and production data between the other departments involved i.e. to exchange data between the CNC's and the PC's.

As the OSAI CNC's are based on standard PC architecture we were able to integrate the Ethernet communication channel directly into the CNC thus implementing interesting features which go beyond the original user requests.

When defining the specifications of network, OSAI adopted the **standard NetBEUI communications protocol** for the 10 Series controllers which is available on all Windows PC's connected to the network and implemented the **Client-Server architecture** on the CNC. Therefore you can share the CNC's hard disks on the network and access the shared network devices on the CNC. In other words, from the CNC the operator can access the program archives or tool data on a central server or the technical department can transfer technology programs and machining data from their PC's to a CNC by simply dragging the files with the mouse from the local hard disk to the shared one on the remote CNC. And that's not all. In addition to the program transfer feature the 10 Series CNC offer an **extensive functions library for Windows (DLL)** and the relative extended command set on NetBEUI level. This allows not only to acquire production data from the production planning PC, but also to synchronise several CNC's connected to the network via a cell or line supervisor.
Therefore it can be safely stated that OSAI has exceeded the requests for connecting CNC's to the factory network ***by delivering solutions for building a distributed architecture of CNC's integrated in the factory network*** as native components and not as strange units interfaced to the net via additional expensive and restricting hardware modules.

Connection of a numerical control Client to a Server PC offers access to the disks and to the directories of the Server itself with the following advantages for the user:

1. Using files made available by the server as if they were local files installed on the machine. The transfer of files from the server to the CNC is launched by the usual “FILE COPY” commands.

   Copying files stored on the hard disks of the server to the local hard disk presents an advantage, as you can maintain a centralised archive on the server.

   The ability to copy the files memorized in the Server's hard disk to the local hard disk can be very useful as the part programs are all contained in a central archive on the server

2. Executing technology programs residing on a dedicated remote server

   WARNING

   In this case the maximum transfer rate of the Ethernet has to be carefully evaluated so that the execution times and machining quality are not jeopardised.

   By using the network you can directly control from the PC the transfer of the various types of file. This is very useful especially when various 10 Series Systems are connected to a single PC Supervisor; further, if the *remote IPC* parameter (InterProcess Communication) is set "Y", messages and commands can be exchanged between tasks that are resident on different machines.

Communication works at a transmission speed of 10 Mbaud; the max. length of the cable between two units is 185 m.

Following Ethernet communication protocols can be used **NETBEUI** and **PATHWORKS**

The PATHWORKS protocol is a sub-group of the DECNet PATHWORKS 4.0 for communication with boards.

NOTE:

According to the manual the PC can be a normal personal computer or a WinLink Operator Panel. For simplicity’s sake this manual always refers to these units as “PC’s” or “computers”.

COMMENTS ON THE NETWORKS

Here follow a few brief comments on the networking on computers (in this case, by 'computers is meant either a PC or a 10 Series System). By networking is meant both the physical link true and proper and the software governing it.
Before giving a brief description of the networks used in the 10 Series System, it is worth getting acquainted with the terms which will be used in this guide as listed below.

Definitions

In the description of the network systems, their installation and configuration and their relative functions, the following terms will be used:

Machine or terminal — Term which identifies the calculator (PC or 10 Series System) to which the network is connected.

Local machine — Is the machine on which you are operating.

Node — Is a machine equipped so that it can communicate with at least one other machine (node) which is equipped in the same way. In general, the node can be identified with the channel in order to avoid confusion in the presence of machines supplied with more than one channel (in this case, every channel is a node).

Network — Is a group of nodes connected by a common physical means which, in our case, can either be an Ethernet cable (Thin Ethernet or Thick Ethernet) or a serial line.

Communication protocol — Is the form which the nodes follow to exchange all the various types of information that need to be sent on the network. Other than the actual data, the exchanged information can be checksum data to verify the exactness of the messages, specifications relating to the sender and receiver, data identifying whether the message belongs to a certain sequence of messages or session, etc.

The nodes which need to exchange messages must adopt identical communication protocols.

The most recent protocols are subdivided into 'levels' which define the various types of information to be sent through the network.

Segment — Is a sub-group of a network. On the same network nodes with different protocols can be connected, as long as they do not interfere with each other, i.e. they can co-exist on the same physical vehicle. Obviously, in this case the nodes with different protocols cannot communicate between themselves. On the other hand, those with the same protocol using the same physical vehicle are said to belong to the same 'segment'.

On a network it is therefore possible to define more than one segment, each with its own nodes with the same protocol, or a single segment when all the nodes are able to inter-communicate. In other words, a segment is a 'logical component' whereas the line on which it operates is a 'physical component'.

Each machine can present itself to a segment through a single channel. If the machine has more than one channel, each of these must be connected to a different segment.

Channel

Is the set-up needed so that a node can present itself to the relative network. This is composed of a hardware part, i.e. a board or a serial connector provided with the relative cable, and a software part, i.e. the protocol. A machine can be supplied with more than one channel (connected to different networks), as long as the various combinations of hardware and software (same or different) are compatible and do not create problems either to the machine or to the networks to which they are connected. A machine with more than one channel is considered the node for each channel: in this way it is possible to say that more than one node can co-exist in a single machine.

NOTE:
At present, only one channel per machine (10 Series) can be installed on the computers.

Address

Is a string of numerical characters which identifies univocally a specific node connected to a certain segment. The address is given by a node on requesting connection with another node. The latter, having recognized its own identification code replies with a sequence of characters so that communication can commence.
The address is necessary for the PATHWORKS protocol.

Configuration or plant supervisor

This is a machine which can configure or monitor the CNC's in the network. Its software is therefore enhanced with some additional features.

Server

A system is called the Server when it places all or part of its own local resources (e.g. the hard disk or the printer) at the disposal of other users connected to the network.

Client

Is the definition used for the system which, on the network, makes use of the resources offered by a Server. From the Client's point of view, the resources made available by the Server are remote.

Master

Is the node from which communication is begun, i.e. that from which a request for connection is launched. It will maintain the denomination of Master for the duration of the session. The Master always has the function of Client.

Slave

Is the node which receives the communication launched by the Master. It will maintain the status of Slave for the duration of the session. The Slave always has the function of Server.

NOTE:
The 10 Series systems offer both the Master and Slave features at the same time.

Network topology

To organize a communication system, it is necessary to define all the elements of which it is composed, i.e. the machines, the channels (or nodes), the segments and the physical networks. The machines and the segments can be considered to be logical components which can be defined independently, whereas the channels and the networks are physical components.

Seen graphically, the logical components can be represented as in the figure below, i.e. components which are not yet correlated.

The two segments can lie on the same physical network or belong to different networks. To allow a machine to communicate with the others, it is necessary for the communication channel it uses to be defined (i.e. install the hardware, board or serial cable and the software), and to specify the segment on which the channel will present itself. The channels therefore have the duty to connect a machine to a segment. In order to be able to define a channel, it is therefore necessary to define at least one machine and a segment first.

The configuration of the most simple network is one with two nodes (in our case a 10 Series System and a PC) which communicate on the same segment as illustrated in the following figure in which Machine 1 equipped with Channel 1 and Machine 2 equipped with Channel 2, both present themselves on Segment 1. Obviously, other than the same hardware, the two channels must have the same protocol. The Machine, with the relative Channel, represents the homonymous node.

For example it can be organised as shown in the figure which comprises three 10 Series systems with the relative channels connected to a segment which also contains a PC which can function as supervisor or server. The Supervisor is provided with a printer which can be placed at the disposal of the network as a shared resource.

IMPORTANT

At present it is only possible to have one channel and one segment for each 10 Series node.

CONFIGURATION OF THE NETWORK

General view

In a network of 10 Series Systems, each machine is able to configure itself as a node in the network and to memorize its own configuration locally.

At installation of the release a default configuration is loaded which you can then modify using the network configurator (**NET CONFIG** softkey of **PERIPHERALS**).

Network configurator

The network configurator, which is started by pressing the **NET CONFIG** softkey, is a program which permits definition of the configuration of each machine as a node in the network.

The configurator foresees an interface towards the user based on a menu of softkeys and input windows which are very similar to those normally used on the system. These are subdivided by function are active or not in accordance with the configuration level and of the model of the machine to be configured.

In fact, two configuration levels are foreseen:

- basic level
- advanced level

For each level a corresponding group of possible operations and information is supplied the user.

The results of the network configurator's work is a set of data inserted into a database, on each of the machines which make up the network system, which is ready to start functioning when the network itself is activated or rather, at the next bootstrap.

FUNCTIONS OFFERED BY THE NETWORK

The functions offered by the network permit distribution of a system's resources, where each machine is delegated a series of specific tasks, thus considerably facilitating the operator's job. Further, with the possibility of future system enhancements, the objective of "distributed architecture" becomes closer to being realized.

File Transfer

With the File Transfer function access and transfer of files by means of the network are allowed.

In File Transfer, the 10 Series System has access to the disks of the *remote PC* for reading and copying operations as if they were local disks. In its own operating environment it is sufficient to supply the names specified in configuration in order to obtain access.

Access to remote disks through the network can be obtained through the 10 Series System with the following functions:

- UTILITY environment
  - DOS SHELL (all foreseen functions)
  - PP MANAGEMENT (for Import/Export of files)
  - SERVO MONITOR (save/restore on remote disks)
  - AMP (backup operations)
  - Softkey OEM, DOS Graphic Interface option (in accordance with definitions)
  - SYSTEM HISTORY (EXPORT)
- PART PROGRAM environment
- DOS REAL TIME environment

For correct operation, the Server machine must have the disks and the printers defined as '*sharable services*' whereas the Client machine must define the same objects as '*remote services*'.

Server services

The server node can make disks and directories available by defining them as shared **services**. They must be associated with:

- the name identifying them on the network (different Servers may define services with the same name)
- the associated protections. This can be: **R** (read only), **RW** (read and write) and **RWC** (read, write and create)
- a password (if required).

Access from a client

In order to obtain access to the services made available by the Server, a Client node must configure the disks that it intends to use. Access will be identical to local type access. Configuration of the remote services on the Client must include:

- the name with which the disk or directory will be known to the system (e.g. K:, R:, etc.)
- the name of the node on which they are defined
- the name of the service as defined by the Server
- a password (if required).

It is possible to add, delete or modify the configuration of a service at any time both on the Server and on the Client. However it has to be kept in mind that only the configuration of a service as Client can take immediate effect, whereas services configured as Server become available on the 10 Series systems only after the next initialisation of the system (switching it off and on).

The 10 Series System can access remote disks defined on the PC if:

- the network is suitably configured on both machines and the remote disks have been defined
- the network has been activated on both machines

Exchanges of messages between tasks

With the **IPC** (Inter Process Communication) feature, direct interaction is permitted between tasks. 'Task to Task' communications are intended to allow a 10 Series System, or a generic host computer (WinLink or a PC, for example), control and monitor another 10 Series System; such a structure is generally realized in point-to-point or network situations.

The messages that are exchanged in this generic network have different forms in function of the systems and environments generating them. Simplifying matters, the lay-outs possible in a point-to-point network are as follows:

Communications between 10 Series systems:

- through **DOS Real Time** instructions
- through **Asset** instructions

both are based on the 'shared environment' concept of the utility 'Net Config' concept, with configuration of a network of logic connections between the different environments of the different machines prior to the communication session.

Communications between host computer and 10 Series System:

- through **DLL NetBIOS/OSAI Netbios**

construction of the network and the messages to be exchanged is the sole responsibility of the OEM application developed independently on the generic host computer, while on the 10 Series System this continues to be handled by the 'Net Config'

IMPORTANT

It is important to use this feature appropriately as it interacts directly with operation of the 10 Series control.

Communication through Asset instructions is based on using 'normal' commands on the 10 Series control; these commands are made remote in configuration of the remote devices/environments (see three-letter code **DCC**).
These instructions are a super-set of the normal part-program commands (see the "10 Series CNC ASSET Language" manual).
Access is possible to remote processes and the Dual Port areas of remote systems.

Communication from the Dos Real Time environment is realized by directing the functions usually available in local to the remote environments, using the same identifiers as configured for these remote environments in the 'Net Config' (see the "10 Series CNC DOS Real Time Interface" manual)

In order to communicate DLL NetBIOS/OSAI messages there has to be a controller interfaced to the net: to it the OEM software can send messages and from it the response is received.

Actual transmission of the messages compiled by the OEM software is managed by **NetBIOS**[1]

NOTES:

- A Host system can be a *Client* of a 10 Series System (and operate in the *file-transfer* and *task-to-task* modes simultaneously) and at the same time operate as *Server* after configuring shared services.
- Configuration of the *sharable* and *remote* environments on 10 Series Systems is performed by the network configurator; the 10 Series System can make its operating environments available as *server*type services. The following must be associated with each service (in the same way as file services):
  - the logic name it is known by in the network
  - the protections: R (read only), RW (read and write) and RWE (read, write and execute)
  - a password, where required.
- The environments can be configured on 10 Series Systems at any time: connection to a new remote environment can be immediate (provided that this environment has already been made available on the network), whereas definition of a new service for the network will have effect only after the next time the 10 Series System is initialized.
- For configuration of the remote environments on Host, see the initialization sequence indicated for the client session in Chapter 2, "OSAI Netbios" Manual 10 Series CNC Mini-DNC Ethernet Library Functions.

[1] NetBIOS

is the IBM standard protocol for network communications between Personal computers; there is a vast range of publications on the matter, notably:

- 3Com Corporation. 1987. ***NetBIOS Programmer's Reference (3260-00)***. Burlington, Massachusetts.
- IBM Corporation. 1984. ***IBM PC Network Technical Reference Manual (6322916)***. Armonk, New York.
- IBM Corporation. 1988. ***Migration Considerations for NetBIOS***.IBM Washington System Center Internal Flash 8736.
- Schwaderer, W. David. 1988. ***C Programmer's Guide to NetBIOS (22638)***. Howard W. Sams & Company.

NOTE:

The NetBIOS and OSAI Netbios messages that can be used to exchange operating commands in the network are documented in detail in chapter 2 Manual "10 Series CNC Mini-DNC Ethernet - Library Functions" and in more general terms in Appendix A. Chapter 1 contains the DLL functions for development of applications in the Windows environment.
Supplied with the DLL is the source code of an application provided by way of example only.
The application creates a user interface using the MDI mode.

USER INTERFACE

The communication environment's user interface serves for configuration of the environment itself and for working on the network.

Configuration of the systems connected in the network is performed by characterizing a certain number of parameters needed for unique identification of all the components included in the network itself.

This means that a series of values must be introduced on given variables to which access is obtained through a series of displays, softkeys and data entries which make up the network configurator's user interface.

Access to the configuration environment

To activate the Network Configurator screen, proceed as follows:

1. From the system's main screen or from emergency screen, press the **PERIPHERALS** softkey. The following screen is obtained:

PERIPHERAL CONFIGURATOR			DATE: dd/mm/yy	TIME hh:mm:ss
INTERLNK CFG	INTERLNK	INTERSRV		HELP
	PRINT DEVICE	REMOTE BOOT	NET CONFIG	MINI DNC CFG
		SERVICES CON.	ENVIR. CONNECT	EXIT

2. Press the **NET CONFIG** softkey. The main screen of the Network Configurator appears.

Network configurator screen displays

In the network configurator, a certain number of screens are foreseen which are briefly described below. For more details, refer to the following paragraphs.

Main screen

This is the first screen displayed when the Network Configurator environment is entered and changes according to the enabled configuration level.

The meaning of the fields and the columns contained in the screen are the following:

First screen display: *Advanced Level*

NETWORK CONFIGURATOR			DATE: dd/mm/yy	TIME hh:mm:ss
			Current level: Advanced	
Op. System	OSAI	Protocol Type	NETBEUI	
Channel Name	NC0001	File Transfer	Y	
Board Type	PCNET	Remote IPC	Y	
				page 1 of 3

LEVEL								HELP
SHARED ENV		REMOTE ENV		CHAN GENERAL		ENVIR. CONNECT.		
SERVER		CLIENT				SERVICES CONN.		EXIT

The fields and the columns present in the screens shown above have the following meaning:

First screen display

Current level Is the active configuration level in that moment. Two values are possible which correspond to visualization of a greater or lesser amount of information and the specific softkeys (and consequent commands) are enabled. The possible levels are:

- **Basic:** Is the lowest level which only enables visualization of basic information but does not permit configuration modification.
- **Advanced:** Is the highest level in which all the foreseen functions are enabled for system configuration.

Op. System Operation System on the machine.

Channel Name: Is the name of the visualized channel.

Board Type: Is the type of physical communication means. Possible value is:

PCNET Integrated Ethernet on the OSAI system

Protocol Type: Is the type of protocol envisaged for that segment. It can assume the following values:

PATHWORKS for the NETBIOS DIGITAL protocol
NETBEUI for the NETBIOS protocol for MS DOS/WINDOWS environments

File Transfer: Indicates whether the file transfer function on that channel is enabled or not. If **[Y]**, the channel can effect file transfer, or rather that it is possible to define Server and/or Client services. If **[N]**, file transfer cannot be carried out on that channel.

Remote IPC: Indicates whether the function of remote IPC for task-to-task activities is enabled or not. For this software release it is always **[N]**.

Second screen display

NETWORK CONFIGURATOR		DATE: dd/mm/yy	TIME hh:mm:ss
		Current level: Advanced	
Offered Service	**Service path**	**Access**	
DISK1	F:\UPP	R	
DISK2	F:\NEWPP	RW	
Remote drive	**Required Service**	**Server name**	
K:	DISKC	PC1	
			page 2 of 3

LEVEL				HELP
SHARED ENV	REMOTE ENV	CHAN GENERAL	ENVIR. CONNECT.	
SERVER	CLIENT		SERVICES CONN.	EXIT

This screen is only present if Server and/or Client services have been defined on the channel.

Data concerning Offered service, Service path and Access identify services of the Server type (configured on that channel with the **SERVER** softkey), which are present when the machine was configured as Slave or Master/Slave. Details relating to Remote drive, Required service and Server name identify services of the Client type (configured on that channel with the **CLIENT** softkey) which are present when the machine was configured as Master or Master/Slave.

Offered service: Lists the names of all the services configured on the selected channel with the **SERVER** softkey.

Service path: Is the path inside the Server which must be followed in order to reach the corresponding service shown in the Offered service column. It coincides with the DOS path.

Access: Identifies the type of access allowed for that service. The values contained in this column have the following meaning:

- **R** access is allowed for reading of the data contained in the service
- **W** access is allowed for writing and modification of the data contained in the service
- **C** creation of files in the specified service is allowed (disk or directory).

Remote drive: Specifies the local name (on Client) of the remote drive, corresponding to the service with its name in the *Required service* column, on the node indicated in the *Server name* column. These services are defined with the **CLIENT** softkey.

Required service: Is the name of the service, to which the Client wishes to have access, as defined on the remote node.

Server name: Is the name of the node and therefore of the channel on which the service requested resides.

In the example shown in the described screen display, the 10 Series system offers the network its directories F:\UPP and F:\NEWPP, naming them DISK1 and DISK2. The host can gain access to DISK1 for reading and to DISK2 for reading and up-dating, but it cannot create new files on the 10 Series.
Further, from this channel it is possible to gain access to a service on the host using drive K:. Connection with the remote disk is active when the data line is highlighted.

Third screen display (optional)

NETWORK CONFIGURATOR			DATE: dd/mm/yy			TIME hh:mm:ss
			Current level: Advanced			
Local environment	**Service name**	**Type**	**Access**			
61	**PROCESS**	**1**	**RWE**			
.	.	.	.			
.	.	.	.			
Remote Environment	**Environment name**	**Server name**	**Service name**	**Type**	**Access**	**Filter**
99	**REM UTY**	**NC0002**	**UTY 2**	**3**	**R**	**0**
						page 3 of 3

LEVEL				HELP
SHARED ENV	REMOTE ENV	CHAN GENERAL	ENVIR. CONNECT.	
SERVER	CLIENT		SERVICES CONN.	EXIT

The "Local environment" data define the "server environment" type services made available for the network *channel*, whereas the "remote environment" data identify the "client environment" type remote services that are to be accessed through the channel selected.

ENVIRONMENT

An "environment" is a set of *tasks* and *system* objects that combine to form an independent "virtual machine" (stand-alone), able to communicate with other virtual machines through the exchange of predefined messages and commands.
The 10 Series System internally comprises a number of environments, each with features of its own (e.g. Numeric Control environment, Machine Logic environment, UTILITY environment, etc.).

This mechanism may be exploited for the transparent mode exchange of messages, in other words for performance of the same operations between environments resident on different machines.
Again in this case the concepts of "client" and "server" are used for environments, in the same way as for "client" and "server" files.
A "server" environment is a local environment on a machine at the disposal of the network, i.e. to which the other machines are provided access through the network.

A "client" environment is an environment that does not exist on the machine in which it is defined, but on a remote machine (where it must have been defined as "server environment"): messages for a "client environment" are sent through the network to the remote environment and replies are received from the latter as though it were a local environment (except for the delay caused by transmission on the network).

Local environment Lists the ID's (in Hex) of all the environments made available by the network and configured by means of the **SHARED ENV** softkey on the channel selected.

Service name: The name by which the service of local "server environment" configured is recognized on the network.

Type: This is a decimal value defined by the user, identifying type, i.e. the features, of the environment.

Access: Identifies the type of access permitted for the environment. Significance of the values in this column is as follows:

R: Read
W: Write
E: Execute

The "service name", "type" and "access" values are checked by the remote machines at the time connection is requested.

Remote environment: The ID (in Hex) of the remote environment corresponding to the service made available by the node indicated in the "server name" column with the name in the "service name" column.

This value may be used by user application programs (e.g. Dos Real Time).

Environment name: The mnemonic of the ID of the remote environment: may be used directly (for example, in ASSET programs).

Server name: The name of the node (and therefore of the channel) on which the remote service (environment) has been defined and resides.

Service name: The name of the service (environment) for which client type access is required, as defined on the remote node.

Type: The type associated with the remote environment on the server.

Access: The type of access required for that service; is the same as, or a subset of, the accesses defined on the server.

Filter: The ID (in Hex) of another environment designed to filter certain commands and see if they can be executed by the remote environment; if the value is zero, no filter is required.

NETWORK CONFIGURATOR SOFTKEYS

When pressing the **NET CONFIG** softkey the softkeys shown below are displayed. They are common to all screens of the network configurator.

Some softkeys can be active or not according to the configuration level selected as shown in the two columns B (Basic) and A (Advanced).

LEVEL				HELP
SHARED ENV	REMOTE ENV	CHAN GENERAL	ENVIR. CONNECT.	
SERVER	CLIENT		SERVICES CONN.	EXIT

The following table summarizes the function of each of these softkeys. A detailed description follows on the next pages:

Level

B	A	Softkey	Meaning
❑	❑	**LEVEL**	Effects selection of the three configuration levels in a cyclical sequence.
❑	❑	**HELP**	This softkey opens a contextual help window which gives certain practical information that depends on the activated softkey and on eventual data entries opened.
	❑	**SHARED ENV**	This softkey is used to configure the environments made available to the channel in the network; it is activated when the message exchange features are configured (remote IPC=Y in the CHAN GENERAL screen)
	❑	**REMOTE ENV**	This softkey is used to configure the remote environments that the channel wants to use; activated when the message exchange features are configured (remote IPC=Y in the CHAN GENERAL screen).
	❑	**CHAN GENERAL**	With this softkey it is possible to configure certain general information concerning the channel selected previously with the **SELECT CHANNEL** softkey. Causes opening of a data entry which requests configuration of different facts according to the type of protocol for that channel.
	❑	**ENVIR. CONNECT**	For immediate connection of all environments configured as remote.

B	A	Softkey	Meaning
	❑	**SERVER**	This softkey's function is to configure the various resources, disks or printers, placed at the network's disposal by the channel. It is active when the node has a Slave/File Server function. It provokes opening of two data entries (the second appears on confirmation of the logical service name) in which determinate values must be introduced which characterize the specified resource (see Chapter 4 for details).
	❑	**CLIENT**	This softkey's function is to configure the remote services, disks and printers, which the channel intends using and eventually to connect them automatically. It is activated on nodes with a Master function. It provokes opening of two data entries (the second appears on confirmation of the logical service name) in which determinate values must be introduced which characterize the specified service (see Chapter 4 for details).
	❑	**SERVICES CONN.**	For immediate connection of all services configured as remote.
❑	❑	**EXIT**	Causes exit from the channel configuration environment and return to the previous softkey configuration (10 Series) or to the DOS prompt (PC).

Level selection (LEVEL Softkey)

To select the desired level, simply press the **LEVEL** softkey.

There are two levels corresponding to different degrees of visualized data and above all, differences in how the softkeys are enabled with consequent changes to the commands.

Basic: Is the lowest level which enables visualization of basic information but does not permit configuration modifications.

Advanced: Is the highest level in which all the foreseen system configuration functions are enabled and in particular, the ones which act on the network's structure.

Access to the *Advanced* level is conditioned by enabling SECURITY: the Advanced level is enabled, if the 4th bit of byte 5 is set to 1.

Channel configuration (CHAN GENERAL Softkey)

By pressing the **CHAN GENERAL** softkey you open an input window similar to the following:

```
                    - - CHANNEL CONFIGURATION - -

Channel name     :  NC0001        Board type        :  PCNET
File transfer    :  Y             Remote IPC        :  N
Active channel   :  Y             Protocol type        NETBEUI
Address (only pathworks protocol):                  :
```

The fields contained in this data entry have the following meaning:

Channel name: Is the name of the selected channel. The name is an alphanumerical string, with a maximum of 15 characters, which can coincide with the name defined for the machine. With the PATHWORKS protocol, the maximum string length is 6 characters.
The default value is NC 0001.

Board type: Is the type of network board used. It can only assume the value **PCNET** which indicates the integrated OSAI network.

File transfer: Enables **[Y]** or does not enable **[N]** the file transfer function.
The default value is **[Y]**.

Remote IPC: InterProcess Communication.
This field must be set **[Y]** when configuring task-to-task exchanges of commands and messages.
The default value is **[N]**.

Active channel: Enables or disables the channel at the next system initialization. Only if it has been placed on **[*Y*]** will the network be activated after the next initialization otherwise, even though the supplied information will be held in the memory, the system will be isolated from the network itself.
The default value is **[N]**.

Protocol type: Is the protocol used. Possible values are:

NETBEUI for the NETBIOS protocol used in the DOS and Windows environments (default value)

PATHWORKS for DIGITAL's Pathworks protocol

Address: Is the logical address of the node. This field is only valid, if the Pathworks protocol is used.

The data introduced in this field must have the following format:

XX.YYYY

with:

XX that indicates the area (from 1 to 63)

YYYY that indicates the node (from 1 to 1023)

IMPORTANT

To function correctly, the value of the *Channel name* parameter must be unique on the network, i.e. two channels cannot exist with the same name.

The default value '***NC0001***' defined on all the 10 Series Systems **must** therefore be modified if more than one of these systems are connected to the host.

Configuration of the channel's parameters is only activated when the system on which the channel resides is next bootstrapped.

Configuration of server type services (SERVER Softkey)

These services identify the resources (10 Series disks and directories).

In order to characterize Server type services, follow the procedure described below for each of them:

1. Press the **SERVER** softkey. A data entry is opened which will have a similar appearance to the one shown below:

```
                    - - SERVER RESOURCES - -

Service name     :                Delete          :  N
```

where:

Service name: Is the name of the service to be configured, made up of a maximum of 8 alphanumerical characters (e.g. DISCO1).

Delete: This field serves to cancel the previously configured service. To delete, set it **[Y]**. The service will still remain active on the 10 Series System until the next machine initialization.
If a service is to be created or re-defined, indicate **[N]**.

2. Type in the name of the service required. The example given below refers to a directory (DISCO1), to be made available to the other users.
3. Press the **[Enter]** key to confirm the name of the service to be configured. A data entry will appear similar to the one illustrated below:

```
                    - - SERVER RESOURCES - -

Service name     :  DISCO1        Delete          :  N

Access           :                Password        :

Service path     :
```

The fields to be completed in the two data entries shown above have the following meaning:

Access: Defines the type of access to be associated to that service. The letters to be used in this field are:

- **R** permits access to the service so that the data contained in it can be read. This letter must always be present.
- **W** permits access to the service so that the data contained in it can be read or modified. This letter, when present, must always be preceded by **R** (i.e. must be written **RW**).
- **C** allows the user to create files in the specified shared directory. When present, this letter must always be preceded by the other two mentioned above, i.e. **RWC** must be written.

Password: (Optional). It is the key word which may be associated to the service. With this password, the service will only be accessible to the user who knows it. The password is made up of no more than eight letters, numbers and/or punctuation marks but spaces are not accepted.

If this field is empty, no password will be associated to the service and if one was defined previously it will be removed.

Service path: Is the full path of the directory to be shared.

5. Fill in all the fields in this data entry with the appropriate data or, if necessary, modify those proposed. When this is done, confirm with the **[Enter]** key or the same **SERVER** softkey.

The procedure given above must be repeated for each of the services which the Server intends to offer the network users.
On the 10 Series System, Server type services are either enabled or disabled as from the next bootstrap of the system on which the channel resides.

IMPORTANT

If a service defined in such way is to be used on a **Windows NT** client machine, the configuration program Start.Programs.Administrative Tool.User Manager, and then User.New User has to be run on this machine with the following data for defining the new user:

Username	XP9F01
Full Name	none
Description	none
Password	none
Confirm Password	none
User must change password	not enabled
User cannot change password	not enabled
Password never expires	enabled
Account disabled	not enabled

The same operation has to be repeated varying Username and associating to it the values **XP9003**, **XP6103** for the first process, **XP6203** for the second process and so on up to ***XP7403*** for the 20th process.
The standard procedure for creating a virtual drive has to be carried out which enables access to the service.
Only in this way the service defined on page 10 can be accessed from Windows NT.

Configuration of client type services (CLIENT Softkey)

The client services are those which the channel under configuration ask from the network (disks of another 10 Series system or disks of the host computer).

In order to characterize the services of the Client, follow the procedure described below for each of them:

1. Press the **CLIENT** softkey. A data entry is opened similar to the one shown below:

```
                    - - REMOTE DEVICE - -

Remote drive     :                Delete          :  N
```

   The meaning of each field contained in this data entry is the following:

   Remote drive: Is the local name of the disk associated with the service (Service name) belonging to a remote host or 10 Series system (Server name). It consists of a letter from I to Z not followed by a colon (e.g. K).

   Delete: This field serves to cancel the definition and the usage of a previously defined service. For cancellation indicate **[Y]**. In the case of creation or re-definition, indicate **[N]**.

2. Type the name of the remote device and press the **[Enter]** key to confirm. A data entry window similar to the one below will appear:

```
                    - - REMOTE DEVICE - -

Remote drive     :  K             Delete          :  N

Service name     :                Server name     :

Password         :                Automatic connection :
```

The fields contained in the two data entries shown above have the following meaning:

Service name: Is the name of the service the Client wants to access which is provided by a host computer or a 10 Series system connected to the network.

Server name: Is the name of the channel on which the service reside (host computer or 10 Series).

Password: (Optional). The password associated with the service to limit access to authorized persons only. The password is made up of no more than eight letters, numbers and/or punctuation marks, but spaces are not accepted. It must coincide with the one specified in the service definition of the service configuration data entry of the Server on the remote machine.

Automatic connection: With this parameter it is possible to effect automatic connection of the Client to the specified service. If **[Y]**, connection will take immediate effect. If **[N]**, connection will have effect only from the next system initialization.
In order for this to succeed, both the local channel and the one on the host have to be active and the server must offer the service currently requested.

3. Fill in the fields of this data entry with all the appropriate data. When complete, confirm the data entered by pressing **[Enter]** or the **CLIENT** softkey again.

The procedure described above must be repeated for all the services on the host to which access is required.

IMPORTANT

At the moment of connection to a disk which resides on the host, it is necessary for the service to have been defined. If, on the other hand, the disk resides on the 10 Series System, the service must be defined and the system must be bootstrapped. It is always necessary for the network to be enabled on both channels.

CONFIGURATION OF TASK-TO-TASK TYPE SERVICES

Enabling Task-To-Task communication

Remote task-to-task communication is enabled by setting the field Remote IPC to (Y) in the **CHAN GENERAL** data entry.

At this point the task-to-task communication feature between NC and PC becomes active. In the case of task-to-task communication between NC's the necessary parameters still have to be configured using the softkeys **SHARED ENV** and **REMOTE ENV.**

Defining a local environment as sharable in the network (SHARED ENV softkey)

A shared local environment identifies a local virtual machine as accessible by other systems for the exchange of messages and commands.
To define a local environment on a *system* as accessible and *sharable* by the network, the fields of the two **SHARED ENV** data entry windows must be compiled appropriately.

On completion of data entry, press **[Enter]** or **SHARED ENV** again.

IMPORTANT

The values entered in the two data entry windows will be available for the network following the next *system* bootstrap.

SHARED LOCAL ENVIRONMENT (First data entry)

```
-- SHARED LOCAL ENVIRONMENT --

Environment Id:                Delete: N
```

Environment Id: The ID of the local environment to be made *sharable*.
This can be:

ID	Environment
061H	Numeric control, process 1 (always present)
062H ÷ 074H	Numeric control, processes 2 to 20 (if configured)
050H	Machine logic (always present)
090H	Human Interface (always present)
09FH	DOS Graphic Interface (if installed)
0B0H	DOS Real Time Interface (if installed)
0B1H	End User DOS option (if installed)

Delete: Press **[Y]** to remove the availability of the previously configured environment or **[N]** to create or modify the *service;* creation/modification, and deleting will be effected at the next *system* bootstrap.

NOTE:
Deleting a *service* could also be occasioned by a physical problem in the network (e.g. cable

Declaring a local environment, or other machines connected to the network, as *available* must always be done with caution as the external commands interact directly with the *system,* with the possibility of altering normal work.

SHARED LOCAL ENVIRONMENT (Second data entry)

```
-- SHARED LOCAL ENVIRONMENT --

Environment Id:      61          DELETE: N
Service Name:                    Type:
Access:                          Password:
```

Service Name The name to be associated with the network *sharable* environment.

Range: from 0 to 8 alphanumeric characters (e.g. PROCESS1) and must be unique on any one machine.

Type Value identifying the type of service, the predefined types being:

1 - Numeric control environment
2 - Machine logic environment

Range admitted: 0÷127

NOTE:
the value of this field is not controlled during this characterisation stage, but when a request for connection is received from a remote system; this value must represent the type of *service* connection is required to.

Example:
The ASSET environment on a *remote system* can connect to the *service* by specifying type (e.g. 2=machine logic).

Access The type of access allowed on the remote machine.
This parameter is tested at connection time, exactly as for the previous parameters.
The same type of access, or a subset of it, must be specified by the remote machine.

Range of values: R/W/E:

R - Read access
W - Write access
E - Execute access

Password Optional field.
The password that may be associated with the environment to select accesses from the remote machines.
Requests for connection from remote machines must specify the same password.

Connection to a remote environment (REMOTE ENV. Softkey)

A *Remote environment* is a virtual machine physically resident on a *remote* 10 Series *system,* which receives messages and/or commands as if they were sent to it locally.

To connect a remote environment (defined locally as *shared*), the **REMOTE ENV** data entry windows must be compiled appropriately.

On completion of data entry, press **[Enter]** or **REMOTE ENV** again.

REMOTE ENVIRONMENT (First data entry)

```
-- REMOTE ENVIRONMENT --

Environment Id:                    Delete: N
```

Environment Id: The ID of the remote environment

Range of values: 030H ÷ 0FFH

NOTE:
The value of this ID must obviously be different from the one of the local *machine*; for example, to access a process 1 of a remote environment (defined on the machine with 61H), a different ID will have to be used (e.g. 31H) to avoid any conflicts with the local process 1 environment (see list of ID's in SHARED ENVIRONMENTS).
If an incorrect ID is specified, no error is reported during this parameterization phase, but connection will prove impossible on the next bootstrap.

Delete: Type **[Y]** to remove access to the remote environment; type **[N]** to add or modify access to a remote environment.
Cancellation, generation and modification are executed at the next *system* bootstrap.

WARNING

When sending requests to a *remote environment*, there is direct interaction with the 10 Series structure, altering the course of normal work.

REMOTE ENVIRONMENT (Second data entry)

-- REMOTE ENVIRONMENT --		
Environment Id:	**Environment name:**	
Server name:	**Service name:**	**Type:**
Access:	**Password:**	**Filter:**

Environment name The name associated with the new *remote environment,* to be used, for example, in an ASSET part program to specify the *remote machine* connected.
This name must be unique inside the system.
Range of values: from 0 to 8 alphanumeric characters

Server name Name of the channel on which the environment is local.
Range of values: from 0 to 8 alphanumeric values

Service Name Name associated with the *sharable* environment as **Service Name** when configured on its local channel.

IMPORTANT

The **Server Name** and **Service Name** provide unique identification of each environment available in the network.

Type The type of *service*; this parameter must be consistent with the value declared in the *sharable* environment with the **Type** parameter on its local channel.

Access The type of access required for the remote environment; must be the same as, or a subset of, the value specified for the **Access** parameter declared in configuration of the *sharable* environment.

Password Optional field.
If the password was specified in the *sharable* environment on its local channel, this field must be compiled with the same string.
If no password was specified in the *sharable* environment, this field must not be filled in either.

Filter Optional field.
Must be declared when the commands to the remote environment are sent first to another environment for preprocessing or verification.
The *filter environment* must be suitably studied for the processing of filter operations and for transmission of the filtered message to the *target environment.*
If the filter environment is on a remote channel, it must be defined as **Remote environment.**

NOTE:
This field is normally used to filter commands sent to a numeric control distant from its machine logic.

Immediate connection of remote environments (Softkey ENVIR. CONNECT)

Using the **ENVIR. CONNECT** softkey of the environment "**Peripherals**", all environments configured as remote can be connected immediately without having to wait for the next bootstrap; the environments will be connected provided that the corresponding local environment has been configured as "*SHARABLE*" on the *SERVER* machine.

When the **ENVIR. CONNECT** softkey is pressed, the following window is displayed, in which the environments to be connected will be displayed one after another:

Connecting local environment ..... to .....

indicating for each

local ID of the remote environment	host machine channel name

NOTE:
This softkey re-initializes the connections stored.
If used when connections are already active and functioning, this operation first disconnects and then reconnects the link. Therefore operations taking place between the environments are aborted.

Automatic connection of remote devices (Softkey SERVICES CONN.)

The **SERVICES CONN.** softkey can be used to connect the *CLIENT* immediately to all services configured as remote, on condition that the services requested are effectively offered by the server specified and that the latter is up and running.

When the **SERVICES CONN.** softkey is pressed, the following window is displayed, in which the services to be connected are displayed one after another, e.g.:

Connecting drive K: to \\PC1\DISKC

indicating each time:

id of the local drive	host name and remote services

NOTE:
This softkey re-initializes stored connections.
If used when connections are already active and functioning, this operation first disconnects and then reconnects the link. Therefore operations running between the environments are aborted.

USE OF THE SERVICES IN THE NETWORK

Network activation on the 10 Series system

If during configuration the Active channel parameter has been set at **[Y]**, the network is automatically activated on the system as from the next start up (except for start up in emergency status). If the network is not activated, check the above mentioned parameter in the configuration environment (see Chapter 4 for details).

Loading of the network drivers is indicated on the 'CNC Software Diagnostic' screen during initialization, by one of the following two messages:

OD101 PATHWORK software loading (for installations with PATHWORKS protocol)

OD101 NETBEUI software loading (for installations with NETBEUI protocol).

IMPORTANT

If modifications are carried out on the configuration parameters, it is necessary to re-initialize the system (switch off and back on) for the modifications to have effect.

Enabling the net on a Windows PC

In order to enable the network on a PC under a WINDOWS operating system you have to correctly install the driver for the network board on the PC using Control Panel.

When installing the driver you also have to configure the NETBIOS configuration protocol and MS NETWORK.

After having enabled the network you can proceed to configure CLIENT SERVER services of the PC and CNC's connected to it by selecting the network icon on the Control Panel.

ACCESS TO A SERVICE FROM DOS SHELL

DOS SHELL is the utility in which the functions offered by the network are exploited most.

The resources offered the network by the Server and configured on the Client machine with their ID's are seen by the 10 Series System as if they were own resources.

All the DOS SHELL commands react in the same way in relation to remote services and therefore only a few examples are given below. For further details, please refer to the chapter concerning the DOS SHELL in the 10 Series User Manual. In the descriptions which follow, it is presumed that the user has already entered the DOS SHELL environment by means of the relative softkey.

Display of the drives

On entering the DOS SHELL environment, (or, in this environment, on pressing the **SHOW DRIVES** softkey), all the drives available to the system are visualized in a window with a similar aspect to the one which follows.

DRIVE	HOST PATH
A:	Local
E:	Local
K:	\\PC1\DISKC
LPT1	Local

In the example, the drive K is a remote disk connected to the DISKC service of the PC1 Server.

ACCESS TO A PART PROGRAM

If access to a part program resident on a remote disk is required (e.g. a directory of the PC which contains all the part programs used to determine operation), you only have to use the COPY command out of PART PROGRAM or Import out of the PART PROGRAM MANAGEMENT environment (if working with logic directories). You cannot execute a part program residing on a remote server.

To be executed, the part program must reside in the local hard disk. It is for this reason that it must be 'imported' from the remote Server. It is also possible to transfer part programs from the 10 Series System to the Server with the Export command of PP MANAGEMENT or COPY of a PART PROGRAM.

Here follows a brief description of the copy procedure of a part program to the NC.

1. Enter the part program management environment by pressing the **PART PROGRAM** softkey.
2. Press the **COPY** softkey.
3. In the input window which is opened, introduce the pathname of the file which is to be imported, specifying the remote disk, and the name of the file of destination (local). Confirm with **[Enter]**.

To export a file, the procedure is the same.

DOS REAL TIME ENVIRONMENT

This environment permits access to the network services through the DOS applications developed by the user.

The access to the remote services is possible for all the DOS tasks definable in this partition and also for the Utility environment.

Configuration of the services must be done by the network configurator. Enabling and disabling of this services comes about in the standard way for the Utility environment, whereas for the DOS tasks of the DOS Real Time environment they are effected by means of the E:\ABNET\DS_DRIVE command.

This command must be repeated for each one of the DOS tasks that must gain access to this services.

END OF CHAPTER

FUNCTION TABLES

FUNCTION TABLE: BYTE 0 - Reserved

7	6	5	4	3	2	1	0	Bit
0	0	0	0	0	0	0	0	

- Bit 0: Reserved
- Bit 1: Reserved
- Bit 2: Reserved
- Bit 3: Not defined
- Bit 4: Not defined
- Bit 5: Not defined
- Bit 6: Not defined
- Bit 7: Not defined

Description: These functions are reserved for project design by R&D. They do not provide context-sensitive help and are indicated as 'reserved'. They cannot be accessed by users occupying higher levels.

FUNCTION TABLE: BYTE 1 - Reserved

7	6	5	4	3	2	1	0	Bit
0	0	0	0	0	0	0	0	

- Not defined
- Not defined
- Not defined
- Not defined
- Not defined
- Not defined
- Not defined
- Not defined

Description: Enables access to functions used by QA for HW and SW testing.

FUNCTION TABLE: BYTE 2 - Maintenance

7	6	5	4	3	2	1	0	Bit
0	1	1	1	1	1	1	1	

- Possibility to install new options
- Information about HW and installed SW
- Access to Servo Monitor environment
- Access to Dos Shell environment
- Access to Peripherals environment
- Access to Compiler utility
- Enabling of user programs running in emergency mode
- Enabling of OEM console at the START-UP (valid only for security levels above 2°).

FUNCTION TABLE: BYTE 3 - Directory Protection

7	6	5	4	3	2	1	0	Bit
1	0	0	0	1	1	1	0	

- Bit 0: Not defined
- Bit 1: Access to USER part programs
- Bit 2: Access to SYSTEM part programs
- Bit 3: Access to OEM part programs
- Bit 4: Not defined
- Bit 5: Not defined
- Bit 6: Not defined
- Bit 7: Enables paramacro display

Description: These functions provide reading/writing protection. Access to executable programs is always enabled.

FUNCTION TABLE: BYTE 4 - System Tables Protection

7	6	5	4	3	2	1	0	Bit
0	0	0	0	1	1	1	1	

- Bit 0: Possibility of deleting Tooling & Warehouses Tables
- Bit 1: Possibility of deleting Checkers Tables
- Bit 2: Possibility of deleting Sources Tables
- Bit 3: Possibility of deleting User Tables
- Bit 4: Not defined
- Bit 5: Not defined
- Bit 6: Not defined
- Bit 7: Not defined

FUNCTION TABLE: BYTE 5 - Configuration

7	6	5	4	3	2	1	0	Bit
1	1	1	1	0	1	1	1	

- Bit 0: Access to AMP
- Bit 1: AMP parameter recalculation
- Bit 2: Access to configuration utilities: TOOL TABLE EDITOR, USER TABLE EDITOR, LANGUAGES UTILITY, GRAPHIC EDITOR)
- Bit 3: Not defined
- Bit 4: Access to NETWORK CONFIGURATOR intermediate level
- Bit 5: Access to language selection
- Bit 6: Enable DSI SRV CHAN
- Bit 7: Enables advanced ODM functions

FUNCTION TABLE: BYTE 6 - PLUS Execution

7	6	5	4	3	2	1	0	Bit
0	0	0	1	1	1	1	1	

- Bit 0: Access to PLUS MENU
- Bit 1: Access to PLUS Debugger
- Bit 2: Access to PLUS and WinPlus Reboot and Reboot from WinNBI
- Bit 3: Activate PLUS EDIT and ASCI EDIT
- Bit 4: Activate PLUS GEN
- Bit 5: Not defined
- Bit 6: Not defined
- Bit 7: Not defined

FUNCTION TABLE: BYTE 7 - Part Program Development

7	6	5	4	3	2	1	0	Bit
0	0	0	0	1	1	1	1	

- Bit 0: Access to Editor Digitizing
- Bit 1: Not defined
- Bit 2: Access to PP Management
- Bit 3: Access to Electronic Cam
- Bit 4: Not defined
- Bit 5: Not defined
- Bit 6: Not defined
- Bit 7: Not defined

FUNCTION TABLE: BYTE 8 - Execution of CNC Functions

7	6	5	4	3	2	1	0	Bit
0	0	0	0	0	1	1	1	

- Bit 0: Access to TABLE EDITOR
- Bit 1: Access to Axes REF
- Bit 2: Access to USER TABLE EDITOR
- Bit 3: Access to RESET REQUEST
- Bit 4: Not defined
- Bit 5: Not defined
- Bit 6: Not defined
- Bit 7: Not defined

FUNCTION TABLE: BYTE 9 - Reserved OEM

7	6	5	4	3	2	1	0	Bit
1	1	1	1	1	1	1	1	

- Bit 0: Selection from OEM console
- Bit 1: Not defined
- Bit 2: Not defined
- Bit 3: Not defined
- Bit 4: Not defined
- Bit 5: Not defined
- Bit 6: Not defined
- Bit 7: Not defined

Description: This table is reserved for the OEM who can use the Operating System dedicated function for managing the passwords of his own applications and utilities.

END OF CHAPTER

ERROR MESSAGES

An error message is made up of a two-letter code that indicates the environment where the error has been detected, a three-digit numerical code, and a string of characters that describes the error condition. In this Appendix error messages have been classified according to the environment in which they occur. Here is a list of the environments with their two-letter codes:

HIxxx	Human interface
PPxxx	Part program
TExxx	Table editor
DSxxx	DOS SHELL
PKxxx	Product key
EExxx	Emergency
BDxxx	Hardware - Basic unit
HDxxx	Hardware - Other boards
ODxxx	Software - Operating system
SDxxx	Software - Software configuration
APxxx	Applications
EDxxx	Editors errors
PFxxx	Peripherals
FDxxx	Front panel
CNxxx	Network configurator

NOTES:

- Certain messages describe an action that is in progress or has been performed by the system rather than an actual error.

- The error messages that occur during the system diagnostics (refer to Chapter 2) are displayed in conjunction with the messages FAILED (for a locking error), WARNING (for a non-locking error), and PASSED or REPORT (to indicate that the diagnosed part has passed the tests).

- Error messages are of critical importance to detect programming or operating errors as well as machine failures. Before calling the Technical Assistance make a note of the following information and be prepared to quote it:
  - message code and text
  - conditions in which the error was detected (start up diagnostics, machine status, etc.).

USER INTERFACE ERRORS

Code and message		Description	Remedial action
HI002	**Extension too long**	The part program extension has more than 3 characters.	Check and correct the extension of the part program file
HI003	**Logic name too long**	The logical part program name has more than 48 characters.	Check and correct the part program name
HI004	**The file name already exists**	The part program file exists in the directory.	Modify the part program name
HI005	**Extension error**	The file extension is not compatible with the operation. For example, a request has been made to activate a graphic file with the ".dfp" extension.	Check and correct the file extension
HI006	**File already opened**	The requested part program file is open	---
HI007	**File already opened for writing**	The requested file is open for writing.	Check the file name or close the file open for writing.
HI008	**ISO file protected or opened**	The ISO file is open or protected.	Check the file name or the security level for access.
HI009	**The current directory is empty**	There are no files in the current directory	This is not an error message but a warning.
HI010	**Char not allowed for file name**	The specified character is not allowed in a logical file name.	Check the specified logical file name and deleted illegal characters.
HI011	**Page number out of table**	The requested page is not in the data table.	Check and correct the page number.
HI012	**Parameter index out of table**	The requested index is not in the data table of the selected variable.	Check and correct the specified index.
HI013	**Error during value assignment**	This error occurs when assigning a value.	Check that the input value is in the allowed range for the variable and, if necessary, correct it.
HI014	**Variable name does not exist**	The input variable name does not exist.	Check and correct the variable name
HI015	**Wrong block number format**	The format if the block number is not allowed. For example, Nxxxxxx with x and N.	Check and correct the block number format.
HI016	**Wrong axis limits**	The specified machine axes limits are not correct.	Check and correct the specified limits.
HI017	**Error during axes reading**	System error.	Contact the Technical Assistance service.
HI018	**Duplicate axes name**	The same name has been assigned to two different axes in the machine data entry window.	Check and correct the names assigned to the axes.
HI019	**Error in plane rotation axis**	There is an error in the name of the axis on the rotation plane.	Check and correct the axis name.
HI020	**Command not allowed from logic**	This command is not allowed by the logic.	Check and correct the specified command name.
HI021	**Value out of range**	The assigned value is out of the allowed range.	Check and correct the value assigned to the variable.
HI022	**Syntax Error**	Syntax error in a field of the data entry window.	Check and correct the input parameters.
HI023	**Format Error**	The format of the parameters written in the data entry window is not allowed.	Check and correct the format of the input parameters.
HI024	**Undefined symbol**	Symbol not recognised by the logic	Check and correct the symbol
HI025	**Overflow**	The expression is too long	Split the expression into several blocks
HI026	**Division by zero**	A division by zero has been found while calculating an expression with the <EVALUATE> procedure.	Check and correct the expression.
HI027	**End of file**	End of file message.	---
HI028	**Beginning of file**	Beginning of file message.	---
HI029	**Part program not select**	The part program has not been selected.	Select a part program or abort the operation.

Code and message		Description	Remedial action
HI030	**Wrong axis name**	The specified axis name does not exist. Example: this error occurs in the data entry windows providing information about the axes (SET ORIGIN/TOOL, PROGRAM SET-UP, etc.), when the specified axis does not exist.	Check and correct the name of the specified axis.
HI031	**Failed to write variable**	Writing error in the value of a variable.	Check and correct the input data
HI032	**Failed to read variable**	Reading error in the value of a variable.	Check the name of the variable
HI033	**Offset not defined for the axis**	The presetting refers to the offset of an axis for which no length offset has been configured in the AMP.	Check the table of the offset.
HI034	**Error from PLUS environment**	Internal system error.	Contact the Technical Assistance service
HI035	**Error from servo environment**	Internal system error	Contact the Technical Assistance service
HI036	**Command & sys state not congruent**	The input command is not allowed in the current system status.	Check and correct the input command or alter the system status.
HI038	**Data Entry not configured**	Internal system error	Contact the Technical Assistance service
HI039	**Axis not referenced**	The requested operation cannot be performed if the axes have not been homed. For example, offset presetting.	Home the axes before performing the requested operation.
HI040	**Too many files in directory**	The number of files in the directory has reached the maximum limit.	Delete files from the directory.
HI041	**Access denied. Check security.**	The requested operation is not allowed without using a password.	Key in the correct password and repeat the operation
HI042	**Access denied to User directory**	Access to the USER directory is not allowed without using a password.	Key in the correct password and repeat the operation
HI043	**Access denied to SYS directory**	Access to the SYTEM directory is not allowed without using a password.	Key in the correct password and repeat the operation
HI044	**Access denied to OEM directory**	Access to the OEM is not allowed without using a password.	Key in the correct password and repeat the operation
HI045	**Not enough memory!**	Internal system error.	Contact the Technical Assistance service
HI046	**Illegal argument for TAN**	The argument of the TAN operator is 90 degrees (the result of which tends to infinite). This error has been detected by the EVALUATE utility.	Check and correct the expression before calculating it with EVALUATE
HI047	**Illegal argument for SQR**	The SQR operator argument is negative. This error happens when operating with EVALUATE.	Check and correct the terms of the operation requested with EVALUATE.
HI048	**ISO file already exist**	A request has been made to convert a graphic file with the .dfp extension into an ISO that already exists in another directory.	Modify the name of the .dfp file.
HI050	**Logical names doesn't exists**	The file that associates logical file names to part program names does not exist.	Contact the Technical Assistance service
HI051	**Press Enter to confirm the block**	You must press **[Enter]** to confirm an instruction before executing it with **Cycle Start**	---
HI052	**Failed to write on System History**	Writing error in System History file.	Delete the file with DOS Shell.
HI053	**TCP activation not congruent**	The request to enable TCP is not allowed by the current system status.	TCP can be enabled if the following are not active: - other TCP blocks - canned cycles - offsets - measuring cycles - GTL programming

HI054	**Axis not congruent**	Virtual axis ID not configured. To enable the tool direction you must configure in AMP the name and ID of the virtual axis associated to the tool direction.	Edit the part program name.
HI055	**Driver not permitted**	You tried to access protected drives.	--
HI056	**Part program not exists**	Part program you tried to activate does not exist.	--
HI057	**Error in message visualization**	Error in visualization of messages coming from processes.	--
HI058	**Char not allowed for directory names**	The *pathname*set in data entry **PATH NAME** contains characters not allowed.	--
HI059	**Path name too long**	The *pathname*set in data entry **PATH NAME** contains more than 30 characters.	--
HI060	**Dos name too long**	DOS name is too long.	--
HI061	**Driver not ready or not configured**	You tried to access a not configured or not connected drive.	--
HI062	**Wrong path name**	The *pathname* set in the data entry is wrong.	Check the *pathname* validity, if not present, insert it.
HI065	**File protected or opened**	The file is protected or is already open.	--
HI066	**File not found**	The file does not exist.	Check the file name.
HI067	**File delete failed**	Deleting operation failed.	Check consistency of set data.
HI069	**File rename failed**	Renaming operation failed.	Check consistency of set data.
HI070	**File copy failed**	File copy failed.	Check consistency of set data.
HI071	**Insufficient space on disk**	Space disk insufficient to complete the operation.	Increase the available space disk.
HI072	**Process not configured**	Selected process has not been configured in AMP	Configure process in AMP or rectify process number

PART PROGRAM ERROR MESSAGES

Code and message		Description	Remedial action
PP002	**Extension too long**	The extension of the part program has more than 3 characters.	Check and correct the extension of the part program file
PP003	**Logic name too long**	The logical part program name has more than 48 characters.	The logical name cannot have more than 48 characters.
PP004	**The filename already exists**	The specified part program name exists in the directory.	Edit the part program name.
PP005	**Extension error**	The file extension is not compatible with the requested operation.	Check and correct the file extension.
PP006	**File already opened**	The requested part program file is already open.	---
PP007	**Wrong drive or directory**	The specified drive or directory is not correct.	Check and correct the name of the drive or the directory.
PP008	**ISO file protected or opened**	The ISO file is protected or open.	---
PP009	**The current directory is empty**	The current directory does not contain any part program file.	Check and correct the name of the directory.
PP010	**Char not allowed for file names**	The specified character is not allowed in a logical file name.	Check and correct the logical file name.
PP011	**File can't be copied onto itself**	The destination file cannot have the same name as the source file.	Check and correct the file of the destination file.
PP012	**The file doesn't exist**	The requested file does not exist.	Check the file name.
PP013	**Destination file can't be opened**	The requested destination file cannot be opened.	Check the name or the protections of the destination file
PP014	**The source file cannot be opened**	The requested source file cannot be opened.	Check the name or the protections of the source file
PP015	**The source file cannot be closed**	The requested source file cannot be closed.	Contact the Technical Assistance service.
PP016	**Destination file cannot be closed**	The requested destination file cannot be closed.	Contact the Technical Assistance service.
PP017	**String was not found**	The requested string has not been found.	Check the string name.
PP018	**End of search**	The search has been terminated.	---
PP019	**Copy complete**	The copy has been terminated.	---
PP020	**Import complete**	The import operation has been terminated.	---
PP021	**Export complete**	The export operation has been terminated.	---
PP022	**Print complete**	The printing session has been terminated.	---
PP026	**Data Entry not configured**	Internal system error.	Contact the Technical Assistance service
PP028	**Operation on file is failed**	Internal system error.	Contact the Technical Assistance service
PP029	**Too many files in directory**	The number of files in the directory has reached the maximum limit. (For the limits in each directory refer to Chapter 10).	Delete files from the directory.
PP030	**Physical file does not exist. Enter restore directory. Esc abort.**	The physical file associated to the part program does not exist.	Press **[Enter]** to restore the directory and delete the program or **[Esc]** to abort the operation.
PP042	**Access denied to USER directory**	Access to the USER directory is not allowed without using a password.	Key in the correct password and repeat the operation.
PP043	**Access denied to SYS directory**	Access to the SYTEM directory is not allowed without using a password.	Key in the correct password and repeat the operation.
PP044	**Access denied to OEM directory**	Access to the OEM is not allowed without using a password.	Key in the correct password and repeat the operation.
PP050	**Logical name tables doesn't exist**	The file that associates logical file names to part program names does not exist.	Contact the Technical Assistance service.

TABLE EDITOR ERRORS

Code and message		Description	Remedial action
TE001	**Illegal process number**	The selected process has not been configured.	Check and correct the selected process.
TE002	**Impossible open directory window**	There is not enough memory to open the directory window in order to select a file.	Contact the Technical Assistance service
TE003	**Directory file access error**	This error occurs when accessing the directory the contains the Table Editor files.	Contact the Technical Assistance service
TE004	**No memory available for directory**	There is not enough memory to manage the names of the desired files.	Contact the Technical Assistance service
TE005	**No files on directory**	There are no files in the selected directory.	This is not an error message but a warning.
TE006	**Element not found**	The element searched for with FIND has not been found.	Check the search string specified in the FIND command.
TE007	**Calculation not allowed**	The requested ADD INC or SUB INC operations cannot be calculated in this field.	Check and correct the requested functions.
TE008	**File creation error**	This error occurs while creating a file.	Contact the Technical Assistance service
TE009	**File access error**	This error occurs while reading or writing a file.	Contact the Technical Assistance service
TE010	**Warning tool offset id changed**	The offset number associated to a given tool has been modified.	This is not an error message but a warning.
TE011	**Dual port illegal access**	There has been an attempt to access the dual port memory (MEMORY file name) but the table is not defined in the dual port memory.	Load the requested table in the dual port memory.
TE012	**Illegal sort criteria**	The selected sorting criteria has not been defined.	Check and correct the selected sorting criteria.
TE013	**No memory available for sort**	There is not enough memory to handle the selected sorting criteria.	Contact the Technical Assistance service
TE014	**No memory for Backup/Restore**	There is not enough memory to back up or restore the dual port memory.	Contact the Technical Assistance service
TE015	**File/Dualport config mismatch**	The configuration of the axes saved in the accessed file is different from that saved in the dual port memory.	Check and correct the axes configuration.
TE016	**No axis for the selected process**	There are no axes configured in the Origins Table of the selected process.	Check or compile the Origins Table associated to the selected.
TE017	**Illegal character in tool code**	There are illegal characters in the tool code. Legal characters are the digits without signs or separators.	Check and correct the format of the tool code.
TE018	**Graphic file creation error**	Error while creating a shape file shape.	Reinstall the VERIFY option or contact the Technical Assistance service.
TE019	**Data Base record not found**	When importing a tool code into the Tools Table a database record was not found.	Check the tool code or the Tool Data Base table.
TE020	**Offset file creation error**	This error occurs when creating the tool offsets file while accessing the Tools Table.	Contact the Technical Assistance service.
TE021	**Illegal shape**	This error is generated by a parameter associated to shape.	Check and correct the dimensions of the visualised shape.
TE022	**Illegal character in find field**	A character in the search string written in the FIND window is illegal.	Check and correct the search string written in the FIND window.
TE023	**Warning: Table locked read only**	Access to the table stored in the dual port memory denied by PLUS.	This is a read-only table.
TE024	**Illegal file size**	The length of the file table is illegal.	Delete the file with DOS Shell.
TE025	**Printer out of paper. Print lost**	The printer is out of paper.	Put more paper in the printer and re-start printing.
TE026	**Printer timed out. Print lost**	The printer does not print.	Check the printer status.
TE027	**I/O error. Print lost**	The printer is not connected.	Check the printer status.

Code and message		Description	Remedial action
TE028	**Record changed during elaboration**	While modifying a record saved in the dualport memory, PLUS modified the contents of the record and the visualised data are no longer valid.	Repeat the operation.
TE029	**Pockets go out of magazine**	The number of pockets configured in the magazine overflows the allowed maximum.	Check and correct the number of pockets configured in the magazine
TE030	**Illegal pocket per row**	In a planar magazine the number of pockets per line is missing or overflows the maximum total number. In other types of magazine the number of pockets per line must be 0.	Check and correct the number of pockets per line configured for a planar magazine.
TE031	**Pocket conflict among magazines**	The pockets configured in this magazine overlap those configured in another magazine.	Check and correct the configured pockets.
TE032	**No selected magazine**	The mandatory magazine has not been selected before accessing the Tool Table.	Select the mandatory magazine.
TE033	**Offset file access error**	This error occurs when reading or writing the tool offsets files while modifying the Tools Table.	Contact the Technical Assistance service.
TE034	**Empty magazine**	The pockets of the selected magazine have not been defined.	Check and correct the selected magazine.
TE035	**Pocket out of magazine**	The selected pocket/s overflow the magazine capacity.	Check and correct the selected pocket/s.
TE036	**Warning pocket changed**	Indicates that the number of the pocket associated to a tool has been modified.	This is not an error message but a warning.
TE037	**Pocket is still busy**	The pocket defined for a tool is occupied by another tool or reserved for it.	Check and correct the specified pocket.
TE038	**Illegal previous pocket**	A multiple tool occupying several pockets interferes with the pocket occupied by another tool (previous tool).	Check and correct the contents of the pockets.
TE039	**Illegal following pocket**	A multiple tool occupying several pockets interferes with the pocket occupied by another tool (subsequent tool).	Check and correct the contents of the pockets.
TE040	**Illegal random**	The random class stored in memory is not found.	Check the tool class and correct it.
TE041	**Tool table is full**	The dual port is full when the Tools Table associated to a magazine is copied with a LOAD function.	Cut down the number of tools configured in the table.
TE042	**Slave tool not selectable**	The multi-cutter tool cannot be defined as slave.	Make this assignment for a master tool.
TE043	**Too many slaves**	It is not possible to definse another slave for the multi-cutter tool (max. 2).	Check and correct the master tool.
TE044	**Warning multi cutter tool**	This warning indicates that the multi-cutter tool has been modified.	This is not an error message but a warning.
TE045	**Warning reserved pocket overlap**	The selected tool occupies a reserved tool.	If the tool is confirmed the error will be ignored but the pocket will be in any case overlapped.
TE046	**Pocket less than or equal to start**	The final pocket must be greater than or equal to the initial one when the pocket state in the magazine is altered.	Check and correct the final magazine pocket.
TE047	**Illegal tool offset**	The tool offset number must be less the number of configured offsets.	Check and correct the tool offset number.
TE048	**Tool code already existing**	There is another tool with the same name in the database. Tool codes must be assigned on a one-to-one basis.	Check and correct the tool code in the database.
TE049	**Illegal character in file name**	There is an illegal character in the input filename.	Check and correct the input file name.
TE050	**Offset file illegal size**	The length of the tool offsets file is illegal.	Delete the file with DOS SHELL.
TE051	**User stop print request**	The operator has aborted the printing session by pressing **[Esc]**.	---
TE052	**Illegal double format for editor**	The format of this variable is not accessible to the editor (max 5.5).	Check and correct the variable.

Code and message		Description	Remedial action
TE053	**Illegal magazine number into file**	The magazine number specified in the Tool file refers to a non-existing magazine.	Delete the file with DOS SHELL.
TE054	**Too many magaziner into file**	In the Tool File there are tools associated to different magazines.	Delete the file with DOS SHELL.
TE055	**Active and save file has same name**	SAVE MEM is not allowed on the current file.	Abort the operation.
TE058	**Pocket not initialised**	Pocket assigned to tool has not been initialised.	Initialise Pocket and reassign to tool.
TE059	**Pocket not compatible**	Pocket assigned to tool is not compatible with type of tool.	Assign Pocket compatible with tool.
TE101	**Same axis name**	Two axes configured in the same process has been given the same name.	Check and correct the axes names assigned in the process configuration.
TE102	**Error writing configuration file**	This error occurs when the configuration file is saved with a SAVE command.	Contact the Technical Assistance service.
TE103	**Error reading configuration file**	This error occurs when the configuration file is read with a LOAD command.	Contact the Technical Assistance service.
TE104	**Same name of default configuration**	The TBC_CONF name cannot be used for saving the configuration file because it has been used by the configuration for other purposes.	Modify the file name.
TE105	**Configuration file doesn't exist**	The configuration file specified in the LOAD command does not exist.	Check and correct the name of the configuration file.
TE106	**Wrong length of file**	The length of the configuration file specified in the LOAD command is illegal.	Check and correct the name of the configuration file.
TE107	**Reset operation failed**	Error during configuration reset. An internal configuration file is probably corrupt.	Reinstall the release o contact the Technical Assistance service.
TE108	**Value less than or equal to min value**	The maximum allowed value for a field cannot be less than the minimum configured value for that field.	Check and correct the maximum value.
TE109	**Value out of range**	The maximum or minimum value defined for the field is beyond the allowed range for the number of digits in that field.	Check and correct the maximum or minimum field value.
TE110	**Field type mismatch**	The fields matched with the LINK function must belong to the same class.	Check and correct the fields.
TE111	**Undefined sort**	The selected sorting criteria has not been defined.	Check and correct the selected sorting criteria.
TE112	**Files compilation error**	Compilation error in the last translated configuration file.	Contact the Technical Assistance service after saving the configuration file. Supply the configuration file to the Technical Assistance service.
TE113	**File not found**	Internal working files not found.	Reinstall the release or contact the Technical Assistance service.
TE114	**Write error**	Disk writing error while generating the new configuration.	Contact the Technical Assistance service.
TE115	**Read error**	Disk reading error while activating the new configuration.	Contact the Technical Assistance service.
TE116	**Value must be greater than zero**	The maximum and minimum values defined for this field must be positive, i.e. greater than zero.	Check and correct the field value.
TE117	**Incomplete sort**	As many as 2 fields can be defined with a sorting criteria, or the sorting headline has not been defined.	Check and correct the number of fields or the sorting headline.
TE118	**Sort field can be maximum 6**	The number of fields defined for table sorting cannot be greater than 6.	Check and correct the number of sorting fields
TE119	**Sort number must be less than 20**	The number of fields defined for sorting is greater than maximum value (20).	Modify a previous sorting criteria rather than deleting it or creating a new one.
TE120	**No field link change allowed**	This field cannot be altered and the link to another field cannot be removed.	---
TE121	**Impossible open directory window**	There is not enough memory for opening the file selection window in LOAD and SAVE operations.	Contact the Technical Assistance service.
TE122	**No files on directory**	No configuration files have been saved.	This is not an error message but a warning.
TE123	**Illegal char in file name**	There is an illegal character in the file name.	Check and correct the file name.

Code and message		Description	Remedial action
TE124	**Error reading directory files**	This error occurs while reading the directory files.	Contact the Technical Assistance service.
TE125	**Help config not allowed on field**	Help messages cannot be altered in the selected field.	---
TE126	**Unlink operation non allowed**	The selected field cannot be used for unlinking matched items.	---
TE127	**Too many digit (integer + decimal)**	This number may have as many as 10 digits.	Correct the input value.
TE201	**The variables limits are wrong**	There is overlapping between elements belonging to two different tables.	Check and correct the number of variables in both tables.
TE202	**Error writing configuration file**	This error occurs while writing the configuration file.	Contact the Technical Assistance service.
TE203	**Error reading configuration file**	This error occurs while reading the configuration file.	Contact the Technical Assistance service.
TE204	**Same name of default configuration**	A file name reserved for the editor has been selected.	Check and correct the file name.
TE205	**The configuration file not exist**	The configuration file cannot be found.	Contact the Technical Assistance service.
TE206	**Wrong length of file**	The length of the configuration file is illegal: the file is corrupt or has not been generated by the editor.	Contact the Technical Assistance service.
TE207	**Table modified during Data Entry (R)etry (I)gnore ?**	User variables have been modified from PLUS during a data entry session. The visualised values are therefore not valid.	Edit the table or ignore it and move to other tables.
TE208	**Table access denied (R)etry (I)gnore**	The table is write protected. The table is used by PLUS and the contents of the data entry window cannot be written on it.	Retry later or abort the operation.
TE209	**Warning : Table locked, read only**	The table is write protected. The table is used by PLUS and the visualised variables are read-only.	Retry later or abort the operation.
TE210	**User stop print request**	The operator has interrupted the printing session.	---
TE211	**I/O error. Print lost**	This error occurs during a printing session. The data transmitted to the printer have been lost and must be retransmitted from scratch.	Restart the printer. If the error recurs, contact the Technical Assistance service.
TE212	**Error file creation**	This error occurs while creating the file for saving the variables.	Contact the Technical Assistance service.
TE213	**Illegal file size**	The length of the variables saving file is illegal: the file is corrupt or has not been generated by the editor..	Contact the Technical Assistance service.
TE214	**File not found**	The variables saving file does not exist.	Contact the Technical Assistance service.
TE215	**Illegal file access**	This error occurs while accessing the variables saving file.	Contact the Technical Assistance service.
TE216	**Write error**	This error occurs while writing the variables saving file.	Contact the Technical Assistance service.
TE217	**Read error**	This error occurs while reading the variables saving file.	Contact the Technical Assistance service.
TE218	**Can't copy a table on itself (ESC)**	It is not possible to copy this table on itself.	Press **[Esc]** to exit and retry selecting another table.
TE219	**Can't copy an empty table (ESC)**	It is not possible to copy an empty table.	Press **[Esc** to exit or retry selecting another table.
TE220	**Value cannot be less than zero**	The value must be positive.	Alter the input value.
TE221	**Max value must be greater than min**	The maximum value must be greater than the minimum.	Alter the input value.
TE222	**Too many digit (integer + decimal)**	The format of this number is 10 digits.	Correct the input value.
TE223	**Value out of range**	The minimum or maximum value is out of the allowed range for this variable.	Alter the input value.
TE224	**Files compilation error**	This error occurs while configuring the configuration file.	Contact the Technical Assistance service.
TE225	**No blank line allowed**	The name of a field or table cannot be a blank.	Correct the input name.

Code and message		Description	Remedial action
TE226	**Table access denied**	This table cannot be accessed. The LOAD MEMORY or SAVE MEMORY operation has failed.	Check the requested operation and retry.
TE227	**No files on directory**	There are no Editor files in the selected directory.	This is not an error message but a WARNING.
TE228	**Error reading directory files**	Error while reading the Editor directory.	Contact the Technical Assistance service.
TE229	**Illegal char in file name**	The input file name includes illegal characters.	Check and correct the file name.
TE230	**Impossible open directory window**	There is not enough memory for handling the directories of files to be visualised.	Contact the Technical Assistance service.
TE231	**Illegal double format for editor**	The variable format can not be accessed by the editor (max 5.5).	Check and correct the variable.

DOS SHELL ERROR MESSAGES

Code and message		Description	Remedial action
DS001	**Write-protected disk**	This error occurs when writing a write-protected disk.	If the write-protected disk is the diskette, remove the write-protection and retry. If it is a remote disk writing is not allowed..
DS002	**Internal error 2401H**	DOS INT24H has been invoked with error code 001 (unit unknown) while accessing the disk.	Try another disk
DS003	**Drive not ready**	This error occurs when the diskette is not well inserted in the drive or the remote disk in inaccessible.	Re-insert the diskette in the drive or connect the remote disk and repeat the operation
DS004	**Internal error 2403H**	Internal disk error. DOS INT24H has been invoked with error code 003 (command unknown) while accessing the disk.	Try another disk
DS005	**Internal error 2404H**	Internal disk error. DOS INT24H has been invoked with error code 004 (CRC data error) while accessing the disk.	Try another disk
DS006	**Internal error 2405H**	Internal disk error. DOS INT24H has been invoked with error code 005 (requested length illegal) while accessing the disk.	Try another disk
DS008	**Internal error 2407H**	Internal disk error. DOS INT24H has been invoked with error code 007 (magnetic support unknown) while accessing the disk.	Try another disk.
DS009	**Internal error 2408H**	Internal disk error. DOS INT24H has been invoked with error code 008 (sector not found) while accessing the disk.	Try another disk
DS010	**Printer out of paper**	The printer is out of paper.	Load more paper
DS011	**Write fault error**	This error occurs when the writing operation cannot be carried out on the selected diskette.	Check that the disk is not defective and has a compatible format, and repeat the operation. If the error occurs during a printing session, check that the printer is on line.
DS012	**Read fault error**	This error occurs when reading from a diskette. DOS INT24H has been invoked with error code 0BH (DOS message: Read fault error; Abort, Retry, Fail?).	Check that the disk is not defective and has a compatible format, and repeat the operation.
DS013	**General failure error**	This error occurs when operating on a non formatted disk. DOS INT24H has been invoked with error code 0CH (DOS message: General failure error; Abort, Retry, Fail?) .	Check that the disk is not defective and has a compatible format, and repeat the operation. Also check whether the selected file is already open or is an active part program.
DS014	**Invalid softkey pressed**	The softkey pressed by the operator is not active.	Check which are the active softkeys.
DS015	**Insufficient memory**	Internal software anomaly.	Internal software anomaly.
DS016	**Unknown device**	The specified device has not been configured either as local or as remote.	Check the device connection (if the ETHERNET option is present) and repeat the operation.
DS019	**Access not allowed from this level**	The current security level does not allow access to the specified device.	Repeat the operation on another device or select a lower access level (if allowed).
DS020	**Device is not reachable/connected**	The accessed device is configured but not connected.	Retry the operation after checking whether the server is connected and working and whether the relevant service has been adequately configured.
DS021	**Use LPT1 instead of PRN**	PRN and LPT1 refer to the same printer but the system accepts only the LPT1 string.	Change PRN to LPT1 and repeat the operation
DS033	**Invalid directory path**	The specified pathname does not exist.	Check the specified pathname and repeat the operation
DS034	**Directory creation error**	The specified directory cannot be created.	Check whether the path and/or the directory exist and repeat the operation
DS035	**File not found**	The specified file does not exist.	Check whether the specified file exists and repeat the operation
DS036	**Open file error**	The specified file cannot be opened.	Check whether the specified file and the access password exist and repeat the operation

Code and message		Description	Remedial action
DS037	**Creation file error**	The specified file cannot be created.	Check the path of the specified file and the password and repeat the operation. This error can also occur when accessing the currently active part program file.
DS038	**Destination disk full**	There is not enough space in the destination diskette to save the file.	Delete unnecessary files from the diskette and repeat the operation.
DS039	**Insufficient memory for directory**	Internal software error. There is not enough space to load all the directories.	Select a subset of files, for example, *.exe.
DS040	**Too many files in directory**	There are more than 1522 files in the requested directory: the visualised directory is not complete	Select a subset of files, for example, *.exe, delete unnecessary files or move files to another directory.
DS041	**'*' and '?' not allowed**	Wildcards are not allowed in the 'pathname' field.	Remove the wildcards and repeat the operation
DS042	**Read-only file**	This message occurs when attempting to write or delete a read-only file.	Check the type of access to the specified file. Access cannot be altered by DOS SHELL.
DS043	**Directory not empty**	The specified directory cannot be deleted because it is not empty.	Delete all the files in the directory and repeat the operation
DS044	**Missing drive**	No drive has been specified.	Write the drive name with the pathname and repeat the operation
DS045	**Invalid filename**	The specified file name is not valid because it is a directory name.	Check the file name and repeat the operation
DS047	**Can't find BACKUP command**	The BACKUP command cannot be executed because the executable file is missing.	Contact the Technical Assistance service
DS048	**Can't run BACKUP command**	The BACKUP command cannot be executed because the executable file cannot be used.	Contact the Technical Assistance service
DS049	**Error running BACKUP command**	The BACKUP command ends because of an error.	Check the input parameters and repeat the command.
DS050	**Can't find RESTORE command**	The RESTORE command cannot be executed because the executable file is missing.	Contact the Technical Assistance service
DS051	**Can't run RESTORE command**	The RESTORE command cannot be executed because the executable file cannot be used.	Contact the Technical Assistance service
DS052	**Error running RESTORE command**	The RESTORE command ends by an error	Check the input parameters and repeat the command.
DS053	**Can't find FORMAT command**	The FORMAT command cannot be executed because the executable file is missing.	Contact the Technical Assistance service
DS054	**Can't run FORMAT command**	The FORMAT command cannot be executed because the executable file cannot be used.	Reinstall the release and contact the Technical Assistance service.
DS055	**Error running FORMAT command**	The FORMAT command ends because of an error.	Check the input parameters and repeat the command.
DS059	**Can't find XCOPY command**	The XCOPY command cannot be executed because the executable file is missing.	Contact the Technical Assistance service.
DS060	**Can't run XCOPY command**	The XCOPY command cannot be executed because the executable file cannot be used.	Contact the Technical Assistance service.
DS061	**Error running XCOPY command**	The XCOPY command ends because of an error.	Check the input parameters and repeat the command.
DS062	**Can't find EDIT command**	The EDIT command cannot be executed because the executable file is missing.	Contact the Technical Assistance service.
DS063	**Can't run EDIT command**	The EDIT command cannot be executed because the executable file cannot be used.	Contact the Technical Assistance service.
DS064	**Error running EDIT command**	The EDIT command ends because of an error.	Check the input parameters and repeat the command.

SECURITY ERRORS

Code and message		Description	Remedial action
PK001	**Library w.a. allocation error**	Internal software error.	Reinstall the release and contact the Technical Assistance service.
PK002	**Library w.a. not initialized**	Internal software error.	Reinstall the release and contact the Technical Assistance service.
PK003	**Softkey allocation error**	Internal software error.	Reinstall the release and contact the Technical Assistance service.
PK004	**Softkey w.a. not initialized**	Internal software error.	Reinstall the release and contact the Technical Assistance service.
PK005	**FUNCTION structure allocation error**	Internal software error.	Reinstall the release and contact the Technical Assistance service.
PK006	**Data entry w.a. allocation error**	Internal software error.	Reinstall the release and contact the Technical Assistance service.
PK007	**Data entry w.a. not initialized**	Internal software error.	Reinstall the release and contact the Technical Assistance service.
PK008	**Error w.a. allocation error**	Internal software error.	Reinstall the release and contact the Technical Assistance service.
PK009	**Error w.a. not initialized**	Internal software error.	Reinstall the release and contact the Technical Assistance service.
PK010	**Help w.a. allocation error**	Internal software error.	Reinstall the release and contact the Technical Assistance service.
PK011	**Help w.a. not initialized**	Internal software error.	Reinstall the release and contact the Technical Assistance service.
PK012	**Open file error**	This error indicates an anomaly when opening a file used by the Security utility.	Reinstall the release and contact the Technical Assistance service.
PK013	**Read file error**	This error occurs indicates an unsuccesfull reading operation of a Security file.	Reinstall the release and contact the Technical Assistance service.
PK014	**Write file error**	This error indicates an unsuccesfull writing operation of a Security file.	Reinstall the release and contact the Technical Assistance service.
PK015	**Seek error**	This message indicates a file positioning error during a reading or writing operation.	Reinstall the release and contact the Technical Assistance service.
PK016	**SECURITY CORRUPTION: reinstall release**	This error occurs when the password file does not exist or its checksum is not correct.	Reinstall the release.
PK017	**Wrong password confirmation**	This error occurs when the confirmed password is different from the new one.	Reinstall and confirm the new password.
PK018	**Invalid password**	This error occurs when the input password for selecting a new security level or defining a new password is different from the confirmed one.	Rewrite the password.
PK019	**Level must be higher than current**	This error occurs when the security level selected in the SET SECURITY window is lower than the current one.	Alter the current security level with the SET LEVEL function and select SET SECURITY again.
PK020	**Wrong keys syntax**	This error occurs when the three product keys defined in the PRODUCT KEYS data entry window include illegal characters. (Only capital alphanumeric characters are allowed).	Rewrite the product keys.
PK021	**Wrong product keys**	This error occurs when the three product keys written in the PRODUCT KEYS data entry window are not correct.	Rewrite the keys. If the error recurs contact the Technical Assistance service.
PK022	**Security violation**	This error occurs after three consecutive erroneous passwords have been written in the SET LEVEL or SET PASSWORD data entry windows or after three consecutive erroneous product keys have been written in the PRODUCT KEYS data entry window.	Select the data entry window again and key in the correct passwords or product keys.
PK023	**Security violation: turn system off**	This error occurs when the hardware product key is not correct.	Contact the Technical Assistance service.
PK024	**Set not permitted for this level**	This error occurs when the operator tries to modify the 0 level password.	This operation is illegal.

Code and message		Description	Remedial action
PK025	**General failure! Press ENTER to continue, ESC to abort**	This error occurs when installing an optional package.	Check the diskette and repeat the installation.
PK026	**Drive not ready! Press ENTER to continue, ESC to abort**	This error occurs when the operator repeats the installation of the optional package but the diskette is not well inserted in the drive.	Re-insert the diskette and re-confirm the installation.
PK027	**Wrong floppy disk. Reinsert the disk and press ENTER to continue**	This error occurs when the diskette of the optional package is not an *OSAI* diskette.	Re-insert an appropriate diskette and press ENTER.
PK028	**Only 0 and 1 are permitted**	This error occurs when the operator writes digits other than 0 and 1 in the bit map of the SET SECURITY data entry window.	Check and correct the values written in the bit map of the SET SECURITY data entry window.
PK029	**Disk D: access denied**	Internal software error.	Reinstall the release and contact the Technical Assistance service.

EMERGENCY MESSAGES

Code and message		Description	Remedial action
EE001	**Servo error axis**	Positioning or following error greater than the configured value. Non-locking error (ERRO status).	Exit with reset.
EE002	**Skew Error Axis**	Split axes positioning or following error greater than the configured value. Non-locking error (ERRO status).	Exit with reset.
EE003	**Encoder phase error**	Erroneous encoder configuration. Non-locking error (ERRO status).	Exit with reset.
EE004	**Timeout tolerance error**	The axis is out of tolerance after the specified time. Non-locking error (ERRO status).	Exit with reset.
EE005	**Inpos tolerance error**	The axis is out of tolerance. Non-locking error (ERRO status).	Exit with reset.
EE006	**Broken wire detection**	Broken wire detected in the transducer cable. Locking error (EMERG status).	Switch off the system.
EE007	**Watch dog axis**	The axes boards have not been refreshed in the specified time. Locking error (EMERG status).	Switch off the system.
EE008	**Divide by zero axis**	Calculation error (division by zero). Locking error (EMERG status).	Switch off the system.
EE009	**Invalid command axes**	Request not valid for this board. Locking error (EMERG status).	Switch off the system.
E010	**Escape found axis**	Coprocessor error. Locking error (EMERG status).	Switch off the system.
EE011	**Software overtravel limit with HPG**	This emergency is given by the axis board when the axis reaches the software travel limit. Non-locking error.	Jog the axis in the opposite direction.
EE012	**HPG software overtravel exit**	The axis board generates this message to indicate that the axis has returned within the software travel limit.	---
EE013	**WatchDog - CAN I/O module**	CANOpen I/O devices have not been refreshed within the allotted time. Locking error (EMERG status)	Switch off the system
EE014	**CAN communication error**	Transmission or reception error has occurred on CANOpen bus	Check connections and turn off the system
EE015	**WatchDog - FastWire I/O module**	FastWire I/O devices have not been refreshed within the allotted time. Locking error (EMERG status)	Switch off the system
EE016	**Fastwire communication error**	Transmission or reception error has occurred on Fastwire bus.	Check connections and plug of Fastwire bus.
EE020	**Plus shutdown request**	Machine shutdown request. Non-locking error (ERRO).	Exit with reset. **Note:** Not displayed as filtered from human interface.
EE023	**Axis Interpolator Exception**	Interpolator/Interlocking Exception. The exception is notified through an appropriate code. Locking error (EMERG status).	Switch off the system.
EE024	**Plus I/O ring failure**	I/O ring failure, or an 'O' (output) variable has been used as output parameter in a machine logic SUBROUTINE. Locking error (EMERG status).	Switch off the system or modify the machine logic.
EE025	**Plus -WinPlus Exception**	WinPlus exception. This exception is notified through an appropriate code. Locking error (EMERG status).	Switch off the system or modify the machine logic.
EE026	**Plus realtime task overrun**	The logic task has not been assigned a sampling time interval. Locking error (EMERG status).	Switch off the system.
EE027	**Axis hw overtravel limit reached**	This emergency is generated by the user logic when the axis reaches the hardware overtravel microswitch. Non-locking error (ERRO).	1) Exit with reset. 2) Jog the axis within the limits.

Code and message		Description	Remedial action
EE028	**Exit over travel hardware**	This message is generated by the logic task to indicate that the axis has returned within the hardware overtravel limits..	**Note**: It is a warning.
EE029	**Too many no wait requests from Logic**	The logic has sent too many no wait requests.	Contact the Technical Assistance service
EE030	**Ghost interrupt**	The system cannot interpret the interruption. It is a locking error (EMERG status).	Switch off the system.
EE031	**Front Panel anomaly**	General front panel failure: watchdog, coprocessor, parity error, etc. Locking error (EMERG status).	Switch off the system.
EE032	**False interrupt**	This message indicates that a false interrupt has been detected.	Contact the Technical Assistance service.
EE033	**Unrecoverable Remote I/O Failure**	An anomaly has been detected during data reception/transmission on I/O Ring.	Switch-off the system.
EE034	**Recoverable Remote I/O Failure**	An anomaly has been detected during data reception/transmission on RIO.	Try to remove the anomaly cause and reset the system.
EE049	**Axis enable request not allowed**	The request to enable an axis reaches the system while it is already performing the calculations for the movement of the axis that has not yet been enabled (e.g. the request to enable the axis by PLUS arrives after “consent for the movement”)	Eliminate the error by executing a Reset

MESSAGES FOR D.S.I. ANOMALIES

Code and message		Description	Remedial action
EE040	**Over signal distorsion on optical fibre**	An excessive distorsion of the optical fibre signal has been detected.	Switch off and then on again.
EE041	**Interrupt trasmission on optical fibre**	The optical fibre signal has been interrupted. Check the optical fibre cable and all the connections.	Switch off and then on again.
EE042	**AT Telegram missing**	Data loop communication with D.S.I. driver missing.	Switch off and on again; if the anomaly persists, contact the Technical Assistance service.
EE043	**D.S.I. Initialization phase error**	Error during initialisation phase.	Contact the Technical Assistance service.
EE044	**Anomaly on service channel**	Anomaly when sending commands from Master to D.S.I. driver.	System reset.
EE045	**Anomaly on D.S.I. command execution**	Anomaly during command execution.	Reset the system and remove the cause of the error.
EE046	**Drive shut down: class 1 diagnostic**	Error coming from "class 1 diagnostic" drive.	See the D.S.I. drive documentation.
EE047	**Shut down warning: class 2 diagnostic**	Anomaly signal non-locking coming from D.S.I. "class 2 diagnostic" driver.	See the D.S.I. drive documentation.
EE048	**D.S.I. Fault: anomaly during anomaly**	Locking error, when handling an anomaly caused by the drive.	Switch off the system and contact the Technical Assistance service.

MESSAGES ON FASTWIRE ANOMALIES

Code and Message		Description	Remedial action
EE055	**Axes board memory test failed**	Axes board OS8531 failure detected by hardware diagnostics at CNC power-up.	Fit new OS8531 board.
EE056	**Drive/Bridge not connected**	At power-up, CNC did not find on the Fastwire bus either an OS3 drive or a Bridge matching the address configured in AMP for the axis specified.	Check whether address on drive or Bridge corresponds to the one configured in AMP, or check wiring of Fastwire communication cable.
EE057	**Drive command error**	Following transmission of a command, CNC received an error message from the drive; or an anomaly occurred in management of command.	Check drive integrity and parameters, if anomaly persists, contact the Technical Assistance service.
EE058	**Drive alarm**	Alarm signal given out by the drive associated with the axis specified.	Eliminate cause of drive alarm and RESET system.
EE059	**Drive fault**	Severe error notified by the drive associated with the axis specified.	Check drive integrity and parameters, if anomaly persists, contact the Technical Assistance service.
EE060	**Drive initialization error**	Drive initialization not completed successfully during power-up.	Check drive integrity and parameters, if anomaly persists, contact the Technical Assistance service.

RESERVED MESSAGES

The emergency messages coded from EE050 to EE054 are reserved.

HARDWARE DIAGNOSTICS ERRORS

Base unit (main CPU)

Code and message		Description	Remedial action
BD001	**PRODUCT KEY checksum failed**	Product key checksum error.	Contact the Technical Assistance service.
BD002	**CMOS memory battery back up low**	The backup battery of the dual port memory is low and the data stored in the CMOS may be lost.	Leave the control on 24 hours. If the message recurs, contact the Technical Assistance service.
BD003	**Disk boot failure**	DOS cannot be loaded from the hard disk.	Check that there is no floppy disk present in the drive. If the error persists, call the customer engineering service.
BD004	**Fixed Disk: 1 Present**	Indicates that the HDU is present.	---
BD004	**Floppy Disk: 1 Present**	Indicates that the floppy disk unit is present.	---
BD004	**Fixed Disk Not Present**	Indicates that the HDU is not responding.	Contact the Technical Assistance service.
BD004	**Floppy Disk Not Present**	Indicates that no floppy disk unit is present.	---
BD004	**1780-Disk 0 failure 1782-Disk controller failure 1790-Disk 0 error**	A failure has occurred on the HDU or the HDU controller.	Contact the Technical Assistance service.
BD011	**Serial configuration error**	Hardware anomaly. Serial port configuration error.	Contact the Technical Assistance service.
BD012	**Print configuration error**	Hardware error. Printer configuration error.	Contact the Technical Assistance service.
BD013	**Internal Serial Port Fail**	Serious hardware error. Serial; port failure.	Contact the Technical Assistance service.
BD014	**Internal Parallel Port Fail**	Hardware error. Parallel port failure.	Contact the Technical Assistance service.
BD016	**Non-system disk or disk error**	The diskette in drive A has not been formatted as a system disk.	Remove or replace the diskette
BD017	**Virtual mode cannot be accessed**	Serious hardware error. Access to virtual mode is not allowed.	Contact the Technical Assistance service.
BD020	**Address line error**	Serious hardware error. The line address is not correct.	Contact the Technical Assistance service.
BD028	**Clock stopped**	Hardware anomaly. The clock is locked.	Contact the Technical Assistance service.
BD029	**Clock/Calendar error**	Hardware error. Clock/calendar error.	Contact the Technical Assistance service.
BD032	**CMOS RAM checksum error**	Serious hardware error. CMOS RAM checksum error.	Contact the Technical Assistance service.
BD033	**CMOS video configuration error**	Software error. Video CMOS configuration error.	Contact the Technical Assistance service.
BD034	**Motherboard video jumper error**	Software anomaly. The set up of the motherboard video jumpers is not correct.	Contact the Technical Assistance service.
BD035	**Base memory configuration error**	Serious hardware error. base memory configuration error.	Contact the Technical Assistance service.
BD036	**CMOS base memory size error**	Serious hardware error. The size of the CMOS RAM memory is not correct.	Contact the Technical Assistance service.
BD037	**CMOS extended memory size error**	Hardware error. The size of the CMOS RAM extended memory is not correct.	Contact the Technical Assistance service.
BD038	**Turn key to unlocked position**	Software error. Request to set the key to the unlocked position.	Contact the Technical Assistance service.
BD039	**Fixed disk present**	Signals that the HDU is present.	---
BD043	**Option ROM found**	Signal that the optional ROMs are present.	---
BD045	**Option ROM checksum error**	Serious hardware error. Optional ROM checksum error.	Contact the Technical Assistance service.
BD050	**R/W disk error**	Serious hardware error. Disk reading/writing error.	Contact the Technical Assistance service

Code and message		Description	Remedial action
BD052	**Disk controller failure**	Serious hardware error. Disk controller failure.	Contact the Technical Assistance service
BD053	**R/W disk 0 error**	Serious hardware error. Disk 0 reading/writing error.	Contact the Technical Assistance service
BD055	**Parity error on main circuit board**	Serious hardware error. Parity error on the base unit board.	Contact the Technical Assistance service.
BD056	**Parity error on expansion board**	Serious hardware error. Parity error on the expansion board.	Contact the Technical Assistance service.
BD057	**Fatal error - system stopped**	Serious error. Locked system.	Contact the Technical Assistance service.
BD061	**Dedicated memory error**	Hardware anomaly. Serious error detected during extended memory test.	Contact the Technical Assistance service.
BD062	**Dedicated memory fail**	Hardware anomaly. Serious error detected during extended memory test.	Contact the Technical Assistance service.
BD063	**Math processor (I80387) present**	Indicates that the math processor is present.	
BD065	**Video configuration error**	Software error. Signals a video configuration error.	Contact the Technical Assistance service.
BD066	**CMOS RAM battery failure**	Hardware error. CMOS RAM backup battery failure.	Contact the Technical Assistance service.
BD101	**CMOS memory battery back up passed**	The CMOS RAM backup battery has passed the test.	---
BD102	**DOS primary boot strap**	Indicates that the system starts loading DOS in memory.	---
BD103	**Non system disk or disk error**	The diskette in drive A has not been formatted as a system disk.	Remove or replace the diskette
BD104	**Remove the diskette and strike any key**		Remove the diskette and strike any key to continue.
BD105	**PRODUCT KEY checksum passed**	The product key checksum is correct.	---

Other boards

Code and message		Description	Remedial action
HD001	**Configuration file not found**	Configuration file generated through AMP has not been found.	Execute AMP to check for errors, then enter the parameters again. If this operation fails again, contact our technical service.
HD002	**Dispatcher file not found**	The .EXE file is missing and the board initialisation manage cannot be executed. Software error.	Reinstall the release: if the problem persists, contact our technical service.
HD003	**No configured boards**	There are no software configured boards in the system.	Reinstall the release: if the problem persists, contact our technical service.
HD004	**System time out on memory**	Serious error. System timeout during initialisation inspection of the amount of memory available of the boards. Hardware anomaly.	Contact the Technical Assistance service.
HD005	**Diagnostic time out**	Serious error. Board timeout during execution of the initialisation diagnostics. Hardware anomaly.	Contact the Technical Assistance service.
HD006	**Time out on card bootstrap.**	Serious error. Board timeout. Hardware anomaly.	Contact the Technical Assistance service.
HD007	**PAK not found**	Product Authorization keys have not been found	Enter the PAK's
HD021	**Dual port error**	Serious error. Hardware error on the dual port memory.	Contact the Technical Assistance service.
HD022	**File not found**	The .EXE file to be loaded on the board does not exist..	Reinstall the release and contact the Technical Assistance service.
HD023	**Path not found**	The pathname of the .EXE file to be loaded on the board does not exist.	Reinstall the release and contact the Technical Assistance service.

Code and message		Description	Remedial action
HD024	**File format error (1)**	This error occurs when the system reads the .EXE file to be loaded on the board. It returns from the DOS 0X3F INT21H function.	Reinstall the release and contact the Technical Assistance service.
HD025	**Program too large**	This internal software error occurs when the .EXE board initialisation file is larger than the dual port memory.	Reinstall the release and contact the Technical Assistance service.
HD026	**Timeout on download**	No response from the board after loading the initialisation file.	Reinstall the release and contact the Technical Assistance service.
HD027	**Software error**	This error message is not used by the board initialisation firmware.	Reinstall the release and contact the Technical Assistance service.
HD028	**Insufficient system memory**	This error message is not used by the board initialisation firmware. There is not enough memory for the system to load the requested configuration.	Contact the Technical Assistance service.
HD029	**File format error (2)**	The file to be loaded is not a .EXE file.	Reinstall the release and contact the Technical Assistance service.
HD030	**No board memory to load project**	There is not enough memory for the system to load the requested configuration.	Contact the Technical Assistance service.
HD042	**Dual port has been reset**	The dual port did not have correct data and has been reset. This message indicates that the reset operation has been terminated.	---
HD049	**RAM check error**	Hardware error during RAM board diagnostics. The message indicates that the diagnostics has been completed and an error has been found.	Contact the Technical Assistance service.
HD050	**DMA error**	Serious error. Hardware error during board DMA diagnostics.	Contact the Technical Assistance service.
HD051	**NPX errror**	Serious error. Hardware error during board mathematic coprocessor diagnostics.	Contact the Technical Assistance service.
HD052	**Time unit error**	Serious error. Hardware error during board timer diagnostics.	Contact the Technical Assistance service.
HD053	**Erdc error**	Serious error. The transducer is damaged or the axes boards have been configured as having more channels than those present in the system.	Contact the Technical Assistance service.
HD061	**Loaded bootstrap program**	This is not an error message but a warning. It indicates that loading of the initialisation program has terminated and no error have been detected.	---
HD062	**Check ram**	This is not an error message but a warning. It indicates that the RAM verification has been completed.	---
HD063	**Loaded diagnostic**	This is not an error message but a warning. It indicates that the diagnostic program has been loaded.	---
HD064	**Ethernet board**	The system has detected an Ethernet board.	---

SOFTWARE DIAGNOSTICS ERRORS

Operating system

Code and message		Description	Remedial action
OD001	**Operating system loaded**	This is not an error message but a warning. It indicates that the operating system has been loaded.	---
OD002	**Operating system loading**	This is not an error message but a warning. It indicates that the operating system is being loaded.	---
OD003	**Application loading**	This is not an error message but a warning. It indicates that an application program is being loaded.	---
OD011	**MINI DNC not installed local boot**	Remote controller configured but MINI DNC not installed. Boot continues without controller	Install MINI DNC or remove configuration of bootstrap remote controller
OD012	**Channel disabled,local boot**	Bootstrap remote controller configured but communication channel disabled. Boot continues without controller	Reactivate communication channel or remove configuration of bootstrap remote controller
OD020	**RTSCOPE user alias file error**	The file is corrupt.	Check the file contents and contact the Technical Assistance service.
OD021	**Incorrect DOS version**	The DOS version is not compatible with the system.	Reinstall the release.
OD022	**Disk boot failure**	The initialisation software of the system disk is corrupt.	Reinstall the release and contact the Technical Assistance service.
OD023	**Missing OSYDRIVD.SYS**	System disk failure. The OSYDRIVD.SYS file has been lost.	Reinstall the release and contact the Technical Assistance service.
OD024	**Configuration file error**	System disk failure. The configuration file has been lost.	Reinstall the release and contact the Technical Assistance service.
OD025	**Operating system load**	The operating system cannot be loaded.	Reinstall the release and contact the Technical Assistance service.
OD030	**EMERGENCY TEST missing**	The Emergency Test program cannot be executed.	Reinstall the release and contact the Technical Assistance service.
OD040	**FRONT PANEL LOADER missing**	The operator panel initialization program has been lost.	Reinstall the release and contact the Technical Assistance service.
OD042	**EMERGENCY PROGRAM not loadable**	The Emergency Program cannot be loaded.	Reinstall the release and contact the Technical Assistance service.
OD043	**HARWARE DIAGNOSTIC not loadable**	The hardware diagnostics program cannot be loaded.	Reinstall the release and contact the Technical Assistance service.
OD044	**OPERATING SYSTEM not loadable**	The operating system programs cannot be loaded.	Reinstall the release and contact the Technical Assistance service.
OD045	**OPERATING SYSTEM CONFIGURATION missing**	The operating system configuration has been lost.	Reinstall the release and contact the Technical Assistance service.
OD046	**MULTITASKING can't be started**	System disk error.	Reinstall the release and contact the Technical Assistance service.
OD047	**FILE SERVER not bootable**	The file server cannot be initialised.	Reinstall the release and contact the Technical Assistance service.
OD048	**APPLICATION LOADER missing**	The application loader program has been lost.	Reinstall the release and contact the Technical Assistance service.
OD049	**SOFTWARE DIAGNOSTIC not allowed**	The wrong key has been pressed..	Release the key and restart the system.
OD050	**OPERATING SYSTEM LOADER missing**	The operating system loader program has been lost.	Reinstall the release and contact the Technical Assistance service.
OD051	**OPERATING SYSTEM missing**	The operating system program has been lost.	Reinstall the release and contact the Technical Assistance service.
OD060	**COMMAND FILE not found**	The commands file program has been lost.	Reinstall the release and contact the Technical Assistance service.

Code and message		Description	Remedial action
OD070	**E44/E47 option not allowed**	The MINIDNC option cannot be used. The hardwre security key is not allowed.	Insert the allowed PAK.
OD071	**Can't install net**	The system configuration is not compatible with the MINIDNC option.	Check the installed options and contact the Technical Assistance service.
OD072	**Invalid protocol selection**	A MINI DNC file has been lost or is corrupt.	Reinstall the MINI DNC option and contact the Technical Assistance service.
OD100	**RS232 software loading**	This is not an error message but a warning. It indicates that the network software of the MINI DNC option is being loaded.	---
OD101	**SK-Netbios software loading**	This is not an error message but a warning. It indicates that the network software for the SK-Netbios protocol of the MINI DNC Ethernet option is being loaded.	---
OD102	**DEC-Netbios software loading**	This is not an error message but a warning. It indicates that the network software for the DEC-Netbios protocol of the MINI DNC Etehrnet option is being loaded.	---
OD103	**E65 option not allowed**	The Serial MINI DNC option cannot be used. The hardwre security key is not allowed.	Insert the allowed PAK.
OD104	**E66 option not allowed**	The MINI DNC Ethernet option cannot be used. The hardwre security key is not allowed.	Insert the allowed PAK.
OD105	**E67 option not allowed**	The Ethernet Communication Remote Control option cannot be used. The hardwre security key is not allowed.	Insert the allowed PAK.
OD106	**Local station Network Name not found**	Bootstrap remote controller manager software error	Reinstall release and call for technical assistance
OD107	**Add Name - NetBios error: xx**	Bootstrap remote controller manager software error	Reinstall release and call for technical service
OD108	**Listen - NetBios error: xx**	Bootstrap remote controller manager software error	Reinstall release and call for technical service
OD109	**Hang Sessione - NetBios error: xx**	Bootstrap remote controller manager software error	Reinstall release and call for technical service
OD110	**DelName - NetBios error: xx**	Bootstrap remote controller manager software error	Reinstall release and call for technical service
OD111	**MINIDNC error COM1 not configured in AMP**	This is not an error message but a notice that COM1 is not configured in AMP	Configure COM1 in AMP
OD112	**MINIDNC error COM2 not configured in AMP**	This is not an error message but a notice that COM2 is not configured in AMP	Configure COM2 in AMP
OD113	**See Fig. 1**	This is not an error message but a notice that the system is operating as a network server	Reinstall release and call for technical service
OD116	**Receive - NetBios error: xx**	Bootstrap remote controller manager software error	Reinstall release and call for technical service
OD117	**Send session - NetBios error: xx**	Bootstrap remote controller manager software error	Reinstall release and call for technical service
OD118	**Unknow request from remote station**	Bootstrap remote controller manager software error	Reinstall release and call for technical service

OD113

NETWORK EMERGENCY MODE
This station is working as a Network Server.

Fig. 1

Software Configuration Errors

Code and message		Description	Remedial action
SD000	**PLUS environment creation**	This is not an error message but a warning that indicates that creation of the PLUS environment is terminated.	---
SD000	**Servo environment creation**	This is not an error message but a warning that indicates that creation of the SERVO environment is terminated.	---
SD000	**Process 1 environment creation**	This is not an error message but a warning that indicates that creation of the PROCESS 1 environment is terminated.	---
SD000	**Process 2 environment creation**	This is not an error message but a warning that indicates that creation of the PROCESS 2 environment is terminated.	---
SD000	**Process 3 environment creation**	This is not an error message but a warning that indicates that creation of the PROCESS 3 environment is terminated.	---
SD000	**Process 4 environment creation**	This is not an error message but a warning that indicates that creation of the User Interface environment is terminated.	---
SD000	**H-I environment creation**	This is not an error message but a warning that indicates that creation of the User Interface environment is terminated.	---
SD000	**OEM environment creation**	This is not an error message but a warning that indicates that creation of the OEM environment is terminated..	---
SD011	**PLUS error file not found**	The file containing PLUS errors has not been found.	Reinstall the release and contact the Technical Assistance service.
SD012	**PLUS loader file not found**	The PLUS loader file has not been found.	Reinstall the release and contact the Technical Assistance service.
SD013	**Servo error file not found**	The file containing servo errors has not been found.	Reinstall the release and contact the Technical Assistance service.
SD014	**Servo loader file not found**	The servo loader file has not been found.	Reinstall the release and contact the Technical Assistance service.
SD015	**Process error file not found**	The file containing process errors has not been found.	Reinstall the release and contact the Technical Assistance service.
SD016	**Process loader file not found**	The process loader file has not been found.	Reinstall the release and contact the Technical Assistance service.
SD017	**Human error file not found**	The file containing user interface errors has not been found.	Reinstall the release and contact the Technical Assistance service.
SD018	**Human loader file not found**	The user interface loader file has not been found.	Reinstall the release and contact the Technical Assistance service.
SD019	**OEM error file not found**	The file containing OEM errors has not been found.	Reinstall the release and contact the Technical Assistance service.
SD020	**OEM loader file not found**	The OEM loader file has not been found.	Reinstall the release and contact the Technical Assistance service.
SD021	**AMP loader file not found**	The AMP loader file has not been found.	Reinstall the release and contact the Technical Assistance service.
SD101	**AMP file not found**	The PLUS AMP utility has not been found.	Contact the Technical Assistance service.
SD102	**pl_confax error**	This error occurs during axes configuration.	Contact the Technical Assistance service.
SD103	**pl_putvar error**	This error occurs during assignment of PLUS variables.	Contact the Technical Assistance service.
SD104	**Logic information file not found**	The file containing information about the PLUS logic has not been found.	Contact the Technical Assistance service.
SD105	**EXE information file not found**	The executable file has not been found.	Contact the Technical Assistance service.
SD106	**Error opening i/o configuration file**	This error occurs when opening an I/O configuration file.	Contact the Technical Assistance service.
SD107	**Too many I/O for configured board**	Too many I/O requests for the configured board..	Contact the Technical Assistance service.
SD108	**Error reading i/o configuration file**	This error occurs when reading an I/O configuration file.	Contact the Technical Assistance service.

Code and message		Description	Remedial action
SD109	**Error reading PLUS hw description file**	This error occurs when reading the PLUS hardware configuration file.	Contact the Technical Assistance service.
SD110	**pl_checkhw error**	This error occurs during the hardware PLUS test.	Contact the Technical Assistance service.
SD111	**Error on axis definition**	Axis definition error.	
SD112	**Error loading table**	Table loading error.	Contact the Technical Assistance service.
SD113	**Error on end configuration**	Error detected when PLUS configuration is complete.	Contact the Technical Assistance service.
SD114	**Too many MTB PANEL for the configurated board**	Too many MTB panels for the configured board.	Contact the Technical Assistance service.
SD115	**Too many analog device for the configurated board**	Too many analog devices for the configured board.	Contact the Technical Assistance service.
SD116	**Local I/O board(s) not present or not working**	The local I/O board is missing or is not working.	Contact the Technical Assistance service.
SD117	**Reboot enabled (local watch dog disabled)**	On-line loading of the PLUS logic is enabled.	This is not an error message but a warning that the PLUS watchdog is disabled.
SD121	**Dual Port error**	Dual port memory malfunctioning.	Contact the Technical Assistance service.
SD122	**File not found**	The specified file does not exist.	Re-create the PLUS environment or contact the Technical Assistance service.
SD123	**Path not found**	The specified patname does not exist.	Re-create the PLUS environment or contact the Technical Assistance service.
SD124	**File format error (1)**	The file format is illegal or the file is corrupt.	Re-create the PLUS environment or contact the Technical Assistance service.
SD125	**Program too large**	The program is too large.	Reduce the file length and re-create the logic.
SD126	**Timeout on download**	Program download to board has reached the timeout.	Contact the Technical Assistance service.
SD129	**File format error (2)**	The file is corrupt or is not a .EXE file.	Contact the Technical Assistance service.
SD130	**Insufficient board memory for loading**	There is not enough memory board to load the program.	Reduce the logic program length or contact the Technical Assistance service.
SD141	**PLUS board O.S. configuration error**	This error occurs during PLUS board configuration.	Contact the Technical Assistance service.
SD142	**Insufficient memory to run project**	There is not enough memory on the PLUS board to load and handle the user logic.	Reduce the logic program length or contact the Technical Assistance service.
SD143	**I/O ring: wrong number of devices**	The number of on-line devices is different than the configured one.	Check configuration values and repeat the operation.
SD144	**I/O ring: wrong module code**	The device found in a ring position is different than the configured one.	Check configuration values and repeat the operation.
SD145	**I/O ring: wrong device number**	The DIP switch configurable device number is different than the configured one.	Check DIP switch settings and configuration values and repeat the operation.
SD146	**I/O ring: wrong slot map (1771)**	The 1771 slot map (hardware structure) is different than the configured one.	Check the correspodence between actual and configured module position and repeat the operation.
SD147	**I/O ring: unknown device**	The I/O ring device is unknown to the system or is inadequate for the 10 Series ring. A hardware failure has been detected on the I/O ring device.	Contact the Technical Assistance service.
SD148	**I/O ring: communication error**	TX/RX transmission over the ring is not allowed: broken wire, device not working, etc. Hardware error.	Contact the Technical Assistance service.
SD149	**I/O ring: 1771 device not manager by OS 8090**	The 1771 module cannot be managed by OS 8090. Hardware incompatibility.	Contact the Technical Assistance service.
SD150	**I/O Setup File Error**	File does not contain sections or parameters needed to configure the I/O's.	Reformulate the Setup file as necessary.
SD159	**Timeout on error management**	The PLUS board does not provide an OK or an error after the firmware or the logic have been loaded.	Contact the Technical Assistance service.

Code and message		Description	Remedial action
From SD159 to SD187	**See PLUS and WINPLUS Manuals Application in chapter CANOPEN**		
From SD190 to SD194	**See PLUS and WINPLUS Manuals Application in chapter OS-Wire**		
SD201	**AMP file not found**	The AMP axes file is missing.	Restart the AMP utility or contact the Technical Assistance service.
SD202	**EXE code not found**	The executable code is missing.	Contact the Technical Assistance service.
SD203	**Error loading code**	This error occurs during code loading.	Contact the Technical Assistance service.
SD204	**Error on axis definition**	Axes definition error.	Contact the Technical Assistance service.
SD205	**Error loading table**	This error occurs when loading calibration or compensation tables.	Contact the Technical Assistance service.
SD206	**Error on end configuration**	This message indicates that an error has been detected at configuration end.	Contact the Technical Assistance service.
SD207	**Unknow error on board**	This message indicates that an unknown board error has been detected.	Contact the Technical Assistance service.
SD208	**D.S.I. stopped at phase 2**	This is not an error message but a warning that the communication phase of the digital interface is stopped at second level.	---
SD209	**D.S.I. stopped at phase 3**	This is not an error message but a warning that the communication phase of the digital interface is stopped at third level.	---
SD210	**Continuos signal light test mode**	This is not an error message but a warning. that the standard digital interface is in "Continuos signal light" test mode	---
SD211	**Zero bit stream test mode**	This is not an error message but a warning that the standard digital interface is in “Zero bit stream" test mode	---
SD212	**Too many axes for this CNC model**	The maximum number of axes for this model has exceeded.	Verify in AMP the number configurate of axes.
SD213	**Axis board not congruent with AMP declaration**	The axis board detected by the hardware is not congruent with the AMP declaration in the "board code number" field (only for systems 110 and 510)	Check value of "board code number" in AMP.
SD214	**Loading DSI parameters**	The CNC is loading parameters in the drives with Standard Digital Interface.	---
SD215	**Verifying drives parameters**	After checking the database, the CNC is loading parameters in the OS3 drives (Fastwire interface).	---
SD216	**Drives parameterisation not valid!**	The CNC could not load the parameters in the OS3 drives (Fastwire interface).	The parameter database may contain an error, or a drive is damaged. Check the parameters again and the functionality of the drives.
SD217	**Default drives parameters loaded!**	The CNC has found that the database containing the OS3 drive parameters is not aligned with the number of digital axes characterised in AMP: one or more are missing. In this case, the parameter files of the missing axes are created and saved in the database with the default values. The new files are saved as DEFAPLxx.CSV where ‘xx’ is the ID of the axis associated with the drive. The root-table of the AXLINTAB.CSV database is updated with the newly created files.	The drives may be able to work with the default settings, but it is always advisable to perform the tuning again and update the database with optimal values.
SD301	**AMP file not found**	The AMP process file is missing.	Restart the AMP utility or contact the Technical Assistance service.
SD302	**EXE code not found**	The executable file is missing.	Contact the Technical Assistance service.

Code and message		Description	Remedial action
SD401	**AMP file not found**	The AMP human interface file is missing.	Restart the AMP utility or contact the Technical Assistance service.
SD402	**Part Program Directory error**	This message indicates an internal part program directory error.	Contact the Technical Assistance service.
SD403	**Soft key allocation error**	This error occurs during sotfkey allocation.	Contact the Technical Assistance service.
SD404	**Soft key initialization error**	This error occurs during sotfkey initialization.	Contact the Technical Assistance service.
SD405	**Plus screen not found**	The PLUS video screen is missing	Check PLUS screen configuration or contact the Technical Assistance service.
SD406	**Human task not found**	The human interface task is missing.	Contact the Technical Assistance service.
SD411	**P.P. Dir writing file error**	This error occurs when writing a file in the part program directory.	Contact the Technical Assistance service.
SD412	**P.P. Dir reading file error**	This error occurs when reading a file in the part program directory.	Contact the Technical Assistance service.
SD413	**P.P. Dir closing file error**	This error occurs when closing a file in the part program directory.	Contact the Technical Assistance service.
SD414	**P.P. Dir creating directory error**	This error occurs when creating the part program directory.	Contact the Technical Assistance service.
SD415	**P.P. Dir file number error**	This error occurs when numbering a file in the part program directory.	Contact the Technical Assistance service.
SD416	**P.P. Dir removing file error**	This error occurs when deleting a file from the part program directory.	Contact the Technical Assistance service.
SD417	**P.P. Dir memory allocation error**	This error occurs during memory allocation for the part program directory.	Contact the Technical Assistance service.
SD418	**P.P. Dir opening file error**	This error occurs when opening a file in the part program directory.	Contact the Technical Assistance service.
SD419	**P.P. Dir system variable not found**	System variable not found.	Contact the Technical Assistance service.
SD420	**P.P. Dir record length error**	The length of the record in the part program directory is not correct.	Contact the Technical Assistance service.
SD421	**P.P. Dir is being recovered**	This message indicates that the part program directory is being recovered.	This is not an error message but a warning.
SD422	**P.P. Dir has been recovered**	This message indicates that recovery of the part program directory has been completed.	This is not an error message but a warning.
SD423	**P.P. Dir is being rebuilt**	This message indicates that the part program directory is being rebuilt.	This is not an error message but a warning. Logical names are taken as physical names.
SD424	**P.P. Dir has been rebuilt**	This message indicates that part program directory rebuilding has been completed.	This is not an error message but a warning. Logical names are taken as physical names.
SD425	**P.P. Dir has been created**	This message indicates that the part program directory has been created.	This is not an error message but a warning.
SD426	**P.P. Dir too many part program**	This message indicates that there are too many part programs in the part program directory.	Delete unnecessary files from the part program directory and repeat the operation.
SD427	**P.P. Dir physical file not found**	The physical file is missing.	Contact the Technical Assistance service.
SD501	**AMP file not found**	The AMP configuration file is missing.	Restart the AMP utility or contact the Technical Assistance service.

APPLICATION ERRORS

Code and message	Description	Remedial action
AP000 **Option not installed or directory file not found**	The requested option has not been installed or has not been installed completely. With the DOS Graphic Interface (E43), check that the directory that should contain the OEM exec is present.	Install the option or contact the Technical Assistance service
AP005 **Error spawnl: File or path name not found**	The executable file that corresponds to the requested option is missing.	Re-install the option or contact the Technical Assistance service.
AP006 **Error spawnl: argument list or space required too big**	System error	Contact the Technical Assistance service.
AP007 **Error spawnl: File not executable**	The installed file is not executable.	Re-install the option or contact the Technical Assistance service.
AP008 **Error spawnl: not enough memory available**	There is not enough memory to run the requested application.	Contact the Technical Assistance service.
AP009 **Error spawnl: invalid mode flag argument**	System error.	Contact the Technical Assistance service.
AP010 **Insuff. memory to read actual environment**	There is not enough memory to read the actual environment.	Contact the Technical Assistance service.
AP011 **Error actual path too long**	System error	Contact the Technical Assistance service.
AP016 **Undefined error**	System error	Contact the Technical Assistance service.

EDITOR ERRORS

Code and message		Description	Remedial action
ED021	**<Esc> to continue. Error in file handling: arg list too long.**	DOS error during file management. The original version is restored and all the modifications of the current session are lost.	No action in the current session. Check the problem cause and use the recovery function.
ED022	**<Esc> to continue. Error in file handling: access denied.**	DOS error during file management. The original version is restored and all the modifications of the current session are lost.	No action in the current session. Check the problem cause and use the recovery function.
ED023	**<Esc> to continue. Error in file handling: bad file number**	DOS error during file management. The original version is restored and all the modifications of the current session are lost.	No action in the current session. Check the problem cause and use the recovery function.
ED024	**<Esc> to continue. Error in file handling: file exists.**	DOS error during file management. The original version is restored and all the modifications of the current session are lost.	No action in the current session. Check the problem cause and use the recovery function.
ED025	**<Esc> to continue. Error in file handling: invalid arguments.**	DOS error during file management. The original version is restored and all the modifications of the current session are lost.	No action in the current session. Check the problem cause and use the recovery function.
ED026	**<Esc> to continue. Error in file handling: too many open files.**	DOS error during file management. The original version is restored and all the modifications of the current session are lost.	No action in the current session. Check the problem cause and use the recovery function.
ED027	**<Esc> to continue. Error in file handling: no such file/dir.**	DOS error during file management. The original version is restored and all the modifications of the current session are lost.	No action in the current session. Check the problem cause and use the recovery function.
ED028	**<Esc> to continue. No free space on device.**	Not enough space on disk for saving the current file. The original version is restored and all modifications of the current version are lost.	No action in the current session. Free space on disk and use the recovery function.
ED029	**<Esc> to continue. Error in file handling: cross device link.**	DOS error during file management. The original version is restored and all the modifications of the current session are lost.	No action in the current session. Check the problem cause and use the recovery function..
ED030	**File too big.**	The file is too long to be edited.	No action (if the file cannot be reduced).
ED031	**<Esc> to continue. The recovery file can not be open.**	File recovery cannot be activated.	No action in the current session.
ED032	**Editor will abort. Fatal error.**	End of session signal. An error message more specific follows.	No action.
ED033	**Configuration error.**	An error in one of the cycle general configuration files has been detected.	Check these files.
ED034	**Too many lines.**	Source file has too much lines. The maximum allowed number of lines is 200.000 in the 10 Series text section, 32.000 in the PC text section and 400 in the graphic section (10 Series and PC).	Type a lower number of lines.
ED035	**Increment too big.**	During <sequence>, there is too much increment between the line numbers. The last line of the file could reach a value over the maximum.	Reduce the increment value.
ED036	**<Esc> to continue. Value out of range.**	The value just introduced is out of the interval.	Type a consistent value.
ED037	**Bad file name.**	The file name specified in the <include>procedure has not been found.	Change the file name.
ED038	**File incomplete.**	File terminator not found.	No action in the current session; it could work, if the file is not corrupted. Check and eventually insert the terminator in the file in which it is expressly required (control file or mimic file).

Code and message		Description	Remedial action
ED039	**Line too long.**	A line longer than the maximum has been found during file reading. The line is divided in two or more parts.	No action.
ED040	**<Esc> to continue. System error in file handling**	Generic error in file management.	No action in the current session. Check the problem cause and use the recovery function.
ED041	**Current file with recovery: (E)xecute recovery, (D)estroy recovery.**	In the previous session, the Editor has been interrupted when working on the current file; the recovery file has been created.	- (E) restores modifications temporarily lost. - (D) deletes the recovery file and allows to continue the current session.
ED042	**ABORT: error on environment file.**	Error in "config" file.	No action in the current file. Check the file when exiting from Editor.
ED043	**String not found.**	String not found during <search> procedure.	No action.
ED044	**Function not allowed on this block.**	The required function cannot be executed on the current block.	No action.
ED045	**Requested file is protected.**	The file you selected cannot be edited.	No action in the current session. Check the problem reason whwn exiting from Editor.
ED046	**Syntax error.**	Error in the current line syntax.	Delete the line error.
ED047	**Invalid number.**	The current parameter contains a not allowed value.	Modify the value.
ED048	**Undefined element**	The used element has not been defined.	Define the used element before the current line.
ED049	**Element previously defined.**	The defined element for the current line has already been defined.	Change the index of the current element or the previous one.
ED051	**Invalid axis value.**	Value not allowed for the parameter related to the axis.	Modify the parameter value.
ED052	**Too many fields in the block (max 40).**	The maximum number of the parameters related to a cycle is 40.	Reduce the parameter number.
ED053	**Invalid A.**	Value not allowed for the A parameter.	Modify the A parameter value.
ED054	**Invalid S.**	Value not allowed for the S parameter.	Modify the S parameter value.
ED055	**Invalid C.**	Value not allowed for the C parameter.	Modify the C parameter value.
ED056	**Invalid element.**	User element idetifier not known.	Correct the error.
ED057	**Not tangent element.**	During the search of the intersection of two non-tangent elements, you must specify the intersection number (S1, S2).	Insert the solution identifier at the line end.
ED059	**Incomplete line.**	The line is broken and does not contain all the requested information.	Insert the missing information.
ED060	**No space in internal structures.**	Not enough memory to complete the required operation.	Reduce the elements number.
ED061	**Profile not defined or wrong.**	Impossible to create an ISO file if the profile has not been created or is wrong.	Define the profile if missing, correct the profile if wrong or do not activate the ISO file creation.
ED062	**Not enough memory.**	This message can indicate end of memory if dinamic allocations or full internal areas are required. In the first case, check if graphic packages are not already used on the system.	In the second case, no corrections are possible.
ED063	**No intersection.**	The two or more elements required to create a new element do not intersect.	Replace one or more build elements or modify their definition.
ED064	**No coincidence between elements.**	The proposed elements cannot interact in the proposed mode.	Replace one or more build elements or modify their definition.
ED065	**Circle internal or coincident.**	Internal or coincident circumference.	Replace the build circumference or modify its definition.
ED066	**Point inside circle.**	Internal point of the circumference.	Replace one or more entities for building the current element or modify its definition.
ED067	**Parallel lines.**	Lines that must intersect are instead parallel.	Replace one or more creation lines or modify their definition.
ED068	**Wrong element.**	Not allowed element.	Replace the current element
ED069	**Points on same line.**	When creating a circumference using three points, these points cannot lie on the same line.	Replace one or more points

Code and message		Description	Remedial action
ED070	**Element does not exist.**	The element used to create another element or the profile, has not been defined.	Replace the identifier of the wrong element with an existing one, or define the element before the line that generated the error.
ED071	**Impossible radius.**	The specified joint cannot be realized.	Depending on the circumstances, modify the radius of the joint, or select other elements.
ED072	**Element out of range.**	The drawing cannot be displayed, because a too high zoom has been used.	Reduce the zoom.
ED073	**Check expression not verified.**	The value defined for the parameter does not follow the norms defined in the control files.	Modify the value introduced, or, if the field is optional, do not type any value.
ED074	**Error on default or check expression.**	Error in the default or control expression syntax.	Modify the control file of the current cycle.
ED075	**No allowed solution.**	Entities you selected cannot be used for creating a new element.	Replace one or more entities for creating the element or modify their definition.
ED076	**Impossible to enlarge again.**	It is not possible to enlarge the image on the displayed window, because the maximum definition is already used.	No action.
ED077	**Element defined after current line.**	One of the entities used for creating the current element has been defined after the current line.	Move the definition of the new element after the line of the build element or viceversa or use another element you defined before.
ED078	**Error on mathematical routine.**	Error in mathematic procedures.	Check the related expression.
ED079	**Profile error.**	Error in profile generation.	Correct the profile.
ED080	**Error opening graphic scale.**	An error has been generated when opening the graphic scale.	Verify the parameter correctness.
ED093	**Error reading configuration file.**	The default configuration is used instead of the information configured on file.	No action in the current session. When exiting from editor environment, check the ED_CONFG.DAT file.
ED094	**Error writing configuration file.**	The configuration information just defined is not saved on file.	No action in the current session. When exiting from editor environment, check the ED_CONFG.DAT file.
ED095	**Has recovery file: (E)xit from Editor, (D)elete recovery.**	In the previous session the Editor has been interrupted during editing of the <name> source file and a recovery file has been created. In the current session you try to edit a file that is not the result of the previous session. E the current execution is aborted to allow editing of the <name> source file and restore the temporarily lost modifications.	D destroys the recovery file of the <name> source file and allows to continue the current session.
ED096	**Error creating ISO file.**	Error during generation of the ISO code starting from geometry.	Delete the profile error or do not generate the profile.
ED097	**Unrecognized command in file.**	An identification command of an unknown entity has been found in the drawing file.	Delete the unknown command from the file.
ED098	**: help not initialized.**	Cycle helps are missing. They will not be displayed.	Copy the files on disk.

ERROR FROM PERIPHERALS

Code and message	Description	Remedial action
PF000 **Illegal device name**	Select device name is not applicable.	Use one of following names: COM1, COM2, LPT1, NULL. service.
PF001 **Baud rate out of range**	Baud rate as set in not among the following values: 9600, 4800, 2400, 1200, 600, 300, 150 and 110.	Change value.
PF002 **Exe net configurator file not found. Press ESC to continue...**	Exe for net configurator not avaible	Load MINI DNC.
PF007 **Netowork not installed**	Exe for network management not avaible.	Load MINI DNC.
PF008 **Channel disabled or Mini DNC not installed**	Communication channel not active or exe for network management not avaible.	Reactive communication channel or load MINI DNC
PF009 **Interlink - no device selected**	A communication line has not been configured (e.g. LPT1 or LPT2).	Configure the line using the INTERLINK-CFG softkey.
PF010 **Interlink - device not available**	The selected communication line is not available (it has already been acquired by another utility).	Select a free line or release the busy line.
PF011 **Interlink - server not available on system**	Intersvr is only available on the system when it is in “emergency” status.	Power up the system in emergency status to use Intersvr.

ERRORS COMMON TO ALL OPERATOR PANELS

Code and message		Description	Remedial action
FD001	**System anomaly**	System anomaly. Software error.	Contact the Technical Assistance service.
FD002	**Internal serial port error**	Serial port internal error. Hardware failure.	Contact the Technical Assistance service.
FD003	**System anomaly**	System anomaly. Software error.	Contact the Technical Assistance service.
FD004	**Internal parallel port error**	Parallel port internal error. Anomalia hardware.	Contact the Technical Assistance service.
FD005	**CPU (I80286)**	Serious error. CPU failure. Serious error hardware.	Contact the Technical Assistance service.
FD006	**SHUTDOWN FAILED**	Serious error. Shutdown failure. Serious error hardware.	Contact the Technical Assistance service.
FD007	**Protected mode failed**	Serious error. Anomaly during protected CPU initilisation. Serious hardware error.	Contact the Technical Assistance service.
FD009	**NMI failed**	Serious error hardware.	Contact the Technical Assistance service.
FD010	**Interrupt controller failed**	Serious error. Interrupt controller internal error. Serious error hardware.	Contact the Technical Assistance service.
FD011	**Interrupt controller failed**	Serious error. Interrupt controller internal failure. Serious error hardware.	Contact the Technical Assistance service.
FD012	**Timer Sync failed**	Serious hardware error. Timer synchronisation internal failure.	Contact the Technical Assistance service.
FD013	**DMA registers failed**	Serious hardware error. DMA register failure.	Contact the Technical Assistance service.
FD014	**DMA controller failed**	Serious hardware error. DMA controller failure.	Contact the Technical Assistance service.
FD015	**DM controller failed**	Serious hardware error. DM controller failure.	Contact the Technical Assistance service.
FD016	**Speaker failure**	Serious hardware error. Speaker failure.	Contact the Technical Assistance service.
FD018	**BUS expansion not inserted**	Serious hardware error. The expansion bus has not been inserted.	Contact the Technical Assistance service.
FD019	**Mouse present**	This message indicates that a mouse is present (this peripheral is not allowed with 10 Series CNC).	---
FD020	**Shutdown failed**	Serious hardware error. Failure during shutdown.	Contact the Technical Assistance service.
FD021	**System anomaly**	Serious error software. System failure.	Contact the Technical Assistance service.
FD022	**Parity on mother board failed**	Serious hardware error. Parity error on the mother board.	Contact the Technical Assistance service.
FD023	**Parity on expansion board failed**	Serious hardware error. Parity error on the expansion board.	Contact the Technical Assistance service.
FD024	**Fatal error - Computer stopped**	Serious hardware error. The computer is locked.	Contact the Technical Assistance service.
FD025	**Press F1 to continue**	Software failure.	Contact the Technical Assistance service.
FD026	**System anomaly**	Software error. System anomaly.	Contact the Technical Assistance service.
FD027	**Unrecoverable power up error**	Hardware anomaly. Non recoverable error at power up.	Contact the Technical Assistance service.
FD028	**Coprocessor present**	This is not an error message. It indicates that the front panel coprocessor is present.	---
FD029	**Timer counting failed**	Serious hardware error. Timer counter failure.	Contact the Technical Assistance service.
FD030	**System anomaly**	The eproms that contain the front panel firmware have not been detected by the BIOS.	Check the index of the eprom chips with the front panel firmware and whether they have been mounted.
FD031	**ROM BIOS checksum failed**	Checksum error on the eprom that contains the front panel BIOS.	Replace the eprom that contains the front panel BIOS.
FD032	**ROM firmware checksum failed**	The BIOS has detected a checksum error in the first 64K of the upper and lower front panel firmware.	Replace the eprom that contains the upper and lower front panel firmware.
FD033	**System anomaly**	Software error. System anomaly.	Contact the Technical Assistance service.
FD034	**CMOS Timer error**	Hardware error. CMOS timer anomaly.	Contact the Technical Assistance service.
FD035	**System anomaly**	Software error. System anomaly.	Contact the Technical Assistance service.

Code and message		Description	Remedial action
FD036	**Memory refresh failed**	Serious hardware error. Memory refresh cycle failure.	Contact the Technical Assistance service.
FD037	**Extended memory size report**	This is a REPORT massage that visualises the amount of extended memory available on the front panel.	---
FD039	**Virtual mode error**	Hardware error. Virtual mode management error.	Contact the Technical Assistance service.
FD040	**Base memory failed**	Serious hardware error. Base memory failure.	Contact the Technical Assistance service.
FD041	**CMOS RAM error**	Hardware error. RAM CMOS failure.	Contact the Technical Assistance service.
FD042	**Parity circuitry failed**	Serious hardware error. Parity circuitry failure.	Contact the Technical Assistance service.
FD043	**CMOS RAM error**	Hardware error. RAM CMOS error.	Contact the Technical Assistance service.
FD044	**Date and time setup**	This warning indicates that the current date and time are not correct.	Set up the date and time with the relevant softkey.
FD045	**System anomaly**	Software error. System anomaly.	Contact the Technical Assistance service.
FD046	**System anomaly**	Software error. System anomaly.	Contact the Technical Assistance service.
FD047	**Shadow memory error**	Hardware error. Shadow memory failure.	Contact the Technical Assistance service.
FD048	**System anomaly**	Software error. System anomaly.	Contact the Technical Assistance service.
FD050	**Keyboard controller failed**	Serious hardware error. Keyboard controller failure.	Contact the Technical Assistance service.
FD051	**Keyboard failed**	Serious hardware error. Internal keyboard failure.	Contact the Technical Assistance service.
FD052	**HDU 0 INIT. error**	Hardware error. Hard disk 0 initialisation error.	Contact the Technical Assistance service.
FD053	**HDU 1 INIT. error**	Hardware error. Hard disk 1 initialisation error.	Contact the Technical Assistance service.
FD054	**HDU controller error**	Hardware error. Hard disk controller failure.	Contact the Technical Assistance service.
FD055	**HDU 0 R/W error**	Hardware error. Hard disk 0 writing/reading error.	Contact the Technical Assistance service.
FD056	**HDU 1 R/W error**	Hardware error. Hard disk 1 writing/reading error..	Contact the Technical Assistance service.
FD057	**Non system disk or disk error**	The diskette in drive A has not been formatted as a system disk.	Remove or replace the diskette.
FD058	**Replace disk and strike any key**		Replace the diskette with a system disk and press any key to continue.
FD059	**Fixed disk not present**	Hardware anomaly. The hard disk has been configured but does not respond to the system.	Contact the Technical Assistance service.
FD060	**Floppy disk not present**	Hardware anomaly. The floppy disk drive does not respond to the system.	Contact the Technical Assistance service.
FD061	**Fixed disk present**	This message indicates that a hard disk is present in the system.	---
FD090	**CPU test passed**	The CPU has passed the diagnostic tests.	---
FD091	**Firmware test passed**	The ROM has passed the diagnostic tests.	---
FD092	**Memory test passed**	The memory has passed the diagnostic tests.	---
FD093	**Keyboard test passed**	The keyboard has passed the diagnostic tests.	---
FD094	**Magnetic device test passed**	The magnetic devices have passed the diagnostic tests.	---
FD095	**Waiting for rack connection max 60 sec.**	The front panel is waiting for the rack to be connected in order to complete the hardware diagnostics.	---

CONFIGURATOR ERROR MESSAGES OF NETWORKS

Code and message		Description	Remedial action
CN001	**Library w.a. allocation error**	Internal error.	Call technical assistance.
CN002	**Library w.a. not initialized**	Internal error.	Call technical assistance.
CN003	**Softkey w.a. allocation error**	Internal error.	Call technical assistance.
CN004	**Softkey w.a. not initialized**	Internal error.	Call technical assistance.
CN005	**FUNCTION structure allocation error**	Internal error.	Call technical assistance.
CN006	**Data entry w.a. allocation error**	Internal error.	Call technical assistance.
CN007	**Data entry w.a. not initialized**	Internal error.	Call technical assistance.
CN008	**Error w.a. allocation error**	Internal error.	Call technical assistance.
CN009	**Error w.a. not initialized**	Internal error.	Call technical assistance.
CN010	**Help w.a. allocation error**	Internal error.	Call technical assistance.
CN011	**Help w.a. not initialized**	Internal error.	Call technical assistance.
CN020	**Unknown device**		
CN021	**Data Base open error**	Internal error.	Call technical assistance.
CN024	**Record selection error. Press [Esc] to continue!**	Internal error.	Call technical assistance.
CN026	**Record insertion error. Press [Esc] to continue!**	Internal error.	Call technical assistance.
CN027	**Wrong protocol type**	This message is displayed when the protocol type entered is different from NETBEUI or PATHWORKS.	Correct the entered name and repeat the operation.
CN033	**Reserved device**	Use of a disk reserved to the system has been requested.	Change the disk name.
CN035	**Wrong access type**	This message appears when the access type defined for a service is different from R, W or C.	Verify the entered data and correct it.
CN038	**Wrong service path**	This error appears when the specified pathname for a local service is not correct or corresponds to a file.	Verify and correct the service pathname.
CN039	**Input file not found**	This message is displayed when the input file read by the configurator is missing.	Call technical assistance.
CN040	**Wrong address sintax**	This message is displayed when the configured address does not have the expected format.	This is valid only for the PATHWORKS protocol. The format must be XX.YYYY.
CN042	**Unknown remote drive**	It appears when the operator has entered a request to cancel a remote drive that has not been configured.	Abort the operation.

Code and message		Description	Remedial action
CN043	**Unable to connect remote drive**	This message appears when the automatic connection of a Client service fails.	Verify the hardware connections, network activation on the remote machine and the remote channel name and repeat the operation. If the error appears again, call technical assistance.
CN044	**Service not configured**	This message appears when an attempt to cancel a non configured service is made.	Verify the service name and repeat or cancel the operation.
CN046	**Segment or chan configuration incomplete, [Esc] to exit, any key to continue**	This message appears after selecting **Exit** in cases where configuration of a segment or channel is not complete.	Verify the values entered in the **CHAN GENERAL** data entry windows.
CN047	**Reserved service name**	The service name is erroneous.	Verify and correct the service name.
CN049	**File transfer not allowed**	This message appears when an attempt is made to enable the File Transfer when this is not allowed by the installed option.	Call technical assistance.
CN050	**Remote IPC not allowed**	This message appears when you try to enable the Remote IPC when this is not allowed by the installed option.	Cancel the operation.
CN060	**Wrong board type**	This message appears when the value for the configured board type is different from PCNET.	Correct the entered data and repeat the operation.
CN063	**Data base files can't be open, press any key to continue.**	Internal error.	Retry and, if the error appears again, call technical assistance.
CN071	**Max length of channel name is 6 characters**	This message appears when the length of a channel name configured with the PATHWORKS protocol is more than 6 characters.	Verify and correct the channel name.
CN072	**Logical name already defined**	Displayed when the logical name assigned a given remote environment has already been used for another one.	
CN076	**Access denied: check security**	This appears when an attempt is made to select either the *Advanced* level without having provided for the corresponding enabling bit in the Security utility	Provide the bit corresponding to the Security utility. (Byte 5, bit 4 = enables the *Advanced* level)
CN077	**Too many services declared**	It appears when the number of configured services is greater than the allowable maximum.	The maximum number of configurable services is 8.
CN078	**Environment Variable not set**	Internal error.	Re-install the option. Should the error message reappear, call the technical assistance.
CN086	**Driver not ready**	May be displayed during reading/writing on a remote disk where the latter is no longer connected.	
CN089	**General failure**		

IMPORTANT

When exiting from the configuration environment with **[Esc]**, the next network activations may not be congruent with the values displayed by the configurator because the entered data can be incomplete.

ERROR MESSAGES DURING NETWORK BOOTSTRAP

Code and message		Description	Remedial action
OD071	**Can't install net**	The network cannot be started, because the network hardware is either missing or out of order.	Verify the existance of the card. If the error persists, call technical assistance.
OD072	**Invalid channel name**	The name of the local channel is incorrect or already exists on the network.	Configure the local channel with another name.
OD101	**PATHWORKS/ NETBEUI software loading**	It is not an error but a warning that network software is being loaded.	

END OF APPENDIX

10 Series CNC
Programming Manual

Code: 45004457K
Rev. 15

PUBLICATION ISSUED BY:

OSAI S.p.A.
Via Torino, 14 - 10010 Barone Canavese (TO) – Italy

Phone: +39-0119899711

Web: www.osai.it

e-mail: sales@osai.it
service@osai.it

Copyright © 2001-2004 by OSAI
All rights reserved

Edition: December 2004

IMPORTANT USER INFORMATION

This document has been prepared in order to be used by OSAI. It describes the latest release of the product.
OSAI reserves the right to modify and improve the product described by this document at any time and without prior notice.
Actual application of this product is up to the user. In no event will OSAI be responsible or liable for indirect or consequential damages that may result from installation or use of the equipment described in this text.

SUMMARY OF CHANGES

General

This publication is issued with reference to Software Release 7.4 (E69).

PAGE	UPDATING TYPE
INDEX	Updated
CHAP. 2	
page 13	Updated description: value of CRT
page 14	Added description of function: CRT
page 33	Added new function: XDA - Master/Slave Axes
page 58	Updated function table: MOV
page 59	Added new description: meaning of bit 4.
page 97	Added description of function: UPR
page 105	Added description: example 8
page 112	Added new function: UVA
page 123	Added description: Offset mode - Table. user 366
page 136	Updated description: Offset mode - Table. user 366
CHAP. 3	
page 7	Added new function: TTR
CHAP. 4	
page 16	Updated description of function: TPO
page 19	Updated description of drawings
page 20	Updated description of drawings
page 24	Added description: mode TPO=32
CHAP. 9	
page 7	Added new functions: PMS – PMT - PMM
CHAP. 12	
page 2	Added: CLD,CLT to CLS command
page 8	Added description of commands CLD and CLT
page 13	Added: example 6
CHAP. 13	
page 9	Updated description
page 10	Updated description
page 11	Added new description: UVW Paraxial Compensation Factors
page 16	Added: UVW to Versor Programming
page 19	Added parameter: UVW and relative description
APP. B	
page 13	Added code: NC149
page 23	Added codes: NC445-NC446-NC447-NC448-NC456-NC457-NC458

PREFACE

This manual describes the procedures used for writing part programs with the 10 Series CNC system. It provides programmers with all the information they need for creating machine control programs.

REFERENCES

For further information:

- 10 Series CNC - AMP Software Characterization Manual
- 10 Series CNC - User Guide

The chapters in this manual are organised in sections. They describe the language elements (commands and functions) used for managing a specific task, e.g. axis programming, tool programming, probe management. Programming examples have been introduced in the command description.

SUMMARY

1. Programming with 10 Series System

This chapter contains the general programming rules of the International Standards Organization (ISO) standard. The chapter also provides an overview of the programming environment and a summary of the most used codes.

2. Programming the Axes

This chapter describes axis programming. The G codes and extended commands involved in this activity are provided with their characteristics. Several examples complete the command description and give suggestions for programming the major types of movements.

3. Programming Tools and tool offsets

This chapter describes tool programming and provides the functions and instructions used in tool operation.

4. Cutter Diameter Compensation

This chapter describes cutter compensation. T functions and G codes used in tool compensation are provided with characteristics and several examples.

5. Programming the Spindle

This chapter describes spindle programming. The G codes and extended commands involved in this activity are provided with their characteristics. Several examples complete the command description and give hints for solving the main cases of spindle programming.

6. Miscellaneous Functions

This chapter describes miscellaneous functions and provides a list of M functions with their meaning and characteristics.

7. Parametric Programming

This chapter deals with special programming applications that use local and system variables.

8. Canned Cycles

This chapter provides a description of the canned cycles available with the control. The G codes and extended commands used in this activity are provided with their characteristics. Several examples complete the command description.

9. Paramacros

This chapter describes how paramacros can be used in programs.

10. Probing Cycles

This chapter provides a description of the probing cycles available with the control. The G codes and extended commands involved in probe management are provided complete with examples.

11. Managing the Screen

This chapter discusses the commands used to handle the system screen from a part programs. Examples are given to complete the command description.

12. Modifying the Program Execution Sequence

This chapter contains the commands used for modifying the sequence of execution of a part program. It describes commands for branching, repeating blocks and executing subprograms, as well as commands for putting the part program on hold and releasing it.

13. High Speed Machining

This chapter describes the high-speed milling features on machine tools with 3 axes.

14. Multiprocess management commands

This chapter shows 10 Series CNC's multi process potentials.

15. High level geometric programming (GTL)

This chapter discusses the set of programming instructions available with the GTL utility.

16. Working Cycles for Turning Systems

This chapter provides the instructions for programming macro-cycles of rough-shaping, threading and groove cutting.

A. Characters and Commands

Appendix A provides a summary of all the characters allowed in the system and gives lists of G codes, mathematical functions and extended commands.

B. Error Messages

Appendix B provides a list of all the error messages that can occur during programming..

C. Error management

COMMANDS

Commands are dealt with in the chapters that describe the specific task. A common structure has been adopted in the command description. For each command, the following information is provided:

- Command name
- Command function
- Command syntax
- Parameters
- Characteristics and notes
- Examples

Where possible, examples consist of a portion of program and a diagram that shows how the commands in that portion work.

Syntax conventions

Use these conventions with the commands:

SYMBOL	MEANING
[ ]	Brackets enclose optional entries. Do not enter the brackets.
{ }	Braces enclose entries which may be repeated more than once. This could also be described as a series of alternative entries, i.e. only one of these may be entered. Alternative entries are separated by a (**|**). Do not enter the braces in the command itself.
|	A vertical bar separates alternative entries. Do not enter the bar.

Key-words are written in **bold**. They must be entered exactly as they are represented in the syntax description.
Parameters that must be passed with commands are indicated by a mnemonic written in italics. Appropriate values must be entered in place of the mnemonic. Leading zeros can be omitted. For example, you can program G00 as G, G01 as G1.

Example:
(SCF,[*value***])**
SCF, the comma and parenthesis are key-words and must be written as described. *value* is a parameter name and must be replaced by an appropriate value. The brackets indicate that *value* is an optional value.

Warnings

For correct control operation, it is important to follow the information given in this manual. Take particular care with topics bearing one of the mentions: WARNING, CAUTION or IMPORTANT, which indicate the following types of information:

Draws attention to facts or circumstances that may cause damage to the control, to the machine or to operators.

Indicates information to be followed in order to avoid damage to equipment in general.

Indicates information that must be followed carefully in order to ensure full success of the application.

Terminology

Some terms appearing throughout the manual are explained below.

Control Refers to the 10 Series numerical control unit comprising front panel unit and basic unit.

Front Panel Is the interface module between machine and operator; it has a monitor on which messages are output and a keyboard to input the data. It is connected to the basic unit.

Basic Unit Is the hardware-software unit handling all the machine functions. It is connected to the front panel and to the machine tool.

END OF PREFACE

INDEX

PROGRAMMING WITH 10 SERIES SYSTEMS

THE PROGRAM FILES 1-1
Program Components 1-2
Blocks 1-2
Block Types 1-4
Programmable Functions 1-6
G Codes 1-9
SYNCHRONISATION AND PROGRAM EXECUTION 1-13
Default Synchronisation 1-13
Overriding Default Synchronisation 1-14
Part Program Interpreter 1-14
Sequence of execution 1-15
Programming restrictions for long real (double) formats 1-15

PROGRAMMING THE AXES

AXIS MOTION CODES 2-1
Defining Axis Motion 2-1
G00 - Rapid Axes Positioning 2-2
G01 - Linear Interpolation 2-3
G02 G03 - Circular Interpolation 2-4
CET (PRC) - Circular Endpoint Tolerance 2-7
FCT - Full Circle Threshold 2-8
ARM - Defining Arc Normalisation Mode 2-9
CRT - Circular interpolation speed reduction threshold 2-13
CRK - Circular interpolation speed reduction constant 2-13
Helical Interpolation 2-15
G33 - Constant or Variable Pitch Threading 2-17
Rotary Axes 2-21
Axes with Rollover 2-23
G90 - Absolute mode 2-23
G91 - Incremental mode 2-25
Pseudo Axes 2-26
Diameter Axes 2-26
UDA - Dual Axes 2-29
SDA - Special Dual Axes 2-31
XDA - Master/Slave axes 2-33
Master/Slave Association 2-33

Syntax **2-33**
Mode 0 2-34
Mode 2 2-36
Releasing the Slave(s) from the Master 2-39
Syntax **2-39**
Defining/Changing the following ratio 2-40
Syntax **2-40**
Activating the following function 2-41
Syntax **2-41**
Deactivating the following function 2-42
Syntax **2-42**
ORIGINS AND COORDINATE CONTROL CODES **2-43**
G17 G18 G19 - Selecting the Interpolation Plane 2-44
G16 - Defining the Interpolation Plane 2-45
G27 G28 G29 - Defining the Dynamic Mode 2-46
AUTOMATIC DECELERATION ON BEVELS IN G27 MODE **2-51**
DLA - Deceleration Look Ahead 2-52
DYM - Dynamic Mode 2-53
MDA - Maximum Deceleration Angle 2-54
VEF - Velocity Factor 2-55
Jerk Limitation 2-57
MOV - Enable Jerk Limitation 2-58
Meaning of bits 0 – 3: 2-58
Meaning of bit 4: 2-59
Meaning of bits 6 - 7: 2-60
JRK - Jerk Time Constant 2-61
JRS - Jerk Smooth Constant 2-62
ODH - Online Debug Help 2-64
MBA – MultiBlock retrace Auxiliary functions 2-66
REM – Automatic return to profile at end of move 2-67
IPB (DTL) - In Position Band 2-68
G70 G71 - Measuring Units 2-69
G90 G91 G79 - Absolute, Incremental and Zero Programming 2-70
G92 G98 G99 - Axis Presetting 2-72
G04 G09 - Dynamic Mode Attributes 2-73
t - Block Execution Time 2-74
DWT (TMR) - Dwell Time 2-74
G93 - V/D Feedrate 2-75
VFF - Velocity Feed Forward 2-76
CODES THAT MODIFY THE AXES REFERENCE SYSTEM **2-77**
SCF - Scale Factors 2-78
MIR - Using Mirror Machining 2-79
ROT (URT) - Interpolation Plane Rotation 2-82
UAO - Using Absolute Origins 2-85
UTO (UOT) - Using Temporary Origins 2-86
UIO - Using Incremental Origins 2-88
RQO - Requalifying Origins 2-90
OVERTRAVELS AND PROTECTED AREAS **2-91**
SOL (DLO) - Software Overtravel Limits 2-92
DPA (DSA) - Define Protected Areas 2-93
PAE (ASC) - Protected Area Enable 2-95
PAD (DSC) - Protected Area Disable 2-95
VIRTUAL AXES MANAGEMENT **2-96**
Virtual Axes 2-96
Virtual modes available on 10 Series CNC 2-96

UPR - Rotation of Cartesian axes 2-97
Using UPR 2-102
UVP - Programming polar coordinates 2-106
Programming examples with polar coordinates 2-108
UVC - Programming cylindrical coordinates 2-110
UVA - Programming non orthogonal axes 2-112
Example of programming with polar coordinates 2-113
TCP - Tool Center Point for machines with "Double Twist" head 2-114
Programming the "m" and "n" parameters (angles) 2-130
Programming the "m", "n" and "0" parameters (vector) 2-131
TCP - Tool Center Point for generic 5-axis machines 2-132
Programming 2-137
TCP - Tool Center Point for machines with fixed tool and rotary table 2-141
Programming 2-147
TCP on multi-processor 2-148

PROGRAMMING TOOLS AND TOOL OFFSETS

T address for programming tools 3-2
T address for multi-tool programming 3-3
h address 3-5
TTR – Toroidal Tool Radius 3-7
AXO - Axis Offset Definition 3-8
RQT (RQU) - Requalifying Tool Offset 3-10
RQP - Requalifying Tool Offset 3-11
TOU (TOF) - Tool Expiry Declaration 3-12
LOA - Table loading 3-13

CUTTER DIAMETER COMPENSATION

G40 G41 G42 - Cutter Diameter Compensation 4-2
Enabling Cutter Diameter Compensation 4-3
Notes on using cutter diameter compensation 4-5
Tool path optimisation (TPO) 4-5
Disabling Cutter Diameter Compensation 4-6
Disabling Compensation with TPO active 4-7
TOOL DIAMETER COMPENSATION CHANGE 4-8
Linear/Linear tool path 4-8
Linear/Circular, Circular/Linear, Circular/Circular tool paths 4-10
r - Radiuses in Compensated Profiles 4-12
b - Bevels in Compensated Profiles 4-13
TPO - Path optimisation on bevels with G41/G42 4-16
Examples of profile optimisation with TPO=1 4-19
Examples of TPO=2 mode 4-22
TPT - Tool Path Threshold 4-25
u v w - Paraxial Compensation 4-27
Examples of compensation factor applications u, v, w 4-28
MSA (UOV) - Defining a Machining Stock Allowance 4-32
AUTOMATIC CONTOUR MILLING 4-33
Limits to use of automatic contour miling 4-33
GTP - Get Point 4-34
Determining the approach point 4-35
CCP - Cutter Compensation Profile 4-37

PROGRAMMING THE SPINDLE

SPINDLE FUNCTIONS 5-1
G96 G97 - CSS and RPM Programming 5-1
SSL - Spindle Speed Limit 5-3
M19 - Oriented Spindle Stop 5-4

MISCELLANEOUS FUNCTIONS

Standard M functions 6-1

PARAMETRIC PROGRAMMING

LOCAL VARIABLES 7-4
E Parameters 7-4
! - User Variables 7-6
SYSTEM VARIABLES 7-8
SN - System Number 7-8
SC - System Character 7-9
TIM - System Timer 7-11
@ - PLUS Variables 7-12
L Variables 7-13
Multiple Assignments 7-14

CANNED CYCLES

CANNED CYCLES G8N 8-1
Canned Cycle Features 8-2
Canned Cycle Moves 8-3
G81 - Drilling Cycle 8-5
G82 - Spot Facing Cycle 8-7
G83 - Deep Drilling Cycle 8-9
DRP – G83 hole reworking distance 8-12
G84 - Tapping Cycle with no Transducer 8-13
G84 - Tapping Cycle with Transducer 8-16
G84 - Rigid tapping cycle with a transducer mounted on the spindle 8-17
TRP (RMS) - Tapping Return Percentage 8-18
G85 - Reaming Cycle (or Tapping by Tapmatic) 8-19
G86 - Boring Cycle 8-20
G89 - Boring Cycle with Spot Facing 8-21
Using two R dimensions in a canned cycle 8-22
Updating Canned Cycle Dimensions 8-23
Updating R dimensions (upper limit and lower limit) during execution 8-24

PARAMACRO

Paramacro Definition 9-1
HC Parameters 9-3
DAN - Define Axis Name 9-6
PMS – S as a paramacro 9-7
PMT – T as a paramacro 9-7
PMM – M as a paramacro 9-7

Syntax **9-7**
S, T, M ANCILLARY FUNCTIONS EXECUTED AS PARAMACROS **9-8**
Syntax: **9-8**

PROBING CYCLES

MANAGING AN ELECTRONIC PROBE **10-1**
PRESETTING A PROBING CYCLE **10-3**
DPP (DPT) - Defining Probing Parameters 10-3
Dynamic Measurement of the Ball Diameter 10-4
Probe Requalification 10-4
Dynamic Measurement of the Probe Length 10-4
Probe Presetting 10-4
PROBING CYCLES **10-6**
G72 - Point Measurement with Compensation 10-7
G73 - Hole Probing Cycle 10-9
G74 - Tool Requalification Cycle 10-11
UPA (RTA) - Update Probe Abscissa 10-13
UPO (RTO) - Update Probe Ordinate 10-13
ERR - Managing Probing Errors 10-13
OPERATIONS WITH A NON-FIXED PROBE **10-14**
Requalifying Origins by Probing Reference Surfaces 10-14
Requalifying Origins by Centring on a Hole 10-16
Checking Diameters 10-16
Checking Plane Dimensions and Hole Depths 10-18
OPERATIONS THAT USE A FIXED PROBE **10-19**

MANAGING THE SCREEN

GRAPHICS VISUALIZATION **11-1**
UGS (UCG) - Use Graphic Scale (Machine plot) 11-2
UGS (UCG) - Use 3D Graphic Scale 11-3
CGS (CLG) - Clear Graphic Screen 11-3
DGS (DCG) - Disable Graphic Scale 11-4
DIS - Displaying a Variable 11-4

MODIFYING THE PROGRAM EXECUTION SEQUENCE

GENERAL **12-1**
COMMAND FOR PROGRAM BLOCKS REPETITION **12-4**
RPT - ERP 12-4
COMMANDS FOR SUBROUTINE EXECUTION **12-8**
CLS – CLD - CLT - Call Subroutine 12-8
PTH - Declaration of the default *pathname* 12-14
EPP - Executing a Portion of a Program 12-15
EPB - Execute Part-Program Block 12-17
BRANCHING AND DELAY COMMANDS **12-19**
GTO - Branch Command 12-19
IF ELSE ENDIF 12-23
DLY - Defining Delay Time 12-24
DSB - Disable Slashed Blocks 12-25
REL - Releasing the part program 12-25
WOS - WAIT on signal 12-26

DEVICE DEFINING COMMANDS 12-27
GDV - Definition of the *device* for file access 12-27
RDV - Release *device* 12-28

HIGH SPEED MACHINING

GENERAL CONSIDERATIONS 13-1
PROGRAMMING POINTS AND CHARACTERISTICS OF THE PROFILE 13-3
Considerations on the use of the G62,G63,G66 and G67 functions (transition codes) 13-6
GENERAL HIGH SPEED MACHINING PROGRAMMING STRUCTURE 13-7
Interaction with Machine Logic 13-7
POINT DEFINING CONVENTIONS 13-8
Points and machining coordinates 13-8
Tool Direction: IJK 13-9
Normal to the Surface Direction: MNO 13-9
Tool Radius Application Direction: PQD 13-10
Paraxial Compensation Factors: UVW 13-11
Tangential Axis 13-12
FEATURES PROVIDED BY HIGH SPEED MACHINING 13-13
Tool Radius and Length Compensation 13-13
Tool Length Compensation 13-14
No Tool Compensation 13-15
Tangential Axis Management 13-15
SETUP 13-16
Type of points described in the part program 13-17
Versor management methods 13-18
Look Ahead management 13-20
Thresholds 13-22
Tool definition 13-24
Tool direction (3D) 13-25
Change in curvature management 13-26
Edge management 13-27
Axis definition 13-28
Axis parameters 13-30
Axis dynamics 13-31
Example 13-32

MULTIPROCESS MANAGEMENT COMMANDS

GENERAL 14-1
SYNCHRONIZATION AMONG PROCESSES 14-2
Notes On The "Wait" Function: 14-2
Notes On The "Send" Function: 14-2
Exchanging data 14-3
Resetting synchronised processes 14-3
Channels table 14-3
DCC - Definition of the communication channel 14-4
PVS - PLUS channel selection 14-5
PRO - Definition of the process 14-6
SND - Send a synchronisation message 14-7
WAI - Wait for a synchronisation message 14-9
EXE - Automatic part program execution 14-11
ECM - Manual block execution in a process 14-12

Example of synchronisation of two process using EXE: 14-13
SHARED AXES 14-14
General 14-14
Conditions for axis acquisition 14-14
GTA - Axes acquisition 14-15
SYSTEM PARAMETERS AFTER GTA 14-16
INFORMATION RETAINED AFTER A GTA COMMAND 14-17
Error Management 14-26

HIGH LEVEL GEOMETRIC PROGRAMMING (GTL)

ORIENTED GEOMETRY 15-2
DEFINING GEOMETRIC ELEMENTS 15-5
DEFINITION OF A REFERENCE ORIGIN 15-8
DEFINITION OF POINTS 15-9
DEFINITION OF STRAIGHT LINES 15-15
DEFINITION OF CIRCLES 15-26
DEFINITION OF A PROFILE 15-40
Profile types 15-40
Connecting the elements 15-45
EXAMPLES OF GTL PROGRAMMING 15-49

WORKING CYCLES FOR TURNING SYSTEMS

PROFILE PROGRAMMING 16-1
Restrictions to the definition of a profile to be recalled by the macro-instructions of roughing/finishing. 16-2
SPECIAL CYCLES PROGRAMMING 16-3
MACRO-INSTRUCTIONS OF PARA-AXIAL ROUGHING WITHOUT PRE-FINISHING 16-3
MACRO-INSTRUCTIONS OF PARA-AXIAL ROUGHING WITH PRE-FINISHING 16-7
MACRO-INSTRUCTION OF ROUGHING PARALLEL TO THE PROFILE 16-9
MACRO-INSTRUCTION OF A PROFILE FINISHING 16-11
THREADING CYCLE 16-12
GROOVE CUTTING CYCLE 16-16

CHARACTERS AND COMMANDS

TABLE OF CHARACTERS A-1
G CODES A-5
MATHEMATICAL FUNCTIONS A-6
LOCAL AND SYSTEM VARIABLES A-6
THREE-LETTER CODES A-7
ASCII CODES A-10

ERROR MESSAGES

Description of error messages and remedial actions B-1

ERROR MANAGEMENT

GENERAL C-1
ERR - Enable/disables error management from part program C-2
Probing cycle errors C-3
Shared axes errors C-4

END OF INDEX

PROGRAMMING WITH 10 SERIES SYSTEMS

10 Series part programs are written with a specific language defined by the ISO standard. This chapter describes the language elements and discusses programming techniques and rules.

THE PROGRAM FILES

The 10 Series part programs are stored in files which may be identified with 10 SERIES names or with DOS names.

- 10 SERIES names are a maximum of 48 characters in length; they identify the programs stored in the logic directories configured on the machine.
  Logic directories are configured during the installation stage (PPDIR config - human interface menu in AMP characterization).
- DOS names are a maximum of 8 characters in length, plus an extension and path where applicable; they identify files resident in DOS type directories.

Mixed management of part programs is not allowed; in fact if a program is activated after being called by a DOS type name, all it subroutines must be identified with DOS names.
Similarly, programs with 10 SERIES names can use only subroutines identified in the same way.

NOTE:
Part programs can also be resident on remote devices, defined in advance through the triliteral GDV (see chap. 12).

Program Components

- ***Address***

  An address is a letter that identifies the type of instruction. For example, these are addresses:

  G, X, Y, F

- ***Word***

  A word is an address followed by a numerical value. For example, these are words:

  G1 X50.5 Z-3.15 F200 T1.1

  When you assign a numeric value to a word, no zeroes must preceed or follow the value. Insert decimal values after the decimal point.

- ***Block***

  A program block comprises a set of words that identify an operation or a series of operations to be performed. The maximum length of a block is 126 characters.
  A technological program is a sequence of blocks that describe a machining operation.

  Each block must end with: <CR> <LF>.

Blocks

Blocks may include one or several fields.
When several fields are used in the same block, they must appear in the order shown in the following table:

block delete	label	sequence number	synchronisation asynchronisation	words codes
/	LABEL	NUMBER	# or &	ALL ALLOWED CHARACTERS

- ***Comment blocks***

  It can be inserted in any position within the current block. Any character after ";" is considered as a comment.

- ***Block delete***

  The block delete field is optional. It allows the operator to choose whether to execute program blocks that begin with the "/" character that are called slashed blocks.

  Example:
  /N100 G00 X100

  The block shown in the example can be enabled or disabled using the PROGRAM SET UP softkey, or typing the three-letter code DSB on the keyboard.

- ***Label***

  The label field is optional. It allows the programmer to assign a symbolic name to a block. A label can have up to six alphanumeric characters which must be between quotes. In case of a slashed block, the label must be inserted after the slash.

  Example:
  "START"
  /"END"

  When a label field is used in a 'GTO' command, the label defines the block that the control should jump to.

- ***Sequence number***

  The "sequence number" field is optional. It allows the programmer to number each program block. A sequence number begins with the letter N and is followed by up to six digits (N0-N999999).

  The sequence number must appear in front of the first operand and after the label.

  Example:
  N125 X0
  "START" N125 X0
  "END" N125 X0

- ***Synchronisation/asynchronisation***

  Characters & and # are used to override the default synchronisation/asynchronisation status. For further information on synchronisation, see "Synchronisation and Program Execution".

  Example:
  #(GTO,START, @PL1=1)

Block Types

Four types of blocks can be used in a part program:

- Comment blocks
- Motion blocks
- Assignment blocks
- Three-letter command blocks

- ***Comment blocks***

  A comment block allows the programmer to insert free sentences in the program. These sentences may describe the function to be executed or provide other pieces of information that make the program more understandable and documented.
  A comment block does not produce messages for the operator. The control ignores a comment block during execution of the program.
  The first character of a comment block must be a semicolon (;). The rest of the comment block is a sequence of alphanumeric characters. For example:

  ;THIS IS AN EXAMPLE OF COMMENT BLOCK

  A comment can be inserted not only in a single block, but also in other types of blocks after the character ";".All characters after a ; considered as a comment. For example:

  G1 X100 Y50 ; Motion block
  E1=10 ; Local variable E
  (ROT,45) ; Rotation command

♦ ***Motion blocks***

Motion blocks conform to ISO and ASCII standards for programming blocks. There is no particular order for programming the components of a motion block.

Example:
G1 X500 Y20 F200

♦ ***Assignment blocks***

Assignment blocks are used to write variables' values directly from the program. Several types of assignments are possible as shown in the following table:

TYPE OF ASSIGNMENT	EXAMPLE
Simple assignment	E10=123.567
Multiple assignment	E1=10, 15.5, 123.467 In multiple assignments values are loaded as follows: 10 to E1 15.5 to E2 123.467 to E3
Math expression assignment	E20=(E10+125*SQR(E23))
System number	SN=1.5

♦ ***Three-letter command blocks***

Three-letter command blocks define an operation with a three-letter instruction in conformity with the RS-447 standard. For example:

(ROT,45)
(DIS,"message text")

For the sake of compatibility between 10 Series and Series 8600 certain commands may be programmed with either of the following three-letter codes.

UGS	UCG
CGS	CLG
DGS	DCG
RQT	RQU
DPA	DSA
PAE	ASC
PAD	DSC
DPP	DPT
IPB	DTL
ROT	URT
SOL	DLO
UTO	UOT
TOU	TOF

Programmable Functions

♦ ***Axis coordinates***

Axis coordinates can be named with letters ABCUVWXYZPQD (according to the configuration set in AMP) and can be programmed in the following ranges:

-99999.99999	-0.00001	mm/inch
+0.00001	+99999.99999	mm/inch

NOTE:
It is impossible to program coordinates in the ±0.00001 range because 0.00001 is the minimum value accepted by the control.

♦ ***R coordinate***

In a circular interpolation (G02 G03) R represents the radius of the circle.
In a standard canned cycle (G81-G89), the R coordinate defines the initial position value and retract value. This function is programmable in the following ranges:

-99999.99999	-0.00001	mm/inch
+0.00001	+99999.99999	mm/inch

NOTE:
It is impossible to program values in the ±0.00001 range because 0.00001 is the minimum value accepted by the control.
In a threading block (G33), the R coordinate represents the offset from the zero angular position of the spindle for multi-start threads.

♦ ***I J coordinates***

In circular interpolation (G02-G03), I and J specify the coordinates of the center of an arc. I specifies the abscissa (typically X) and J the ordinate of the center (typically Y). I and J always specify the center coordinates regardless of the active interpolation plane.
This function is programmable in the following ranges:

-99999.99999	-0.00001	mm/inch
+0.00001	+99999.99999	mm/inch

NOTE:
It is impossible to program values in the ±0.00001 range because 0.00001 is the minimum value accepted by the control.

When the values of the corresponding axis are expressed in diametrical units (according to the configuration set in AMP), the values of the center coordinates (I and J) are also expressed in diametrical units.

I and J coordinates are also used in the deep hole drilling cycle (G83).
In a threading block (G33), the I address defines the pitch variation for variable pitch threads:

I+ Increasing pitch
I- Decreasing pitch

♦ ***K function***

In the deep hole drilling cycle (G83), K defines the incremental value to be applied to the minimum depth value (J) in order to reduce the initial pitch depth (I).
This function is programmable in the following ranges:

-99999.99999	-0.00001	mm/inch
+0.00001	+99999.99999	mm/inch

NOTE:
It is impossible to program values in the ±0.00001 range because 0.00001 is the minimum value accepted by the control.

In a threading block (G33) or a tapping cycle (G84), K defines the thread pitch. In helical interpolation (G02-G03), K defines the helix pitch.

♦ ***F and t function***

The F function defines the axes feedrate. This function is programmable in the following range:

+0.00001	+99999.99999	mm/inch

In G94, F function defines the feedrate in millimetres per minute (G71) or inches per minute (G70).

A "t" value can be programmed in a block to specify the time in seconds needed to complete the move defined in the block. In this case the block feedrate will be:

$$F = \frac{\text{total distance}}{\text{time}} * 60$$

A "t" value is valid only in the block in which it is programmed.
In G93, the F function defines the inverse of the necessary time in minutes to complete the movement:

$$F = \frac{\text{speed}}{\text{total distance}} = 1/t \text{ (minutes)}$$

The F function is mandatory in the blocks when G93 is active and only affects that block.
In G95, F specifies the axes feedrate in millimetres per revolution (G71) or inches per revolution of the spindle (G70).

♦ ***a Function***

The a function defines the acceleration to use on the part program block and may be programmed in the range:

+0.00001	+99999.99999	mm/sec^2 or inches/sec^2

The a function is considered in mm/sec^2 in presence of G71 and in inches/sec^2 in presence of G70. This function is active only in the block it is programmed in and is in any case limited to the acceleration on the profile as calculated by the system in function of the accelerations configured.

♦ ***M function***

The M address can activate various machine operations. The programmable range goes from 0 to 999. See Chapter 6 for further information about these functions.

♦ ***S function***

The S function specifies the spindle rotation speed. It is programmable in the following range:

+0.001	999999.999	rpm/fpm

In G97, the S function defines spindle rotation speed expressed in revolutions per minute.

In G96, the S function defines the cutting surface speed expressed in metres per minute (G71) or feet per minute (G70). The above cutting speed remains constant on the surface.

Refer to Chapter 5 for further information about S function programming.

♦ ***T function***

The T function defines the tool and tool offset needed for machining. It is programmable in the 0.0 to 999999999999.300 range. The 12 digits on the left of the decimal point represent the tool identifier code and the three digits on the right represent the tool offset number.

Chapter 3 provides a detailed description of T functions.

IMPORTANT

M, S and T functions vary according to their characterisation in AMP.
From SW release 3.1 it is possible for the system to execute these functions inside a continuous move (G27-G28).
When planning an application the manufacturer must:

- configure the desired function as "ALLOWED IN CONTINUOUS" in AMP.
- write a machine logic to handle such a function.

In turn, the programmer must remember that these functions produce different effects depending on how they are programmed:

- in **continuous mode** a function configured as "ALLOWED IN CONTINUOUS" will be executed in the sequence in which it has been programmed. In order not to lock the program the function will be executed in "NO WAIT" mode.
- in **point-to-point mode** a function configured as "ALLOWED IN CONTINUOUS" will be executed in standard mode.

♦ ***h functions***

h functions permit to alter an offset during both continuous and point to point moves.
An h function must be programmed by itself in a block. Its value may range from 0 through 300 and may be either an integer or an E variable.

♦ ***G functions***

G codes program machining preparatory functions for machining. The following section deal with this codes.

G Codes

This section shows how to write preparatory G codes in part program blocks. A preparatory G code is identified by the G address followed by one or two digits (G00-G99). At present, only some of the 100 possible G codes are available.

Paramacro subroutines can be called with a three-digit G code. This class of G codes is described in Chapter 9. Three-digit G codes are classified as follows:

G100 - G299	Reserved
G300 - G599	Non modal paramacro range
G600 - G998	Modal paramacro range
G999	Reset modal paramacro

The G code must be programmed after the sequence number (if defined) and before any other operand in the block. For example:

N100 G01 X0 - operand

It is possible to program several G codes in the same block, provided they are compatible with each other. The table that follows defines compatibility between G codes. Zero indicates that the G codes are compatible and can be programmed in the same block; 1 means that the G codes are not compatible and cannot be programmed in the same block without generating an error.

Compatible G Codes

G	00	01	02 03	33	81 89	80	72 73 74	93 94 95	96 97	41 42	40	27 28	29	04	09	90 91	79	70 71	16 17 18 19	92 99
G00	1	1	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	1
G01	1	1	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	1
G02	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	1
G03	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	1
G04	0	0	0	1	1	0	1	0	0	0	0	1	0	1	1	0	0	0	1	1
G09	0	0	0	0	1	0	1	0	0	0	0	0	0	1	1	0	0	0	1	1
G16	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
G17	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
G18	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
G19	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
G27	0	0	0	0	1	0	1	0	0	0	0	1	1	1	0	0	0	0	1	1
G28	0	0	0	0	1	0	1	0	0	0	0	1	1	1	0	0	0	0	1	1
G29	0	0	0	0	1	0	1	0	0	0	0	1	1	0	0	0	0	0	1	1
G33	1	1	1	1	1	1	1	0	0	1	1	0	0	0	0	0	0	0	1	1
G40	0	0	0	1	1	1	1	0	0	1	1	0	0	0	0	0	1	0	1	1
G41	0	0	0	1	1	1	1	0	0	1	1	0	0	0	0	0	1	0	1	1
G42	0	0	0	1	1	1	1	0	0	1	1	0	0	0	0	0	1	0	1	1
G70	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	1
G71	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	1
G72	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
G73	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
G74	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
G79	0	0	0	0	1	1	1	0	0	1	1	0	0	0	0	1	1	0	1	1
G80	1	1	1	1	1	1	1	0	0	1	1	0	0	0	0	0	1	0	1	1
G81	0	0	1	1	1	1	1	0	0	1	1	1	1	1	1	0	1	0	1	1
G82	0	0	1	1	1	1	1	0	0	1	1	1	1	0	0	0	1	0	1	1
G83	0	0	1	1	1	1	1	0	0	1	1	1	1	0	0	0	1	0	1	1
G84	0	0	1	1	1	1	1	0	0	1	1	1	1	0	0	0	1	0	1	1
G85	0	0	1	1	1	1	1	0	0	1	1	1	1	0	0	0	1	0	1	1
G86	0	0	1	1	1	1	1	0	0	1	1	1	1	0	0	0	1	0	1	1
G89	0	0	1	1	1	1	1	0	0	1	1	1	1	0	0	0	1	0	1	1
G90	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0	1	1
G91	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0	1	1
G92	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
G93	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	1
G94	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	1
G95	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	1
G96	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	1	1
G97	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	1	1
G99	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

NOTE:
0 means compatible G codes
1 means incompatible G codes

The following table gives a summary of the G codes available in the control. This default configuration can be modified through the AMP utility.

G code summary

CODE	GROUP	MODAL	DESCRIPTION	POWER UP MILL	POWER UP GRINDING
G00	a	yes	Rapid axes positioning	yes	yes
G01	a	yes	Linear interpolation	no	no
G02	a	yes	Circular interpolation CW	no	no
G03	a	yes	Circular interpolation CCW	no	no
G33	a	yes	Constant or variable pitch thread	no	no
G16	b	yes	Circular interpolation and cutter diameter compensation on a defined plane	no	no
G17	b	yes	Circular interpolation and cutter diameter compensation on 1st-2nd axes plane	yes	no
G18	b	yes	Circular interpolation and cutter diameter compensation on 3rd-1st axes plane	no	yes
G19	b	yes	Circular interpolation and cutter diameter compensation on 2nd-3rd axes plane	no	no
G27	c	yes	Continuous sequence operation with automatic speed reduction on corners	yes	yes
G28	c	yes	Continuous sequence operation without speed reduction on corners	no	no
G29	c	yes	Point-to-point operation	no	no
G92	d	no	Axis presetting	no	no
G99	d	yes	Delete G92	yes	yes
G40	e	yes	Cutter diameter compensation disable	yes	yes
G41	e	yes	Cutter diameter compensation-tool left	no	no
G42	e	yes	Cutter diameter compensation-tool right	no	no
G20		yes	Closes GTL profile		
G21		yes	Opens GTL profile		
G60		yes	Closes the HSM profile	no	no
G61		yes	Opens the HSM profile	no	no
G62		no	Splits the HSM profile in two with continuity	no	no
G63		no	Splits the HSM profile in tw with link	no	no
G66		no	Splits the HSM profile in two with edge	no	no
G67		no	Splits the HSM profile in two with reduced speed on edge	no	no

CODE	GROUP	MODAL	DESCRIPTION	POWER UP MILL	POWER UP GRINDING
G70	f	yes	Programming in inches	no	no
G71	f	yes	Programming in millimetres	yes	yes
G80	g	yes	Disable canned cycles	yes	yes
G81	g	yes	Drilling cycle	no	no
G82	g	yes	Spot-facing cycle	no	no
G83	g	yes	Deep hole drilling cycle	no	no
G84	g	yes	Tapping cycle	no	no
G85	g	yes	Reaming cycle	no	no
G86	g	yes	Boring cycle	no	no
G89	g	yes	Boring cycle with dwell	no	no
G90	h	yes	Absolute programming	yes	yes
G91	h	yes	Incremental programming	no	no
G79	i	no	Programming referred to axis home switch	no	no
G04	j	no	Dwell at end of block	no	no
G09	j	no	Deceleration at end of block	no	no
G72	k	no	Point probing with probe tip radius compensation	no	no
G73	k	no	Hole probing with probe tip radius compensation	no	no
G74	k	no	Probing for theoretical deviation from a point without probe tip radius compensation	no	no
G93	l	yes	Inverse time (V/D) feedrate programming mode	no	no
G94	l	yes	Feedrate programming in ipm or mmpm	yes	no
G95	l	yes	Feedrate programming in ipr or mmpr	no	yes
G96	m	yes	Constant surface speed (feet per minute or metres per minute)	no	yes
G97	m	yes	Spindle speed programming in rpm	yes	no

SYNCHRONISATION AND PROGRAM EXECUTION

The terms "synchronised" and "asynchronised" apply only to part program blocks that do not imply a movement, that is, assignment or calculation blocks. A motion block is any block containing axes motion together with other actions:

- Axis moves
- M codes
- S codes
- T codes

A synchronisation block is taken into consideration and executed only after the motion block that precedes it in the program is completed, that is after the axis move has been executed.

On there other hand, a non-synchronised block is executed as soon as it is read by the part program interpreter, i.e. when perhaps the previous move is still in progress.

The advantage of asynchronous block execution is that variable assignments and complex calculations can be made between moves. This allows to reduce waiting time between two motion blocks caused by calculations.

Default Synchronisation

At power up, the following commands and codes are automatically synchronised:

- UDA, SCF, RQO, IPB, DLY, WOS, WAI, SND, GTA, REL, UPR, TCP, UVP, UVC
- G16, G17, G18, G19, G72, G73, G74

All the other commands are not synchronised.

This default assignment can be changed. This means that the commands that are synchronised by default at power-up can become asynchronous and that the commands that are not synchronised by default at power-up can become synchronous. The next section explains how to override default synchronisation.

NOTE:
Default synchronisation cannot be modified for GTA, UPR, TCP, UVP, and UVC instructions.

Overriding Default Synchronisation

Under certain circumstances, the part program may request to modify the default synchronisation.

If the command is synchronised by default and the programmer wants it to be executed by the interpreter as soon as it is read (asynchronous operation), an "&" must be programmed in the first position of the block, immediately after the "n" number.

If the command is asynchronous and you wish to activate synchronous operation, the first character in the block must be #.

Both # and & are active only in the block where they are programmed.

To avoid possible damage to the workpiece, note that programming synchronised blocks between contouring blocks clears the motion buffer at each synchronised block. This will result in dwells while the buffer is reloaded and all the calculations are performed.

Part Program Interpreter

When the system reads a part program block it executes various activities, depending on the type of block:

- A motion block will be loaded in the motion buffer queue. If the move is defined by a variable, the stored move values stored are those of the variable. The buffer size is configurable from 2 to 128 blocks through AMP.
- An asynchronous assign or calculation block will be executed.

Three factors cause the part program interpreter to stop reading blocks:

- The motion buffer is full. When the active motion block is completed, the interpreter will read another motion block and load it in the buffer queue.
- A non-motion block that contains a synchronised command or a code that forces synchronisation is read. The interpreter does not start again until the last loaded motion block is completed. At this point the block calling for synchronisation is executed and the interpreter starts reading the following blocks.
- Error conditions

Sequence of execution

1. Diameter axes
2. Scale factors (SCF)
3. Measuring units (G70 G71)
4. Paraxial compensation (u v w)
5. Inch/metric programming (G90 G91)
6. Mirror machining (MIR)
7. Plane rotation (ROT)
8. Origins (UAO UTO UIO G92)

Programming restrictions for long real (double) formats

The following restrictions apply to long real programming:

- Max. 15 numbers in total
- Max. 12 integer digits
- Max. 9 decimal digits

The system will display an error if more than 12 integer digits are programmed.

If more than 9 decimal numbers are programmed, the system does not display any error but cuts off the programmed number at the last allowed digit.

END OF CHAPTER

PROGRAMMING THE AXES

AXIS MOTION CODES

Defining Axis Motion

In this manual axes motion directions are defined in compliance with EIA standard RS-267. By convention, we always assume that the tool moves towards the part, no matter whether the tool moves towards the part or the part moves towards the tool in the actual process.

Basic movements can be defined with the motion G codes listed in the following table:

G CODE	FUNCTION
G00	Rapid axes positioning
G01	Linear interpolation
G02	Circular interpolation clockwise
G03	Circular interpolation counter clockwise
G33	Constant or variable pitch threading

G00 - Rapid Axes Positioning

G00 defines a linear movement at rapid feedrate that is simultaneous and coordinated for all the axes programmed in the block.

Syntax

G00 [*G-codes*] [*axes*] [*offset*] [**F**..] [**a**] [*auxiliary*]

where:

G-codes Other G codes that are compatible with G00 (See "Compatible G codes" table in Chapter 1).

axes Axis name followed by a numerical value. The numerical value can be programmed directly with a decimal value or indirectly with an E parameter. Up to nine axes can be written in a block.

offset Offset factors on the profile. For the X, Y, Z axes these factors are entered with u, v, and w respectively. See "Paraxial compensation" in Chapter 4 for further information.

F Feedrate for coordinated moves. It is given with the F address followed by the feedrate value. This parameter does not affect the move of the axes programmed in the G00 block, but is retained for subsequent feedrate moves. The rapid feedrate forced by G00 is a velocity along the vector of the axes programmed in the block. The maximum rapid feedrate is defined during characterisation with the AMP utility.

a Acceleration to be used on the profile.

auxiliary Programmable M, S, and T auxiliary functions. Up to four M functions, one S (spindle speed) and one T (tool selection) can be programmed in the block.

G01 - Linear Interpolation

G01 defines a linear move at machining feedrate that is simultaneous and coordinated on all the axes programmed in the block.

Syntax

G01 [*G-codes*] [*axes*] [*offset*] [**F**..] [**a**] [*auxiliary*]

where:

G-codes Other G codes that are compatible with G01 (See "Compatible G codes" table in Chapter 1).

axes Axis name followed by a numerical value. The numerical value can be programmed directly with a decimal value or indirectly with an E parameter. Up to nine axes can be written in a block.

offset Offset factors on the profile. These factors are entered for the X, Y, Z axes with the characters u, v, w respectively. See "Paraxial compensation" in Chapter 4 for further information.

F Feedrate used for the move. It is given with the F address followed by the feedrate value. If omitted, the system will use the previously programmed feedrate. If no feedrate has been programmed the control will generate an error.

a Acceleration to be used on the profile.

auxiliary Programmable M, S, T auxiliary functions. Up to four M functions, one S (spindle speed) and one T (tool selection) can be programmed in the block.

Example:
This example shows how to program a G01 code.

Program:

```
N60 (UGS,X,-10,100,Y,-10,50)
N70 G0 X10 Y10
N80 G01 X90 Y40 F200
```

G02 G03 - Circular Interpolation

These codes define the following circular movements:

G02 Circular interpolation clockwise (CW)

G03 Circular interpolation counter clockwise (CCW)

The circular move is performed at machining feedrate and is coordinated and simultaneous with all the axes programmed in the block.

Syntax

G02 [*G-codes*] [*axes*] **I**.. **J**.. [**F**..] [**a**] [*auxiliary*]
or
G02 [*G-codes*] [*axes*] **R**.. [**F**..] [**a**] [*auxiliary*]

G03 [*G-codes*] [*axes*] **I**.. **J**.. [**F**..] [**a**] [*auxiliary*]
or
G03 [*G-codes*] [*axes*] **R**.. [**F**..] [**a**] [*auxiliary*]

where:

G-codes Other G codes that are compatible with G02 and G03 (See "Compatible G codes" table in Chapter 1).

axes Axis name followed by a numerical value programmed directly with a decimal value or indirectly with an E parameter.

If axes are not programmed in the block, the move is a complete circle in the active interpolation plane.

I Abscissa of the circle centre. This is a value in millimetres that can be programmed directly or indirectly with an E parameter. The abscissa is expressed as a diameter unit when the corresponding axis is a diameter axis. No matter what interpolation plane you are using, the symbol for the abscissa is always **I**.

J Ordinate of the circle centre. This is a value in millimetres that can be programmed directly or indirectly with an E parameter. The ordinate is expressed as a diameter unit when the corresponding axis is a diameter axis. No matter what interpolation plane you are using, the symbol for the ordinate is always **J**.
NOTE: The parameter **R** cannot be used for arcs of 360 degrees..

R Circle radius alternative to the I and J coordinates. If the arc of a circle is less than or equal to 180 degrees, the radius must be programmed with positive sign; if the arc of a circle is greater than 180 degrees the radius must be programmed with negative sign.

NOTA: R is not allowed with arc of 360 degrees.

F Feedrate used for the move. It is given with the F address followed by the feedrate value. If omitted, the system will use the programmed value. If no feedrate has been programmed an error will occur.

a Acceleration to be used on the profile.

auxiliary Programmable auxiliary functions M, S, T. Up to four M functions, one S (spindle speed) and one T (tool selection) can be programmed in the block.

Characteristics:

The maximum programmable arc is 360 degrees, i.e. a full circle. Before programming a circular interpolation block, the interpolation plane must be defined with G16, G17, G18, G19. G17 is automatically active after power up.
The coordinates of the start point (determined from the previous block), the end point and the centre of the move must be calculated so that the difference between start and end radius is less than the default value (0.01 mm or 0.00039 inches). If this difference is equal or greater than the default value, the control displays an error message and the circular move is not performed.

Incremental programming (G91) can be used in conjunction with circular interpolation. With G91 the end point and the centre point of the circular move are referenced to the start point programmed in the previous block.
The direction (CW or CCW) of a circular interpolation is defined by looking in the positive direction of the axis that is perpendicular to the active interpolation plane.The following examples show the directions for circular interpolation on the active planes.

Directions of a circular interpolation

Circular interpolation in absolute programming with the I and J coordinates of the centre of the circle.

N14 X10 Y20
N15 G2 X46 Y20 I28 J20 F200
N16 G3 X64 Y38 I46 J38

Circular interpolation in absolute programming with the value R of the radius of the circle.

N14 X10 Y20
N15 G2 X46 Y20 R18 F200
N16 G3 X64 Y38 R18

Circular interpolation in incremental programming with the coordinates J and I.

N14 X10 Y20
N15 G2 G91 X36 I18 J0 F200
N16 G3 X18 Y18 I0 J18

Circular interpolation in incremental programming with the value of the radius R.

N14 X10 Y20
N15 G2 G91 X36 R18 F200
N16 G3 X18 Y18 R18

CET (PRC) - Circular Endpoint Tolerance

In circular interpolations, CET defines the tolerance for the variance between the starting and final radiuses of the circle arc.

Syntax

CET=*value*

where:

value Tolerance expressed in millimetres. The default value is 0.01 mm.

Characteristics:

If the difference between starting and final radius is smaller than the tolerance but not zero, the system normalises the circle data according to the values specified in CET and ARM.

If the difference is equal to or greater than the value assigned to CET, an error occurs and the programmed final points will not be executed. In this case, you must either modify the program or increase the CET tolerance.

The value assigned to CET can be modified as follows:

- In the AMP configuration
- By means of a specific data entry
- By writing a new CET in the part program.

The CET tolerance is always expressed in the characterised measuring unit (G70/G71 apply).

If the variance between programmed start and final radius is higher than the CET value, the circle arc can be executed as follows:

- By making the CET value greater than the actual variance
- By programming the arc with the circle radius rather than with the centre using this format:

  G2/G3, final point and R radius

A RESET re-establishes the default tolerance.

Example:
CET=0.02 defines a 0.02 mm tolerance

FCT - Full Circle Threshold

In a circular interpolation, the FCT instruction defines a threshold for the distance between the first and the last point in an arc. Within this distance the arc is considered a full circle.

Syntax

FCT=*value*

where:

value Threshold expressed in millimetres. The default value is 0.001 mm.

Characteristics:

The FCT command allows to deal with inaccurate program data that would otherwise prevent the system from forcing a complete circle. In other words, if the distance from the first to the last point is less than FCT, the system uses the points as if they were overlapping and forces a full circle.

FCT thresholds can be modified as follows:

- In the AMP configuration
- By means of a specific data entry
- By writing a new CET in the part program.

The FCT threshold is always expressed in the characterised measuring unit (G70/G71 apply).

A RESET re-establishes the default threshold.

Example:
G71
FCT=0.005
In this example, FCT defines a threshold 0.005 millimetres.

ARM - Defining Arc Normalisation Mode

The ARM code defines the method with which the system normalises an arc (programmed with the centre coordinates I and J, and a final point) in order to render it geometrically congruent.

An arc is normalised when the variance between initial and final radius is less than the characterised accuracy tolerance or than the tolerance programmed with the CET command.

Before executing an arc, the system calculates the difference between initial and final radiuses.

- If the difference is zero, the control will execute the programmed arc without normalising it.
- If the difference is greater than the CET value, the control will stop without executing the move, and display a profile error message.
- If the difference is less than the CET value, the control will execute the move normalising the arc with the method specified by ARM.
- If the distance is less than the FCT threshold, the system will force the complete circle. For ISO blocks with radius compensation, the system checks the difference twice: first on the base profile without compensation (normalisation stage) and then on the compensated profile (motion generation stage).

Syntax

ARM=*arc mode*

where:

arc mode Is the numerical value that defines the arc normalisation mode.
Valid values are:

0 displaced centre within the CET tolerance (default mode)
1 displaced starting point displaced the CET tolerance
2 displaced centre independent from the CET tolerance
3 centre beyond the CET tolerance range

The default value is zero.

Characteristics:

The arc normalisation mode can be modified as follows:

- In the AMP configuration
- By means of a specific data entry
- By writing a new CET in the part program.

The examples that follow illustrate ARC normalisation modes.

ARM=0

This is an arc through the initial and final programmed points whose centre is displaced within the tolerance defined by CET. The arc is executed with averaged radius.

ARM=1

This is an arc through the programmed final point and the starting point displaced within the CET tolerance. The arc is executed with final radius.

ARM=2

This is an arc whose centre is displaced irrespective of the tolerance defined with CET. In this case the arc is executed with averaged radius.

ARM=3

If the displacement of the centre arc is within the CET tolerance defined with CET, the arc centre will be displaced and the arc will pass through the programmed starting and final points. If the displacement of the centre is not within the CET tolerance, the arc will have the programmed centre and pass through the displaced starting and final points (both points are displaced within the CET/2 tolerance).
In this case the arc is executed with averaged radius.

IMPORTANT

With ARM = 1 or ARM = 3 the resultant profile can show inaccuracies ("steps"):

With ARM = 1 there will be a step at circle start equal to the difference between starting and final radiuses.

In case of ARM = 3 there will be a step both at circle arc start and end.

To prevent these steps from causing a servo error, we suggest that you program a CET value smaller than the characterised servo error threshold.

CRT - Circular interpolation speed reduction threshold
CRK - Circular interpolation speed reduction constant

The variables CRT (Circle Reduction Threshold) and CRK (Circle Reduction K-Constant) are used for reducing the speed on circular elements according to the radius of the circle.

Sintiaxi

CRT = *value*

where:

value Threshold radius below which the reduction has to be applied. The default value, of 0 (zero), makes it possible to modulate the speed limitation due to centrifugal acceleration.

CRK = *value*

where:

value is a constant for modulating the reduction in speed. The value set by default is 1.

Characteristics:

By assigning any value other than 0 to the variable CRT, the speed is reduced on all circular elements with a smaller radius than the value set. The value assigned to the variable CRK enables this reduction to be modulated. The speed is reduced as shown in the graph below, in which it is assumed that the programmed speed Vp is equal to 1 and the variable CRT is equal to 1.

When CRT becomes 0, the CRK value (if other than zero) is used, in circular movements, in order to recalculate processing speed. In circular movements, in fact, processing speed is generally limited by the radius of the circumference (centrifugal acceleration) according to the following relationship:

$$V\ lav = Min\ (\sqrt{a\ radius},\ V\ Prog)$$

where a is the minimum acceleration between the two axes involved in the circular movement.

CRK changes this relationship as follows

$$V\ lav = Min\ (CRK * \sqrt{a\ radius},\ V\ Prog)$$

and therefore makes it possible to increase (CRK > 1.0) or decrease (CRK < 1.0) the limitation associated with the centrifugal acceleration. With a value of 1.0 the standard calculation is retained.

The values assigned to the variables CRT and CRK may be modified as follows

- by means of the AMP command during configuration
- from the part program with the specified syntax.

The values assigned to CRT are always expressed in the current unit of measurement of the process (the G70/G71 functions are applied).

The RESET command restores the characterization values.

Helical Interpolation

G02 and G03 program a helical path in only one block. The system performs the helical path by moving the plane axes in a circular interpolation while the axis that is perpendicular to the interpolation plane moves linearly.

To program a helical path, simply add a depth coordinate and the helix pitch (K) to the parameters specified in the circular interpolation block.

Syntax

G02 [*G-codes*] [*axes]* **I**.. **J**.. **K**.. [**F**..] [*auxiliary*]
or
G02 [*G-codes*] [*axes]* **R**.. **K**.. [**F**..] [*auxiliary*]

G03 [*G-codes*] [*axes*] **I**.. **J**.. **K**.. [**F**..] [*auxiliary*]
or
G03 [*G-codes*] [*axes*] **R**.. **K**.. [**F**..] [*auxiliary*]

---

where:

G-codes — Other G codes that are compatible with G02 and G03 (See "Compatible G codes" table in Chapter 1).

axes — An axis letter followed by a numerical value programmed (either decimal value or E parameter).

If no axes are programmed in the block, the move will generate a full circle on the active interpolation plane.

I — Abscissa of the circle centre. This is a value in millimetres (decimal number or E parameter). The abscissa is expressed as a diameter unit when the corresponding axis is a diameter axis. No matter what the interpolation plane, the symbol for the abscissa is always **I**.

J — Ordinate of the circle centre. This is a value in millimetres (decimal number or E parameter). The ordinate is expressed as a diameter unit when the corresponding axis is a diameter axis. No matter what the interpolation plane, the symbol for the ordinate is always **J**.

R	Circle radius. It is specified with the R address followed by a length value, and is alternative to the I and J coordinates.
K	Helix pitch. This parameter is specified with the K address followed by the pitch value. It can be omitted if the helix depth is less than one pitch.
F	Feedrate. It is specified by the F address followed by a value. If it is omitted, the system will use the previously programmed feedrate. If no feedrate has been programmed, the system will signal an error.
auxiliary	Programmable M, S, and T functions. Up to four M functions, one S (spindle speed) and one T (tool selection) can be programmed in the block.

Characteristics:

If Z is a multiple of K, it is not necessary to program the final point

If the depth is not an integer number of pitches, i.e. if Z is not equal to n * K), the length of the circle arc must be calculated with the decimal remainder of the pitch number. For example, if Z = 2.7 * K, then the arc that must be programmed is 360 * (2.7 - 2) = 252 degrees.

Example:
G2 X . . Y. . Z . . I . . J . . K . . F. .

In this example, addresses X, Y, I, and J refer to circle programming; addresses Z and K refer to helix programming and are respectively the depth and the helix pitch. The figure below shows the typical dimensions of a helical interpolation.

Dimensions Helix

G33 - Constant or Variable Pitch Threading

G33 defines a cylindrical, taper, or face threading movement with constant or variable pitch. The threading move is synchronised to spindle rotation. The parameters programmed in the block identify the type of thread.

Syntax

G33 [*axes*] **K**.. [**I**..] [**R**..]

where:

axes	An axis letter followed by a numerical value.
K	Thread pitch (mandatory). For variable pitch threads, K is the initial pitch.
I	Pitch variation for variable pitch threading. For increasing pitch threading, I must be positive; for decreasing pitch threading I must be negative.
R	Deviation from the zero spindle angular position in degrees. R is used in multistart threading to avoid displacing the starting point.

Characteristics:

All these numerical values can be programmed directly with decimal numbers or indirectly with E parameters.

In decreasing pitch threads, the initial pitch, the pitch variation, and the thread length must be calculated so that the pitch is greater than zero before reaching the final coordinate. Use the following formula:

$$I \leq \frac{K^2}{2\,(Zf - Zi)}$$

where:

I	Is the maximum pitch variation
K	Is the initial pitch
(Zf - Zi)	Is the thread length.

IMPORTANT

During the threading cycle the control ignores the CYCLE STOP button and the FEEDRATE OVERRIDE selector/softkey, whereas the SPINDLE SPEED OVERRIDE selector must be disabled by the machine logic.

VFF may be disabled with the dedicated softkey or with a VFF command.

Constant Pitch Threading

The figures that follow illustrate examples of constant pitch threading. Note that the U axis is a diameter axis.

Cylindrical threading

Part program block: G33 Z-100 K2

Conical threading

Part program block: G33 U40 Z-80 K3

Cylindrical-conical threading

Part program blocks: G33 Z-95 K2 .5
Z-100 U52 K2.5

Variable Pitch Threading

The figures that follow illustrate variable pitch threading. Note that the U axis is a diameter axis.

Cylindrical threading with increasing pitch

Part program block: G33 Z-50 K4 I1

Conical threading with increasing pitch

Part program block: G33 U50 Z-40 K4 I1

Cylindrical thread with decreasing pitch

Part program block: G33 Z-50 K10 I-1

Multi-start threading

An R word in a G33 block makes the control start moving the axes from an angular position that varies according to the programmed R value.

This permits to program the same start point for all threads, rather than move the start point of each thread by a distance equal to the pitch divided by the number of starts.

Example:
Three-start threading

N37 G33 Z3 K6	**1st thread**
.	
.	
.	
N41 G33 Z3 K6 R120	**2nd thread**
.	
.	
.	
N45 G33 Z3 K6 R240	**3rd thread**

Rotary Axes

In the system characterisation, axes can be configured as rotary axes, i.e. a rotary table.

To program rotary axis moves simultaneous to and coordinated with the other axes programmed in the same block:

- Always use decimal degrees (from +0.00001 to +99999.99999 degrees) starting from a pre-selected origin.
- Select either the rapid rate (G00) or the feedrate (G01). In a rotary move rates are always expressed in degrees per minute (dpm, F5.5 format). For example, with F75.5 the axis moves at 75.5 dpm.

To perform milling operations on a circle with a rotary table, calculate the rotary rate with the following formula.

$$F = \frac{360}{pi} * \frac{A}{D} = 114{,}64 * \frac{A}{D}$$

where:

F	Is the rotary rate in dpm
A	Is the linear rate on the arc in millimetres or inches per minute
D	Is the diameter on which the milling operation is performed (in mm or inches).

To move rotary and linear axes simultaneously in the same block, you may calculate the feedrate with one of the following formulas.

With G94:

$$F = A * \frac{\sqrt{X^2+Y^2+Z^2+B^2+C^2}}{L}$$

where:

F	Is the feedrate
A	Is the feedrate on the part (in mm/min or inches/min)
X Y Z B C	Is the actual travel performed by each axis (in mm or inches for linear axes, in degrees for rotary axes)
L	Is the resultant path length (in mm or inches).

With G93:

$$F = \frac{A}{\sqrt{X^2 + Y^2 + B^2}}$$

where:

F	Is the feedrate
A	Is the desired feedrate (in mm/min or inches/min) on the part
X	Is the X axis incremental distance
Y	Is the Y axis incremental distance
B	Is the B axis incremental distance

The control cannot calculate the desired tool feedrate directly because the radius is not programmed. In these cases, the feedrate can be specified as inverse time with G93.

A block moving only the rotary axes generates an arc. If rotary and linear moves are combined, the resulting path may be an Archimedean spiral, a cylindrical helix or more complex curves, depending on the programmed number of linear axes.

Axes with Rollover

Axes with rollover axis are rotary or linear axes whose position is controlled between zero and a positive value configured in the *rollover pitch* parameter.

In the following description the axsi with rollover is rotary and has a 360 degree rollover pitch. We assume that the axis position is controlled in the 0 to 359.9999 degree range. That is, when the axis reaches 360 degrees, the displayed position rolls over to zero degrees.

An axis with rollover can be programmed in a block or in a MDI in two different modes:

absolute mode (G90)	programs the move in degrees.
incremental mode (G91)	programs the move as increments in degrees from the current axis position.

G90 - Absolute mode

In this mode:

- Displayed position is from 0 to +359.99999 degrees
- Programmed range is from 0 to ± 359.99999 degrees
- Direction of axis rotation depends on the sign of the programmed move. By convention, a positive move is clockwise and a negative move is counter clockwise.

For example, let's assume that rotary axis B is positioned at 90 degrees and the following block is written in the part program or in an MDI:

G90 B45

Clockwise Rotation

The B axis rotates by 315 degrees clockwise from the 90 degree position to reach the absolute position of 45 degrees (the sign of the move is positive).

Now let's assume that the B rotary axis is at 90 degrees and, the following block is written in the part program or in an MDI:

G90 B-0

Counter clockwise Rotation

The B axis rotates by 90 degrees counterclockwise to absolute position 0 degrees because the sign of the move is negative.

G91 - Incremental mode

When an axis with rollover is programmed in incremental (G91) mode, the following conditions apply:

- Displayed position is from 0 to +359.99999 degrees.
- Program range is from +/-0.00001 to +/-99999.99999 degrees.
- Direction of axis rotation depends upon the sign of the programmed move. By convention, a positive move is clockwise and a negative move is counter clockwise.

IMPORTANT

The displayed position is beyond the programmed range when the programmed range is greater than +359.999 degrees.

For example, if the absolute zero position of the B rotary axis is 0 degrees and the following block is written in the part program or in an MDI:

G91 B765

Incremental clockwise rotation

The B axis makes two complete clockwise revolutions plus 45 degrees (360 + 360 + 45 = 765).

Pseudo Axes

A pseudo axis is an auxiliary function that may be addressed as an axis and is handled by the machine logic. The pseudo axis name can be any allowed axis name (X,Y,Z,A,B,C,U,V,W,P,Q,D). In a part program block it is possible to program up to 3 pseudo axes but in the AMP it is possible to configure up to 6 pseudo axes.

Diameter Axes

A reaming/facing head can be mounted on the spindle and controlled simultaneously with other axes. By programming such an axis (typically a U axis) as a diameter, the following can be obtained:

- Boring operations on cylindrical or conical holes
- Circular radiuses (concave or convex)
- Chamfers
- Grooves
- Facing operations
- Threads

Programming a U (diameter) axis is similar to programming other linear axes; however, its coordinates must be expressed in diameters. The measuring units can be inches or millimetres according to the current mode (G70/G71).

When the U axis is programmed in the same block as an X, Y or Z move, it is simultaneous with and coordinated to the other axes. U axis moves can be performed at rapid rate (G00) or feedrate (G01) with F in ipm or mmpm.

Before executing a profile with the U axis, the interpolation plane must be defined with the following command:

G16 Z U

IMPORTANT

The order of Z and U in this command is critical, i.e. G16 UZ and G16 ZU define two different interpolation planes.

Cutter diameter compensation (G41 or G42) and a machining allowance (MSA) can be applied to profiles programmed with U.

Example:
This is an example reaming/facing head used in a finishing operation.

```
N116 (DIS, "FINISHING WITH R/F HEAD")
N117 F60 S630 T9 .9 M6
N118 G16 Z U                ;Defines interpolation plane
N119 (UAO, 2)               ;Calls absolute origin for the head
N120 (UTO, 1, Z-200)        ;Temporary origin for Z (skimming the part)
N121 X Y160 M3              ;Position to hole 1
N122 G41 Z2 U51
N123 G1 Z-1 U44 .98         ;Executes the chamfer
N124 Z-44                   ;Executes hole diameter 45
N125 G G40 U40
N126 Z2 F40 S380
N127 G41 Y U106             ;Positions to hole 2
N128 G1 Z-1 U99 .975        ;Executes the chamfer
N129 Z-15                   ;Executes hole diameter 100
N130 r5                     ;Executes radius R = 5
N131 U60                    ;Executes counter boring
N132 r-3                    ;Executes radius R = 3
N133 Z-40 U40               ;Executes taper
N134 G40 Z-44               ;Continues Z axis travel
N135 G U35
N136 Z100 M5
N137 G16 X Y
N138 (UAO,1)
```

The direction of the arcs programmed with G02/G03 or with the r address and the direction for cutter diameter compensation (G41/G42) can be determined by looking at the profile on the Z-U plane. Since negative diameters are usually not programmed, you must consider only the first two quadrants of the plane.

UDA - Dual Axes

It is possible to treat one or more axes as slaves or subordinate to another defined as the master. In this way only the movements of the Master need be programmed as the movement of the slaves is determined by those of the Master to which they are associated and by whether or not reverse mirror movement has been applied.

Syntax

(UDA,*master1/slave1*[,*master2/slave2*,*master3/slave3*,*master4/slave4*]**)**
(UDA)

where:

master1. . . master4	Are the master axis names (one ASCII character per axis). You can program up to 4 master axes.
slave1. . . slave8	Are the slave axes names (one ASCII character per axis). You can program a maximum of 8 slave axes per master.
no parameters	(UDA) without parameters disables the dual axes mode (UDA)

Characteristics:

Dual axes management does not require any special setting in the system with AMP.
After an (UDA...) command, the positive operating limit is the minimum between the positive limit of the master axis and the current position of the master plus the distance that may be covered by the slave axis. In short:

PositiveLim = min(Master PositiveLim, MasterPosition + Slave PositiveLim - SlavePosition)

In the case of a "mirror", the distance that may be covered by the slave refers to its negative limit, so it will be:

PositiveLim = min(Master PositiveLim, MasterPosition - Slave NegativeLim + SlavePosition)

The considerations made for the positive limit also apply to the negative limit:

NegativeLim = max(Master NegativeLim, MasterPosition + Slave NegativeLim - SlavePosition)

In the case of a mirror, it will be:

NegativeLim = max(Master NegativeLim, MasterPosition - Slave PositiveLim + SlavePosition)

When the (UDA...) command is executed, reference must be made both to the master axis and the slave axes.

The RESET command does not cancel the association between the master and slave axes.

Dual axes may be used on rotated planes or in polar or cylindrical coordinates (UVP, UVC).

Dual axes, whether master or slave, must be defined on real axes and not virtual axes.

NOTE:
The names of masters and slaves must be separated by a / (slash).

IMPORTANT

To mirror the slave axis movement, you must program the - operator before the axis name. This rule does not apply to master axes.

Example:

(UDA,X/-U)	U is slaved and mirrored to X
(UDA, A/B - CD)	B, C, D are slaved to A and C is mirrored to A

SDA - Special Dual Axes

It is possible to treat one or more axes as slaves or subordinate to another defined as the master. In this way only the movements of the Master need be programmed as the movement of the slaves is determined by those of the Master to which they are associated and by whether or not reverse mirror movement has been applied. The movement of the master and slave axes can occur even if the axes are not referenced.

Syntax

(SDA,*master1/slave1*[,*master2/slave2*,*master3/slave3*,*master4/slave4*]**)**
(SDA)

where:

master1. . . master4	Are the master axis names (one ASCII character per axis). You can program up to 4 master axes.
slave1. . . slave8	Are the slave axes names (one ASCII character per axis). You can programa maximum of 8 slave axes per master.
no parameters	(SDA) without parameters disables the special dual axes mode (SDA)

Characteristics:

After an (SDA...) command, the positive operating limit is the minimum between the positive limit of the master axis and the current position of the master plus the distance that may be covered by the slave axis. In short:

PositiveLim = min(Master PositiveLim, MasterPosition + Slave PositiveLim - SlavePosition)

In the case of a "mirror", the distance that may be covered by the slave refers to its negative limit, so it will be:

PositiveLim = min(Master PositiveLim, MasterPosition - Slave NegativeLim + SlavePosition)

The considerations made for the positive limit also apply to the negative limit:

NegativeLim = max(Master NegativeLim, MasterPosition + Slave NegativeLim - SlavePosition)

In the case of a mirror, it will be:

NegativeLim = max(Master NegativeLim, MasterPosition - Slave PositiveLim + SlavePosition)

Upon activating the (SDA,...) command the master and the slaves need not be referenced.
The RESET command does not remove the master/slave association.

This allows one to perform the zero point micro-search cycle with dual movement active. In this case performing the zero point micro-search cycle, the system simultaneously moves the associated slaves. At the end of the search the master axis is referenced where as the slave axes are not. In order to reference the slave axes they should be exchanged, one by one, with the master axis by new SDA programming and the zero point micro-search cycle repeated for each one of them, redefined as the master. The initialisation of the slaves is not however necessary if one intends to program only the master with SDA active.

The use of the SDA function is recommended in cases where a zero point micro-search cycle is to be performed following a shutdown of the system with the work still on the work-bench.

NOTE:
The names of masters and slaves must be separated by a / (slash).

IMPORTANT

To mirror the slave axis movement, you must program the - operator before the axis name. This rule does not apply to master axes.

Example:

(SDA,X/-U)	U is slaved and mirrored to X
(SDA, A/B - CD)	B, C, D are slaved to A and C is mirrored to A

XDA - Master/Slave axes

One or more axes can be used as “slave” axes, i.e. subordinate to another axis, defined as “master”. In this manner, you only need to program the movements of the master, since the movements of the slaves are determined by the master they are associated with and by a following factor, which is defined specifically for each individual slave. Commands of different types can be imparted with this feature, each of them having a specific syntax.

Master/Slave Association

This instruction defines the association between a master axis and its slaves (up to 8 axes). It **does NOT activate the following function**, whose activation is by means of a specific command. This means that after this instruction a movement of the master does not bring about a movement of the slave(s). **After this instruction and until the slave is released from the master, slave movements cannot be programmed.**

Syntax

(XDA, 1, master/slave1[slave2[..]], mode, ratio, space)

---

where:

master — Is the name of the master axis and is denoted by a single ASCII character.

slave1…slave8 — Are the names of the slave axes (each of them denoted by a single ASCII character). You can program up to 8 slaves.

mode — Defines the master axis following mode used by the slave(s). It can be:

- **0** The slave follows the master point by point
- **1** The slave follows the master in terms of speed
- **2** The slave follows the master in terms of position
- **3** The slave follows the master in terms of position, and the synchronisation distance is taken up

ratio — This is the master following ratio specified for the slave(s). It must be viewed as a multiplication factor for the feedrate of the master or the distance covered by it. If the value of this ratio is 1.0, the motion of the master is reproduced exactly by the slave; if it is smaller than 1.0, feedrate/distance are reduced, if it is greater than 1.0 they are increased. This value can be preceded by a sign.

distance — This is the distance to be covered by the slave to synchronise with the motion of the master.

Characteristics:

The master axis can identify either an axis present in the process in which the XDA command is activated or an axis which is not present; in the latter case, a “virtual” axis, having the name specified, will be created; this axis will have the dynamic characteristics inherited from the slave axes (the lowest values of feedrate, accelerations and jerk). This axis may be part of a virtualisation (UPR, UDA,…) and also of a TCP. The operation of master axis Homing cannot be performed.

At least one slave axis must be present in the process in which the XDA command is activated; it can be a SHARED axis, i.e. an axis shared with the machine logic environment. In this connection, the axis may continue to be moved by the machine logic even after the association with the master, however it cannot be moved while it is following the master. It cannot be part of any virtualisation or TCP.

Let us now examine the various following modalities available:

Mode 0

In this mode, the slave axis follows the master proportionately to the value of the ratio (if the ratio = 1, the slave reproduces the movement of the master axis exactly), synchronisation is instantaneous and the variation in the feedrate of the slave is “*in steps*”. Slave position and feedrate values are calculated, instant by instant, according to the following formulas:

$$\mathbf{Vslave = Vmaster * FollowRate}$$

$$\mathbf{PosSlave = PosSlave_{t0} + (PosMaster - PosMaster_{t0}) * FollowRate}$$

If the speed specified for the slave axis as a result of the following command exceeds the maximum admissible value for this axis, the system will reduce the feedrate requested accordingly and will give out an emergency (servo error) message, in that the slave is unable to follow the required position.

Mode 1

In this mode, the slave follows the feedrate of the master proportionately to the value of the ratio (if the ratio = 1, the slave copies the movement of the master axis exactly); the synchronisation depends on the dynamic characteristics of the slave axis and/or the distance parameter which defines the synchronisation distance.

If the distance value = 0, the slave will synchronise with the master based on its maximum acceleration and using linear ramps **only**.

If the value is not 0, the slave will synchronise with the master based on an acceleration calculated as a function of the synchronisation distance and using linear ramps **only**. The acceleration will be calculated again with each sampling process based on the following formula

$$\textbf{Aslave} = ((\textbf{Vmaster} * \textbf{FollowRate})^2 + \textbf{Vslave}^2) / 2 * \textbf{Distance}$$

where the value of the distance is gradually reduced based on the distance covered during the synchronisation stage. No check is made on the ensuing acceleration value, and therefore servo errors may arise if the acceleration exceeds the maximum value that can be withstood by the axis.

Once the synchronisation with the master has taken place, the slave will move according to this formula:

$$\mathbf{Vslave = Vmaster * FollowRate}$$

The feedrate (**Vslave**) determined in this manner is “theoretical”, since it is necessary to determine whether this request is compatible with the dynamic characteristics of the axis (maximum feedrate and maximum acceleration). The moment the feedrate of the master varies, the slave will follow this variation based on its acceleration value. If the feedrate requested of the slave exceeds its maximum admissible feedrate, the system will reduce the feedrate requested accordingly. Hence, the feedrate and acceleration values with which the slave has to be moved, $\mathbf{Vslave}_i$ and $\mathbf{Aslave}_i$, will be determined instant by instant. The position of the slave will therefore be calculated on the basis of these values:

$$\mathbf{PosSlave}_{tn+1} = \mathbf{PosSlave}_{tn} + \mathbf{Vslave}_i + \mathbf{Aslave}_i$$

Mode 2

In this mode, the slave follows the position of the master proportionately to the value of the ratio (if the ratio = 1 the slave reproduces exactly the movement of the master); synchronisation depends on the dynamic characteristics of the slave axis and/or the distance parameter which defines the synchronisation distance.

If the distance value is 0, the slave will synchronise with the master based on its maximum acceleration and using linear ramps **only**.

If the value of distance is not 0, the slave axis will synchronise with the master axis based on an acceleration calculated as a function of the synchronisation distance and using linear ramps **only**. The acceleration value will be calculated again with each sampling step according to this formula

$$\mathbf{Aslave = ((Vmaster * FollowRate)^2 + Vslave^2) / 2 * Distance}$$

where the value of the distance is gradually reduced based on the distance covered during the synchronisation stage. No check is made on the ensuing acceleration value and therefore servo error messages may be generated the moment the acceleration exceeds the maximum value that the axis can withstand.

Once the synchronisation with the master axis has occurred, the slave will move according to the following formulas:

$$\mathbf{PosSlave = PosSlave_{t1} + (PosMaster - PosMaster_{t1}) * FollowRate}$$

$$\mathbf{Vslave = Vmaster * FollowRate}$$

The position, **PosSlave**, and the feedrate, **Vslave**, determined in this manner should be rated as "theoretical" values, since it is necessary to determine whether the values requested are compatible with the dynamic characteristics of the axis (Maximum feedrate and maximum acceleration). The moment the feedrate of the master varies, the slave will follow this variation according to its own acceleration value. If the feedrate requested for the slave exceeds the maximum value admissible for this axis, the system will reduce the feedrate accordingly. To this end, the two values with which to move the slave, $\mathbf{Vslave_i}$ and $\mathbf{Aslave_i}$ will be calculated instant by instant. The actual position of the slave axis will therefore be calculated on the basis of these values:

$$\mathbf{PosSlave_{tn+1} = PosSlave_{tn} + Vslave_i + Aslave_i}$$

The difference between the actual and the theoretical position of the axis is taken up by the slave during its motion (even when the master has stopped moving) by moving, to the extent feasible, at a rate higher than the theoretical value (**Vslave**).

Mode 3

In this mode, the slave follows the position and feedrate of the master proportionately to the value of the ratio (if the ratio = 1, the slave reproduces exactly the movement of the master); synchronisation depends on the dynamic characteristics of the slave.

During the entire movement of the slave (i.e. both during and after the synchronisation stage), the motion of the axis is according to the following formulas (always using linear ramps):

$$\mathbf{PosSlave = PosSlave_{to} + (PosMaster - PosMaster_{t0}) * FollowRate}$$

$$\mathbf{Vslave = Vmaster * FollowRate}$$

The position (**PosSlave**) and the feedrate (**Vslave**) determined in this manner should be rated as “theoretical” values, in that it is necessary to determine whether these requests are compatible with the dynamic characteristics of the axis (max admissible feedrate and max admissible acceleration). The moment the feedrate of the master varies, the slave follows the variation according to its own acceleration value. If the feedrate requested of the slave is higher than its maximum admissible feedrate, the system reduces the feedrate requested accordingly. To this end, the two values with which the axis is to be moved (**Vslave** $_i$ and **Aslave** $_i$) will be calculated instant by instant. The actual position of the slave will therefore be calculated on the basis of these values:

$$\mathbf{PosSlave_{tn+1} = PosSlave_{tn} + Vslave_i + Aslave_i}$$

The difference between the actual and the theoretical position of the axis is taken up by the slave during its motion (even when the master has stopped moving) by moving, to the extent feasible, at a rate higher than the theoretical value (**Vslave**).

Releasing the Slave(s) from the Master

This instruction removes the association between the master and the slave(s). **Following this instruction it will be possible to program any movement of the slave axis.**

Syntax

(XDA)

Defining/Changing the following ratio

This instruction defines/changes the parameter that determines the ratio according to which the master is followed by the slave(s) concerned.

Syntax

(XDA, 2, slave1[slave2[..]], ratio**)**
(xda, 2, slave1[slave2[..]], ratio**)**

---

where:

slave1…slave8 — are the names of the slave axes (each of which is denoted by a single ASCII character). You can program up to 8 slaves.

ratio — This is the master following ratio specified for the slave(s). It must be viewed as a multiplication factor for the feedrate of the master or the distance covered by it. If the value of this ratio is 1.0, the motion of the master is reproduced exactly by the slave; if it is smaller than 1.0, feedrate/distance are reduced, if it is greater than 1.0 they are increased. This value can be preceded by a sign.

Characteristics:

The command can be used both when a slave is already following the master axis (it then brings about the release of the slave from the master and activates a new synchronisation stage using the new following parameter) and when the following function is not active (the command activates the following value to be used in the next movement stage).

If the uppercase syntax is used, the movement is stopped and the continuous command underway, if any, is terminated. If the lowercase syntax is used, instead, a continuous mode command is given out; at any rate, the axes are stopped at zero speed and after that are restarted immediately. If you do not want the movement to stop, this can be accomplished by having the machine logic execute a similar command.

Activating the following function

The following of the Master axis by the slave(s) is immediately activated. The following modality is defined by the “mode” parameter contained in the master/slave association command.

Syntax

(XDA, 3, slave1[slave2[..]]**)**
(xda, 3, slave1[slave2[..]]**)**

---

where:

slave1…slave8 are the names of the slave axes (each of which is denoted by a single ASCII character). You can program up to 8 slaves.

Characteristics:

If the uppercase syntax is used, the movement is stopped and the continuous command underway, if any, is terminated. If the lowercase syntax is used, instead, a continuous mode command is given out; at any rate, the axes are stopped at zero speed and are restarted immediately. If you do not want the movement to stop, this can be accomplished by having the machine logic execute a similar command.

Deactivating the following function

The following of the Master axis by the slave(s) is immediately deactivated. The release modality is defined by the “mode” parameter contained in the master/slave association command.

Syntax

(XDA, 4, slave1[slave2[..]]**)**
(xda, 4, slave1[slave2[..]]**)**

where:

slave1…slave8 are the names of the slave axes (each of which is denoted by a single ASCII character). You can program up to 8 slaves.

Characteristics:

The slave axis remains associated with the master, it just does not follow it any longer. Depending on the “*mode*” parameter defined in the master/slave association command, either of the following will occur:

0 The slave changes abruptly from the current feedrate to zero.

others The slave comes to a halt according to its deceleration ramp.

If uppercase syntax is used, the movement is stopped and the continuous command underway, if any, is terminated. If lowercase syntax is used, instead, a continuous mode command is given out; at any rate, the axes are stopped at zero speed and are then restarted immediately. If you do not want the movement to stop, this can be accomplished by having the machine logic execute a similar command.

Example:

N10 (XDA,1,X/ZA,3,0.8,0.0)	Activates master X and slaves Z and A
N20 (XDA,3,ZA)	Activates following function by A and Z
N30 G1X100F2000	
N40 X300	
N50 (xda,4,Z)	Deactivates following function by Z in continuous mode
N60 X400	
N70 X500	
N80 (xda,4,A)	Deactivates following function by A in continuous mode
N90 X660	
N100 X700	
N110 (xda,3,ZA)	Reactivates following function by A and Z in continuous mode
N120 GX0	
N130 (XDA)	Removes association of slaves Z and A with master X
N140 GX	

ORIGINS AND COORDINATE CONTROL CODES

The functions in this class perform the following operations:

G CODE	FUNCTION
G04	Dwell at end of step
G09	Deceleration at end of step
G16	Define interpolation plane
G17	Circular interpolation and cutter diameter compensation on XY plane
G18	Circular interpolation and cutter diameter compensation on ZX plane
G19	Circular interpolation and cutter diameter compensation on YZ plane
G27	Continuous movement with automatic velocity reduction on bevel
G28	Continuous movement without automatic velocity reduction on bevel
G29	Point-to-point movements
G70	Programming in inches
G71	Programming in millimetres
G79	Programming referred to axes home switch
G90	Absolute programming
G91	Incremental programming
G92	Axis presetting
G93	Inverse time (V/D) feedrate programming mode
G94	Feedrate programming in ipm or mmpm
G95	Feedrate programming in ipr or mmpr

NOTE:
The planes specified in G17, G18, G19 are valid if they have been configured in the following sequence: X, Y and Z.

G17 G18 G19 - Selecting the Interpolation Plane

These G codes are used for defining the interpolation plane as described below:

G17 Active interpolation plane formed by axes 1 and 2 (XY).

G18 Active interpolation plane formed by axes 3 and 1 (ZX).

G19 Active interpolation plane formed by axes 2 and 3 (YZ).

Axes 1 (X), 2 (Y), and 3 (Z) are the first three axes declared in the AMP environment.

Syntax

G17
G18
G19

The syntax for each function is simply the G code by itself in one block without parameters or other pieces of information.

G16 - Defining the Interpolation Plane

Like G17, G18, and G19, G16 defines the abscissa and the ordinate of the interpolation plane but is not linked to the first and second configured axes.

Syntax

G16 *axis1 axis2*

where:

axis1 Is the name of the abscissa of the interpolation plane (typically X). It must be one of the configured axes in the system.

axis2 Is the name of the ordinate of the interpolation plane (typically Y). It must be one of the configured axes in the system.

Characteristics:

G16, G17, G18, G19 cannot be used if the following G codes are active:

- Cutter diameter compensation (G41-G42)
- Standard canned cycles (G81-G89)

Example:
G16 X A specifies the interpolation plane formed by axes X and A .

G27 G28 G29 - Defining the Dynamic Mode

The G functions in this class define how the axis moves on the profile and positions at profile end. These codes are always accepted by the control.

G27 Specifies a continuous move with automatic velocity reduction on bevels. At the end of each element velocity is automatically calculated by the control and optimised according to the profile shape. This calculation is based on DLA, MDA and VEF values.

G28 Specifies a continuous move without automatic velocity reduction on bevels. At the end of each element the velocity on the profile is equal to the programmed feedrate.

G29 Specifies a point-to-point move that is independent from the programmed path function (G01-G02-G03). At the end of each element the velocity on the profile is 0.

Syntax

G27 [*G-codes*] [*operands*]

G28 [*G-codes*] [*operands*]

G29 [*G-codes*] [*operands*]

where:

G-codes Other G codes that are compatible with G27, G28 and G29 (See "Compatible G codes" table in Chapter 1).

operands Any operand or code that can be used in a G function block.

Characteristics:

The following diagram shows how G27, G28 and G29 operate when the programmed feedrate is constant throughout the profile.

The following diagram shows how G27, G28 and G29 operate when the programmed feedrate varies through the profile.

In each block the move is divided into three steps:

1) Acceleration

2) Uniform move at programmed feedrate

3) Decelerated motion

G27 and G28 differ only in the step with decelerated motion.
Positioning at the machining rate (G1, G2, G3) is available in continuous mode (G27, G28 and G29) whereas rapid positioning (G0) is always point to point, i.e. with deceleration down to null velocity and accurate positioning regardless of the system status.

With G27-G28 (continuous mode) the control explores and executes the profile as if it were a single block. For this reason, auxiliary functions M, S and T are not allowed within the profile executed in G27-G28.

Continuous mode can be temporarily closed by a G00 move that is still part of the profile. The allowed M, S and T functions may therefore be programmed in a block following G00.

NOTE:
The G code that has been configured in AMP (typically G27) is automatically selected at power-up or after a reset.

Example:
This is a contouring example in continuous and point-to-point mode.

Program 1 (continuous mode):

```
        (UGS,X,-400,100,Y,-400,100)
        N9 (DIS,"MILL DIA. 16")
        N10 T4.4 M6 S800
1       N11 G X-235 Y-230 M13
        N12 Z-10
2       N13 G27 G1 X75 F500          ;Continuous mode starts (G27)
3       N14 Y .
4       N15 G3 X-70.477 Y25.651 I J
5       N16 G1 X-187 Y-295
        N17 G Z5 M5                  ;Temporary shift to point to point mode
        N18 (DIS,"MILL DIA. 28")     ;for spindle stop, tool change and S functions
        N19 T5.5 M6 S1200
        N20 X.. Y.. M13
        N21 Z-..
        N22 G1 X.. Y..               ;Continuous mode restarts
```

IMPORTANT

If G29 were programmed in block N17, continuous mode would stop and subsequent moves in G1-G2-G3 would be performed in point-to-point mode.

Program 2 (point-to-point mode):

```
        (UGS,X,-400,100,Y,-400,100)
        N9 (DIS, "MILL DIA. 16")
        N10 T4.4 M6 S800
1       N11 G29 G X-235 Y-230 M13        ;Point-to-point operation starts
        N12 Z-10
2       N13 G1 X75 F500 M5               ;Spindle stop
3       N14 Y S1200 M13                  ;Spindle CW with coolant
4       N15 G3 X-70.477 Y25.651 I J
        N16 DWT=2
5       N17 G1 G4 X-187 Y-295            ;Dwell at the end of the element
        N18 G Z5 M5
        N19 (DIS,"MILL DIA. 28")
        N20 T5.5 M6 S1200
        N21 G X.. Y.. M13
        N22 Z-..
        N23 G1 X.. Y..
```

IMPORTANT

By programming point-to-point with G29 in block N11, M and S functions have been included in the profile (blocks N13 and N14). The dwell at the end of the element (block N17), however, can also be programmed in continuous mode.

AUTOMATIC DECELERATION ON BEVELS IN G27 MODE

When G27 mode is active, the control automatically calculates the vector velocity on the bevels (i.e. between two subsequent moves) using a two-step algorithm.

During the first step the vector velocity is calculated with a formula based on profile variations. The variation of the profile is associated to the angle formed by two subsequent moves.

The control compares the actual angle with the MDA value; if the angle is greater, the vector velocity is put to zero as in G29 mode; otherwise, the control calculates for this bevel a velocity that is based on the angle, MDA and VEF values.

The second step of the algorithm, called "***look ahead***", is optional. It can be enabled or disabled according to the value of the DLA variable.

The "***look ahead***" step is an optimisation of the first step. In fact, in order to provide a correct stop at the profile end, the calculated vector velocity is re-processed taking into account the total distance to be covered in G27 mode and the acceleration configured for each axis.

IMPORTANT

The ***look ahead*** feature (G1 G27) does not handle feedrate override. In fact, at this stage a 100% feedrate is assumed. Higher feedrates may generate SERVO ERRORS.

DLA - Deceleration Look Ahead

The DLA code enables/disables look ahead calculation in G27 dynamic mode. The control reads the motion blocks that make up the profile and those that follow the block in execution in order to recalculate the exit feedrate for the various blocks. It also calculates the deceleration on the bevels according to the profile. If the profile includes sudden trajectory variations and there are not enough block lengths to ensure appropriate deceleration, it is critical for the system to anticipate these events so that velocities can be adjusted. The number of motion blocks the system can look ahead after the current block can be specified in the characterisation. It ranges from 2 to 64 blocks.

Syntax

DLA=*value*

where:

value can be: **0** disables look ahead
1 enables look ahead

NOTE:
If DLA=1 system block time increases because the control must execute a greater number of calculations for each instruction. This results in greater accuracy.

It is advisable to set DLA=0 when it is clear that the programmed feedrate and the total distance to be covered in continuous mode are such as to provide a good stop at the end of the profile.

With DLA=0 the control will consider only the deviations from the theoretical profile on bevels.

Characteristics:

The default value of this variable is configurable in AMP.

DYM - Dynamic Mode

The DYM defines the type of algorithm to use for calculating the velocity between one element and the next with G27 active.

Syntax

DYM=*value*

where:

value It is a numeric value which can be:

0 to use the standard 10 Series formula
1 to use the standard 8600 Series formula
2 to use the alternative 10 Series formula

The standard Series 10 algorithm is based on precise mathematical formulas which assume a linear response from the machine and that the dynamic parameters configured are always applicable under any condition.

The algorithm already present in the 8600 series uses approximate formulas, therefore applying greater restrictions on movement.

The alternative algorithm keeps into consideration the dynamic components (acceleration) that the axes can bear when passing from one block to the other. In this way it recalculates the final speed at of the first block in order to pass to the next one without excessive stress on the machine.

It is recommended that you verify the behaviour of both algorithms on the machine and then decide which default algorithm best suits that particular machinery and that particular type of work.

Characteristics

The default value of this variable can be set in AMP.

MDA - Maximum Deceleration Angle

The MDA code defines the maximum angular axis departure in which G27 is active. The selected value (from 0 to 180 degrees) defines an angle that is the limit of G27 operation.

Syntax

MDA=*value*

where:

value is a numeric value with the following characteristics:

- angle between 0° and 180° if DYM = 0
- number between 0 and 2 if DYM = 1

In both cases, it represents the maximum deviation between two consecutive elements beyond which, in G27, the stop is forced on the final point.

If DYM = 1, the value is to be calculated as:

sine of the maximum angle for deviations ≤ 90°

1 + [sine (angle - 90°)] for deviations > 90° and ≤ 180°.

- Not significative if DYM = 2

Characteristics:

In order to alter the default value (MDA = 90 degrees) you may assign MDA in the configuration or enter it through a specific data entry or a part program block.

IMPORTANT

The system forces the axis to decelerate to zero velocity when the direction is greater than the angle defined by the MDA value. The system calculates a deceleration ramp for the programmed axis if the direction is less than or equal to the angle defined by the MDA value.

Since the system calculates deceleration on bevels from the actual angle and the MDA and VEF values, it is possible to alter velocity reduction by changing the MDA value. Small values of MDA generate dramatic deceleration on bevels.

The system RESET restores the configured MDA value.

Examples:
DYM=0
MDA=90°
MDA=180°

DYM=1
MDA=1
MDA=2

VEF - Velocity Factor

The VEF code defines a velocity determining factor on bevels in the G27 mode. The velocity calculated from the MDA value can be increased or decreased by changing the VEF value. Small VEF values dramatically reduce velocity on bevels.

Syntax

VEF=*value*

where:

value is a number with the following characteristics:

- number from 0.1 to 8 if DYM = 0 The default value is 0.8.
- number from 0 to 99999 if DYM = 1 The default value is 0.8.
- number from 0 to 99999 if DYM = 2 The default value is 0.8

Characteristics:

The characteristics of the velocity calculation vary according to the value of the DYM variable.

DYM = 0
The following diagram shows different decelerations calculated by the system by varying the VEF value and keeping the MDA value constant.

where:

V	is the velocity on the bevel
α	is the angle between two subsequent movements
Vprog	is the programmed feedrate

DYM = 1

The VEF code defines the maximum form error admissable on the bevel. If the value is 0, at the end of each block the system deccelerates the axes to zero.

DYM = 2

The VEF value defines the maximum speed “step” for the axis in the passage from one block to the next: for example if VEF=0.8 the axis will have a speed “step” of 1+0.8 of the acceleration of set working acceleration.
The system will calculate the speed on the edges according to all the axes that are part of the movement; each axis will have a different speed and the system will choose the minimum among these.

IMPORTANT A system RESET restores the default value (0.8) only if DYM = 0.

Jerk Limitation

The speed diagrams shown in the previous sections show the continuity of the speed function **V(t)**, while the acceleration function **a(t)** has a step pattern. Depending on the characteristics of the machine and the type of machining process, this may cause defects in the finish of the part.

This problem may be solved using an acceleration function **a(t)** with a continuous pattern.

The purpose of the "Jerk Limitation" function is to limit variations in acceleration, so as to control its maximum value, resulting in smoother movement and, consequently, a better surface finish.

MOV - Enable Jerk Limitation

The MOV code is used to define some characteristic of the movements management.

Syntax:

MOV = *value*

where:

value movements behaviour to be enabled.
The value to be specified is obtained from the sum of the decimal weights corresponding to each of the features desire d.

Meaning of bits 0 – 3:

The values allowed for the first 4 bits are as follows:

0 Non-linear ramps disabled and feedrate override active.

1 Movement change optimization during profiling with G27 and G28. This option must be used particularly for curves defined “by point”. It allows a better management of the profile curve variation (especially if defined with very short movements) avoiding axes movements out of the profile.

This option is activated by default when the Jerk Limitation is enabled (bit 1 and bit 2 set).

2 Non-linear ramps enabled and feedrate override managed as variation in the ramp time (i.e. variation in acceleration).

6 Non-linear ramps enabled and feedrate override managed as distortion in shape of the ramps (i.e. almost complete maintenance of acceleration).

8 Jerk Limitation with non-linear ramps enabled and feed rate override managed as a variation in ramp time (in practice, variation in accelerations).

12 Jerk Limitation with non-linear ramps enabled and feed rate override managed as a distortion in the form of the ramp (in practice, almost complete maintenance of accelerations).

Characteristics:

We recommend you use the value **8** for machines that typically run at 100% of the programmed speed. The value **6 (12)** is recommended during the tooling phase, when it is important to have an immediate response to the feedrate override.

Linear ramps with or without Jerk Limitation work both on point to point movements (G29) and continuous movements (G27, G28). This algorithm cannot however be enabled and disabled within a continuous movement.

The default value of this variable is 0. Anyway the MOV variable can be configured in AMP. The system RESET command restores the default value.

IMPORTANT

When non linear ramps are activated with MOV=6 (12), feed rate override is only active between 0 and 100% and values of over 100% cannot be reached.

Meaning of bit 4:

Bit 4 (value 16): specifies the type of speed management mode to be adopted, with UPR 2 or 6 active; if this bit = 0, standard management is maintained; if this bit = 1, the rotating axes will use controlled speed diagrams.

In the former case, the positioning of the rotary axes conforms to the transformation requested by the UPR, but the particular computation algorithms employed may cause appreciable showdowns.

In the latter case, slowdowns are eliminated, but the positioning of rotating axes may fail to conform to the transformation requested by the UPR: in such cases, the axes will lag behind.

Meaning of bits 6 - 7:

The values of bits 6 and 7 have the following meaning:

Bit 6 (value 64) : defines that the feed rate programmed in a movement block refers to linear axes. The programming of a rotary axis, in addition to the linear axes, entails an automatic recalculation of the speed so that it remains the same along the linear axes. The recalculation of the speed is only applied if both the linear axes and the rotary axes are present in a movement. It is not applied in the case of circular movements (G2/G3) in which a rotary axis is part of the interpolation plane.

Bit 7 (value 128): enables the VFF algorithm also for manual movements that are normally executed using the “tracking error” algorithm only.

The value obtained from the setting of these bits is to be added to the one given in bits 0-3.
Example: if you want to use Jerk Limitation with non-linear ramps enabled and feed rates managed as variations in the ramp time and, at the same time, have VFF enabled for normal movements, the MOV variable must be set to the value 128 + 8 = 136.

IMPORTANT

This feature **must NOT** be activated if speed programming is executed in G93 or with t. In this case, the resulting machining time would not be correct..

JRK - Jerk Time Constant

The JRK code defines the acceleration management mode in non-linear ramps. It is used with MOV=2 or 6. With MOV=8 or 12, the JRK value is calculated automatically by the system on the basis of the dynamics set for each axis in AMP; hence the JRK used depends on the axes involved in the single movement.

Syntax:

JRK = *value*

---

where:

value a numeric value greater than 0.5, which is used to define the acceleration management mode in a non-linear ramp

Characteristics:

The default value is 1. The JRK value can be configured in AMP. The RESET command restores the default value.
By setting JRK = 1, the acceleration ramp retains the values configured for axis accelerations. Acceleration decreases with a value lower than 1, and increases with a value higher than 1.

JRK=1 **JRK<1** **JRK>1**

JRS - Jerk Smooth Constant

The JRS code defines the threshold value for limiting the speed with Jerk Limitation activated.

Sintyax:

JRS = *value*

where:

value is a a numeric value greater than 0, which used for defining a speed threshold below which the speed is to be limited if the programmed speed cannot be reached.

Characteristics:

A characteristic of the lookhead algorithm for Jerk Limitation is that it avoids continuous acceleration and deceleration that would cause oscillation in the movement of the axes. This could happen if the if the programmed axes do not allow the speed set **(Vi)** to be reached.

To this aim, the speed diagram is "cut" as illustrated in the figure below:

To prevent the "cut" part from being too large and considerably increasing the machining times, a threshold below which the speed is to be limited is defined.
This threshold is calculated as a function of the JRS parameter on the basis of the following rule:

Vmax The maximum speed value calculated by the algorithm on the movements taken into consideration

Vmin The minimum speed value calculated on the movements taken into consideration.

The system checks whether

$$| Vmax - Vmin | < Vmin \cdot JRS$$

If it is, the Vmax is set equal to Vmin, otherwise it is recalculated so that there is a section with a constant speed in the upper part of the ramp.

IMPORTANT

The default value is 1. The RESET command restores the default value.

ODH - Online Debug Help

This system variable allows to check whether the part program includes continuous moves (G28) in which the velocities programmed on the profile are too high and may generate path errors.

Syntax

ODH=*value*

Characteristics:

To obtain reliable results from the use of ODH, the variable itself must be reset before it is used in the part program: that is, ODH must be forced to zero before the blocks to be tested. This variable is normally used when the Part Program is debugged: the system modifies the ODH variable when it detects critical conditions.

This information is coded in bits with the following meanings:

Bit 0: Indicates that the queue of the elements processed continuously is too short. It occurs when, in G28 (and in some cases in G27), the machining speed is too high with respect to the value set in AMP as the number of blocks to be processed during continuous motion. The possible solutions are to decrease the speed or increase the size of the queue of precalculation elements.

Bit 1: Indicates an excessively high speed and occurs when there are too short elements in the profile, which are skipped due to the excessively high speed. In this case, the machined profile is deformed. The possible solutions are to reduce the speed, or to recalculate the points in the program. To avoid profile deformation, we recommend that bit 0 (MOV = 1) be activated in the MOV variable.

Bit 2: Occurs when an excessively high acceleration is requested in G26 with respect to the one set, in moving on from one movement element to the next

Bit 3: Occurs when the system has forced a brake to avoid deforming the profile when it cannot completely calculate the acceleration/deceleration ramps for the movement elements analysed, during machining with non-linear ramps enabled.

Example:

```
. . . . . . . .
. . . . . . . .
ODH=0                                ;the variable is taken to zero for velocity verification
G1G28F1000                           ;continuous mode starts
. . . . . . . .
. . . . . . . .
. . . . . . . .
. . . . . . . .                      ;continuous mode blocks
. . . . . . . .
. . . . . . . .
. . . . . . . .
G29M5                                ;continuous mode end
(GTO,KO,ODH=1)                       ;branches if ODH=1
(DIS;"VEL.WITHIN SYSTEM LIMITS")
(GTO,CONTINUES)
. . . . . . . .
"KO"
(DIS,"VELOCITY TOO HIGH")
. . . . . . . .
. . . . . . . .
"CONTINUES"
```

MBA – MultiBlock retrace Auxiliary functions

The MBA (Boolean) variable makes it possible to enable/disable the auxiliary functions (M functions) during Retrace operations:

Syntax:

MBA = *value*

Characteristics:

The emission of auxiliary functions is performed only during forward retrace

(see user manual)

value = 0 Emission disabled

value = 1 Emission enabled

IMPORTANT

After a reset, the variable is reset with the value configured in AMP.

REM – Automatic return to profile at end of move

The REM (Boolean) variable determines the Jog return mode.
If the variable is set on 0 the return point is the point at which the hold stage begins (standard execution).
If the variable is set on 1 the return point is the final point of the movement in which the hold stage begins. The return movement is performed automatically, without manual movements (JOG).

Syntax:

REM = *value*

value = 0 Return to input point in hold

value = 1 Return to input point at end of movement

IMPORTANT Reset does not modify the value of the variable.

IPB (DTL) - In Position Band

The IPB command lets you define the axes positioning tolerance.

Syntax

(IPB, *axis1*,[*axis2*,. . .,*axis9*]**)**

where:

axis1 . . axis9 Is an axis address and a numerical value. In the IPB command you can program up to nine axes, each of which must be configured in the system. You should program tolerance dimensions in the measuring unit (G70/G71) that is active when the IPB command is executed.

Characteristics:

This value specifies the tolerance between the theoretical and the actual axis positions at motion end. In G00 positioning mode, the axis must be within this tolerance before the next block is executed.

If you program 0 in the IPB command, the control uses the default positioning tolerance specified in the characterisation.

If no axes are specified, the control uses the positioning tolerance that is currently active for that axis.

You cannot specify the same axis twice in one IPB command.

The IPB command causes an error when the following conditions/modes are active:

- Cutter diameter compensation (G41-G42)
- Continuous mode (G27-G28)
- In internal computations the value entered is rounded out to < 1 digit.
  The error generated is NC 133.

Before programming an IPB command you must disable these codes from program.

IMPORTANT The programmed value overrides the one configured in AMP. A RESET does not restore the configured value.

Example:
(IPB, X 0.1, Y 0.05)
This block specifies a positioning tolerance of 0.1 for X and 0.05 for Y.

G70 G71 - Measuring Units

The G70 and G71 codes define the measuring unit used by the control.

G70 specifies inch programming

G71 specifies millimetre programming

Syntax

G70 [*G-codes*] [*operands*]
G71 [*G-codes*] [*operands*]

where:

G-codes Other G codes that are compatible with G70 and G71 (See "Compatible G codes" table in Chapter 1).

operands Any operand or code allowed in a G function block.

Characteristics:

If neither G70 nor G71 is programmed, the system will assume the default measuring unit stored in the system configuration (typically G71).

When the system switches from G71 to G70 or from G70 to G71 it also converts all position and feedrate information into the relevant unit.

IMPORTANT

However, offset and origin tables are not automatically converted into the alternative unit when the system switches between G71 and G70.

Shifts between G70 and G71 do not affect the values read from probing cycles G72, G73 and G74.

G90 G91 G79 - Absolute, Incremental and Zero Programming

These G codes define whether programming dimensions are absolute, incremental or referred to the machine zero.

G90 Sets absolute programming, i.e. moves referred to the current origin (position to move).

G91 Sets programming in the incremental system, i.e. moves referred to the position reached with the previous move.

G79 Sets programming in absolute reference to home position. It is valid only in the block in which it is programmed (distance to move).

Syntax

G90 [*G-codes*] [*operands*]

G91 [*G-codes*] [*operands*]

G79 [*G-codes*] [*operands*]

where:

G-codes Other G codes that are compatible with G90, G91 and G79 (See "Compatible G codes" table in Chapter 1).

operands Any operand or code that can be used in G function blocks.

Characteristics:

If none of these codes is programmed, the default programming mode is absolute or G90, i.e. coordinates referred to the programmed origin.

G90 and G91 are modal whereas G79 is not. After programming a block with G79, the control restores the programming mode of the previous block.

Using characters >> a mixed incremental/absolute programming in the same block is also possible.

Characters >> positioned before the numeric value of an operand, indicate that it must be considered as an incremental value and that it is valid for that operand only. Characters >> have a meaning only if G90 absolute programming is active. They may be used for all operands on which it is possible to use G91 function.

Example:
G90 G1 X + 80 Y >> + 35 Z-70
The value associated to Y must be considered as incremental.

Example:
This example shows how to use the different reference systems: absolute, incremental and HOME position.

Program:

(UGS,1,X,-50,100,Y,50,100)	
(UAO, 1)	;Enables absolute origin 1
N1 G X Y	;X and Y positioned to absolute origin 1 (assuming default mode G90)
N2 X30 Y40	;X and Y positioned to point 1
N3 G91 X50 Y25	;Incremental mode positioning to point 2 (X+50, Y+25 from point 1)
N4 X-71 Y12	;Incremental mode positioning to point 3 (X-71, Y+12 from point 2)
N5 G90 X110 Y35	;Absolute mode positioning to point 4 (X+110, Y+35 from the origin)
N6 G79 X70 Y55	;Positioning referred to home position on point 5 (X+70, Y+55 from home position)

G92 G98 G99 - Axis Presetting

G92 and G98 represent an alternative method for introducing axis offsets. Since G92/G98 define the reference position, they are used in a part program block by themselves. The difference between the actual and the new position is stored in a separate G92/G98 offset register. In this way, other active offsets such as tool offset and origins, will not be destroyed when a G92/G98 offset is introduced. The G99 code resets the G92/G98 code.

Syntax

G92 *axes*
G98 *axes*

where:

axes Is the number of axes (6 axes max.)

Characteristics:

G98 works the same way as G92, save that G92 does not take into account the MIRROR applied to the programmed axes and G98 does.

Codes G92 and G98 are cancelled by the following functions:

- G99
- M2
- M30
- System reset
- PLUS

Active reset has no influence on a offset programmed with G92 or G98. The G92 or G98 offset shifts the origin for a part program but does not cause any axis motion. When the axis value is entered in a G92 or G98 block it becomes the current axis position.

Example:

```
. . . . . . . .
. . . . . . . .
G0 X100 Y80
G92 X0 Y0
. . . . . . . .
. . . . . . . .
```

G04 G09 - Dynamic Mode Attributes

Two G codes belong to this class:

G04 Dwell at end of block

G09 Deceleration at end of block

Syntax

G04 [*G-codes*] [*operands*]

G09 [*G-codes*] [*operands*]

where:

G-codes Other G codes that are compatible with G4, G9 (See "Compatible G codes" table in Chapter 1).

operands Any operand or codes allowed in a G function block.

Characteristics:

G04 causes a dwell at motion end in a block. The DWT command must be programmed in a block that precedes G04. If no DWT is be programmed, the system assumes the characterised dwell.
G04 is allowed only when the control is in point-to-point (G29) and is valid only in the block in which it is programmed. The value set in DWT may be expressed in seconds (G94 or G95 with G0 active) or revolutions (G95).

G09 forces a feedrate equal to zero at the end of the block in which it is programmed, but does not vary the machining status of the profile in progress. G09 does not cause any change in the control status and is valid only in the block in which it is programmed.

t - Block Execution Time

Both in G93 and G94 it is possible to establish the block execution time by programming the **t** function at the end of the block.

Example:
G1 X10 Y1 t6

Characteristics:

The **t** function is valid only for the block in which it is programmed. The time is expressed in seconds and the control automatically calculates the feedrate at which the moves of the axes programmed in the block will be executed.

DWT (TMR) - Dwell Time

The DWT command lets you assign a dwell time at block end. This dwell time is used by G04 and canned cycle blocks.

Although the DWT command can be programmed anywhere in the part program, but must come before any G04 or fixed cycle linked to it.

Syntax

DWT = *value*

---

where:

value Is a time in seconds or revolutions. It can be programmed directly with a decimal number or indirectly with E parameters.

Example:

DWT	= 12.5	assigns a dwell time of 12.5 seconds
E32	= 13.4	assigns the value 13.4 to the E32 variable
DWT	= E32	assigns a dwell time of 13.4 seconds

G93 - V/D Feedrate

G93 defines the axes feedrate (F) as the reciprocal of the time in minutes required to execute the entity:

$$F = \frac{1}{T}$$

Linear interpolation:

$$F = \frac{Feedrate}{Distance}$$

Circular interpolation:

$$F = \frac{Feedrate}{Arc}$$

where:

Feedrate	linear or circular speed expressed in mmpm (G71) or ipm (G70)
Distance	vectorial distance of linear motion expressed in inches or mm
Arc	arc length expressed in inches or mm.

IMPORTANT The F word is mandatory in blocks where G93 is active.

Example:

```
G93 G1   X. . Y. . F. .
         X. . Y. . F. .
```

VFF - Velocity Feed Forward

This command enables/disables VFF. Il VFF has an effect on the interlocking of the axes allowing the speed to be controlled as well as the position.

Syntax

VFF=*value*

where:

value Can be:

- **0** disables VFF: (only the position of the axes is controlled, that is taking into consideration the error margin between the tracking and its theoretical position).
- **1** enables VFF: (the speed of the axes is also controlled).

NOTE:
The VFF default value, 1, is configured in AMP.

CODES THAT MODIFY THE AXES REFERENCE SYSTEM

The commands in this class let you change the cartesian reference system in which the profile is programmed.

COMMAND	FUNCTION
SCF	Scale factor
MIR	Mirror machining
ROT	Active plane rotation
UAO	Absolute origin
UTO	Temporary origin
UIO	Incremental origin
RQO	Origin requalification

When active, the following functions must be programmed in this sequence:

SCF - G70/G71 - MIR - ROT - ORIGINS.

SCF - Scale Factors

The SCF command assigns a scale factor to programmed axis dimensions. The control applies the scale factors to the axes specified in the SCF command.

Syntax

(SCF [*axis1*, . . . , *axis9*]**)**

or

(SCF [,*value*]**)**

where:

value Programs the scale factor. You can program the scale factor directly with a decimal number or indirectly with an E parameter.

axis1 . . . *axis9* Addresses of axes configured in the system and scale factor.

Characteristics:

You can specify up to nine axes in the SCF command. The control cancels scaling for axes that are not specified in the command. A SCF command programmed without a scale factor or axes cancels scaling for all axes.

Example:

.
.
.

(SCF, 3) Applies a scale factor of 3 to the programmed dimensions for all configured axes.

.
.

(SCF, X2.5, Y3) Applies a scale factor of 2.5 to the programmed dimensions of the X axis and 3 to the Y axis and deactivates scaling for all other axes.

(SCF) Deactivates scaling for all axes.

IMPORTANT The system RESET disables the scale factor for all axes

MIR - Using Mirror Machining

The MIR command reverses (mirrors) the programmed direction of motion for the axes you specify in the command referred to the current origin.

Syntax

(MIR ,[*axis1*, . . . , *axis*9]**)**

where:

axis1. . .*axis9* Is the name of the axis to be moved in mirror.

Characteristics:

You can program up to nine axes in the MIR command. Program an axis only once per each MIR command. If a given axis is not programmed in MIR, any mirror function for that axis will be turned off.

If no axis is programmed in the MIR command, the mirror function will be reset for all configured axes.

The control mirrors the programmed axis move with respect to the current origin, applying the mirror function from the first motion block including that axis after the MIR command.

If you use plane rotation (ROT) in conjunction with the mirror (MIR) command, the control processes them in the following order: MIR then ROT.

IMPORTANT

The system RESET disables the MIR command for all axes. It is equivalent to programming (MIR) without parameters.

Example 1:
The following example shows how mirror machining works.

.	Mirroring not active. Moves occurring the first quadrant. Moves referred to the current origin.
.	
N24 (MIR,X)	Mirroring active for the X axis only. Programmed +X moves generate a move shown in the 2nd quadrant.
.	
.	
.	
N42 (MIR, X, Y)	Mirroring active for the X and Y axes. Result of programmed moves shown in the 3rd quadrant.
.	
.	
.	
N84 (MIR, Y)	Mirroring active for the Y axis and inactive for the X axis. Moves in the 4th quadrant.
.	
.	
.	
N99 (MIR)	Mirroring inactive for all axes. Moves in the 1st quadrant.

Example 2:
This example shows how to use the MIR command. Also note the use of RPT and ERP.

Program:

```
(UGS, X, -100, 100, Y, 0, 80)
N199 (DIS."MILLING CUTTER D =16")
N200 S1500 T8.8 M6
N201 (RPT, 2)
N202 G X90 Y20 M3
N203 Z-100
N204 G1 Z-105 F150
N205 X40 F200
N206 G2 Y60 I40 J40
N207 G1 X90
N208 G Z-100
N209 (MIR, X)
N210 (ERP)
N211 (MIR)
N212 Z
```

ROT (URT) - Interpolation Plane Rotation

ROT is a part program command that rotates the active interpolation plane by a programmed angular value. The centre of rotation is the current origin. ROT can be activated through MDI or as a part program code.

Syntax

(ROT, *angle***)**

where:

angle Represents the value of an angle expressed in decimal degrees. You can program the angle directly with a value numerical value or indirectly with an E parameter. Positive angles are measured CCW from the abscissa of the current interpolation plane. Negative angles are CW. If the angle is zero plane rotation is disabled.

Characteristics:

The control rotates programmed coordinates beginning with the first block after the ROT command. Coordinates referred to the machine zero (G79) are not rotated.

If you use axes rotation (ROT) in conjunction with mirroring (MIR), the control performs them in the following order: MIR first then ROT.

IMPORTANT The system RESET disables the plane rotation. It is equivalent to programming (ROT,0).

Example 1:

Program:

(UTO, 1, X100, Y50)	Activates absolute origin 1 with temporary origin (absolute offset) X100 and Y50
(ROT, 30)	Specifies 30 degree rotation CCW referred to temporary origin
.	
.	Moves in this portion of the program are referred to the temporary origin and rotated 30 degrees CCW
(UAO, 1)	Reactivates absolute origin 1
(ROT, 0)	Deactivates rotation by specifying a 0 degree rotation around origin 1

Example 2:

Program:

```
(UGS, X, 0, 70, Y, 0, 70)
N99 (DIS, "DRILLING D=6")
N100 S2000 F200 T3. 03 M6
N101 (UTO , 1, X30, Y22)
N102 (ROT, 20)
N103 G81 R-90 Z-110 M3
N104 X25 Y25
N105 X40 Y10
N106 X55
N107 X70 Y25
N108 G80 Z
N109 (UAO, 1)
N110 (ROT, 0)
N111 S1000 T4.4 M6
```

Example 3:
This example shows how to use Plane Rotation (ROT) with Repeat (RPT) and Parametric Programming.

Program:

```
(UGS, X, -100, 100, Y, -100, 100)
N148 (DIS, " ...")
N149 S1500 T5.5 M6
N150 E25 =0
N151 (RPT, 8)
N152 ( ROT, E25)
N153 G X40 Y M3
N154 Z0
N155 G29 G1 Z-10 F150
N156 X80 F200
N157 Z-18 F150
N158 X40
N159 G Z0
N160 E25 =E25 + 45
N161 (ERP)
N162 (ROT, 0)
```

UAO - Using Absolute Origins

The UAO commands lets you activate and use one of the absolute origins stored in memory.

Syntax

(UAO, *n* [,*axis1*, . . . , *axis9*]**)**

where:

n Defines the absolute origin number. It may be an integer between 0 and 10 or an E parameter. 0 enables the Home Position.

axis1,...,axis9 Is the address of the axis referred to absolute origin *n*.

Characteristics:

The UAO command allows up to nine axes. Only one absolute origin can be active at a time for a specific axis.

When an axis is not specified in the UAO command, it continues to use its currently active absolute origin. A UAO command programmed without axes (UAO, n) activates origin n for all axes. At power-up, after a control reset, or with n=0 and no axes, all axes are referred to the home position.

If the program requires different origins for different axes, program a separate UAO command for each origin required.

The origin values are automatically converted into and displayed in the unit of the current active mode (G70/G71).

The origins are referred to the HOME position that has been characterised in AMP.

Example:

(UAO,1)	Activates origin 1 for all axes.
.	
.	This portion of the program uses origin 1 for all axes.
.	
(UAO, 2, X, Y)	Activates absolute origin 2 for axes X and Y only.
(UAO, 3, Z)	Activates origin 3 for the Z axis.
.	
.	This portion of the program uses origin 2 for X and Y, origin 3 for Z, and origin 1 for all other axes.
.	
(UAO, 1)	Reactivates origin 1 for all axes.
.	
.	
.	
(UAO, 0)	Reactivates Home position for all axes.

UTO (UOT) - Using Temporary Origins

The UTO command temporarily increments the position of the specified absolute origin by a programmed amount for each declared axis.

Syntax

(UTO, *n* **,***axis1* [**,***axis2*, **. . . ,** *axis9*]**)**

where:

n Defines the number of the absolute origin to be temporarily modified. It is an integer from 0 to 10. If n=0, the programmed offset is added to the home position value.

axis1,...axis9 Is an axis and a dimension. The control treats the dimension as an absolute offset and adds it to the value of the absolute origin for that axis.

Characteristics:

You must declare at least one axis with a dimension in the UTO block but may declare up to nine axes with dimensions. An axis can only be declared once in each UTO command.

The axis dimension in the UTO command must be programmed in the currently active measuring unit (G70/ G71).

If an axis is omitted from in the UTO block, the current absolute origin for that axis stays active.

Once you activate a temporary origin it stays active until you:

- activate a new temporary origin with the UTO command
- activate a new absolute origin with the UAO command
- perform a control reset.

If a scale factor (SCF) is set, the control applies it to the UTO temporary origin.

Example 1:
Dimensions can be E parameters as shown in the block below:
(UTO,1,XE100)

Example 2:

Program:

N10 (UAO,1)	Activates absolute origin 1 for all axes.
.	
.	In this portion of the program all axes use origin 1.
.	
N20 (UTO, 1, X100, Y100)	Applies a temporary origin (absolute offset) to origin 1, X100 and Y100.
.	
.	In this portion of the program axes use origin 1, with X100 as temporary origin.
N30 (UTO, 2, X-250, Y50)	Activates a temporary origin (absolute offset) to origin 2, X-250 and Y50.
.	
.	This portion of the program uses absolute origin 2, with X- 250 and Y50 as temporary origin and origin 1 for the other axes.
.	
N40 (UAO, 2)	Re-establishes absolute origin 2 for all axes.

UIO - Using Incremental Origins

The UIO command causes an incremental shift of the currently active origin for each axis specified in the command.

Syntax

(UIO, *axis1* [**,***axis2***, . . . ,** *axis9*]**)**

where:

axis1,...axis9 Is an axis and a dimension. The control treats the dimension as an incremental offset and adds it to the value of the current origin for that axis.

Characteristics:

You must declare at least one axis in the UIO command, but you may declare up to nine axes. An axis can only be declared once in each UIO command.

The axis dimension in the UIO command must be programmed with the current measuring unit (G70/ G71).

If an axis is omitted from in the UIO block, the current absolute origin for that axis stays active.

Once you activate an incremental origin for an axis it stays active until you:

- activate a new incremental origin for the axis with the UIO command
- activate an absolute origin with the UAO command
- perform a control reset.

If a scale factor (SCF) is set, the control applies it to a UIO incremental origin.

Example:

Program:

N12 (UAO,1)	Activates absolute origin 1 for all axes.
N65 (UIO, X20, Y20)	Applies an incremental offset of X20 and Y20 from origin 1. Absolute origin 0 for other axes remains in effect.
N121 (UIO, Y-40)	Applies a Y-40 increment to the last origin. The X20 incremental origin remains in effect.
N180 (UIO, X-45)	Applies an X-45 increment to the last origin. The Y-40 incremental origin remains in effect.
N230 (UIO, Y35)	Applies a Y35 increment to the last origin. The X-45 increment remains in effect.
N300 (UAO,1)	Restores absolute origin 1 for all axes.

RQO - Requalifying Origins

The RQO command lets you requalify, i.e. update and modify, an absolute origin from program. The RQO command modifies the specified origin by the specified amount for each axis you declare in the block. The origin must be stored in the origins table.

Syntax

(RQO, *n* **,***axis1* [**,***axis2***, . . . ,** *axis9*]**)**

where:

n Defines the absolute origin number (1 to 10). The absolute origin must be stored via softkey driven procedure. n can be programmed directly with a positive integer number, or indirectly with an E parameter.

axis1,...axis9 Is an axis address and a dimension. The dimension is the increment added to the specified absolute origin of the specified axis and can be programmed directly with a decimal number, or indirectly with an E parameter.

Characteristics:

You must specify at least one axis and its dimension in the RQO command. Up to nine axes with their dimensions can be programmed. Program a specific axis only once for each RQO command.

The dimensions specified in the RQO command must be in the default measuring unit. No scale factor (SCF) must be applied to dimensions programmed in an RQO command.

The origin is requalified both in the origins file (so that the requalification becomes permanent) and in memory (if the origin is active when the requalification is applied).

In the table of the origins the requalification values are applied in the unit of measure in which the selected origin is expressed.

In the case of a diametrical axis, the requalification dimension must be programmed in radial terms, since the dimension is an increment to be added to the value already present in the origin, which is a radial value.

Example:
(RQO, 3, X (E31)) Modifies absolute origin no. 3 for axis X of the value contained in E31.

OVERTRAVELS AND PROTECTED AREAS

The commands in this class define the overtravel limits and the protected areas as described below.

COMMAND	FUNCTION
SOL	Defines software overtravels
DPA	Defines protected areas
PAE	Enables a protected area
PAD	Disables a protected area

SOL (DLO) - Software Overtravel Limits

The SOL command defines axis travel limits measured from the current origin.

Syntax

(SOL,*axis-name***,***lower-limit***,***upper-limit***)**

where:

axis-name Is the name of the axis whose travel limits must be defined.

lower-limit Is a dimension for the lower limit.

upper-limit Is a dimension for the upper limit. It must be greater than the lower limit.

Characteristics:

If the programmed software travel ends exceed the limits specified in AMP, the control signals an error.

Software overtravels must be programmed in the measuring unit (G70/G71) that is active when you program the SOL command. Active scale factors (SCF) are applied to the travel limits.

Example:

Program:

(SOL, X, -300, 300)
.
.
.
(SOL, Y, -200, 200)

DPA (DSA) - Define Protected Areas

The DPA command defines a protected area.

Syntax

(DPA, *n*,*axis-name1*,*lower-limit1*,*upper-limit1*,*axis-name2*,*lower-limit2*,*upper-limit2* **)**

where:

n	Is the area identifier number (1-3).
axis-name1	Is the name of the abscissa.
lower-limit1	Is the lower abscissa limit.
upper-limit1	Is the upper abscissa limit.
axis-name2	Is the name of the ordinate.
lower-limit2	Is the lower ordinate limit.
upper-limit2	Is the upper ordinate limit.

Characteristics:

Before beginning a programmed move with linear or circular interpolation, the control checks if the move enters a "protected area".

In a program you can define up to three protected areas for the machine tool. Each area is referred to the origin that is active when the DPA command is specified.

Example:

Program:

```
N1 (DPA,1, X, -300, -100, Y, -100, 100)
N2 ( DPA, 2, X, 200, 450, Y, -50, 50)
N3 (PAE, 1)
N4 (PAE, 2)
N5 T1.1 M6
    .
    .
    .
N80 (PAD, 1)
    .
    .
    .
N99 M30
```

PAE (ASC) - Protected Area Enable

The PAE command enables protected area control.

Syntax

(PAE, *n***)**

where:

n Is the number of a protected area defined with the previous DPA command (1-3 range).

Example:
(PAE,2) enables area 2 control

PAD (DSC) - Protected Area Disable

The PAD command disables protected area control.

Syntax

(PAD, *n***)**

where:

n Is the number of the protected area to be disabled (1-3 range).

Example:
(PAD,3) disables control on area 3.

VIRTUAL AXES MANAGEMENT

Virtual axes management provides a series of features that allow to manage the actual machine too axes in any three-dimensional space.

Virtual Axes

Virtual axes are declared in the configuration but are not associated to any physical parameter (for further information, refer to the AMP Manual).

Virtual axes are used:

- to handle planes rotated in space
- when cartesian planes are transformed into planes measured in polar coordinates
- when cartesian planes are transformed into planes measured in cylindric coordinates
- to perform moves in the tool direction
- to perform tool length compensation during the execution of profiles with five simultaneous axes, such as a tool mounted on a wrist with two degrees of freedom.

IMPORTANT

Prior to performing circular moves (G02-G03) on a programmed virtual plane you must define the interpolation plane with the G16 code.

When virtual axes are disabled the system forces the default plane configured in AMP as interpolation plane.

Virtual axes are disabled by a system reset.

Virtual modes available on 10 Series CNC

COMMAND	FUNCTION
UPR	(USE PLANE ROTATED) Rotation of cartesian axes XY and Z in space.
UVP	(USE VIRTUAL POLAR) Machining transformations from cartesian coordinates to polar coordinates
UVC	(USE VIRTUAL CYLINDRICAL) Machining transformations from cartesian coordinates to cylindrical coordinates.
TCP	(TOOL CENTER POINT) Programming in part ranges rather than in machine ranges.

UPR - Rotation of Cartesian axes

UPR (USE PLANE ROTATED) allows definition of a system of three virtual axes translated and rotated.

Syntax

(UPR,*type, af1af2af3, av1av2av3*[*rot1,rot2,rot3* [,*q1,q2,q3*]]**)**
(upr,*type, af1af2af3, av1av2av3*[*rot1,rot2,rot3* [,*q1,q2,q3*]]**)**
(UPR)
(UPR,99)

where:

type Specifies the rotation mode:

- 0= absolute
  rotation angles are referenced to physical axes
- 1= incremental
  rotation angles are referenced to the system of axes used for incremental programming.
  This type of rotation is possible if a previous rotation has been programmed (absolute the first time, incremental for subsequent times.)
- 2= absolute, with transformation on 5 axes
  The characteristics are similar to those of type 0 but the rotation of the linear axes influences the position of the two rotating axes. Upon activation of the UPR function, the rotating axes take on a new value which identifies the position of the tool axes with respect to the virtual system.
- 3= incremental, with transformation on 5 axes
  The characteristics are similar to those of type 1 and also encompasses the points outlined in the previous item.
- 4 = rotating plane automatically determined according to the direction of the tool axis.
- 6 = as 4 but with 5-axis transformation (see 2).
- 10 = global, type 0
  The characteristics are similar to those of type 0, but the programmed rotation remains active even with subsequent non global UPR's.

- 12 = global, type 2
  The characteristics are similar to those of type 2, but the programmed rotation remains active even with subsequent non global UPR's.

- 14 = global, type 4
  The characteristics are similar to those of type 4, but the programmed rotation remains active even with subsequent non global UPR's.

- 16 = global, type 6
  The characteristics are similar to those of type 6, but the programmed rotation remains active even with subsequent non global UPR's.

- 99
  without any further parameter, disables all active UPR modes, including global rototranslation.

af1af2af3 Names of the three physical axes to be handled in virtual mode (e. g. XYZ).

av1av2av3 Names of the three virtual axes to be moved (e. g. UVW).

rot1,rot2,rot3 Rotation angles expressed in degrees. Direction is selected with the right hand rule, which is discussed in the next section.

q1,q2,q3 These parameters allow the origin of the reference system to determine the origin of the reference system:

- In an absolute rotation, q1, q2 and q3 are the co-ordinates of the system origin. They are relative to the origin that was active when UPR was programmed.
- In an incremental rotation, q1, q2 and q3 are the increments applied to the current origin. The new system origin is the result of adding these increments to the co-ordinates of the current one.
- If these parameters are not specified, the origin of the reference system coincides with that of the current origin.

no parameters (UPR) without parameters disables the UPR mode, leaving the global mode only, if present.
If no global rotation has been programmed, commands (UPR) and (UPR,99) are equivalent.

lower case programming (UPR) If the three-letter block is programmed in lower case, the angles and/or origins can be disabled using the UPR algorithm without exiting from the "CONTINUOUS MOVEMENT" mode.
The lower case three-letter block (upr,..... must only be used to alter the parameters of the (UPR, ...) programmed in advance.
The axis sequence cannot be altered.
A global type rotation cannot be programmed.

To select the direction of rotation of the system of virtual axes you must apply the right hand rule.

Right hand rule

Characteristics:

UPR allows programming of any machine function in a space that has been rotated by the specified angle with respect to the machine tool Cartesian system.

This lets you program the profile in the normal Cartesian space (XYZ) and then have 10 Series recalculate the axes moves according to the virtual planes resulting from rotation.

By programming a **global UPR** you can apply an initial rototranslation to serve as a term of reference for all subsequent UPR's. This makes it possible to perform the same part program, which, in its turn, may contain other geometric transformation instructions, even when the part to be processed is, for instance, fastened to the bench by brackets set along different directions.

The activation of a global type UPR is not rated as a previous UPR for purposes of incremental or lowercase programming.

The activation of a global type UPR cancels any previous virtualisation, resulting in the reset of any displacements.

When a **UPR type 2, 3, 4 or 6** is used, or any of the corresponding global type UPR's are used, the system needs the machine characteristics, which must be entered in the **TCP** table according to the rules established by the TCP itself. This feature is separate form TCP, or rather, it can work whether TCP is active or not. When UPR and (TCP, 5) are active simultaneously, the movement of the axes will influence the movement of the virtual axes.

With **modes 2 and 3**, the tool axis, once identified by the programming of the rotating axes, takes on the same position within the virtual reference system as it has in the reference system identified by the Cartesian system.

Modes 4 and 6 and the corresponding global type modes determine the new rotation plane based on tool direction (i.e. based on the positions of the rotary axes). On the new plane, supposing the programming UPR, 4, XYZ, UVW ..., the W axis will coincide with the direction of the tool, the U axis will lie on the original XY plane and, consequently, the V axis will be based on the rule that determines the Cartesian axes.

By means of angles *rot1, rot2 and rot3*, the *UVW* axes determined in this way may be further rotated; an increase (see type 1 or 3) will be applied to the new axes.

The **RESET** command disables the UPR mode.

IMPORTANT

When a **global UPR** is programmed, the values of *rot1, rot2, rot3* must be between –90° and +90°.

For angles outside this range, when passing from one virtualisation to another, the algorithm generating the rotary axis position could generate unwanted rotations (unwanted, but still congruent with the position of the tool).

IMPORTANT

The **UPR is insensitive to the DAN**: the axes to be specified in the UPR are the names of the physical and virtual axes desired, regardless of any DAN commands executed previously.

Rotation modes:

The three-axis cartesian system rotates sequentially by the programmed angles. This means that:

A) The *af1 af2 af3* system of coordinates rotates by a *rot1* angle around the *af1* axis

B) The new *av1' av2' av3'* system of coordinates, which results from the above described rotation, rotates by a *rot2* angle around the *av2'* axis

C) The *av1" av2" av3"* system of coordinates, which results from the rotations described in A) and B), rotates by a *rot3* angle around the *av3'* axis

After these three operations have been completed, the virtual axes of the resulting system will be *av1 av2 av3.*

IMPORTANT

Since steps A) B) and C) are carried out sequentially, the order in which angles and axes are programmed in the block is critical.

Using UPR

In the following examples it is supposed that the the three-axis cartesian system XYZ is the reference system.

In the examples 1, 2, 3, 4, 5 the reference system origin coincides with the origin in use at the time of programming UPR.

Example 1:

```
.
.
GXYZ
(UPR,0,XYZ,UVW,30,45,60)
G1F5400
U100.4V9.12 W-70
.
U70.345 W-20
.
.
.
.
.
G16UV
G02 U100 V70 R15
G1 U120 W10
.
.
(UPR)
X1 Z4.9
.
```

The UVW system may be obtained by:

A) rotating by 30 degrees the XYZ system around the X axis

B) rotating the U'V'W' system resulting from A) by 45 degrees around V'

C) rotating the U''V''W'' resulting from A) and B) by 60 degrees around W''

Example 2:

```
.
.
.
GXYZ20
(UPR,0,XYZ,UVW,10,0,80)
G1F4000
U50 V70
.
U90
V80 W60
.
.
(UPR)
.
.
.
```

The UVW system may be obtained by:

A) rotating the XYZ system by 10 degrees around the X axis.

B) rotating the U'V'W' system resulting from A) by 80 degrees around the W' axis

Example 3:

```
.
.
.
GXYZ20
(UPR,0,ZYX,ABC,80,0,10)
G1F3000
A50 B70
.
A90
B80 C60
.
.
(UPR)
.
.
```

The ABC system may be obtained by:

A) rotating the XYZ system by 80 degrees around the Z axis

B) rotating the A'B'C' system resulting from A) by 10 degrees around the C' axis

IMPORTANT

The UVW virtual system discussed in Example 2) is different from the ABC system in Example 3).

Example 4:

```
.
.
.
(UPR,0,ZYX,WVU,0,50,60)
U90V30
W10
.
.
.
.
(UPR)
.
.
.
```

The WVU system may be obtained by:

A) rotating the XYZ system by 50 degrees around the Y axis.

B) rotating the W'V'U' system resulting from A) by 60 degrees around the U' axis

Example 5:

```
.
.
GXYZ
(UPR,0,XYZ,UVW,30,0,0)

G16 UV
G1 F1000 U50 V0
V30
.
.
U25 V35
(UPR,1,XYZ,UVW,60,0,0)
U30 V20
.
.
(UPR)
GX10 Y25
.
.
```

The UVW system may be obtained by rotating the XYZ system by 30 degrees around the X axis

The UVW system resulting from the previous rotation rotates by another 60 degrees around U

Example 6:

```
.
GXYZ
(UPR,0,XYZ,UVW,30,45,60,10.8,20,-30.2)
G1 F5400
U100.4V9.12W-70
U70.345 W-20
.
.
.
G16UV
G02 U100 V70 R15
G1 U120 W10
.
.
(UPR)
X1 Z4.9
.
```

The UVW system may be obtained by:

A) rotating the XYZ system by 30 degrees around the X axis.

B) rotating the U'V'W' system resulting from A) by 45 degrees around V

C) rotating the U"V"W" resulting from A) and B) by 60 degrees around W"

The origin of the reference system coincides with the point whose coordinates are X10.8, Y20, Z-30.2

Example 7:

```
.
.
GXYZ
(UPR,0,XYZ,UVW,30,45,60,10.8,20,-30.2)
G1F500
U100.4 V9.12 W-70
U70.345 W-20
.
.
.
.
.
```

The UVW system may be obtained by:

A) rotating the XYZ system by 30 degrees around the X axis.

B) rotating the U'V'W' system resulting from A) by 45 degrees around V'.

C) rotating the U"V"W" resulting from A) and B) by 60 degrees around W"

The origin of the reference system coincides with the point whose coordinates are X10.8, Y20, Z-30.2

```
.
(UPR,1,XYZ,UVW,10,0,0,3,8,5)
U120 V30
.
.
.
(UPR)
GX70.5 Y10 Z25
.
.
```

The UVW system resulting from C) rotates by another 10 degrees around U.

The origin of the reference system coincides with the point whose coordinates are X13.8, Y28, Z-25.2

Example 8:

In this case, we want to apply an initial rotation, say of 90° around the Z axis, to the part program discussed in example 2:

```
.
.
GXYZ
(UPR,10,XYZ,UVW,0,0,90)
(UPR,0,XYZ,UVW,10,0,80)
G1F4000
U50 V70
U90
V80 W60
.
.
(UPR,99)
```

The UVW system is achieved through:

A) a 90° rotation of the XYZ system around the Z axis

B) a 10° rotation of the U'V'W' system originating from point A) around U’

C) an 80° rotation of the U”V”W” system originating from point B) around W”

Disables all active UPR’s

UVP - Programming polar coordinates

UVP permits to program axes moves with virtual coordinates.

Syntax

(UVP,*af1af2*,*av1av2*,r**)**
(UVP)

where:

af1	physical linear axis (for example, X)
af2	physical rotary axis (for example, C)
av1	virtual abscissa axis (for example, u)
av2	virtual ordinate axis (for example, v)
r	minimum radius the tool path must not enter
no parameters	disable UVP by programming (UVP) without parameters

Characteristics:

This virtual mode permits to move an X linear axis and a C rotary axis by programming their coordinates in a UV cartesian plan. The (U,V) coordinates of any point on the virtual plane are translated into the (X,C) coordinates of the physical axis.

Machining with polar coordinates

IMPORTANT

To avoid the rotary axis is requested to exceed the rapid feed, the ***r*** parameter must be programmed bear in mind the F programmed feed.

The minimum radius should be calculated using the following formula:

$$r = \frac{F}{Vmax} * \frac{360}{2\pi}$$

where:

r = minimum radius

F = programmed feed (mm/min or inch/min)

Vmax = rotary axis rapid speed

Whether to the **r** parameter is attributed a negative value, the feed limitation is executed dinamically, in such a way to allow high feeds in working point that are quite far from the working centre.

IMPORTANT

Accelerations are calculated in a dynamic mode, therefore it is preferable to use this (**r-)** programming only when positioned at a certain distance from the working centre as, in this case, the rotary axis might be subject to too high accelerations

Programming examples with polar coordinates

Example 1:

```
E0=110*180/(3.14159*800)
T1.1M6
S1000M3
GC0X50Y0
(UVP,XC,UV,E0)
G16 UV
G1G42U20VF110
V20
r3
U-15
b5
V-20
r5
U0
G40G2U20V0I20J-20
(UVP)
GX50
```

Example 2:

```
T1.1M6
S2000M3F300
GC0X80Y0-Z-5
(UVP,XC,UV,10)
G16UV
p1=U20V0
l1=p1,a90
c1=l0J35r-25
l2=U-15V0,a90
l3=U0V-20,a0
c2=l25J-30r-25
G21G42p1
l1
r3
c1
r4
l2
r5
l3
r3
c2
r3
l1
p1
G20G40
(UVP)
GX80
```

UVC - Programming cylindrical coordinates

The UVC command (USE VIRTUAL CYLINDRICAL) permits to program cylindrical coordinates.

Syntax

(UVC,*af1,av1,*r**)**
(UVC)

where:

af1	physical rotary axis (for example, B)
av1	virtual axis (for example, W)
r	cylindrical radius on which the profile is executed
no parameters	without parameters disables **UVC** mode.

Characteristics:

This mode permits to move a Y linear axis and a B rotary axis by programming their coordinates on a WY plane. While the moves of virtual Y coincide with those of physical Y, each move of W corresponds to a circle arc, which is a function of the cylinder radius and must be translated into an angular move of the B physical axis.

Machining with cylindrical coordinates

Example:

```
(DIS,"EXAMPLE UVC");- Programming with cylindrical coordinates -
T1.1M6
S2000F300M3
B0
XY20Z10
E30=60;CYLINDER RADIUS
(UVC,B,W,E30)
G16WY
p1=W0Y20
E31=2*3.1415*E30
p2=WE31Y20
l1=p1,p2
c1=I80J45r25
c2=I140.71J35r-25
l2=c1,c2
c3=I180J35r-25
l3=c2,c3
c4=c3,l1,r15
G21G41p1
Z-12
l1
c1
l2
c2
l3
c3
c4
l1
G20G40p2
GZ20
(UVC)
M30
```

UVA - Programming non orthogonal axes

The UVA (USE VIRTUAL ANGULAR) command makes it possible to program non orthogonal axes by treating them as if they were orthogonal.

Syntax

(UVA,*af1af2*,*av1av2*,a**)**
(UVA)

where:

af1	is the name of the first physical linear axis (e.g. X)
af2	is the name of the second physical linear axis (e.g. Y)
av1	is the name of the virtual abscissa axis (e.g. U)
av2	is the name of the virtual ordinate axis (e.g. V)
a	is the angle, in degrees, subtended by two positive semi-axes of the physical linear axes
no parameters	**(UVA)** without parameters disables the UVA mode.

Characteristics:

This type of virtualisation is used when you want to move two linear X Y axes that are not mutually orthogonal by programming the coordinates on a virtual orthogonal UV plane. The U axis, the abscissa of the new pair of axes, will be coaxial with axis X, whereas axis V will be perpendicular to U, forming an angle of 90°- with axis Y.
The origin of UV will coincide with the origin of XY. The position of a generic point P with coordinates (U,V) in the virtual plane is translated into the (X,Y) coordinates of the physical axes. While the movements of U coincide with those of the corresponding physical axis (X), every movement of V is translated into an interpolated movement of X and Y.

View of the XY plane and the corresponding UV plane

IMPORTANT

The sign of parameter ***a*** is positive if the X axis, as it reaches its position on the Y axis, rotates around itself in the counterclockwise direction; it is negative if the direction of rotation is clockwise (in the example shown in the figure below, ***a*** is positive).

Values of ***a*** of 0° and 180° **are not permissible**, as this would entail two linear physical axes arranged parallel to one another.

Example of programming with polar coordinates

Example 1:
N1 T1.1M6
N2 S1000M3
N3 GX0Y0
N4 (UVA,XY,UV,45)
N5 G1U20VF110
N6 V30
N7 U0
N8 V0
N9 U-10V10
N10 (UVA)
N11 GX50

TCP - Tool Center Point for machines with "Double Twist" head

This feature permits to refer the programs of a 5-axis machine (with 3 linear + 2 rotary axes) to the tool tip rather than to the center of rotation of the axes (or the head center).

The position controlled by the system depends on position of the rotary axes and on the geometric characteristics of the head.
The algorithm may be activated by simply programming the TCP three letter code in a block;

it is disabled by programming (TCP) or by giving a RESET command.

If the three-letter code is programmed in lower case (tcp,n), some of the parameters can be altered without disabling the TCP algorithm and without exiting from"CONTINUOUS MOVEMENT" mode.
The lower case three-letter code must only be used to alter the parameters of the (TCP,n) programmed previously, and the *n* parameter must be the same as specified earlier; furthermore, (tcp,n) must not be used in mode n=5.

Syntax

(TCP[,n])
(tcp,[n])

---

where:

n Offset mode code (1÷5); the various modes are shown in the pages that follow.

Characteristics:

In the TCP block you must also write information about the current tools. The system may handle a head simultaneously mounting as many as four different tools.

Configuration parameters must be assigned after the head has been positioned at the angles shown in figure 2.1. These parameters are included in the user table. There are three tools you can use for entering data in the user table: the user table editor, the specific machine logic functions, or the L variables in the program. Also you may upload the predefined configuration files available with the SETUP utility of the USER TABLE EDITOR.

NOTE:

- For more information about user table management from PLUS refer to the description of $TBLPUTD and $TBLGETD functions in the PLUS Library Manual.
- To access the user table from the TABLE EDITOR environment refer to the User Manual.

TCP Table for machines with "Double Twist" head

Variable	User Table	Record in PLUS User Table	Part Program	Meaning	TCP
Offset mode code	385	Record 97 - Var 1	L384	Defines the offset calculation mode according to the machine configura-tion. Allowed values are as follows: 0 - Enables dynamic part of TCP only, geometric calculation excluded 1 - No mobile table in the machine 2 - Mobile table on first axis 3 - Mobile table on second axis 4 - Mobile table on first and second axes 5 - Reserved.	1, 2, 3, 5
Tool length ①	386	Record 97 - Var 2	L385	Defines the length of the current mounted tool. The length is obviously referred to the active tool holder.	1, 2, 5
Active tool holder ¹	387	Record 97 - Var 3	L386	Active tool holder number 1÷4.	1, 2, 3, 5
Parameter A ① (figures 2.2 e 2.3)	388	Record 97 - Var 4	L387	Specifies the distance --with sign and expressed in mm-- between the rotation axis of the horizontal head and the parallel plane supporting the rotation axis of the vertical head.	1, 2, 5
Horizontal head offset①	389	Record 98 - Var 1	L388	It is the offset --expressed in degrees-- to be applied to the horizontal head axis so that the plane that supports the tool coincides with the YZ plane of the machine. To calculate this offset value, rotate the horizontal axis until the plane that supports the tool coincides with the YZ plane of the machine, and then direct the vertical head towards Y- with a positive head move. Then pick up the displayed coordinate, multiply it by the rotation coefficient of the horizontal head (Horizontal head rotation coefficient) and change the sign of the result.	1, 2, 3, 4, 5
Horizontal head rotation ① (figure 2.3)	390	Record 98 - Var 2	L389	Defines horizontal head rotation as seen from above. Allowed values for positive programming are as follows: +1 for clockwise rotation - 1 for counter clockwise rotation.	1, 2, 3, 4, 5
Vertical head rotation① (figure 2.2)	391	Record 98 - Var 3	L390	Defines vertical head rotation as seen from the left hand side. Allowed values for positive programming are as follows: +1 for clockwise rotation - 1 for counter clockwise rotation.	1, 2, 3, 4, 5

① Parameter alterable in continuous through lower case programming (tcp,n).

TCP Table (continued)

Variable	User Table	Record in PLUS user table	Part Program	Meaning	TCP
1st linear axis ID	392	Record 98 - Var 4	L391	ID of the axis to be used as abscissa in the offset algorithm.	1, 2, 3, 4, 5
2nd linear axis ID	393	Record 99 - Var 1	L392	ID of the axis to be used as ordinate in the offset algorithm.	1, 2, 3, 4, 5
3rd linear axis ID	394	Record 99 - Var 2	L393	ID of the axis to be used as vertical axis in the offset algorithm.	1, 2, 3, 5
Horizontal head ID	395	Record 99 - Var 3	L394	ID of the axis on which the horizontal head is mounted. If this ID=0, the head inclination angle will be the one specified in the *Horizontal Head Inclination* parameter.	1, 2, 3, 5
Vertical head ID	396	Record 99 - Var 4	L395	ID of the axis on which the vertical head is mounted. If this ID=0, the head inclination angle will be the one specified in the *Vertical Head Inclination* parameter.	1, 2, 3, 5
Horizontal head inclination	397	Record 100 Var 1	L396	Horizontal head inclination angle to be used when the horizontal head axis ID=0. In this case the horizontal head is to the specified position. It is a value with sign that is a function of the configured "horizontal head direction".	1, 2, 3, 5
Vertical head inclination	398	Record 100 Var 2	L397	Vertical head inclination angle to be used when the vertical head axis ID=0. In this case the vertical head is to the specified position. It is a value with sign that is a function of the configured "vertical head direction".	1, 2, 3, 5
Radius ① (figure 2.1)	399	Record 100 Var 3	L398	Defines the radius of the tool currently mounted on the machine, with reference to the active tool holder.	1 2 5
Angle ① (figure 2.1)	400	Record 100 Var 4	L399	Defines the angle formed between the tool center and the tool contact point after the vertical head has been rotated in order for the tool to be in the Y- direction.	1 2 3 5

① Parameter alterable in continuous through lower case programming (tcp,n).

Tool holder 1

Variable	User Table	Record in PLUS user table	Part Program	Meaning	TCP
Parameter B ① (figures 2.1, 2.3)	369	Record 93 - Var 1	L368	Distance with sign expressed in mm from the tool plane --in the Y- direction-- to the parallel plane that contains the horizontal head rotation axis.	1 2 5
Parameter C ① (figures 2.1, 2.2)	370	Record 93 - Var 2	L369	Distance with sign expressed in mm from the tool plane and the parallel plane that contains the vertical head rotation axis.	1 2 5
Parameter D ① (figures 2.2, 2.3)	371	Record 93 - Var 3	L370	Distance with sign expressed in mm from the vertical head rotation axis to the tool holder grip.	1 2 5
Offset vertical head ①	372	Record 93 - Var 4	L371	It is the offset, expressed in degrees, to be applied to the vertical head axis in order to position the tool axis parallel to the machine -Y direction. To calculate this value, position the tool in the Y- direction, read the displayed position, multiply it by the configured vertical head direction and change the sign of the result.	1 2 3 5

① Parameter alterable in continuous through lower case programming (tcp,n).

Tool holder 2

Variable	User Table	Record in PLUS user table	Part Program	Meaning	TCP
Parameter B ① (figures 2.1, 2.3)	373	Record 94 - Var 1	L372	As in tool holder 1	1 2 5
Parameter C ① (figures 2.1, 2.2)	374	Record 94 - Var 2	L373	As in tool holder 1	1 2 5
Parameter D ① (figures 2.2, 2.3)	375	Record 94 - Var 3	L374	As in tool holder 1	1 2 5
Vertical head offset ①	376	Record 94 - Var 4	L375	As in tool holder 1	1 2 3 5

Tool holder 3

Variable	User Table	Record in PLUS user table	Part Program.	Meaning	TCP
Parameter B ① (figures 2.1, 2.3)	377	Record 95 - Var 1	L376	As in tool holder 1	1 2 5
Parameter C ① (figures 2.1, 2.2)	378	Record 95 - Var 2	L377	As in tool holder 1	1 2 5
Parameter D ① (figures 2.2, 2.3)	379	Record 95 - Var 3	L378	As in tool holder 1	1 2 5
Offset vertical head ①	380	Record 95 - Var 4	L379	As in tool holder 1	1 2 3 5

Tool holder 4

Variable	User Table	Record in PLUS user table	Part Program	Meaning	TCP
Parameter B ① (figures 2.1, 2.3)	381	Record 96 - Var 1	L380	As in tool holder 1	1 2 5
Parameter C ① (figures 2.1, 2.2)	382	Record 96 - Var 2	L381	As in tool holder 1	1 2 5
Parameter D ① (figures 2.2, 2.3)	383	Record 96 - Var 3	L382	As in tool holder 1	1 2 5
Offset vertical head ①	384	Record 96 - Var 4	L383	As in tool holder 1	1 2 3 5

① Parameter alterable in continuous through lower case programming (tcp,n).

The figures that follow illustrate the offset characterization parameters applicable to a machine. The machine is seen from above, from the front and from the side.

Fig. 2.1 Front View

Fig 2.2 Side View

Fig 2.3 Machine seen from above

Dynamics

Variable	User Table	Record in PLUS user table	Part Program	Meaning	TCP
Dynamic mode	357	Record 90 - Var 1	L356	These fields may have the following values: 0 - Speeds and accelerations of linear axes are not limited, and may exceed the configured values *run time* 1 - Speeds of linear and rotary axes are limited *run time*, if necessary, in such as never to exceed the configured values. Accelerations of the linear and rotary axes are limited in advance so that overlapping of movements does not require accelerations in excess of those configured. 2 - Speeds and accelerations of the linear and rotary axes are limited in advance so that the speeds and accelerations calculated *run time* never exceed the configured values. In this case, if for example only the linear axis is moved, it remains limited unlike what happened in case 1. It must be stressed that in case 2, the constant speed is guaranteed on the profile, which does not happen in case 1 if speed of an axis is cut *run time.*	1 2 3 5
Corner radius[①] (figures 2.5 and 2.6)	358	Record 90 - Var 2	L357	Radius of the tool corner in cases of spherical or toroidal tools	1, 2, 3, 5
Angle of contact a (figures 2.5 and 2.6)[①]	359	Record 90 - Var 3	L358	Angle a of contact for spherical or toroidal tools (0 £ to £90)	1, 2, 3, 5
Programming of "m", "n" and "o"	360	Record 90 - Var 4	L359	Defines how contact between tool and piece is to be handled. With values "0" and "1", the control is defined through angles which are in turn defined as follows: 0 = through U.T. variables 359 and 400 1 = through m and n from the part program With the 2 value, on the other hand, the contact is established through a normal vector on the profile defined in the 3 components m, n, o.	1, 2, 3, 5

[①] Parameter alterable in continuous through lower case programming (tcp,n).

Dynamics (continued)

Variable	User Table	Record in PLUS user table	Part Program	Meaning	TCP
Threshold angle (figure 2.4)	361	Record 91 - Var 1	L360	Angle beyond which the system adds a horizontal rotary axis move --after the element has been completed-- in order to position the tool perpendicular to the subsequent element. Below this angle, the horizontal rotary axis move starts when the element is completed and continues during execution of the subsequent element.	2
Rotary axis velocity	362	Record 91 - Var 2	L361	Velocity expressed in degrees per minute at which the horizontal rotary axis moves when it is automatically positioned between two part program blocks. If it is 0 the system uses the programmed feedrate.	2
Interpolation mode	363	Record 91 - Var 3	L362	Specifies whether or not the rotary axes are interpolated in conjunction with the linear axes. If not interpolated, the rotary axes follow the linear axes moves. Allowed values are: 0 - Rotary axes interpolated in conjunction with linear axes 1 - Rotary axes follow linear axes Select mode 1 if the velocity on the profile should not be affected by rotary axes motion.	1 2 3 5
Interpolation type	364	Record 91 - Var 4	L363	Specifies whether the rotary axes must be interpolated only by the error or must be treated as linear axes. Allowed values are: 0 - Rotary axes interpolated as linear axes 1 - Rotary axes interpolated only by the error.	1 2 3 5
Integrator	365	Record 92 - Var 1	L364	With interpolation mode 1, this parameter defines whether acceleration/deceleration ramps will be used for rotary axes or whether velocities proportional to the linear axes moves will be used. Allowed values are as follows: 0 - No ramps 1 - Ramps If 1 is selected, the velocity diagram will be calculated using the configured parameters for rotary axes, irrespective of linear axes moves.	1 2 3 5

Dynamics (continued)

Variable	User Table	Record in PLUS user table	Part Program	Meaning	TCP
Offset mode	366	Record 92 - Var 2	L365	Defines the modalities of activation of the length corrector and the tool radius (and possibly the tool corner radius based on the TTR variable). 0 - values taken from the TCP table 1 - values with sign active in the system (length 1 and tool radius) - absolute values active in the system (length 1 and tool radius) - absolute value of length 1 only; the tool radius is considered = 0.	1 2 3 5
Minimum move	367	Record 92 - Var 3	L366	Specifies the minimum angle beyond which the system automatically generates a move of the horizontal rotary axis in order to position the tool perpendicular to the subsequent element.	2
Travel limits control	368	Record 92 - Var 4	L367	Defines how the configured travel limits will be controlled. Allowed values are: 0 - travel limits controlled both in the program block (before execution) and at each point in real time. 1 - travel limits controlled both only at each point in real time. **NOTE:** Since part programs are typically referred to the tool tip, program block control is also referred to the tool tip. Real time control is referred to the center of rotation of the axes. When the system detects that the point is beyond the configured limit, it gives an error signal and locks the axes motion with deceleration.	1 2 3 4 5

NOTE:
In the above tables distances are expressed in mm. They must be intended in the configured length unit.

Fig. 2.4 Threshold angle (α)

The following figures illustrate significance of the TCP parameters used for management of spherical and toroidal tools; the abbreviation U.T. indicates the corresponding number in the User Table.

Fig. 2.5 Toroidal tool

Fig. 2.6 Spherical tool

- **Mode 1 (TCP,1)**

With (TCP,1) programmed coordinates are referred to the part. This mode is recommended when the profile must be executed with both linear (1÷3) and rotary (1÷2) axes, the tool is expected to remain permanently in contact with the machining surface, and the linear axes coordinates are referred to the part.

To keep the tool tip on the profile defined by the linear axes, the system offsets displacements by automatically adjusting the position of the three linear axes to the rotary axes.

- **Mode 3 (TCP,1)**

This mode permits to use CAD programs in which the tool length, radius, corner radius and contact angles are different from the ones with which the program (machine coordinates) was generated.

Tool length and diameter variances must be input via machine logic (refer to the $TCPWRT instruction in the PLUS Library Manual).

As with (TCP,1) in order to keep the tool tip on the profile defined by the linear axes, the system offsets displacements due to variations of length, radii and contact angles by automatically adjusting the position of the three linear axes to the rotary axes.

Profile obtained without TCP 1 or TCP 3

Profile obtained with TCP 1 or TCP 3

- **Mode 2 (TCP,2)**

Mode 2 is an extension of mode 1.

In addition to (TCP,1) features, Mode 2 incorporates a move of the horizontal rotary axis so that its inclination remains constant throughout the profile defined by the first two linear axes configured in the TCP table.

The horizontal rotary axis move may be generated:

- *between two elements*, to allow for rotation on bevels. In this case, the horizontal rotary axis moves either between the end of an element and the start of the subsequent one, or while the subsequent element is executed (refer to the Threshold Angle parameter).
- *on a circular element*, to keep inclination constant throughout the element. In this case the element and the horizontal rotary axis move will occur simultaneously.

These rules apply to all the elements in the profile except for the first one. In fact, the first element is seen as the profile start point even when it is circular and therefore involves no rotary axis move. The system will start generating the horizontal rotary axis move from the second element.

- **Mode 4 (TCP,4)**

(TCP,4) permits the inclination of the horizontal rotary axis to remain constant throughout the profile defined by the first two linear axes configured in the TCP table.

The horizontal rotary axis is adjusted continuously, i.e. between each pair of interpolated points rather than between two elements as in (TCP,2).

For this reason the profile must not be discontinuous. If it is, a "SERVO ERROR" may occur.

Axes moves with TCP,2 or TCP,4

IMPORTANT

(TCP,4) does NOT include the features of (TCP,1).

- **Mode 5 (TCP,5)**

This mode permits to execute moves in the tool direction.

When (TCP,5) is active the system generates a virtual axis whose name has been configured in AMP (VIRTUAL AXES section of the process configuration).

The virtual axis move generates a move of the linear axes in which the tool displaces in its direction, i.e. in the direc(TCP,5) may be programmed both separately or in conjunction with the other axes, in which case (TCP,5) features add to (TCP,1) features. may be moved manually.

Mode 5 may be used for drilling inclined holes. Also, it may be used for removing the tool from the part when a failure interrupts the machining cycle. When this oxccurs, we may have three different cases:

A) (TCP,5) is enabled, the rotary axes are not homed, and the interruption occurred during a cycle using any TCP mode.
In this case the system automatically selects the tool direction according to the position of the rotary axes before the failure. The same may occur after power off, because the rotary axes position is stored in the permanent memory. The values that determine the rotary axes position may be altered via machine logic by means of the MANUAL SETUP data entry or the $TCPWRT function. In particular, $TCPWRT permits to alter the direction in which the tool will be extracted.

B) (TCP,5) is enabled, the rotary axes are not homed, and the interruption occurred during a cycle not using TCP.
In this case the system *must* be given the tool direction, i.e. the position of the rotary axes, because the stored values may not be significant. The rotary axes position may be input via machine logic by means of the MANUAL SETUP data entry or the $TCPWRT function.

C) (TCP,5) is enabled, the rotary axes are homed.
In this case the tool direction is given by the current rotary axes position rather than by their position when the interrupt occurred, no matter whether a TCP mode was active during the previous move. However, in normal working conditions stored and current rotary axes positions coincide.

NOTE:
For more information about the MANUAL SETUP data entry or the $TCPWRT function please refer to the PLUS Library Manual and the User Manual.

Removing the tool with TCP,5

NOTE:
In all TCP modes, the machine logic may send to the interpolator in real time any variance of tool length and tool radius, or of the corner radius, as well of the angles formed between the tool center and the contact point (parameters 386, 399, 400, 358 and 359 in the User Table).

This permits to offset tool wear during machining (refer to the description of the $TCPWRT function in the PLUS Library Manual).

Programming the "m" and "n" parameters (angles)

In all the modes of use of TCP described above, in addition to the axis positions, it is also possible to program the "m" and "n" parameters, significance of which is as follows:

m — angle subtended between the tool center and the point of contact of the tool with the piece (coincides with U.T. parameter 400) - see figures 2.1, 2.5 and 2.6.

n — angle of contact between cutting corner and piece (coincides with parameter 359) see figures 2.5 and 2.6.

By programming these parameters (enabled by setting U.T. variable 359 to 1), you can vary the point of contact between tool and piece from one block to the next.

Example:

G1X10Y0m270
(TCP,1)
G1XY
X10Y10
G3X12Y12|10J12m180
G1X12Y20
(TCP)

Programming the "m", "n" and "0" parameters (vector)

In all the modes of use of TCP described above, in addition to the axis positions, it is also possible to program the "m", "n" and "o" parameters, which represent the 3 components of the normal vector on the profile to be machined.

By programming these parameters (enabled by setting U.T. variable 360 to 2), you can vary the point of contact between tool and piece from one block to the next and obtain tool diameter offset from the resulting vector.

The parameters m, n, o and i, j and k must be programmed in alphabetical order (e.g. mno, mo, no, ijk, ik, etc are correct; nmo, on, om, jik, ki, etc. are incorrect).

Example:
m..n..o ;initial value of vector
(TCP),.... ;TCP activation
X..Y..Z.. AB m..n..o..
X..Y..Z.. AB m..n..o..
X..Z.. m..n..o..
A..B.. m..n..o..
X..Y..
.
.
.
TCP

TCP - Tool Center Point for generic 5-axis machines

This feature permits programming of a 5-axis machine (with 3 linear + 2 rotary axes) irrespective of tool orientation and type (spherical or toroidal). The position controlled by the system is that of the point of attachment of the tool in the spindle (see figures 2.7 and 2.8).

The algorithm may be activated simply by programming the TCP code in a block; it is disabled by programming (**TCP**) or by giving a RESET command.

Syntax

(TCP[,n])

where:

n Offset mode code (1, 3, 5); the various modes are illustrated in the pages that follow.

Characteristics:

In the TCP block you must also write information about the current tools: these tools must be inserted in the "User Table" - abbreviated U.T.

There are three tools you can use for entering data in the user table: the user table editor, the specific machine logic functions, or the L variables in the program.

Also you may upload the predefined configuration files available with the SETUP utility of the USER TABLE EDITOR.

NOTES:

- For more information about user table management from PLUS refer to the description of $TBLPUTD and $TBLGETD functions in the PLUS Library Manual.
- To access the user table from the TABLE EDITOR environment refer to the User Manual.

The following figures illustrate significance of the TCP parameters used for management of spherical and toroidal tools; the abbreviation U.T. indicates the corresponding number in the User Table.

Fig. 2.7 Toroidal tool

Fig. 2.8 Spherical tool

TCP Table for generic 5-axis machines

Variable	User Table	Record in PLUS User Table	Part Program	Meaning	TCP
Offset mode code	385	Record 97 - Var 1	L384	Defines the offset calculation mode according to the machine configuration. **NOTE:** Must be set with value = 6.	1 3 5
Tool length	386	Record 97 - Var 2	L385	Defines the length of the current mounted tool.	1 5
Active tool holder	387	Record 97 - Var 3	L386	Not used.	
Parameter A	388	Record 97 - Var 4	L387	Not used.	
Horizontal head offset	389	Record 98 - Var 1	L388	Not used.	
Horizontal head rotation	390	Record 98 - Var 2	L389	Not used.	
Vertical head rotation (figure 2.2)	391	Record 98 - Var 3	L390	Not used.	
1st linear axis ID	392	Record 98 - Var 4	L391	ID of the axis to be used as abscissa in the offset algorithm.	1 3 5
2nd linear axis ID	393	Record 99 - Var 1	L392	ID of the axis to be used as ordinate in the offset algorithm.	1 3 5
3rd linear axis ID	394	Record 99 - Var 2	L393	ID of the axis to be used as vertical axis in the offset algorithm.	1 3 5
Horizontal head ID	395	Record 99 - Var 3	L394	ID of the axis on which the horizontal head is mounted. If this ID=0, the axis does not exist.	1 3 5
Vertical head ID	396	Record 99 - Var 4	L395	ID of the axis on which the vertical head is mounted. If this ID=0, the axis does not exist.	1 3 5
Horizontal head inclination	397	Record 100 Var 1	L396	Not used.	
Vertical head inclination	398	Record 100 Var 2	L397	Not used.	
Radius (figure 2.1)	399	Record 100 Var 3	L398	Defines the radius of the tool currently mounted on the machine, with reference to the active tool holder.	1 5
Angle (figure 2.1)	400	Record 100 Var 4	L399	Not used.	

Dynamics

Variable	User Table	Record in PLUS user table	Part Program	Meaning	TCP
Dynamic mode	357	Record 90 - Var 1	L356	These fields may have the following values: 0 - Speeds and accelerations of linear axes are not limited, and may exceed the configured values *run time* 1 - Speeds of linear and rotary axes are limited *run time*, if necessary, in such as never to exceed the configured values. Accelerations of the linear and rotary axes are limited in advance so that overlapping of movements does not require accelerations in excess of those configured. 2 - Speeds and accelerations of the linear and rotary axes are limited in advance so that the speeds and accelerations calculated *run time* never exceed the configured values. In this case, if for example only the linear axis is moved, it remains limited unlike what happened in case number 1. It must be stressed that in case 2, the constant speed is guaranteed on the profile, which does not happen in case 1 if speed of an axis is cut *run time.*	1 3 5
Corner radius (figures 2.7 and 2.8)	358	Record 90 - Var 2	L357	Radius of the tool corner in cases of spherical or toroidal tools	1, 3, 5
Angle of contact a	359	Record 90 - Var 3	L358	Not used.	
Programming of "m", and "n"	360	Record 90 - Var 4	L359	Not used.	
Threshold angle	361	Record 91 - Var 1	L360	Not used.	
Rotary axis velocity	362	Record 91 - Var 2	L361	Not used.	
Interpolation mode	363	Record 91 - Var 3	L362	Specifies whether or not the rotary axes are interpolated in conjunction with the linear axes. If not interpolated, the rotary axes follow the linear axes moves. Allowed values are: 0 - Rotary axes interpolated in conjunction with linear axes 1 - Rotary axes follow linear axes Select mode 1 if the velocity on the profile should not be affected by rotary axes motion.	1 3 5
Interpolation type	364	Record 91 - Var 4	L363	Specifies whether the rotary axes must be interpolated only by the error or must be treated as linear axes. Allowed values are: 0 - Rotary axes interpolated as linear axes 1 - Rotary axes interpolated only by the error.	1 3 5

Variable	User Table	Record in PLUS user table	Part Program	Meaning	TCP
Integrator	365	Record 92 - Var 1	L364	With interpolation mode 1, this parameter defines whether acceleration/deceleration ramps will be used for rotary axes or whether velocities proportional to the linear axes moves will be used. Allowed values are as follows: 0 - No ramps 1 - Ramps If 1 is selected, the velocity diagram will be calculated using the configured parameters for rotary axes, irrespective of linear axes moves.	1 3 5
Offset mode	366	Record 92 - Var 2	L365	Defines the modalities of activation of the length corrector and the tool radius (and possibly the tool corner radius based on the TTR variable). 0 - values taken from the TCP table 1 - values with sign active in the system (length 1 and tool radius) 2 - absolute values active in the system (length 1 and tool radius) 3 - absolute value of length 1 only; the tool radius is considered = 0.	1 2 3 5
Minimum move	367	Record 92 - Var 3	L366	Not used.	
Travel limits control	368	Record 92 - Var 4	L367	Defines how the configured travel limits will be controlled. Allowed values are: 0 - travel limits controlled both in the program block (before execution) and at each point in real time. 1 - travel limits controlled both only at each point in real time. **NOTE:** Since part programs are typically referred to the tool tip, program block control is also referred to the tool tip. Real time control is referred to the center of rotation of the axes. When the system detects that the point is beyond the configured limit, it gives an error signal and locks the axes motion with deceleration.	1 3 5

Programming

With this type of TCP, for all the modes described below, both direction (orientation) of the tool and normal direction on the surface to be machined must be defined.

Two vectors are associated with each point programmed, one normal on the surface being machined (parameters m,n,o) and one representing the tool direction (parameters i,j,k).

The parameters m, n, o and i, j and k must be programmed in alphabetical order (e.g. mno, mo, no, ijk, ik, etc are correct; nmo, on, om, jik, ki, etc. are incorrect).

Syntax

[XYZ] [AB] [m n o] [i j k]

where:

XYZAB	movement of axes
mno	normal vector on surface being machined
ijk	vector representing tool direction

Characteristics:

All the parameters are optional; if none are programmed, the previous programming values are reconfirmed.

Before activation of TCP, initial position of the two vectors must be defined, by programming values for them (this block does not cause any movement).
With TCP active, it is not possible to program *m n o* and *i j k* without a movement of the axes XYZAB; furthermore, the vector the define must not be null ([0,0,0]).

NOTE:
The RESET command resets the axes.

Example:

```
m..n..o..  i..j..k..          ;initial value of vectors
(TCP,....)                    ;TCP activation
X..Y..Z..  A..B.. m..n..o..  i..j..k.
X..Y..Z..  A..B.. m..n..o..
X..Z..  m..n..o..  i..j..k.
A..B.. m..n..o..
X..Y..  i..j..k.
.
.
.
.
(TCP)
```

- **Mode 1 (TCP,1)**

With (TCP,1) programmed coordinates are referred to the part.
This mode must be used when the profile must be executed with both linear (1÷3) and rotary (1÷2) axes, the tool is expected to remain permanently in contact with the machining surface, and the linear axes coordinates are referred to the part.
To keep the tool tip on the profile defined by the linear axes, the system offsets displacements by automatically adjusting the position of the three linear axes to the rotary axes.

- **Mode 3 (TCP,3)**

This mode permits to use CAD programs in which the tool length, radius and corner radius are different from the ones with which the program (machine coordinates) was generated.
Tool length, radius and corner radius variances must be input via machine logic (refer to the $TCPWRT instruction in the PLUS Library Manual).

As with (TCP,1) in order to keep the tool tip on the profile defined by the linear axes, the system offsets displacements due to variations of length and radii by automatically adjusting the position of the three linear axes to the rotary axes.

Profile obtained without TCP1 or 3 active

- **Mode 5 (TCP,5)**

This mode permits to execute moves in the tool direction.
When (TCP,5) is active the system generates a virtual axis whose name has been configured in AMP (VIRTUAL AXES section of the process configuration).
The virtual axis move generates a move of the linear axes in which the tool displaces in its direction, i.e. in the direction indicated by the vector [i,j,k].
(TCP,5) may be programmed both separately or in conjunction with the other axes, in which case (TCP,5) features add to (TCP,1) features. The move may also be made manually.

Mode 5 may be used for drilling inclined holes.

NOTE:
For more information about the MANUAL SETUP data entry or the $TCPWRT function please refer to the PLUS Library Manual and the User Manual.

Removing the tool with TCP,5

NOTE:
In all TCP modes, the machine logic may send to the interpolator in real time any variance of tool length, tool radius, or of the corner radius, (parameters 386, 399 and 358 in the User Table). This permits to offset tool wear during machining (refer to the description of the $TCPWRT function in the PLUS Library Manual).

TCP - Tool Center Point for machines with fixed tool and rotary table

This performance allow an easy programming on machines in which the tool is in a fixed position and the part moves by means of three axes, two linear ones and a rotary one solid to the linear ones. The system identifies in real time, according to the rotary axis position, the points that the axes must reach to satisfy the programmed profile. It is also possible to make the system calculate the rotary axis position and therefore to define the plane profile defined by the two linear axes. The activation of this algorithm is carried out directly in the program by means of the three-letters TCP.

The TCP mode is disabled with (**TCP**) or with RESET.

Syntax

(TCP[,n])

where:

n Type of desired compensation (1, 2,)

Characteristics:

The TCP algorithm needs, for its functioning, of some information relative to the used tool, which have to be inserted into the “User Table” - abbr..U.T.

The “User Table” compilation may be executed by means of the proper editor, or by the machine logic (by means of proper logic functions), or also from part program by means of L variables.

It is possible to use a system configuration already pre-arranged contained in the files chargeable by means of "SETUP" of USER TABLE EDITOR.

NOTE:
For further information relative to the use of the “User Table” read the functions description from PLUS $TBLPUTD and $TBLGETD in the manual of PLUS library.

To use the same table from TABLE EDITOR environment see the User manual.

TCP Table for machines with fixed tool and rotary table

Variable	User Table	Record in PLUS User Table	Part Program	Meaning	TCP
Offset mode code	385	Record 97 - Var 1	L384	Defines the offset calculation mode according to the machine configuration. **NOTE:** Must be set with value = 8.	1 2
Tool length	386	Record 97 - Var 2	L385	Not used	
Active tool holder	387	Record 97 - Var 3	L386	Number of the active tool holder 1 ÷ 4	1 2
Parameter A	388	Record 97 - Var 4	L387	Not used.	
Horizontal head offset	389	Record 98 - Var 1	L388	Not used.	
Horizontal head rotation	390	Record 98 - Var 2	L389	Defines the rotation direction of the rotary axis (table) and has a value of: + 1 clock-wise - 1 anticlock-wise for positive programming	1 2
Vertical head rotation	391	Record 98 - Var 3	L390	Not used.	
1st linear axis ID	392	Record 98 - Var 4	L391	ID of the axis to be used as abscissa in the offset algorithm.	1 2
2nd linear axis ID	393	Record 99 - Var 1	L392	ID of the axis to be used as ordinate in the offset algorithm.	1 2
3rd linear axis ID	394	Record 99 - Var 2	L393	Not used.	1 2
Horizontal head ID	395	Record 99 - Var 3	L394	ID of the rotary axis (table)	1 2
Vertical head ID	396	Record 99 - Var 4	L395	Not used	
Horizontal head inclination	397	Record 100 Var 1	L396	Not used.	
Vertical head inclination	398	Record 100 Var 2	L397	Not used.	
Radius	399	Record 100 Var 3	L398	Not used.	
Angle	400	Record 100 Var 4	L399	It defines the angle between the tool center and the cutter contact point after having rotated the vertical head in order to have the tool orientated in y-direction	1 2

Tool holder 1

Variable	User Table	Record in PLUS user table	Part Program	Meaning	TCP
Parameter B	369	Record 93 - Var 1	L368	Sign distance expressed in mm of the table centre from the tool tip as regards the abscissa axis.	1 2
Parameter C	370	Record 93 - Var 2	L369	Sign distance expressed in mm of the table centre from the tool tip as regards the ordinate axis.	1 2
Parameter D	371	Record 93 - Var 3	L370	Not used	
Offset vertical head	372	Record 93 - Var 4	L371	Not used	

Tool holder 2

Variable	User Table	Record in PLUS user table	Part Program	Meaning	TCP
Parameter B (figures 2.1, 2.3)	373	Record 94 - Var 1	L372	As in tool holder 1	1 2
Parameter C (figures 2.1, 2.2)	374	Record 94 - Var 2	L373	As in tool holder 1	1 2
Parameter D	375	Record 94 - Var 3	L374	Not used	
Vertical head offset	376	Record 94 - Var 4	L375	Not used	

Tool holder 3

Variable	User Table	Record in PLUS user table	Part Program.	Meaning	TCP
Parameter B	377	Record 95 - Var 1	L376	As in tool holder 1	1 2
Parameter C	378	Record 95 - Var 2	L377	As in tool holder 1	1 2
Parameter D	379	Record 95 - Var 3	L378	Not used	
Offset vertical head	380	Record 95 - Var 4	L379	Not used	

Tool holder 4

Variable	User Table	Record in PLUS user table	Part Program	Meaning	TCP
Parameter B	381	Record 96 - Var 1	L380	As in tool holder 1	1 2
Parameter C	382	Record 96 - Var 2	L381	As in tool holder 1	1 2
Parameter D	383	Record 96 - Var 3	L382	Not used	
Offset vertical head	384	Record 96 - Var 4	L383	Not used	

Dynamics

Variable	User Table	Record in PLUS user table	Part Program	Meaning	TCP
Dynamic mode	357	Record 90 - Var 1	L356	These fields may have the following values: 0 - Speeds and accelerations of linear axes are not limited, and may exceed the configured values *run time* 1 - Speeds of linear and rotary axes are limited *run time*, if necessary, in such as never to exceed the configured values. Accelerations of the linear and rotary axes are limited in advance so that overlapping of movements does not require accelerations in excess of those configured. 2 - Speeds and accelerations of the linear and rotary axes are limited in advance so that the speeds and accelerations calculated *run time* never exceed the configured values. In this case, if for example only the linear axis is moved, it remains limited unlike what happened in case 1. It must be stressed that in case 2, the constant speed is guaranteed on the profile, which does not happen in case 1 if speed of an axis is cut *run time.*	1 2
Corner radius[1]	358	Record 90 - Var 2	L357	Not used	
Angle of contact a	359	Record 90 - Var 3	L358	Not used	
Programming of "m", "n" and "o"	360	Record 90 - Var 4	L359	Not used	
Threshold angle (figure 2.4)	361	Record 91 - Var 1	L360	Angle beyond which the system adds a horizontal rotary axis move --after the element has been completed-- in order to position the tool perpendicular to the subsequent element. Below this angle, the horizontal rotary axis move starts when the element is completed and continues during execution of the subsequent element.	2
Rotary axis velocity	362	Record 91 - Var 2	L361	Velocity expressed in degrees per minute at which the horizontal rotary axis moves when it is automatically positioned between two part program blocks. If it is 0 the system uses the programmed feedrate.	2

Dynamics (continued)

Variable	User Table	Record in PLUS user table	Part Program	Meaning	TCP
Interpolation mode	363	Record 91 - Var 3	L362	Specifies whether or not the rotary axes are interpolated in conjunction with the linear axes. If not interpolated, the rotary axes follow the linear axes moves. Allowed values are: 0 - Rotary axes interpolated in conjunction with linear axes 1 - Rotary axes follow linear axes Select mode 1 if the velocity on the profile should not be affected by rotary axes motion.	1 2
Interpolation type	364	Record 91 - Var 4	L363	Specifies whether the rotary axes must be interpolated only by the error or must be treated as linear axes. Allowed values are: 0 - Rotary axes interpolated as linear axes 1 - Rotary axes interpolated only by the error.	1 2
Integrator	365	Record 92 - Var 1	L364	With interpolation mode 1, this parameter defines whether acceleration/deceleration ramps will be used for rotary axes or whether velocities proportional to the linear axes moves will be used. Allowed values are as follows: 0 - No ramps 1 - Ramps If 1 is selected, the velocity diagram will be calculated using the configured parameters for rotary axes, irrespective of linear axes moves.	1 2
Offset mode	366	Record 92 - Var 2	L365	Not used	
Minimum move	367	Record 92 - Var 3	L366	Specifies the minimum angle beyond which the system automatically generates a move of the horizontal rotary axis in order to position the tool perpendicular to the subsequent element.	2

Dynamics (continued)

Variable	User Table	Record in PLUS user table	Part Program	Meaning	TCP
Travel limits control	368	Record 92 - Var 4	L367	Defines how the configured travel limits will be controlled. Allowed values are: 0 - travel limits controlled both in the program block (before execution) and at each point in real time. 1 - travel limits controlled both only at each point in real time. **NOTE:** Since part programs are typically referred to the tool tip, program block control is also referred to the tool tip. Real time control is referred to the center of rotation of the axes. When the system detects that the point is beyond the configured limit, it gives an error signal and locks the axes motion with deceleration.	1 2

NOTE:
In the above tables distances are expressed in mm. They must be intended in the configured length unit.

Programming

The rotary table virtualization allows the programming of machines with tool in fixed position and the part moving by means of three axes, two linear ones and one rotary solid to the two linear ones. It is activated by commands (TCP, 1) or (TCP, 2).
By programming (TCP, 2) the automatic insertion of rotary axis is obtained. (TCP, 1) and (TCP, 2) performances and characteristics described for 5-axis machines are valid also for this kind of compensation.
For a correct programming on rotary table machines, the following must be taken into consideration:

a) the system considers to have, at the end of the homing cycle, coincident tool tip and table centre. In case this situation does not happen, the values representing the distance of the tool centre from the tool tip must be defined in the TCP table (T.U.369 e T.U. 370)

b) the profile must be defined as regards an origin coincident with the table centre, therefore, as described in the previous point, if table centre and tool tip do not coincide, the origin must be defined with the two values of abscissa and ordinate equal to the phase displacement defined in the TCT table (T.U. 369 e T.U. 370).

Example:

T.U. 369 = - 40 Distance, as regards X, of the table centre from the tool tip
T.U. 370 = + 110 Distance, as regards Y, of the table centre from the tool tip

```
ORIGIN 1 (POINT O)  X-40      Y+110
(UOT,1,X-70,Y-70)             ; POINT A
GX-10Y-10
(TCP,2)
G1 Y0  F2000
X+140
Y+130
G3 X0  Y130  I70  J90
G1Y-10
(TCP)
```

TCP on multi-processor

The TCP feature is now extended to 4 processes, each process will use a setup area of its own in the User tables as shown in the table below.

Process	Tabble	Plus Table Record	Part Program LVariable
1	TCP Table	Record 97	L384 ÷ L399
	Tool-holder 1	Record 93	L368 ÷ L371
	Tool-holder 2	Record 94	L372 ÷ L375
	Tool-holder 3	Record 95	L376 ÷ L379
	Tool-holder 4	Record 96	L380 ÷ L383
	Dynamics	Record 90	L356 ÷ L367
2	TCP Table	Record 86	L340 ÷ L355
	Tool-holder 1	Record 82	L324 ÷ L327
	Tool-holder 2	Record 83	L328 ÷ L331
	Tool-holder 3	Record 84	L332 ÷ L335
	Tool-holder 4	Record 85	L336 ÷ L339
	Dynamics	Record 79	L312 ÷ L323
3	TCP Table	Record 75	L296 ÷ L311
	Tool-holder 1	Record 71	L280 ÷ L283
	Tool-holder 2	Record 72	L284 ÷ L287
	Tool-holder 3	Record 73	L288 ÷ L291
	Tool-holder 4	Record 74	L292 ÷ L295
	Dynamics	Record 68	L268 ÷ L279
4	TCP Table	Record 64	L252 ÷ L267
	Tool-holder 1	Record 60	L236 ÷ L239
	Tool-holder 2	Record 61	L240 ÷ L243
	Tool-holder 3	Record 62	L244 ÷ L247
	Tool-holder 4	Record 63	L248 ÷ L251
	Dynamics	Record 57	L224 ÷ L235

END OF CHAPTER

PROGRAMMING TOOLS AND TOOL OFFSETS

This chapter explains how to program tools and tool offsets. All the functions described in this chapter must be handled by the machine logic.

The system integrator develops the interface between the control and the machine tool, as well as the application-specific T and M functions and also of the code requalifying functioning and the presetting: RQT and RQP.

For more information about these T functions you must refer to the machine tool documentation provided by the machine manufacturer.

You can also find further information about M functions in Chapter 6.

T address for programming tools

The T address defines the tool and the tool offset used in a given machining process.

Syntax

T [*tool*], [.] [*tool offset*]

where:

tool is the tool number. This can be programmed explicitly (with an integer) or implicitly with a local or system variable.

tool offset is the tool offset number. It can be an integer or an E parameter.

Characteristics:

Allowed values for both parameters ranges from 0.0 to 999999999999.300. The meaning of the 15 digits is as follows:

999999999999.300

tool offset number

tool number

Examples:
The following examples show different ways of programming tools and offsets.

T1 Selects tool 1 and the tool offset defined in the tool table.

T1.1 Selects tool 1 and tool offset 1.

T1.0 Selects tool 1 without tool offset.

T.0 Removes the tool offset from the active tool.

T0 Disables the active tool and tool offset.

T.1 Enables tool offset 1 with the active tool.

T address for multi-tool programming

The T address defines the tools to be used simultaneously for a given machining process.

Sintax

T [*master*] [.] [*tool offset*] [/{*slave* l] [first_*slave, last slave*]l **{***variable, num_variables***}**}]

where:

master	Is the number of the tool. This can be an integer or a local or system variable between 0 and 999999999999.
tool offset	Is the number of the tool offset associated with the master tool. This can be an integer or a local or system variable between 0 and 300.
slave	Is the number of the tool. This can be an integer or a local or system variable.
first_slave	Is a tool number representing the first of a series of tools. This can be an integer or a local or system variable.
last_slave	Is a tool number representing the last of a series of tools. This can be an integer or a local or system variable.
variable	Is a local or system variable containing the first of a series of tools.
num_variable	Is an integer or a local or system variable representing the number of variables to take into consideration starting from "variable".

Characteristics:

Multi-tool programming is used on perforating machines.

The management of tools associated with the T code is handled by the logic of the machines to which the values of the programmed tools are sent..

The programmable values for the slave tool codes vary from 0 to 65535. The maximum number of slave tools which can be simultaneously programmed is 60.

As you can see from the T code sintax, the list of tools to be used can be specified in three different formats:

1. Single Format

Examples:

T1.2/ 50	tools	1, 50
	offset	2
T1.2 /20,33,45,46	tools	1,20,33,45,46
	offset	2

2. Numerical Sequence Format

This simplifies programming of multiple tools by using sequential code.

Examples:

T1.3 /[30, 35]	equivalent to T1.3 /30,31,32,33,34,35
T1.3 /[56, 51]	equivalent to T1.3 /56,55,54,53,52,51
T1.3 /[50, 52], [10,13]	equivalent to T1.3 /50,51,52,10,11,12,13

As you can see, the starting tool number can be > or < the final tool number. In the first instance the tools codes are considered in ascending order and in the second instance they are considered in descending order.

3. Variable Sequence Format

This is based on an array of variables from which to select the tool numbers.

Example:

E0 = 1, 30, 45	load in E0, E1, E2 the values 1, 30, 45
T1.2 /{ E0, 3 }	equivalent to T1.2 /E0, E1, E2 & to T1.2 /1, 30, 45

E0 = 1, 30, 45	load in E0, E1, E2, E3 the values 1, 30, 45
SN0 = 4, 77	load in SN0, SN1 the values 4, 77
SN4 = 3	load in SN4 the value 3
T1.2 /{ E0, SN4 }, { SN0,2 }	equivalent to T1.2 /E0, E1, E2, SN0, SN1 & to T1.2 /1,30,45,4,77

The three formats shown above may be mixed.

Example:

E0 = 29, 56	load in E0, E1 the values 29, 56
SN6 = 2	load in SN6 the value 2
T1.3 /[7, 10], 15, { E0, SN6 }	equivalent to T1.3 /7,8,9,10,15,29,56

h address

The h address permits to apply a tool offset both in point-to-point (G29) and in continuous (G27-G28) modes.

Syntax

h [*tool offset*]

where:

tool offset is the tool offset number. It can be an integer or a local or system variable between zero and 300.

Characteristics:

The "h" address must be programmed by itself in one block.

An "h" address disables the active offsets programmed with a "T" command.
The axes to which *tool offset* is applied are those programmed in a "T *tool.offset"* command.

The *offset* values are applied to the axes when the system reads the "h" address in the part program. The "h" address must not be synchronised either with the logic or with the axes moves.

IMPORTANT

If “h” is not synchronized, it is displayed when it is read by the part program and not when it is implemented.
The same occurs in the case of synchronous uses of “h”, as in G96. In this case, the application of “h” must be synchronized with “#” for the spindle revolutions to function correctly.

If you program "h" without *tool offset* or with *tool offset=0,* the value programmed with the previous

*"*h *tool offset"* command. This value may be re-established with a RESET command.

IMPORTANT

The machine logic decides whether or not to apply offset values after a RESET.

IMPORTANT

The use of "h" and **T** for offsets in the same part program can cause problems if it is not handled correctly by the machine logic. It is recommended that only one of these modes be used at the same time.

The offset values programmed with h address are usually displayed in the field reserved for the T address.

IMPORTANT

Offsets may be introduced with an h address only if the default configuration of the offsets table is used.

The offset table fields from which offset values are read are as follows:

Offset value	Offset table fields
Length 1	Length 1 + Current requalification length 1
Length 2	Length 2 + Current requalification length 2
Diameter	Diameter + Current requalification diameter

Example 1:

.....
.....
.....
T[n].x programs a tool and a tool offset
.....
.....
hy cancels the **x** tool offset and enables the **y** tool offset
.....
.....

Example 2:

.....
.....
.....
hy enables **y** tool offset
.....
.....
T[n].x cancels the **y** tool offset enabled by **h**
.....
.....

Example 3:

.....
.....
.....
hx
.....
.....
hy cancels the programmed **x** tool offset and enables **y**
.....

TTR – Toroidal Tool Radius

This process variable makes it possible to enable the management of the tool corner radius when using spherical or toroidal tools.

Syntax

TTR = value

where:

value This value can be:

0 Disables the management of the tool corner radius

1 Enables the management

NOTE

The moment the tool corner radius management is enabled, the length 2 field (+ requalification length and max. length) is used by the system to define the tool tip radius. In this connection, the length is no longer applied as a length corrector.
The tool corner radius is used only within the TCP and the HSM for programming a T or an h.

AXO - Axis Offset Definition

This command makes it possible to associate the length offset values contained in the offset table to the machine axes.

Syntax

(AXO, [-] *axis name1* [**,** [-] *axis name2*]**)**

where:

axis name1 name of the axis associated to length offset 1 in the offsets table. If the "-" sign is written before the name, the offset value is used with the inverse sign.

axis name2 name of the axis associated to length offset 2 in the offsets table. If the "-" sign is written before the name, the offset value is used with the inverse sign.

Characteristics:

The default association between length offsets and characterised axes is made in the AXIS GENERAL INFO page of the AMP utility.

This association can be changed with the AXO command.

The preset values introduced in the offset tables always have a positive sign. The AXO command enables these values to be associated with axes bearing negative signs.

The following are two examples of length offset values applied with and without AXO commands:

Example 1:

.
.
.

N1 T1.4 M6 activates length 1 of offset 4 on the axis associated to length 1 in the offset characterisation phase.

.

N100 T0 M6 disables the length offset value applied to the axis.

Example 2:

.
.
.

N1 (AXO,-X,Z)	associates X to length offset 1 with negative sign and Z to length offset 2 with positive sign
.	
.	
N50 T1.4 M6	applies length offset values 1 and 2 from offset 4 to axes X and Z as defined in the AXO command. Length offset 1 will be applied to the X axis with negative sign.
N100 T0 M6	disables the length offset applied to axes X and Z.

IMPORTANT

The system RESET command or the selection of a new part program re-establish the axis/offset default association characterised with AMP.

RQT (RQU) - Requalifying Tool Offset

The RQT command requalifies the length and diameter dimensions memorised in the offset tables.

Syntax

(RQT,*tool***,***offset* [**,L**..][**,I**..][**,d**..]**)**

where:

tool Is the tool number.

offset Is the number of the offset to be requalified. The offset number is a value between 1 and 300.

L Is the value to be added to offset length 1.

I Is the value to be added to offset length 2.

d Is the offset increment to be added to the diameter.

Characteristics:

The values can be either numbers or contents in the variables.

You must specify the length and diameter increments in the RQT command with the measuring unit configured in the system (inches or millimetres, G70/G71). These values cannot be associated to a scale factor (SCF).

Examples:

(RQT,10,1,L E40,d E41) This block requalifies tool 10, offset 1. The length 1 increment is contained in E40 and the diameter increment is contained in E41.

(RQT,10,1,L E50,I E51) This block requalifies tool 10, offset 1. The length 1 increment is contained in E50 and the length 2 increment is contained in E51.

RQP - Requalifying Tool Offset

The RQP command requalifies and presets a specific tool offset according to programmed length and diameter dimensions. When the control executes this command, it writes the corresponding dimensions in the tool offset table by adding the specified length and diameter values. All the offset values are reset.

Syntax

(RQP,*tool***,***offset* [**,L**..] [**,I**..][**,d**..])

where:

tool Is the tool number.

offset Is the number of the offset to preset. It is a value between 1 and 300. The upper limit of the offset number depends on the number of records in the tool offset file.

L Is the length increment to write to length offset 1.

I Is the length increment to write to length offset 2.

d Is the diameter increment to write to the offset field.

Characteristics:

The values can either be some numbers or contents in the variables.

You must specify the values of length and diameter increments in the RQP command with the measuring unit configured in he system (inches or millimetres, with G70/G71). These values cannot be associated to a scale factor (SCF).

Examples:

(RQP,10,1,L E40 ,d E41) This block presets offset 1 of tool 10. The value of length 1 is contained in E40 and the value of diameter is contained in E41.

TOU (TOF) - Tool Expiry Declaration

The TOU command allows to declare the specified tool expired.

Syntax

(TOU,tool**)**

where:

tool is the number of the tool number to be declared expired. It can be a local or a system variable

Example:

```
(TOU,5)     ;Declares tool 5 in the tool table expired
E1=10       ;
(TOU,10)    ;Declares tool 10 in the tool table expired
```

LOA - Table loading

The LOA command allows dual port loading of the disk-resident table.

Syntax

(LOA, *table name* [.*extension*] [,*par1*] [,*par2*]]**)**

where:

table name it is the file name of the table to be dual port loaded.

extension it is the extension of the file name and refers to the type of table to be loaded:

.TOL	table of tools
.OFS	table of adapters
.SPN	table of spindles (for the WOOD option)
.ORG	table of origins
.USR	table of user variable
.MAG	table of magazines

When there is no extension, the table referred to is the one pertaining to the Electronic Cam option (for description see relevant manual).

When tool and adapter have been associated in the loading of the tools table, also the relevant strings are loaded automatically.

The name of the table with its extension may also be defined by means of the variable SC.

par 1 It is a parameter having a different meaning according to the type of table to be loaded:

table of tools ⇒ *par 1* stands for the number of magazine the tools belong to and it is compulsory when magazines are being used.

table of origin ⇒ *par 1* par1 is the number of the process to which the axes relevant to the origins to be loaded belongs to.

par 2 It is considered only when loading the table tools together with the parameter par 1 and indicates whether the pocket values of the tools table need to be updated.

par 2 = 0 no pocket update

par 2 = 1 pocket update

When no parameter is available, 0 is chosen by default.

Characteristics:

The LOA command enables a dual port loading of the tables previously saved on file by TABLE EDITOR.

The names for the relevant extensions which identify the type of table are given by default and can be different if modified through the SET UP utility of TABLE EDITOR.

When using the magazines, the operation "pocket initialization" is to be accomplished by TABLE EDITOR. Such operation is necessary each time tables referred to magazines featuring different pockets are loaded or after reset of the dual port memory.

Examples:

(LOA, NEW_TOOL.MAG)	loads the magazine table named NEW_TOOL.
(LOA, DRILL_T.TOL, 1, 1)	loads the DRILL_IT tools table relevant to magazine 1 and updates the pocket field of each tool.
(LOA, ORIG_1.ORG, 1)	loads the ORIG_1 table relevant to process 1.

END OF CHAPTER

CUTTER DIAMETER COMPENSATION

Cutter diameter compensation is based on the tool geometry. Since the typical tool section is a circle, the correction is applied to the tool diameter.

Cutter diameter compensation is perpendicular to a programmed profile consisting of straight line segments and circle arcs. The figure below shows how the tool is positioned when cutter diameter compensation is applied.

Tool Positioning for Cutter Diameter Compensation

When cutter diameter compensation is active, the tool is positioned on the intersection or tangency point of the two geometrical elements translated by the cutter radius.

G40 G41 G42 - Cutter Diameter Compensation

These G codes disable or enable cutter diameter compensation:

G40 Disables cutter diameter compensation

G41 Enables cutter diameter compensation when the tool travels left of the profile

G42 Enables cutter diameter compensation with the tool travels right of the profile

Syntax

G40 [*G-codes*] [*operands*]

G41 [*G-codes*] [*operands*]

G42 [*G-codes*] [*operands*]

where:

G-codes Other G codes that are compatible with G41, G42 and G40 (see “Compatible G codes” table in Chapter 1).

operands Any operand or code allowed in a G function block.

Enabling Cutter Diameter Compensation

Cutter diameter compensation is enabled by codes G41 or G42 . The movement to the first point in the profile must be linear (G00-G01).

On the first point in the profile, cutter diameter compensation is perpendicular to the first linear or circular move programmed after G41 or G42 on the active plane.

The examples below show how cutter diameter compensation is applied.

- **Linear move first in the profile**

Program:

Cutter diameter compensation on the right of the profile:

G1 G42 X-50 Y15 F200
X-20 Y45

Cutter diameter compensation on the left of the profile:

G1 G41 X-50 Y15 F200
X-20 Y45

- **Circular move first in the profile**

Program:

Cutter diameter compensation on the right of the profile:

G1 G42 X-31.622 Y40 F200
G2 X33.541 Y35 I J25
G1 X.....
Cutter diameter compensation on the left of the profile:

G1 G41 X-31.622 Y40 F200
G2 X33.541 Y35 I J25
G1 X....

Notes on using cutter diameter compensation

Once enabled, cutter diameter compensation is applied to all the moves programmed at machining or rapid feedrate. After cutter diameter compensation has been enabled by G41 or G42, the following G functions cannot be programmed:

- G81 - G89 (canned cycles)
- G70 - G71 (mm/inch programming)
- G79 (programming referred to machine zero)
- G33 (threading)
- G72 - G73 - G74 (measuring cycles)
- G16 - G17 - G18 - G19 (change of interpolation plane)

When cutter diameter compensation is active, the control displays an error signal if:

- the programmed internal radius is shorter than the tool radius.
- the execution of a compensated linear move would reverse the tool direction with respect to the original profile.

Inside a profile compensated with G41 or G42 you can program up to two consecutive movements of axes that are not on the interpolation plane.

Tool path optimisation (TPO)

Tool path optimisation (TPO) may be enabled both from part program or with an MDI.

The function is programmed through the TPO system variable and may be customised to specific application requirements through the TPT system variable.

TPO permits to define two optimisation modes:

- automatic “reduction” of the tool path on corners between two linear or circular blocks
- infeed/exit tangent to profile (on a circle arc).

IMPORTANT

TPO and TPT are discussed in detail later in this chapter.

Disabling Cutter Diameter Compensation

Cutter diameter compensation is disabled by the G40 code. Then the cutter may exit from the part as follows:

- if TPO is not active (standard offset mode), the move programmed by G40 is still considered offset;
- if TPO is active, the move programmed by G40 is considered as the exit move from the offset.

These options are illustrated in the examples below.

- ***The last move in the profile is linear (TPO=0)***

Program:

N88 G1 G40 X50 Y15
N100 G X.. Y..

- ***The last move in the profile is circular (TPO=0)***

Program:

N99 G2 G40 X36.62 Y40 I J25
N100 G X.. Y..

Disabling Compensation with TPO active

With TPO active, the cutter diameter is always compensated through G40; the only difference with respect to the standard procedure is how the last block is executed.

These options are illustrated in the examples below.

- ***The last move in the profile is linear (TPO active)***

Program:

N87 G1 X20 Y40
N88 G1 G40 X50 Y15
N100 G X.. Y..

- ***The last move in the profile is circular (TPO active)***

Program:

N98 G1X-33.541 Y35
N99 G2 G40 X36.62 Y40 I J25
N100 G X.. Y..

If your release is earlier than 3.0 you must check whether G40 is programmed in your old part programs. When TPO is active, G40 exits from cutter diameter compensation somewhat differently.

If you program a block including only G40 (i.e. not associated to any final point), you will obtain the same result you would with standard diameter compensation (TPO=0).

Prior to writing a program read this section and the descriptions of TPO and TPT carefully.

TOOL DIAMETER COMPENSATION CHANGE

This section describes how changes of compensation type (G41 --> G42 and vice versa) are handled during offset profile machining. Compensation type changes may occur at the point of intersection of the programmed paths (with left/right or right/left compensation) or by the addition of a new movement block automatically by the system.

Type of compensation change (on the intersection or with an additional connection block) depends on type of the previous movements and of the subsequent movement. The different possibilities are discussed in the following pages:

- Linear/Linear with tangential movement blocks
- Linear/Linear with inversion of direction
- Linear/Linear with automatic generation of a new movement block
- Linear/Linear without automatic generation of a new movement block
- Linear/Circular - Circular/Linear
- Circular/Circular

Linear/Linear tool path

The following figure illustrates the tool path when compensation changes from G41 to G42 during execution of two linear type movements.

In changing from G41 to G42, the control generates two points, which shall be called *point 1* and *point* 2.

- *Point 1* is the final position of the tool before compensation type change.
- *Point 2* is the desired starting position for the first block using the changed compensation direction.

The control automatically generates the movement block connecting point 1 with point 2:

Linear/Linear change with tangential movement blocks

Linear/Linear change with inversion of direction

Linear/Linear change with generation of a new block

Linear/Circular, Circular/Linear, Circular/Circular tool paths

For each of the following types of tool path, in which a change of compensation direction occurs, the 10 Series system will try to find a point of intersection between the path programmed in G41 and that programmed in G42 (or vice versa).

If the 10 Series finds a point of intersection, it modifies the *final point* of the original compensated tool path whereas the *starting point* of the new compensated tool path will coincide with the intersection (see figure below).

Change of compensation with intersection of current path

However, cases may arise in which there is no intersection between the tool paths; in these cases, when changing from G41 to G42 (or vice versa) the system behaves as illustrated in the figures that follow.

- *Point 1* is the final position of the tool before the change of compensation type
- *Point 2* is the desired position for start of the first block using the changed compensation direction.

The control automatically generates the movement block connecting point 1 with point 2:

Change of compensation without possibility of intersection between the tool paths

r - Radiuses in Compensated Profiles

When machining convex profiles, you may want a circular radius between geometric elements.

Syntax

r value

where:

value The radius to be programmed. For clockwise moves program a negative radius; for counter clockwise moves program a positive value.

IMPORTANT

Programming r0 (radius equal to zero) causes the tool centre to follow a circular arc whose centre is on the profile corner.

Example:

Without radius

1) N20 G1 X100 Y100
2) N21 X-100

With radius

1) N20 G1 X100 Y100
   N21 r0
2) N22 X-100

b - Bevels in Compensated Profiles

By programming a value without a sign after the b address you can insert a bevel rather than a radius between two linear or circular motion blocks that generate intersecting paths.

Syntax

b value

where:

value It is the bevel length measured from the intersection point.

This value **b** may be interpreted as follows:

- **Bevel between two linear profiles:** ***b*** is the distance from the generated final point to the theoretical intersection point between the extended profile segments.
- **Bevel between a linear and circular profile:** ***b*** is the distance from the theoretical intersection point between the extension of the linear profile to the tangent of the circular profile
- **Bevel between two circular profiles:** ***b*** is the distance to the theoretical intersection point between the extensions of the tangents to the circular profiles

Examples:

Bevel between two linear blocks

Programming example:

```
        .
        .
N10 G1 X10 Y100     ;block 1
N11 b3
N12 X-100           ;block 2
        .
        .
```

Bevel between circular and linear motion blocks

Programming example:

```
.
N10 GXY
N11 G42
N12 G1X10Y70F1000
N14 G2X70Y20R50      ;block 1
N16 b10
N18 G1X150Y90        ;block 2
N20 G40
.
.
.
```

Bevel between linear and circular motion blocks

Programming example:

```
      .
N20   G42
N22   GXY
N24   G1X40Y40F1000      ;block 1
N26   b10
N28   G2X80Y5R70         ;block 2
N30   G40
      .
      .
```

Bevel between two circular motion blocks

Programming example

```
        .
        .
N10     G42
N20     GX10Y60
N30     G2X50Y40R50F1000    ;block 1
N40     b5
N50     G2X100Y5050R30      ;block 2
        .
        .
```

TPO - Path optimisation on bevels with G41/G42

TPO (Tool Path Optimisation) allows to optimise the tool path when G41 or G42 are active. The algorithm automatically introduces circular interpolations at profile start and end, and radiuses on the profile bevels.

Syntax

TPO = *n*

where:

n Optimisation mode. Allowed values are:
The value to be specified for *n* in the instruction is obtained from the sum of the decimal weights corresponding to each of the features desired.
If *n* = 0 the optimisation algorithm is disabled.

Characteristics:

A block programming the TPO=1 instruction activates the tool optimisation algorithm. Depending on the profile, the algorithm automatically introduces from 1 to 3 optimisation moves.

IMPORTANT

TPO is ignored in a GTL program.

IMPORTANT

At power up the system is enabled with the TPO mode configured in AMP. This optimisation mode may be altered from the program.
RESET restores the mode configured in AMP.

IMPORTANT

The activation of bit 5 works only if bit 0 is enabled too, i.e. when you want to introduce a circular radius in lieu of a linear one, set **TPO = 33**.

- **TPO=1 and TPO=33 modes**

Mode TPO=1 activates the tool path optimisation algorithm by introducing linear radiusing movements between the profile elements as a function of the angular deviation between the elements themselves.

Furthermore, if bit 5 has been set too (TPO=33), then, provided that the conditions for corner optimisation are met, through the introduction of linear movements a circular radius is introduced, having its centre at the intersection of the two non-compensated elements and a radius corresponding to tool radius.

The cases of application of the algorithm are as follows:

A) Right angle-right angle with angle of deviation between 90° and 180° ($90° < \alpha \Leftrightarrow 180°$).

B) Circle-right angle, right angle-circle, circle-circle with angle of deviation between 0° and 180° ($0° < \alpha \Leftrightarrow 180°$).

If a bevel (b) or a radius (r) is programmed between the elements described at points A) and B), the algorithm is not applied.

Example:

Without optimization (TPO =0)

With optimization (TPO=1)

Con ottimizzazione (TPO=33)

IMPORTANT

Note that in the above examples the TPT threshold has been ignored, as if TPT=0.
If a threshold distance from the corner is programmed instead (TPT other than zero), the afore-mentioned circular radius will have its centre translated according to the same calculation criteria as are used for linear radiuses, and the radius will be reduced accordingly.

Examples of profile optimisation with TPO=1

In these profiles the angle deviation requires an optimisation algorithm.

Angles from 0° to 90° formed by two straight lines

180° angle with circle-circle intersection

Angle from 0° to 180° formed by a circle-straight line intersection

Examples of profiles where algorithm introduces optimization moves

In these examples the algorithm recalculates the profile and introduces from 1 to 3 optimisation moves between bevels.

1) Line - line

The algorithm generates 1 move

2) Line - circle

The algorithm generates 2 moves

3) Circle - line

The algorithm generates 2 moves

4) Circle - circle

The algorithm generates 3 moves

- **TPO=2 mode**

TPO=2 enables an infeed/exit algorithm that keeps the tool tangent to the profile by introducing circular elements at profile start and end.

At profile infeed the algorithm generates a circle between the first point of the offset profile (P1) and the preceding point (P0). The first point is the one programmed in the G41 or G42 block.

Example:

```
.....
X80 Y70              ;P0
G41 X100 Y100        ;P1
X140                 ;P2
.....                ;etc.
```

At profile exit the algorithm generates a circle between the last point of the offset profile (P99) and the last point of the exit element in the profile. The exit element is programmed in the G40 block.

Example:

```
.....
G41 X100 Y100        ;P1
.....
.....
.....
X170 Y160            ;P99
G40 X180 Y185        ;P100
.....                ;etc.
.....
```

- **TPO=3 mode**

TPO=3 simultaneously enables TPO=1 and TPO=2 algorithms, which allows to introduce both radiuses between elements and circular moves at profile start and end.

The rules defined for TPO=1 and TPO=2 also apply to TPO=3.

Examples of TPO=2 mode

Example 1:

```
N1    S1000 M3 T1.1M6          ;tool radius = 2.5
N2    X5 Y5
N3    G1 G42 X15 Y15 F500      ;first point in the profile
N4    X30
N5    X40 Y30
N6    X32.5 Y42.5              ;last point in the profile
N7    G40 X32.5 Y52.5
N8    GX100
```

Example 2:

```
N1    S1000 M3 T1.1 M6        ;tool radius = 2.5
N2    X0 Y0
N3    G1 G41 Y15 F800         ;first point in the profile
N4    X -20
N5    Y45
N6    X40
N7    Y15
N8    X0                      ;last point in the profile
N9    G40 Y0
```

- **TPO=4 mode**

The TPO=4 mode activates the profile inversion algorithm.

This feature permits machining of a profile starting from the last movement, and finishing on the first (reverse machining).

IMPORTANT

- This feature in managed in connection with commands GTP and CCP only.
- The algorithm also provides for inversion of any offset modes, so as to maintain position of the inside/outside tool as required.
- For further details, see the pages dealing with the triliterals GTP and CCP (page 4-34).

- **TPO=8 and TPO=16 modes**

TPO=8 and TPO=16 modes activate the algorithm that keeps the machining speed in G41/G42 constant. In particular, the speed of contact between the tool and the part is kept constant, varying the feed rate with reference to the centre of the tool.

This variation is a function of the tool radius and is only applied to circular movements. It will produce an increase or decrease in the feed rate with reference to the centre of the tool depending on whether the radius of the circle moved by the centre of the tool is greater or less than the radius of the programmed circle. The feed rate is increased by setting bit 3 of variable TPO (TPO=8) while it is decreased by setting bit 4 (TPO=16).

Bits 3 and 4 of TPO may be set together and even at the same time as all the other bits of the same variable to enable the corresponding features.

- **TPO=32 mode**

Is the same as having programmed TPO = 0.
This is so because the activation of bevel optimisation through the introduction of a circular radius requires the prior programming of TPO = 1. If bit 0 = 0 and bit 5 = 1 (TPO=32), the optimisation algorithm by means of circular radiuses is not enabled.

TPT - Tool Path Threshold

This instruction specifies a threshold for bevels during tool path optimisation when TPO=1.

Syntax

TPT = value

where:

value Threshold expressed in the default unit of measure (mm/inch). It represents the distance between the tool cutter and the bevel generated by the programmed profile. It can also be configured in AMP.

Example:

Characteristics:

The threshold programmed with TPT will be ignored by the system if it takes the mill centre beyond the theoretical point obtained through standard tool diameter compensation (G41/G42 without TPO programming).

In this case the TPO algorithm will be temporarily disabled, G41/G42 will be applied without TPO and there will be no error signal.

Example:

u v w - Paraxial Compensation

When compensation factors u,v,w are programmed in a block, the axes position to a point whose coordinates are equal to the programmed coordinates plus the product of the cutter radius by the compensation factor (u, v, w):

X position = programmed X + (cutter radius * u)

Y position = programmed Y + (cutter radius * v)

Z position = programmed Z + (cutter radius * w)

These compensation factors are used both for extremely simplified profiles (contouring parallel to axes) and three-dimensional milling surfaces. You cannot use the u,v,w factors when the control is in cutter diameter compensation mode (G41-G42).

When factors u,v,w are negative, they must be followed by a minus sign. If they are positive, the plus sign can be omitted.

In order to determine the value and the sign you must consider paraxial factors u,v,w respectively as the X, Y, Z coordinates of the profile vertex corrected by a unit. X, Y and Z are referred to a system of cartesian axes that are parallel to the axes of a machine whose origin is represented by the point to be compensated.

Determining the sign of u, v, w compensation factors

The unit vector represents a cutter radius offset that is one unit long; the control, the increments of translation are the product of the u,v,w values multiplied by the cutter radius.

You can compensate profiles that consist of:

- straight line segments parallel to the axes or forming an angle with the axes
- straight line segments and arcs tangent to the straight lines
- tangent arcs (provided they are still tangent to one another after they have been translated on a parallel path).

The u,v,w factors are only valid in the block in which they are programmed. They are used by the control only if they are associated to the corresponding coordinates:

- u for the 1st configured axis (typically X)
- v for the 2nd configured axis (typically Y)
- w to the 3rd configured axis (typically Z).

Examples of compensation factor applications u, v, w

Example1:

Program:

N5 T1.01 M6 S. . F. .
N6 G X Y30
N7 G1 Y10 v1
N8 X40 u-1
N9 Y30
N10 G X. . Y. .

Example 2:

Program:

N13 G X Y
N14 Y10 v-1
N15 X40 u1
N16 Y30
N17 G X. . Y. .

Example 3:

Program:

N13 G X Y
N14 G1 Z-10
N15 X-20 Y-20 u1 v1
N16 X20 u-1
N17 Y20 v-1
N18 X-20 u1
N19 Y-20 v1
N20 G X Y

FIG57.PIC

Example 4:

Program:

N13 G X-35 Y-35
N14 Z-10
N15 G1 X-20 Y-20 u-1 v-1
N16 X20 u1
N17 Y20 v1
N18 X-20 u-1
N19 Y-20 v-1
N20 GZ

Example 5:

Program:

N12 . . .
N13 G X-30 Y
N14 G1 Y20 v-1
N15 X30
N16 Y-20 v1
N17 X-30
N18 . . .

Example 6:

Program:

N12 G X40 Y B0
N13 G1 X30 u1
N14 X25 B360 u1
N15 G X40

MSA (UOV) - Defining a Machining Stock Allowance

The MSA command defines the value of the machining stock allowance along the profile. It is used in roughing and pre-finishing cycles. MSA can be programmed in a block, assigned in an MDI or entered via softkey.

Syntax

MSA = *value*

where:

value It can be a decimal number or an E parameter. It is programmed in the same measuring unit (G70-G71) that is currently active in the program.

Characteristic:

The system uses the programmed MSA to calculate the offset value to be applied to the profile when cutter diameter compensation (G41 or G42) is active.

The offset value is the sum of the tool radius and the MSA.

Example:

MSA	= 0.5	Assigns a 0.5 machining allowance
E30	= 1.5	
MSA	= E30	Assigns a 1.5 machining allowance

AUTOMATIC CONTOUR MILLING

The automatic contour milling commands (GTP and CCP) are designed for machining of profiles with radius compensation (G41/G42), generally defined using graphic tools.

The graphic tools (e.g. the Graphic Editor option) translate the drawing of the part to be machined into a technological program written in elementary ISO language.

One of the main characteristics of this feature is that the profiles for machining are seen as *subprograms*.

These profiles must be defined on a plane (e.g. XY) and can only contain the ***G*** operators for movement type (G1, G2, G3).

The contour milling commands will do the following:

- Automatically determine an approach point off the profile;
- Perform machining as described in the subprogram;
- Profile rotation, taking the first point of the profile as the rotation origin;
- Perform machining, starting from the last element of the subprogram through to the first (reverse machining).

In all cases, approach with compensation is scheduled on the first point of the profile (first block of program with abscissa/ordinate movements).

Limits to use of automatic contour miling

The limits regarding use of automatic contour milling are as follows:

- The profile must be described in full in a subroutine, indicated previously by *profile name*.
- The profile shall be considered closed if the first point described in the part program subroutine coincides with the last.
- Radius correction (*offset_mode*) will be activated automatically on the first block of the subroutine and de-activated on the last block.
- The blocks of the subroutine must be exclusively ISO type, programming of the following is allowed:
  - Axis names [dimensions]
  - *G* movement functions (*G1, G2, G3* with possibly their operands)
  - Three-letter commands in general shall NOT be taken into consideration (during the profile analysis stage), especially any commands that could modify the description of the profile (origins, mirror, scale factor).

NOTE:
Failure to respect these indications may lead to incorrect interpretation of the profile described in the subroutine.

GTP - Get Point

Determines the approach point off the profile.

Syntax

(GTP, profile_name, offset_mode, *E-par*, approach_type, angle)

where:

profile_name — An ASCII string representing the name of the part program that the profile is described in.

offset_mode — Possible values are 0, 1, 2.
Is the tool diameter offset mode to be used:
0 = no offset
1 = offset with tool on left of profile
2 = offset with tool on right of profile

E-par — The variable in which the coordinates of the approach point calculated by the system will be loaded:
index → abscissa
index+1 → ordinate

approach_type — Possible values are 0, 1, 2, 3.
Identifies the type of external approach to the profile; a different calculated point corresponds to each code (see next page).

angle — Angle of rotation:
Represents the value of the rotation angle with which the profile will be executed.

NOTES:

- The distance between the external approach point to the profile and the machining starting point is determed on the basis of the value in system variable MSA.
- Activation is possible of a feature that inverts direction of motion of the profile using system variable TPO (TPO=4 profile inversion)
  For further details of the TPO coding values see page 4-17.

WARNING

If you program offset_mode=0 , the approach_type is forced to “1” (approach tangential to profile) without any further indications from the system.

Determining the approach point

approach_type=0

Positioning occurs perpendicular to the first element of the profile.

The approach will be linear or circular, depending on value of the variable TPO.

Perpendicular approach to profile

approach_type=1

Positioning occurs tangential to the first element of the profile.

The approach is always linear.

Tangential approach to the profile

approach_type=2

Positioning occurs on the intersection between the first and last points of the profile; the approach will be linear or circular, depending on value of the variable TPO.

Approach on intersection of first and last point of the profile.

approach_type=3

Positioning ocurs on the bisector of the angle formed by the movements generated with *approach_type=0 and approach_type=1*; the approach will be linear or circular, depending on the value of the variable TPO.

Approach on bisector

CCP - Cutter Compensation Profile

Performs contour milling of a profile, with tool diameter compensation.

Syntax

(CCP, profile_name, offset_mode, exit_type, angle)

where:

profile_name	An ASCII string representing the name of the part program the profile is described in.
offset_mode	Possible values 0, 1, 2. Represents the offset mode with which machining is performed.
exit_type	Possible values 0, 1, 2, 3. Identifies the type of exit from the profile; a different calculated point corresponds to each code.
angle	Angle of rotation. Determines the value of the rotation angle the profile will be executed with.

Characteristics:

The point of exit off the profile is determined by considering a value, seen as the safety distance.

This value, associated with the exit point type, determines position on the plane of the exit point taken from system variable MSA.

NOTE:
Activation is possible of a feature that inverts direction of motion of the profile using system variable TPO (TPO=4 profile inversion active).

For further details of the TPO coding values see page 4-17.

END OF CHAPTER

PROGRAMMING THE SPINDLE

WARNING

The functions described in this chapter must be managed by the machine logic.

SPINDLE FUNCTIONS

With G96/G97 the S address programs the spindle speed in rpm or as constant surface feedrate. In addition, several M functions affect the spindle on/off state, range, etc.

G96 G97 - CSS and RPM Programming

The following G codes control spindle speed programming.

G96 Speed programming in constant surface speed

G97 Speed programming in rpm

Syntax

G97 [*G-codes*] [*operands*]

G96 [*G-codes*] [*operands*]

---

where:

G-codes Other G codes that are compatible with G96 and G97 (see Table "Compatible G codes" in Chapter 1).

operands Any operand or code that can be used in a G function block.

Characteristics:

G97 activates S programming in rpm. G97 is the default mode of the control and is modal with G96.

G96 activates constant surface speed (CSS) programming for the S word in feet per minute (G70), or metres per minute (G71). G96 forces the spindle speed to be controlled by the position of the diameter axis, so that it remains constant on the machining surface

When G96 is activated the position of the diameter axis is assumed to be the radius for which constant surface speed (S) is programmed. G96 is modal with G97.

Changing the S Word during G96. Cancelling CSS mode

If you want to change the S word value while G96 is active, the control must be in G00 or G29 mode. To cancel G96 CSS, the control must be in G00 mode and a block with G97 and an S word that defining the spindle rpm must be programmed.

Example:
The following example illustrates CSS programming.
Assumed power-up G codes: G00, G27, G70, G95, G97
Assumed gear range: 1 (M41) = 800 rpm max.

```
N1  G90 G00
N3  U5                  ;Presets diameter (U) axis
N4  SSL=700             ;700 rpm limit
N5  G96 S400 M3         ;CSS at 400 ft /min
N6  G1 U0 Z -2 F10      ;Sets contouring mode and U in feed 10 ipm
N7  U5 Z0
N8  G00                 ;Prepares to change S
N9  G01 S300 U0 Z-2     ;New surface feed = 300ft/min
N10 G00                 ;Prepares to cancel G96
N11 G97 S100            ;Set G97, S=100 rpm
N12 G01 Z0              ;Contouring to G0
N13 U5
N14 G00                 ;Rapid mode, contouring off
N15 M05                 ;Stop spindle
N16 M02                 ;Program end
```

IMPORTANT

S operators cannot be programmed when the process is on hold.

SSL - Spindle Speed Limit

The SSL command is used with G96 to set the maximum rpm that the spindle is allowed to run during CSS.

Syntax

SSL=*value*

where:

value Is a value that can be programmed directly with a decimal number or indirectly with an E parameter.

Example:

SSL = 2000	Assign a spindle speed limit of 2000 rpm
E32 = 1500	
SSL = E32	Assign a spindle speed limit of 1500 rpm

Characteristics:

As the diameter axis approaches the spindle centreline, the spindle approaches maximum speed to maintain the programmed S word value. The SSL command limits the spindle to some value below the maximum rpm. If you program the value of SSL above the maximum rpm for the current gear range, the current gear range limit will be the maximum allowable rpm.

The SSL command must be programmed before the number of rounds **S**.

IMPORTANT

Make sure you enter this value in the part program prior to entering G96 blocks.

M19 - Oriented Spindle Stop

Typically, the M19 function programs a spindle stop with a predetermined angular orientation. This feature is convenient in back spot-facing operations, because it allows to position the spindle, move the X or Y axis (depending on blade orientation), enter the hole, position the spindle again and start machining.

M19 can also be used for increased accuracy in boring operations, to avoid scoring the bored surface during the return move of Z axis. In this case, you would finish the hole, orient the spindle, move the X or Y axis according to the blade orientation, and perform the Z axis return.

M19 is deleted by M03, M04, M13, M14. When the control reads M19 in a block that also contains movement information, M19 precedes the movement.

IMPORTANT

The M functions that are implemented in your application may be different than those described here. Consult your machine documentation for more information on application-specific M functions.

Example:

Program:

N32 (DIS,"BACK SPOT- FACING BAR D=136")
N33 S115 F20 T7.7 M6
N34 G X250 Y-12 M19
N35 Z-306
N36 Y M3
N37 DWT=2
N38 G1 G29 G4 Z-300
N39 G Z-302 M5
N40 Y-12 M19
N41 Z

In block N34 the X and Y axes are positioned and spindle orientation is controlled so that the tool can go through the hole. In the next block, the Z axis is positioned to start spot-facing.

In block N36 the tool axis is moved to coincide with the spot-facing axis.
Spot-facing is performed in block N38. In subsequent blocks, the tool is oriented, withdrawn from the workpiece and then positioned along the Y axis, so that it can go through the hole.

END OF CHAPTER

MISCELLANEOUS FUNCTIONS

Standard M functions

In this chapter we briefly describe the miscellaneous functions (hereinafter M functions) that are considered standard for most applications. These functions are listed in the table below.
Since the machine integrator has programmed the interface between the control and your application, the M functions that are implemented in your machine may be different than those described here. Consult your machine documentation for more information on application-specific M functions.

M FUNCTION	ACTIVE PRE	ACTIVE POST	CANCELLED BY	MEANING
M0		x	Cycle Start	Program Stop
M1		x	Cycle Start	Optional Program Stop
M2		x		End of Program
M3	x		M4-M5-M14-M19	Spindle CW
M4	x		M3-M5-M13-M19	Spindle CCW
M5		x	M3-M4-M13-M14	Spindle Stop
M6		x		Tool Change
M7	x		M9	Auxiliary Coolant On
M8	x		M9	Main Coolant On
M9		x	M7-M8	Coolant Off
M10	x		M11	Axes Lock
M11	x			Axes Unlock
M12	x		M11	Rotary Axes Lock
M13	x		M4-M5-M14-M19	Spindle CW and Coolant On
M14	x		M3-M5-M13-M19	Spindle CCW and Coolant On
M19	x		M3-M4-M5-M13-M14	Spindle Stop and Angular Orientation
M30		x		End of Program and Reset to 1st Block
M40	x			Deactivate Spindle Range
M41	x		M42-M43-M44-M40	
M42	x		M41-M43-M44-M40	Spindle Range
M43	x		M41-M42-M44-M40	1-2-3-4
M44	x		M41-M42-M43-M40	
M45	x		M41-M42-M43-M44	Automatic Range Change
M60		x		Part Change

In the AMP environment you may declare that certain M functions can be disabled by a control reset.

IMPORTANT

The block that includes an expedite M code must also program an axis move.

Depending on how the M has been configured, the move may be point-to-point (G29) or continuous (G27-G28).

M CODE	DESCRIPTION
M0	**Program Stop**: M0 stops program execution, spindle rotation and coolant flow after the control has performed all the operations of the block in which it appears. The control retains all current status information after executing an M0.
M1	**Optional Program Stop**: If the command is enabled using the appropriate softkey, M1 operates like the M0 code.
M2	**End of Program**: M2 defines the end of a program.
M3	**Spindle Rotation Clockwise**: M3 defines spindle clockwise rotation. It is enabled by the control either when it is entered from the keyboard or when it is read from a program. In a block it is enabled before any other axis move.
M4	**Spindle Rotation Counter clockwise**: M4 defines spindle clockwise rotation. It is enabled by the control either when it is entered from the keyboard or when it is read from a program. In a block it is enabled before any other axis move.
M5	**Spindle Stop**: M5 stops spindle rotation. It is enabled by the control either when it is entered from the keyboard or when it is read from a program. In a block it is enabled after any axes moves.
M6	**Tool Change**: M6 temporarily stops the system from reading the program. It activates the offsets selected by the T function. In a block it is enabled before any other axis move. If it also stops spindle rotation and coolant flow, then M6 does not disable M3, M4, M7, M8, M13, M14. These functions become active again after M6 is completed.
M7	**Auxiliary Coolant On**: M7 turns the auxiliary coolant on. It is enabled by the control either when it is entered from the keyboard or when it is read from a program. In a block it is enabled before any other axis move
M8	**Main Coolant On**: M8 turns the main coolant on. It is enabled by the control either when it is entered from the keyboard or when it is read from a program. In a block it is enabled before any other axis move.
M9	**Coolant Off**: M9 turns all coolant systems off. It is enabled by the control either when it is entered from the keyboard or when it is read from a program. In a block it is enabled before any other axis move.

M functions (cont'd)

M CODE	DESCRIPTION
M10	**Linear and Rotary Axes Lock**: M10 locks the axes that are not involved in the current machining process.
M11	**Deactivates M10 and M12**
M12	**Rotary Axes Lock**: M12 locks only the rotary axes that are not involved in the current machining process.
M13	**Spindle Rotation Clockwise and Coolant On**
M14	**Spindle Rotation Counter clockwise and Coolant on**
M19	**Oriented Spindle Stop**: In a block, M19 spindle is oriented before any motion in the block. M19 deactivates M3, M4, M13, M14.
M30	**Automatic Reset at End of Program:** M30 deletes all information from the control dynamic buffer. Absolute origin 0 is automatically enabled and the selected program is set for restart. M30 does not deactivate the offset of the tool in the spindle.
M40	**Spindle Range Reset**
M41	**Spindle Range 1**
M42	**Spindle Range 2**
M43	**Spindle Range 3**
M44	**Spindle Range 4**
M45	**Automatic Spindle Range Change**
M60	**Workpiece Change**

END OF CHAPTER

PARAMETRIC PROGRAMMING

Parametric programming requires the programmer to use system and local variables.

System variables are stored in the dual port memory of the system. They are seen by all the active processes and are retained after the system is switched off.

Local variables are stored in a memory area local to the system and are seen only by the process they refer to. Their value is lost when the system is switched off. At power up they are re-initialised with the value defined in AMP.

The following table summarises the variables available with the system.

VARIABLE	TYPE	FUNCTION
E	local	Variables
!*name*	local	User variables
SN	system	System number
SC	system	System character
TIM	system	System timer (read only)
@*name*	system	PLUS variables

Except for System Characters, all system and local variables can be used in mathematical or trigonometric operations in place of the geometrical and technological data of the machining cycle.

A mathematical operation is formed by arithmetic operators, functions and operands (variables or numerical constants). The following are arithmetic operators:

- + addition
- - subtraction
- * multiplication
- / division

TRIGONOMETRIC FUNCTIONS

Trigonometric functions used by the system are listed in the table below.

FUNCTION	DESCRIPTION
SIN (A)	sine of A
COS (A)	cosine of A
TAN (A)	tangent of A
ARS (A)	arc sine of A
ARC (A)	arc cosine of A
ART (A)	arc tangent of A
SQR (A)	square root of A
ABS (A)	absolute value of A
INT (A)	integer portion of A
NEG (A)	negation of A
MOD (A,B)	remainder of A to B ratio

The arguments of a function (A,B) can be variables or numerical constants. When the control solves a mathematical equation, it considers the priority of brackets and signs. The result is converted into the format of the variable written to the left of the equal sign.

IMPORTANT

Arguments of trigonometric functions (SIN, COS, TAN) must be expressed in degrees. The result of inverse trigonometric functions (ARS, ARC, ART) is also expressed in degrees.

BOOLEAN FUNCTIONS

The boolean functions available with the system are listed in the table below.

FUNCTION	DESCRIPTION
AND(A,B)	Executes AND at the bit between two numbers in the -32768 +32767 range*
OR(A,B)	Executes OR at the bit between two numbers in the -32768 +32767 range*
NOT(A)	Complement to 1 of a number in the -32768 +32767 range*

IMPORTANT

* The parameters of boolean functions must be integers in the -32768 +32767 range (short format with sign). They may be long real E parameters as long as they remain in this range.

GTL FUNCTIONS

GTL functions available with the system are listed in the table below.

FUNCTION	DESCRIPTION
FEL(A,B)	Calculates the element having a B(1,2,3) index from the straight line whose index is A(1=sine of the angle, 2=cosine of the angle, 3=distance from origin to the straight line). **Example:** E30=FEL(5,1) assigns to E30 the sine of the angle formed by the l5 straight line and the abscissa.
FEP(A,B)	Calculates the element having a B(1,2) index from the point whose index is A(1=point abscissa, 2=point ordinate). **Example:** E34=FEP(4,2) assigns to E34 the ordinate of the p4 point.
FEC(A)	Calculates the element having a B(1,2,3) index from the circle whose index is A(1=center abscissa, 2=center ordinate, 3=circle radius). **Example:** E42=FEC(8,3) assigns to E42 the radius of the c8 circle.

LOCAL VARIABLES

E Parameters

The maximum number of E parameters must be defined during system configuration. In theory, there can be up to 8000 E parameters.

E parameters are of the Long Real type, which allows 15 digits in total, 12 maximum before the decimal point and 9 decimals. The system accepts several statements per block, the only restriction being block length. When in block-by-block mode, multiple statements will be executed as if they were in a single block. Two levels of parametric indexes are allowed. For example: E(E(E..)).

E parameters receive values in special assignment blocks. The format of an assignment block is:

E*n* **=** *expression*

where:

n	Is the identification number of the E parameter.
expression	Can be a numerical value, a character or a mathematical equation whose result is stored in the E parameter identified by *n*.

Examples:
The following are assignment blocks for parameter calculation.

E37=(E31*SIN(E30)+123.4567)/SQR(16)	Solves the mathematical equation and assigns the result to parameter E37.
E39=-0.00000124+5	Calculates the expression and assigns the result to parameter E39.
E40=TAN(35)	Finds the tangent of 35° and assigns the result to parameter E40.
E31=NEG(E31)	Changes the sign of parameter E31.
E7=81	Assigns the value 81 to parameter E7.
E25=E25+30	Adds 30 to the current value of E25 and assigns the result to E25.
E29=1,2,3,4,5	Assigns the value 1 to parameter E29, the value 2 to parameter E30, the value 3 to parameter E31, the value 4 to parameter E32 and the value 5 to parameter E33.

E parameters can be used inside programs and subroutines. To display the current value of an parameter, use the DIS command. For instance,

Example:
(DIS,E39) displays the current value of E39.

Example:
This example shows how to assign an ASCII character:

SCO="P" Assigns the P character to the SC0 string variable
E1=SC0
(DIS,E1) Displays 80 (ASCII code of P)

Examples of motion blocks or commands with parameters.

XE1
X-E1
X(E1)
X(-E1)
X(E8-14*SQR(E14))
X(-(E8-14*SQR(E14)))
X(E(E(E3)))

FE1
SE2
TE1.E2

! - User Variables

User defined variables can be of two types:

- Long Real
- Character

User variables must be defined in the AMP configuration. A user variable name can be up to 8 characters long. The first character must be !. The extension of a user variable may be .LR or .CH.

With user variables of the character type apply this rule:

!name_var[(*index*)].[*number of characters*]CH = Parameter

where:

index	Number indicating the starting character in the variable character array. If index is not specified, it is taken to be zero. If specified, it must be programmed between round brackets.
number of characters	Specifies how many characters after the index must be read/written. The default value is 1. The sum of index+number of characters must not be greater than the maximum number of characters configured for the specified variable.
parameter	Can be: – a string constant enclosed between apexes or quotes; – a string variable not longer than *length* – a numerical constant in the 0 - 255 range – a numerical variable in the 0 - 255 range.

Example:
!ABC(1) = 125
G0 X(!ABC(1))

125 is assigned to the !ABC(1) user variable, then this variable is used as argument of the X address in a G0 block.

!CHAR(2).8CH="ABC"

This instruction writes "ABC" in the first three characters of the !CHAR user variable, starting from the second character. The remaining 5 characters (8-3) will be automatically set to zero; to prevent this, program !CHAR(2).3CH="ABC"
!CHAR(1).CH="A" or!CHAR(1).CH=65

Long Real type user variables have the following format:

!name_var[(*index*)]= Expression

where:

index Number identifying the variable.
If *index* is not specified, it is taken to be zero.
If specified, it must be programmed between round brackets.

expression Can be a numerical value or a mathematical expression, whose result is stored in the user variable identified by the index.

Example:
!ABC(1) = 125
G0 X(!ABC(1))

125 is the value assigned to the !ABC(1) user variable; this variable is then used as an argument of address X in a G0 code.

NOTES:
The index of the variable can be a number or an E parameter.
In motion blocks, the user variable must always be written in brackets.

Examples of motion or command blocks with Long Real user variables:

X(!USER1(2))
X(!USER1(2)∗10)
F(!USER1(1))
S(!USER1(1))
T(!USER1(1).(!USER1(2))

NOTES:
The index of the variable may be a number or an E parameter.

SYSTEM VARIABLES

There are four types of system variables that can be used in part programs:

- System Number
- System Character
- System timers
- Plus variables

These variables can be used to read or write values or strings for assignment operations within part programs.

SN - System Number

System Number variables are of the Long Real type, which allows 15 signed digits with 12 integer digits maximum. Up to 25 System Numbers can be defined in the 200 byte area that is available in the dual port memory of the system. These variables do not allow index levels.

The format of a System Number variable is as follows:

SN*n***=** *expression*

---

where:

n	Is the identification number of the System Number variable. The n parameter can be a number or an E parameter.
expression	Can be a numerical value, an equation, or a character whose result is stored in the System Number identified by *n*.

NOTE:
You can assign one System Number variable to each defined System Number.

Examples:

SN20 = 326.957	the decimal value 326.957 is assigned to the SN20 variable.
SN20 = (SN9*SIN(30) + 12.5)/SQR(81)	the result of the mathematical expression is assigned to the SN20 variable.

Examples of motion or command blocks with SN variables:

X(SN0)
X(SN0·∗SN1)
F(SN1)
S(SN2)
T(SN1).(SN3)

SC - System Character

System Character variables are of the character type and are stored in the dual port memory of the system, where a 100 byte array is reserved for them. This means that all the defined System Characters cannot occupy more than 100 bytes.

Each System Character is identified by an index that specifies the start address inside the array and by a length that specifies how many bytes the variables occupies starting from that address.

The format of a System Character variable is as follows:

SC*index***.***length* **=** *parameter*

where:

index Is the index that specifies the start position of the variable in the array. It may be in the 1 to 100 range. The index of the variable can be a number or an E parameter.

length Is the length of the variable expressed in number of characters (bytes). The maximum allowed length for a single variable is 80 characters. The length can be a number or an E parameter.

parameter A parameter may be:

- a string constant enclosed between apexes or quotes;
- a string variable not longer than *length*
- a numerical constant in the 0 - 255 range
- a numerical variable in the 0 - 255 range.

NOTES:

- For each variable, the *index+length* sum must not exceed 100.
- One System Character variable can be assigned to another System Character variable.
- The SC variable may also be assigned the numeric variables (decimal) corresponding to ASCII characters. In this way it is also possible to assign those characters that are not displayed. For example, 10 (LF) and 13 (CR).
- Numerical values are programmed without double quotes (" ").

Example:
SC3.5="PIPPO"
SC9.3="ABC" or SC9.3=65,66,67

The string "PIPPO" is written starting from byte 3 of the array and occupies 5 bytes.

The string "ABC" is written starting from byte 9 of the array and occupies 3 bytes.

IMPORTANT

When defining the index and the length, care must be taken not to overlap two variables.

Example:

SC0.1=80	assigns 80 to the string variable
(DIS,SCO.1)	displays P (ASCII code 80)
E1=80	assigns 80 to E1
SC0.1E1	
(DIS,SC0.1)	displays P (ASCII code 80)

TIM - System Timer

The TIM instruction defines a variable used by the programmer to read the time shown by the control timer. Its value is expressed in seconds.

A TIM instruction can be read, displayed or stored in a support variable. Computation of this value starts as the control is switched on.

Example:
(DIS, TIM)
E10=TIM

NOTE:
TIM contents cannot be modified from part program.

@ - PLUS Variables

From part-program it is possible to read or write the PLUS variables managed by the machine logic.

Syntax

@name

where:

name Is the name of the variable declared in the configuration (AMP).

There are three types of PLUS variables:

- Short
- Boolean
- Double

There are 256 Short variables, but the part program can only access 128 of them. The names of the Short variables that can be accessed by a part program must be configured in AMP. They are made up of 16 bits and can contain values between -32768 and +32767.
You can also address a single bit of a Short variable with a Boolean variable. There are 256 Boolean variables, but a part program can only access 128 of them. The names of the Boolean that can be accessed by part programs must be configured in AMP.

There are 64 Double variables, all accessible by the part program. The names of the Double variables that can be accessed by a part program must be configured in AMP.
The total number of Boolean, Short and Double variables must be configured in AMP.

The values of all the PLUS variables configured in AMP are loaded when the system is started up. If the *value* field of a variable is empty, the variable is not initialised and retains the value it had when the system was switched off.
PLUS variables can be used in a part program as follows:

1. Assignment blocks and three-letter codes
   E10=@LOG1
   (GTO,END,@LOG3=1)
2. Motion or command blocks
   G0 X (@LOG2)
   X(@LOG2*2)
   F(@LOG2)
   S(@LOG3)
   T(@LOG2)·(@LOG3)

WARNING

Because these variables are directly linked to the machine logic, the machine tool manufacturer must provide you with the list of the variables used in your application as well as with all the information you need in order to handle them correctly.

L Variables

L variables have a long real format, with 15 signed figures and a maximum of 12 integer figures. There are 400 L variables indexed from 0 to 399.

These variables conform with the user tables available in the table editor and the PLUS environment. They can be utilised both in the programming and the logic environment, either separately or for communications between both environments.

Examples:
L10 = 26.9570

L15 = (L10*SIN (30)+9)/SQR(81)

(GTO,END,L2=1)

G0XL15

X(L15)

X(L15∗L1)

FL1

SL1

TL1.L3

WARNING

Because these variables are directly linked to the machine logic, the machine tool manufacturer must provide you with the list of the variables used in your application as well as with all the information you need in order to handle them correctly.

Multiple Assignments

With the multiple assignment operators **{ }** it is possible to assign the contents of other variables to a certain number of variables.

Multiple assignment is accepted only for numeric variables.

Syntax

destination_variable = **{** *source_variable, number_variables* **}**

where:

destination_variable	Is the first destination variable.
source_variable	Is the first of the source variables.
number_variables	Is the number of variables to transfer. This can be an integer or a local or system variable.

Examples:
E0 = **{** SNO,4 **}**

is equivalent to the following four assignments:

E0 = SN0
E1 = SN1
E2 = SN2
E3 = SN3

E100 = 5
E50 = **{** LO,E100 **}**

is equivalent to the following five assignments:

E50 = L0
E51 = L1
E52 = L2
E53 = L3
E54 = L4

END OF CHAPTER

CANNED CYCLES

CANNED CYCLES G8N

Codes G81 through G89 define canned cycles that let you program multiple operations (drilling, tapping, boring, etc.) without repeating parameters and commands that are common to all the operations.

In the block programming the G81-G89 canned cycle it is not possible to also program axes moves. The cycle is stored, but not executed.

Canned cycle execution starts from the block that follows the definition of the G81-G89 block. To repeat a cycle a second time you must simply program the coordinates of thet points at which the cycle must re-start.

The spindle axes for the canned cycle can be assigned in the G81-G89 block.

For example, in the G81 R Y-20 block the Y axis is the spindle of the canned cycle.

G8n functions are modal. Before programming a new canned cycle you must first cancel the current one with G80. The G80 function must be programmed in the block that follows the last canned cycle.

You cannot program a G8n block if cutter diameter compensation (G41/G42) is active.

When a canned cycle such as G82, G83, G89 requires a dwell time you can:

- use the default dwell time defined in AMP
- program a block containing the variable DWT=time (expressed in seconds)

IMPORTANT Canned cycles can also be performed on virtual axes.

Canned Cycle Features

This table lists canned cycles available with 10 Series and their features.

CANNED CYCLE	INFEED	DWELL	ROTATION	RETURN
G81 drilling	working	no	normal	rapid
G82 spot-facing	working	yes	normal	rapid
G83 deep drilling with chip take out	intermittent working (down at machining rate spaced with rapid retracts	no	normal	rapid
G84 tapping	working spindle rotation start	no	rotation reversal	working to R1 rapid to R2 if present
G85 reaming or tapping by Tapmatic	working	no	normal	working to R1 rapid to R2 if present
G86 boring	working spindle rotation starts	no	stop	rapid
G89 boring with spot facing	working	yes	normal	working to R1 rapid to R2 if present
G80 deletes the canned cycle				

Canned Cycle Moves

When the system reads a canned cycle programmed in a block the axes execute the following sequence of moves:

1. Rapid positioning to the centre line of the hole
2. Rapid approach to the work plane (R1 dimension)
3. Machining feedrate down to the programmed depth dimension (Z)
4. Cycle functions at the bottom of the hole
5. Rapid or machining feedrate return to the R dimension (R2 if the return dimension is different from the approach R1)

Examples of Canned Cycles

Two typical canned cycles are shown in this section.

The following is a canned cycle in which the approach coordinate is equal to the return coordinate.

G81 R_1 . . Z. .

R_1 = approach coordinate
Z = hole depth

In this canned cycle the return position is different from the approach position.

G81 R_1. . R_2. . Z. .

R_1 = approach coordinate
R_2 = return coordinate
Z = hole depth

G81 - Drilling Cycle

Syntax

G81[*G-codes*] *R1.. [R2..] Z.. [F..]* [*auxiliary*]

where:

G-codes Other G codes compatible with G81 (see "Compatible G codes" Table in Chapter 1).

R1 Approach dimension (mandatory). Defines the coordinates for rapid positioning on the machining plane when the cycle starts. The R address is followed by the rapid approach value. It can be programmed directly with a decimal number or indirectly with an E parameter.

R2 Defines the coordinates for return after machining (R planes). The R address is followed by the return value. If it is omitted, the control assumes by default the approach dimension (R). R2 can be programmed directly with a decimal number or indirectly with an E parameter.

Z Defines the hole depth (typically Z). The Z address is followed by the depth value, which can be programmed directly with a decimal number or indirectly with an E parameter.

F Defines the feedrate used in the canned cycle operation. It is programmed with the F address followed by the feedrate value.

auxiliary M, S or T programmable auxiliary functions. In a canned cycle block you can program up to four M functions, one S (spindle speed) and one T (tool selection).

Example:

NOTE:
0.Z means Z=0

Program:

```
    (UGS,X,0,110,Y,0,80)
    N31 (DIS,"TWIST DRILL D=6.5")
    N32 S1100 T3.03 M6
    N33 G81 R3 Z-15 F95 M3
1   N34 X15 Y15
2   N35 Y60
3   N36 X80
4   N37 Y15
    N38 G80 Z50 M5
    N39 M30
```

G82 - Spot Facing Cycle

Syntax

G82[*G-codes*] *R1.. [R2..] Z.. [F..]* [*auxiliary*]

where:

G-codes Other G codes compatible with G81 (see "Compatible G codes" Table in Chapter 1).

R1 Defines the value of hole start (see G81).

R2 Defines the coordinates for return (see G81).

Z Defines the value of hole end (see G81).

F Defines the value of feerdare(see G81).

auxiliary M, S or T programmable auxiliary functions.

Example:
The following figure shows a spot facing cycle.

G82 R5 Z-15 F.. M..
X.. Y..

G83 - Deep Drilling Cycle

Syntax

G83 [*G-codes*] *R1.. [R2..] Z.. I.. [J..] [K..] [F..]* [*auxiliary*]

where:

G-codes Other G codes compatible with G81 (see "Compatible G codes" Table in Chapter 1).

R1 Defines the value of hole start (see G81).

R2 Defines the coordinates for return (see G81).

Z Defines the value of hole end (see G81).

I Defines the depth increment after each pull-out in for wood pecking cycles. The I address is followed by the depth increment value.

J Defines the minimum depth increment after which the cycle applies a constant increment. The J address is followed by the minimum depth increment value.

K Defines the reduction factor applied to *I* until the *J* value is reached. The K address is followed by a numerical value.

R1 Defines the value of hole start (see G81).

R2 Defines the coordinates for return (see G81).

Z Defines the value of hole end (see G81).

F Defines the value of feerdare (see G81).

auxiliary M, S or T programmable auxiliary functions.

Characteristics:

With G83 different moves may be generated no matter whether **I, J** and **K** are programmed in the block.

If **I,J** and **K** are programmed axes moves are as follows:

1. Rapid approach to the hole centre line
2. Rapid approach to the **R** coordinates
3. Machining feedrate to the **R** + **I** coordinates
4. Rapid retract to the intial **R** coordinates each time a chip discharge occurs.
5. Calculation of the new **R** value with the formula:

   $R = R_{old} + I - X$ (where X is 1 by default, or, if the DRP variable is programmed, it assumes the value assigned to this variable)
6. Calculation of the new I value with the formulas:

   $I = I_{old} * K$ if $I * K > J$

   $I = J$ if $I * K < J$
7. Repetition of points 2 through 6 until the Z dimension is reached.

Chip breaking cycle

If J and K parameters are not programmed, the following movements are generated:

1. Rapid approach to the hole centre line
2. Rapid approach to the R coordinates
3. Machining rate to R + I
4. Spindle dwell for the time programmed in the DWT function, or for the characterized time if DWT is not programmed.
5. Repetition of points 3 and 4 until the Z dimension is reached.

Example:

NOTE:
0.Z means Z=0

Program:

```
   N65 (DIS,"TWIST DRILL D=6")
   N66 S930 F65 T6.6 M6
   N67 G83 R8 Z-55 I20 K.8 J6 M13
1  N68 X-15.81 Y-22.2
2  N69 X23
3  N70 X9 Y35.8
   N71 G80 Z50 M5
   N72 M30
```

DRP – G83 hole reworking distance

The DRP command defines in mm the hole reworking distance in G83 cycles (with IJK versors programmed).
This value can be initialised at 1 by the system. It can be assigned from the Keyboard or the Part Program.

Syntax:

DRP=value

where:

value can be programmed directly or indirectly with an E parameter.

Example:

DRP=0.01	hole reworked at 0.01 mm
DRP=5	hole reworked at 5 mm

WARNING

RESET restores the value of 1.

G84 - Tapping Cycle with no Transducer

This G84 code operates when a transducer is not mounted on the spindle.

Syntax

G84 [*G-codes*] *R1.. [R2..] Z.. [F..]*

where:

G-codes Other G codes compatible with G81 (see "Compatible G codes" Table in Chapter 1).

R1 Defines the rapid approach coordinates and the the fixed return to work.

R2 Defines the rapid return coordinates.

Z Defines the hole depth (typically Z). The Z address is followed by the depth value, which can be programmed directly with a decimal number or indirectly with an E parameter.

F Defines the feedrate used in the canned cycle operation. It is programmed with the F address followed by the feedrate value.

Characteristics:

The tool that is approaching the workpiece at rapid must stop at five times the tap pitch (for depths less than 3.O) or seven times the tap pitch (for depths greater than 3.O) from the workpiece.

The tapping feedrate must be calculated with the following formula:

$$F = S * p * 0.9$$

where:

S spindle rotation speed

p the tap pitch

0.9 the feedrate decrease factor to keep the tool holder spring compensator stretched

The final Z must be decreased by 10% of the actual tap working travel.

The final Z must be long enough for the axis to reach the programmed feedrate and stop with controlled deceleration. It must be calculated according to the time it takes spindle rotation to stop. If the final Z is not long enough, the control displays an error.

M functions are not allowed in the G84 block.

Example:

NOTE:
0.Z means Z=0

Program:
(UGS,X,-60,60,Y,-70,70)
N90 (DIS,"TAP M8-TRACTION TYPE COMPENSATOR")
N91 S280 F315 T8.8 M6 M13
N92 G84 R7 Z-15
N93 X-51.96 Y-30
N94 X51.96
N95 X Y60
N96 G80 Z50 M5

This program is valid for R.H. tapping operations, because M13 is programmed in block N91. For L.H. tapping operations, simply program M14 (or M04) instead of M13 (or M03).

G84 - Tapping Cycle with Transducer

This G84 code operates when a transducer is mounted on the spindle.

Syntax

G84 [*G-codes*] *R1.. [R2..] Z.. [K..] [F..] [auxiliary]*

where:

G-codes Other G codes compatible with G81 (see "Compatible G codes" Table in Chapter 1).

R1 Approach dimension (mandatory). Defines the coordinates for rapid positioning on the machining plane when the cycle starts. The R address is followed by the rapid approach value. It can be programmed directly with a decimal number or indirectly with an E parameter.

R1 Defines the rapid approach coordinates and the the fixed return to work.

R2 Defines the rapid return coordinates.

K Defines the thread tap pitch. The K address is followed by a value.

F Defines the feedrate used in the canned cycle operation. It is programmed with the F address followed by the feedrate value.

auxiliary M, S or T programmable auxiliary functions (see G81).

Characteristics:

With a transducer is mounted on the spindle, the G84 code can be programmed as follows:

- by calculating the F feedrate as if there were no spindle on the transducer.
- by using the K thread pitch. In this case the control automatically calculates the feedrate by multiplying K by the spindle speed in rpm.

WARNING

During the tapping cycle the control ignores the CYCLE STOP pushbutton (except in the rapid traverse approach section) and the FEEDRATE OVERRIDE selector (or softkey).
The SPINDLE SPEED OVERRIDE selector must be disabled by the machine logic. To abort the tapping cycle, the "INTP-ABO" logical function may be used (see variable description on Plus Variables Manual).

Example:
N90 (DIS,"TAP M8")
N91 S280 T8.8 M6 M3
N92 G84 R7 Z-15 K1
N93 X-51.96 Y-30
N94 X51.96
N95 X Y60
N96 G80 Z50 M5

G84 - Rigid tapping cycle with a transducer mounted on the spindle

Syntax

G84 [*G-codes*] **R1**.. [**R2**..] **Z**.. **K**..[*auxiliary*]

where:

G-codes	Other G codes compatible with canned cycle G84 (see "Compatible G codes" Table in Chapter 1).
R1	Defines the rapid approach and working return location points (mandatory).
R2	Defines the rapid return location points.
Z	Defines the end of tapping points (normally Z). It is given by the Z address followed by a value which can be programmed either directly with a decimal number or indirectly with an E parameter.
K	Defines the male threading pitch. It is given by the K address followed by a value.
auxiliary	Other M, S, T functions (see G81).

Characteristics:

To perform the tapping cycle correctly it is necessary to load two machine parameters with the appropriate values, which, as a rule, vary from machine to another
These parameters are called TKG and TAG.
Accordingly, rigid tapping cycles must be preceded by the definition of these two parameters with the values obtained at the machine installation stage.

We recommend creating a paramacro for the loading of these values (for instance, G840), to be recalled only once within the program that performs the rigid tapping operation.
Hence, the paramacro will be structured as follows:

TKG=……
TAG=……

If a reset command (which stops the program underway) is executed, the TKG and TAG parameters are deleted and will have to be assigned again for the correct execution of rigid tapping.

The methods for the determination of the correct values of the parameters are described in the AMP manual (Chapter 1)

TRP (RMS) - Tapping Return Percentage

The TRP command defines the feedrate percentage variation applied in the retract phase of the tapping cycle. This command is normally defined in a program, but can also be used in blocks entered with a keyboard command or by means of a softkey.

Syntax

TRP = *value*

where:

value can be programmed directly with an integer number or indirectly with an E parameter of the byte type.

Example:
TRP = 110 represents +10% of the programmed F
TRP = 10 represents - 90% of the programmed F

G85 - Reaming Cycle (or Tapping by Tapmatic)

Syntax

G85 [*G-codes*] *R1.. [R2..] Z.. [F..]* [*auxiliary*]

where:

G-codes Other G codes compatible with G81 (see "Compatible G codes" Table in Chapter 1).

R1 Defines the rapid approach coordinates and the the fixed return to work.

R2 Defines the rapid return coordinates.

Z Defines the hole depth (typically Z). The Z address is followed by the depth value, which can be programmed directly with a decimal number or indirectly with an E parameter.

F Defines the feedrate used in the canned cycle operation. It is programmed with the F address followed by the feedrate value.

auxiliary M, S or T programmable auxiliary functions (see G81).

G86 - Boring Cycle

Syntax

G86 [*G-codes] R1.. [R2..] Z.. [F..*] [*auxiliary*]

where:

G-codes Other G codes compatible with G81 (see "Compatible G codes" Table in Chapter 1).

R1 Defines the rapid approach coordinates and the the fixed return to work.

R2 Defines the rapid return coordinates.

Z Defines the hole depth (typically Z). The Z address is followed by the depth value, which can be programmed directly with a decimal number or indirectly with an E parameter.

F Defines the feedrate used in the canned cycle operation. It is programmed with the F address followed by the feedrate value.

auxiliary M, S or T programmable auxiliary functions (see G81).

G89 - Boring Cycle with Spot Facing

Syntax

G89 [*G-codes*] **R1**.. [**R2**..] **Z**.. [**F**..] [*auxiliary*]

where:

G-codes Other G codes compatible with G89 (see "Compatible G codes" Table in Chapter 1).

R1 Defines the rapid approach coordinates and the the fixed return to work.

R2 Defines the rapid return coordinates.

Z Defines the hole depth (typically Z). The Z address is followed by the depth value, which can be programmed directly with a decimal number or indirectly with an E parameter.

F Defines the feedrate used in the canned cycle operation. It is programmed with the F address followed by the feedrate value.

auxiliary M, S or T programmable auxiliary functions. In a canned cycle block you can program up to four M functions, one S (spindle speed) and one T (tool selection).

Using two R dimensions in a canned cycle

You can program two R's in a canned cycle when holes must be drilled, tapped, reamed, etc., on the same plane, but are separated by obstacles (clamps, holes on repeated subplanes, etc.)

Example:

NOTE:
0.Z means Z=0

Program:

N41 (UGS,X,0,100,Y,0,150)
N42 (DIS,"TWIST DRILL D=10")
N43 S850 F100 T4.4 M6
N44 G81 R-10 R2 Z-36 M3
N45 X35 Y40
N46 X65 Y80
N47 X35 Y120
N48 G80 Z50 M5

Updating Canned Cycle Dimensions

When a canned cycle is active in a program, you can program blocks with rapid approach, return and depth coordinates in order to update the cycle moves without re-programming the cycle.

Program blocks whose format is X, Y, R1, R2, Z are performed in this order:

1. X and Y
2. Updated R1 - new rapid approach coordinates
3. Updated Z - new depth
4. Updated R2 - new return coordinates

The table below summarizes the program block formats used for updating canned cycles:

FORMAT	ACTION
X.. Y.. Z..	Performs a canned cycle at XY with new Z depth
X.. Y.. R..	Performs a canned cycle at XY with new R1 rapid plane
X.. Y.. R.. R..	Performs a canned cycle at XY with new (R1) rapid plane and (R2) return dimension
X.. Y.. R.. Z..	Performs a canned cycle at XY with new R1 rapid plane, and Z depth
X.. Y.. R.. R.. Z..	Performs a canned cycle at XY with new R1 rapid plane, R2 return dimension, and new Z depth
R..	Updates the R1 rapid plane and does not the perform canned cycle at the current location
R.. R..	Updates R1 rapid plane and R2 return dimension; does not perform a canned cycle at the current location
R.. Z..	Updates R1 rapid plane and Z depth; does not perform a canned cycle at the current location
R.. R.. Z..	Updates R1 rapid plane, R2 return, and Z depth dimensions; does not perform a canned cycle at the current location

Updating R dimensions (upper limit and lower limit) during execution

Example 1:

NOTE:
0.Z means Z=0

Program:

```
    N35 (DIS,"TWIST DRILL D=8")
    N36 S1000 F100 T4.4 M6
    N37 G81 R3 Z-42 M3
1   N38 X15 Y15
2   N39 X65
3   N40 Y85 R-13
4   N41 X15
    N42 G80 Z50 M5
```

Example 2:

NOTE:
0.Z means Z=0

Program:

```
   (UGS,X,0,100,Y,0,200)
   N42 (DIS,"TWIST DRILL D=8")
   N43 S1000 F100 T5.5 M6
   N44 G81 R-18 Z-46 M13
1  N45 X25 Y25
2  N46 X75 R-18 R-8
3  N47 Y75 R-8 R2 Z-25
4  N48 Y175 R-3 Z-46
5  N49 X110
   N50 G80 Z50 M5
```

Example 3:

NOTE:
0.Z means Z=0

Program:

```
   (UGS,X,0,100,Y,0,200)
   N42 (DIS,"TWIST DRILL D=8")
   N43 S1000 F100 T4.4 M6
   N44 G81 R-18 Z-42 M3 M8
1  N45 X25 Y25
2  N46 X75 R-18 R4
3  N47 Y125 R-14 R4
4  N48 Y225 R-18
5  N49 X110
6  N50 G80 Z2 M5
```

END OF CHAPTER

PARAMACRO

Paramacro Definition

Paramacro subroutines can be used in user-defined cycles. They are called with a 3-digit code.

Modal paramacros are active only in motion blocks that do not include M functions. No other function or code can be programmed when a modal paramacro is active.

Syntax

G*n* *par-name-1* [*value-1*] . . . [*par-name-n*] [*value-n*] . . . [*"string"*]

---

where:

n	Is a number from 300 to 998
par-name-1. . . par-name-n	Are letters (refer to the correspondence table)
value-1 . . . value-n	Can be a number or an E parameter or a parametric expression.
string	Character string (max. 99)

Characteristics:

There are two groups of paramacros:

- from G300 to G699 non modal paramacros
- from G700 to G998 modal paramacros

Modal paramacros are reset by G999.

The H, HF and HC parameters are used in the paramacros, The system offers 100 H parameters. Some of them can be used in paramacros. H parameters are classed as follows:

- Parameters from H0 to H51 are associated to letters and cannot be used in paramacros.
- Parameters from H52 to H99 can be used for math operations in paramacros.

When H parameters are used in paramacros, all the operations can be calculated with E parameters.

Paramacros allow 4 nesting levels.

When a series of paramacros are nested, the H, HF and HC parameters in a nested paramacro will reset the equivalent H, HF and HC parameters in upper paramacro levels.

Example:
G300 A1 B2 C3 D4

H0 is assigned to 1
H1 is assigned to 2
H2 is assigned to 3
H3 is assigned to 4

HF0 is set
HF1 is set
HF2 is set
HF3 is set

All other HF are reset.

If G300 calls G400 A10 C30, then the value of H0 is 10 and the value of H2 is 30. HF0 and HF2 will be set and all other HF's will be reset.

The values assigned to H0 and H2 in G300 are now lost.

All the other parameter types available with 10 Series can theoretically be used in paramacros. However, to avoid interactions with E parameters used elsewhere, it is preferable to use H parameters where possible.

HC Parameters

If a block calling a paramacro also includes specification of a character string between inverted commas, this string is made available for the paramacro in the HC character array (100 characters).

If no such string is specified, the entire HC array is reset.
If the string was programmed, it is put in the HC variables with a terminator '0' added; for this reason, maximum length of the string is 99, even if the HC array is 100 characters long.

Examples:

```
!PROFILE=.............
G600 "PROFILE" A50 ;call to paramacro G600
. .
. .

;------------------ in the paramacro.............
(DIS,HC0.10)
(CLS,?HC0.10)
```

Situation of the HC parameters after the call to the paramacro:

HC0="P" HC1="R" HC2="O" HC3="F" HC4="I" HC5="L" HC6="E" HC7=0

HO=50 HFO=1

The following table shows the correspondence between letters and H parameters.

LETTER	PARAMETERS H	HF	LETTER	PARAMETERS H	HF
A	H0	HF0	a	H26	HF26
B	H1	HF1	b	H27	HF27
C	H2	HF2	c	H28	HF28
D	H3	HF3	d	H29	HF29
E	H4	HF4	e	H30	HF30
F	H5	HF5	f	H31	HF31
G	H6	HF6	g	H32	HF32
H	H7	HF7	h	H33	HF33
I	H8	HF8	i	H34	HF34
J	H9	HF9	j	H35	HF35
K	H10	HF10	k	H36	HF36
L	H11	HF11	l	H37	HF37
M	H12	HF12	m	H38	HF38
N	H13	HF13	n	H39	HF39
O	H14	HF14	o	H40	HF40
P	H15	HF15	p	H41	HF41
Q	H16	HF16	q	H42	HF42
R	H17	HF17	r	H43	HF43
S	H18	HF18	s	H44	HF44
T	H19	HF19	t	H45	HF45
U	H20	HF20	u	H46	HF46
V	H21	HF21	v	H47	HF47
W	H22	HF22	w	H48	HF48
X	H23	HF23	x	H49	HF49
Y	H24	HF24	y	H50	HF50
Z	H25	HF25	z	H51	HF51

Example 1:
N45 G777 A(E8) R22.5 F(E2) S(E3+5-E1)

E8 is passed to H0
22.5 is passed to H17
E2 is passed to H5

The result of (E3+5-E1) is passed to H18

In this example Boolean parameters HF0, HF17 and HF5 are set to 1.

Example 2:

D=H3=Major diameter
R=H17=Minor radius
F=H5=Feedrate

Program:

This is an example of milling/boring cycle using paramacros.

```
;MAIN PART PROGRAM
    .
    .
    .
N20  G601  D125  R25  F160
N21 . . .
```

```
; PARAMACRO G601
G0 X90 Y80
G92 XY
(GTO,END,HF3=0)
(GTO,END,HF17=0)
(GTO,END,HF5=0)
H57=H3/2
H58=H57-H17
G1 G41 XH17 YH58 F2000
G3 X0 YH57 I0 JH58 FH5
I0 J0
H59=NEG(H17)
G40 XH59 YH58 I0 JH58
G1 X0 Y0 F2000
(GTO,F)
"END"
G99
(DIS,"OMITTED PARAMETERS")
M ..
"F"
```

DAN - Define Axis Name

This command associates the name of a characterised axis to the name used in a paramacro.

Syntax

(DAN,*par-ax1 char-ax1*[**,***par-ax6 char-ax6*]**)**

(DAN)

where:

char-ax1 **...** *char-ax6*	Are the names of up to six characterised axes.
par-ax1 ... par-ax6	Are the names of six axes used in a paramacro.
no parameters	(DAN) without parameters disables (DAN) mode.
Example 1: (DAN,PX,QY,DZ)	X,Y,Z are replaced by P,Q,D so X,Y,Z are not usable.
Example 2: (DAN,PX,QY,DZ)	X,Y,Z are replaced by P,Q,D so X,Y,Z are not usable.
(DAN,WA)	A is replaced by W, furthermore X,Y,Z are re-enabled and P,Q,D,A are not now usable. By reprogramming (DAN,...) all previous associations are cancelled and the current ones become active.

IMPORTANT

After the three-letter mnemonic DAN has been used, if the axes concerned are those of the interpolation plane, this plane must be redefined using the new names.

PMS – S as a paramacro

PMT – T as a paramacro

PMM – M as a paramacro

These commands enable and disable the management of the relative ancillary functions S, T, M as paramacros.
For further information on how to program these paramacros see the paragraph on the “S, T, M ancillary functions executed as paramacros” in this chapter.

Syntax

PMS=value
PMT=value
PMM=value

---

where:

value This value can be:

0 Disables the management of the S, T, M functions as paramacros – these functions are executed in standard mode

1 Enables the management of the S, T, M functions as paramacros – these functions associate the execution of the corresponding paramacro

NOTE:

The values ascribed to the PMS, PMT and PMM variables can be changed as follows:

- Through AMP at the configuration stage
- Through a part program with the syntax described

The characterisation values are restored with a reset.

S, T, M ANCILLARY FUNCTIONS EXECUTED AS PARAMACROS

The execution of the S T M ancillary functions can be associated with paramacro type subprograms, i.e. in the presence of a block containing at least one S, T and/or M function, previously configured as a paramacro, the corresponding subprogram is executed, according to the criteria currently defined for paramacros.

Syntax:

same as the syntax used for ancillary functions executed in standard mode.

ANCILLARY CODE	PARAMACRO
S	In the presence of an “Sx” function, e.g. on process “n”, paramacro “P0nS” is executed with parameter H18 = x
T	In the presence of a “Tx” function, e.g. on process “n”, paramacro “P0nT” is executed with parameter H19 = x
M	In the presence of an “Mx” function, e.g. on process “n”, paramacro “P0nM00x” is executed with parameter H12 = x

Characteristics:

- The functions generating paramacro calls need not be programmed at the start of the block.
- If M, S and/or T are programmed simultaneously, the name of the procedure to be called is determined by the first function which is configured as a paramacro, and any other functions programmed in the block are called with the respective parameters H, HF and HC, as described in the programming manual (chapter 9). For example, if both M and T are configured as paramacros and you program TxMy, this will generate a call to macro P0nT whilst the programming of MyTx will call the "P0nM00x" macro, with parameters H19=x and H12=y in either case.
- For the execution of the paramacro in MDI mode, see paragraph “Executing blocks entered from the keyboard” in the user manual.
- At the end of the paramacro, the execution mode (MDI, AUTO, STEP) in which the system was working before the call is restored.
- The type of paramacro being executed can be read in the **SW21** variable and it is:
  - 1000 for a paramacro called by the S function
  - 2000 for a paramacro called by the T function
  - 3xxx for a paramacro called by the Mxxx function

- The *pathname* to be used to call the paramacros can also be specified with the PTH triliterals described below in this chapter. If the pathname is omitted, the search for the directory containing the subroutine is performed as follows:

  1. case: no pathname has been specified through a PTH instruction.
  The system searches for the paramacro in the directory containing the calling program and, if it does not find it, searches for it in the DOS directories to which logic names have been associated during the machine characterisation process.

  2. case: a pathname has been specified through a PTH instruction.
  The system searches for the paramacro in the directory specified with a PTH and, if it does not find it, searches for it in the DOS directories to which logic names have been associated.

Configuration: three process variables (PMM, PMS, PMT, defined previously), if set on 1/0, enable/disable the execution of the corresponding functions as paramacros at a general level.
Moreover, the execution of ancillary function M as a paramacro is identified by an ad hoc bit, defined in AMP.

Hence, if PMM = 1 and the “Mx” function is:

- configured in the “n” process but its bit = 0, the paramacro is not executed (standard Mx is executed instead)
- configured in the “n” process and its bit = 1, the “P0nM00x” paramacro is executed
- not configured in the “n” process, an “M not defined” error message is displayed

Example:

Execution of the block described below in process 1, in which the auxiliary function M2 is configured with the paramacro bit enabled

N20 X10 M2 T1.0 S2000

PMM	PMT	PMS	Paramacro name	Parameters
1	1 or 0	1 or 0	P01M002	H23 = 10; H12 = 2; H19 = 1.0 H18 = 2000
0	1	1 or 0	P01T	H23 = 10; H12 = 2; H19 = 1.0 H18 = 2000
0	0	1	P01S	H23 = 10; H12 = 2; H19 = 1.0 H18 = 2000

This is an example of how the relative paramacros are developed:

```
;PARAMACRO P01M002
E0 = H12
PMM = 0
ME0
"TEST_T"
(GTO,TEST_S,HF19=0)
E1 = H19
PMT = 0
TE1
"TEST_S"
(GTO,END,HF18=0)
E2 = H18
PMS = 0
SE2
"END"
```

```
;PARAMACRO P01T
E1 = H19
PMT = 0
TE1
"TEST_M"
(GTO,TEST_S,HF12=0)
E0 = H12
PMM = 0
ME0
"TEST_S"
(GTO,END,HF18=0)
E2 = H18
PMS = 0
SE2
"END"
```

```
;PARAMACRO P01S
E2 = H18
PMS = 0
SE2
"TEST_M"
(GTO,TEST_T,HF12=0)
E0 = H12
PMM = 0
ME0
"TEST_T"
(GTO,END,HF19=0)
E1 = H19
PMT = 0
TE1
"END"
```

END OF CHAPTER

PROBING CYCLES

MANAGING AN ELECTRONIC PROBE

An electronic probe is a measuring device that can be mounted on the spindle and controlled like a tool. The probe can also have length and diameter offsets associated with it. Besides, it can be mounted in a fixed position and used as an electronic gauge to requalify the tool length.
The figure that follows shows an electronic probe with its relevant dimensions.

Electronic Probe

Some G codes perform specific measuring or probing cycles when an electronic probe is mounted on the spindle.

G CODE	FUNCTION
G72	Measures the co-ordinates of a point
G73	Measures the co-ordinates of the circle center and radius
G74	Measures variances from theoretical points (for probes mounted in a fixed position rather than on the spindle).

The control stores measured values in E parameters that you define in the probing cycles (G72-G74).
When the control executes a probing cycle, it measures a point through the following sequence of moves:

1. Rapid to the approach point (Pa).
2. Move at measuring speed (Vm) to the point where the probe triggers, then stop and store the dimensions. If the probe does not trigger, move only as far as the end safety point (Ps).
3. Rapid return to the start position of the probing cycle (hole center in G73). The execution of that phase depends on an (AMP) machine characterisation parameter or the corresponding DPP three-letter block mode parameter.

The diagram below shows moves and feed of a probing cycle.

The following errors may occur during a probing cycle:

- If the probe does not trigger before reaching the safety point, it returns to the start point of the cycle. The control panel displays a message and the system goes into an error condition.
- If the probe does not reset properly on the way back to the start point after a successful probing cycle, a message is displayed and the system goes on error condition.
- If the probe is carried out during the rapid approach phase, it returns to the start point of the cycle. The control panel displays a message and the system goes into an error condition.

NOTE:
If the probing cycle is executed with G27 or G28 active, both the approaching movement and probing are carried out in continuous mode, while the stopping movement obtained after the probe input has been found is executed in point to point mode.

WARNING

Values obtained from probing cycles are always expressed in the characterised unit (AMP). Therefore they are not affected by G70/G71 functions.

PRESETTING A PROBING CYCLE

The first time you use a probe, or whenever probing cycle conditions change, you must:

1. define the probing parameters
2. dynamically measure the diameter of the probe ball
3. requalify the probe with respect to the spindle axis
4. dynamically measure the length of the probe.

DPP (DPT) - Defining Probing Parameters

Use the DPP command to define the probing parameters from the keyboard or in a program. These parameters can also be defined on the control with a specific data entry via softkey (MACHINE SET-UP and PROBE SET-UP soft-keys). When these are not defined the system uses the values defined in AMP.

Syntax

(DPP,*approach***,***safety***,***speed*[*, mode*])

where:

approach Is the approach tolerance expressed in mm or inches.

safety Is the safety tolerance expressed in mm or inches.

speed Is the measuring speed expressed in mm/min or inches/min.

mode Is the value which indicates whether to perform the rapid return phase at the end of the probe. It can be one of the following values :

0 = perform the return phase

1 = do not perform the return phase

If no parameter is present then the default 0 is assumed.

Example:
Instruction used from the program
(DPP,10,12,1000)

WARNING

"Approach", "speed" and "safety" values are displayed in the unit of measurement that is active when the display is requested (G70 or G71).

Dynamic Measurement of the Ball Diameter

To dynamically measure the apparent diameter of the probe ball, use a requalification ring, cylinder, or similar device with a center assumed to be the absolute origin 9 of axes X and Y.

Probe Requalification

Typically, before using the probe, the center of the probe ball is not centered on the spindle axis. To center the probe ball on the spindle axis, you must requalify the probe.
Use the hole of the requalifying ring, cylinder, or similar device for this operation. The center of the hole is assumed to be origin 9 for axes X and Y.

Dynamic Measurement of the Probe Length

To measure the length of the probe dynamically, use the requalification ring, cylinder, or similar device whose reference surface (top) is assumed to be origin 9 for the length axis Z.

Probe Presetting

The following steps must be executed to preset the probe:

1. Use a requalification ring for reference.
2. Mount the requalification ring on the B rotary axis with its front surface at absolute origin 9 for the B rotary axis.
3. Position absolute origin 9 for axes X and Y at the center of the ring hole.
4. Position absolute origin 9 for the Z axis on the top surface of the ring.

The initial values stored in the probe offset are:

length1 = nominal length of the probe with respect to the axis of the ball.

Diameter = 0

Example:

```
N1  (DIS, "DPP, UPA, UPO, ")
N2 T30.30 M6        - probe on spindle
N3  (UAO, 9)
N4  (DPP, 10, 12, 600)  - defines probing parameters
N5  UPA=0
N6  UPO=0
N7  E30=...         - assigns diameter to the hole of the sample ring
N8  E31=E30/2
N9  E32=...         - assigns distance from Z=0 to probing surface on the  Z axis (usually = 0)
N10  E33=E31+10
N11  GB0            - only if the ring is mounted on the indexing table
N12  XY
N13  Z-4
N14  G73 rE31 E40       - measures hole co-ordinates (center and radius)
N15  Z100
N16  (DIS, "UPA=", E40, "UPO=", E41)
N17  M0
N18  UPA=E40      - requalifies probe abscissa
N19  UPO=E41      - requalifies probe ordinate
N20  E34=(E30-E42*2)            - diameter of apparent ball
N21  (DIS, "DIAMETER=", E34)
N22  M0
N23  (RQP, 30, 30, dE34)         - stored ball diameter on d offset
N24  T30.30 M6     - enables new offset
N25  GXYE33
N26  G72 ZE32 E43      - measures Z dimension on ring surface
N27  E35=E43-E32       - variance between nominal and real value
N28  Z100
N29  (DIS "VARIANCE.Z=", E35)
N30  M0
N31  (RQP, 30, 30, LE35)         - requalifies Z length offset
N32  M30
```

PROBING CYCLES

These are G codes that define probing cycles:

G72 Point measurement (with probe ball diameter compensation)

G73 Measurement of hole parameters

G74 Point measurement (without probe ball diameter compensation)

In the cycles G72, G73 and G74, E parameters are coupled with the axes in the order in which the latter are characterised (AMP) and displayed, and not in the order in which they are programmed.

With probing on rotated plane virtual axes, the E parameters are matched to the virtual axes in the order in which they have been defined in the three-letter code UPR.

G72 - Point Measurement with Compensation

G72 lets you use linear moves and a probe to measure the co-ordinates of a point.

Syntax

G72 *axis1* [*axis2*] [*axis3*] *E-par*

where:

axis1 . . . *axis3* Are the axes that will move during the probing cycle. The dimensions programmed in the G72 block are the theoretical positions where the probe is expected to impact the target point. You can program up to three axes and dimensions in the G72 block.

E-par Defines the E(x) parameter for storing the measured dimension of the first axis, in order of configuration, programmed in G72; the dimensions of other axes are stored sequentially in order of configuration in E(x+1) , E(x+2) and E (x+3) in virtual axes probing.

Characteristics:

The co-ordinates measured by the probe are stored in E parameters that are defined by the G72 cycle. The control stores the co-ordinates of the axes beginning with the E parameter specified in the cycle. Measures are taken applying cutter diameter compensation to the probe. To ensure high accuracy the surface must be perpendicular to the measuring move.

The probe can also be performed on virtual axes activated by the UPR function or on the axis which identifies the direction of the activated tool (TCP, 5).

When the probe is performed on the virtual axes defined with UPR, the values measured in relation to the virtual axes are stored in E (x), E (x+1), E (x+2).

When the probe is performed on the tool directional axes, the system stores the value of dimensions measured in relation to those axes in E (x) and the values measured, in relation to the part of the Cartesian linear axes defined in the TCP table or corresponding virtual axes if UPR virtual mode is active in E (x+1), E (x+2), E (x+3).

Example:
The axes are configured in AMP in the following order: XYZ

G72 X100 Y50 E32
G72 Y50 X100 E32

In both cases the values that the control calculates for X and Y are stored sequentially in E32 and E33.

Example:
Probing on plane rotated with virtual axes.

(UPR,0,XYZ,UWV,30,25,0)
G72 U25 V-10 W-25 E30

The dimensions of the system of three virtual axes are stored in E30, E31 and E32 according to the order defined in the UPR command or U in E30, W in 31 and V in 32.

G73 - Hole Probing Cycle

G73 lets you use a probe to measure the dimensions of a hole on the active interpolation plane.

Syntax

G73 *r.. E-par*

where:

r Defines the theoretical hole radius. r is followed by the radius length.

E-par Is the first E parameter in which the system starts storing values:

first E parameter: center abscissa (typically X)

second E parameter: center ordinate (typically Y)

third E parameter: radius

Characteristics:

If you program only one E parameter, the three values detected during the probing operation (circle center co-ordinates and radius) are stored in E parameters sequentially starting from the specified E parameter.

The probe can also be performed on virtual axes activated by the UPR function. In this case the values measured for the centre in relation to the virtual plane are stored in E (x) and E (x+1) and the radius is stored in E (x+2).

Before activating the G73 cycle, the axes of the machine tool must be positioned on the hole center. Measures are taken applying cutter diameter compensation to the probe.

Probing moves

Example:
G73 r100 E55 probing moves

The co-ordinates of the circle center (abscissa and ordinate) and the actual radius are stored in E55, E56, and E57 respectively.

G74 - Tool Requalification Cycle

G74 is a tool requalification cycle. G74 uses a fixed probe (such as an electronic gauge) to measure variances from theoretical points with the tool mounted on the spindle.

Syntax

G74 *axis1* [*axis2*] [*axis3*] *E-par*

where:

axis1. . . *axis3* Are up to three simultaneous axes.

E-par Defines the E(x) parameter for storing the deviation from the point measured and the point programmed for the first axis programmed in G74. The deviation values of the other axes, if programmed, are stored sequentially in E(x+1) and E(x+2).

Characteristics:

G74 can be used for requalifying a tool or checking tool wear. The calculation of measured dimensions does not take into account the tool offset, since the cycle is checking the actual "tool" dimension.

The steps in the G74 cycle are similar to those in the G72 cycle. The difference between both cycles is how the control executes calculations based on measured dimensions. The control does not consider the diameter of the probe ball and stores the variance from theoretical dimensions in the parameters specified in the G74 block.

The probe can also be performed on virtual axes activated by the UPR function or on the axis which identifies the direction of the activated tool (TCP, 5).

When the probe is performed on the virtual axes defined with UPR, the deviation values between the points measured and the programmed points in relation to the virtual axes are stored in E (x), E (x+1) and E (x+2).

When the probe is performed on the tool directional axes, the system stores the deviation value between the point measured and the programmed point in relation to that axis in E (x) and the values measured, in relation to the part of the Cartesian linear axes defined in the TCP table or corresponding virtual axes if the UPR virtual mode is active in E (x+1), E (x+2) and E (x+3) .

Example:
G74 X60 E41

E41 is given by the formula: E41 = Pm - Pt

where: Pm is the measured point
Pt is the theoretical point

UPA (RTA) - Update Probe Abscissa

Defines the probe requalification value for the abscissa (typically X).

Syntax

UPA=*value*

where:

value is the abscissa requalification value expressed in millimetres.

UPO (RTO) - Update Probe Ordinate

Defines the probe requalification value for the ordinate (typically Y).

Syntax

UPO=*value*

where:

value is the ordinate requalification value expressed in millimetres.

ERR - Managing Probing Errors

Cycles G72, G73 and G74 enable you to handle process errors either automatically or from part program. You can choose the method by setting the ERR parameter.

For further information on probing error management see Appendix C

OPERATIONS WITH A NON-FIXED PROBE

With probing cycles G72 and G73 you can:

- Requalify origins by:
  - probing reference surfaces
  - centring on a hole
- Check the dimensions of:
  - diameters
  - planes and depth of holes.

Requalifying Origins by Probing Reference Surfaces

Origins may change due to:

- Changes in temperature (thermal drift)
- New pallet used for machining

These changes may require that you to requalify the system origins by probing a reference surface.

Thermal drift

This procedure uses a requalifying cube that is placed at a precise location on the machine.

Main Program:

```
    .
    .
    .
N99 E33=-300
    .
    .
    .
/N100 (CLS,TEST3)
```

Subprogram TEST3:

```
N500 (DIS,"RQO-DT")
N501 G72 Y(E33) E32          ;Measured distance stored in E32
N502 E32=E32-E33
N503 (RQO,1,Y(E32))          ;Requalify origin 1 for Y axis
N504 (RQO,2,Y(E32))          ;Requalify origin 2 for Y axis
N505 (RQO,3,Y(E32))          ;Requalify origin 3 for Y axis
```

New pallet for machining

Main Program:

```
M . . . ;                    ;Pallet Change
N199 T30.30M6;               ;Probe on spindle
N200 (UAO,2)
N201 GXY
N202 E10=2
N203 E34=-250
/N204 (CLS,TEST4)
```

TEST4 subroutine:

```
N500 G72 Y(E34) E30
N501 E31=E30-E34
N502 (RQO,E10,Y(E31))        ;Requalify origin 2 for Y axis
```

Requalifying Origins by Centring on a Hole

Before you perform this requalification cycle, you must program the X and Y axes to position on the hole axis and program the probe to position into the hole.

Example:

```
N200 (DIS,"REQUALIFYING ORIGIN FOR X AND Y AXES")
N201 T11.11 M6
N202 GX180 Y60
N203 Z-130
N204 G73 r50 E35              ;Measuring cycle diameter 100. X and Y coordinates are
N205 E35=E35-180              ;stored in parameters E35 and E36
N206 E36=E36-60
N207 (RQO,1,X(E35),Y(E36))    ;Requalifies origin 1 for X (E35) and Y (E36)
```

Checking Diameters

You can use a probe to check the hole diameters. With proper programming you can compare detected values with allowed values, and decide whether to continue machining or branch to a block containing a label.

In the following program, the control measures the deviation between the actual and the theoretical diameter of a hole and compares this deviation with the tolerances. The comparison may have three different results and courses of actions:

- Hole diameter is within the allowed tolerance - the program continues.
- Hole diameter is greater than the allowed tolerances - tool is automatically timed out of use (label A3). The program stops (M00) and the part is rejected.
- Hole diameter is less than the allowed tolerances - tool is automatically timed out of use (label A4). The program branches to label A1, where the hole reaming cycle is repeated using the alternative tool that is specified in the tool life file.

Example:
The following is an example of diameter checking.
Hole diameter and tolerance: D=100 +0.02/-0.015

Program:

```
"A1" N111
N112 GZ-150
N113 (DIS,"REAMING HOLE D=100")
N114 F..S..T13.13 M6
N115 GX-120 Y80 M13
    .
    .
    .
N129 (DIS,"HOLE TOLERANCE CHECK D=100")
N130 T14.14 M6                                  ;probe in spindle
N131 GX-120 Y80                                 ;XY positioning to hole center
N132 Z-85                                       ;Z positioning
N133 G73 r50 E30                                ;hole radius stored in E32
N134 E32=E32*2
N135 (DIS,E32)
N136 (GTO,A3,E32>100.02)
N137 (GTO,A4,E32<99.985)
N138 GZ150
N139 (DIS,"WORKPIECE IN TOLERANCE")
    .
    .
    .
N2100 M30
"A3" N2101 (DIS, "Hole too big")
N2102 M00
"A4" N2103 (DIS, "Hole too small")
/N2104 (GTO,A1)
```

Checking Plane Dimensions and Hole Depths

You can use a probe to check the dimensions of planes and the depths of holes. With proper programming you can compare detected values with allowed values, and decide whether to continue machining or branch to a block containing a label.

In the following programming example, the control detects the deviation of actual dimensions from programmed dimensions. It compares this deviation with allowed tolerances. The comparison has three possible results:

- Continue machining
- Stop machining and reject the part because the dimension is too long
- Requalify the tool and repeat the program from label C1 because the dimension is too short

Example:
The following example check of the dimensions of a plane
Theoretical dimension, with respect to Z axis zero: - 80 ± 0.1

Program:

```
    .
    .
    .
"C1" N252
N253 (DIS," . . .")
N254 F. . S. . T23.23 M6
    .
    .
    .
N267 (DIS,"CHECK PLANE DIMENSION 80")
N268  T30.30 M6                    ;Probe on spindle
N269 GX150 Y35                     ;XY positioning on the point to be checked
N270 G72 Z-80 E30                  ;Distance measured and stored in E30
N271 (GTO,C2,E30 <-80.1)           ;Branch to C2 if E30 < min coordinate
N272 (GTO,C3,E30>-79.9)            ;Branch to C3 if E30 > max coordinate
N273 (DIS,"WORKPIECE IN TOLERANCE")
    .
    .
    .

N2100 M30
"C2" N2101 (DIS,E30)               ;Dimension too long
N2102 M00
"C3" N2103 (DIS,E30)               ;Dimension too short
N2104 E31=-80-E30                  ;Variance between virtual and actual dimension
N2105 (RQT,23,23,LE31)             ;Requalify length offset 23 (Z)
N2106 (GTO,C1)
```

OPERATIONS THAT USE A FIXED PROBE

When you use the G74 probing cycle with a probe that is fixed in position (like an electronic gauge) and a tool mounted on the spindle, you can:

- requalify tools (automatic tool offset modifications)
- check tool wear.

Example 1:
This is an example of tool length offset requalification:

```
N170 G X100 Y100            ;Positioning on the measuring point (probe position)
N180 G74 Z-50 E30           ;Deviation measured and value stored in E30
N190 (RQT,10,1,LE30)        ;Requalification of offset 1 for tool 10 in length (Z) by E30
```

Example 2:
This is an example of tool length offset and tool diameter offset modification.

```
N200 G X100 Y100            ;Positioning on the measuring point (probe position)
N210 G74 Z-50 E30           ;Measured deviation (Z axis and value stored in E30)
N220 G X150
N230 Z-60
N240 G74 X130 E31           ;Measured deviation (X axis) and value stored in E31
N250 E31=E31*2
N260 (RQT,10,1,LE30,dE31)   ;Requalify offset 1 of tool 10 in length (Z) by E30 and in
                             diameter (d) by E31
```

Example 3:
This is an example of tool wear inspection with probe error management (Error management methods are described in Appendix C):

```
N480 (DIS,"TOOL D=10")
N490 T10.10M6
. . . . .
. . . . .
N600 (DPP,10,5,500)
N610 (UAO,9)
N620 GXY
N630 ERR=1                  ;Error managed by program
N640 G74 Z0 E35             ;Tool length measurement
N650 (GTO,A2,STE=1)         ;Branch to A2 if tool is broken (no point probed within the
                            ;5 mm safety distance)
N660 ERR=0                  ;Error managed by system
. . . . .
N1500 M30
"A2" N2001 (TOU,10)         ;Declares tool 10 out of use
N2002 (DIS, "TOOL K.O.")
N2003 M0
```

END OF CHAPTER

Chapter 11

MANAGING THE SCREEN

GRAPHICS VISUALIZATION

This chapter discusses a class of commands that let you control visualisation of graphics and variables from part program. These commands are listed in the table below.

COMMAND	FUNCTION
UGS	Enables a graphic scale
CGS	Clears the graphic scale
DGS	Disables the graphic scale
DIS	Displays a variable

UGS (UCG) - Use Graphic Scale (Machine plot)

The UGS command initialises the graphic display and establishes the limits and the orientation of the graphic display.

Syntax

(UGS [,*ax-orient*],*abs-axis*,*val1*,*val2*,*ord-axis*,*val3*,*val4* [,*third-axis*]**)**

where:

ax-*orient* Is a number (from 1 to 4) that selects the type of axis orientation (see figure). The default value is 1

abs-axis Is the name of the abscissa on the display

val1 Is the lower limit of the abscissa

val2 Is the upper limit of the abscissa

ord-axis Is a name of the ordinate on the display

val3 Is the lower limit of the ordinate

val4 Is the upper limit of the ordinate

third-axis Is the name of the third axis (generally a spindle axis).

Axes Orientation

Example:
(UGS,1,X,100,150,Y,50,250,Z)

The graphic display shows movements between X100 and X150 on the abscissa and between Y50 and Y250 on the ordinate referred to the current origin.

UGS (UCG) - Use 3D Graphic Scale

Syntax

(UGS ,5,*axis1,val1,val2,axis2,val3,val4,axis3,val5,val6* **[**α,β**])**

where:

5	Selects 3D graphic scale
axis1,axis2,axis3	Are the names of the three axes to be displayed
val1,val2	Lower and upper limit of the first axis
val3,val4	Lower and upper limit of the second axis
α	α angular parameter It is the rotation angle to be applied to the horizontal plane during the 3D the display. The typical horizontal plane is XY.
β	β angular parameter. It is the rotation angle to be applied to the vertical plane during 3D display. Typical vertical planes are XY or XZ.

NOTE:
α and β parameters are optional. If they are omitted , the system will take by default:

α = 30°

β = 30°

CGS (CLG) - Clear Graphic Screen

The CGS command clears the profile from the screen leaving the system of coordinates.

Syntax

(CGS)

DGS (DCG) - Disable Graphic Scale

The DGS command disables the graphic display, deletes the displayed profile, and removes the system of coordinates from the screen. After using the DGS command you need to use another UGS command to reinitialise the graphic display.

Syntax

(DGS)

DIS - Displaying a Variable

The DIS command allows values to be displayed to the operator. The control will show the value in the screen area that is reserved for communications with the operator.

Syntax

(DIS,*operand* [**,***operand*] [**,***operand*] [**,***operand*] [**,***operand*]**)**

where:

operand It can be a number, a variable or an ASCII string. Up to five operands can be displayed. All 5 operands cannot exceed be more than 80 characters long.

If *operand* is a number it is within the normal range for variables (5.5 format).

If *operand* is a variable it can be any variable used in assignment blocks.

If *operand* is an ASCII string, it can be a message for the operator. The message can be up to 80 ASCII characters long. Program the message between quotes (" ") in the DIS command.

Examples:

(DIS,100)	displays the value 100
(DIS,E27)	displays E27 current value on the screen
(DIS,MSA)	displays current MSA value (machining allowance)
(DIS,"THIS IS AN EXAMPLE")	displays the following string: THIS IS AN EXAMPLE

END OF CHAPTER

Chapter 12

MODIFYING THE PROGRAM EXECUTION SEQUENCE

GENERAL

This chapter discusses commands that let you modify the program execution sequence by:

- repeating part programs
- executing subprograms
- modify the flow of the program
- delaying and disabling
- releasing the program or suspending its execution
- define devices for access operations on files

Block repetition commands

These commands allow a specified set of program blocks to be executed several times. They can be used for repetitious machining operations such as drilling multiple holes.

COMMAND	FUNCTION
RPT	Opens the set of program blocks to be repeated
ERP	Closes the set of program blocks to be repeated

Commands for executing subprograms

The commands in this class are :

COMMAND	FUNCTION
CLS,CLD,CLT	Calls a subroutine to perform
PTH	Defines the default pathname for the subroutine

A subroutine is a series of blocks defining a machining cycle. The subroutine is stored as a separate file with its own name. Control passes to blocks of the subroutine each time that they are called by the CLS command. The subroutine may be called at any time and from any part of the main program.

Commands for modifying program flow

The commands in this class are:

COMMAND	FUNCTION
EPP	Executes a section of a part-program delimited by two labels
EPB	Executes a block of a part-program
GTO	Performs a jump or skip during the execution of the program
IF ELSE ENDIF	Executes sessions of a part-program depending on certain conditions

The GTO command makes the execution of a program jump to a block which contains a specific label. A jump may be unconditional or conditional based on E parameters, machine logic signals or numeric values.
A conditional jump is performed only if the result is true. No jump is performed if the condition is false.
The commands IF, ELSE, ENDIF allow the flow of the program to be modified conditionally without the need to define labels and program skips.

Commands for delaying program execution and disabling blocks

The commands of this class are :

COMMAND	FUNCTION
DLY	Causes a delay in the execution of the program
DSB	Disables slashed blocks

The delay commands can be used to delay the execution of the program for reasons due to synchronisation.

Commands for the releasing or suspending the execution of a program

The commands in this class are :

COMMAND	FUNCTION
REL	Releases the part-program
WOS	Puts the program in wait mode until a signal is received

Commands for device definition

The commands in this class are:

COMMAND	FUNCTION
GDV	Defines the remote device or drive A for file access operations
RDV	Releases the device defined with GDV

These commands define the remote devices or drive A and are used when read/write operations are performed on the file via the language ASSET.

COMMAND FOR PROGRAM BLOCKS REPETITION

RPT - ERP

The RPT and ERP commands define a set of part program blocks that must be executed a specified number of times. The set of blocks begins with the RPT command and ends with the ERP command.

Syntax

```
(RPT,n)
                    .
                    .
blocks
                    .
                    .
(ERP)
```

where:

n Is the number of times the specified block must be executed. It is an integer from 1 to 65535 that can be programmed directly with a number or indirectly with an E parameter. The control allows five nesting levels, i.e. in a repeat block you can program up to four repeat commands.

blocks It is the set of blocks that must be executed *n* times.

IMPORTANT

The GTO command can be programmed inside an RPT-ERP section. However, if at program end all the cycles programmed with RPT are not executed, the system returns the following error: **'NC063 RPT/ERP CYCLE OPEN AT END OF PROGRAM'**.

Example 1:
The following is an example of repeat command.

Program:

```
(UGS,X,-50,100,Y,-50,100)
(DIS,"N.3 POCKETS")
(DIS,"MILL D12")
N1  S600 T6.6 M6
N2  (RPT,3)
N3  X40 Y35 M3
N4  Z 2
N5  (RPT,2)
N6  G91 Z-8
N7  G90 G1 G41 X40 Y20 F300
N8  X60
N9  Y50
N10 X20
N11 Y20
N12 G40 X40
N13 Y35 F1000
N14 (ERP)
N15 G Z2
N16 (UIO,X80,Y20)
N17 (ERP)
N18 (UAO,0)
N19 Z20
N20 X Y M30
```

Example 2:
This example shows how three repeat levels are nested.

Machining Equidistant Holes

The following example uses a repeat command for machining equidistant holes.

Program:

```
(UGS,X,-50,100,Y,-50,100)
(DIS,"EQUIDISTANT HOLES")
N1  F200 S900 T1.1 M6
N2  G81 R5 Z-10 M3
N3  X10 Y10
N4  (RPT,7)
N5  G91 X10
N6  (ERP)
N7  Y40
N8  (RPT,7)
N9  X-10
N10 (ERP)
N11 G80 G90 XY M5
```

Machining with Roughing and Finishing Cuts

The following example shows repeat commands for machining with one roughing cut and one finishing cut.

Program:

```
(UGS,X,-20,150,Y,-65,60)
(DIS,"DEFINITION OF MACHINING ALLOWANCE")
N1  S350 T6.6 M6
N2  X60 Y M3
N3  Z-50
N4  MSA=0.5
N5  (RPT,2)
N6  G1 G41 X60 Y60 F500
N7  G3 Y-60 I60 J
N8  G1 X100
N9  G3 Y60 I100J
N10 G1 G40 X60
N11 MSA=0
N12 (ERP)
N13 GZ20 M5
N14 X Y M30
```

COMMANDS FOR SUBROUTINE EXECUTION

CLS – CLD - CLT - Call Subroutine

Commands CLS, CLD and CLT recall the execution of a subroutine, i.e. a separate program stored in a file. These commands can call a subroutine from a main program and from another subroutine. Up to four nesting levels are possible.

CLS	Call to a static subroutine:	analysed upon the activation of the main part program
CLD	Call to a dynamic subroutine:	analysed during the execution of the part program and only if the name has not been found earlier
CLT	Call to a temporary subroutine:	analysed during the execution of the part program whenever its call is encountered

Syntax

(CLS,*name***)**
(CLD,*name***)**
(CLT,*name***)**

where:

name Subroutine file name.
name must be specified by a string of upper-case characters, or a string variable preceded by the key "?" (question mark).

name may be a 10 SERIES type name (40 alphanumeric characters) or a file with DOS type name (8 characters plus extension and path).

Example:
(CLS,E:\FILE\PROGRAM.PRG) ;call to subroutine with DOS name

Example:
(CLS,PROGRAM.MIO)

NOTE:

- If *name* is a string variable preceded by the "?" code, the subroutine is not analysed upon the activation of the main part program; it is analysed instead during the execution of the latter. This has the following implications, depending on the triliteral used to call the subroutine:
  - CLS: the subroutine cannot contain skip instructions
  - CLD or CLT: the subroutine can contain skip instructions in as much as run times are analysed

- If the MAIN is a 10 SERIES directory (with file names of 48 characters) the subroutine cannot be resident in a DOS directory, and accordingly three-letter code CLS cannot specify a complete *path*.
- The *pathname* to use for calls to subroutines in a DOS directory may also be declared through the three-letter code PTH described later in this chapter. If the pathname is omitted the directory search for the subroutine is carried out as follows :

  1. No pathname specified via PTH
  The system searches for the subroutine in the directory in which the calling program is found and if it is not there, it looks in the DOS directories to which logical names have been associated during the characterisation phase of the machine.

  2. Pathname specified via the PTH instruction
  The system looks for the subroutine in the directory specified with PTH, if it is not there, it looks in the DOS directories to which logical names have been associated.

- For the **THROUGH mode**: compatibility with the CLS instruction has been maintained, i.e. if the subroutine called contains labels, an error is returned, in as much as this mode does not envisage the activation of the main program upstream. CLD, CLT and paramacros, instead, are managed the same way as the non-through mode.

- The **number of subprograms and "static" labels** managed by the control depends on the corresponding values configured in AMP. For full tables, a memory management procedure is activated, which, to the extent feasible, removes the dynamic and paramacro subroutines from the tables and compacts the static ones (CLS). When it is not possible to free any further space in the memory, a table full message is given out and the system shifts to the same error conditions as we would have during part program selection. At this point, the operator can proceed by either changing the part program or increasing the maximum limits specified for the tables.

- When the tables are full and the **execution of a program is not blocked** thanks to the memory compacting feature, program execution is slowed down, during the subsequent stage of analysis, which inevitably takes place in the event of a blocking error.

- The successful completion of a program activation stage is no guarantee that the program will be executed without errors, if the memory is full. This is the case, for instance, of a CLD calling a number of CLS instructions large enough to fill up the tables.

- **Re-executing a main program** may generate a tables full message even if the program has been activated in its entirety the first time (after its activation). This is so because after the activation of a program, the tables do not contain the file names and the static subroutine tables called either by static subroutines or by dynamic or temporary subroutines.

- A command for a **call to the same program name** entered through triliterals of different types (e.g. first CLT and then CLS) may generate a label unknown message; such a message can be obviated by replacing the CLS call with a CLD call.

- A **run time change to a subroutine** not followed by the reactivation of the main program will be interpreted only if the call is made through a CLT; otherwise, the one analysed previously will be re-executed, with unpredictable effects on program execution.

Example 1:
The following is an example of a call to a subroutine.

Example 2:
Sequence for execution of four nested subroutines.

Example 3:
Call to subroutine specified in implicit mode

SC0.5="PIPPO"
(CLS,?SC0.5)

Example 4:
This example uses subroutines for repeated drilling operations.

Main Program:

```
N19 (DIS,"...")
N20 S2000 F180 T2.02M6
N21 (UTO,1,X-20,Y-25)
N22 (CLS,S600)
N23 (UTO,1,X-15,Y50)
N24 (CLS,S600)
N25 (UTO,1,X60,Y20)
N26 (CLS,S600)
N27 Z0
```

Subroutine S600

```
N501 G81 R-108 Z-130 M3
N502 XY
N503 Y-15
N504 X30
N505 Y0
N506 G80
```

Example 5:
Parabolic profile programming with a parametric subroutine.

Main Program:

```
;N10 MAIN PAR
N20 T1.1M6 S1000 M3 F700
N30 E30=72.795                ;start X
N40 E31=24.28                 ;focal length (twice the focus)
N50 E32=2                     ;Y increment
N60 E33=108.24                ;start Y
N70 E34=0                     ;final Y
N80 GX0 Y120
N90 (CLS,PAR)
```

Subprogram PAR:

```
;N500 PAR
;N501 Parametric subroutine: Complete parabola execution, internal profile.
N502 G1 G42 XE30 YE33
N503 E36 = E33
"START" N504
N505 E36=E36-E32
N506 (GTO,END,E36<E34)
N507 E35=SQR( 2*E31*ABS(E36))
N508 XE35 YE36
N509 (GTO,START)
"END" N510
N511 E35=SQR (2*E31*ABS(E34))
N512 XE35 YE34
;N513 Second part of the parabola.
N514 E42=E32
N515 E43=E34
N516 E44=E33
N517 XE35 YE43
"START2" N518
N519 E43=E43+E42
N520 (GTO,END2,E43>E44)
N521 E35= -(SQR (2*E31*ABS(E43)))
N522 XE35 YE43
N523 (GTO,START2)
"END2" N524
N525 E35= - (SQR(2*E31*ABS(E44)))
N526 G40 XE35 YE44
N527 GX0
```

EXAMPLE 6:

Call to the dynamic subroutine, specified in an implicit manner with the PAR subprogram of the previous example

Main program:

```
;N10 MAIN PAR
N20 T1.1M6 S1000 M3 F700
N30 E30=72.795          ;Initial value of X
N40 E31=24.28           ;Focal length
N50 E32=2               ;Increment of Y
N60 E33=108.24          ;Initial value of Y
N70 E34=0               ;Final value of Y
N80 GX0 Y120
SC0.5="PAR"
(CLD,?SC0.5)
```

PTH - Declaration of the default *pathname*

The PTH command declares the *pathname* to be used as the default in calls to subroutines and paramacros with DOS names.

Syntax

(PTH,*mode* [,*pathname*]**)**

where:

mode *mode* may have the following values and significance:

mode=0
Taken as the default *pathname* for CLS calls, the *path* of the main program.

mode=1
Declares the *path* to use when there is a (PTH,2) instruction active.
This mode may be of use when a *pathname* defined initially with (PTH,1,*pathname*) has to be called numerous times in the part program with (PTH,2).

mode=2
Activates the *path* specified in the instruction or, if none is specified, the one declared previously with a (PTH,1,*pathname*) instruction.

pathname Path to be declared as the *path* for CLS calls; this is an optional parameter.

NOTE:
The *pathname* declared with three-letter code PTH is preserved even after a RESET of the process in which it was programmed.

EPP - Executing a Portion of a Program

The EPP command allows you to execute a subprogram, i.e. a portion of a program delimited by two blocks with label fields.

Syntax

(EPP,*label1***,***label2***)**

where:

label1 Label field of the first block to be executed. A label is a sequence of up to six alphanumeric characters.

label2 Label field of the last subprogram block.

Characteristics:

Program the block label between quotes ("LABEL1") even if in EPP they must be declared without quotes. The system accepts up to 5 nesting levels.

In contouring operations, you can use the EPP command for finish milling with the same program blocks you wrote for roughing. During the roughing phase, use the MSA command to program the machining allowance for finishing.

In positioning operations, you can program points for a centring operation, then use the EPP command to call for different tools in order to execute different operations on each hole. The EPP command can be used to execute a complete machining operation at different orientations on the active interpolation plane.

Example 1:

```
    .
    .
"START"N25                  First block with label
    .
    .                       An EPP cannot occur here
    .
"END"N100                   Last block with label
    .
    .
    .
N150 (EPP,START,END)EPP command that specifies the labels. The control executes blocks N25
                            to N100; after this point you must resume execution with the block
                            written after EPP (N150).
```

IMPORTANT

The **'NC062 NESTING OF EPP > 5'** error occurs if more than five instructions are nested in an EPP command.
The GTO command can be programmed inside an EPP section. However, if at program end all the cycles programmed with RPT are not executed, the system returns the following error: **'NC063 RPT/EPP CYCLE OPEN AT END OF PROGRAM'**.

Example 2:
This example shows how to use the EPP command in a positioning operation:

Program:

```
(UGS,X,-110,110,Y,-110,110)
N1 (DIS,"DRILLING CENTRE HOLES")
N2 F300 S2000 T1.1 M3 M6
N3 G81 R0 Z-3
"D6"N4
N5 X100 Y100
N6 X-100
N7 Y-100
N8 X100
"D10"N9
N10 X40 Y40
N11 X-40
N12 Y-40
N13 X40
"END"N14
N15 G80
N16 (DIS,"BIT D6")
N17 F200 S1800 T2.02 M3 M6
N18 G81 R Z-22
N19 (EPP,D6,D10)
N20 G80
N21 (DIS,"BIT D10")
N22 F220 S1600 T3.3 M3 M6
N23 G81 R Z-24
N24 (EPP,D10,END)
N25 G80
```

EPB - Execute Part-Program Block

The three-letter code EPB allows execution of a part-program block.
Execution can be conditioned by the result of a comparison specified within the command; If the condition is not met an alternate part-program block may be defined so as it can be executed.

Syntax

(**EPB,** *part_program_block1 [, par1 operator par2 [, part_program_block2])*

where:

part_program_block1 — Is the part-program block to be executed.
This can be a string in inverted commas or a character, local or system variable.

par1 — Is a local or system variable or a constant whose value is compared to the value contained in the *par2 parameter.*

operator — Logic operators which can be used in the expressions :

=	equal to
<	less than
>	greater than
<>	not equal to
<=	less than or equal to
>=	greater than or equal to

par2 — Is a local or system variable or a constant whose value is compared to the value in the parameter *par1.*

part_program_block2 — Is the block of the part-program to execute only if the specified condition for *part_program_block1* is not met. It is an optional parameter.
This can be a string in inverted commas or a character, local or system variable.

Characteristics:

If *par1 ,operator* and *par2* conditions are not specified, the program executes the *part_program_block1 block.*

Only one level of nesting is accepted by the system with the three-letter code EPB.

Example:
(EPB, “(EPB,’E1=1’)”) : accepted by the system
(EPB, “(EPB,’(EPB,SC0.100)’)”) : NOT accepted by the system (error: FORMAT ERROR)

The part-program blocks specified in the three-letter code EPB are not analyzed in the activation phase of the part-program, therefore it is up to the programmer to see that these blocks do not cause any malfunctioning of the part-program.

Example:

(EPB,”(CLS,SUBROUT)”) : the subprogram SUBROUT is not pre-analyzed in the activation phase, therefore it cannot contain any jump (GTO) instructions.

(EPB,’ “LABEL” ’) : The label LABEL will not be inserted in the label table of the part-program.

(EPB, “ (DIS, E1)”) : equivalent to the block (DIS, E1)

SC0.30= ” E1 =10”
(EPB, SC0.30) : equivalent to the block E1=10

SC40.30 = “# X10”
(EPB, SC40.30) : equivalent to the block #X10

(EPB,”(CLS,SUBROUT)”,E1=34) : calls the subprogram SUBROUT only if E1=34

(EPB,” E1=100”, SN1=25,” E1=0”) : assigns 100 to E1 only if SN1=25, otherwise assigns 0 to E1

(EPB,”(EPP,LAB1,LAB2)“,SC0.2=“OK”,“(EPP,LAB3,LAB4)“) : executes from the label LAB1 to the label LAB2 if SC0.2 is equal to OK, otherwise executes from the label LAB3 to the label LAB4.

(EPB, “(EPB,’ E1 = 2’, E0 < 100)”, E0 > 70) : assigns 2 to E1 only if E0 is between 70 and 100.

(EPB,“ (EPB,’ E0 = 5’, E0 < 5) ",E0 < 10,E0=10) : assigns the value 5 to E0, if E0<5; assigns 10 to E0 if E0>10.
If 5<E0<0, E0 does not change.

BRANCHING AND DELAY COMMANDS

GTO - Branch Command

The GTO command performs a jump to the specified block if the condition expressed in the command is true.

Syntax

(GTO,*label* [**,***par1 operator par2*])

---

where:

label Label of the program block to branch to. A label is a string of up to 6 alphanumeric characters. In the destination block the label must be between quotes while in GTO command the label must be programmed without quotes. Also, the label can be a character variable preceded by "?"

par1 It is the local or system variable whose value is compared to the value of the *par2* variable.

operator Logical operators that can be used in expressions:

=	equal
<	less than
>	greater than
<>	different than
<=	less than or equal to
>=	greater than or equal to

par2 It is the local or system variable whose value is compared to the value of the *par1* variable.

Characteristics:

If *par1, operator* and *par2* are not specified, the program always jumps to the block marked by the label.

The label the part program must branch to can also be expressed as a local or system variable (preceded by the "?" character).

This facilitates execution of programs having a great number of blocks. In particular, by programming the label as a variable it is possible to program a default label and call it "DEFLAB": The part program positions on "DEFLAB" if the label in the variable does not exist.

Example 1:
;SC0.3 contains the positioning label

```
(GTO,?SC0.3)
"ONE"
.
.
.
.
"TWO"
.
.
.
.
"THREE"
.
.
.
"DEFLAB"
.
.
.
```

If SC0.3 does not contain "ONE", "TWO" OR "THREE" the part program positions on the "DEFLAB" block. If the "DEFLAB" label exists, the **'NC054 Undefined label'** error is displayed and part program execution is interrupted.

Example 2:

N01 (GTO,START)	the part program always branches to "START"
N10 (GTO,END,E1 > 123)	branch to "END" if the value of E1 is greater than 123
N20 (GTO,LAB1,@COOLANT = 1)	branch to "LAB1" if the PLUS variable @COOLANT is on
N30 (GTO,START,E1 <> E5)	branch to "START" if the value of E1 is different from that of E5
N40 (GTO,LAB1,SC1.2H = "OK")	branch to "LAB1" if the two characters beginning with SC1 are equal to OK
N50 SC1.3="ABC"	prepares the variable for the following block
N60 (GTO,?SC1.3)	branch to ABC label

Example 3:
The instruction:

(GTO,END,SC2.3H = "ABC")

branches to the "END" label in the program if the three characters (.3) beginning from SC2 are ABC.

NOTE:
The character string to be compared in the branch command must always be programmed between quotes.

Example 4:
Here is an example of conditional branching in slot milling.

Program:

```
(UGS,X,-50,50,Y,-50,50)
N1 (DIS,"MILL A SLOT")
N2 F500 S2000 T1.1 M3 M6
N3 E31=-3.5
N4 E32=-24
"START"
N6 G X Y-10
N7 Z(E31)
N8 G1 G42 X Y-20
N9 X-30
N10 Y20
N11 X30
N12 Y-20
N13 G40 X
N14 Y-10
"END"
N16 E31=E31-3.5
N17 (GTO,START,E31>E32)
N18 E31=-25
N19 (EPP,START,END)
N20 G Z10
```

Example 5:
This is an example conditional branching in cylindrical thread machining.

E30 = Diameter of Next Cut
E31 = Cut Depth Increment (Diameter)
E32 = Return Diameter
E33 = Final Diameter

Program:

```
N1 (DIS,"THREAD DIA 60")
N2 S150 T5.5 M6
N3 G0 X66 Y98 Z5 M3
N4 E30=56.8
N5 E31=0.5
N6 E32=50
N7 E33=60
"I" N8
N9 G0 Z5
N10 U(E30)
N11 G33 Z-45 K3
N12 GU(E32)
N13 E30=E30+E31
N14 (GTO,F,E30>E33)
N15 (GTO,I)
"F" N16
N17 GU(E32)
N18 Z5
N19 U(E33)
N20 G33 Z-45 K3
N21 GU(E32)
N22 Z20
```

IF ELSE ENDIF

The IF command allows the opening of a part-program session which will be executed only if a condition specified within the command is met. This part-program session must terminate with the ENDIF command : the ELSE command can be optionally included in the IF command, which delimits, up until the ENDIF command, a part-program session which will be executed if the condition specified in the IF command is not met.

Syntax

(IF, *par1 operator par2* **)**
(ELSE)
(ENDIF)

where:

par1 — Is a local or system variable or a constant whose value will be compared with the *par2* parameter.

operator — Logic operators which can be used in the expressions :

=	equal to
<	less than
>	greater than
<>	not equal to
<=	less than or equal to
>=	greater than or equal to

par2 — Is a local or system variable or a constant whose value is compared to the value in the parameter *par1.*

Characteristics.

If the condition specified in the IF command is met, the blocks specified up until the ELSE command (if present) or until the ENDIF command (if the ELSE command is not present) will be executed, otherwise they will be ignored.

The blocks between the ELSE command and the ENDIF command will be executed only if the condition specified in the IF command is not met.

5 levels of nesting are allowed with the IF ELSE ENDIF command.

Every IF command must have a corresponding ENDIF command.

Examples:

```
( INP," E0 VALUE " ,30 , E0)
( IF , E0 = 3)
( DIS , " E0 is equal to 3 ")
(ELSE)
( IF E0 > 3)
( DIS , " E0 is greater than  3 ")
(ELSE)
( DIS , " E0 is less than  3 ")
( ENDIF)
( ENDIF)
```

```
( INP," E0 VALUE" ,30 , E0)
( IF , E0 >10 )
( IF , E0 >20 )
( IF , E0 >30 )
( IF , E0 >40 )
( IF , E0 >50 )
( DIS , " E0 is greater than 50  ")
( ENDIF)
( ENDIF)
( ENDIF)
( ENDIF)
( ENDIF)
```

DLY - Defining Delay Time

The DLY command specifies a delay in program execution.

Syntax

(DLY,*time***)**

where:

time Delay time in seconds (minimum value 0.1). You can program the delay with a numerical constant or an E parameter.

DSB - Disable Slashed Blocks

This command enables/disables slashed blocks.

Slashed blocks have "/" symbol as first character and their execution is conditioned by the DBS value.

Syntax

DSB= *value*

where:

value The value may be:

DSB=0 slashed blocks are executed
DSB=1 slashed blocks are not executed

REL - Releasing the part program

The REL command releases the part program and all subroutines.

Syntax

(REL)

Characteristics:

The REL function disables the part program. It can be entered through MDI.

After the program has been released, the following message will be displayed: ***'NC058 END OF PROGRAM'***.

WOS - WAIT on signal

Syntax

(WOS,*par1 operator par2***)**

where:

par1 Is a local or system variable or a constant to be compared to the value of *par2*.

operator Logic operators that can be used in expressions:

=	equal to
<	lower than
>	higher than
< >	different than
<=	lower or equal to
>=	higher or equal to

par2 Is a local or system variable or a constant to be compared to the value of *par1*.

Characteristics:

The process where the WOS function is programmed goes on WAIT until the condition is satisfied.

By default the WOS command is synchronised.

If the system is on WAIT during program debugging (because a WOS function has been programmed), it is possible to assign to the *par1* variable the value that satisfies the condition specified in WOS.

To make this assignment, follow these steps:

- Press CYCLE STOP
- Key in the correct value of the *par1* variable.
- Press CYCLE STOP

Example:
(WOS,E1>4)
(WOS,SC10.5 = "GOOFY")
(WOS, TOOL_STATUS>=E3)
(WOS,!USER_VAR3.5CH 0 SC4.5)

DEVICE DEFINING COMMANDS

GDV - Definition of the *device* for file access

This three-letter code is used to reserve use of drive A for file read/write operations and to define remote *devices* on which to work.

Syntax

(GDV, *device***)**

where:

device Name of *device*.
May be a letter or, if preceded by a "?" key, a local or system variable specifying the drive to be used.
For the floppy drive, the letter "A" must be used as the identifier.
For remote devices, the name to be given is that with which it was defined with the option MINI DNC E65/E66 (e.g. "K").

Characteristics:

If drive A is already being used by another process or UTILITY (e.g. DOS SHELL), the error **NC251 Driver busy or not configured** is generated; this error can be handled by the part program (for further details, refer to appendix C).

Examples:
(GDV,A)

Reserves use of drive A.

SC0.1='A'
(GDV,?SC0.1)

Use of drive A is reserved, as specified in variable SC0.1

IMPORTANT

A system RESET cancels access to drive A but not to remote devices.

RDV - Release *device*

This three-letter code is used to release drive A or the remote *device* defined previously with the three-letter code GDV, so as to make it available for other users.

Syntax

(RDV, *device***)**

where:

device Name of *device*.
May be a letter or, if preceded by a ? key, a local or system variable specifying the drive to be used.
For the floppy drive, the letter "A" must be used as the identifier.
For remote devices, the name to be given is that with which it was defined with the option MINI DNC E65/E66 (e.g. "K").

Characteristics:

A process reset is tantamount to automatic execution of a (RDV,A) command.

Examples:
(RDV,A)

Releases drive A previously allocated with GDV.

SC0.1='A'
(RDV,?SC0.1)

Similar to the previous example, but with the drive specified in variable SC0.1.

END OF CHAPTER

HIGH SPEED MACHINING

GENERAL CONSIDERATIONS

The High Speed Machining feature is used for machining surfaces (profiles) defined by points, created by CAD/CAM systems, on machine tools having 3 to 5 axes (3 linear + 2 rotary).

The High Speed Machining feature must be enabled in the AMP environment, by selecting the appropriate field in the PROCESS CONFIG section (**PROC CHAR** softkey).

This feature may only be enabled for the first 4 processes.

To program this feature, proceed as follows:

1. Create a setup file (part program) that contains all the parameters for handling the profile: tools, axes and kinematics of the machine.

   The configuration file will be recalled in the main program through a three-letter command, with the following format:

*(**HSM**, setup file name)*

2. Create the profile by inserting it directly in the main program:

   Example:

```
; HSM PROGRAMMING EXAMPLE
G0 X..Y..Z.. A.. B.. F…
------
(HSM, CONFIG1)
------
(CLS,PROFILO1 ----------------------------------------------→ G61
------                                                        G1 X…Y..Z..A..B..
------                                                        ---------------------------
; END OF PROGRAM                                              ---------------------------
------                                                        G60
```

The profile may also be inserted in a specific file which will be recalled as a subprogram by the ***CLS*** instruction.

Example 2:

```
; HSM PROGRAMMING EXAMPLE
G1 X..Y..Z.. A.. B.. F…
 -----
(HSM, CONFIG1)
G61
G1 X..Y..Z..A..B..
 -----
G60
; END OF PROGRAM
```

PROGRAMMING POINTS AND CHARACTERISTICS OF THE PROFILE

A profile is a set of points that make up the ISO part-program of the surface to be machined, created by the CAD/CAM system in which the characteristics described in this chapter are to be respected.

On the basis of the programmed points a polynomial curve will be constructed, defining the path to be followed. This path will pass through the programmed points with a configurable tolerance. The methods by which the points are to be linked will be defined by the G01 and G00 codes that may be programmed together with the points.

IMPORTANT

To ensure that the polynomial curves are calculated correctly, we recommend the points be programmed with at least 5 figures after the decimal point (e.g. 10.37854); programming with fewer figures may result irregularities on the profile.

The sections executed in **G00** will be considered as individual positioning operations; each point will be linked to the next by means of a “linear” movement to be performed with the dynamic traverse positioning (each section in G00 will start at zero speed and end at zero speed). For this reason, G00 mode will not calculate the polynomial curve. At the end of a section in G00 there will be no pause (end of movement synchronism, entry into tolerance status, etc..) and the next movement will be carried out immediately. This behaviour is similar to the programming of G01 and G09 codes in the same block.

With **G01,** each point will be linked geometrically to the following ones by means of a polynomial curve, so the generated path may be considered “**continuous**”. This link will be interrupted by the programming of a G00 or the programming of special G codes described below. The dynamics of the sections in G01 are the same as the “cutting” movements (such as normal G01 movements in ISO programming).

In addition to the **G01** and **G00** functions, the following G functions, specific for the HSM feature, may be programmed in the profile.

G61
Determines the start of the profile and must be programmed in a block on its own. When the G61 function is activated, there must be no form of virtualization active (UPR,UVP,UVC,TCP).

Before activating the G61 function, the setup file must be defined by means of the instruction:

*(**HSM**, setup file name)*

The G61 command may ONLY be executed within a part program in AUTO or BLK BY BLK status.

G60
Determines the end of the profile and must be programmed in a block on its own.

If the machine is in single STEP execution, the profile between G61 and G60 is considered as a single instruction. To stop its execution, it is necessary to switch to HOLD status.

G62
Splits a profile in two parts and determines the point where one profile ends and another begins, maintaining continuity between the two curves.

The points preceding the G62 function will be used to generate a first curve, while the subsequent points will be used to generate another one. These curves **will be** linked and will therefore be continuous; the initial inclination of the second curve will correspond to the final inclination of the previous curve.

As regards dynamics, with G62 no deceleration and acceleration ramp will be generated to link the two curves. This G function must be programmed in a part program block on its own.

G63
Splits a profile into two parts and determines the point where one profile ends and the other one begins, **maintaining continuity** between the two curves. The points preceding the G63 function will be used to generate a first curve, while the subsequent points will be used to generate another one. While with G62, the initial inclination of the second curve depends strictly on the final inclination of the first, with G63, the initial inclination of the second curve **IS NOT** influenced by that of the first. To maintain continuity, a “radius” that depends on the chordal error with which the splines are to be calculated is inserted.

With G63 a reduction in speed may occur at the point where the two curves are linked. This G function must be programmed in a part program block on its own.

G66
Splits a profile into two parts and determines the point where one profile ends and the other one begins, creating a **discontinuity** between the two curves, that is, the point preceding the G66 represents an edge. At this point, two curves are generated, the first using the points preceding the G66 function and the second using the subsequent points. These curves **will NOT be** linked, and so there will be a discontinuity.

This discontinuity will be reached at zero speed; the first curve will therefore end with a deceleration ramp to 0 (zero) speed after which the second curve will be tackled with an acceleration ramp to reach the required machining speed. This G function must be programmed in a part program block on its own.

G67
With **G67**, a “discontinuity” may be defined on the profile defined with G66. What changes is the dynamic approach to the edge, that is, the end of the curve is not reached at zero speed but at a speed value (vs) that enables the axes to reach the edge without any dynamic problems. This speed value is calculated on the basis of the acceleration that may be withstood by each axis. This G function must be programmed in a part program block on its own.

Considerations on the use of the G62,G63,G66 and G67 functions (transition codes)

The transition G codes are particularly useful when “similar”, repetitive curves are to be defined (providing the programmed points are also similar and repetitive).

Supposing we have a profile defined by 100 points of which the first 50 represent the first machining pass (from **p1** to **p50**) and the other 50 (from **p50** to **p99**) the same profile shifted slightly (second pass).

As the points between **p1** and **p50** are “similar” to the points between **p50** and **p99**, the conditions for calculating the two polynomial curves will also be similar. Two “parallel”, almost identical curves will therefore be generated.

If the G62 function has not been programmed on point p50 the NC may generate curves that are not perfectly parallel. This normally undesired effect is due to the fact that the calculation of the polynomial takes into account the “history” along the calculated paths.

The “history” of point p1 is clearly different from that of point p50. In fact, point p1 has no history while in point p50, the NC has followed a path determined by the first 49 points.

When the G62 function is inserted, it cancels the “history” and produces a geometrical pattern almost identical to the one calculated starting from point p1.

In machining processes that entail several passes, failure to program G62 would have the undesirable effect of producing different levels of machining between one pass and another.

GENERAL HIGH SPEED MACHINING PROGRAMMING STRUCTURE

Between the G61 and G60 blocks it will only be possible to program the points that make up the profile to be machined or the G codes for defining their management method: no other type of programming will be allowed.

Points may be programmed using, absolute programming may be used by means of **(G90)** or incremental programming by means of **(G91)**. All numerical parameters required may be defined directly or by means of **E** or **L** variables: programming with expressions is not valid so XE(E2) or X(E1+E2) type programming is not allowed, while XE1 is allowed.

The syntax of the allowed program lines will be:

N… [**G00** | **G01**] [**G90** | **G91**] [points] F….

N… [**G62** | **G63** | **G66** |**G67**]

Activation of the HSM (High Speed Machining) feature **G61** forces of G01 and G90, modes while at the exit (**G60**) the G functions active when G61 was programmed will be restored.

The first point programmed **MUST** be expressed in absolute positions (G90) and must contain the programming of all axes associated with the HSM programming (axes configured in the HSM setup file).

Interaction with Machine Logic

The G61/G60 program section will be considered, from the system point of view, as a single program block. As regards interfacing with the machine logic, a request for consent for movement will be made when the G61 function is reached, and an end of movement request will be made when the G60 function is reached (in the same way as for the G27 and G28 continuous movements).

A regards consent for movement, the **XW03** variable, which contains the type of movement, will be set as shown below:

POINT DEFINING CONVENTIONS

Points and machining coordinates

Before defining how the points are handled, it is necessary to specify what they represent as programming may be executed in relation to three types of coordinates, that is:

- **Cutter Contact Points**, which refer to the actual cutting point
- **Cutter Location Points**, which refer to the point normally indicated as the centre of the tool
- **Axis Location Points**, which refer to an arbitrary point fixed to the machining axes

The cutter contact points are linked to the cutter location points through the geometry and orientation of the tool. The axis location points are linked to the cutter location points through the geometry of the machine tool. In machining processes with three axes, the coordinates will simply be translated while, in those with five coordinated axes, rototranslation matrices that take into account the geometrical transformations due to the movement of the rotary axes will be applied. The figure below shows what is meant by cutter contact points, cutter location points and axis location points.

Points are defined by means of normal axis coordinates in the format [**Axis name**][**Position**]; example: X100 Y200 Z40.

Tool Direction: IJK

The tool direction represents the orientation of the tool (from the tip to the attachment) within the part reference system.

In the sections that follow, versor means a vector of unit length.

Two methods may be used to define the tool direction. The first is by directly programming the versor that identifies the tool direction. This versor is expressed using the **ijk** coordinates in the format:

[**i**] [**X-coordinate component**] [**j**][**Y-coordinate component**] [**k**][**Z-coordinate component**]

The system will automatically normalize the length of the versor to the unitary length (1.0).

The second way of defining the tool direction is by programming the rotary axes. The system will automatically determine the three components of the **ijk** versor depending on the kinematics of the machine.

Normal to the Surface Direction: MNO

The normal to the surface direction represents the direction of the “line” perpendicular to the surface to be machined (starting from the surface) within the part reference system.

There are two ways of defining the direction normal to the part. The first is by directly programming the versor that identifies the normal direction. This versor (of a unitary length) is expressed using the **mno** coordinates in the format:

[**m**] [**X-coordinate component**] [**n**][**Y-coordinate component**] [**o**][**Z-coordinate component**]

The system will automatically normalize the length of the versor to the unitary length (1.0).

The second possibility consists of asking the system to determine this direction automatically. The direction will be calculated on the basis of the tangent to the profile (displacement direction), on the basis of tool direction (vector **ijk**) and angle of contact between the part and the tool (angle α). This angle is conventionally assumed to be 0° if the tool touches the part with its tip, 90° if the tool touches the part with its left side and -90° if it touches it with its right side (in the example below, α=90°). Thanks to this calculation, the **mno** versor is normal to the tangent to the profile and defines an angle α (angle of contact) with the tool versor **ijk**.

This type of approach is only significant when the contour is to be machined. When a surface is to be machined, this approach could fail as there is no information about the “surface” to be machined, only information about the “direction” of displacement.

Tool Radius Application Direction: PQD

The direction of application of the tool radius represents the direction in which radius compensation is to be applied (starting from the centre of the tool) within the part reference system.

There are two ways of defining the tool radius application direction. The first is by directly programming the versor that identifies the direction. This versor (of a unitary length) is expressed using the **pqd** coordinates in the format

[p] [X-coordinate component] [q][Y-coordinate component] [d][Z-coordinate component]

The system will automatically normalize the length of the versor to the unitary length (1.0).

The second way is to have this direction calculated automatically by the system. The direction is calculated automatically on the basis of the tool direction (**ijk** versor) and the normal to the part (**mno** versor). This calculation ensures that the **pqd** versor is normal to the tool direction and is on the plane formed by the **ijk** and **mno** versors.

Programming of the versor pqd is only significant when specific cutting strategies are applied.

Paraxial Compensation Factors: UVW

These factors can be used instead of mno and pqd versors for determining tool radius compensation: instead of working perpendicularly to the programmed profile (mno), this compensation is obtained by moving the programmed point along the uvw direction by a quantity equivalent to the radius. The module of the uvw vector need not be unitary.
In order to define the paraxial compensation factors two techniques can be adopted. One consists of programming directly the vector that identifies this direction, as expressed through the **uvw** coordinates in this format:

[u] [Abscissa Component] [v][Ordinate Component] [w][Vertical Component]

The other technique consists of having the system calculate the direction automatically. The direction is calculated through a similar procedure as is used for the mno versor, but angle α can only assume the following values: +90°, if the tool touches the work piece with the left side; –90° if the tool touches the work piece with the right side. For this reason, the management of uvw vectors is compatible with cylindrical tools as well as with spherical routers working on the side. Toroidal tools are not admitted. Furthermore, the uvw vector differs from mno at corners where the length of the vector is multiplied by an appropriate correction factor.

The basic difference with respect to mno versors, in fact, is observed at the “pointed” parts of the profile, as shown in the figure below.

Tangential Axis

The tangential axis is an axis whose position is calculated so as to remain tangential to the profile described. It is calculated on the basis of the tangent to the polynomial curve on the work plane.

An initial value of the tangential axis (first programmed point) may be defined and the subsequent positions may be calculated on the basis of this value.

FEATURES PROVIDED BY HIGH SPEED MACHINING

Depending on the type of machine tool used, the points programmed and a series of additional parameters, different features may be obtained using “High Speed Machining”.

Generic machines or machines with 3 axes are characterized by the fact that they do not have axes for orienting the tool during the machining phase. The tool direction is generally fixed.

Tool Radius and Length Compensation

In order to use tool radius and length compensation, the type of axis positions and vectors as defined in the table below must be included in the part program. The feed rate may refer to the point of contact between the tool and the part (chip removal speed, therefore at the Cutter Contact Point) or the centre of the tool (at the Cutter Location Point). The difference between these two types of settings is significant when tools with a large diameter are used, that is, where the path of the centre of the tool is significantly different from that of the profile of the part.

Setting	Description
Axis positions	The points must be expressed in **Cutter Contact Points**
Tool Direction	The tool direction vector is defined during the set-up phase (on a 3-axis machine, it is fixed and defined by the TOD parameter) and remains unchanged throughout the machining process. ijk programming is ignored.
Normal to Surface	The mno vector must be programmed or automatically calculated by the system
Radius Application	The pqd vector must be programmed or automatically calculated by the system

Example:
N001 G61
N002 G1 X10Y10Z10 m0n0o1 p0q0d1 F10000
N002 X20Y10Z10 ….
…..
N100 G60

Tool Length Compensation

In order to use tool length compensation, the type of axis position and vectors as defined in the table below must be included in the part program. The feed rate will refer to the centre of the tool (at the Cutter Location Point). However, starting from the points referred to the tool centre, it is possible to recalculate the Cutter Contact Points to be able to apply the tool diameter compensation (see TOL configuration parameters) and to define the parameters of feed rate relative to the profile actually machined (on the Cutter Contact Point).

Setting	Description
Axis Position	The points must be expressed as **Cutter Location Points**
Tool Direction	The tool direction vector is defined in the set-up phase (on a 3-axis machine, it is fixed and defined by the TOD parameter) and remains unchanged throughout the machining process. ijk programming is ignored.
Normal to Surface	It is necessary to program the mno vector when you want to apply tool diameter compensation.
Radius Application	It is necessary to program the pqd vector (or request its automatic determination) when you want to apply tool diameter compensation.

Example:
N001 G61
N002 G1 X10Y10Z10 F10000
N002 X20Y10Z10 ….
…..
N100 G60

No Tool Compensation

In this case, only the positions of the axes as defined in the table below have to be included in the part program. The feed rate will refer to the actual movement of the axes (Axis Location Point). However, starting from points referred to the part dimensions, it is possible to recalculate the tool centre points (Cutter Location Point) to apply tool length compensation and, if necessary, recalculate the Cutter Contact Points in order to apply tool diameter compensation (see the TOL configuration parameters). Thus, it will be possible to set feed rate relative to the motion of the tool centre (on the Cutter Location Point for tool length compensation) or relative to the machine profile (on the Cutter Contact Point).

Setting	Description
Axis Positions	The points must be expressed in **Axis Location Points**
Tool Direction	The tool direction vector is defined during the set-up phase (on a 3-axis machine it is fixed and defined by the TOD parameter) and remains unchanged throughout the machining process. It is **only** used if the setting of the feed rate in relation to the centre of the tool has been requested, otherwise it is ignored. The setting of ijk is ignored.
Normal Surface	It is necessary to program the mno vector when you want to apply tool diameter compensation.
Radius Application	It is necessary to program the pqd vector (or request its automatic determination) when you want to apply tool diameter compensation.

Example:
N001 G61
N002 G1 X10Y10Z10 F10000
N002 X20Y10Z10 ….
…..
N100 G60

Tangential Axis Management

When management of the tangential axis is requested, its position is calculated automatically by the system. Any programming of the tangential axis within the part program defines further rotations of the axis with respect to the tangent calculated by the system. The programming of the value 0 (zero) on the axis activates the calculation of the position by the High Speed algorithms.

SETUP

The configuration of the HSM environment is contained in a specific setup file: all dimensional parameters must be expressed in the unit of measurement of the machine tool.
This setup is activated whenever the G61 code is programmed.
All the numerical values of the part program can be defined directly or through **E** or **L** parameters.

The setup file may be divided into four sections:

- a section of *General three-letter codes*
- a section of *Axis Setup three-letter codes*
- a section of *Machine Configuration three-letter codes* or machine kinematics definition section.
- a section of *PathOptimizer Configuration three-letter codes*

General Three-Letter Codes

(**PNT**,Type,Param,Poly)	Definition of types of points
(**VER**,Ijk,Mno,Pqd,uvw)	Versor management
(**JRK**,Enable,Jrs,Entities,Tstab,Tintgr,Ttop)	Look Ahead management
(**THR**,Tol,TolV,Scale,NullMov,Chord, ErrRot,MaxLen)	Threshold programming
(**TOL**,Len,Radius,EdgeRad,Angle,OriginLen,Rorig, RgOrig)	Tool definition
(**TOD**,Xcomp,Ycomp,Zcomp)	Tool direction (3D)
(**CRV**,Len,Ratio,Mode)	Change in curvature management
(**EDG**,Angle,VAngle,Mode,Acc)	Edge management

Axis Setup Three-Letter Codes

(**AXI**,Name,Id,Type,CinType)	Definition of types of axis
(**AXP**,Name,NullMov,Pitch,Lim-,Lim+)	Definition of axis parameters
(**DIN**,Name,Vmax,Amax,Jmax,Vrap,Arap,Jrap)	Definition of axis dynamics

Machine Configuration three-letter codes

(**MAC**,Type,Bed,Um)	Defines machine type
(**CIN**,Name,Xoff,Yoff,Zoff,Xrot,Yrot,Zrot)	Defines machine kinematics

PathOptimizer Configuration three-letter codes

(**SMT**,Toll,TollX,TollV,Racc,RaccX,RaccV,C0,C0X,C0V)	Smoothing parameters and tolerances
(**SMS**,Split,SplitX,SplitV,Cond,CondX,CondV,Len,Pnt,C1,Progr,Rcnc)	Special smoothing parameters

Type of points described in the part program

The syntax of the three-letter code that defines the type of points described in the part program is as follows:

(PNT, Type, Param, Poly) *example: (PNT, CLP,CLP,QUI)*

Parameter	Type	Values	Description
Type	Character	CCP CLP AXI Obligatory	Defines the type of points described in the Part Program **CCP** Cutter Contact Points: the points entered are defined in “Cutter contact points”, so tool compensation (radius and length) may be executed on these points. **CLP** Cutter Location Points: the points entered are defined in “Cutter location points“, so tool compensation (length only) may be executed on these points. **AXI** Axis Location Points: the points entered are defined in “Axis location points”, so no type of compensation may be executed on these points.
Param	Character	CCP CLP AXI Obligatory	Defines the type of profile for which the feed rate is programmed in the Part Program and depends on the type of points entered. The feed rate always refers to the “3D” profile. Any rotary or additional axes present are not involved in the calculation of the feed rate but “follow” the execution of the 3D profile. The feed rate is however limited when the latter axes, in following the 3D axes, tend to exceed their dynamic limits. **CCP** The programmed Feed rate refers to the profile generated on the “Cutter contact points”; this setting is clearly only valid if the points entered are of the CCP type. **CLP** The programmed Feed rate refers to the profile generated on the “Cutter location points”; it may be used for the CCP and CLP type points, while for AXI type points it is only valid when the tool direction can be determined (either from the programming of the rotary axes or from the programming of the ijk versors). **AXI** The programmed Feed rate refers to the profile generated on the “Axis location points”; it may only be used for AXI type points.
Poly	Car.	CUB or QUI Obligatory	Defines the degree of the polynomial curve generated for the execution of the programmed profile. **CUB** A cubic polynomial is generated. **QUI** A quintic polynomial is generated.

Versor management methods

If programmed, the versors must refer to the abscissa, ordinate and vertical axes, respectively. The syntax of the three-letter code defining the management modes for these versors is as follows:

(VER, Ijk, Mno, Pqd, Uvw) *example: (VER , REL , PRG , PRG)*

(VER , ROT, , REL , PRG)

Parameter	Type	Values	Description
Ijk	Character	PRG or REL Obligatory	Defines whether the "Tool Direction" vector is programmed directly by means of the ijk components or whether it is to be calculated on the basis of the position of the rotary axes. When the versor is not necessary, this setting is ignored. **PRG** The versor is programmed using the ijk components. **REL** The versor is to be determined from the programmed position of the rotary axes. **ROT** The versor is to be determined from the current position of the rotary axes.
Mno	Character	PRG or REL Optional	Defines whether the "Normal to Surface" vector is programmed directly using the mno components or whether it is to be calculated automatically by the system. If the versor is not necessary, this setting is ignored. **PRG** The versor is programmed using the mno components. **REL** The versor is to be determined automatically by the system. If omitted, this means that a uvw vector is used.
Pqd	Character	PRG or REL Obligatory	Defines whether the "Tool Radius Application Direction" vector is programmed directly using the pqd components or whether it is to be calculated automatically by the system. When the versor is not necessary, this setting is ignored. **PRG** The versor is programmed using the pqd components. **REL** The versor is to be determined automatically by the system.

Uvw	Character	PRG or REL Optional	Indicates whether the “Tool Radius Compensation Factor” has been programmed directly through the uvw components or has to be calculated by the system automatically. When the versor is not required, the programming mode is disregarded. **PRG** The vector is programmed through the uvw components. **REL** The vector has to be calculated automatically by the system. If omitted, this means that an mno versor is used.

NOTE:
if tool radius compensation is required, vectors MNO and UVW cannot both be omitted.

Look Ahead management

The syntax of the three-letter code that defines the method of look ahead and dynamics management is as follows:

(JRK, Enable, Jrs, Entities, Tstab, Tintgr,Ttop) *example: (JRK, ENA,,)*

Parameter	Type	Values	Description
Enable	Character	ENA,DIS or AXI Obligatory	Defines the type of dynamics to be used. **ENA** Enables the use of Jerk Limitation (Mov=8). The specified jerk refers to the dynamics described on the profile. **DIS** Disables Jerk Limitation, only uses “S ramps” (Mov=2). **AXI** Enables the use of Jerk Limitation (Mov=8). The specified jerk refers to the dynamics described by the axes and how this affects the dynamics of the profile. The use of this method is closely associated with the quality of the points entered (number of decimals and distribution).
Jrs	Number	Taken from JRS variable Optional	Modifies the value of the JRS variable of the system only for the G61 section being processed; if the value is omitted, the JRS variable active in the system is used.
Entities	Number	Optional	The High Speed algorithm uses a dynamic queue of 64 elements (polynomial curves) and starts movement after calculating the 64th. The start may be adjusted by programming the number of elements to be calculated before starting movement. 0 means maximum number of elements.
Tstab	Number	Optional	Defines the time window (in ms) within which the speed smoothing algorithm is to be applied. This algorithm removes unnecessary accelerations so as to avoid machine oscillations. (See Figure 1)

Parameter	Type	Values	Description
Tintgr	Number	Optional	Defines the maximum time (in ms) for integrating the acceleration (or deceleration) phases with the sections at a constant feed rate. (See Figure 1)
Ttop	Number	Optional	Defines the minimum time (in ms) for executing a section with a constant feed rate V. (See Figure 1)

Thresholds

The syntax of the three-letter code that defines the value of thresholds used in generating the polynomial curves is as follows:

(THR, Tol, TolV, Scale, NullMov, Chord, ErrRot, MaxLen)
example: (THR , 0.01 , 0.0001 , 0 , 0 , 0.1)

Parameter	Type	Values	Description
Tol	Number	Obligatory	Defines the tolerance range to be used in generating the polynomial curves. As defined previously, the polynomial curves generated pass through the programmed points, within a given tolerance threshold. This parameter defines the tolerance value. It is defined in mm (or inches if the machine is configured in inches).
TolV	Number	Obligatory	Defines the tolerance range to be used in generating the polynomial curves calculated on the versors. This value is important because the precision with which the rotary axes are positioned depends on the precision with which the curves are generated on the versors (in particular, on the ijk versor). The number has no dimension in that it refers to entities like versors which always have a dimension of 1. We suggest a value equal to 0.1 times the Toll value defined previously be set. 0 disables the management of a tolerance on the versors.
Scale	Number	Obligatory	Scale factor to be applied to the programmed axes. If 0 is set, the scale factor is not to be applied. It is not applied to the versors and rotary axes, when programmed.
NullMov	Number	Obligatory	Null movement threshold. If the distance between one point and the next is less than this threshold in mm (or inches if the machine is configured in inches) the next point is eliminated.
Chord	Number	Obligatory	Maximum allowed chordal error at input and output during the generation of a polynomial. Value expressed in mm (or inches if the machine is configured in inches).
ErrRot	Number	Optional	Maximum admissible chordal error for the determination of polynomials on rotary axes. It is expressed in degrees.

Parameter	Type	Values	Description
MaxLen	Number	Optional	Maximum length of generated polynomials. Value expressed in mm (or inches). It allows a major control of the position tolerance. (Toll) Advised values >5 mm, lower values could slow the working.

Tool definition

In the High Speed Machining system, “Cylindrical”, “Ball-ended” and “Toroidal” type tools may be managed. The syntax of the three-letter code that defines the characteristics of the tool to be used for machining is as follows:

(TOL, Len, Radius, EdgeRad, Angle, OriginLen, Rorig, RgOrig) *example: (TOL , , , 1 , -90 , 0)*

Parameter	Type	Values	Description
Len	Number	Taken from the offset active on G61 Optional	Defines the length of the tool to be used for tool length compensation. If no value is set, the tool length is taken from the offsets active in the system on the activation of the G61. The value set here is only active during the machining of the current section of G61/G60.
Radius	Number	Taken from the offset active on G61 Optional	Defines the radius of the tool to be used for tool radius compensation. If no value is set, the tool radius is taken from the offsets active in the system on the activation of G61. The value set here is only active during the machining of the current section of G61/G60.
EdgeRad	Number	Obligatory	Defines the radius on the edge of the tool. As this value is not handled by the system offsets, it has to be set.
Angle	Number	Optional	Defines the angle of contact between the tool and the part. It is used in the automatic calculation of the vector normal to the surface.
OriginLen	Number	Optional	Defines the length of the tool for which the part program was generated. This field is used when the points entered refer to “Axis location points” and you wish to apply the compensation for tool length.
ROrig	Number	Optional	Defines the radius of the tool for which the part program was generated. This field is used when the points entered refer to “Axis location points” and you wish to apply the compensation for tool length and diameter.
RgOrig	Number	Optional	Defines the radius of the tool for which the part program was generated. This field is used when the points entered refer to “Axis location points” and you wish to apply the compensation for tool length and diameter.

Tool direction (3D)

Defines the tool direction (to be used for compensation purposes) for generic machines or machines with 3 axes (it is not required on machines with 5 axes). In practice, the unitary vector (similar to ijk) which identifies the tool direction in the part reference system has to be defined.
The syntax of the three-letter code that defines the direction of the tool to be used for machining is as follows:

(TOD, Xcomp, Ycomp, Zcomp) *example: (TOD , 0 , 0 , 1)*

Parameter	Type	Values	Description
Xcomp	Number	Obligatory	Component of the tool direction along the X axis.
Ycomp	Number	Obligatory	Component of the tool direction along the Y axis.
Zcomp	Number	Obligatory	Component of the tool direction along the Z axis.

Change in curvature management

The algorithm used by the CAM systems to generate the points of a profile takes into account the “chordal error”, that is, it generates points closer together when the radius of curvature of the profile to be described becomes more accentuated. A change in curvature may therefore be generically identified by a variation in the distance between one set of points and the following ones. This variation in curvature may be identified automatically by the High Speed Machining algorithms so as to avoid oscillations around the point where this change takes place.

Supposing, for example, we have a curved section followed by a straight section. The spline on the straight section would tend to oscillate or generate a “hump”, so it is important to identify it.

At the "curvature change" point, the system will automatically introduce a G62 or a G63 depending on user requests. As an alternative, it is possible to add more points (closer spacing) in the longer portion; the number of points added depends on the ratio between the lengths of the two portions (long/short).

The syntax of the three-letter code that defines how to identify and then manage the change in curvature is as follows:

(CRV, Len, Ratio, Mode) *example: (CRV , 1 , 6 , G63)*

Parameter	Type	Values	Description
Len	Number	Obligatory	Defines the minimum length of the “long section” for the change in curvature to be managed. It is defined in mm (or inches if the machine is configured in inches).
Ratio	Number	Obligatory	Defines the ratio between the long section and short section so that the change in curvature may be identified. For example, the value 6 defines that the distance between p2 and p3 must be greater than 6 times the distance between p1 and p2 for the change in curvature to be activated.
Mode	Character	G62 or G63 or PNT Obligatory	Defines the transition code to be set on the “change in curvature” point **G62** The two segments are generated with two splines tangential to one another, so the second spline will have an initial inclination equal to the final inclination of the first spline. **G63** The two segments are generated by two non-tangential splines, but are linked on the basis of the calculation tolerance set. **PNT** Additional points (closely spaced) are added on the longer portion.

Edge management

The automatic identification of edges is important for the same reason as the identification of changes in curvature. Failure to identify edges could generated incorrect oscillations on the splines. The optimum dynamic approach for handling an edge is to stop at zero speed and then restart on the next section. Stopping may be damaging however as the tool, as it turns, continues to remove material and so some “notches” may be visible on the part. For this reason, it is possible to define whether and how to stop at the edge.

The syntax of the three-letter code that defines how to identify and therefore how to manage the presence of edges is as follows:

(EDG, Angle, VAngle, Mode, Acc) *example: (EDG , 30 , 0 , G66 , 1)*

Parameter	Type	Values	Description
Angle	Number	Obligatory	The system is capable of automatically determining the edges defined by the programmed points (edges on the linear axes) and automatically programming a G66. This value, expressed in degrees, defines the threshold angle beyond which a point is defined an “edge” point. See the figure below.
VAngle	Number	Obligatory	It is similar to the previous value and is to be used for the versors. The presence of an edge on the versors generates an “edge” in the movement of the rotary axes. It is therefore advisable to add a G66 also in these cases. See the figure below.
Mode	Character	G63, G66 or G67 Obligatory	Defines the transition code to be set on the edge **G63** The edge is executed by inserting a link which is generated by taking the configured chordal error into account. **G66** Movement ends at zero speed. **G67** Movement ends at the maximum speed at which the edge may be faced in the best possible way.
Acc	Number	Obligatory	Acceleration that may be withstood by the axes in tackling the edge: 1 means that an axis may withstand an increase in speed equal to 1 acceleration. It only applies to G67.

Axis definition

The axes to be subjected to the High Speed Machining algorithms may be defined. A maximum of 6 axes may be defined, of which 3 axes make up the three Cartesian axes, 2 are rotary axes (for machines with 5 axes) and other additional axes. The syntax of the three-letter code for defining the axes is as follows:

(AXI, Name, Id, Type , CinType, Diam, LinRot) *example: (AXI , X , 1 , ABS , LI1, DIS)*

example: (AXI , C, 4, OTH, OTH, LIN)

Parameter	Type	Values	Description
Name	Character	Obligatory	Defines the name of the axis. This name may be different from the one set in AMP for the axis, and may therefore be compared to an implicit "DIN" in the HSM setup. This association only applies to the G61/G60 section being machined.
Id	Number	Obligatory	Defines the ID of the axis. This axis must be under the control of the Process in which the part program is being executed.
Type	Character	ABS, ORD, VRT, OTH TAN or TAO Obligatory	Defines the axis type, and influences the method by which the axis is managed by the HSM algorithms. **ABS** The axis is of the x-coordinate type (in the Cartesian coordinate system) **ORD** The axis is of the y-coordinate type (in the Cartesian coordinate system) **VRT** The axis is of the z-coordinate type (in the Cartesian coordinate system) **TAN** Tangent axis, calculated automatically by the system **TAO** Tangent axis, calculated automatically by the system, with the axis moving (positive direction) clockwise, and hence in a direction opposite to that of the trigonometric convention **OTH** Other type of axis, different from the previous ones.

CinType	Character	LI1, LI2, LI3, WRK, TOL or OTH Obligatory	Defines the position of the axis in the kinematic chain of the machine (see following sections). For generic machines or machines with 3 axes for which the kinematic chain does not have to be defined, we recommend the following setup: **LI1** To be associated with the axis defined as ABS (x-coordinate) in the previous field. **LI2** To be associated with the axis defined as ORD (y-coordinate) in the previous field. **LI3** To be associated with the axis defined as VRT (z-coordinate) in the previous field. **OTH** Additional axis. For machines with 5 axes, the setup values are as follows: **LI1** First axis in the kinematic chain. **LI2** Second axis in the kinematic chain. **LI3** Third axis in the kinematic chain. **WRK** Rotary axis closest to the part. **TOL** Rotary axis closest to the tool. **OTH** Additional axis.
Diam	Character	ENA or DIS Optional	Makes it possible to characterise the axis as Diametral. If this is not defined, the corresponding parameter currently active in the system is used. **ENA** Axis defined as diametral. **DIS** Normal axis.
LinRot	Character	LIN or ROT Optional on CN Required on Path Optimizer	Makes it possible to specify the type of added axis, as either linear or rotary. If no value is set, for an execution on the CN, the corresponding parameter currently active in the system is used; for an execution on the Path Optimizer, an error is notified. **LIN** Linear type added axis. **ROT** Rotary type added axis.

Axis parameters

Some characteristics of the axis set on the system may be varied. These variations are only active in the G61/G60 section being machined. The syntax of the three-letter code is as follows:

(AXP, Name, NullMov, Pitch, Lim-, Lim+) *example: (AXP , X , 0 , , ,)*

Parameter	Type	Values	Description
Name	Character	Obligatory	Axis name as defined in the three-letter code (AXI).
NullMov	Number	Obligatory	Defines the null movement for the axis in question. If the position programmed for an axis differs from the current position (or last programmed position) for a value lower than the null movement, the movement of the axis (the new position) is not considered. We recommend it be left set to 0 and activated only in cases in which the part program under execution contains imprecisions in the programming of the axes.
Pitch	Number	Taken from the system Optional	Used for redefining the rollover pitch for the axis in question. It may be used, for example, to program a non-rollover rotary axis with rollover values and vice versa.
Lim-	Number	Taken from the system Optional	Used for redefining the lower operating limit for the axis in question.
Lim+	Number	Taken from the system Optional	Used for redefining the upper operating limit for the axis in question.

Axis dynamics

Some dynamic characteristics of the axis set in the system may be varied. These variations are only active in the G61/G60 section being machined. The syntax of the three-letter code is as follows:

(DIN,Name, Vmax, Amax, Jmax, Vrap, Arap, Jrap) *example: (DIN , X , , , , , ,)*

Parameter	Type	Values	Description
Name	Character	Obligatory	Axis name as defined in the three-letter code (AXI).
Vmax	Number	Taken from the system Optional	Used for redefining the maximum speed at which the axis may be moved when a machining operation in G01 is in progress. It must be defined in mm/min.
Amax	Number	Taken from the system Optional	Used for redefining the maximum acceleration at which the axis may be moved when a machining operation in G01 is in progress. It must be defined in mm/sec^2.
Jmax	Number	Taken from the system Optional	Used for redefining the maximum Jerk to which the axis may be moved when a machining operation in G01 is in progress. It must be defined in mm/sec^3.
Vrap	Number	Taken from the system Optional	Used for redefining the maximum speed at which the axis may be moved when a machining operation in G00 is in progress. It must be defined in mm/min.
Arap	Number	Taken from the system Optional	Used for redefining the maximum acceleration at which the axis may be moved when a machining operation in G00 is in progress. It must be defined in mm/sec^2.
Jrap	Number	Taken from the system Optional	Used for redefining the maximum Jerk at which the axis may be moved when a machining operation in G00 is in progress. It must be defined in mm/sec^3.

Example

Setup for machine with 3 axes:

```
(PNT , AXI , AXI , QUI )
(VER , REL , PRG , PRG )
(JRK, ENA ,  ,  ,400 ,150 ,200)
(THR , 0.01 , 0.0001 , 0 , 0.001 , 0.05 )
(TOL ,  ,  , 0 , -90 ,  )
(TOD , 0 , 0 , 1 )
(CRV , 1 , 6 , G63 )
(EDG , 30 , 0 , G66 , 1 )
;
(AXI , X , 1 , ABS , LI1 )
(AXP, X , 0 ,  ,  ,  )
(DIN , X ,  ,  ,  ,  ,  ,  )
;
;
(AXI ,Y, 2 , ORD, LI2 )
(AXP,Y, 0 ,  ,  ,  )
(DIN ,Y,  ,  ,  ,  ,  ,  )
;
;
(AXI , Z , 3 , VRT, LI3 )
(AXP, Z , 0 ,  ,  ,  )
(DIN , Z ,  ,  ,  ,  ,  ,  )
;
```

END OF CHAPTER

MULTIPROCESS MANAGEMENT COMMANDS

GENERAL

This chapter discusses interprocess commands, i.e. instructions that allow the programmer to synchronise parallel part program execution various processes or to shift the "axes" resource between processes in order to meet specific requirements.

The following table summarises these commands:

3-LETTERS CODE	FUNCTION
DCC	Defines remote channels
PVS	Defines the remote PLUS variables environment
PRO	Specifies the default process to which synchronisation codes and commands must be sent
SND	Sends a synchronisation message to the specified process
WAI	Puts the current process on wait until another message arrives from another process
EXE	Activates automatic part program execution by the specified process
ECM	Executes an MDI block in the specified process
GTA	Shifts the "axes" resource among processes

IMPORTANT

By default SND and WAI are synchronised.

SYNCHRONIZATION AMONG PROCESSES

Notes On The "Wait" Function:

The synchronisation messages received by a process are recorded in a memory area called MESSAGE QUEUE. This local area is process-dedicated: there is one message queue for each process.

If the (WAI,Pn) instruction is executed by a part program EXE, the process must wait for a message from process <n>. The system looks for the message in the MESSAGE QUEUE:

- ***If the message is not found in the MESSAGE QUEUE,*** the process puts itself on WAIT and suspends part program execution.
- ***If the message is in the MESSAGE QUEUE***, part program execution continues and the message is deleted from the message queue.
- ***If the queue contains a message coming from a process other than <n>,*** the message is not deleted but it is kept aside in order to be utilised when a WAI for the appropriate process is programmed.
  The process can exit from WAIT mode only when the queue contains a message coming from the process specified in WAI.

Notes On The "Send" Function:

Three-letter code SND allows synchronisation messages to be sent to the specified process; the messages may be synchronous or asynchronous:

If the message is synchronous, the process that sent SND goes on WAIT and waits for an ACK (acknowledge to use the message) from the destination process;

If the message is asynchronous, the process that sent SND continues executing the part program without waiting for the message to resume the SND destination process.

IMPORTANT

A process (e.g. process 1) cannot send another process (e.g. process 2) more than one message at a time, at least not until process 2 exits from WAIT and the message is cleared from the queue. However, the sender can send messages to other processes (e.g. process 3).

IMPORTANT

The message queue may be cleared by resetting the process or enabling another part program.

Exchanging data

Commands SND and WAI permit the exchange between processes of data, such as local or system variables, constants, strings.

The data is sent to (SND) or received (WAI) from the processes in question and may be specified in programming of the respective functions; for further details, see the syntax of the two three-letter codes on the following pages.

Resetting synchronised processes

If a reset is made, it must be made both on the processes sending the synchronisation messages (SND) and on the processes receiving the synchronisation messages (WAI).

This is to avoid unwanted messages remaining in the "MESSAGE QUEUE" of the processes, which could lead to malfunctionings.

Channels table

The channels table is a internal system table, containing the identifiers of the logic *channels* on which a process can operate, where *channel* is taken to mean a remote or local environment.

Each process has its own table, each of which contains 40 *channels*. When the system is powered on, it defines by *default* the 20 local processes as the first 20 *channels* of the table, identifying them with numbers 1,2,3,4.....20.

In the event of 10 Series systems connected in a network through the option MINI DNC E65/E66 or higher, other channels may be defined in the table by associating the identifiers (1÷40) with remote environments previously configured with the option MINI DNC.

If a value from 1 to 20 was used as the identifier of a remote environment, the corresponding local process is deleted from the channels table.

DCC - Definition of the communication channel

The three-letter code DCC allows the programmer to define the channels of the table of the process it is programmed in.

Syntax

(DCC ,*channel number***,** *channel name***)**

where:

channel number Number of the channel being defined; may be a number or a local or system variable, of value between 1 and 40.

channel name Name of the remote environment defined with the network configurator of the MINI DNC option.

Characteristics:

The *channel* defined with this command must be configured previously with the MINI DNC option. Among the types of channel that may be defined with MINI DNC, command DCC can operate on two predefined types.

- Type 1 - PROCESS channel:
  A remote process to which all the commands accepted by local processes can be sent.

  Examples:
  (DCC,25,remote_1) ;defines channel n° 25
  (WAI,P25) ;wait for message from channel (remote process) n°25

  (PRO,25)
  (EXE,PROGRAM) ;run part program on remote process n°25

- Type 2 - PLUS channel:
  A channel on which reading and writing of PLUS variables (SN, SC, L, @, $ASSET) can be rerouted with a PVS command.

IMPORTANT

A RESET of the process or activation of a part program resets the channels table:
channel 1 = local process 1
channel 2 = local process 2
channel 3 = local process 3
channel 4 = local process 4

NOTE:

Each command issued must be consistent with the type of channel it is sent to.
For example, if a command accepted from the process channels (e.g. WAI, SND,EXE) is sent to a channel type 2, the system returns an **NC221 Wrong process type** error.

PVS - PLUS channel selection

Three-letter code PVS permits reading and writing of PLUS variables to be rerouted to a type 2 remote channel, previously defined with the DCC command.

Syntax

(PVS ,*channel number***)**

where:

channel number Number of the channel being defined; may be a number or a local or system variable, of value between 1 and 40.

Characteristics:

After programming the PVS command, reading and writing of the PLUS variables (SN, SC, L) and @xxxx, $xxxx (with ASSET option enabled) occurs on the remote channel selected.

To restore local PLUS variable reading, program (PVS,0).

If the channel specified is not type 2 (defined with the network configurator), the system returns error: **NC221 Wrong process type**.

If the channel specified was not defined previously with DCC, the system returns error: **NC220 Process undefined**.

IMPORTANT

A RESET of the process cancels programming of the PVS command (tantamount to programming (PVS,0)).

PRO - Definition of the process

The PRO command allows the programmer to specify the process synchronisation commands refer to. The process number is an optional parameter in all commands of this type.

Syntax

(PRO, *process number***)**

where:

process number It can be a number or a system or local variable in the 1-20 range (1-40 with E65/E66 option). If it is omitted the system will select the default process.

IMPORTANT

A reset of the system or activation of a part program resets the previous definitions of the default process.

SND - Send a synchronisation message

SND allows to send a synchronisation message and data to the specified process.

Syntax

(SND,*[,P process number], S|A [,data] [,data] .....[,data]***)**

where:

process number Destination process number (from 1 to 20, or 1 to 40 with the E65/E66 option). It can be a number, a local or system variable.
If it is the number of the process the SND command is executing, an **NC222 Wrong process number** message will be generated.
If it is omitted, the information will be sent to the default process declared with PRO.

S | A S and A specify the two alternative execution modes:

S (synchronous) the sender or current process goes on WAIT until the destination process has not received the message, cancelled it from the queue and resumed part program execution.

A (asynchronous) part program execution begins immediately after the message has been sent.

data The data (optional and max. 20 items) may be Long Real numbers, strings between quotes, and local or system variables.
The system reads the programmed data and transmits them to the specified process together with the synchronisation message. It may send up to 20 parameters, which may occupy up to 174 bytes.

The length of the various data types is as follows:

Data type	Length (in bytes)
Boolean / Byte	2
Short	3
Long / Real	5
Double (Long Real)	9
String	number of characters + 2

Example:
(SND,P1,S,E1,SN12,"TOOL",SC0.30,@BOOL_PLUS(0),33.6)

The number of bytes transmitted by this block is:

Parameter	Format	Length (in bytes)	
E1	Long Real	9	
SN12	Long Real	9	
"TOOL"	String	6	
SC0.30	String	32	
@BOOL_PLUS(0)	Boolean	2	
33.6	Long Real	9	
	Total:	67	transmitted bytes

The system generates the **NC224 Data sending too long** error if there is an attempt to send more than 174 bytes.

– **Error situations:**

The **NC225 Data loading failed** error message occurs in two cases:

- *Example 1 (SND synchronous mode)*

  Process 1 sends a synchronous (SND,P2,S,E1) to process 2 while it is on WAIT (WAI,P1,SC0.30):

  – Both processes will display the **NC225 DATA LOADING FAILED** error message and their programs will remain on WAIT.

- *Example 2 (SND asynchronous mode)*

  Process 1 sends the asynchronous (SND,P2,S,E1) to process 2, in which a (WAI,P1,SC0.30) is programmed.
  In this case, as the data is not consistent with that of process 2, and the SND is asynchronous, the result is that:

  – process 1 resumes part program execution

  – process 2 displays the **NC 225** error message and its part program is not allowed to start.

WAI - Wait for a synchronisation message

WAI puts the process on WAIT for a synchronisation message sent by a specified process. If the message is available in the message queue, part program execution continues and the receiver responds to the sender task no matter whether the synchronous SND has been executed or not.

Syntax

(WAI,*[,P process number], [,variable] [,variable] .....[,variable]***)**

where:

process number Sender process number (from 1 to 20, or 1 to 40 with the E65/E66 option). It can be a number, a local variable or a system variable.

If it is the number of the process the WAI command is executing, a **NC222 Wrong process number** message will be generated.
If it is omitted, the information will be sent to the default process declared with PRO.

variable List of the variables, maximum 20, in which to receive the alphanumeric values arriving with the synchronisation message. If number or type of the variables programmed is not consistent with those sent, the sender process is sent error message **NC255 Data loading failed** (see also the SND command for further details).

IMPORTANT

When a process is on WAIT and a synchronisation message with data arrives, the ACK requested by the sender is sent after the data have been written in the programmed variables. If data fail to be written because of format or process number errors, a negative ACK is sent to the sender.

The figure below illustrates how SND and WAI work.

Example of using SND and WAI commands

EXE - Automatic part program execution

EXE allows to start execution of a part program in the specified process.

Syntax

(EXE,*part program number[,P process number]***)**

where:

part program name Name of the part program to be executed in AUTO mode. It may be up to 48 characters long. It may be a string of characters (not between quotes) or a character variable preceded by ?.

process number Number of the process (from 1 to 20, or 1 to 40 with E65/E66 option) that runs the program. It can be a number, a local variable or a system variable.
If it is the number of the process that is executing the EXE command, an **NC222 Wrong process number** message will be generated.
If it is omitted, the information will be sent to the default process declared with PRO.

Characteristics:

The EXE command forces execution of the specified process in AUTO mode, and activates and launches the part program (CYCLE ON). Both AUTO and CYCLE ON can be filtered by the machine logic.

If AUTO or CYCLE ON are received by a process that cannot execute them because it is on CYCLE STOP, RUN or HOLD, the **NC227 EXE or ECM failed** error message will be displayed.

Examples:
(EXE, MAIN_PROG ,P3) activates execution of the MAIN_PROG program by process 3
(PRO,2)
SC1.10="MAIN_2"
EXE,?SC1.10) activates execution of the MAIN_2 program by process 2

ECM - Manual block execution in a process

ECM allows to manually execute a block in the specified process.

Syntax

(ECM,*command*[,P *process number*]**)**

where:

command Command code. It may be a string or a string variable between apexes or quotes. Example

process number Number or the process number from 1 to 20 (1 to 40 with E65/E66 option)that runs the program. It can be a number, a local variable or a system variable.
If it is the number of the process executing the EMC command, an **NC222 Wrong process number** message will be generated.
If it is omitted, the information will be sent to the default process declared with PRO.

Characteristics:

The EMC command forces the destination process in MANUAL mode, sends the specified command to it and executes it via CYCLE ON. Both AUTO and CYCLE ON can be filtered by the machine logic.

The ECM command is of the asynchronous type: this means that after executing ECM, the part program continues with the following blocks irrespective of whether the command sent via ECM to another process has been concluded or not.

It is therefore important not to execute another ECM command on the same process until the execution of the previous command has been completed.

If this occurs, the "**NC 227 EXE or ECM command failed**" error is signalled.

If MANUAL or CYCLE ON are received by a process that cannot execute them because it is on CYCLE STOP, RUN or HOLD, the **NC227 EXE or ECM failed** error message will be displayed.

Examples:

(ECM,"E1=12",P2)	Sets to 12 the value of local variable E1 in process 3
(PRO,2)	
SC0.5="E1012"	Prepares the command in variable SC0.5
(ECM,SC0.5)	Executes ECM for a command written in the content of the variable

Example of synchronisation of two process using EXE:

```
;N1 SNA_N                                  ;N1 SNB_N
;N2 MULTIPROCESSO: SINCRONIZZAZIONE        ;N2  MULTIPROCESSO; SINCRONIZZAZIONE
N3 (DIS,"PROGRAMMA SYSTEM 1")              N3 (DIS,"PROGRAMMA SYSTEM 2")
N4 (EXE,SNB_N,P2)                          /N4 (UGS,X,-100,100,Y,-130,150,Z)
N5 (UGS,X,-100,100,Y,-100,100,Z)           N5 SN5=0
N6 T2.2M6                                  N6 T4.4M6
N7 S1500F500M3                        →    N7 (WAI,P1)
N8 XYZ                                     N8 S400F400M3
N9 E25=0                                   N9 XY
N10 G81R-14Z-35                            N10 E25=0
N11 (RPT,6)                                N11 G84R-10Z-22
N12 (ROT,E25)                              N12 (RPT,6)
N13 X-80Y                                  N13 (ROT,E25)
N14 E25=E25+60                             N14 X-80Y
N15 (ERP)                                  N15 E25=E25+60
N16 G80Z50                                 N16 (ERP)
N17 (ROT,0)                                N17 (ROT,0)
N18 (SND,P2,A)                        →    N18 G80ZM5
N19 (CLS,MA)                          →    N19 (WAI,P1)
N20 (SND,P2,S)                        →    N20 GZ40
N21 GZ90                              →    N21 (WAI,P1)
N22 (SND,P2,A)                        →    N22 E8=0
N23 (WOS,SN5>=9)                      ←    N23 T2,2M6
N24 S200F500M3                             N24 S1300F500M3
N25 X-10Y50                                N25 G81R2Z-25
N26 Z-25                                   "D" N26
N27 G1G41X                                 N27 G91X10
N28 G2IJ                                   N28 G90
N29 G1G40X10                               N29 E8=E8+1
N30 GZM5                              ←    N30 #SN5=E8
N31 M30                                    N31 (GTO,D,E8<10)
                                           N32 G80M5
```

SHARED AXES

General

The "shared axes" feature allows one or more processes to acquire or release axes resources. An axis is acquired or released by associating its name (X,Y,Z) to one of the identifiers configured in AMP (from 1 to 32).

- A name cannot be assigned to different axes in a process.
- An identifier cannot be assigned to different machine axes.

Conditions for axis acquisition

Before acquiring an axis check whether the following conditions apply:

- the axis is configured in AMP and the identifier exists in the configured processes;
- the axis has been released from all processes;
- the axis name is specified in the program;
- the identifier (ID) is a number between 1 and 32 and is associated neither to a spindle nor to an auxiliary axis;
- the name to be assigned to the axis is not used for a pseudo axis in a process.

GTA - Axes acquisition

GTA allows one or more axes to shift between processes. In addition, it redefines the axis/offset association and sets the sequence in which axes are displayed.

Syntax

(**GTA,**[*T value*,]*ID_axes_name* [, *ID_axes_name*...] [/] [*axis_1_name*][,*axes_2_name*])

where:

T value	value = 0	loose activate origins (this is the same as omitting the T0 parameter)
	value = 1	maintain active origins on the re-programmed axes (with the same name, ID and order) and on the programmed axes with a different name but the with the same ID and order.
	value = 2:	maintain active origins on all set axes, regardless of the programming origin of the ID's.
/	Is the separator between the axis name and the optional length offset.	
axis_1_name	Are the axes to which length offsets 1 and 2 will be applied the next time an	
axis_2_name	offset is enabled.	

Characteristics:

- In a GTA block you must declare all the controlled axes, including those already belonging to the process.
- If an axis is not specified in a GTA command it will be released and made available to other processes.
- When a GTA command is executed the axes are displayed in the order in which they are specified. GTA can be used for changing this order.
- The RESET command cancels the changes made by GTA.
- At power up axes and processes will be associated as in AMP.
- When the control executes a tool change or enables an offset, the axes to which length offsets are associated must already be controlled by the process.

INHIBITING GTA

Axes acquisition/release with GTA may be inhibited in the following cases:

- when a radius offset is active
- when a canned cycle (G81-89) is active
- when the system is on HOLD or IDLE-MAS
- when a virtual mode is active.
- When the tool correction mode (h or T) is active and you are releasing an axis where the tool correction has been applied or are acquiring a new axis to which the tool correction must be applied.
- Release of axes shared with the logic.

SYSTEM PARAMETERS AFTER GTA

A GTA command permits to initialise the following information:

Interpolation plane

- The interpolation plane is defined by two of the first three axes programmed by the current G (G17, G18, G19). If GTA defines less than three axes, the first and second axes will be forced as abscissa and ordinate.
- To ensure correct operation, the desired plane must be programmed with G16.

Origins

- If the "Tvalue" parameter is omitted or set to "value=0", the active origins will be reset and the RESET ORIGINS logic function is called with the list of axes controlled by the process.
- If the "Tvalue" parameter is set to value=01, the origins will be maintained on the reprogrammed axes with the same name, ID and order and on the programmed axes with a different name but the same ID and order.

Mirror:

- If the “Tvalue” parameter is set as value = 2, the origins will be maintained on all programmed axes, even if their ID’s are not the same.
- Mirroring is reset for all axes

Scale factors:

- Scale factors are reset for all axes

ROT rotation:

- Rotation programmed with a ROT command will be cancelled.

Travely limits:

- Travel limits for all the axes return to configured values.

Axis selection for manual movements:

- After a GTA command, the first displayed axis will be selected for manual move execution.

Length offsets

- The axes identifiers to which length offsets 1 and/or 2 are associated will be programmed as an option on GTA. Default associations defined in AMP will be lost. They will be restored after the control has been switched on again.

INFORMATION RETAINED AFTER A GTA COMMAND

- **Locked axes**
- **Plane rotation activated with through a manual PROGRAM SETUP command**
- **Incremental and provisional origins**

Example 1:
Initial conditions:

- Controlled axes are X (with ID 1) and Y (with ID 2).

Goals:

- Maintain control over axes X and Y (ID 1 and 2)
- Acquire axes associated to ID 5 and 6 and name them Z and A
- Associate length offset 1 to axis Y (ID 2)
- Associate length offset 2 to axis Z (ID 5)
- Obtain the following display order: X, Y, Z, A.

Programming example:

IMPORTANT

Programming all the axes (new and old) is mandatory. It permits to reassign axes names and including those already belonging to the process is mandatory and allows to redefine both the axis names and the display order.

Example 2:

Initial conditions:

– Controlled axes are X (with ID 1) and Y (with ID 2).

Goals:

– Release axes X and Y (ID 1 and 2)

– Acquire axes associated to ID 5 and 6 and name them X and Y

– Associate length offset 1 to axis Y (ID 6)

– Obtain the following display order: X, Y.

Programming example:

Example 3:

Initial conditions:

– An indefinite number of controlled axes

Goals:

– Release all its axes

Programming example:

Example 4:

Initial conditions:

- Controlled axes are X,Y,Z,A,B with ID 1,2,3,4,5.

Goals:

- Retain control of X and Y axes with (ID's 1 and 2)
- Release axes Z, A and B (with ID's 3, 4 and 5)
- Associate length offset 1 to axis Y (ID 6)
- Obtain the following display order: X, Y.

Programming example:

Example 5:

Initial Conditions:

- Controlled axes are X,Y,Z axes with ID 1,2,3.
- To the X,Y,Z axes, the 1 origin is associated.

Goals:

- Retain control of the axes with ID 1 and 2 (X,Y).
- Release the axis Z (ID 3).
- Acquire the P axis with ID 4.
- ssociate length offset 1 with the P axis (ID 4).
- Obtain the following display order: X,Y,P.
- Loose the origin associated to all axes.

Programming example:

(GTA,TO,X1,Y2,P4/P)

- **Name of the axis to associate to the length offset**
- **Acquisition of axis with ID 4 associated to the name P**
- **Axis ID 2 already belonging to the process**
- **Axis ID 1 already belonging to the process**
- **Parameter to loose the origins on all axes**
- **Get axes command**

Example 6:

Initial Conditions:

- Controlled axes are X,Y,Z axes with ID 1,2,3.
- To the X,Y,Z axes, 1 origin is associated.

Goals:

- Retain control of the axes with ID 1 and 2 (X,Y).
- Release the axis Z (ID 3).
- Acquire the P axes with ID 4.
- Associate the length offset 1 with the P axis (ID 4).
- Obtain the following display order: X,Y,P.
- Retain the associated origins with the axes with ID 1 and ID 2.

Programming example:

(GTA,T1,X1,Y2,P4/P)

- **Name of axis to which to associate the length offset**
- **Acquisition of the axis with ID 4 associated to the name P**
- **Axis ID 2 already belonging to the process**
- **Axis ID 1 already belonging to the process**
- **Parameters to retain the origins on the axes**
- **Get Axes Command**

Example 7:

Initial Conditional :

- Controlled axes are the X,Y,Z axes with ID 1,2,3.
- To the X,Y,Z axis 1 origin is associated.

Goals:

- Retain control over the axis with ID 1 (X).
- Retain control over the axis with ID 2 ID 2 (Y) changing the name to (A).
- Release the Z axis (ID 3).
- Acquire the P axis with ID 4.
- Associate the length offset 1 with the P axis (ID 4).
- Obtain the following display order: X,A,P.
- Retain the origin associated with ID 1 and ID 2.

Programming example:

(GTA,T1,X1,A2,P4/P)

- **Name of axis to which to associate the length offset**
- **Acquisition axis with ID 4 associated to the name P**
- **Axis ID 2 already belonging to the process but with a different name**
- **Axis ID 1 already belonging to the process**
- **Parameter to retain the origins on the axes**
- **Get Axes Command**

Example 8:
Initial conditions:

- Controlled axes are X,Y,Z axes with ID 1,2,3.
- Origin 1 is associated with axes X,Y,Z.
- A 2nd process retains axes A, B with ID 4, 5, with which an origin 2 is associated.

Goals:

- Retain control over axes with ID 1, 2 and 4
- Release axis Z (ID 3).
- Acquire axis B, with ID 4, previously released by process 2.
- Associate length offset 1 with axis B (ID 4).
- Obtain the following display order (on system video): Y,X,B.
- Maintain the origins associated with all the axes, i.e. origin 1 for axes with ID's 1 (Y), 2 (X) and origin 2 for axis with ID 4 (B).

Programming example:

(GTA) on process 2 for the release of A and B

(GTA,T2,Y1,X2,B4/B)

- **Name of axis to be associated with length offset**
- **Acquisition of axis with ID 4 associated with name B**
- **Axis ID 2 already belonging to the process, but under a different name**
- **Axis ID 1 already belonging to the process, but under a different name**
- **Parameter to retain origins of axes**
- **Get Axes command**

Example 9:
Initial conditions:

- Controlled axes are X,Y,Z axes with ID 1,2,3,4.
- Axis with ID 4 is “shared with the logic”.

Goal:

- Exchanging the abscissa with the ordinate.

Programming example:

(GTA,X2,Y1, Z3, W4)

NOTE:

An axis shared with the logic cannot be released through a GTA command (the system gives out error message NC124) and, needless to say, since it belongs to the logic, even though it is visible from the process environment, it cannot be programmed.
Hence, if you want to use the GTA command on a process involving one or more axes shared with the logic, you must first reconfirm such shared axes as described in the example.

Error Management

If an error such as axis resource not available, ID axis not found, etc. occurs during part program execution, the programmer may stop the program and display the error or manage it from the program through the ERR and STE variables.

For further information about these variables, refer to Appendix C.

Example of axes acquisition from a process:

```
;N1 SINUS_AM
;N2 ESECUZIONE SU PROC.1
;N3 ACQUISIZIONE ID9-ID10 (NUOVI ASSI A-B) DA PROC. 2
N6 (EXE,GTA1_AM,P2)
N7 (DLY,0.1)
N8 (GTA,X1,Y2,Z3,A9,B10,-Z)
N9 T1.1M6
N10 S1000M3
N11 G0G90XYZAB
N12 G1G91G28
/N13 (UGS,1,Z,-80,80,Y,-50,50,Z)
N14 E25=0.03
N15 X0.2713Y0.0947Z-0.1852A-0.1815B0.1007tE25
N16  X0.2711Y0.0886Z-0.1887A-0.1776B0.1066tE25
N17 X0.2708Y0.0825Z-0.1922A-0.1736B0.1125tE25
N18 X0.2703Y0.0764Z-0.1955A-0.1696B0.1184tE25
N19 X0.2697Y0.0702Z-0.1987A-0.1654B0.1241tE25
N20 X0.2689Y0.0640Z-0.2017A-0.1610B0.1299tE25
.................................
.................................
.................................
N2010 X0.2697Y0.1241Z-0.1654A-0.1987B0.0702tE25
N2011 X0.2703Y0.1184Z-0.1696A-0.1955B0.0764tE25
N2012 X0.2708Y0.1125Z-0.1736A-0.1922B0.0825tE25
N2013 X0.2711Y0.1066Z-0.1776A-0.1887B0.0886tE25
N2014 X0.2713Y0.1007Z-0.1815A-0.1852B0.0947tE25
N2015 G0
N2016 (SND,P2,S)
N2017 (GTA,X1,Y2,Z3)
N2018 M30
```

END OF CHAPTER

Chapter 15

HIGH LEVEL GEOMETRIC PROGRAMMING (GTL)

10 Series CNC systems permit to program a geometric profile on the plane using either the standard programming language (G1-G2-G3) or GTL, a high level programming language.

GTL makes it possible to use the information provided by the drawing in order to program a profile made up of straight lines and circles. The 10 Series CNC calculates tangency and intersection points between these elements.

GTL and standard programming codes can be used in the same program but not in the same profile. GTL requires absolute programming mode (G90).

ORIENTED GEOMETRY

With GTL a profile definition is based on four geometric elements:

- reference origins
- points
- straight lines
- circles

A profile is defined not only by its geometric elements but also by a direction of motion. GTL defines the profile elements according to an oriented geometry. For each element you must provide its coordinates on the plane as well a direction.

For example, through points A and B pass two straight lines: l', which moves from A to B, and l, which moves from B to A.

Straight lines l and l'

In an oriented geometry l and l' are two different lines.

GTL programming, which is based on oriented geometry, requires the assignment of an assumed direction for each line

IMPORTANT

By convention the direction of a line is given by the angle formed between the line and the positive X semiaxis.

To obtain the angle rotate the positive X semiaxis until it has the direction of the straight line to be defined. The angle will be positive if the positive X semiaxis rotates counterclockwise; it will be negative if the positive X semiaxis rotates clockwise.

Conventional angle signs

IMPORTANT

Circle definitions must also include a direction of motion. By convention, a counterclockwise motion defines a positve circle and a clockwise motion defines a negative circle.

Errore. Non si possono creare oggetti dalla modifica di codici di campo.

Clockwise and counterclockwise circles

The radius of a positive circle has a positive sign whereas the radius of a negative circle has a negative sign.

Circle radius signs

If necessary you can reassign the direction of an element during profile definition.

DEFINING GEOMETRIC ELEMENTS

GTL uses lower case letters a-l-c-d-m-o-r-p-s-b to define, respectively, angles, straight lines, circles, distances, modules. reference origins, radiuses, points, intersection numbers, and bevels.

Lower cases are mandatory, because the NC language uses the same letters in capitals to designate other parameters.

Geometric elements must be saved in the program before defining the profile.

GTL straight lines, circles, points, reference origins are geometric variables identified by a NAME and an INDEX. The element must be defined in an assignment block.

The block format is as follows:

NAME INDEX = <Expression>

where

NAME is one of the symbols for geometric element:

- **o** defines reference origins
- **p** defines points
- **l** defines straight lines
- **c** defines circles

INDEX defines the number of the **NAME** geometric variable (element).
It is a number ranging from 0 to the maximum configured limit.

expression provides all the information that defines the geometric element.

Geometric elements may be defined:

- **directly (explicitly)**, by writing in a block all the necessary information about the geometric element;
- **indirectly (implicitly)**, by writing in a block other geometric elements specified in previous definitions.

Examples of element definitions:

```
o1  =  X30 Y30 a45
p1  =  o1 X15 Y15
p2  =  X60 Y30
l1  =  p1, p2
l2  =  X30 Y50, a45
c1  =  l1,l2,r15
l3  =  X0 Y0, X100 Y60
p3  =  l3, c1
c2  =  p3,r8
................
```

The maximum number of geometric elements must be specified in the system configuration.

The syntax of a geometric definition includes a comma (",") as separator between two geometric elements (straight line - point - circle) or between a geometric element and the subsequent parameter ("r" radius or "a" angle).

Examples:

p1 = X30 Y30 the separator must be omitted
c1 = I10 J20 r30

I1 = X20 Y20, X100 Y-10
point point

I2 = I30 J30 r10, X80 Y80
circle point

I3 = X100 Y100, a45
point angle

p5 = c1, c2
circle
circle

c3 = I1, I2, r18
radius
straight line
straight line

s2 is a discriminator that selects the second intersection (s2).

Intersection points between straight lines and circles

p1 = c1, I3
p2 = c1, I3, s2

GEOMETRIC DEFINITIONS

Reference origin	on = X.. Y.. a..
Points	pn = [on] X.. Y.. pn = [on] m.. a.. pn = ln, ln' pn = [-] ln, cn [,s2] pn = cn, [-]ln [.s2] pn = cn, cn [,s2]
Straight lines	ln = [on] X.. Y.., [on] X.. Y.. ln = [on] X.. Y , a.. ln = [on] I.. J.. r.., [on] I.. J.. r.. ln = [on] I.. J.. r.., a.. ln = [on] I.. J.. r.., [on'] X.. Y.. ln = [on] X.. Y.., [on'] I.. J.. r.. ln = pn, pn' ln = pn, a.. ln = [-]cn, [-]cn' ln = [-]cn, a.. ln = [-]cn, pn' ln = pn, [-]cn' ln = [-]ln, d..
Circles	cn = [on] I.. J.. r.. cn = [on] m.. a.. r.. cn = [-]ln, [-]ln, r.. cn = [-]ln, [-]cn, r.. cn = [-]cn, [-]ln, r.. cn = pn, [-]ln, r.. cn = [-]ln, pn, r.. cn = [-]cn, [-]cn', r.. cn = pn, [-]cn, r.. cn = [-]cn, pn, r.. cn = pn, pn', r.. cn = pn, [-]ln cn = pn, [-]cp [,s2] cn = pn, pn', pn" cn = pn, r.. cn = [-]cn, [-]d..

NOTE:
Two dots in a definition idicate that a numerical value is mandatory. Parameters enclosed between square brackets [ ] are optional and may be omitted.

DEFINITION OF A REFERENCE ORIGIN

Function GTL permits to define reference origins using a direct (explicit) format.

Description The information provided by the program typically describes one system of axes that correspond to the machine axes.
For specific design purposes, however, the part may have been drawn according to other cartesian systems, such as the absolute system or other origins obtained by translating or rotating the axes.
The GTL geometry may be defined for any system of axes.

Direct format:

on = X.. Y.. a..

where

on identifies the name of the reference origin

X.. Y.. coordinates of the new origin

a.. rotation angle (positive if counterclockwise).

Example:

o4 = X20 Y30 a-45

DEFINITION OF POINTS

Function GTL permits to define points both directly (explicitly) or indirectly (implicitly). The definition may be written in cartesian or in polar coordinates.

Description The polar reference system consists of an origin (pole) and an X semiaxis also known as polar axis.

Polar axis

Each point on the plane (P) may be defined by its module (i.e. the length of the segment through the point and P) and the value of the angle formed by the polar axis and the P-pole segment.

Polar coordinates

Direct format

Point in cartesian coordinates

pn = [on] X.. Y..

Point in polar coordinates

pn = [on] m.. a..

Indirect format

Intersection point between two predefined straight lines

pn = ln, ln'

Intersection point between a predefined straight line and a predefined circle

pn = [-]ln, cn' [,s2]

pn = cn, [-]ln [,s2]

Intersection point between two circles

pn = cn, cn' [,s2]

where

pn	is the name of a point whose index (n) is a number ranging from 1 and the configured maximum.
X.. Y..	coordinates of the point
[on]	defined reference origin to whose n index refer the X and Y coordinates
m..	polar vector module
a..	polar vector angle
cn **cn'**	predefined circles with indexes n and n'
[-]ln **[-]ln**	predefined straight lines with indexes n and n'. The direction may be inverted by writing a "-" sign
[,s2]	second intersection discriminator

NOTE:

In intersections between straight lines and circles, there are always two possible intersection points. The c1 circle and the l2 straight line intersect in points p1 and p2. If you move along the l2 straight line, the first intersection point is p1 and the second intersection point is p2 . To select the second intersection (p2) use the s2 discriminator. If s2 is omitted, the default intersection point will be the first one (p1).

Intersection between a straight line and a circle

In circle-circle intersections there are also two possible solutions. The intersection points between c1 and c2 are p1 and p2. Take the oriented straight line through the centers of the first and the second circles. This line devides the plane into two halfplanes. To select a point from the right hand halfplane (p2) write the s2 discriminator. If s2 is omitted, the default selection will be the point from the left hand halfplane (p1).

Circle-circle intersection.

Point in cartesian coordinates

p1 = X30 Y160

Point in cartesian coordinates

o1 = X30 Y20 a-20
p5 = o1 X20 Y10

Point in polar coordinates

p2 = m55 a60

Intersection point between two predefined straight lines

p1 = l1, l2

Intersection points between a straight line and a circle

p1 = l4, c3
p2 = l4, c3, s2
p1 = -l4, c3, s2

Intersection points between two circles

p1 = c1, c2
p2 = c1, c2, s2
p1 = c2, c1, s2

DEFINITION OF STRAIGHT LINES

Function GTL permits to define straight lines both directly (explicitly) or indirectly (implicitly). The definition may be written in cartesian or in polar coordinates.

Description The straight line is directed from the first to the second defined element. If the straight line is tangent to a circle there are two possible solutions because the straight line may be tangent to either side of the circle. To select one solution check that the circle and the straight line have the same direction in the tangency point.

Incompatible directions between two geometric elements

Compatible directions between two geometric elements

Direct format

Straight line through two points

ln = [on] X.. Y.., [on'] X.. Y..

Straight line through a point that forms an angle with the abscissa axis:

ln = [on] X.. Y.., a..

Straight line tangent to a circle that forms an angle with the abscissa axis:

ln = [on] I.. J.. r.., a..

Straight line tangent to two circles:

ln = [on] I.. J.. r.., [on'] I.. J.. r..

Straight line tangent to a circle and through a point:

ln = [on] I.. J.. r.., [on] X..Y..
ln = [on] X.. Y.., [on] I.. J.. r..

Indirect format

Straight line through two points:

ln = pn, pn'

Straight line through a point that forms an angle with the abscissa axis:

ln = pn, a..

Straight line tangent to two circles:

ln = [-]cn, [-]cn'

Straight line tangent to a circle that forms an angle with the abscissa axis:

ln = [-]cn, a..

Straight line tangent to a circle and through a point:

ln = [-] cn, pn
ln = pn, [-]cn

Straight line parallel to a straight line at d distance:

ln = [-]ln, d..
ln = [-]cn, a..

where

ln	identifies the name of the straight line whose index n is a number from 0 through the configured maximum
X.. Y..	point coordinates
a..	angle formed by the abscissa axis and the straight line (positive if counterclockwise)
r..	circle radius (positive if counterclockwise)
pn pn'	predefined points with indexes n and n'
[-]cn **[-]cn'**	predefined circles with indexes n and n'. To ensure compatibility between the direction of the straight line and that of the circle in the tangency point, you may write the - sign to change the direction of the circle.
[-]ln	predefined straight line with n index
d..	distance between two straight lines. It is positive if looking in the direction of the predefined straight line the straight line is on the left hand side. It is negative if looking in the direction of the predefined straight line the straight line is on the right hand side.

Straight line through two points

I1 = X40 Y20, X60 Y70

Straight line tangent to a circle and passing through a point

I1 = X10 Y15,I45 J30 r-15
I2 = X10 Y15,I45 J30 r15

Straight line through a point and forming an angle with the abscissa axis

l2 = X50 Y60, a-20

Straight line through a point and forming an angle with the abscissa axis

l5 = o1 X25 Y30, a60

Straight line tangent to a circle and forming an angle with the abscissa axis

I1 = I60 J80 r20, a45

Straight line tangent to a circle and forming an angle with the abscissa axis

I4 = o3 I25 J15 r10, a115

Straight line tangent to two circles

I3 = I25 J35 r-17,I70 J20 r13

Straight line tangent to two circles

I4 = I25 J35 r17, I70 J20 r13

Straight line through two points

l9 = p7, p8

Straight line tangent to a circle and passing through a point

l1 = p1, c1
l2 = p1,-c1

Straight line tangent to two circles

l3 = c1,c2

Straight line tangent to two circles

l4 = -c1,c2

Straight line through a point and forming an angle with the abscissa axis

l3 = p1,a30

Straight line tangent to a circle and forming an angle with the abscissa axis

l1 = c1,a50
l2 = -c1,a50

Straight line parallel to another straight lines

I2 = I1,d20
I3 = I1,d-15

Straight line parallel to another straight line

I2 = -I1,d-50

DEFINITION OF CIRCLES

Faction GTL permits to define circles both directly (explicitly) or indirectly (implicitly).

Description When defining circles indirectly, the programmer must check the compatibility between the directions of the circles and write a "-" sign to alter the direction of a predefined element. Given a circle of known radius and a straight line (and regardless of their directions), there are from 1 to 8 solutions to a circle tangent to two elements, as shown in the figure below:

Circles tangent to a straight line and a circle

After checking that the directions of the predefined elements are compatible with that of the circle to be defined, there are two possible solutions.

To select one of the two possible circles (both of which have the same radius and the same direction), it is necessary to check the direction of the elements specified in the definition and the two arcs in which the resulting circle will be divided by the points of tangency with the predefined elements. GTL always generates a circle that goes from the first to the second arc corresponding to the smallest central angle, as shown in the figure below:

Tangent circles with the smallest ceangle

The c3 circle can be obtained by writing in the definition first the l1 straight line and then the c2 circle, because c3 allows to move from l1 to c2 (whose arc corresponds to the smallest angle).

The c4 circle can be obtained by writing in the definition first the c2 circle and then the l1 straight line, because c4 allows to move from the c2 (whose arc corresponds to the smallest angle) to l1.

The same criteria must be applied when programming a circle tangent to two predefined circles. Also in this case there are from 1 to 8 possible solutions, as shown in the figure below.

Circles tangent to two predefined circles

Circles tangent to two predefined circles

After checking compatibility of directions between predefined circles and programmed circles, there are only two possible solutions left.

To select one of them, consider the two arcs into which the new circle is divided.

To obtain c3, program first c1 and then c2.

To obtain c4, program first c2 and then c1.

Direct format

Circle given the cartesian coordinates of center and radius:

cn = [on] I.. J.. r..

Circle given the polar coordinates of center and radius:

cn = [on] m.. a.. r..

Indirect

Circle of known radius tangent to two predefined straight lines:

cn = [-] ln, ln, r..

Circle tangent to a straight line and predefined circle of given radius:

cn = [-]ln,[-]cn,r..

cn = [-]cn,[-]ln,r..

Circle of known radius through a predefined point, tangent to a predefined straight line:

cn = pn,[-]ln,r..

cn = [-]ln,pn,r

Circle of known radius tangent to a two predefined circles:

cn = [-]cn,[-]cn',r

Circle of known radius through a predefined point, tangent to a predefined circle:

cn = pn,[-]cn,r..

cn = [-]cn,pn,r..

Circle of known radius through two predefined points:

cn = pn,pn',r..

Circle whose center is a predefined point tangent to a predefined straight line:

cn = pn, [-]ln'

Circle whose center is a predefined point tangent to a predefined circle:

cn = pn,[-]cn [,s2]

Circle through three points:

cn = pn,pn',pn''

Circle of known radius whose center is a point:

cn = pn,r..

Circle concentric to a predefined circle and lying at a known distance from it:

cn = [-]cn,d..

where

cn	is a circle whose n index is a number ranging from 0 to the configured maximum.
I..J..	circle center coordinates: I=abscissa, J=ordinate.
r..	circle radius (positive if counterclockwise, negative if clockwise).
[-]ln **[-]ln'**	predefined straight lines whose indexes are n and n'. To program the opposite direction write a "-" sign.
pn pn' **pn''**	predefined points with indexes n, n' ed n''.
[-]cn **[-]cn'**	predefined circles with indexes n and n'. To program the opposite direction write a "-" sign.
[s2]	selects the largest possible circle.
d..	distance between two circles. Look in the direction of [-]cn: if cn' is on the right hand side, d is positive; cn' is on the left hand side, d is negative.

Circle with center and radius in cartesian coordinates

c2=I50J100r-40

Circle with center and radius in cartesian coordinates

c1=o1I20J20r-15

Circle with center and radius in polar coordinates

c2 = m70 a30 r15

Circle of r radius tangent to two predefined straight lines

c3 = l1,l2,r-15

Circles tangent to a straight line and a circle

c3 =l1,-c2,r8
c4 = -c2,l1,r8

Circles tangent to a straight line and a circle

c9 = -c2,l1,r-8
c10 = l1,-c2,r-8

Circles tangent to a straight line and a circle

c4 = -l2,c1,r-40
c5 = c1,-l2,r-40

Circles through a point and tangent to a straight line

c3 = p1,-l1,r25
c4 = -l1,p1,r25

Circle tangent to two circles

c5 = c1,c2,r-8
c6 = c2,c1,r-8

Circle tangent to two circles

c9 = -c2,c1,r-8
c10 = c1,-c2,r-8

Circle through a point and tangent to a circle

c2 = c1,p1,r60
c3 = p1,c1,r60

Circles through two points

c1 = p1,p2,r20
c2 = p2,p1,r20

Circle with predefined center tangent to a straight line

c3 = p1,l2

Circle with predefined center tangent to another circle

c2 = p1,c1
c3 = p1,c1,s2

Cerchio passante per tre punti

c1 = p1,p2,p3
c2 = p3,p2,p1

The circle goes from the first to the second point and then to the third point.

Circle whose center is a point

c1 = p1,r-40

Circle concentric to another circle

c6 = c5,d-10

DEFINITION OF A PROFILE

Profile start and end

A profile programmed in GTL is delimited by two codes:

G21 identifies the profile start

G20 identifies the profile end

Profile types

A profile is a sequence of predefined geometric elements. Profiles may be open or closed.

Open profiles

In an open profile the starting point is different from the end point (pn).
Tool radius offset is perpendicular to the first point of the first element and to the last point of the last element.
To open tool offset radius on the first profile point you must program G21 G41/G42 in the block. To close tool offset radius on the last profile point you must program G20 G40 in the block.

Closed profiles

To obtain a closed profile you must first program the last element and then re-call the first programmed element.
The first offset point in the profile is the intersection between the first and the last translated elements (first point = last point).
Tool radius offset must be opened at profile start, i.e. on the block that programms the last element, by programming codes G21 G41/G42. To close tool offset radius at profile end you must program G21 G41/G42 in the block.
If the first or the last element are circles, there are two possible intersections. By default the system selects the first intersection. To select the second intersection, write s2 both in the block that programs the last element at profile start and in the block that programs the last element at profile end.

Allowed G codes

The only G codes accepeted by GTL are as follows: G27, G28, G04, G09, G41 and G42.

IMPORTANT

If pn is programmed in the starting profile block then the profile is open. pn may be programmed both at profile start and at profile end but cannot be programmed in the profile.

Open profile

Example of open profile

```
................
l1      = X Y25,a
p1      = X-20 Y25
p2      = X90 Y25
c1      = I30 J25 r-14
c2      = I45 J25 r15
................
G21 G42 p1          - first point
l1
r3
c1 s2
c2 s2
l1
G20 G40 p2          - last point
................
```

IMPORTANT

Tool radius offset must start on the first point of the profile and end on the last point. Tool radius offset is disabled on the first motion block of the plane axis programmed after G40.

Closed profile

Example of straight line-straight line closed profile

```
.................
I5        = X Y-15,a180
.................
I1        = X-30 Y-15,a135
.................
G21 G42 I5              - last element
I1                      - first element
.................
I5                      - last element
G20 G40 I1              - first element
```

Example of straight line-circle closed profile

```
c1        = I.. J.. r
.....................
l1        = X.. Y.. a90
.....................
l5        = X.. Y..,a180
.....................
G21 G42 l5 s2          - last element
c1 s2                  - first element
l1
.....................
l5 s2                  - last element
G40 c1                 - first element
.....................
```

IMPORTANT

Tool radius offset must be opened on the block that programs the last element of the profile and end on the block that programs the first element of the profile. Tool radius offset is disabled on the first motion block of the plane axis programmed after G40.

Spindle axis motion

Non-contouring axes can be moved on any point in the profile, for example, to penetrate the part.

Spindle axis motion at the first point must be programmed after the point in open profiles, and between the definition of the last support element and that of the first element in closed profiles.

Examples:

```
. . . . . . . .
G21G42 p1
Z-10
I1
. . . . . . .
```

```
. . . . . . . .
G21G42 I5
Z-10
I1
. . . . . . .
```

Connecting the elements

The geometric elements of the profile may be interlinked by tangential or intersecting lines, by an automatic radius or by a bevel.

1. Intersection between elements

The intersection between two straight lines has only one solution.
In straight line-circle intersections the default solution is the first of the two possible intersections. To select the second intersection it is necessary to program the s2 discriminator after the definition of the first element.

N24 l1
N25 c1

N28 c1 s2
N29 l1

N33 c1
N34 c2

N36 c1 s2
N37 c2

In straight line-circle intersections, the sequence of solutions is given by the direction of the line. In circle-circle intersections the first intersection is on the left hand side of the straight line that links the centers of the first and second circle, whereas the second intersection is on the left hand side of the straight line that links the centers of the first and second circle.

2. Automatic radiuses

GTL permits to define radiuses between elements by simply programming the radius length and sign. A positive value defines a counterclockwise radius; a negative value defines a clockwise radius.

```
...................
N20 l1
N21 r-8
N22 l2
...................
```

```
...................
N24 l1
N25 r7
N26 c1
N27 r7
N28 l1
```

```
...................
N90 c1
N91 r10
N92 c2
...................
```

Automatic radius between elements

If tool offset is enabled, the tool is positioned on the intersection between both geometric elements translated by the tool radius. To insert a radius between the elements you must program an r0 radius.

no radius
.....
I1
I2
.....

radius r0
.....
I1
r0
I2
.....

Zero radius with intersection and tool offset

IMPORTANT

The block with the r radius cannot be programmed either inmmediately after the block that includes the G21 or immediately before the block that includes the G20 code. In other words, the profile cannot either start or end with a radius.

3. Bevels

GTL permits to define a bevel between two lines by programming the bevel length without sign. The bevel length is the distance to the intersection point.

```
N30 I1
N31 b3          Bevel
N32 I2
```

NOTE:

- The block with the b bevel cannot be programmed either inmmediately after the block that includes the G21 or immediately before the block that includes the G20 code. In other words, the profile cannot either start or end with a bevel.
- With GTL displacements occur always at the configured feedrate. To program a move at a rapid rate it is necessary to declare a very high F.
- If the interpolation plane is not formed by axes X and Y you must define the plane and then program the elements with respect to the its abscissa and ordinate.

Example:

```
N1      G16BY
N2      I1 = B70,Y40,a150
N3      I2 = B8,Y8,a-95
N4      p1, = I1,I2
N5      I3 = B8,Y8,B70,Y15
N6      c1 = I70,J40,r-25
.........
.........
N12     G21 G42 I2
.........
```

EXAMPLES OF GTL PROGRAMMING

Example 1

```
N1   (DIS,"EXAMPLE GTL")
N2   l1=X70Y40,a150
N3   l2=X8Y8,a-95
N4   p1=l1,l2
N5   l3=X8Y8,X70Y15
N6   l4=X50Y,a90
N7   c1=I70J40 r-25
N8   c2=p1,r-20
N9   F250 S800 T1.1 M6M3
N10 GXY
N11 Z-10
N12 G21G42l2
N13 l3
N14 r3
N15 l4
N16 r3
N17 c1
N18 r5
N19 l1
N20 r5
N21 c2s2
N22 l2
N24 G20 G40 l3
N25 GZ2
N23 XY M30
```

Example 2

N1	(DIS,"EXAMPLE GTL")
N2	I1=X-50Y10,X30Y50
N3	I2=X30Y50,X70Y10
N4	I3=X70Y0,a-90
N5	I4=X=Y-20,a180
N6	I5=X10Y-20,X0Y0
N7	I6=X0Y0,X-10Y-20
N8	I7=X-50Y0,a90
N9	c1=I-10J40r18
N10	c2=I50J30r-14
N11	c3=I40J-20r10
N12	S...M...T3.3M6M....
N13	G0X-30Y0
N14	Z-10
N15	G21G42I7
N16	I1
N17	r-8
N18	c1
N19	r-8
N20	I1
N21	I2
N22	c2s2
N23	I2
N24	I3
N25	r-10
N26	I4
N27	c3s2
N28	I4
N29	r-8
N30	I5
N31	I6
N32	r-8
N33	I4
N34	r -10
N35	I7
N36	G20G40I1
N37	G0Z20

0
20
40
55
Y
r13
p3
80
r15
r40
c5
r5
c1
r55
p2
c3
c2
45°
35
c4
21
r55
r40
38
0
X
l8
l7
r10
c6
l6
45°
l3
l4
l5
r12
l2
10
l1
60°
60
65
r5
65
50
25
0
50
70

Example 3

```
N1    (DIS,"EXAMPLE 3")
N2    S...F...T1.1M6 M...
N3    o1=X20 Y21 a45
N4    l1=X0 Y-60,a180
N5    l2=X50 Y0,a90
N6    c6=o1 I-38 J-35 r10
N7    l3=c6,a135
N8    l4=c6,a-45
N9    l5=X0 Y-50,a180
N10   l6=X-50 Y-65,a60
N11   l7=X-25 Y0,a90
N12   c3=I-65 J0 r55
N13   c4=I0 J80 r55
N14   p2=c3,c4
N15   c1=p2,r-15
N16   p3=X40 Y80
N17   c2=c1,p3,r40
N18   c5=I55 J80 r13
N19   l8=X70 Y0,a-90
N20   G21 G42 l8
N21   Z-10
N22   l1
N23   r-5
N24   l2
N25   r12
N26   l3
N27   c6
N28   l4
N29   l5
N30   l6
N31   l7
N32   r40
N33   c1
N34   c2
N35   r-5
N36   c5
N37   r-5
N38   l8
N39   G20 G40 l1
N40   G Z20
N41   X... Y... M30
```

Example 4: 8 profile repetitions executed with 2 passes

```
N1        (DIS,"GTL EXAMPLE WITH ROTATION")
N2        F...S...T2.2 M6
N3        MSA=2
N4        p1=X50 Y0
N5        c1=I0 J0 r50
N6        c2=I0 J0 r10
N7        l1=c2,a180
N8        l3=X0 Y0,a45
N9        l2=c2,a45
N10       p2=l3,c1,s2
N11       GX60 Y0
N12       Z-2
"START"   N13
N14       E25=0
N15       (RPT,8)
N16       (ROT,E25)
N17       G21 G42 p1
N18       c1
N19       r3
N20       l1
N21       r-3
N22       l2
N23       r3
N24       c1
N25       G20 G40 p2
N26       E25=E25+45
N27       (ERP)
N28       (ROT,0)
"END"     N29
N30       MSA=0
N31       (EPP,START,END)
N32       GZ20
N33       XY M30
```

END OF CHAPTER

WORKING CYCLES FOR TURNING SYSTEMS

PROFILE PROGRAMMING

The profile is a program composed by blocks of movement that can be recalled by a roughing or finishing cycle.

There are three methods to program the profiles:

1. Write the profile in ISO language using the line editor

```
G1Z0X30
Z-15
Z-20.80127X50.09618
G3 Z-2629165X61.07694 I-33.79165 J35.09618
G1 Z-55.82309 X95.17692
G3 Z-64.82309 X100 I-64.82309 J64
G1 Z-76
X140
```

2. Write the profile in GTL language using the line editor

```
p1=Z0 X40       l1
l1=p1, a180     r-5
c1=I-40 J40     c1
l2=c1, a180     l2
p2=Z-55 X100    r-5
l3=p2, a90      l3
G21 p1          G20 p2
```

3. Write the profile with the graphic editor.In this case the profile is formed with the support of the direct geometry available at the editor level and therefore the program will be translated in output into the ISO language.

For further information , see the User Manual.

Restrictions to the definition of a profile to be recalled by the macro-instructions of roughing/finishing.

- The profile must be completely described in a subroutine, indicated in the following pages as “profile_name”.
- The blocks contained in the subroutine will have to be ISO/GTL types only, and only the programming of :
  - Axis names and co-ordinates
  - Movement G functions (G1, G2, G3 and possible operands)
  - Therefore, in general the three-letter commands will NOT be taken into consideration and neither will any commands that might modify the profile description (origins, mirror, scale factor)

NOTE:

The non-conformity to the listed points, might lead to a wrong interpretation of the profile described in the subroutine.

Figure 16-1 Para-axial roughing with pre-finishing and finishing

```
T1.1 M6 S.. F..
G X143 Z1.5
(SPF,PROF2,Z,9,X1,Z1)
X300 Z200
T2.2 M6 S.. F..
G X30 Z2
(CLP,PROF2)
G X300 Z200
.........
```

```
PROF2
p1 = Z0X30
l1 = p1, a180
l2 = Z-15X30, a120
l3 = Z-60X100, a150
l4 = Z-75X100, a180
p2 = Z-75X140
l5 = p2, a90
G21 p1
l1
l2
r15
l3
r18
l4
l5
G20 p2
```

SPECIAL CYCLES PROGRAMMING

The following three-letter codes allow defining and recalling of special cycles inside the program:

SPA	para-axial roughing without pre-finishing
SPF	para-axial roughing with pre-finishing
SPP	roughing parallel to the profile
CLP	finishing cycle
FIL	threading cycle
TGL	groove cutting cycle

MACRO-INSTRUCTIONS OF PARA-AXIAL ROUGHING WITHOUT PRE-FINISHING

Profile roughing, with cutting passes parallel to the ordinate or abscissa axis.

(SPA,profile_name, axis_name, n_pass. [,axis1 stock] [,axis2 stock])

Example : (SPA,PROF1,Z,6,X1,Z2)

where:

profile_name	is an ASCII string representing the part program name in which the profile is described.
axis_name	specifies the abscissa or ordinate name (Z o X) along which the roughing passes must be executed or in which the programmed stock has to be left.
n_pass	specifies the number of roughing passes . It may be a value between 1 and 255.
axis1 stock	indicates the machining allowance to be left on axis 1 (normally the X axis). The value must be positive.
axis2 stock	indicates the machining allowance to be left on axis 2 (normally the Z axis). The value must always be positive.

The stock values must be always positive.

The control decides automatically if the roughing refers to an internal or an external form by taking into consideration the approach point and the profile direction, and therefore assigns the correct sign to stock allowance.

The approach point must be external to the roughing field by at least as much as the programmed stock allowance.

If the profile is not monotonous, (that is it contains cavities), during the executing of the roughing cycle these are automatically passed over.

IMPORTANT

The profile must be monotonous for the roughing axis ,otherwise the system will display an error.
Therefore, the profile may contains cavities, but as regards the points trend along the roughing axis, they must be always increasing or decreasing.

Figure 16-2 Example of a profile roughing with working passes parallel to the X axis

```
.........
.........
T1.1 M6 S.. F..
G X143 Z1.5
(SPA,PROF1,X,12,X1,Z1)
.........
.........
```

```
PROF1
p1 = Z-76X140
I1 = p1, a-90
I2 = Z0X100, a0
I3 = Z-60X100, a-30
I4 = Z-15X30, a-60
p2 = Z0X30
I5 = p2, a0
G21 p1
I1
I2
r-18
I3
r-15
I4
I5
G20 p2
```

Figure 16-3 Example of roughing parallel to the Z axis

PROF2
p1 = Z0X30
l1 = p1, a180
l2 = Z-15X30, a120
l3 = Z-60X100, a150
l4 = Z-75X100, a180
p2 = Z-75X140
l5 = p2, a90
G21 p1
l1
l2
r15
l3
r18
l4
l5
G20 p2

.....
.....

T1.1 M6 S..F..
GX143 Z1.5
(SPA, PROF2, Z, 9, X1, Z1)
.....
.....

Diagram to determinate the profile development direction according to the approach point.

The profile development direction must be defined by taking into consideration the direction taken by the tool to perform the roughing cycle, which is also the profile development direction.

Figure 16-4 Tool profile and direction

MACRO-INSTRUCTIONS OF PARA-AXIAL ROUGHING WITH PRE-FINISHING

Roughing with cutting passes parallel to the ordinate and abscissa axis and final pass parallel to the profile:

(SPF, profile_name, axis_name, n_pass [,axis1 stock] [,axis2 stock])

For the meaning of the parameters and the chosen attach point , refer to the SPA macro-instructions. If the profile presents cavities, they will be rough-shaped during the SPF cycle as indicated in the following examples:

```
PROF3
p1 = Z0X62
l1 = p1, a135
l2 = Z0X68, a180
l3 = Z-15X0, a90
l4 = Z0X76, a180
l5 = Z-38X76, a198
l6 = Z-77X76, a105
p2 = Z-97, X86
l7 = p2, a135
G21 p1
l1
l2
l3
r2
l4
r10
l5
r-8
l6
r4
l4
l7
G20 p2
```

```
...........
...........

T1.1 M6 S..F..
GX90 Z2
(SPF, PROF3, Z, 7, X1, Z1)
...........
...........
```

IMPORTANT

The profile must be monotonous for the roughing axis or the system will indicates an error.
Therefore, the profile may present cavities but as regards the points trend along the roughing axis, it must be always increasing or always decreasing.

Figure 16-5 Example of internal para-axial roughing with pre-finishing parallel to the X axis

PROF4
p1 = Z-47X20
l1 = p1, a90
l2 = Z0X40, a0
l3 = Z-20X68, a45
p2 = Z0X68
l4 = p2, a0
G21 p1
l1
l2
l3
l4
G20 p2

.....
.....

T..S..F..
GX15 Z2.5
(SPF, PROF4, X, 10, X2, Z2)
.....
.....
(CLP, PROF4)

MACRO-INSTRUCTION OF ROUGHING PARALLEL TO THE PROFILE

This macro-instruction has been designed for roughing of pre-formed parts where the stock allowance is more or less constant on the rough stock.

The format is:

(SPP,profile_name, n_pass, axis_name, stock1, stock 2, axis_name, stock1, stock 2)

where:

profile_name	is an ASCII string representing the part program name in which the profile is described.
n_pass	number of the roughing passes. It may be a value between 1 and 255.
axis_name	it is the axis name (abscissa Z / ordinate X) on which the indicated stock must to be left.
stock1	stock to be left on the finished part in Z / X
stock2	stock on the rough part Z / X

The stock values must always be programmed, even when their value is zero are valid. The same considerations made for the SPA and SPF macro-instructions must be applied to the approach point.

Figure 16-6 SPP use

```
.........
.........
T1.1 M6 S...F...M...
T.. S.. F.. M..
G X143 Z2
(SPP,PROF2,4,Z,1,10,X,1,10)
.........
```

Figure 16-7

```
.........
.........
T.. S.. F..M..
G X84 Z1
(SPP,PROF5,4,X,0,10,Z,0,0)
.........
.........
PROF5
p1 = Z0X60
l1 = p1, a180
l2 = Z-30X60,Z-50X40
l3 = Z0X40, a180
l4 = Z-70X40, z-90X60
p2 = Z-110X60
l5 = p2, a180
G21 p1
l1
r10
l2
r-10
l3
r-10
l4
r10
l5
G20 p2
```

MACRO-INSTRUCTION OF A PROFILE FINISHING

The programming format is:

(CLP,profile_name)

where:

profile_name is an ASCII string representing the part program name in which the profile is described.

CLP is the only profiling cycle during which any F functions programmed inside the profile are used.

Figure 16-8 Para-axial roughing with pre-finishing and finishing

```
.........
T1.1M6 S..F..
GX143 Z1.5
(SPF, PROF2,Z,9,X1,Z1)
X300 Z200
T2.2 M6 S..F..
GX30 Z2
(CLP, PROF2)
GX300 Z200
.........
```

THREADING CYCLE

The threading cycle allows a cylindrical or taper thread programmed in a single block to be executed in several passes. The programming format is:

(FIL,axis1 , [axis2 ,] pass, n_ pass [,F .] [,R .] [,G ..] [,P...] [,M .] [,T ..] [,H ..] [,a ...] [,b..] [,r ..])

where:

axis1 abscissa and/or ordinate axis (normally Z or X)

axis2 name of abscissa and/or ordinate axis (normally Z or X) and final co-ordinate for the indicated axis (optional, it is programmed in tapered threading)

pitch threading pitch

n_pass number of threading passes. It may be a value between 1 and 255.

F... number of polishing passes. It may be a value between 1 and 255.
Default: 0

R... distance of tool from return part (default value 1)

G... internal/external threading:
0 = external thread
1 = internal thread.
Default: external (0)

P... starts number (default value 1)

M... entry mode

0 = rough-shaping at 1 angle (side)
1 = rough-shaping at 2 angles (zig zag).
Default: 0

T... output mode:

0 = thread with final groove
1 = thread without final groove (tearing)
2 = output with circular interpolation (connection).
Default: 0

H... metric/inches/non-standard

0 = metric thread
1 = whitworth thread
2 = non standard thread with depth and angle defined by parameters a and b.
Default: 0

a... angle of non-standard thread

b... depth of non-standard thread

r... output radius for the exit thread. Obligatory if output mode 2 (T2) has been programmed.

Parameters included between square parenthesis may be omitted.

When non-standard threads are programmed (H2), the (K) pitch must be:

$$K > 2 * b * tg (a/2)$$

NOTES

a) The unit calculates the movements along the thread so that passes are performed with constatn swarf removal.

In case of threads with mulitples starts, the pitch to be defined is that of a single turn. The system performs each pass on all starts before executing subsequent passes. The management of multiple starts principles is carried out without moving the threads start point, but by introducing an angular displacement with respect to the zero angular position of the spindle.

b) For thread with a final groove, the theoretical final Z position must be programmed because the cycle will automatically stop and retract the axes , half a pitch after the theorical final position.The parameters **a** (thread angle) and **b** (thread depth) are required only for programming non-standard thread (H2).

	metric	whitw.	others
angle	60°	50°	a
H	f(pitch)	f(pitch)	b

Figure 16-9

Single start thread

Figure 16-10

Two start thread

Figure 16-11

c) For threads without a final grove, the tool reaches the programmed endpoint and then reaches the return diameter with a tapered thread.

d) In threads without a final groove, SEMIAUTO must not be used, otherwise the result will be a final groove .

e) The threading cycle must not be defined in G28.

f) For taper threading, the maximum taper allowed is equal to 1/2 of the thread angle.

Figure 16-12 Example of threading cycle

```
N35  T5.5 M6
N36  G0 G97 X24 Z37 S250 M3 M8
N37  (FIL,Z4,2,5,F1,R2)
N38  G0 X250 Z215
```

GROOVE CUTTING CYCLE

The cycle generates a sequence to generate a groove parallel to the abscissa or ordinate axis (generally Z or X), internal or external.

The programming format to obtain a parallel groove to a plane axis is:

(TGL,axis1, axis2, width_tool [, coord_external,B/R..., B/R...,])

The minimum requested format is:

(TGL axis1, axis_2, width_ tool)

where:

axis1	Plane abscissa or ordinate axis name and final co-ordinate of the groove (the initial co-ordinate must be programmed before defining the groove cutting cycle)
axis2	Name of 2nd axis of plane and internal position of the groove (see the drawing)
width_tool	Tool width
coord_external	Co-ordinate of top of grove.
B/R	Initial radius/bevel (optional)
B/R	Final radius/bevel (optional)

Figure 16-13 Groove cutting cycle

The TGL command must be preceded by a block of movement in G0/G1 on the cycle beginning point.

Figure 16-14 Example of groove parallel to the Z axis

Axis1	Z-20
Axis2	X30
Width-tool	5
Coord-external	40
B/R	R1,R0

N1 T1.1 M6 S.. F..
N2 G X50 Z-40
N3 TMR=2
N4 (TGL,Z-20,X30,5,40,R1,R0)
N5G X10 Z100

Figure 16-15 Example of groove parallel to the X axis

Axis1	X50
Axis2	Z-5
Width-tool	5

N1 T1.1 M6
N2 G X20 Z5
N3 TMR=2
N4 (TGL,X50,Z-5,5)

The systems automatically forces a dwell at the end of the groove.

The stop time is defined by the three-letter TMR. If a stop is not wanted, program TMR=0 before the groove cutting cycle.

Figure 16-16 Internal groove

Axis1	Z-10
Axis2	X40
Width-tool	5

N1 T1.1 M6 S.. F.. M..
N2 G X25
N3 Z-25
N4 TMR=2
N5 (TGL,Z-10,X40,5)
N6 .....

At the cycle end the tool returns to the cycle start point programmed in the previous block.

END OF CHAPTER

CHARACTERS AND COMMANDS

TABLE OF CHARACTERS

Following is the punched tape format of the ASCII, EIA and ISO characters recognized by the control. Note that, depending on the code, there may be odd parity (EIA), even parity (ISO) or no parity (ASCII) .

CHARACTER	DESCRIPTION
0 to 9	• Numbers
+	• Addition
-	• Subtraction
*	• Multiplication
/	• Division
.	• Decimal Point
"	• Label Identifier
(	• Open Parenthesis
)	• Close Parenthesis
[	• Open Parenthesis
]	• Close Parenthesis
{	• Open Parenthesis
}	• Close Parenthesis
;	• Comment Symbol
,	• Parameter Separator

CHARACTER	DESCRIPTION
=	• Assignment Symbol
<	• Less than (jump symbol)
>	• Great than (jump symbol)
LF (ISO or ASCII line feed) CR (EIA carriage return)	• Block End
#	• Synchronization
&	• Asynchronization
!	• Prefix User variable
@	• Prefix PLUS variable
>>	• Increase on single operand
A a	• Axis name • Acceleration on profile • Angle with GTL
B b	• Axis name • Bevel in cutter diameter compensation
C c	• Axis name • Circle with GTL
D d	• Axis name • Distance with GTL • Tool diameter (RQP,RQT)
E	• Parameters used in machining cycles
F	• Axes feedrate in G1-G2-G3
G	• Preparatory Code (G00 - G99) • Reserved (G100 - G299) • Paramacro subroutines (G300 - G999)
H h	• Parameters used in PARAMACROS • Offset change during continuous move
I i	• Coordinates of the arc centre in a circular interpolation (G2 G3) (abscissa) • Variable pitch in G33 • Tool dimension vector (with j and k)

CHARACTER	DESCRIPTION
J j	• Coordinates of the arc centre in a circular interpolation (G2-G3) (ordinate) • Min depth increase in (G83) • Tool dimension vector (with i and k)
K k	• Reduction factor for I and J in drilling cycle • Threading Pitch (G33) • Threading Pitch (G84) • Helix pitch in helical interpolation • Tool dimension vector (with i and j)
L l	• User table variables • Length 1 of tool offset (RQT, RQP) • Length 2 of tool offset (RQT, RQP) • Straight line with GTL
M m	• Auxiliary functions • Normal vector on surface (with n and o)
N n	• Part program block number • Normal vector on surface (with m and o)
o	• Normal vector on surface (with n and m) • Source with GTL
P p	• Axis name • Point with GTL
Q	• Axis name
R r	• Rapid positioning in cycles G81 - G89 • Deviation from the spindle zero (used in multi-start threads) adius in a circular interpolation G02-G03 • Profile radius (used for cutter diameter compensation only) G73 cycle • Radius with GTL
S s	• Spindle speed • Intersection with GTL
T t	• Tool and tool offset address • Time needed to complete the move in one block
U u	• Axis name • Compensation factors in axis 1 (offset)
V v	• Axis name • Compensation factors in axis 2 (offset)

CHARACTER	DESCRIPTION
W w	• Axis name • Compensation factors in axis 3 (offset)
X	• Axis name
Y	• Axis name
Z	• Axis name

G CODES

The table below lists the G codes available with 10 Series systems

G CODE	FUNCTION
G00	Rapid axes positioning
G01	Linear interpolation
G02	Circular interpolation CW
G03	Circular interpolation CCW
G04	Dwell at end of step
G09	Deceleration at end of step
G16	Defined interpolation plane
G17	Circular interpolation and cutter diameter compensation in the XY plane
G18	Circular interpolation and cutter diameter compensation in the ZX plane
G19	Circular interpolation and cutter diameter compensation in the YZ plane
G20	Closes GTL profile
G21	Opens GTL profile
G27	Continuous sequence operation with automatic speed reduction on corners
G28	Continuous sequence operation without speed reduction on corners
G29	Point-to-point mode
G33	Constant or variable pitch thread
G40	Disables cutter diameter compensation
G41	Cutter diameter compensation - tool left
G42	Cutter diameter compensation - tool right
G70	Programming in inches
G71	Programming in millimetres
G79	Programming referred to machine zero
G80	Disables fixed cycles
G81	Drilling cycle
G82	Spot-facing cycle
G83	Deep hole drilling cycle
G84	Tapping cycle
G85	Reaming cycle
G86	Boring cycle
G89	Boring cycle with dwell
G90	Absolute programming
G91	Incremental programming
G92	Axis presetting
G93	Inverse time (V/D) feedrate programming
G94	Feedrate programming in ipm or mmpm
G95	Feedrate programming in ipr per revolution or mmpr
G96	Constant surface speed in fpm or mpm
G97	Spindle speed programming in rpm
G72	Point probing with probe ball radius compensation
G73	Hole probing with probe ball radius compensation
G74	Probing for theoretical deviation from point without probe ball radius compensation
G99	Deletes G92

MATHEMATICAL FUNCTIONS

- **SIN**
- **COS**
- **TAN**
- **ARS**
- **ARC**
- **ART**
- **SQR**
- **ABS**
- **INT**
- **NEG**
- **MOD**
- **AND**
- **OR**
- **NOT**
- **FEL**
- **FEC**
- **FEP**

LOCAL AND SYSTEM VARIABLES

VARIABLE NAME	FUNCTION
L	Plus User Table variable
SN	System Number
SC	System Character
TIM	System Time
E	Local variable
H	Paramacro variable
HF	Paramacro flag variable
HC	Paramacro string variable
!name	User variable
@name	PLUS variable
VEF	Velocity Factor
TPO	Tool path optimization
TPT	TPO threshold
CET (PRC)	Circular Endpoint Tolerance
FCT	Full Circle Threshold
ARM	Defining Arc Normalization Mode
DLA	Enables/disables look ahead
MDA	Maximum Deceleration Angle Computation
DWT (TMR)	Dwell time
SSL	Spindle Speed Limit
ERR	Ables/Disables Part Program Errors handling
STE	System Error
MSA (UOV)	Defining a Machining Stock Allowance
TRP (RMS	Tapping Return Percentage
DSB	Disable slashed blocks
UPA (RTA)	Update Probe Abscissa
UPO (RTO)	Update Probe Ordinate
VFF	Velocity Feed Forward
ODH	Online Debug Help
DYM	Execution mode with G27

THREE-LETTER CODES

CODE	FUNCTION
DAN	Define axis name
IPB (DTL)	In Position Band
UAO	Use absolute origin
UTO (UOT)	Use temporary origin
UIO	Use incremental origin
RQO	Requalify origin
SOL (DLO)	Software overtravels
DPA (DSA)	Define protected areas
PAE (ASC)	Protected area enable
PAD (DSC)	Protected area disable
MIR	Mirror machining
ROT (URT)	Active plane rotation
SCF	Scale factor
AXO	Axis Offset Definition
RQT (RQU)	Requalifying Tool Offset
RQP	Requalifying Tool Offset
TOU (TOF)	Declare a tool out of life
DPP (DPT)	Defining Probing Parameters
LOA	Dual port loading of tables
RPT	Open repetition of a set of program blocks
ERP	Close repetition a set of program blocks
CLS	Call a subroutine for execution
EPP	Execute a portion of a part program (subprogram)
PTH	Set *path* for subroutines and paramacros
EPP	Execution of a part of a program
EPB	Execution of a program block
GTO	Branch Command
IF, ELSE, ENDIF	Conditional Execution of parts of a program

CODE	FUNCTION
GDV	Device Definition
RDV	Release Device
UDA	Dual Axes
SDA	Special Dual Axes
UGS (UCG)	Use the graphic scale
CGS (CLG)	Clear the graphic field
DGS (DCG)	Disable the graphic scale
UPR	Uses Plane Rotated
UVP	Use Virtual Polar
UVC	Use Virtual Cylindrical
TCP	Tool Center Point
DIS	Display a variable
DLY	Cause a delay in program execution
DSB	Disable slashed blocks
REL	Disactivate a part-program
WOS	Put the system on hold for a signal
GTA	Axes acquisition
SND	Send a synchronization message
WAI	Wait for a synchronization message
EXE	Automatic activation of a part program
ECM	Execution in MDI mode of a block in a specified process
PRO	Definition of default process
DCC	Definition of communication channel
PVS	PLUS variable selection
GTP	Determine approach point for automatic contour milling
CCP	Perform automatic contour milling

CODE	FUNCTION
SPA	Para-axial rough shaping without pre-finishing
SPF	Para-axial rough shaping with pre-finishing
SPP	Parallel rough shaping to the profile
CLP	Finishing cycle
FIL	Threading cycle
TGL	Groove cutting cycle

ASCII CODES

The tables that follow show the 256 elements of the extended ASCII character set, together with their decimal and hexadecimal equivalents.

DEC	HEX	CHARACTER
000	00	BLANK (NULL) (NULL)
001	01	☺ (SOH)
002	02	☻ (STX)
003	03	♥ (ETX)
004	04	♦ (EOT)
005	05	♣ (ENQ)
006	06	♠ (ACH)
007	07	● (BEL)
008	08	◘ (BS)
009	09	○ (HT)
010	0A	◙ (LF)
011	0B	♂ (VT)
012	0C	♀ (FF)
013	0D	♪ (CR)
014	0E	♫ (SO)
015	0F	☼ (SI)

DEC	HEX	CHARACTER
016	10	► (DLE)
017	11	◄ (DC1)
018	12	↕ (DC2)
019	13	‼ (DC3)
020	14	¶ (DC4)
021	15	§ (NAC)
022	16	▬ (SYN)
023	17	↨ (ETB)
024	18	↑ (CAN)
025	19	↓ (EM)
026	1A	→ (SUB)
027	1B	← (ESC)
028	1C	∟ (FS)
029	1D	↔ (GS)
030	1E	▲ (RS)
031	1F	▼ (US)

DEC	HEX	CHARACTER
032	20	BLANK (SPACED)
033	21	!
034	22	"
035	23	#
036	24	$
037	25	%
038	26	&
039	27	'
040	28	(
041	29	)
042	2A	*
043	2B	+
044	2C	,
045	2D	-
046	2E	.
047	2F	/

DEC	HEX	CHARACTER
048	30	0
049	31	1
050	32	2
051	33	3
052	34	4
053	35	5
054	36	6
055	37	7
056	38	8
057	39	9
058	3A	:
059	3B	;
060	3C	<
061	3D	=
062	3E	>
063	3F	?

DEC	HEX	CHARACTER
064	40	@
065	41	A
066	42	B
067	43	C
068	44	D
069	45	E
070	46	F
071	47	G
072	48	H
073	49	I
074	4A	J
075	4B	K
076	4C	L
077	4D	M
078	4E	N
079	4F	O

DEC	HEX	CHARACTER
080	50	P
081	51	Q
082	52	R
083	53	S
084	54	T
085	55	U
086	56	V
087	57	W
088	58	X
089	59	Y
090	5A	Z
091	5B	[
092	5C	\
093	5D	]
094	5E	^
095	5F	_

DEC	HEX	CHARACTER
096	60	`
097	61	a
098	62	b
099	63	c
100	64	d
101	65	e
102	66	f
103	67	g
104	68	h
105	69	i
106	6A	j
107	6B	k
108	6C	l
109	6D	m
110	6E	n
111	6F	o

DEC	HEX	CHARACTER
112	70	p
113	71	q
114	72	r
115	73	s
116	74	t
117	75	u
118	76	v
119	77	w
120	78	x
121	79	y
122	7A	z
123	7B	{
124	7C	|
125	7D	}
126	7E	~
127	7F	△

DEC	HEX	CHARACTER
128	80	Ç
129	81	ü
130	82	é
131	83	â
132	84	ä
133	85	à
134	86	å
135	87	ç
136	88	ê
137	89	ë
138	8A	è
139	8B	ï
140	8C	î
141	8D	ì
142	8E	Ä
143	8F	Å

DEC	HEX	CHARACTER
144	90	É
145	91	æ
146	92	Æ
147	93	ô
148	94	ö
149	95	ò
150	96	û
151	97	ù
152	98	ÿ
153	99	Ö
154	9A	Ü
155	9B	¢
156	9C	£
157	9D	¥
158	9E	Pt
159	9F	f

DEC	HEX	CHARACTER
160	A0	á
161	A1	í
162	A2	ó
163	A3	ú
164	A4	ñ
165	A5	Ñ
166	A6	ª
167	A7	º
168	A8	¿
169	A9	⌐
170	AA	¬
171	AB	½
172	AC	¼
173	AD	¡
174	AE	«
175	AF	»

DEC	HEX	CHARACTER
176	B0	░
177	B1	▒
178	B2	▓
179	B3	│
180	B4	┤
181	B5	╡
182	B6	╢
183	B7	╖
184	B8	╕
185	B9	╣
186	BA	║
187	BB	╗
188	BC	╝
189	BD	╜
190	BE	╛
191	BF	┐

DEC	HEX	CHARACTER
192	C0	└
193	C1	┴
194	C2	┬
195	C3	├
196	C4	─
197	C5	┼
198	C6	╞
199	C7	╟
200	C8	╚
201	C9	╔
202	CA	╩
203	CB	╦
204	CC	╠
205	CD	═
206	CE	╬
207	CF	╧

DEC	HEX	CHARACTER
208	D0	╨
209	D1	╤
210	D2	╥
211	D3	╙
212	D4	╘
213	D5	╒
214	D6	╓
215	D7	╫
216	D8	╪
217	D9	┘
218	DA	┌
219	DB	█
220	DC	▄
221	DD	▌
222	DE	▐
223	DF	▀

DEC	HEX	CHARACTER
224	E0	α
225	E1	β
226	E2	Γ
227	E3	π
228	E4	Σ
229	E5	σ
230	E6	µ
231	E7	τ
232	E8	ϕ
233	E9	θ
234	EA	Ω
235	EB	δ
236	EC	∞
237	ED	∅
238	EE	∈
239	EF	∩

DEC	HEX	CHARACTER
240	F0	≡
241	F1	±
242	F2	≥
243	F3	≤
244	F4	⌠
245	F5	⌡
246	F6	÷
247	F7	≈
248	F8	°
249	F9	∙
250	FA	·
251	FB	√
252	FC	n
253	FD	2
254	FE	■
255	FF	BLANK "FF"

Extended ASCII Character Set

END OF APPENDIX

ERROR MESSAGES

Description of error messages and remedial actions

This Appendix lists error messages that may appear during system operation with a short description and suggested remedial action.

Code	Message description and remedial action
NC001	**Syntax Error** Syntax error found in the part program block or in the MDI block
NC002	**Wrong number of axes for G code** This message is displayed to indicate that: • At least one axis must be programmed in G04 • Only one axis must be programmed in a canned cycle block (from G81 to G89).
NC003	**Canned cycle parameters missing** Canned cycle parameters (i.e. K, I, ...) are missing
NC004	**Missing parameters for G code** Parameters for G code are missing (i.e. G33 ...K)
NC005	**Missing J and/or K for G83 cycle** K or J parameter are missing in the G83 canned cycle
NC006	**Missing I and/or J for G2/G3 code** I and/or J parameters are missing in G2/G3 codes (circles)
NC007	**Probing cycle parameters missing** Probing cycle parameters (i.e., E or r) are missing

Code	Message description and remedial action
NC008	**Format error** This error is displayed in the following cases: • Wrong variable index • Feedrate (F) = 0 or negative • Wrong variable format • Repeat number is illegal (number of repetitions must be from 1 to 65535) • Format error in assignment, e.g. assignment to strings with different lengths • PLUS variables writing/reading error • Character variable format error in DIS code: not specified as CHAR • Protected area not allowed: 0< protected area number <4 • Variable not configured
NC009	**Undefined symbol** This message is displayed in the following cases: • Axis name not configured in AMP • Variable does not exist
NC010	**Overflow** Expression too long
NC011	**Function not allowed** This message is displayed in the following cases: • Activation of an M type "block calculation" requested with calculation block already active or on HOLD • Feedrate override with non linear active block requested
NC012	**Wrong use of axis slave** A slave axis previously declared in the UDA block is programmed directly in a part program or input in an MDI block. This error is also displayed in the following cases: 1. When an attempt is made to move the slave axis manually. 2. When programming UDA, if the slave axis is already involved in TCP programming as a linear or rotating axis. 3. When programming UDA, if the slave axis is already involved in virtual programming (UPR, UVP, UVC) as a real or virtual axis. 4. When programming UDA, if the master axis is already involved in virtual programming (UPR, UVP, UVC) as a virtual axis.
NC013	**Operand not allowed in canned cycle** Operand not allowed in G72 G73 G74 canned cycles
NC014	**K parameter not allowed in G84** K parameter not allowed during G84 programming of a spindle without transducer
NC015	**Wrong programming of G2/G3 code** Both the centre and the radius (R) of the circle are specified Eliminate either the radius or the IJ centre coordinates

Code	Message description and remedial action
NC016	**Illegal number of operands** Illegal number of operands in the AXO block
NC017	**Illegal number of pseudo axis** Too many pseudo axes programmed in the block (max. 6)
NC018	**Illegal number of axes in G33 code** More than 2 axes programmed in G33
NC020	**G not allowed** G not allowed in the threading cycle
NC021	**Operand not allowed with G code** Operand incompatible with type of movement
NC022	**Block and system state not congruent** • UDA enable/disable while other virtual mode is active • Synchronisation while blocks in execution are suspended. For example, synchronisation with cutter diameter compensation active. • Attempt to execute an MDI block while a macro is active
NC024	**G and program state not congruent** This message occurs when • G41-G42 canned cycles cannot be programmed when cutter diameter compensation is active • Threading cannot be programmed when cutter diameter compensation (G41 G42) is active or when a canned cycle is programmed • Interpolation planes (G17, G18, G19) cannot be programmed when cutter diameter compensation (G41 G42) is active
NC025	**G and dynamic mode not congruent** G function not compatible with current dynamic mode. For example: functions G72, G73, G74 do not accept in continuous mode (G27, G28) Switch from G27 to G28 or vice versa with active non-linear ramps (MOV > 1).
NC026	**G41/G42 and part program state not congruent** Cutter diameter compensation (G41/G42) not compatible with current program state.
NC027	**G needs spindle with transducer** G33 and the threading macro-cycle FIL need a spindle with transducer.
NC028	**G not congruent with feedrate mode** G72, G73, G74 must be executed when G94 is active
NC029	**Operand and part program state not congruent** Operand incompatible with current part program state For example: r, b operands are not allowed in the ISO standard state (G40)

Code	Message description and remedial action
NC030	**M and dynamic mode not congruent** Machine logic operands incompatible with active dynamic mode For example: M at motion end not compatible with (G27-G28) T programmed with G41/G42 active
NC031	**M/T/S and motion type not congruent** Machine logic operands incompatible with the type of move For example: G33 + end of motion M function
NC032	**Probing cycle operands inhibited** Probing cycle operands not allowed For example: Operands I,J,K,R,u,v,w,b,t are not allowed in G72-G73
NC033	**Missing third axis for helix** The third axis for helix programming is missing
NC034	**"Expedite" function without motion** An "expedite" M is present a block that does not program a move; "expedite" M's must always be associated motion to a move
NC035	**Feed or speed not programmed** • Feedrate or speed not programmed for canned cycle execution • Motion block in G1/G2/G3 without programmed feed
NC036	**Z-axis not found for G87 cycle** The z-axis has not been programmed for the cycle G87.
NC037	**Read only variable** The specified variable is of the read-only type. For example: TIM.
NC038	**Part program record too long** The programmed record has more than 127 characters. It is displayed in conjunction with the PART PROGRAM NAME message.
NC039	**Part program access denied** The part program file specified with this error message is not read-accessible as it is open in write mode for another user (e.g. Editor, DOS real time, etc.).
NC040	**P.P. block not allowed from serial line** Block not allowed during part program execution from serial line.
NC041	**Wrong serial line configuration for EPS**
NC042	**Nesting of IF greater then 5** The maximum number of nested IF commands has been exceeded.

Code	Message description and remedial action
NC043	**ELSE not allowed** An ELSE command has been programmed without a previous IF command.
NC044	**ENDIF not allowed** An ENDIF command has been programmed without a previous matching IF command
NC048	**Illegal argument for TAN** The argument of the TAN operator is 90 degrees (the result would be infinite)
NC049	**Illegal argument for SQR** The argument of the SQR operator (square root) is a negative number
NC050	**Too many programmed axes** More than 6 axes have been programmed in the block
NC051	**Division by zero** A division by zero has been detected in the expression that calculates an axis dimension (e.g. X10/0 or in the expression calculated by the Evaluate utility)
NC052	**String too long** The max. string length can be 80 characters. This message is displayed when a longer string is used in the following cases: • display of a string with the DIS code • string variable (SC) assignment
NC053	**Label duplicated** This message is displayed when the program is selected or activated. It shows that there are two identical labels in the part program. The duplicated label is also displayed.
NC054	**Undefined label** The label programmed in a branch instruction (GTO) or in a call for a subroutine (EPP) does not exist
NC055	**Label too long** This message is displayed when the system reads an SPG block. It indicates that a label having more than 6 characters has been programmed. The illegal label is also displayed
NC056	**Program table overflow** This message is displayed when the program is selected or activated. It indicates that the number of CLS for subroutines overflows the maximum configured in AMP. You can alter this parameter in AMP with the procedure described in the PROCESS CONFIGURATION section.

Code	Message description and remedial action
NC057	**Label table overflow** This message is displayed when the program is selected. It indicates that the number of programmed labels overflows the maximum configured in AMP. You can alter this parameter in AMP with the procedure described in the PROCESS CONFIGURATION section.
NC058	**End of program** End of file marker for: • block skipping • block editing • string search • program execution
NC059	**Beginning of program** Program start marker for: • block skipping • string search
NC060	**Nesting of RPT greater than 5** RPT max. nesting level (5) overflown
NC061	**Nesting of subroutine greater than 4** Subroutine max. nesting level (4) overflown
NC062	**Nesting of EPP greater than 5** EPP max. nesting level (5) overflown
NC063	**RPT/EPP cycle open at end of file** This message is displayed when: • The end of the file has been reached without finding the (ERP) block that closes the programmed (RPT) cycle • The end of the file has been reached without completing the subroutine defined with (EPP)
NC064	**ERP without RPT** (ERP) has been programmed without previously programming (RPT)
NC065	**Error during part program file handling** This message occurs during program reading/writing to indicate that: • a part program block has been skipped • error in program SPG/REL • error in subroutine opening/closing management • the accessed file does not exist or is protected
NC066	**Part program not found** The selected part program/subroutine is not stored in the E:\UPP directory

Code	Message description and remedial action
NC067	**Part program not selected** This error occurs when: • CYCLE START is given in AUTO but no part program has been selected • SKIP, MODIFY, ESCAPE commands are given but no part program has been selected • a branch instruction (GTO) is executed by the system in MDI mode
NC068	**Process number cut of range** The process number written in the three-letter code cannot be higher than the one set in AMP or less than 1.
NC069	**Paramacro modal already active** A paramacro is programmed when a modal paramacro is already active
NC070	**Paramacro not configured** The programmed paramacro has not been configured in AMP
NC078	**Software option not installed**
NC079	**Software option not available. Check security**
NC080	**Axis not referenced** This message occurs when: • The programmed axis is not referenced • The axis specified in the definition of a protected area with DPA is not referenced • The offset to be preset/requalified is associated to a non referenced axis
NC081	**Undefined DPP for probing cycle** Probing cycle parameters (approach coordinate, safety distance, velocity) are not defined in the DPP block
NC082	**Too many "Expedite" M codes** More than one expedite M code has been programmed in the block
NC083	**Undefined M code** The programmed M is not configured in AMP. Configure the M in AMP and restart the system
NC084	**Circle not congruent** The circle is not geometrically congruent: the radius or the final points are not correct

Code	Message description and remedial action
NC085	**Wrong threading parameters (I, K, R)** The programmed threading parameters (I, K and R) are not allowed. Calculate the I parameter with the following formula: $$\frac{16\ k}{2(\text{threading distance})}$$
NC086	**Helix pitch not congruent** The helix pitch is not geometrically correct
NC087	**Axes of plane needs same scale factor** Plane axes in G02/G03 programming (circle) must have the same scale factor. Change the scale factor with an SCF instruction
NC088	**Profile not congruent** The programmed ISO-offset profile is not correct
NC089	**Wrong direction on profile** Offset value in G41-G42 reverses the tool path direction
NC090	**Err. disabling cutter compensation** Wrong exit from cutter diameter compensation (G40)
NC091	**Too many blocks to resolve** Too many extra plane moves programmed with cutter diameter compensation active (G41-G42) (max. 2 extra plane moves).
NC092	**Entry in safety zone** The programmed move enters one of the three safety areas
NC093	**Canned cycle on rotate plane** Canned cycle programmed on rotated plane Disable plane rotation
NC094	**Canned cycle data not congruent** The parameters specified in the canned cycle (I, J, K, R) are not allowed. For example: canned cycle K = 0
NC095	**Missing parameters for G87** There are parameters missing in the G87 fixed drilling cycle . This cycle is used in the WOOD macros.
NC096	**Wrong probing cycle programming** This message appears when: • probing approach distance is null • hole probing is programmed with null radius (for example G73r0E5)

Code	Message description and remedial action
NC097	**Hole probing cycle not complete** The hole probing cycle not complete has not been completed
NC098	**Probing cycle not executed** This message occurs when the probe does not find the point to be probed before reaching the safety zone
NC099	**Probe has not been retracted** When measuring cycle starts the probe is already touching the part surface
NC100	**Hardware overtravel** The programmed axis has overflown the hardware overtravel. Jog it back within hardware travel limits
NC101	**Positive software overtravel** The programmed move causes the axis to exit the programmed or configured positive software travel limits
NC102	**Positive hardware overtravel limit** This message appears if the axis is jogged in the positive direction after it has reached its positive hardware overtravel limit. Select JOG DIR - and press CYCLE START to jog the axis back within the positive overtravel. **NOTE:** there is not other way of returning an axis to the HW operating limits
NC103	**Negative hardware overtravel limit** This message appears if the axis is on the programmed or configured negative hardware overtravel limit and you try to further jog it in the negative direction.
NC104	**Positive software overtravel limit** This message appears if the axis is on the programmed or configured positive hardware overtravel limit and you try to further jog it in the positive direction.
NC105	**Negative software overtravel limit** The axis is on the negative SW overtravel limit and we set a JOG DIR move
NC106	**JOG past software overtravel limit** The JOG INCR value would take the axis past the software overtravel limit
NC107	**Axes not on profile** This message appears if we try to quit CYCLE STOP after a series of jog moves without taking the axes back to the profile. Select JOG RETURN and return the axes to the profile
NC108	**Home and JOG DIR not congruent** This message appears when we try to home an axis in a JOG DIR opposite to the configured homing direction. **NOTE:** if the homing cycle is configured as automatic the system will automatically correct JOG DIR without displaying the error. Press the JOG DIR softkey to align the jogging direction to the configured homing axis direction

Code	Message description and remedial action
NC109	**Error in exit HOLD: mode changed** This message occurs when we try to exit from HOLD by setting an operating mode (BLK_BLK, AUTO, MANUAL) that is different from the one in which the system went on HOLD. Select the correct mode and re-try.
NC110	**Block not allowed in HOLD** This message occurs when: • we try to execute an MDI motion block with the system on HOLD. When the system is on HOLD axes can only be jogged. • the programmed M is configured as not allowed on HOLD
NC111	**Active reset denied** This message occurs when we tried to execute an ACTIVE RESET in the following conditions: • while a block is executed with G27-G28 • during execution of a block followed by a circular block (G02/G03) • during execution of the last block before a syntactically inappropriate block The system only accepts another ACTIVE RESET (particularly convenient for bypassing the circular block) or RESET
NC112	**Wrong use of roll-over axis with G90** The programmed coordinate for the axis with rollover in G90 is greater than the roll over pitch configured in AMP
NC113	**Wrong JOG DIR for jog return** If the jog direction is negative during automatic or manual JOG RETURN, the system forces positive jog direction. This message appears if reversal is prevented by the machine logic
NC115	**Probing cycle executed before the end of approach movement.** Probing cycle carried out during fast probe approach
NC116	**Wrong use of real axis during a virtualization modality** This message occurs when a real axis is programmed when virtual mode is active
NC117	**Tool direction active: movement not permitted** This message occurs when only the tool direction is active and all other movements are not allowed.
NC118	**Negative software overtravel** The programmed move causes the axis to move past the programmed or configured software negative travel limits
NC119	**Command not allowed during search in memory** Command not allowed during the search in memory

Code	Message description and remedial action
NC120	**Mode to select out of range** This message occurs when the selected mode is out of range. Allowed modes are in the 1-8 range: 1 MDI — 5 INCREMENTAL JOG 2 AUTO — 6 RETURN ON PROFILE 3 BLOCK by BLOCK — 7 HOMING FILE 4 CONTINUOUS JOG — 8 HPG
NC121	**Axes number to select out of range** The number of axes selected for manual moves with library call NC NC_SELAXI is out of range. The ***allowed range*** is from 1 to the number of axes configured for the process 1 < ***allowed range*** < n. of configurated axes +1
NC122	**Too many axes selected for manual move** A larger number of axis names than accepted have been inserted in the part program block. Edit the part program block.
NC123	**Bad select mode for cycle** This error is displayed when CYCLE START is pressed in the following conditions: • a mode other than MDI has been selected during execution of a tool change axis move • system on HOLD, AUTO or BLK/BLK with MBR (multiblock retrace) not configured in AMP • system on HOLD with MBR active and selected mode other than AUTO or BLK/BLK • system in IDLE and ACTIVE_RESET with selected mode other than AUTO or BLK/BLK • system in IDLE with MBR active and selected mode other than AUTO or BLK/BLK • system in HRUN with MBR active and selected mode other than AUTO or BLK/BLK • ACTIVE RESET command in HOLD status with selected mode other than MDI, AUTO or BLK/BLK. **NOTE:** For further information about the machine status (HOLD, MDI, HRUN, etc.) refer to the USER GUIDE.
NC124	**Wrong axis name** This error is displayed in the following when: • The name of the selected axis is not configured in the axes table associated to the process • The definition of the interpolation plane is not correct because its axis/axes are not configured in the axes table. • The plane to be defined with G17, G18, G19, G16 cannot be defined because one of the specified axes is not configured in the axes table • The axis specified in the NC_ACTUALOFS call does not exist

Code	Message description and remedial action
	• The axis specified in the SCF, MIR three-letter blocks is not configured • An axis coordinate reading error has occurred because the specified axis does not exist • The axis specified in the SOL, DPA, UDA, UGS, AXO, UAO three-letter blocks is not configured or is duplicated • the axis specified in the AX_SHARE Library call does not exist • you are releasing an axis shared with the logic through the GTA command.
NC125	**Data length out of range** The keyboard buffer for MDI blocks has been overflown. Allowed entry length ranges from 1 to 127 characters
NC126	**Failed to write variable** Value of variable not written
NC127	**Failed to read variable** Value of variable not read
NC128	**Operative limit definition wrong** • Error in defining the software operating limits with the three-letter mnemonic SOL. • The programmed software limits must be defined in configured software. • Software operating limits are not configured in AMP.
NC129	**Protected area not defined** This message occurs when you try to enable with a PAE a protected area which does not exist. Define a protected area with PAE.
NC130	**Offset length not defined for the axis** This message occurs when you try to preset or requalify an offset that is not associated to the specified axis.
NC131	**Tool orientation code wrong** The specified tool orientation code is illegal
NC132	**Error from PLUS environment** Error in the PLUS environment generated by PLUS library calls PL_SET92, PL_RESG92, PL_PRESCOR, PL_UAO, PL_UTO, PL_UIO, PL_RQT, PL_RQP, or PL_RQO Error during execution of: RQO, UAO, UTO, UIO, RQT, RQP, G92
NC133	**Error from servo environment** Error in the SERVO environment during origin or offset presetting The error can also be caused by the IPB command when the In Position Band value transferred to an axis is rounded out, in the internal computations, to less than 1 digit.
NC134	**Manual movement not executed , no axes configured** Manual movements are not allowed because no axes have been configured

Code	Message description and remedial action
NC135	**Axis not configured** The ID programmed in the GTA block has not been configured
NC136	**Programmed id identifies an auxiliary axis** The ID programmed in the GTA block corresponds to an auxiliary axis and is not allowed
NC137	**Axis not available** • The ID of the axis programmed in the GTA block corresponds to another process • ID of axis programmed in AX_SHARE Library function is not available.
NC138	**Axis id duplicated** The axis ID is duplicated in the GTA block
NC139	**Programmed ID identifies a spindle** The ID programmed in the GTA block corresponds to a spindle axis and is not allowed
NC140	**Set spindle speed failed** The machine logic (task $SPROG) does not accept the variation of spindle speed.
NC141	**New tool request failed** The machine logic (task $nTPROG) does not accept the T code programming.
NC142	**M executed failed** The machine logic (task $mDECOD) does not accept the M code programming
NC143	**Pseudo axes programming failed** The machine logic (task $nPSEUDO) does not accept the pseudo axes programming.
NC144	**Axis motion inhibited** Axis motion denied by the machine logic (task $nCONMOV).
NC145	**End of move failed** The machine logic answers with error on the end of move signal (task $nENDMOV).
NC146	**Too many blocks without motion in continuous mode** Too many blocks without motion have been programmed in continuous mode (G27, G28)
NC149	**Program already selected** This error occurs in MDI mode when you try to activate the same paramacro several times.
NC150	**Axis homed** This message indicates that the axis has been homed.
NC151	**Axis on profile** This message indicates that RETURN TO PROFILE has successfully terminated and the axis has returned to the profile.

Code	Message description and remedial action
NC152	**End of automatic return to profile** This message indicates that automatic RETURN TO PROFILE has successfully terminated and all the axis have returned to the profile.
NC153	**End of block retrace** This message occurs when backward multiblock retrace. To retrace a greater number of blocks, alter the configured maximum.
NC156	**End of search in memory** End search in memory
NC160	**Command and system state not congruent** This message indicates that the command is not allowed in the present system status
NC161	**Internal error: class not exist**
NC162	**Internal error: NC message error** Switch the control off and then on again. If the message is retained, contact technical services.
NC190	**Insufficient length for tapping cycle** This error occurs when the distance covered in the acceleration and deceleration phases of the canned tapping cycle without transducer is longer than the total distance, and no space is left for machining.
NC191	**Insufficient length for tapping cycle with transducer** This error occurs when the distance covered in the acceleration and deceleration phases of the canned tapping cycle with transducer on the spindle is longer than the total distance, and no space is left for machining.
NC192	**Insufficient length for threading cycle** This error occurs when the distance covered in the acceleration and deceleration phases of the canned threading cycle is longer than the total distance, and no space is left for machining.
NC199	**Spindle not activated**
NC200	**File access error** Error in reading or writing a file.
NC201	**Set up file loading error** The axes configuration in the file accessed is different from the configuration on dual port.
NC202	**File/Dual port config. mismatch** The axes configuration in the file accessed is different from the configuration on dual port.

Code	Message description and remedial action
NC203	**Warning: table locked read only** PLUS denies access to the table on dual port.
NC204	**Illegal file size** The table on file is of wrong size.
NC205	**Empty magazine** The selected magazine doesn't have defined pockets.
NC206	**Pocket is still busy** The pocket defined for a tool is already reserved to a different tool.
NC207	**Illegal previous pocket** A tool taking up more than one pocket interferes with the pocket occupied by another tool (previous pocket).
NC208	**Illegal following pocket** A tool taking up more than one pocket interferes with the pocket occupied by
NC209	**Illegal random** An illegal random class has been traced in memory.
NC210	**Tool table is full** Dual port full during the loading of a tool table related to a certain magazine.
NC211	**Illegal double format for editor** A variable format non accessible to editor has been traced.
NC212	**Illegal magazine number into file** Error in reading or writing
NC213	**Pocket not initialized**
NC214	**Pocket not compatible**
NC215	**Illegal table name** The name of the table to be loaded is invalid. Make sure the extension of the table name is one the following: .TOL .USR .MAG .OFS .ORG .SPN
NC220	**Process undefined** The process has not been defined or configured. Define the default process with the PRO command or select an existing process for synchronisation commands.
NC221	**Wrong process type** A communication channel unsuitable for the command set has been used. Example: channel type 2 (PLUS) for EXE command execution.

Code	Message description and remedial action
NC222	**Wrong process number** The process number specified for synchronisation commands identifies the current process
NC223	**Process queue is full** The process queue (local or remote) that a message was sent to is full.
NC224	**Data sending too long** Data to be transmitted with SND are longer than 174 characters
NC225	**Data loading failed** The type or number of data transmitted with SND is not allowed
NC226	**Message already exists in queue** A SND command towards a process has been given before the process cleared the previous message.
NC227	**EXE or ECM failed** This message occurs when: • The status of the process to which the EXE or ECM command is sent does not allow automatic part program execution commands (RUN, HRUN, RUNH, HOLD) or an MDI instruction. • There is a syntax error in the program to which the EXE command is addressed
NC320	**UPR programming not allowed** UPR cannot be programmed when another virtual mode is active.
NC321	**Wrong incremental UPR programming** Incremental UPR can only be programmed if UPR is active.
NC322	**UPV programming not allowed** UPR cannot be programmed when another virtual mode is active. This error is also displayed if one of the real axes is a slave in UDA/SDA programming.
NC323	**Wrong axis type on UPV programming** The programmed type of real axis is not compatible with the virtual mode.
NC324	**Wrong programmed radius value** The radius programmed in the UVP block is not compatible with the linear axis position.
NC325	**UVC programming not allowed** UVC cannot be programmed when another virtual mode is active. This error is also displayed if the real axis is a slave in UDA/SDA programming.
NC326	**Programmed TCP code value out of range** The code that enables TCP is illegal.

Code	Message description and remedial action
NC327	**TCP programming not allowed** (TCP,5) cannot be programmed when another virtual mode is active. This error is also displayed if one of the linear or rotating axes of the TCP is a slave in UDA/SDA programming.
NC328	**TCP programming not congruent** The request to enable TCP is not compatible with the current TCP mode.
NC329	**Error on tangential TCP activation** Error during (TCP,4) enable. Check whether the specified axes ID's are configured in the user table.
NC330	**Error during get or release axes** GTA cannot be enabled when offsets, canned cycles or a virtual mode are active.
NC331	**Axis interpolator clock not congruent** The interpolator clock of one or more axes programmed in the GTA block is different from that of the current process.
NC332	**Zero value of ijk module** ijk error programming with active TCP: the module with such values is equal to zero.
NC333	**Wrong programming of ijk, mno** ijk and/or mno wrong programmed.
NC334	**Number of contouring blocks overflow** The max. number of blocks defined in AMP for automatic contouring or for rough-machining cycles is lower than required.
NC340	**Circles/lines not defined** The circle/line programmed in the GTL profile has not been defined.
NC341	**Wrong definition of circles/lines** There is an error in the definition of a GTL circle/line.
NC342	**Circles/lines not intersecting** The intersection requested by the GTL profile involves two circles/lines that do not intersect.
NC343	**Coinciding circles** The intersection requested by the GTL profile must be generated by two circles that do not intersect.
NC344	**Coinciding circles/lines/points** The circles/lines/points programmed in the GTL profile are coincident.
NC345	**Points inside circle** Profile error: the programmed point is inside a circle.

Code	Message description and remedial action
NC346	**Parallel lines** Point/circle programming error: the profile lines are parallel.
NC347	**Aligned points** Profile error: the points programmed in the circle definition are on the same line.
NC360	**Too many blocks of movement** The maximum number of blocks of movement allowed inside a profile recalled by a macro rough-shaping (SPA, SPF). has been reached. Check this limit value set in AMP.
NC361	**Profile error** The profile recalled by the macro rough-shaping (SPA, SPF) can not be rough-shaped. In general, only monotonous profiles can be roughed shaped for the rough-shaping axis (which is X or Z always decreasing or always increasing).
NC362	**Undefined work area** Switch off and switch on the control, if error persists, contact the assistance.
NC363	**Axis not congruent with interpolation plane** In the rough-shaping macros (SPA, SPF) the rough-shaping axis must pertain to the interpolation plane, as the axes for which the swarf is defined. Also in the threading macro the thread axis and the return axis must pertain to the interpolation plane
NC364	**Wrong approach to profile** Approach point not allowed for the rough-shaping macro (SPA, SPF). The approach point must always be external to the rough-shaping field in X, for rough-shapings parallel to the X axis, and external to the rough-shaping field in Z, for rough-shapings parallel to the Z axis
NC365	**Interpolation type not allowed** In the profile recalled by the rough-shaping macro (SPA, SPF) only linear or circular blocks of movement are allowed.
NC366	**Aligned points during rough-shaping** During the stage of profile rough-shaping an area that can not be rough-shaped has been reached. Control the consistency of profile and of macro parameters.
NC367	**Profile non consistent with approach** The approach point and the profile development direction don't allow to continue the rough-shaping.
NC370	**R or B parameters not allowed** In the groove cutting macro is not allowed a connection or initial or final bevel in case the external level has not been programmed.
NC371	**Tool width greater than groove width** Error in the groove cutting macro due to the fact that the tool width is superior to the groove width.

Code	Message description and remedial action
NC372	**Tool width null or not consistent with R or B parameters** Error in the groove cutting macro due to the fact that the tool width is null or inferior to the sum of connections and programmed bevels.
NC373	**Wrong initial position for TGL** The approach position for the groove cutting macro is not consistent with the parameters declared in the block.
NC375	**Missing parameters “a” and/or “b”** If the programmed threading is a non-standard one, in the block must be present also the parameters “a” and “b”.
NC376	**Wrong step for thread** In case of non standard threading it is necessary that the programmed pitch respects the following formula. It must be: $\text{Pitch} \geq 2 * \text{Thread depth} * \text{tg} \frac{\text{thread angle}}{2} * \text{principles number.}$
NC377	**Thread angle greater than 180°** Error in the threading macro due to the thread angle ≥ 180°
NC378	**Null thread length** Error in the threading macro if the thread length along the spindle axis is null.
NC379	**Wrong conical angle** In case of conical threading, the maximum conical admitted is equal to the half of the thread angle.
NC380	**Plane rotation not allowed with thread** It is not allowed to perform a threading cycle if there is active rotation for the interpolation plane.
NC381	**Circular exit not allowed without “r” parameter** Error in the threading macro due to the programming of an output with connection without radius value.
NC401	**HSM Part program not found or open part program error** Possible part program sharing error between the executable modules of the control. Reload the program or contact the customer engineering service.
NC402	**Error reading HSM part program** Part program in execution corrupt Turn the control off and on again or call the customer engineering service
NC403	**HSM configuration file not found or open configuration file error** Setup file not present. Check the presence of the file and the HSM three-letter code that defines the name

Code	Message description and remedial action
NC404	**Syntax error in HSM configuration at line** The specified line contains a syntax error. Check the syntax of the setup three-letter code in the manual.
NC405	**Starting position requested for all HSM defined axis** The first programmed point after the G61 must contain all the axes associated with the HSM setup. Program all missing axes, confirming any positions that do not change.
NC406	**Mandatory HSM param requested into configuration at line** The setup three-letter code set on the specified line requires other parameters Check the syntax of the setup three-letter code in the manual.
NC407	**Mandatory HSM param error** The setup three-letter code set on the specified line does not contain an obligatory parameter Check the syntax of the setup three-letter code in the manual.
NC408	**HSM param at wrong line position** Reserved for future developments. Call the customer engineering service.
NC409	**HSM param not allowed into part program** Reserved for future developments. Call the customer engineering service.
NC410	**Two points are requested to define a segment** There must be at least two points between the G61 and G60 codes. Edit the part program and do not use G61/G60.
NC411	**HSM defined axes not found among the process axis param at line** There must be at least two points between the G61 and G60 codes. Edit the part program and do not use G61/G60.
NC412	**General HSM params must be setted before axis params at line** The setup three-letter code set on the specified line refers to an axis identifier not associated with the process on which the part program is executed. Check the setup three-letter code in the manual or the identifiers of the axes associated with the process.
NC413	**Axis params must be setted after general HSM params at line** In the setup file, the general three-letter codes must be defined first and then the axis setup codes. Check the setup sequence in the manual.
NC414	**HSM needs more configuration params** In the setup file, the general three-letter codes must be defined first and then the axis setup codes. Check the setup procedure in the manual.

Code	Message description and remedial action
NC415	**HSM needs more axis params** Axes with missing setup three-letter codes have been specified in the setup file. Check the setup procedure in the manual.
NC416	**HSM needs more tools params** The tool setup three-letter codes have not been specified in the setup file. Check the setup procedure in the manual.
NC417	**HSM needs more cinematic params** The setup three-letter codes of the axes that refer to the tool and the axes that refer to the clamping of the part have not been specified in the setup file. Check the setup procedure in the manual for the CIN,t and CIN,w three-letter codes.
NC418	**Axis not defined into HSM params at line** An axis not previously defined with the AXI three-letter code has been configured in the setup file on the specified line. Check the setup procedure in the manual.
NC419	**Axis already defined into HSM params at line** A previously defined axis has been defined in the setup file on the specified line. Check the setup procedure in the manual.
NC420	**Too many adding axis (max 3) into HSM params** A maximum of 3 additional axes may be defined in the setup file (axes not belonging to the Cartesian system or rotary) Check the setup procedure in the manual.
NC421	**Too many axis (max 6) into HSM params** A maximum of 6 axes may be defined in the setup file. Check the setup procedure in the manual.
NC422	**Axis type error into HSM params at line** An incorrect axis type or one previously associated with other axes has been defined in the setup file. Check the setup procedure for the AXI three-letter code in the manual.
NC423	**Operative limit reached into HSM part program for axis** The software operating limits have been reached for the specified axis. Check the part program.
NC424	**Virtualization or TCP not allowed with HSM** When the G61 is activated, neither virtualisations nor the TCP must be active. Check the part program.

Code	Message description and remedial action
NC425	**Error reading HSM configuration file** Setup file corrupt. Reload the setup file or call the customer engineering service.
NC426	**HSM not enabled in AMP** The HSM feature has not been enabled in AMP. Enable it.
NC427	**HSM option not allowed by HW key** The HSM option has not been enabled. To use the feature on a machine with more than 3 axes, the option must be enabled using the Product Key Call the customer engineering service.
NC428	**HSM option not loaded** The HSM option has been enabled using the Product Key but has not been loaded onto the NC. Load the option.
NC429	**Illegal param value into HSM** A parameter with an incorrect value (must be positive) has been defined in the setup file on the specified line. Check the setup procedure in the manual
NC430	**Illegal feed value into HSM** The Feed rate value is missing or less than 0. Set a valid feed rate value.
NC431	**Syntax error in HSM** Syntax error in the part program during an HSM machining process. Correct the program and see the programming manual to find out which blocks are allowed between G61 and G60.
NC432	**Illegal use of tangent axes** Only one tangential axis may be present or the tangential axis is being incorrectly used. Check the setup procedure in the manual.
NC433	**Invalid parameter set-up modality** Programming of points type and relative parameter set-up wrong in three-letter PNT code of high speed set-up file.
NC434	**Polynomial programming does not admit parameter set-up requested** Configuration of three-letter PNT code in high speed set-up file for entire polynomial programming is wrong.
NC435	**Nodes must be programmed in increasing mode** Nodes of Bsplines programmed as inputs must be sorted in increasing order.

Code	Message description and remedial action
NC436	**Node programming requested** Number of nodes programmed is insufficient: for Bspline inputs, the number of nodes must be the same as the number of control points, plus the degree of Bspline + 1.
NC437	**Final point of previous Bspline must be confirmed** Programmed Bsplines must be continuous, i.e., last point in a Bspline must be the same as first point in the next.
NC438	**Control points for correct definition of Bspline missing** Minimum number of control points, for Bspline input, must be: (degree of Bspline + 1) *2.
NC439	**Programmed polynomials lack continuity** Programmed polynomials must be continuous, i.e., last point in a polynomial must be the same as first point in the next.
NC440	**ROT type IJK vector invalid when Tangent axes computation is requested** When working with a tangent axis, the ijk vector must not be ROT type in three-letter VER code of high-speed set-up file.
NC441	**ROT type IJK vector invalid when (TOD) parameters are used** In three-letter VER code of high-speed set-up file, ijk vector cannot be set as ROT when, for instance, drive chain includes fewer than two rotating axes.
NC442	**PRG type IJK vector invalid when type AXI/CLP points are programmed** When programming type (PNT, AXI/CLP... or (PNT, AXI/CCP... points, ijk vectors cannot be used. Use one of the following instead: (VER, REL/ROT, ...
NC443	**REL type MNO vector invalid when type AXI/CCP points are programmed** When programming type (PNT, AXI/CXP... or (PNT, CLP, CCP... points, mno vectors cannot be used. Use: (VER..., PRG, ...
NC444	**Axis shared with PLUS environment** The axis you are trying to move, or on which you wish to perform a virtualisation, has been previous acquired by the logic through the AX_SHARE function.
NC445	**Machine unit of measurement not congruent in HSM** This error occurs during the execution of a part program optimised with Path Optimizer when the unit of measurement specified in the setup file is not congruent with the unit of measurement of the machine.
NC446	**mno/uvw programming wrong in HSM file** This error occurs when both the mno/uvw vectors are defined, programmed or calculated in the VER three-letter code of the high speed setup file.
NC447	**uvw programming not compatible with toroidal tool** The uvw compensation factors are not supported with toroidal tools.

NC448	**Invalid corner for the determination of uvw** This error is generated when you try to determine the uvw compensation factors for a point where an overelongation has occurred. That is to say, tilting by 180° relative to the point of contact on the tool occurs.
NC456	**ijk versor is null** The ijk module versor programmed is null
NC457	**mno versor is null**
NC458	**pqd versor is null**

END OF APPENDIX

ERROR MANAGEMENT

GENERAL

This Appendix discusses how the operator can manage errors in order to prevent machining process interruptions.

The following system variables permit to configure the error management mode:

SYSTEM VARIABLE	FUNCTION
ERR	Enables error management from part program
STE	Is a read only system variable that contains the error generated by the previous command when automatic error management is active.

ERR - Enable/disables error management from part program

ERR is a system variable that permits to select how to manage the error.

Syntax

ERR = *value*

where:

value It may be 0 or 1 and can be expressed with a number or a local or system variable.

ERR=0 (default) disables error management from program is disabled. Errors are displayed and can interrupt program execution.

ERR=1 enables error management by the part program.

Characteristics:

The tables that follow list a series of programming errors that can be managed from program. For each error with provide the error code (value written in the STE read-only variable) and a short description.

In addition, the table shows the NCxx that is displayed when ERR=0. For the complete list of NCxx codes refer to Appendix B.

Probing cycle errors

Command	STE value	Description	Error code displayed if ERR=0
G71,G72,G73	0	probing cycle finished regularly	----------
	1	probing has not taken place	NC098
	2	probe has not been released	NC099
	3	probe parameters not specified	NC007
	4	probing has taken place with rapid approach	NC115

Example:
Presence of the part is verified with both methods:

```
(DPP,30,15,500)
G0 Z50
X80 Y100
ERR = 1
G72  Z0  E1
(GTO,END,STE = 1)
ERR = 0
.....................................
.....................................
"END" (DIS, "PART NOT PRESENT")
M...
.....................................
```

IMPORTANT

Since G71,G72,G73 modify the value written in the STE variable when ERR=1, we recommend that you use STE immediately after giving a command that might alter its contents.

Shared axes errors

Command	STE value	Description	Error code displayed if ERR=0
GTA	0	execution without errors	----------
	10	axis ID not configured	NC135
	11	axis ID belongs to logic	NC136
	12	axis ID belongs in another process	NC137
	13	axis ID associated to spindle axis	NC139

Example:

```
;. . . . .
;. . . . .
ERR = 1
;. . . . .
"RETRY"
(GTA,X1,Y2,Z5)          ;Requests acquisition of ID's  1,2,5
(GTO,NEXT,STE<>12)          ;Axis error test busy
(DLY,0.5)                   ;If the axis is busy the program waits until it is released
(GTO,RETRY)
"NEXT"
(GTO,ERROR,STE<>0);If other error occurs the cycle will be aborted
G1 X10 Y10 F1000            ;After the axes have been acquired the move is executed
Z50
;. . . . .
;. . . . .
;. . . . .
"ERROR"
;. . . . .                  ;Error management
```

IMPORTANT

- Error management from part program also provides an easy way of synchronizing two processes when one of them must wait for one or more axes to be available before executing a given process.
- Since GTA modifies the value written in the STE variable when ERR=1, we recommend that you use STE immediately after giving the command that might change its contents.

END OF APPENDIX

10 Series CNC
ASSET
Reference Manual

Code: 45004472R
Rev. 06

PUBLICATION ISSUED BY:

OSAI S.p.A.
Via Torino, 14 - 10010 Barone Canavese (TO) – Italy

Tel. +39-0119899711

Web: www.osai.it

e-mail: sales@osai.it
service@osai.it

Copyright © 2001-2004 by OSAI
All rights reserved

Edition: Oktober 2004

IMPORTANT USER INFORMATION

This document has been prepared in order to be used by OSAI. It describes the latest release of the product.
OSAI reserves the right to modify and improve the product described by this document at any time and without prior notice.
Actual application of this product is up to the user. In no event will OSAI be responsible or liable for indirect or consequential damages that may result from installation or use of the equipment described in this text.

SUMMARY OF CHANGES

General

This publication was issued together with the release of version 7.4.

CHAPTER	UPDATING TYPE
INDEX	updated
CHAPTER 1	
Page 8	Changed tool corrector table
Page 9,10,11,12	Added new pages on I/O management
CHAPTER 4	
Page 13	CPY copy file: added example
CHAPTER 5	
Page 7	RAP: added parameters and examples
CHAPTER 7	
Page 11	FH0: changed parameter

PREFACE

This manual is intended for 10 Series CNC programmers.
It describes the standard language extensions provided with the ASSET option.

REFERENCES

This publication constitutes an extension of the 10 Series CNC standard programming language.
Further information may be found in the other manuals of the 10 Series:

- 10 Series CNC Programming Manual
- 10 Series CNC User Manual
- 10 Series CNC AMP - SW Characterisation Manual
- 10 Series CNC PLUS Lybrary - User Manual
- 10 Series CNC PLUS Application Manual
- 10 Series CNC PLUS Language and PLUSEDIT

SUMMARY

1. General

This chapter provides an introduction to the ASSET language and lists the complete set of ASSET variables.

2. Input commands

This chapter discusses the commands that permit to define and handle manual data entry windows.

3. Screen management commands

This chapter discusses the commands used for configuring and handling configurable video screens.

4. File management commands

This chapter describes the ASSET commands used for opening, reading, writing, closing, deleting and saving ASCII files in the directory of programs.

5. Process management commands

This chapter deals with ASSET commands that permit to read process parameters such as current and programmed tool number, current and programmed tool offset number, process state, sub-state and mode, and axes coordinates.

6. Serial line management commands

This chapter describes the commands for management of the serial line: modes of configuration and operation of the triliteral functions for data reception/transmission.

7. Operating commands

This chapter describes the commands by way of which a process can be issued operating commands such as CYCLE ON, CYCLE OFF, HOLD, etc.

Appendix A - ASSET error messages

This appendix lists the error messages that may be displayed by the system when executing ASSET commands.

Appendix B - Error management from part program

This appendix contains information on how to manage certain types of error from the part program so as not to interrupt execution of the part program itself.

Appendix C - ASSET triliterals table

This appendix provides the complete list of ASSET triliteral functions.

WARNINGS

For correct use of the system, it is important to follow the indications given in the manual, and in particular those items marked: WARNING, CAUTION or IMPORTANT.

Indicating facts or circumstances that may cause damage to the system, items of equipment or operators.

Indicating information to be taken into consideration in order to avoid damage to the equipment in general.

Indicating operations to be carried out with particular care to ensure full success of the application.

END OF PREFACE

INDEX

GENERAL

INTRODUCTION 1-1
VARIABLES 1-3
User Table Variables 1-4
PLUS Tables Variables 1-5
Axes Table 1-6
Tools Table 1-7
Tool Offsets Table 1-8
PLUS I/O 1-9
Inputs (the first 85 arrays) 1-9
Inputs (after the 85th array) 1-10
Outputs (the first 85 arrays) 1-11
Outputs (after 85th array) 1-12
FILE 1-13
ASCII files 1-13
Binary files 1-14
COMMANDS 1-15
LCK - Locking/Unlocking PLUS tables 1-16

INPUT COMMANDS

DIF - Definition of a data field 2-2
INP - Manual data input 2-7

SCREEN MANAGEMENT COMMANDS

OUT - Screen line parameters 3-3
SCR - Screen selection 3-6

FILE MANAGEMENT COMMANDS

OPN - Opening a file 4-2
WRT - Writing a file 4-4
REA (RED) - Reading a file 4-6
CLO - Closing a channel 4-8
DEL (CAN) - Deleting a file 4-9
INS - Saving an ASCII file in the directory of programs 4-11
CPY - Copy a file 4-12

PROCESS MANAGEMENT COMMANDS

PRO - Definition of the default process 5-2
RTP - Reading the number of the programmed or active tool 5-3
ROP - Reading the number of the programmed or active tool offset 5-4
GPS - Reading the process state, sub-state or mode 5-5
RAP - Reading the axis coordinates 5-7
PLS - Reading PLUS SW variables 5-9

SERIAL LINE MANAGEMENT COMMANDS

SOP - Activate and configure the serial port 6-2
GET - Data reception from serial line 6-6
PUT - Data transmission on serial line 6-8
SCL - Close the serial port 6-10
EPS - Execute part program from serial line 6-11

OPERATING COMMANDS

CON - CYCLE ON command 7-2
COF - CYCLE OFF command 7-3
HON - HOLD ON command 7-4
HOF - HOLD OFF command 7-5
RES - RESET command 7-6
SMD - Set operating mode 7-7
SAX - Select axis for manual movement 7-8
DIR - Direction of manual movements 7-9
JOG - Jog step value 7-10
FHO - Enable/Disable FEEDHOLD 7-11

ASSET ERROR MESSAGES

Description of messages and remedial action A-1

ERROR MANAGEMENT FROM PART PROGRAM

ERR - Enabling/disabling automatic error management from part program B-2
Manual data input errors B-3
File management errors B-3
Locked table condition B-4
Serial line management errors B-4

ASSET TRILITERALS TABLE

List of the Asset Triliterals C-1

END OF INDEX

GENERAL

INTRODUCTION

ASSET (Advanced Super Set Extension Tool) is a programming language that enhances 10 Series standard programming capabilities. It provides a set of instructions, variables and variables handling rules that facilitate customisation of 10 Series functions.

In particular, ASSET permits to:

- create personalised video screens.
- create data entry windows and personalise their size, position, graphic layout, background and foreground colors, and the number and length of the various data entry fields.
- open, read, write and close ASCII files.
- manage PLUS tables and read and/or write accessible parameters.
- read and/or write PLUS I/O variables.
- handle a series of process commands used for reading the process status, the axes position, the tool parameters, etc.
- handle the serial line from part programs.
- send out operating commands in emulation of the Front Panel/Teach Pendant.

These operations can be carried out by means of specific 3-letter ASSET codes or by writing in the program blocks the ASSET variables that provide access to parameters that are normally inaccessible to the programming environment. These commands can be given by the part program or entered from keyboard, i.e. written by the operator when the system is in MDI mode.

For example, ASSET can be used for programming a machining operation that needs to be automatically interrupted and kept on hold by the system until the operator fills in a given data entry window. Each ASSET command described in the manual is supplied with application examples.

Like all high level languages, ASSET must also be used by adequately trained personnel that are also well acquainted with the characteristics of the system. If inappropriate alterations are made to the system and logic parameters accessible via ASSET, serious system errors and malfunctions may occur.

ASSET instructions can be easily combined with the traditional 10 Series programming language. Actually, ASSET enhances 10 Series programming capabilities by making it possible for 10 Series commands to handle ASSET variables.
In addition to standard commands, functions and variables, ASSET uses 10 Series syntactic rules and conventions. For more information about these programming rules and conventions, please refer to 10 Series CNC Programming Manual.

VARIABLES

ASSET permits to write, read and alter any system variable, with the exception of those to which special restrictions apply. In doing so it applies the same syntactic rules and conventions as the standard 10 Series programming language. For more information about standard programming, please refer to Chapter 7 of 10 Series Programming Manual.

Here is a list of variables handled by ASSET:

- **Local variables:** E and H.
- **System variables:** SN and SC.
- **User variables:** They are identified by an exclamation mark followed by a name. For example, !USER1. They can contain long real numerical formats and alphanumeric characters.
- **PLUS variables:** They are identified by an @ followed by a name. For example, @PROG They can contain short, long or boolean numerical values.
- **User Table variables:** They are identified by an L followed by a numerical index from 0 through 399 which indicates the table cell. The index-cell relationship will be discussed in a later chapter.
- **PLUS Table variables:** They are identified by the $ character followed by the name given to the variable in the table and an index enclosed between brackets which provides the axes ID (axes table), the record number (tools table) or the tool offset number (tool offsets table).

Examples:

$AXORIG(2)	Identifies the current origin of the axis whose ID is 2 (read only)
$TSTATUS(25)	Identifies the condition of the tool stored in record 25 of the tools table
$TACTL1(125)	Identifies the current length of the tool stored in record 125 of the tools offset table.

User Table and PLUS variables are described later in this chapter. For more information about local, system user variables, please refer to the following 10 Series documents: Programming Manual, Characterisation Manual, and PLUS Library Manual.

User Table Variables

The User Table (refer to Chapter 3 in the "PLUS Application Manual") has 100 records, each one of which is made up of 4 variables, for a total 400 variables. The variable index ranges from 0 to 399.

The records and the variables stored in them are arranged sequentially, as shown in the following table:

Record N°	Variable 1	Variable 2	Variable 3	Variable 4
000001	L0	L1	L2	L3
000002	L4	L5	L6	L7
000003	L8	L9	L10	L11
000004	L12	L13	L14	L15

.....

.....

000099	L392	L393	L394	L395
000100	L396	L397	L398	L399

To calculate the index when the position of the cell in the table and the variable are known, the following formula can be used:

Index = (Record Number - 1) x 4 + (Number of the variable in the record - 1)

For example, the index of the highlighted cell in the table is as follows:

Index = (4 - 1) x 4 + (3 - 1) = 14

To calculate the record number and the variable when the index is known, the following formula may be used:

Record = (Index / 4) + 1
Variable = [Remainder of (Index / 4)] + 1

For example, L145 addresses record 37 (145 / 4 + 1) and variable 2 (remainder of the division + 1 = 1 + 1).

PLUS Tables Variables

PLUS Tables include the Axes Tables, the Tools Tables and the Tools Offsets Tables. The contents of these tables are normally handled by the machine logic. Each PLUS Table is made up of a given number of pages which contain the parameters that describe the object of the table. The number of pages in each table is as follows:

Table	N° of Pages	Contents
Axes table	32	Axes identification
Tools Table	250	Record number in the table
Tool OffsetsTable	300	Tool offsets number

The table page number coincides with the index that follows the variable name.

The type, number and symbol of a parameter vary from table to table. The sections that follow illustrate a typical page of each table. For further information about these tables, refer to Chapter 3 of the "PLUS Application Manual".

The ASSET instruction LCK permits to write protect PLUS tables. When write protection is active, PLUS tables may be accessed only via ASSET. More information about the LCK triliteral is provided in the final section of this chapter.

IMPORTANT

Values read or written in PLUS tables using ASSET are not affected by the current unit of measure (G70/G71) but are considered as absolute values. It should be remembered therefore that any numeric values representing lengths are with reference to the machine's unit of measure configured in AMP; it is up to the operator to perform any conversion required.

Axes Table

Each of the 32 pages in this table is made up of 19 fields identified by a name, as shown in the example below. The index of each variable can range from 1 through 32 and corresponds to the axes ID. The index may be a numerical value or another variable.

Field	Type	Variable name	Format	Meaning
reserved	reserved	---	---	---
AXNAME	read only	$AXNAME(x)	S	axis name in ASCII
AXORIG	read only	$AXORIG(x)	D	value of current origin
reserved	reserved	---	---	---
AXOFG92	read only	$AXOFG92(x)	D	current G92 offset value
AXTOFF	read only	$AXTOFF(x)	D	current tool offset value
PRO_OFFS	read only	$PRO_OFFS (x)	D	total current corrector value applied by the process when an 'h' is enabled
TOT_OFFS	read only	$TOT_OFFS(x)	D	total offset of the current axis
ORIG1	read/write	$ORIG1(x)	D	value of origin # 1
ORIG2	read/write	$ORIG2(x)	D	value of origin # 2
ORIG3	read/write	$ORIG3(x)	D	value of origin # 3
ORIG4	read/write	$ORIG4(x)	D	value of origin # 4
ORIG5	read/write	$ORIG5(x)	D	value of origin # 5
ORIG6	read/write	$ORIG6(x)	D	value of origin # 6
ORIG7	read/write	$ORIG7(x)	D	value of origin # 7
ORIG8	read/write	$ORIG8(x)	D	value of origin # 8
ORIG9	read/write	$ORIG9(x)	D	value of origin # 9
ORIG10	read/write	$ORIG10(x)	D	value of origin # 10
reserved	reserved	---	---	

x = page number or axis identification number
S = short, D = double

Example:
To write in variable E1 the value of the current total offset of the axis identified by ID4, which is the sum of the axis origin, the G92 offset and the tool offset, key in the following instruction:

E1 = $TOT_OFFS(4)

To assign to the origin # 1 of the axis identified by ID 5 the numerical value 1.4, key in the following:

$ORIG1(5) = 1.4

Tools Table

Each of the 250 records of the table has 13 fields whose ID's are shown in the following diagram. Each variable can take an index from 1 to 250. They can be expressed by a number or an E parameter.

Field	Type	Variable name	Format	Meaning
TCODE	read/write	$TCODE(x)	T**	tool code
TOOLPOS*	read/write	$TOOLPOS(x)	S	Information regarding the tool position
TFAMCOL	read/write	$TFAMCOL(x)	S	reserved
TOOLTYPE*	read/write	$TOOLTYPE(x)	S	Information regarding the type of tools
TSTATUS	read/write	$TSTATUS(x)	S	tool status
TCNTRL	read/write	$TCNTRL(x)	S	tool control word
MAXLIFE	read/write	$MAXLIFE(x)	D	Initial life
REMLIFE	read/write	$REMLIFE(x)	D	real life
TUSER1	read/write	$TUSER1(x)	D	User variable # 1 for tool
TUSER2	read/write	$TUSER2(x)	D	User variable # 2 for tool
TUSER3	read/write	$TUSER3(x)	D	User variable # 3 for tool
TUSER4	read/write	$TUSER4(x)	D	User variable # 4 for tool
TOLOFNR	read/write	$TOLOFNR(x)	S	default tool offset number

x = page number or tool record number
S = short, D = double

** = these fields of the table may have various meanings depending on the configuration and on how they are used by the machine logic (see the 10 Series User Manual for details).*

*** = integer number (max. 12 digits)*

Example:
To read the code of the tool stored in record 35 key in the following function:

(DIS, $TCODE(35))

To assign to the tool stored in record 42 a life equal to 500 cycles, key in the following:

$LIFETYPE(42) = 3
$MAXLIFE(42) = 500

Tool Offsets Table

Each of the 300 pages or tool offsets in this table is made up of 9 fields. Each field is identified by a name which is shown in the table below. Variable indexes range from 1 to 300. An index may be a numerical value or another variable.

Field	Type	Variable name	Format	Meaning
TACTL1	read/write	$TACTL1(x)	D	current tool length # 1
TCMAXL1	read/write	$TCMAXL1(x)	D	maximum offset for length # 1
TCACTL1	read/write	$TCACTL1(x)	D	current offset for length # 1
TACTL2	read/write	$TACTL2(x)	D	current tool length # 2
TCMAXL2	read/write	$TCMAXL2(x)	D	maximum offset for length # 2
TCACTL2	read/write	$TCACTL2(x)	D	current offset for length # 2
TDIAMETER	read/write	$TDIAMETER(x)	D	tool diameter
TCACDIAM	read/write	$TCACDIAM(x)	D	diameter offset value
TORIENT	read/write	$TORIENT(x)	S	tool orientation

x = page number or offset number
S = short, D = double

Example:
If you want to supply the maximum requalification of the length # 1 of the corrector 137 with the value in the user # 3 variable in the Tool Table at record 35, write:

$TCMAXL1(137) = $TUSER3(35)

IMPORTANT

If the value of process variable TTR (Thoroidal Tool Radius) has been set or configured as 1, the fields relating to length # 2 are assumed to coincide with the size of the tool tip radius (in TCP and HSM applications).

PLUS I/O

PLUS inputs are arranged in 85 11-bit arrays followed by 427 16-bit arrays; outputs are arranged in 85 6-bit arrays followed by 427 16-bit arrays. Accordingly, a total of 7767 inputs and 7342 outputs has to be managed.

An array and its Input or Output are identified by an I or an O preceded by $ and followed by the bit index between brackets: $I(index) or $O(index), as in the User Table. For example, $I(35), $O(812).

The Input/Output arrays can be pictured as tables, as shown in the figures below.

Inputs (the first 85 arrays)

Arrays or groups of inputs may be numbered from 00 to 84, whereas the 11 bits in the array are numbered from 0 to 10. The index of the first input variable will be 0, whereas the index of the last input variable will be 934, as shown in the table below:

Array	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	bit 8	bit 9	bit 10
00	$I(0)	$I(1)	$I(2)	$I(3)	$I(4)	$I(5)	$I(6)	$I(7)	$I(8)	$I(9)	$I(10)
01	$I(11)	$I(12)	$I(13)	$I(14)	$I(15)	$I(16)	$I(17)	$I(18)	$I(19)	$I(20)	$I(21)
02	$I(22)	$I(23)	$I(24)	$I(25)	$I(26)	$I(27)	$I(28)	$I(29)	$I(30)	$I(31)	$I(32)
03	$I(33)	$I(34)	$I(35)	$I(36)	$I(37)	$I(38)	$I(39)	$I(40)	$I(41)	$I(42)	$I(43)

.....

.....

83	$I(913)	$I(914)	$I(915)	$I(916)	$I(917)	$I(918)	$I(919)	$I(920)	$I(921)	$I(922)	$I(923)
84	$I(924)	$I(925)	$I(926)	$I(927)	$I(928)	$I(929)	$I(930)	$I(931)	$I(932)	$I(933)	$I(934)

To calculate the index from the cell position, i.e. when the array and bit numbers are known, use this formula:

Index = array number x 11 + bit number

For example, to address bit 7 from array 03, the $I variable must be assigned the following index:

Index = 3 x 11 + 7 = 40

The resulting variable will be $I(40), as shown in the table.

To calculate the array and bit numbers from the index, divide the index by 11: the ---- will be the array number and the remainder will indicate the bit number.

For example, the $I(172) variable will address the 15 (172 / 11) array and the 7 (172 - 15 x 11) bit.

Inputs (after the 85th array)

The numbering of the individual input arrays or input groups goes from 85 to 511; the 16 bits within an array are numbered from 0 to 15. In these conditions the index of the first input variable will be 935, the index of the last variable will be 7766, as can be seen from the table below:

Array	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	bit 8	bit 9	bit 10	bit 11	bit 12	bit 13	bit 14	bit 15
85	$I(935)	$I(936)														$I(950)
86	$I(951)	$I(952)														$I(966)

.....

.....

Array	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	bit 8	bit 9	bit 10	bit 11	bit 12	bit 13	bit 14	bit 15
510	$I(7735)	$I(7736)														$I(7750)
511	$I(7751)	$I(7752)														$I(7766)

The formula used to define an index starting from the position of the cell (i.e., the array number and the bit number), is as follows:

index = (array number – 85) X 16 + 935 + bit number

For example, if we want to address bit 13 in array 86 (i.e., the grey-coloured cell), the index of variable $I must be determined as follows:

Index = (86 – 85) X 16 + 935 +13 = 964

In this case, the variable will therefore turn out to be $I(964), as shown in the table. Conversely, if we want to determine an array and a bit, knowing the relative index, we must subtract 935 from the index and then divide the value obtained by 16. By adding 85 to the whole number determined in this manner, we get the array number, while the difference will identify the bit within the array. For example: for variable $I (7736) we get array 510 ((7736-935)/16+85) and bit 1 ((7736-935) / 16).

Outputs (the first 85 arrays)

Arrays or groups of outputs may be numbered from 00 to 84, whereas the 6 bits in the array are numbered from 0 to 5. The index of the first output variable will be 0, whereas the index of the last output variable will be 509, as shown in the table below:

Array	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5
00	$O(0)	$O(1)	$O(2)	$O(3)	$O(4)	$O(5)
01	$O(6)	$O(7)	$O(8)	$O(9)	$O(10)	$O(11)
02	$O(12)	$O(13)	$O(14)	$O(15)	$O(16)	$O(17)
03	$O(18)	$O(19)	$O(20)	$O(21)	$O(22)	$O(23)

.....

.....

83	$O(498)	$O(499)	$O(500)	$O(501)	$O(502)	$O(503)
84	$O(504)	$O(505)	$O(506)	$O(507)	$O(508)	$O(509)

To calculate the index from the cell position, i.e. when the array and bit numbers are known, use this formula:

Index = array number x 6 + bit number

Examples:

1. To address bit 2 from array 03, the $O variable must be assigned the following index:

   Index = 3 x 6 + 2 = 20

   The resulting variable will be $O(20), as shown in the table.

2. To calculate the array and bit numbers from the index, divide the index by 16: the integer value obtained through the division will be the array number and the remainder will indicate the bit number.
   For example, the $O(214) variable will address the 35 (214/6) array and the 4 (214 - 35 x 6) bit.

Outputs (after 85th array)

The numbering of the individual output arrays or output groups goes from 85 to 511; the 16 bits within an array are numbered from 0 to 15. In these conditions the index of the first output variable will be 510, the index of the last variable will be 7341, as can be seen from the table below:

Array	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	bit 8	bit 9	bit 10	bit 11	bit 12	bit 13	bit 14	bit 15
85	$O(510)	$O(511)														$I(525)
86	$O(526)	$I(527)														$I(541)

.....

.....

Array	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	bit 8	bit 9	bit 10	bit 11	bit 12	bit 13	bit 14	bit 15
510	$O(7310)	$O(7311)														$O(7325)
511	$O(7326)	$O(7327)														$0(7341)

The formula used to define an index starting from the position of the cell (i.e., the array number and the bit number), is as follows:

index = (array number – 85) X 16 + 510 + bit number

For example, if we want to address bit 7 in array 86 (i.e., the grey-coloured cell), variable $O must be assigned the following index:

Index = (86 – 85) X 16 + 510 + 7 = 533

In this case, the variable will therefore turn out to be $O(533), as shown in the table. Conversely, if we want to determine an array and a bit, knowing the relative index, we must subtract 510 from the index and then divide the value obtained by 16. By adding 85 to the whole number determined in this manner, we get the array number, while the difference will identify the bit within the array. For example: for variable $O (680) we get array 95 ((680-510)/16+85) and bit 10 ((680-510) / 16).

FILE

10 Series ASSET can manage two types of files:

- ASCII files
- Binary (data) files

Except where otherwise stated, these files are located in default directory E:\FILE or F:\FILE, which ASSET can access. However it is also possible to read or write with ASSET in files belonging to other directories. In this case, the full pathname and extension of the file in question must be specified (e.g. E:\USER\FILE|FILE1.DAT).

These files are located in the E:\FILE or F:\FILE directories, which are accessible to ASSET. Their main characteristics are discussed in the sections that follow.

ASCII files

ASCII files may contain part programs, messages, etc. They are not formatted and have records of undefined length that must be written and read sequentially. ASCII files are given the .ASC extension by the system by default, in cases where the file was opened in read or write mode without the full pathname and extension being specified.

In an ASCII file it is not possible to read, write or search for a specific record. Maximum record length is 127 characters.

The system always starts reading an ASCII file from the first record, on which it positions automatically as it opens the file.

Writing an ASCII file means adding a new record after the last record in the file. To edit an ASCII file it is necessary to read it sequentially, make the necessary alterations and write a new file that will replace the old one.

Binary files

Binary files are formatted files that are used for storing the parameters managed by the system, i.e. variables, numbers, parameters, etc. Records have a fixed length that is declared when the first record is written. Binary files are given the .DAT extension by the system by default, in cases where the file was opened in read or write mode without the full pathname and extension being specified.

For example, to write in a file the tool code with the relevant TUSER1-4 user variables the binary file record will be as follows:

TCODE	TUSER1	TUSER2	TUSER3	TUSER4

When the system writes the first record it automatically creates a header that is invisible to the user and defines the format of all the file records.

Binary file records can be searched for and addressed for reading and writing operations.

A record can be up to 300 bytes long.
To calculate the record length in bytes it is necessary to know the length of each of the variables stored in the record. This information is listed in the table below:

Variable	Length in bytes
Boolean	1
Byte	1
Short	2
Long	4
Real	4
Long Real (Double)	8
String	Number of characters in the string. For example: SC0.4 --> 4 characters --> 4 bytes

Example:
if you write the following variables in a table:

(WRT,1,E1,E5,L3)

The record will occupy 24 bytes because there are 3 double variables and 3 x 8 = 24

For the following writing command:

(WRT,1,SC0.60,L5,"TEST")

the record length will be 72 bytes (60 + 8 + 4).

Unless otherwise specified in the reading or writing command, access to the file occurs at the first record for reading and after the last record for writing. After a record has been read or written the cursor will position to the subsequent record.
When reading or writing commands including ASSET binary files it is possible to declare a parameter that specifies the number of the target record.

COMMANDS

ASSET ensures full compatibility with the commands and instructions typical of the standard programming environment. For more information about parametric programming, please refer to Chapter 7 of the 10 Series Programming Manual.

Here is a list of syntactic and typographical conventions used throughout the manual:

The function name and the mandatory signs will be printed in boldface type. Mandatory parameters associated to a function will be indicated by an italicised mnemonics. Parameters may be enclosed between brackets.

[ ] Square brackets enclose optional parameters that may be omitted. Do not write these brackets in the block.

{ } Graphs enclose parameters that are alternative to one another and are separated by a |. Do not write graphs in the block.

| The vertical bar is the separator between two alternative parameters. Do not write this bar in the block.

Parameters may be expressed by letters, alphanumeric characters and numbers. Letters are used as keys or to identify the characteristic of a command. Alphanumeric characters identify file and variable names and are used for messages. Numbers identify parameters, multiple elements, etc. Non significant zeroes can be omitted.

Example:
(OPN, *channel*, *filename*, *{A|B}*, *{R|W}***)**

The three-letter code, the commas and the brackets are mandatory. A is alternative to B and R is alternative to W.

The sections that follow describe ASSET functions. For each function the following information is provided:

- Function name
- Meaning
- Syntax
- Mandatory and optional parameters
- Other characteristics and notes
- Examples.

LCK - Locking/Unlocking PLUS tables

This instruction permits to inform the logic or other applications that the specified table is being edited and is not accessible to them. After the table has been edited it is necessary to give another LCK in order to indicate that the table is available.

Syntax:

(LCK, *table number*, {**0** | **1**}**)**

where:

table number Is a number from 1 to 4 that identifies the write protected table. It can be expressed as a numerical value, a local variable or a system variable with the following mening:

1 axes table (origins)
2 tools table
3 tool offsets table
4 user table (L variables).

0 | 1 Write 1 to indicate that the specified table is write protected by ASSET. If the table is already reserved by another user, such as PLUS or Table Editor, an error message will be displayed:

NC270 PLUS Table already locked

This error can be managed from program by setting ERR = 1. For more information refer to Appendix B.

Write 0 to indicate that the table is no longer reserved for ASSET and can be accessed by other users.

Characteristics

$xxxxxx or L variables are always accessible and need not be write protected with ASSET. However, it is recommended to lock the table to make sure that no other user has access to the memory area written by ASSET.

Use LCK to unprotect a table only when it has been write protected from part program. Otherwise, you may unprotect a table that was reserved for other users (such as the machine logic).

Example:
The following example shows how to write variables in the Tools Table:

```
. . .
. . .
ERR = 1                 enables error management from part program
"LOOP"
(LCK,2,1)
(GTO,LOOP,STE=45)       45 = write protected table; waiting for unlocking command
$TCODE(1) = 12
$TUSER1(1) = 3.45
$TCODE(2) = 13
$TUSER1(2) = 6.21
(LCK,2,0)               unprotected tools table
ERR = 0                 disables error management from part program
. . .
. . .
```

END OF CHAPTER

INPUT COMMANDS

Input commands permit to create, customise and manage data entry windows. In particular, they make it possible to:

- Define the labels of the data entry fields and the destination of input data;
- Define the layout of the data entry window: window size and position, text position, etc.
- Define the background, foreground and text colors.

The three-letter codes that allow to program these features are as follows:

- DIF
- INP

DIF - Definition of a data field

DIF permits to define the size and color of a data entry window that has been programmed with the INP command. It must be used when the characteristics to be configured are different from the default ones.

Syntax

(DIF, *window number*, *first line*, *first column*, *first field line*, *first field column*, *empty lines*, *number of fields* [*,background color, text color, field color*]**)**

where:

window number identifies a predefined window. It is an integer from 1 to 10. DIF,0 invokes the default window, which cannot be modified.

first line is the screen line where the data entry window starts. It is an integer from 0 to 18.

Line 19 is always reserved for error messages.

first column is the screen column where the border of the window is positioned. It is an integer from 0 to 79 which must obviously be selected according to the length of the displayed data.

first field line is the line occupied by the first window field declared in the INP block. This field may be the name of the window or a data field.

It is a value in the 0 to 18 range which depends on the value of the *first line* parameter.

first field column is the screen column occupied by the first INP field and measures the distance from the screen border to the window border in characters. If programmed with the INP command the field will be automatically centered.

It is an integer from 0 to 79 which must obviously be selected according to the length of the displayed data.

empty lines is the number of empty lines that separates two subsequent comment or field lines.

It is an integer from 0 to 18 which must obviously be selected according to the available space and the length of the displayed data. If you write 0 there will be no empty lines.

If INP programs only one data entry field, the *empty lines* value is use for positioning the lower border with respect to the last displayed line.

number of fields specifies the number of fields to be displayed on the same line. It ranges from 1 to 4. The distance between two fields is 3 characters.

background color *(optional)* Defines the color of the data entry window background. If it is omitted, the default background color is **blue**. Allowed values are between 0 and 7. The meaning of these values is shown in the **Characteristics** section.

text color *(optional)* Defines the color of the name and text displayed in the data entry window. If it is omitted, the default color is **yellow**. Allowed values are between 0 and 7. The meaning of these values is shown in the **Characteristics** section.

field color *(optional)* Defines the color of the characters and markers entered from the keyboard on the fields programmed with INP. If it is omitted, the default color is **white**. Allowed values are between 0 and 7. The meaning of these values is shown in the **Characteristics** section.

Characteristics

- The parameters configured in the *background color*, *text color* and *field color* are as follows:
  1. black
  2. blue
  3. green
  4. cyan
  5. red
  6. magenta
  7. yellow
  8. white
- The window layouts established by DIF are retained in the system memory after a reset.

- All the fields in the DIF instruction (included the colors) may be programmed directly, i.e., by writing numerical values, or through local and system variables.
- ASSET includes a default DIF instruction. If no DIF is declared in the INP instruction, the default DIF displays the following window. This window implies that INP has declared 8 fields plus the WINDOW NAME. The numbers of lines and columns are in italics. Default window parameters are as follows:

First line:	0
First column:	0
First field line:	1
First field column:	1
Empty lines:	1
Number of fields on a line:	1
Background color:	Blue
Label color:	Yellow
Field color:	White

These parameters are programmed by the following DIF format:

(DIF,1,0,0,1,1,1,1,6,7,4) *(the last three digits can be omitted)*

Examples:

Let's assume that you need to program a data entry window (WINDOW #3) that is made up of three fields and one window name, like the one shown in the figure below:

The number of fields, the field lengths, the characters in the window name and the fields labels (FIELD 1 through 3) are specified in the INP file.

Let's also assume that the window must be positioned as follows:

distance from the upper border to the status line:	3 lines
distance from the window border to the screen border:	10 characters
distance from the window name to the upper window border:	1 line
distance from the first field to the left window border:	1 character
number of empty lines between fields:	1
number of fields per line:	1

The format of the DIF command will be:

(DIF,3,3,10,1,1,1,1)

To display blue comments and red inputs on a green background, the format of the DIF command will be:

(DIF,3,3,10,1,1,1,1, 2,1,4)

These numbers may be partially or totally replaced by local or system variables. For example, if the contents of the *first line*, *first column,* and *first field column* fields are stored in variables E1, E2 and E3, the format of the above command must be:

(DIF,3,E1,E2,1,E3,1,1)

If a great many fields must be displayed in one window it is possible to write several fields on the same line with the following format:

(DIF,5,3,5,1,3,1,3)

The 6 fields programmed with INP will display this window:

		WINDOW NAME			
FIELD 1:	**xxxxxxxxxx**	**FIELD 2:**	**xxxxxxxxxx**	**FIELD 3:**	**xxxxxxxxxx**
FIELD 4:	**xxxxxxxxxx**	**FIELD 5:**	**xxxxxxxxxx**	**FIELD 6:**	**xxxxxxxxxx**

INP - Manual data input

This command permits to link manual data inputs to their destination variables. It is used for programming the name of the data entry window and the field labels. Data will be arranged according to the window size and layout established with DIF. If the default DIF suits the user needs the window number may be omitted.
When INP is executed, it enables the MDI mode.

Syntax

(INP, [**N***window number*,] [***T*** *WINDOW NAME*,] *label*, *label length*, *variable*
[,label, *label length*, *variable*]
[,label, *label length*, *variable*]
[,label, *label length*, *variable*]
[,label, *label length*, *variable*]
[,label, *label length*, *variable*]
[,label, *label length*, *variable*]
[,label, *label length*, *variable*]**)**

where:

window number *(optional)* It is the number of data entry window established with DIF.
It may be a number in the 0 to 10 range written in the command line or read from a variable.
0 is the default window number, which is associated to the window described by the DIF command.

window name *(optional)* It is the label displayed on the first window line. It may be a character string enclosed between quotes (' or ") or a string variable. For example, SC0.10.
Its maximum length is 40 characters. The name is automatically centered on the first window line.

Each data entry window may contain up to 8 fields. Each field is configured by the following parameters:

label Is the name of the field. For example: *tool*, *feedrate*, etc.
It may be a character string enclosed between quotes (' or ") or a string variable. For example, SC0.10.
The label length may range from 0 and the value specified by the *label length* parameter. The maximum label length is 40 characters.

label length Specifies the maximum length of the field label. The programmer may use this parameter to arrange various fields on one line regardless of the length of the label string.
It may be a number written in the command line or read from a variable.

variable It is a local or system variable whose current value is displayed on the data entry window and updated with manual data input.

Characteristics

When the system reads an INP block it switches to the MDI mode and displays the programmed data entry window with the current variable values specified in the INP command.

The size, position and colors of the data entry window are programmed by the DIF instruction whose number is specified in the INP command. If this number is omitted, the data entry window will have the default characteristics.

Displayed values may be edited with the same procedures described for the other data entry windows.

To exit from the data entry window and quit the MDI mode, press [**ESC**], [**Enter**], [**Page Up**], [**Page Down**], [**Home**] and [**End**].

By pressing [**ESC**], [**Page Up**], [**Page Down**], [**Home**] and [**End**] the input window is closed but the values of the programmed variables are not updated with the values you may have entered. In this case, the STE system variable assumes a value that indicates the type of key with which the input window has been closed. The possible values are the following:

Type of key	STE value with ERR = 0
ESC	20
PAGE UP	22
PAGE DOWN	23
HOME	24
END	25

Press [**Enter**] to close the window saving the alterations to programmed values. If a format error is detected in the assignment (for example, if VEF is assigned a value outside the 0.1 - 8.0 range) it can be managed from program by setting the ERR variable to 1 (refer to Chapter 6, "Error Management from Part Program").

The system automatically synchronises (#) the INP command to the execution of the program (refer to Chapter 1 in the Programming Manual).

NOTE:
The label length must be programmed according to the number of fields that need to be displayed on the same line. To well align fields on the same columns it is advisable to assign the same *label length* to all the fields. This value must be equal to or greater than that of the longest label.

IMPORTANT

The system will not visualise an error message as long as it can display at least one digit from the current or input data assigned to the right-most fields on the window.

This should be remembered when programming position of the window on the screen and the data fields in it and the label length.

It is recommended to display all the digits in the various data entry fields.

Examples:

E1 = 1000; feedrate
(INP,'FEEDRATE',15,E1)

The above commands permit to visualise a data entry window whose size and colors are specified by the default parameters in the DIF command. This window will visualise the "FEEDRATE" label followed by a 7 character space (7 = 15 - the actual label length) and the current value of the E1 variable (1000 in the example). This value may be altered from keyboard or confirmed with [**Enter**].

(INP,T'INPUT TEST','FIELD 1',20,SC0.20)

This command permits to display a data entry window whose name is "INPUT TEST" and a 20 character field to display the contents of the SC0.20 variable preceded by the "FIELD 1" label.

(INP,N1,'FIELD 1',20,SC0.20,"FIELD 2", 20, E3,'FIELD 3',20,@PLUS)

This command permits to display a data entry window whose characteristics are specified by DIF 1, without name, and in which the contents of the SC0.20, E3 and @PLUS variables are preceded by the "FIELD 1", "FIELD 2" and "FIELD 3" labels.

END OF CHAPTER

SCREEN MANAGEMENT COMMANDS

10 Series displays all the parameters and data requested for system operation on a screen made up of 25 lines. Each line has 80 characters. Screen lines are divided as follows.

- 1 status information line
- 21 lines (data area) displaying process and task information.
- 3 softkey lines.

The status line and the softkey lines are managed by the system. The data area may visualise two screen types:

- **Full screens:** occupy 21 80-character lines.
- **Quadrants:** occupy 10 39-character lines.

The screens used on the 10 Series are of two types:

- **System screens:** the information in these screens is displayed through the system software; see the user manual for significance of this information.
- **User screens:** contain information displayed by the user by way of the machine logic functions, or using the ASSET OUT instruction.

Code	Screen Type	Description	Modified with ASSET?
1	(*)	Main process screen	No
2	(*)	Main logic screen	No
3	(*)	Axes coordinate in large font	No
4	(*)	Logic screen 1	Yes
5	(*)	Logic screen 2	Yes
6	(*)	Logic screen 3	Yes
7	(*)	Logic screen 4	Yes
8	(*)	Additional screen 1	No
9	(*)	Additional screen 2	No
10	(*)	Additional screen 3	No
11	(*)	Additional screen 4	No
12	(*)	Additional screen 5	No
25	(**)	Logic quadrant 1	Yes
26	(**)	Logic quadrant 2	Yes
27	(**)	Logic quadrant 3	Yes
28	(**)	Logic quadrant 4	Yes
35	(**)	System status quadrant	No
36	(**)	Axes coordinates quadrant	No
37	(**)	Part program display quadrant	No
38	(**)	Process status quadrant	No
39	(**)	Programmed codes status quadrant	No
40	(**)	Axes offsets quadrant	No

(*) Full screen size
(**) Quadrant

ASSET offers two, three-letter screen management commands:

- OUT for definition of the following line parameters:
  - position
  - contents (type of data, length, etc.)
  - background and foreground colors
  - font.
- SCR for selection of the screen associated with a given process.

OUT - Screen line parameters

OUT defines location, length, colors and fonts on a line basis.

Syntax

(OUT, *Screen code*, *line*, *column*, *background color*, *data color,* [*/length,*] *variable, ... ,variable***)**

where:

Screen code	Is the screen number provided by the table on page 3-2. Allowed values are 4, 5, 6, 7, 25, 26, 27, 28, which correspond to the tables that can be modified with ASSET and described in the previous table. This parameter can be written with a numerical format or read from a local or system variable.
line	Is the line number. It ranges from 0 to 20 in full size screens or from 0 to 9 in quadrants. This parameter can be written with a numerical format or read from a local or system variable.
column	Is the number of the column where the line starts. It ranges from 0 to 79 in full size screens or from 0 to 38 in quadrants. If the maximum value is selected, i.e. 79 or 38, only one character will be displayed. This parameter can be written with a numerical format or read from a local or system variable.
background color	Is the background color of the selected line. It ranges from 0 to 7 (refer to the Characteristics paragraph in this section) and may be written with a numerical format or read from a local or system variable.
data color	Is the color of the contents displayed on the selected line. It ranges from 0 to 15 (refer to the Characteristics paragraph in this section) and may be written with a numerical format or read from a local or system variable.
/length	*(optional)* Is the length of the selected line. It ranges from 1 to the maximum number of characters displayed between the selected *column* number and the last column configured for that screen. It may be written with a numerical format or read from a local or system variable. In any case the slash (/) is mandatory.
variable,...,variable,	Is the list of characters or numerical variables to be displayed on the line. Up to 8 variables of the same or of different types are allowed. Strings of characters must be enclosed between quotes (' or ").

Characteristics

- The allowed values for *background color* and *data color* and their meaning are shown in the table below:

Parameter value	Background color	Data color/display mode
0	Black	Black / Normal
1	Blue	Blue / Normal
2	Green	Green / Normal
3	Cyan	Cyan / Normal
4	Red	Red / Normal
5	Magenta	Magenta / Normal
6	Yellow	Yellow / Normal
7	White	White / Normal
8	Not allowed	Black / Blinking
9	Not allowed	Blue / Blinking
10	Not allowed	Green / Blinking
11	Not allowed	Cyan / Blinking
12	Not allowed	Red / Blinking
13	Not allowed	Magenta / Blinking
14	Not allowed	Yellow / Blinking
15	Not allowed	White / Blinking

- If both *background color* and *data color* are set to zero, the system will nevertheless default to a white-on-black screen.
- To program a blank between variables when the line displays several variables, write the blanks between quotes (for example, " ") or store them in a string variable, which may be repeated several times in the same command.
- The contents specified in the OUT block must belong to the same line and cannot be longer than the line. To display more data it is necessary to program more OUT's.
  In addition, several parameters may be displayed on the same line by programming different OUT blocks with appropriate *line*, *column* and *length* values so as to prevent overlapping.

Example:
The following command:

(OUT,5,3,10,2,8,/60,"TEST VAR E1: ",E1," "TEST VAR SC0.10: ",SC0.10)

allows to display the following screen:

A 60-character green bar starting on column 10 appears on line 3. Its contents will be displayed in black and blinking. Note that the strings programmed between quotes end by blanks, which prevents parameters from being displayed as an uninterrupted line.
This is screen 5 and is associated to logic screen 2.

With the following variables:

SN1 = 5
SN2 = 3
SN3 = 10
SN4 = 2
SN5 = 8
SN6 = 60
SC30.12 = "TEST VAR E1: "
SC5.19 = "TEST VAR SC0.10: "

the OUT command could be rewritten as follows:

(OUT,SN1,SN2,SN3,SN4,SN5,/SN6,SC30.12,E1," ",SC5.19,SC0.10)

SCR - Screen selection

SCR selects the video screen associated to the current process.

Syntax

(SCR, *Screen code [, background color for initialization]***)**

where:

Screen code	Is the screen number provided by the table on page 3-2. It can be written with a numerical format or read from a local or system variable.
background color for initialization (optional)	This parameter is taken into consideration only if the screen code represents a screen that can be modified by ASSET. In this case, the screen will be initialized with the color specified and all data previously on that screen will be deleted. If this parameter is not specified, the screen will not be initialized.

Characteristics

If the system reads the command but no screen has been configured for the selected process the visualised screen will not change. On the contrary, if the specified screen has been configured, it will be immediately visualised.

The background color for initialization is defined by a number from 0 to 7, as illustrated in the table below:

1. Black
2. Blue
3. Green
4. Cyan
5. Red
6. Magenta
7. Yellow
8. White

Example:

Let's assume that the system controls three processes and that the following screens have been associated to them:

Process 1:	Screens 1, 3, 8
Process 2:	Screens 1, 2, 9, 25
Process 3:	Screens 1, 3, 8, 9

In addition, let's assume that the (SCR,8) command was programmed in process 2 when process 1 is selected. The system will display screen 8, which is associated to process 1. If the same command were programmed when process 2 is selected, no changes would be noticed.

END OF CHAPTER

FILE MANAGEMENT COMMANDS

The system may control ASCII and binary data files. The characteristics of these files have been discussed in Chapter 1.

ASSET file management commands make it possible to:

- open a file and link it to a management channel.
- read a binary file or a record sequentially.
- write or edit a binary file or a record sequentially.
- close any file opened by ASSET.
- delete a file.
- copy an ASCII file to the directory of programs.
- allocate and release physical devices for file access.
- copy a file

The functions quoted above are executed by the following three-letter commands, described in this chapter:

- OPN
- REA
- WRT
- CLO
- DEL
- INS
- GDV
- RDV
- CPY

OPN - Opening a file

This instruction permits to open a file and link it to a channel number. It also specifies whether the file will be opened for reading or writing.

The channel is a marker that facilitates access to the file from other ASSET file management commands.

Syntax

(OPN, *channel*, *filename*, {**A** | **B** | **P**}, {**R** | **W**}**)**

where:

channel Is a number from 1 to 5 that identifies the channel. A channel is another file that will be subsequently used by the REA or WRT commands.

It can be written in the block with a numerical format or read from the specified variable.

filename Is the name of the file (max. 8 characters) associated to the channel. It may be an alphanumerical string or the name of the character variable that contains the file name. When it is a variable it must be preceded by a '?'.

The complete path and extension may be specified in the filename.

{A | B | P} Defines the type of file: ASCII (A), binary (B) or if a file belongs to the system directory of programs (P).

{R | W} Specifies the type of access:

R: reading

W: writing

Characteristics

In each process it is possible to open 5 simultaneous files (channels).

If neither the path nor extension are specified in filename, the system provides the file extension .DAT for binary files and .ASC for ASCII files. The file will be read or written in the default directory (E:\FILE or F:\FILE).

If a file is opened for reading it must have been previously created and saved in the system. If the file is opened for writing it need not exist in the system already and will be written in the working directory when the session is closed.

Examples:

1. The TEST ASCII file in directory E:\FILE may be opened for reading and associated to channel 1 by the following command:

   (OPN,1, TEST,A,R)

2. If the name of the DATAFILE file is written in the SC0.8 variable:

   E1=2

   SC0.8 = "DATAFILE"

   and the channel number is stored in the E1 variable, DATAFILE may be opened for reading and associated to channel 2 with the following command:

   (OPN,E1,?SC0.8,B,W)

   The full pathname of the file is E:\FILE\DATAFILE.DAT.

3. With command:

   (OPN,1,E:\TEST.KKK,A,R)

   File ASCII E:\TEST.KKK is opened for reading.

WRT - Writing a file

This command permits to write a record in the file specified and linked by a previous OPN block. Data will be written sequentially, in the same order in which they are specified in the WRT block.

Syntax

(WRT, *channel***,**[**R** record**,**] variable [, ...,variable]**)**

where:

channel Is the channel number linked to the file by the previous OPN command. In addition, OPN must have opened the file for writing (W).

record *(optional)* is a number or a variable that identifies the file record in which the parameters must be written. It is always preceded by an **R.**

This parameter is accepted only if the file is binary.

If *record* is omitted, data will be written in the record used for the latest reading and the system will position at the beginning of the subsequent record. If this is the first writing instruction after OPN, data will be written on the first file record.

The maximum number of records that can be specified is 65535.

variable Is the variable(s) (max. 30) whose data must be written in the file records.

If the destination file is an ASCII file only character variables will be accepted.

If the destination file is binary, it is necessary to use the same number and type of variables written in the first file record.

Characteristics

If the file is binary (i.e. has been opened with the B marker) and the number of records in the file has not been specified in WRT, data will be written in the record that follows the record written during the last writing session. If there have been no previous writing sessions, data will be written in the first file record.

The first data written on an empty binary file are the record type, the number of variables, the types of variable and the sequence in which variables must be arranged. This record provides the access pattern to be followed by subsequent reading or writing operations, in which the same number, type and sequence of variables must be used.

For example, if the first writing operation was programmed by the following command:

(WRT,1,E1, SC3.4,@PLUS_SHORT(1))

that is, 1 double variable, 4 character variables and 1 short variable, subsequent writing or reading commands will resemble the following:

(WRT,1,L1,"JOE",@PLUS_SHORT(4))
(REA,1,SN1,!TEST.4CH,@PLUS_SHORT(6))

If other formats, types or numbers of variables are used or if variables are arranged in a different sequence, an **NC248 Data file access error** will occur.

In ASCII files data are always written in the last file record. To edit an ASCII file you must open it, read it sequentially record by record, make the necessary alterations and write the records in a new file under a new name.

Alterations will become effective only after the file has been closed. To read a just-altered file you must first close it with CLO and then open it for reading.

Examples:

(WRT,1,SC5.80)

This command writes in the ASCII file associated to channel 1 the first 80 characters stored in the SC5 variable.

(WRT,3,R12,E7,E8,E9)

This command writes in record 12 of the binary file associated to channel 3 the data stored in variables E7, E8 and E9. The pointer remains at the beginning of record 13.

REA (RED) - Reading a file

This commands reads a record from the file declared by the previous OPN command and specifies the channel to which the file is associated.

Syntax

(REA, *channel***,[R** record**,**] variable [, ...,variable]**)**

where:

channel Is the channel number linked to the file by the previous OPN command. In addition, OPN must have opened the file for reading (R).

record *(optional)* is a number or a variable that identifies the file record whose parameters must be read. It is always preceded by an **R.**

This parameter is accepted only if the file is binary.

If *record* is omitted, the system will read the record used for the latest reading and positions the pointer at the beginning of the subsequent record. If this is the first reading instruction after OPN, data will be read from the first file record.

The maximum number of records that can be specified is 65535.

variable Is the variable(s) (max. 30) where read data must be written.

If the destination file is an ASCII file only one character variable will be accepted. The length of this variable must be such as to accommodate the data.

If the destination file is binary, it is necessary to use the same number and type of variables written in the first file record. Otherwise, an NC248 error will occur.

Characteristics

The command may be expressed indifferently as REA or RED.

If the file is binary (i.e. has been opened with the B marker) and the number of records in the file has not been specified in REA, data will be read from the record that follows the one used during the last reading session.

In an ASCII file, data will be read from the record on which the pointer remained positioned since the last reading session.

In both cases, if there have been no previous reading sessions, data will be read from the first file record.

Examples:

(REA,1,SC5.80)

This command reads from the ASCII file associated to channel 1 the first 80 characters stored in the SC5 variable. (The maximum allowed string length is 80 characters).

(REA,3,R12,E7,E8,E9)

This command reads data from record 12 of the binary file associated to channel 3 and writes them in variables E7, E8 and E9. The pointer remains at the beginning of record 13. The sample record of this file is made up of 3 double variables.

CLO - Closing a channel

This command closes the file associated to the specified file. If the channel is omitted, all the ASSET files opened by the process will be closed.

Syntax

(CLO [, *channel*]**)**

where:

channel *(optional)* Is the channel number linked to the file in the OPN instruction.

Characteristics

If the *channel* is omitted, CLO will close all the ASSET files that have been opened by the process.

An alternative method for closing all the ASSET files that have been opened by the process is a system reset.

Examples:
(CLO, 2)

This command closes the file associated to channel 2.

(CLO)

This command closes all the ASSET files that have been opened by the process.

DEL (CAN) - Deleting a file

This command permits to delete the specified file from the working directory or the part program directory.

Syntax

(DEL, *filename*, ***A***|***B***|***P***)

where:

filename Is the name of the file.

Full pathname and extension may be specified in *filename*. If the path is not specified, the file indicated present in the ASSET working directory (markers A or B) or the part program file (marker P, with 48 characters max.) will be deleted.

If *filename* is preceded by a **?** the name of the file is stored in a local or system variable.

A|***B***|***P*** These markers describe the characteristics of the file:

A delete the ASCII file (.ASC extension) indicated from the working directory, if the path or extension were not specified

B delete the BINARY (data) file (.DAT extension) indicated from the working directory, if the path or extension were not specified

P delete the part program file specified from the system directory of programs.

If the full pathname or extension is specified in filename, this parameter is not taken into consideration.

Characteristics

The triliterals DEL or CAN, which may be used as alternatives, delete the specified file only if it is not currently used by the system. If the file is a part program it will be deleted only if it has not been selected by any process.

If the specified file does not exist, the **NC238 File not found** error code will be displayed. See appendix B for a detailed description of how to manage this error from part program.

Examples:
The following command

(DEL,TEST,A)

deletes the TEST ASCII file from the ASSET working directory.

This command

(DEL,?SC0.8,B)

deletes the binary command whose name is stored in the SC0.8 variable.

INS - Saving an ASCII file in the directory of programs

This command permits to save the specified ASSET ASCII file in the directory of programs.

Syntax

(INS, *filename***,** *filenamepp***,** *directory***)**

where:

filename Is the name of an ASCII file stored in the working directory, or another directory if the full pathname was specified. It can be written as a string or read from a text variable, in which case it is preceded by a question mark (?).

filenamepp Is the name under which the ASCII file will be saved in the directory of programs. It can be a 48 character string or the content of a variable, in which case it is preceded by a question mark (?).

directory Is a code (written as a numerical value or read from a local or system variable) that specifies the logic subdirectory in which the ASCII file must be written:

Allowed values are:

1. PROGRAM directory
2. USER directory
3. SYSTEM directory
4. OEM directory

Characteristics

If no path or extension were specified, the file is taken from the default working directory (F:\FILE) with the .ASC extension.

The part program file will be saved in the directory of programs without extension.

The file inserted in the directory of programs is deleted from the directory it was in before the INS command.

Examples:

(INS,SAMPLE,PPTEST,0)

This command extracts the SAMPLE ASCII file from the F:\FILE directory and saves it in the PROGRAM directory with the PPTEST name.

(INS,?SC7.8,?!USER3,4CH,1)

With this command the file whose name is stored in the SC7.8 variable will be saved in the USER directory under the name stored in the USER !USER3.4CH variable.

CPY - Copy a file

This command copies the file specified in the working directory or in the part program directory of programs.

Summary

(CPY, *filename1*, *filename2*[,A/B/P,A/B/P[,directory]]**)**

where:

filename1 Is the name of the file to be copied.

In filename1 you can specify the full file name or the file extension. If the path is not specified only those files will be considered which are in the work directory of ASSET (key A or B) or in the part program directory if the file contains a part program (key P).

If filename1 is preceded by the key "?" this indicates a local or system variable containing the name of the file to be copied.

filename2 Is the name of the destination file.

Filename2 follows the same rules described for filename1.

A/B/P (optional). These are the keys which specify the features of the files with filename1 and filename2.

In particular:

A indicates that the specified file is an ASCII file (with extension .ASC) belonging to the working directory (if the path or extension are not specified).

B indicates that the specified file is a binary file (with extension .DAT) belonging to the working directory (if the path or extension are not specified).

P indicates that the specified file belongs to the part program directory.

directory (optional). A numeric code or the contents of a local or system variable which indicates the logical directory where to copy the specified file.

Its value is 0 to 3 with the following meanings:

0 directory “PROGRAM”

1 directory “USER”

2 directory “SYSTEM”

3 directory “OEM”

Characteristics

If the path is not specified, the system assumes as the default working directory (F:\FILE). In addition, if no extension is specified, the system will add an extension based on the basis of the specified key: .DAT for binary files (key B) or .ASC for ASCII files (key A).

Example:

Initial situation: file SOURCE.DAT is present in directory F:\FILE

correct syntaxes for Asset copy: (CPY , SOURCE.DAT , DEST.DAT)

(CPY ,SOURCE , DEST , B , B)

final situation: file SOURCE.DAT will be copied as DEST.DAT in the F:\FILE directory

In cases where the path has been specified it is not necessary to enter a key as the file will be taken as indicated.

The parameter for the directory is significant if filename1 does not belong to the program directory and is to be copied into it or if filename1 belongs to the program directory (key P) and is to be copied into a logical directory different to the one to which it belongs.

END OF CHAPTER

PROCESS MANAGEMENT COMMANDS

The ASSET commands described in this chapter (RTP, ROP, GPS, RAP and PLS) permit to read the following parameters of the selected process:

- current or programmed tool number
- current or programmed tool offset number
- process state and mode
- axis coordinates.

Data read by these commands are sent to the process defined by the PRO block. Although PRO is not an ASSET but a standard command, it has been included in this chapter for the sake of clarity.

The process number may also be specified in the ASSET reading command. If it is omitted, the system will default to the process specified by PRO. If no PRO has been programmed, the data will be sent to the same process that is executing the ASSET command.

The above parameters can be read using the three-letter commands listed below and described in this chapter:

- PRO
- RTP
- ROP
- GPS
- RAP

In addition to the above commands, there is also a general purpose command with which SW PLUS variables can be read both inside and outside a process. This command is:

- PLS

PRO - Definition of the default process

This standard programming instruction permits to specify a default process. This process includes both the synchronization commands and the ASSET commands described in the chapter.

Syntax

(PRO, *process number***)**

where:

process number Is a number from 1 to 20 that identifies the process. It can be written as a numerical value or read from a local or system variable.

IMPORTANT

By resetting the system or invoking another program it is possible to cancel the preceding default process.

RTP - Reading the number of the programmed or active tool

This instruction permits to read the number of the active or programmed tool from the specified process (or from the default process defined with PRO) and then write the number in the specified variable.

Syntax

(RTP [**,P** *num pro*]**,** {**N** | **A**}**,** *variable***)**

where:

num pro	*(optional)* Is the number of the process that includes the tool number. If it is omitted, the tool number will be read from the default process specified with PRO. It is a number from 1 to 20. It may be written as a numerical value or read from a local or system variable.
{N | A}	Specifies whether tool number refers to the current (A) or the programmed (N) tool.
variable	Is the number of the variable in which the tool number must be written.

Characteristics

If the default process has not been defined and the *Pnum pro* parameter is omitted, the system will read the tool number from the current active process.

Examples:

(RTP,N,E1)

This command reads the number of the programmed tool from the current process and writes it in the E1 variable.

(RTP,P3,A,E3)

This command reads the active tool number from process 3 and writes it in the E3 variable.

ROP - Reading the number of the programmed or active tool offset

This instruction permits to read the number of the active or programmed tool offset from the specified process (or from the default process defined with PRO) and then write the number in the specified variable.

Syntax:

(ROP [**,P** *num pro*]**,** {**N** | **A**}**,** *variable***)**

where:

num pro	*(optional)* Is the number of the process that includes the tool offset number. If it is omitted, the tool number will be read from the default process specified with PRO. It is a number from 1 to 20. It may be written as a numerical value or read from a local or system variable.
{N | A}	Specifies whether tool number refers to the current (A) or the programmed (N) tool offset.
variable	Is the number of the variable in which the tool offset number must be written.

Characteristics

If the default process has not been defined and the *Pnum pro* parameter is omitted, the system will read the tool number from the current active process.

Examples:

(ROP,N,SN3)

This command reads the number of the programmed tool offset from the current process and writes it in the SN3 variable.

(ROP,P2,!ABC(1))

This command reads the active tool offset number from process 2 and writes it in the !ABC(1) user variable.

GPS - Reading the process state, sub-state or mode

This instruction permits to read the state, sub-state or mode of the specified or the default process and then write it in the specified variables.

Syntax:

(GPS [**,P** *num pro*]**,** *variable***,** *variable***,** *variable***)**

where:

num pro — *(optional)* Is the number of the process. If it is omitted, the state, sub-state or mode will be that of the default process specified with PRO.

It is a number from 1 to 20. It may be written as a numerical value or read from a local or system variable.

variables — Are three numerical variables in which the process state, sub-state and mode are written.

The meaning of these numerical values is as follows:

Value	State
1	IDLE
2	RUN
3	HOLD
4	RUNH
5	HRUN
6	ERROR
7	WAIT
8	RESET
9	EMERGENCY
10	INPUT

Value	Sub-state
1	DUMMY
2	reserved
3	reserved
4	BLOCK RETRACE
5	ACTIVE RESET
6	MAS
7	reserved
8	
9	
10	

Value	Mode
1	MDI
2	AUTO
3	STEP
4	MANUAL JOG
5	INCR. JOG
6	RETURN PROFILE
7	HOME
8	HANDWHEEL
9	
10	

For a detailed description of the states, sub-states and modes listed in the table refer to 10 Series User Manual.

Characteristics

If the default process has not been defined and the *Pnum pro* parameter is omitted, the system will read the tool number from the current active process.

Examples:

(GPS,E1,E2,E3)

This instruction reads the state, sub-state and mode of the current process and writes them in variables E1, E2 and E3.

(GPS,P4,!UT(1),!UT(2),!UT(3))

This instruction reads the state, sub-state and mode of the current process and writes them in user variable !UT(1 ÷ 3).

RAP - Reading the axis coordinates

This instruction permits to read the position of the specified axis.

Syntax

(RAP [**,P** *num pro*]**,** *axis name***,** *type of coordinate***,** *variable***)**

where:

num pro (optional) Is the number of the process. If it is omitted, the system will read the position of the axis in the default process specified with PRO.

It is a number from 1 to 20. It may be written as a numerical value or read from a local or system variable.

axis name Is the ASCII name of the axis whose position must be read by the system.

type of coordinate It is a number from 1 to 4. It may be written as a numerical value or read from a local or system variable. The meaning of this value is as follows:

0 ABSOLUTE INTERPOLATED coordinate (distance to the axis zero)
1 PROGRAMMED coordinate being executed
2 distance TO GO (DTG)
3 ABSOLUTE REAL coordinate (distance to the axis zero)
4 ERROR coordinate
5 INTERPOLATED coordinate (includes the origins and the tool offset)
6 REAL coordinate (includes the origins and the tool offset)
7 PROGRAMMED coordinate while RAP instruction is being analysed

variable Is the number or the local or system variable where the axis position will be written.

Characteristics

The coordinates read by the RAP triliteral are expressed in the unit of measure configured for the process. Rotary axes coordinates are always expressed in degrees.

IMPORTANT

If the axis name is P, you have to insert the process number (even if it is the default process). In type 7 readings, the command takes into account the visualisations and DAN's currently active on the programmed axis, if any.

Example: the following example makes it possible to read the current coordinates of axis X after a rototranslation and to save it in variable E0.

```
(UPR , 0 , XYZ , UVW)

(DAN , XU , YV , ZW)

41 X 200F 1000
```

(RAP , X , 7, E0) reads the current coordinate of virtual axis 4.

Examples:

To read the absolute real coordinate of the X axis in the default process, write it in the E4 variable and visualise it on the screen program the following instructions:

(RAP,X,3,E4)
(DIS,E4)

To read the distance to go for the Z axis in process 3 and write it in the SN5 variable program the following:

(RAP,P3,2,SN5)

PLS - Reading PLUS SW variables

This instruction permits to read bits or words from PLUS SW variables and write them in the specified local or system variables.

Syntax

(PLS, *SW number*, *bit number*, [**&** *mask*], *variable*
[,*SW number*, *bit number*, [**&** *mask*], *variable*]
[,*SW number*, *bit number*, [**&** *mask*], *variable*]
[,*SW number*, *bit number*, [**&** *mask*], *variable*]
[,*SW number*, *bit number*, [**&** *mask*], *variable*]
[,*SW number*, *bit number*, [**&** *mask*], *variable*]
[,*SW number*, *bit number*, [**&** *mask*], *variable*]
[,*SW number*, *bit number*, [**&** *mask*], *variable*]
[,*SW number*, *bit number*, [**&** *mask*], *variable*]
[,*SW number*, *bit number*, [**&** *mask*], *variable*])

where:

SW number Is the number of the SW variable. It may be written as a numerical value or read from a local or system variable.

bit number Is the number of a bit in the specified SW variable.

It is a number from 0 to 16 that may be written as a numerical value or read from a local or system variable specified in the command. Values from 0 to 15 indicate single bits, whereas 16 indicates the whole word.

mask *(optional)* This parameter can be programmed only if *bit number* is 16 (whole word). *mask* provides a decimal description of the bits of the word setting those be read to 1. It is applied in an AND to the contents of the specified SW variable before saving its in the specified local or system variable.

It may be written as a numerical value or read from the local or system variable specified in the instruction.

variable Is the local or system numerical value in which the contents of the specified bit or word must be written.

Characteristics

This instruction may specify as many as 10 readings, which may be bits of the same SW variable, or the bits/word from different variables.

To calculate the mask use this procedure:

1. Write the word to be used as mask setting the bits to be read to 1. For example, to read bits 0, 5, 7 and 14 write the following word:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**0**	**0**	**1**

2. The decimal value of this word will be:

   $2^0 + 2^5 + 2^7 + 2^{14} = 1 + 32 + 128 + 16384 = 16545$

   The *mask* value will be &16545.

The meaning of the bits and words of each SW variable is described in Chapter 3 (sections "System state flags" and "Process state flags") of the PLUS Application Manual.

Examples:

1. The following command

   (PLS,1,16, E1)

   will save the active Security level in E1.

2. The following command

   (PLS,23,2,SN2)

   writes 1 in the SN2 variable if G18 is active and writes 0 if G18 is not active.

3. With the following command

   (PLS,26,16,&1792,E0)

   0 will be written in E0 if there is no probing cycle in process 1.

4. With this command

   (PLS,27,0,E1,47,0,E2,67,0,E3,87,0,E4)

   The contents of the specified command will be as follows:

   E1 will be 0 if no canned cycle is active in process 1.
   E2 will be 0 if no canned cycle is active in process 2.
   E3 will be 0 if no canned cycle is active in process 3.
   E4 will be 0 if no canned cycle is active in process 4.

END OF CHAPTER

SERIAL LINE MANAGEMENT COMMANDS

Serial line management is achieved by means of a number of commands allowing you to:

- Enable and characterise the serial lines by defining the communication parameters and all the terminators needed for data and message management.
- Transmit data of any kind; these kinds are:
  - immediate numeric data
  - data contained in numeric variables
  - immediate alphanumeric data (string)
  - alphanumeric data contained in string type variables

  The data transmitted constitute the MESSAGE, maximum length of which is 180 characters (terminators included).

  The numeric data are converted into ASCII characters, whereas the string characters are transmitted as they are. If numerous data items are transmitted in a string, they may be separated by data terminators, whereas a message may be closed by a message terminator.

- Receive data of any kind; these kinds are:
  - immediate numeric data
  - data contained in numeric variables
  - immediate alphanumeric data (string)
  - alphanumeric data contained in string type variables

  The variables expected in reception must be consistent with the data transmitted, therefore the transmission and reception commands must correspond in terms of type, format, terminators, and length of the data communicated.

  Length of a message received must not exceed 180 characters, terminators included.

- Close the serial lines after communication is completed.

SOP - Activate and configure the serial port

This triliteral lets you reserve the serial port specified for use and configure the physical communication characteristics.

Syntax

(SOP, *port number* **,** *transmission speed (BAUD RATE)*

, word length
, number of stop bits
, parity
, protocol
, transmission timeout
, reception timeout
/ number of TX data terminator [,term [, term [, term]]]
, number of RX data terminators [, term [, term [, term]]]
, number of TX mess terminators[, term [, term [, term]]]
, number of RX mess terminators [, term [, term [, term]]] **)**

where:

port number — Is the value identifying the serial port to use; may be 1 or 2 and may be a local or system variable.
The selection of the serial port for the communication of the part program with external devices depends on the type of hardware available and whether some devices such as serial printers, Teach pendant, etc., are used. It has to be configured in AMP (HARDWARE section).

transmission speed — The values accepted are (BAUD):

50	75	110	134	150	300
600	1200	1800	2000	2400	3600
4800	7200	9600	19200	38400	56000

word length — The following values are accepted (number of bits):
5 6 7 8

number of stop bits — Values possible are:
1 (= 1 stop bit)
2 (= 2 stop bits).

parity — Type of parity check. May be 0, 1 or 2, with the following meanings:
0 = no parity
1 = odd parity
2 = even parity

protocol	Protocol used for transmission/reception. May be 0, 1,2 ,3 with the following meanings: 0 = no protocol 1 = logical protocol (XON XOFF) 2 = physical protocol (signals DTR, DSR, RTS, CTS) 3 = logical protocol (XON XOFF) with signals DTR and DSR
transmission timeout	Timeout during data transmission, in seconds. The minimum value is 0.01. If a 0 is specified, the timeout in transmission is not activated.
reception timeout	Timeout during data reception, in seconds. The minimum value is 0.01. If a 0 is specified, the reception timeout is not activated.
number of TX data terminators	Is the number of characters added automatically to each data item transmitted (except the last). Allowed values range from 0 to 3 (0 = no data terminator). These characters act as the separator between one data item transmitted and the next inside the same message. If only one data item is transmitted, this number and the TX data terminators are not used. The TX data terminator characters must be specified immediately after this number in the way described later (*term*); they will be added in the order in which they are specified.
number of RX data terminators	Allowed values range from 0 to 3 (0 = no data terminator). This is the number of characters interpreted as separators between the data items received. These characters are not put into the variables specified with the GET triliteral but simply indicate end of a data item to the system. Presence of one of the three TX data terminators is enough for the data item to be considered complete. The terminators specified may be included several times between one data item and the next. If this parameter is set to 0, then only one data item at a time is to be received (GET with one variable only). The RX data terminator characters must be specified immediately after this number in the way described later (*term*).
number of TX mess terminators	Is the number of characters added automatically in the order specified at the end of the last data item received (last data item in the PUT triliteral), constituting the serial message terminator. Allowed values range from 0 to 3 (0 = no mess. terminator). The TX mess terminator characters must be specified immediately after this number in the way described later (*term*).

number of RX mess terminators

Is the number of characters considered as message terminators in reception: as soon as one of these characters arrives, reception is complete and the data items already arrived are put into the variables specified in triliteral GET. Allowed values range from 0 to 3 (0 = no mess terminator)
If a 0 is specified, this means you intend to receive one data item at a time, with only one character.
The RX mess terminators must be specified immediately after this number in the way described later (term).

term

The word used as terminator (data and/or messages).
May be:

- A whole number (0÷255).
- An alphanumeric character in inverted commas.
- A set of alphanumeric characters in inverted commas.

If the number of terminators declared is greater than 1 and alphanumeric characters are used as terminators, these must be put all together between the same inverted commas or put singly between inverted commas and separated by a comma.

Examples:

A) Number of terminators = 1

Terminators: "A"
10
250
"b"

B) Number of terminators = 2

Terminators: "AB" or "A", "B"
"A", 10
20,30
127, "b"

C) Number of terminators = 3

Terminators: "ABC" or "AB", "C" or "A", "BC" or "A", "B", "C"
"AB", 10 or "A", "B", 10

"A", 20, 230
25, "B", 75
25, "b", "C" or 25, "bC"

Characteristics

All the parameters of the SOP triliteral may be expressed directly with a number or with a local or system variable.

When a SOP command is executed, to reconfigure one or more parameters of the serial line, you must first execute an (SCL) command to close the serial line and then a new SOP command.

Both the serial lines available on 10 Series may be enabled simultaneously.

When the SOP triliteral is executed correctly, the serial line enabled is reserved for exclusive use of the current process and cannot be used by another process up until it is closed (SCL).

Examples:

(SOP,1,9600,8,1,1,0,10,0.5/0,1,10,2," ",10,1,13)

serial line 1,
9600 baud, 8 bit, 1 stop bit , odd parity, no protocol ,
10 seconds transmission timeout, 0.5 seconds reception timeout,
no TX data terminator,
one RX data terminator (10)
two TX message terminators (" " and 10),
one RX message terminator (13).

(SOP, 2, 2400, 7, 1, 2, 1, 4, 0/2," "," ",3," ",9,13,2,13,10,1,13)

serial line 2,
2400 baud, 7 bit, 1 stop bit, even parity, logical protocol (XON XOFF),
4 seconds transmission timeout, reception timeout not enabled
two TX data terminators (" ", and " ")
three RX data terminators (" ", 9, 13),
two TX message terminators (13 and 10 (CRLF)),
one RX message terminator (13).

(SOP,1,2400,8,2,0,2,0,0/2," "," ",0,2,13,10,0)

serial line 1,
2400 baud, 8 bit, 2 stop bits, no parity, physical protocol,
transmission timeout not enabled, reception timeout not enabled,
two TX data terminators (" "," " (two blank spaces)),)
no RX data terminator (one data item to be received for each GET)
two TX message terminators (13 and 10 (CRLF)),
no RX message terminator (a single data item consisting of one character only).

GET - Data reception from serial line

This triliteral sets the process in which it is programmed ready for data from the serial line specified, which must be enabled and configured. The data received are stored in the variables in accordance with the order of arrival and programming (the first data items to arrive are stored in the first variable programmed, and so on).

Syntax

(GET, *port number*, *variable* *[, variable] [, variable] [, variable]*
[, variable] [, variable] [, variable]
[, variable] [, variable] [, variable] **)**

where:

port number Is the value identifying the serial port from which the data are to be received, enabled earlier with triliteral SOP; allowed values are 1 or 2.
This value may be a number or a variable.

variable List of the variables (1÷10) that the data will be put in as they arrive from the serial line.

Characteristics

In reception, the data items contained in the reception MESSAGE will be put into the variables specified in triliteral GET. This message may comprise one or more data items, separated by RX data terminator characters: if no such character is specified, the reception MESSAGE comprises a single data item of one character only.

The triliteral GET is suspensive, i.e. execution of the part program does not start again until the GET is considered completed:

- on arrival of a character if no RX message terminators have been programmed
- on arrival of one of the RX message terminators programmed with triliteral SOP
- on arrival of a RESET or a CYCLE STOP in the process the command is being executed in
- on expiry of the reception timeout (if programmed): the characters that arrived up to now are lost and the serial line is automatically reinitialized.

If a numeric data item is expected in reception (for example, : (GET, 1, E1): waiting for numeric data to put in variable E1), the data coming in will contain a numeric value expressed in ASCII characters, which will in turn be converted automatically into binary format. This conversion is not made if:

- the variable specified is character type (string)
- one character is received at a time (no RX message terminators specified).

If several variables were programmed in triliteral GET, at least one RX data terminator character and at least one RX message terminator must have been specified in triliteral SOP.

Some errors that may be generated by the GET command may be handled by the part program: for a full list of these errors, see Appendix B.

When the triliteral GET terminates execution without errors, the variables programmed contain all the data coming in from the serial line.

This triliteral cannot be programmed in MDI mode with CYCLE STOP in ON status.

Examples:
(GET, 1, SC10)

waiting for a single data item of 10 characters (excluding the RX message terminator).

(GET, 1, E1)

waiting for numeric data in ASCII code or a single character, saved in E1 without any conversion (the case of no RX message terminator)

(GET, 2, L30, L31, L32, !MYVAR, $TCODE (1), SCO.10)

waiting for several data items, numeric and character strings, from line 2.

PUT - Data transmission on serial line

This triliteral is used to transmit data on the serial line specified, which was enabled and configured earlier.

Syntax

(PUT, *port number***,** *data [, data] [, data] [, data]*
[, data] [, data] [, data]
[, data] [, data] [, data] **)**

where:

port number Is the value identifying the port from which to transmit the data; this port has been enabled earlier with the SOP triliteral and the accepted values are 1 or 2. This value may be a number or a variable.

data Is the data to transmit; may be a constant, a string between inverted commas or a local or system variable.

Characteristics

The PUT triliteral is suspensive: execution of the part program continues only after all the data programmed have been transmitted.

The numeric data programmed in the PUT triliteral, as immediate or as the contents of numeric variables, are converted into ASCII characters before being transmitted. On the other hand, strings and the contents of character type variables (string) do not undergo any conversion and are transmitted as they are.

The set of characters taken from the variables or the constants specified in the PUT triliteral constitute the MESSAGE to be transmitted, which may be a maximum length of 180 characters, including the TX data terminators and the TX message terminators, if programmed in triliteral SOP.

The transmission (execution of triliteral PUT) terminates:

- when the transmission timeout expires (if one is programmed): in this case, the serial line is automatically reinitialized
- when serial transmission terminates correctly
- on arrival of a RESET or a CYCLE STOP in the process the command is being executed in.

Some errors that may be generated by the PUT command may be handled by the part program: for a full list of these errors, see Appendix B.

When the triliteral PUT terminates execution without errors, the data items programmed have all been transmitted on the serial line.

This triliteral cannot be programmed in MDI mode with CYCLE STOP in ON status.

Examples:
(PUT, 1, SC10)

Transmission of a single item of 10 characters, plus any TX message terminators.

(PUT, 1, E1)

Transmission of the numeric value of variable E1 in ASCII code (plus any terminators).

(PUT, 1, "MESSAGE", E1, SCO.4.$TCODE (1))

Transmission of constants, numeric variables (in ASCII code), string type variables (separated by TX data terminators and with any terminators specified at the end).

SCL - Close the serial port

This triliteral disables the serial line specified, making it available for other possible users. If the serial line number is not specified, all active serial lines in the process where the triliteral is programmed are disabled.

Syntax

(SCL, *[port number]***)**

where:

port number Is the value identifying the serial port to disable; allowed values are 1 or 2.
May be a number or a local or system variable.
If this parameter is not specified, both the serial lines of the system are disabled (if both were enabled previously).

Characteristics

If the serial line to be disabled is not enabled, this command does nothing and no error code is generated.

This triliteral cannot be programmed in MDI mode with CYCLE STOP in the ON status.

Examples:

(SCL, 1)
disable serial line 1

(SCL)
disable all active serial lines in the process.

E1 = 2
(SCL, E1)
disable serial line 2

EPS - Execute part program from serial line

This triliteral code is used to execute a series of part program blocks coming from the serial line for as long as the block to be executed is different from the one specified as **part program end string**.

Syntax

(**EPS,** *serial_line_number , part program_end_string*)

where:

serial line number	is the serial line from which to take the blocks for execution. May be a number or a variable (local or system) with possible values 1 or 2. Must be a serial line suitably opened and configured with the SOP triliteral.
part program end string	is a string between quotes or a character variable. Specifies the string that will be considered as terminating execution from the serial line and command EPS. N.B.: This string is not executed as a part program block.

Characteristics

The EPS command uses the type 2 transmission protocol (physical with signals DTR, DSR, RTS, CTS, see the triliteral SOP). The serial line opened and configured in advance with triliteral SOP; for correct EPS operation, the serial line must be configured with a single terminator given in reception and this must be the character LF (LINE FEED : 10): this is because part program blocks coming from a serial line MUST terminate with characters CR-LF (like part program blocks in a file); in addition, part program blocks from a serial line can must not be more than 127 characters long (CR-LF excluded).
If the serial line is not configured correctly, the system will generate error **NC 41 Wrong serial line configuration for EPS** when it comes to execute an EPS command.
While an EPS is being executed, a reception timeout configured with SOP is not handled.

If a p.p. block coming from a serial line generates an error, it will not be executed on the next cycle start; it will however be copied into the MDI buffer where, on entering MDI mode, it may be displayed, modified and executed before proceeding with execution of the other blocks from the serial line.

Some part program blocks cannot be executed if coming from a serial line. These are the triliterals:

GTO, EPP, RPT, ERP, EPS,
SCL, PUT, GET (the latter three if referring to the serial line used for EPS)

NOTE:

Triliterals CLS and EPB and calls to paramacros are allowed: it is the main program to which these subprograms refer that contains the EPS triliteral.

Any attempt to execute these commands from a serial line generates this error:

NC 040 P.P. block not allowed from serial line.

Triliteral code EPS cannot be executed in HOLD status (in MDI mode in HOLD or CYCLE STOP status). Execution of blocks from a serial line is "seen" by the system as repeated loop execution of the EPS command until arrival of the string configured as the part program end string. Blocks from a serial line can also be executed in BLK/BLK mode; while an EPS command is being executed in BLK/BLK mode, the following operations are inhibited:

SKIP of block with ARROW UP - ARROW DOWN keys
STRING SEARCH
MODIFY

Examples of use of triliteral EPS:

```
(SOP,1,9600,8,1,0,2,0,0/0,0,0,1,10)   ------> Correct configuration of RX terminators
(EPS,1,";;")                          ------> Execution from serial line until arrival of characters
;;
(SCL,1)
```

```
SC0,4 = "END"
(SOP,1,9600,8,1,0,2,0,0/0,0,0,1,10)   --------> Correct configuration of RX terminators
(EPS,1,SC0,4)                         --------> Execution from serial line until arrival of END string
(SCL,1)
```

Use of a personal computer with MS-DOS

If using a personal computer with the MS-DOS operating system, the following command can be used

COPY filename COMx:

to copy a part program file on the serial line, suitably configured with the MODE commands, for execution by the 10 Series control.

Example:
Execute in sequence these operations:

On P.C.

```
EDIT A FILE: c:\example\program          (DIS,"I AM THE FILE PROGRAM")
                                         END

MODE COM1 9600, n, 1, 8, P          --------> 9600 BAUD NO PARITY 1 STOP BITS
                                              8 DATA BYTES INFINITE RETRIES

COPY c:\example\program COM1:
```

On 10 Series

Execute this part program in AUTO or BLK/BLK mode

```
(SOP,1,9600,8,1,0,2,0,0/0,0,0,1,10)
(EPS,1,"END")
(SCL,1)
```

NOTE:

The personal computer does not produce the video prompt until after execution of the SOP command enabling, through the hardware, reception in the 10 Series serial buffer, though this is not the case if the part program is more than 3000 bytes long. In these cases, blocks coming from the serial line must be executed to free space in the serial driver's data area: the personal computer does not display the command prompt until space is found in this data area for the last block of the file.

If the serial line is closed with triliteral SCL or with a RESET, any part program blocks present in the serial buffer data area (arriving after the SOP command), are irreparably lost.

END OF CHAPTER

OPERATING COMMANDS

This chapter contains a certain number of three-letter functions used to generate system operating commands, such as CYCLE START, HOLD, RESET, axis selection, etc., for the process specified in the function or the default process set earlier. These functions are:

- **CYCLE ON**, equivalent to pressing the **CYCLE START** button
- **CYCLE OFF**, equivalent to releasing the **CYCLE START** button
- **HOLD ON**, equivalent to pressing the **CYCLE STOP** (**HOLD**) button
- **HOLD OFF**, equivalent to pressing the **CYCLE STOP** (**HOLD**) button in **CYCLE STOP** or **HOLD** mode
- **RESET**, equivalent to pressing the **RESET** button
- **Set Operating Mode**, to set the control's operating mode
- **Select Axes for Manual Movements**, to specify the axis on which manual movements are to be executed
- **Direction of Manual Movements** to establish the direction of movement on the axis selected
- **Jog Step Value** to set the distance to be travelled with each jog step
- **Enable/Disable FEEDHOLD** to force speed of the axes to zero.

The above functions are implemented by way of the triliterals listed below and are described in detail on the pages that follow:

- CON
- COF
- HON
- HOF
- RES
- SMD
- SAX
- DIR
- JOG
- FHO

IMPORTANT

Implementation of these commands depends on status of the process they are sent to.

CON - CYCLE ON command

This instruction sends the specified process the CYCLE ON command, which is equivalent to pressing the **CYCLE START** button.

Syntax

(CON *[, P numproc]***)**

where:

numproc Is a number between 1 and 20 identifying the process on which the CYCLE ON command will be executed. This parameter may be a number directly identifying the process or a local or system variable containing the number of the process desired.

Characteristics

The result of issuing this command depends on the current mode of operation.

If the *P numproc* parameter was not specified, the CYCLE ON command will be given to the default process, i.e. the one set earlier with the PRO instruction.

Example:
(CON, P3)

Process 3 will be issued the CYCLE ON command.

COF - CYCLE OFF command

This instruction sends the process specified the CYCLE OFF command, which is equivalent to releasing the **CYCLE START** button.

Syntax

(COF *[, P numproc]***)**

where:

numproc Is a number between 1 and 20 identifying the process on which the CYCLE OFF command will be executed. This parameter may be a number directly identifying the process or a local or system variable containing the number of the process desired.

Characteristics

The result of issuing this command depends on the current mode of operation.

If the *P numproc* parameter was not specified, the CYCLE OFF command will be given to the default process, i.e. the one set earlier with the PRO instruction.

Example:
(COF)

The default process will be issued the CYCLE OFF command.

HON - HOLD ON command

This instruction sends the specified process the HOLD ON command, which is equivalent to pressing the **CYCLE STOP** (**HOLD**) button.

Syntax

(HON *[, P numproc]***)**

where:

numproc Is a number between 1 and 20 identifying the process on which the HOLD ON command will be executed. This parameter may be a number directly identifying the process or a local or system variable containing the number of the process desired.

Characteristics

If the *P numproc* parameter was not specified, the HOLD ON command will be given to the default process, i.e. the one set earlier with the PRO instruction.

If no default process was set earlier and the P numproc parameter was not included in this command, the command will be executed on the current process (i.e. the one executing the current instruction).

Example:
(HON, P2)

Process 2 will be issued the HOLD ON command.

HOF - HOLD OFF command

This instruction sends the specified process the HOLD OFF command, which is equivalent to pressing the **CYCLE STOP** (**HOLD**) button in **CYCLE STOP** or **HOLD** mode.

Syntax

(HOF *[, P numproc]***)**

where:

numproc Is a number between 1 and 20 identifying the process on which the HOLD OFF command will be executed. This parameter may be a number directly identifying the process or a local or system variable containing the number of the process desired.

Characteristics

If the *P numproc* parameter was not specified, the HOLD OFF command will be given to the default process, i.e. the one set earlier with the PRO instruction.

Example:
(HOF)

The default process will be issued the HOLD OFF command.

RES - RESET command

This instruction sends the specified process a reset command, which is equivalent to pressing the **RESET** on the operator panel.

Syntax

(RES *[, P numproc]***)**

where:

numproc Is a number between 1 and 20 identifying the process. This parameter may be a number directly identifying the process or a local or system variable containing the number of the process desired.

Characteristics

If the *P numproc* parameter was not specified, the RESET command will be given to the default process, i.e. the one set earlier with the PRO instruction.

If no default process was set earlier and the *P numproc* parameter was not included in this command, the command will be executed on the current process (i.e. the one executing the current instruction).

Example:
(RES,P4)

Process 4 will be issued the RESET command.

SMD - Set operating mode

This instruction defines mode of operation for the process specified.

Syntax

(SMD *[P numproc]* **,***modo***)**

where:

numproc Is a number between 1 and 20 identifying the process. This parameter may be a number directly identifying the process or a local or system variable containing the number of the process desired.

modo The mode of operation desired. Possible values for this parameter are:

1 MDI
2 Auto
3 Block/block
4 Continuous manual
5 Incremental manual
6 Return on profile
7 Homing cycle
8 Handwheel

Characteristics

If the *P numproc* parameter was not specified, the SMD command will be given to the default process, i.e. the one set earlier with the PRO instruction.

If no default process was set earlier and the *P numproc* parameter was not included in this command, the command will be executed on the current process (i.e. the one executing the current instruction).

Example:
(SMD,P4,3)

The block/block operating mode is set on process 4.

SAX - Select axis for manual movement

This instruction selects the axis on which a manual movement is to be executed.

Syntax

(SAX *[, P numproc] , nome asse***)**

where:

numproc Is a number between 1 and 20 identifying the process. This parameter may be a number directly identifying the process or a local or system variable containing the number of the process desired.

nome asse Is the ASCII name of the axis to execute the movement on.

Characteristics

If the *P numproc* parameter was not specified, the SAX command will be given to the default process, i.e. the one set earlier with the PRO instruction.

If no default process was set earlier and the *P numproc* parameter was not included in this command, the command will be executed on the current process (i.e. the one executing the current instruction).

Example:
(SAX,Z)

Axis Z of the current process is selected for manual movement.

IMPORTANT

If the axis name is P, you have to insert the process number (even if it is the default process).

DIR - Direction of manual movements

This instruction defines the direction of the axis selected during the manual movement.

Syntax

(DIR *[, P numproc] , segno***)**

where:

numproc	Is a number between 1 and 20 identifying the process. This parameter may be a number directly identifying the process or a local or system variable containing the number of the process desired.
segno	Sign of the direction of manual movement. Possible values are [**+**] or [**-**].

Characteristics

If the *P numproc* parameter was not specified, the DIR command will be given to the default process, i.e. the one set earlier with the PRO instruction.

If no default process was set earlier and the *P numproc* parameter was not included in this command, the command will be executed on the current process (i.e. the one executing the current instruction).

Example:
(DIR,P3,+)

A positive manual movement is selected for the axis selected earlier with the SAX command in process 3.

JOG - Jog step value

This instruction defines the value of the single steps during jog manual movement in the process specified.

Syntax

(JOG *[, P numproc] , jog value***)**

where:

numproc Is a number between 1 and 20 identifying the process. This parameter may be a number directly identifying the process or a local or system variable containing the number of the process desired.

jog value Is the numeric value for the step in jog manual movement.
May be a number or a local or system variable.
Will be assumed in mm or inches depending of the unit configured for the process.

Characteristics

If the *P numproc* parameter was not specified, the JOG command will be given to the default process, i.e. the one set earlier with the PRO instruction.

If no default process was set earlier and the *P numproc* parameter was not included in this command, the command will be executed on the current process (i.e. the one executing the current instruction).

Example:
(JOG,P2,0.1)

A value of 0.1 mm (or 0.1 inches) will be taken as the step for jog manual movement in process 2.

FHO - Enable/Disable FEEDHOLD

This instruction makes a stop during machining of the process specified. In practice, it forces working speed to zero.

Syntax

(FHO *[, P numproc] , enable***)**

where:

numproc Is a number between 1 and 20 identifying the process. This parameter may be a number directly identifying the process or a local or system variable containing the number of the process desired.

enable Is a number either 0 or 2 representing the FEEDHOLD enable or disable command.

With a value of 1 the FEEDHOLD command is enabled, i.e., the movement is stopped at once; with 2 axes, the programmed movement currently being executed is stopped; with 0 the movement is re-enabled.

Characteristics

If the *P numproc* parameter was not specified, the FEEDHOLD command will be given to the default process, i.e. the one set earlier with the PRO instruction.

Example:
(FHO,P4,1)

Machining movement is stopped in process 4.

END OF CHAPTER

ASSET ERROR MESSAGES

The errors that occur while the system is executing ASSET commands are listed in the table below. Some of these errors may be managed from part program by means of the STE variable (refer to Appendix B for details).

The list below also includes errors listed in the Programming Manual but which may occure during ASSET operation.

Description of messages and remedial action

Code	Error description and remedial action
NC001	**Syntax Error** Syntax error in the programmed or manually input block
NC008	**Format error** This error code appears when: • the variable index is illegal • the variable format is illegal • the assignment format is illegal. For example, the strings are of different lengths • there is a PLUS variable writing/reading error • the parameter in the block is out of range
NC009	**Undefined symbol** This error appears when the specified variable does not exist.
NC037	**Read only variable** This error occurs when the program attempts to write a read only variable.
NC040	**P.P. Block not allowed from serial line** The part program block coming from the serial line cannot be executed (e.g. GTO, EPP, RPT, EPS, etc.).
NC041	**WRONG SERIAL LINE configuration for EPS** The serial line is not configured correctly for EPS.

Code	Error description and remedial action
NC052	**String too long** This error occurs when the following string has more than 127 characters: • string displayed with an INP instruction • assignment of a string variable. For example, SC.
NC234	**ASSET option not installed** This error occurs when the variable or ASSET command has not been installed. Check with the SECURITY utility whether the ASSET option is installed and enabled.
NC235	**File channel busy: use another channel identifier (from 1 to 5)** This error occurs when the channel specified in the file open instruction is already busy. Close the working session that engages the channel or select a new channel.
NC236	**File open error** This error occurs when opening the channel associated to the file. Check the user disk conditions.
NC237	**File already open** This error occurs in the following cases: • the operator tries to write data in a file that has been opened for writing by another channel or process • the operator tries to delete a file currently open for reading or writing • the operator tries to save in the directory of programs an ASCII file currently open for reading or writing
NC238	**File not found** This error occurs: • when the file open for reading does not exist • when the file to be deleted does not exist • when the ASCII file to be saved in the directory of programs does not exist • when the extension of the ASCII file to be saved is not .ASC
NC239	**I/O Error** This is an file access error. Check the user disk conditions.
NC240	**Wrong access type or file channel closed** This error occurs when: • when the operator tries to write in a file open for reading • when the operator tries to read from a file open for writing • when the file channel accessed for reading or writing is closed
NC241	**End of file** This error occurs when when the system reaches the end of the file during a reading operation

Code	Error description and remedial action
NC243	**Part program name duplicated** This error occurs when when the ASCII file to be saved already exists in the directory of programs.
NC244	**Part program directory full** This error occurs when in the directory of programs there is no space for further ASCII files.
NC245	**Character not allowed in program name** This error occurs when when some of the characters in the part program file name are not allowed.
NC246	**No more space on device: disk full**
NC247	**Data file corrupt** This error occurs when when the header of the binary data file includes information that is not compatible with the file contents. Such a file cannot be read.
NC248	**Data file access error** This error occurs when the binary data file has been accessed for reading or writing through a record that is different from the first record in the file.
NC249	**Record too long** This error occurs when the maximum record length (127 characters for ASCII files, 300 bytes for binary files) is overflown.
NC250	**Driver not allowed** An attempt has been made to access a reserved driver.
NC251	**Driver busy or not configured** The device being defined with triliteral GVD is not available or is being used by another user (for example, another process or the DOS SHELL UTILITY.
NC255	**Wrong row for input field** This error occurs when when the instruction associated to a data entry window invokes a field outside the allowed line range (from 0 through 18).
NC256	**Wrong column for input field** This error occurs when when the instruction associated to a data entry window invokes a field outside the allowed column range (from 0 through 79).
NC257	**Input mask not defined** This error occurs when when the DIF instruction specified in the INP command has not been defined before.
NC258	**INP: data format error** This error occurs when when saving the data input with an INP instruction.

Code	Error description and remedial action
NC260	**Wrong screen identifier** This error occurs when when the operator writes in a screen that cannot be modified with ASSET.
NC261	**Screen not configured** This error occurs when when the screen specified in the OUT command for writing has not been configured.
NC262	**OUT on user screen failed** This error occurs when when the OUT operation on the screen fails.
NC270	**PLUS table already locked** This error occurs when when the table to be reserved with LCK has been already reserved for writing by another user, such as PLUS, Table Editor, etc.
NC271	**Serial line already activated** An attempt has been made to activate a serial port with the SOP triliteral, but the port is already active.
NC272	**Serial line closed** Execution of the GET or PUT triliteral on a serial line that has not yet been activated with the SOP triliteral.
NC273	**Serial transmission timeout** The transmission timeout has expired during execution of the PUT triliteral. When the timeout is generated, there is no guarantee that all the data have been sent on the serial line specified. The serial line is reinitialized. This errror can be handled from the part program (see Appendix B).
NC274	**Serial reception timeout** The reception timeout has expired during execution of the GET triliteral. This means that the data expected have not arrived in the maximum time established. The serial line is reinitialized. This errror can be handled from the part program (see Appendix B).
NC275	**Serial line general failure** Problems with physical connection of the serial line.
NC276	**Failed loading data from the serial line** The data expected from the serial line with triliteral GET have arrived properly, but they are not of the number expected by the part program (too many or some missing). This error can be handled from the part program (see Appendix B).

Code	Error description and remedial action
NC277	**Serial communication aborted by HOLD** Serial reception or transmission has been aborted by a HOLD received by the process: this error is displayed during resumption of execution after the HOLD. The serial line is reinitialized.
NC278	**Serial port not available** The serial line specified for activation with triliteral SOP is not available for use by ASSET.
NC279	**Serial port already used** The serial line specified for activation with triliteral SOP is already being used by another process.
NC280	**Data received too long** A message of more than 180 characters has arrived during reception.
NC281	**Serial line overrun error** The serial line driver is in OVERRUN status. Reduce the communication baud rate. This error can be managed by the part program (see Appendix B).
NC282	**Serial line parity error** Parity error. This error can be managed by the part program (see Appendix B).
NC283	**Serial line framing error** This error can be managed by the part program (see Appendix B).

END OF APPENDIX

ERROR MANAGEMENT FROM PART PROGRAM

Standard 10 Series programming provides the programmer with a method for error management from part program. This method permits to correct or recover the error without interrupting the cycle.

Error management from program must be configured through the following system variables:

ERR Enables/disables automatic error management from part program.

STE Is a read only system variable that stores the error code generated by a previous command when automatic error management from part program is configured (ERR=1).

In particular, these variables manage errors detected during probing cycles (G71, G72 or G73) or generated during execution of the GTA command. For more information about ERR and STE refer to the 10 Series Programming Manual.

ASSET permits automatic management from part program of three types of error:

- errors generated during execution of an INP instruction, i.e. manual data input errors
- errors generated during execution of a file management command
- errors generated during use of the serial line.

The sections that follow describe the ERR variable and how its value affects the system conditions. The meaning of the error codes is shown in Appendix A.

NOTE:
A system reset sets both STE and ERR to 0

ERR - Enabling/disabling automatic error management from part program

ERR enables/disables automatic management from part program of errors generated during execution of specific program instructions.

Syntax

ERR = value

where:

value Is a boolean value (0 or 1) that may be written as a numerical format or read from a local or system variable.

If ERR = 0 automatic error management is disabled. When the system detects an error it interrupts the execution of the program and displays the relevent error code.

If ERR = 1 automatic error management is enabled. When the system detects an error it writes an error code in the STE system variable.

Characteristics

When ERR = 1 it is recommended to read the STE variable immediately after the system executes a command in which an error may have occurred. With ERR = 1 error messages are not displayed and the part program is not interrupted.

Manual data input errors

During execution of the INP command the following errors may occur:

Condition	STE value with ERR = 1	Displayed error code with ERR = 0
Execution without errors	0	----
INP closed with **Esc**	20 (*)	----
Input values outside the allowed range (data saving failed)	21	NC258
INP closed with **Page Up**	22 (*)	----
INP closed with **Page Down**	23 (*)	----
INP closed with **Home**	24 (*)	----
INP closed with **End**	25 (*)	----

(*) This value will be written in STE even when ERR = 0

Example:

```
"LAB"
(INP,N0,T'TEST INPUT','FEED',10,E1,'SPEED',10,E2,'COORD',10,E3)
(GTO,LAB,STE=20)
..... ; valid data
```

The above instructions write the feedrate value in E1, the speed value in E2 and the programmed coordinate in E3.

If these data are not confirmed and the data entry window is closed with [**Esc**], the system will write 20 in the STE variable. The GTO instruction uses this value to position the program to the "LAB" label, which re-displays the data entry window for the operator to make the necessary corrections.

File management errors

The table below lists the file management errors that can be managed from part program:

Condition	STE value with ERR = 1	Displayed error code with ERR = 0
Execution without errors	0	----
File channel busy	30	NC235
File currently used	31	NC237
File not found	32	NC238
End of file	33	NC241
Program already exists in directory of programs	34	NC243
Directory of programs full	35	NC244
Program name not allowed	36	NC245
Working disk unit full	37	NC246
Access to binary file through unallowed record type	38	NC248
Record too long (>127 for ASCII; >300 for binary)	39	NC249
Device not available	40	NC251

Example:

(INP,"DEFINE FILE NAME:",30,SC8.8)	*Request to enter file name*
"LABINS"	*File name label*
ERR=1	*Enables error management from program*
(INS,"PPGR3",SC8.8,1)	*Save file in User directory*
(GTO,LAB,STE=34)	*If file already exists branch to "LAB"*
"CONT"	*"Continue" label*
ERR=0	*Disables error management from program*
.....	
"LAB"	*New name label routine*
(INP,"FILE EXISTS. RENAME:",30,SC8.8)	*Request to rename the file*
(GTO,"LABINS")	*Returns to file label routine*
.....	

Locked table condition

While the system executes an LCK instruction, the following errors may be managed from part program:

Condition	STE value with ERR = 1	Displayed error code with ERR = 0
Table reserved without errors	0	
Table already reserved by other users	45	NC270

Serial line management errors

Errors that can be managed from a part program and that may occur during transmission on a serial line are as follows:

Condition	STE value with ERR = 1	Displayed error with ERR = 0
Serial transmission timeout	50	NC273
Serial reception timeout	51	NC274
Data downloading from serial line failed	52	NC276
Overrun error	53	NC281
Parity error	54	NC282
Framing error	55	NC283

END OF APPENDIX

ASSET TRILITERALS TABLE

List of the Asset Triliterals

Triliteral	Function	
(OPN)	OPeN File	opens a file
(CLO)	CLOse file	closes a file
(DEL)(CAN)	DELete file or part program	deletes a file/part program
(WRT)	WRiTe record	writes a record on a file
(REA)(RED)	REAd record	reads a record on a file
(INS)	INSert part program in ppdir file	inserts a part program
(SCR)	SCReen selection	selects a screen
(OUT)	OUTput data on screen	writes data on user screen (logic)
(DIF)	Define Input Fields	defines input fields
(INP)	INPut data from keyboard	inputs data from keyboard
(PRO)	defines default PROcess number	defines the process default
(RTP)	Read actual or programmed Tool Process	reads the programmed or actual tool
(ROP)	Read actual or programmed Offset Process	reads programmed or actual offset
(GPS)	Get Process Status, substatus and mode	reads process status or mode
(RAP)	Read Axis Position	reads axis position
(PLS)	read PLus Sw variables	reads the PLUS SW variables
(LCK)	LoCK/unlock plus tables	locks/unlocks the PLUS tables
(CON)	Cycle ON	cycle on
(COF)	Cycle OFf	cycle off
(HON)	Hold ON	hold on
(HOF)	Hold OFf	hold off
(RES)	RESet process	resets the process
(SMD)	Set MoDe	selects operating mode
(SAX)	Select AXis for manual movement	selects the axis
(DIR)	DIRection for manual movement	direction of manual movements
(JOG)	JOG incremental value	JOG step value
(FHO)	Feed HOld	enables/disables feed hold
(SOP)	Serial line OPen	opens the serial line
(PUT)	PUT data on serial line	trasmits data on serial line
(GET)	GET data from serial line	receives data from serial line
(SCL)	Serial CLose	closes the serial line
(EPS)	Exec Program from Serial line	executes p.p. from serial line

END OF APPENDIX