NC-110, NC-210, NC-201M

SOFTWARE CHARACTERIZATION
MC-TC

Saint-Petersburg
2008

PUBLICATION ISSUED BY:

Balt-System Ltd.
198206, Sankt-Petersburg
Petergofskoje Shosse, 73

Telefax: +7(812)744 70 59
Telephone: +7(812)744 70 59

Important User Information

This document has been prepared in order to be used by Balt-System Ltd. customers. It describes the latest release of the product.

Balt-System Ltd. 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 Balt-System Ltd. be responsible or liable for indirect or consequential damages that may result from installation or use of the equipment or software described in this text.

PREFACE

This manual describes the initialization and characterization procedures for the Balt-System Ltd. NC-110 MC and NC-110 TC systems. The manual is a guide to how system and process software characterization files are created.

This manual is aimed in particular at machine tool manufacturers who use products in the NC-110 line with their machine tools and, in general, at users and installation engineers of the NC-110 systems.

CONTENTS

Chapters 1 and 2 give a general description of the problems involved in configuring NC-110 MC and NC-110 TC systems.

Chapter 3 describes the initialization procedure for activating the system on.

Chapter 4 describes the characterization files diagnostic.

Chapter 5 describes how to characterize the basic system using file FCRSYS.

Chapters 6, 7, and 8 describe how to characterize the process software using files AXCFIL, PGCFIL and IOCFIL.

Chapters 9 and 10 give the error codes and messages concerning the JOB partition and process software characterization.

REFERENCE

NC-110 MC, Use and programming manual.

NC-110 TC, Use and programming manual.

CONTENTS

1. INTRODUCTION 6

2. WHAT IS THE CHARACTERIZATION 7

2.1. WORKING ENVIRONMENT 7
2.1.1. MINICOMPUTER AND NUMERIC CONTROL 7
2.1.2. BASIC SYSTEM AND PROCESS SOFTWARE 7
2.1.3. CHARACTERIZATION LEVELS 8
2.2. CHARACTERIZATION FILES 9
2.2.1. GENERAL 9
2.2.2. FILE STRUCTURE 11
2.3. CHARACTERIZATION ORDER 13
2.3.1. INITIALIZATION 13
2.3.2. BASIC SYSTEM CHARACTERIZATION 13
2.3.3. PROCESS SOFTWARE CHARACTERIZATION 14

3. SYSTEM INITIALIZATION 16

3.1. SYSTEM POWER ON 16
3.2. SOFTWARE CALIBRATIONS 19

4. CHARACTERIZATION DIAGNOSTICS 22

4.1. FCRSYS FILE DIAGNOSTICS 22
4.1.1. DIAGNOSTICS SCREEN 22
4.1.2. ERRORS IN THE HARDWARE MODULES 23
4.1.3. ERRORS IN THE FCRSYS FILE 27
4.2. AXCFIL, PGCFIL, IOCFIL FILE DIAGNOSTICS 27
4.2.1. DIAGNOSTICS SCREEN 27
4.2.2. ERRORS IN THE AXCFIL, PGCFIL, IOCFIL FILES 29

5. BASIC SYSTEM CHARACTERIZATION 30

5.1. THE BASIC SYSTEM 30
5.2. FILE FCRSYS 30
5.3. LOADING FCRSYS FROM AN EXTERNAL UNIT 31
5.3.1. MINIMUM FCRSYS 31
5.3.2. COPYING FCRSYS 32
5.4. CREATING FCRSYS 33
5.4.1. CREATION PROCEDURE 33
5.4.2. STANDARD FCRSYS 35
5.5. NC-110 SYSTEM MEMORY MAP 37
5.6. FCRSYS SECTIONS 38
5.6.1. SECTION 1 38
5.6.2. SECTION 2 43
5.6.3. SECTION 3 50
5.6.4. SECTION 4 51
5.6.5. SECTION 5 53
5.6.6. SECTION 6 55
5.6.7. SECTION 7 61
5.6.8. SECTION 8 66

6. AXIS MANAGER CHARACTERIZATION 67

6.1. AXIS MANAGER....67
6.2. AXIS FEATURES....68
6.2.1. TYPES OF AXIS....68
6.2.2. DIGITAL/DIGITAL ELECTRONIC WHEEL HANDLING....71
6.2.3. DIGITAL/ANALOG ELECTRONIC WHEEL HANDLING....71
6.3. FILE AXCFIL....72
6.3.1. FEATURES....72
6.3.2. SECTIONS....72
6.3.3. SECTION 1....72
6.3.4. SECTION 2....78
6.3.5. SECTION 3....96
6.4. EXAMPLE OF AXCFIL FILE SETTING....100

7. TECHNOLOGICAL PROCESS CHARACTERIZATION....104

7.1. THE TECHNOLOGICAL PROCESS....104
7.2. GENERATION OF FORMATTED FILES....104
7.2.1. FILES OF ORIGINS....105
7.2.2. CORRECTOR FILES....105
7.2.3. TOOL MONITORING FILES....106
7.2.4. RANDOM TOOL FILES....106
7.2.5. AXIS MOVEMENT FILES....107
7.3. FILE PGCFIL....108
7.3.1. FEATURES....108
7.3.2. SECTIONS....109
7.3.3. SECTION 1....109
7.3.4. SECTION 2....113
7.3.5. SECTION 3....119
7.3.6. SECTION 4....120
7.3.7. SECTION 5....130
7.3.8. SECTION 6....138
7.4. EXAMPLE OF PGCFIL FILE SETTING....143

8. MACHINE LOGIC CHARACTERIZATION....144

8.1. MACHINE LOGIC....144
8.2. FILE IOCFIL....144
8.2.1. FEATURES....144
8.2.2. SECTIONS....145
8.2.3. SECTION 1....145
8.2.4. SECTION 2....150
8.2.5. SECTION 3....156
8.2.6. SECTION 4....162
8.3. EXAMPLE OF IOCFIL FILE SETTING....164

9. EDP AND LOGIC I/O ERRORS AND MESSAGES....166

9.1. EDP ERRORS....166
9.2. EDP MESSAGES....167
9.3. I/O ERRORS....169

10. CHARACTERIZATION ERRORS....171

10.1. ERRORS COMMON TO ALL CHARACTERIZATION FILES....172
10.2. ERRORS IN FILE AXCFIL....175
10.3. ERRORS IN THE CPUs....185
10.4. ERRORS IN FILE PGCFIL....186
10.5. ERRORS IN FILE IOCFIL....190

1. INTRODUCTION

All the products in the NC-110 line require, during installation, a series of characterization procedures. This manual describes how to characterize the products NC-110 MC and NC-110 TC which, for convenience sake, shall be referred to as the NC-110 system.

The NC-110 system is a numeric control system used for controlling a category of machine tools. A machine tool is controlled and driven by a minicomputer which is at the centre of the NC-110 system. The very first time the system is switched on, the minicomputer acts as a generic computing system; it does not have the software or hardware features typical of a numeric control system.

Hence the need to "characterize" the system as a numeric control system for a category of machine tools. The NC-110 system is characterized during installation by means of a series of characterization files which are created following the instructions given in this manual.

The characterization files contain all the parameters and the characteristic system and process variables, whose values define a specific hardware and software configuration of the system. Through these files, the minicomputer obtains all the information it requires to process the software for controlling the machine tool.

In reality, having completed the characterization procedure, the NC-110 system is still not able to control a specific machine tool. To achieve this, a further software configuring procedure must be followed in order to "personalize" the control system for a specific machine tool. The system is personalized by setting up the SIPROM interface using appropriate machine logic software. This second configuring procedure is not dealt with in this manual; it is discussed in the manual entitled "NC-110 MC/TC, SIPROM, Interface programming".

This manual and the one mentioned above both deal with the initial phase of characterization and personalizing the NC-110 system, in which the system is programmed to control a specific machine tool.

2. WHAT IS THE CHARACTERIZATION

This chapter is a general introduction and does not describe any part of the characterization procedure for the NC-110 system.

In general, the information in this chapter represents a key to how to read the manual, so that the user is able to find the procedures to be followed during the installation of the system.

2.1. WORKING ENVIRONMENT

2.1.1. MINICOMPUTER AND NUMERIC CONTROL

The NC-110 system consists mainly of a minicomputer which communicates with a hardware structure designed for the process and handles the process software.

During the installation of the system, the very first time the machine is switched on, the minicomputer is not able to communicate with the various modules which make up the hardware structure for the process, and with its initial features, it is unable to handle the process software. From this point of view, the minicomputer behaves like a generic computing system, capable of providing the operating system features only (basic software).

Characterizing the system consists of a series of procedures which "characterize" the minicomputer as a numeric control system. This group of characterization procedures enables the system to control the hardware for the process, on the one hand, and the process software, on the other.

2.1.2. BASIC SYSTEM AND PROCESS SOFTWARE

Basic system

The basic system consists of the NC-110 system minicomputer, which contains mainly the system CPU, the system memory and the peripheral interface. The CPU (central processing unit) is the fundamental hardware component, common to all computing systems, which processes the basic software and the process software. The process software can only be executed after the system has been characterized.

Process Software

The process software handles all the functions involved in controlling a category of machine tools. It consists of a series of software modules (task) which enable all the activities subjected to the NC-110 system process control to be run independently. The process software consists of the following modules:

- Axis manager;

- Technological process;

- Machine logic.

The axis manager controls the movement of the axes on the machine tool. This control may be divided into two distinct activities; axis interpolation and axis drive.

The technological process is the module responsible for handling the user programs concerning the performance of the machine tool technological cycles.

The machine logic controls the communication between the numeric control system and the machine tool, by means of a software interface, personalized and installed on the NC-110 system.

2.1.3. CHARACTERIZATION LEVELS

Characterizing the NC-110 system concerns besically the two functional elements previously mentioned: the process software end the basic system hich executes it. The characterization procedure is thus articulated on two different levels:

- System level;
- Process level.

System level charecterization concerns the basic system and takes priority over process level characterization which the process software. In other words, the basic system is characterized before the process software.

The basic system is characterized to establish the general operating parameters of the system, the process software and how the process software is to be characterized.

The process software is characterized to establish the parameters and specific data concerning the process software.

2.2. CHARACTERIZATION FILES

2.2.1. GENERAL

The NC-110 system is characterized using a series of characterization files to be created by the user. The files contain information on the various hardware and software modules which form the system.

By means of these files, the NC-110 system is "characterized" as a numeric control system, and is informed of the process software features it must execute.

The characterization files for the NC-110 system are:

- FCRSYS;

- AXCFIL;

- PGCFIL;

- IOCFIL.

Each of these files refers to a particular level and module of the NC-110 system.

Fig. 2.1. shows the characterization levels together with the system modules and their respective characterization files.

File FCRSYS characterizes the basic system and contains all the information on the features of the various process software modules. Once the basic system has been characterized it is able to recognize and handle the process software.

Files AXCFIL, PGCFIL and IOCFIL, as shown in Fig. 2.1., characterize the modules which make up the process software. These files contain the parameters which determine the process control features in general.

All the characterization files are read and created by the system at power on (system bootstrap). The information contained in these files is valid until the files are modified.

It is important to note that whenever a file is modified, the modifications are brought into effect by switching the system off and then on again. This operation enables the system to read the new file and apply the new information.

The system, having tested the characterization files, indicates any errors detected by displaying appropriate messages on two display screens: one for the FCRSYS file, and the other for the three process software characterization files AXCFIL, PGCFIL, IOCFIL.

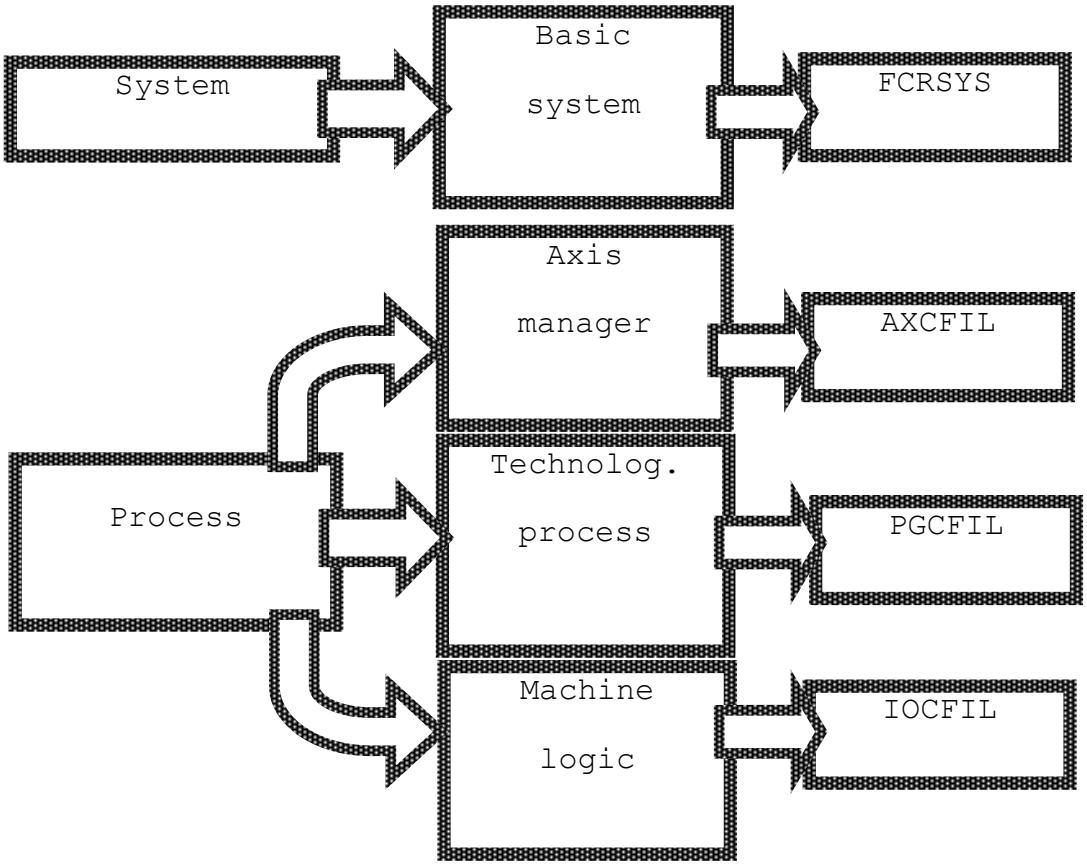


Fig. 2.1. System Modules and Characterization Files

2.2.2. FILE STRUCTURE

All the characterization files have the same general structure, irrespective of the type of file, whereas their internal structure differs according to the information contained. In general, all the files consist of numbered sections, each containing a certain number of records.

The records of the process level files AXCFIL, PGCFIL, IOCFIL all have the same structure. This structure is described in the Section "Record structure" of this chapter. The FCRSYS file records do not have a common structure; their structure depends on the section to which they belong. The structure of the individual records of file FCRSYS is described in Chapter 5: "Basic system characterization".

Each record contains information on a specific hardware or software component of the system, for example, the memory, the drivers, the CPU, the axes, the interpolators, the part programs etc.

It is possible to create comment records within the files, so as to make it easier to read the information in the files. A comment record consists of an alphanumeric string containing a maximum of 32 characters. The first character in the string must be the character ";".

The sections group together the records which contain information on the same software or hardware component. Each section must begin with an asterisk followed by the number of the section. The last section must end with an asterisk. Fig. 2.2. shows the structure of a generic characterization file.

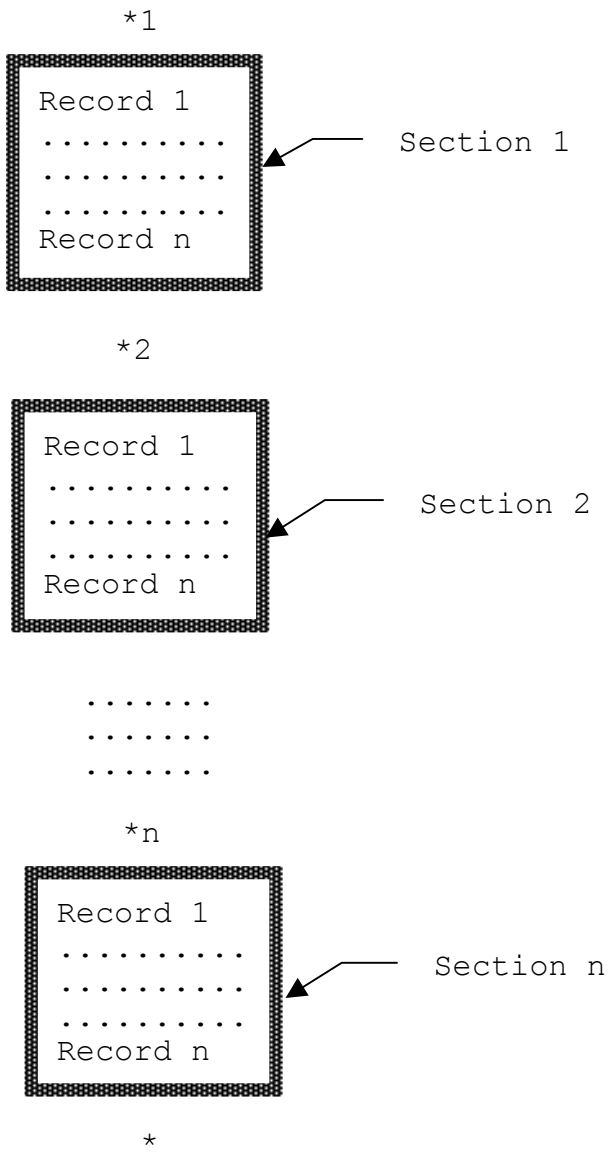


Fig. 2.2. Characterization File Structure

2.3. CHARACTERIZATION ORDER

During the installation of the NC-110 system, the characterization procedure must be performed in a specified order. The following operations must be performed in the order in which they are listed:

- Initialization;

- Characterization of the basic system;

- Charactarization of the process software.

2.3.1. INITIALIZATION

Before beginning the characterization procedure, the system must be initialized. This consists firstly of a few hardware calibrations to be made before switching on the system. Some software adjustments must then be made after power on to reset the CMOS RAM wand to initialize the system memory which is used in the characterization procedure.

During initiwalization, straight after the system has been switched on, the diagnostic test is run on the hardware modules of the basic system. The results of this test are displayed on a system diagnostics display screen described in Chapter 3.

Having initialized the system, we may proceed to configure it following the procedure described in detail in Chapters 5, 6, 7 and 8 of the manual.

2.3.2. BASIC SYSTEM CHARACTERIZATION

The characterization procedure for the basic system is described in detail in Chapter 5. In this section we shall descirbe the general features of file FCRSYS.

File FCRSYS

The basic system is characterized using the characterization file FCRSYS. This file contains a series of records whose contents are common to all the products in the NC-110 line, and some records whose contents are specific for each product.

File FCRSYS consists of 8 sections numbered from 1 to 8. Chapter 5 gives an example of a standard FCRSYS file which ensures the system operation. This file can be installed in the system as it is. To create an extended and "personalized" file, follow the informations given in Chapter 5, in the description of the individual records which form the file.

Testing File FCRSYS

The contents of file FCRSYS are tested by the system the first time it is switched on after the installation of the file in the system (bootstrap). If any errors concerning the information contained in FCRSYS are detected during this phase, error messages appear on the display screen for the FCRSYS file diagnostics.

The diagnostics screen contains the error messages concerning the FCRSYS file records and the memory modules affected by the calibrations made in FCRSYS. This screen is described in Chapter 4 "Characterization diagnostics".

2.3.3. PROCESS SOFTWARE CHARACTERIZATION

The process software characterization procedure is described in detail in Chapters 6, 7 and 8. This section gives a description of the general features of the characterization files used.

Characterization files

The process software is characterized using the following three characterization files, each of which concerns one of the three modules which make up the process software.

- AXCFIL;

- PGCFIL;

- IOCFIL.

File AXCFIL characterizes the axis manager and contains the information on the interpolation and drive of the axes.

File PGCFIL characterizes the technological process and contains the information on the triletterals allowed, the control of user programs (part programs) and the system variables.

File IOCFIL characterizes the machine logic software and contains the information on the communication between the control system and the machine tool established using the SIPROM interface.

Record Structure

The records of the three process level files all have the same structure. Each record consists of a triletteral which is associated with specific parameters. The triletteral names may not be modified, whereas the values of the parameters must be assigned by the user.

The structure of a record is as follows:

TRL=par1,par2,... ,parn

where:

TRL	Mnemonic triletteral which configures a specific software or hardware component
par1,...,n	Series of parameters which provide the system with characteristic values which refer to the software or hardware component considered

When declaring the records it is possible to omit one or more parameters. In this case, however, any separating commas between one parameter and another must be indicated.

Chapters 6, 7 and 8 contain a list and description, section by section, of the individual records which the user must declare in the various files.

Testing the Characterization Files

The contents of the process software characterization files are examined by the system the first time it is switched on after the files have been installed in the system. If any errors are detected, the system indicates the type of error on a screen for the process software characterization files diagnostic.

The screen gives information on the type of file and type of error detected. This screen is described in Chapter 4 "Characterization diagnostics".

3. SYSTEM INITIALIZATION

This chapter describes the initialization procedure which must be followed when the system is first installed, before beginning the software characterization procedure.

3.1. SYSTEM POWER ON

Switch on the system using the switch ON and holding down the HOLD button. After a few seconds the following information regarding the hardware diagnostics of the basic system appears on the display:

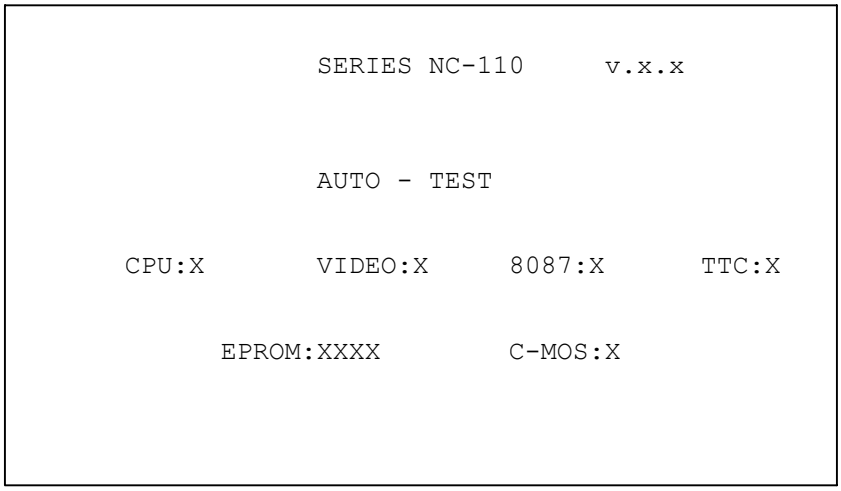


Fig. 3.1. Basic System Diagnostics Screen

The parameters printed in bold face indicate, as follows:

x.x	CPU firmware release number
X	Codes indicating the type of error detected on the specified module. The codes which may be displayed are listed, module by module, in Table 3.1.
XXXX	Codes indicating the type of error detected in the EPROMs. The codes which may appear on the display are listed in Table 3.1.

The display screen shows infswation on the basic system hardware diagnostics, which consists of the functional checks of the following modules:

- Board M2001 (CPU):

. processor 8086, coprocessor 8087, monitor;

. TIMER;

. PIC;

. EPROM;

. CMOS;

- Board M2030 (display):

. TTC interface.

Errors in the basic system modules

If an error is detected in the modules tested, the system does not move on to the initialization process and the execution of the software is interrupted.

The type of error in a basic system module is indicatod on the screen, in the form of a code which appears after the name of the module, as described in Fig. 3.1. In this figure the codes are symbolically represented by one or more "X"s.

Table 3.1. - Error Codes for Hardware Modules

MODULE	CODE	RESULT	NOTE
CPU	Y	Pass	1
	1	Error in RAM	1
	2	Error in TIMER	1
	3	Error in PIC	1
	4	Deffective EPROMS 0L	1
	5	Deffective EPROMS 0H	1
DISPLAY	Y	Pass	2
	N	Fail	
8087	Y	Pass	
	N	Fail or not present	
TTC	Y	Pass	
	N	Fail	
EPROM	0000	Pass	
	----	Not declared	
	HHHH	Hexadecimal code indicating defective EPROMs	
C-MOS	Y	Pass	3
	N	Fail	
	0	Parity error	
	----	Not declared 2 segment	

NOTES:

1. This test is run on the local bus of the CPU.

2. This test is run on the display RAM (M2030).

3. The CMOS memory test is run on segment 2 of the system memory on the board M2001 (base software).

In the event of a parity error on a CMOS memory try switching the system off and then on again. If the error persists, the CMOS memory is defective.

EPROM

The EPROM test is run from the hexadecimal address 60000 to address 7FFFF. These two addresses indicate a memory area of 128 Kbytes containing the base software.

Every EPROM is given an alphanumeric code which indicates the sequential number of the EPROM from the inside of the unit (from 0 to 3) and the position of the memories (high or low). For example the code 3H indicates EPROM number 3 in the "high" position, i.e. containing the eight "high" bits which may be addressed by the system.

Any defective EPROMs are indicated on the display using a hexadecimal code of four figures. The hexadecimal value indicating the defective EPROMs must be converted to BCD (binary-coded decimal) code. In this way, a binary value of 16 bits is obtained (four bits for each hexadecimal figure) of which only the first eight bits are significant. Each bit corresponds to a specific EPROM and has the following meaning:

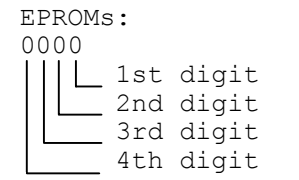
- Each zero bit indicates a working EPROM;

- Each one bit indicates a defective EPROM.

The binary value bits correspond to the codes indicating the individual EPROMs as shown in Table 3.2.

Table 3.2. - Decoding Hexadecimal Values

1st DIGIT	
BIT VALUE	EPROM CODE
1	BxxE
2	BxxH
4	BxxA
8	BxxD



where: xx - number version software.

Example:

If, beside "EPROM" on the basic system diagnostics screen, the following message appears:

EPROM:0007

the hexadecimal value 0007 converted into binary code becomes:

0000 0000 0000 0111

In this binary value bits 1, 2, 3 of the first digit are at 1. From Table 3.2. we can see that EPROMs BxxE, BxxH, BxxA are defective, whereas EPROMs BxxD are working.

3.2. SOFTWARE CALIBRATIONS

If no errors ara detected during tha basic system diagnostic test, the diagnostics screen is displayed for a few saconds and then the debug menu appears (remember that the system is switched on in the hold condition):

[Modify | Copy | Fill | Display | Test | Exit]

Resetting the CMOS RAM

The debug menu is used to reset the CMOS RAM. This operation is required to avoid errors arising during the initialization procedure.

Select option F (Fill) on the debug menu by pressing key F and the system requests the information indicated in the following tables.

Reset initialization:

THE SYSTEM REQUESTS	THE USER MUST ANSWER
MODE	T (TO MODE)
TYPE	B (BYTE TYPE)

Resetting segment 4:

THE SYSTEM REQUESTS	THE USER MUST ANSWER
FROM	4000:0
TO	4000:FFFF
VALUE	0

Wait for the system to display "FROM" again.

Resetting segment 3:

THE SYSTEM REQUESTS	THE USER MUST ANSWER
FROM	3000:0
TO	3000:FFFF
VALUE	0

Wait for the system to display "FROM" again.

Resetting the second part of segment 2:

THE SYSTEM REQUESTS	THE USER MUST ANSWER
FROM	2000:8000
TO	2000:FFFF
VALUE	0

Wait for the system to display "FROM" again.

Resetting the first part of segment 2:

THE SYSTEM REQUESTS	THE USER MUST ANSWER
FROM	2000:0
TO	2000:7FFF
VALUE	0

Wait for the system to display "FROM" again.

To bring the calibrations into effect switch the system off and then on again.

System memory initialization

Having reset the CMOS RAM, the MP0 system memory area must be initialized so that the basic system (FCRSYS) can be configured.

Press key **P0** to change partition and press key **SEND** to move into JOB mode.

To initialize the MP0 memory area, the following command must be executed:

INI,MP0,/MP0

The coded message FILMS2 02 appears on the display and this requests permission to begin initialization. By answering "N" the command is cancelled. By answering "Y" the command is carried out and the coded message FILMS2 03 appears on the display indicating that the MP0 memory area has been initialized. This can be checked by keying in:

```
DIR,/MP0
```

The system displays:

```
DIR, /MP0          MP0
NAME               SECT        LREC        ATT
FREE SECTORS       27
```

This message indicates that the MP0 memory exists, that it has been initialized and that it consists of 27 sectors and does not contain any files.

4. CHARACTERIZATION DIAGNOSTICS

4.1. FCRSYS FILE DIAGNOSTICS

4.1.1. DIAGNOSTICS SCREEN

During the bootstrap, having tested the FCRSYS file, the system displays the following information:

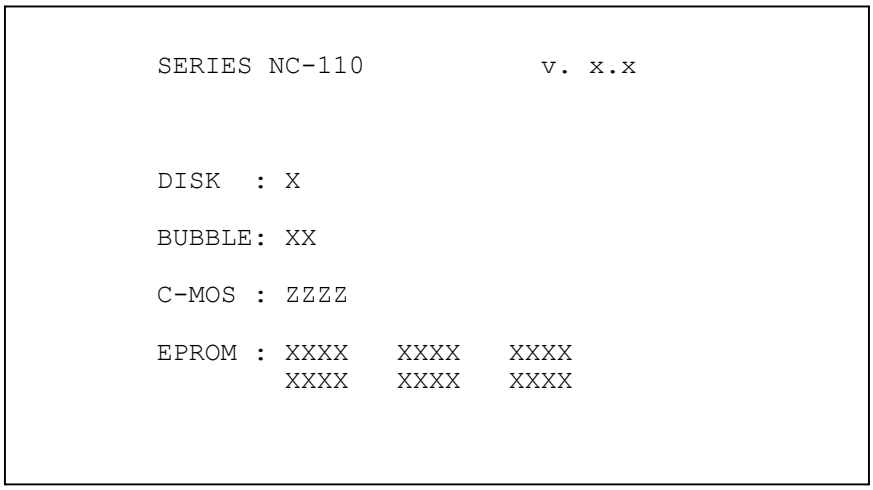


Fig. 4.1. FCRSYS File Diagnostics Screen

This screen shows the errors in the hardwars modules of the basic system declared in the FCRSYS file and the errors contained in ths FCRSYS file records.

The parameters indicated in bold face in the diagram give the following information:

x.x	Basic software release number (operating system)
X	Code indicating the type of error detected on the disk used. The codes which may appear on the display are listed in Table 4.1.
XX	Codes indicating the type of error detected in the bubble memory. Each "X" corresponds to one of the two bubble memories declared. The codes which may appear on the display are listed in Table 4.1.
ZZZZ	Codes indicating the type of error detected in the CMOS memories. Each "Z" corresponds to one of the four CMOS memories declared. The codes which may appear on the display are listed in Table 4.1.

XXXX — Codes indicating the type of error detected in the EPROMs. Each group of four "X"s corresponds to a specific process software module resident in a maximum of four EPROMs. The screen shows the results of a maximum of six blocks of EPROMs. The codes which may appear on the display are listed in Table 4.1.

Table 4.1. - FCRSYS Diagnostics Error Codes

MODULE	CODE	MEANING
DISK	Y	Pass
	N	Fail
	0	Parity error
	?	Not present
	1	8031 error
	2	EPROM error
	3	DUAL-PORT RAM error
	9	Disk not inserted
	A	Write protection
	B	Disk not formatted
	D	Disk with damaged sectors which may not be recovered
BUBBLE	Y	Pass
	N	Fail
	-	Not present
C-MOS	Y	Pass
	N	Fail
	0	Parity error
	-	Not present
EPROM	0000	Pass
	????	Not present
	HHHH	Hexadecimal value indicating the defective EPROMs

If the FCRSYS file is correct and there are no errors is the hardware, the system displays another screen for the diagnostics of the process software characterization files (see the Section "AXCFIL, PGCFIL, IOCFIL file diagnostics" in this chapter).

4.1.2. ERRORS IN THE HARDWARE MODULES

The hardware diagnostics tests the following hardware modules:

- DISK = a disk unit (floppy disk or hard disk);
- BUBBLE = two bubble memories;
- C-MOS = four CMOS RAMs;
- EPROM = six EPROMs.

If any errors are detected in one or more of these hardware modules, the following message appears on the last line of the screen:

** INIT ABORTED **

DISK, BUBBLE, CMOS Memories

The type of error detected in a memory is indicated directly on the display, by means of a code which appears after the name of the module, as indicated in Fig. 4.1.

Table 4.1. shows all the codes, which may be displayed and the corresponding type of error.

For CMOS memories, the codes are based on their position. The defective memory is identified according to the position of the characters in the code. For example, the system displays:

C-MOS : YN0-

The first CMOS memory is working, the second is defective, the third has a parity error and the fourth is not inserted in the system.

EPROM

The diagnostics tests all the EPROM blocks declared in the FCRSYS file. A maximum of six EPROM blocks may be declared. Each module contains a specific NC-110 system software module. Each of these software modules is declared and specified in the FCRSYS file using the EPx record (x = 1-6).

Each individual EPROM, belonging to one of the six groups, is identified by an alphanumeric code indicating the position of the EPROM with respect to the inside of the unit and the position of the memory.

For example the code Bxx5 indicates the position of EPROM block oh the module M2001. The allocation of all the EPROM blocks on module M2001 shown follow.

The defective EPROMs are indicated on the display screen using a four-figured hexadecimal value. The display can show a maximum of six hexadecimal values which correspond to six EPROMs. The specific EPx record, which declares the presence of the EPROM in the FCRSYS file, corresponds to the position in which the hexadecimal value appears on the screen as follows:

(EPl)	(EP2)	(EP3)
XXXX	XXXX	XXXX

(EP4)	(EP5)	(EP6)
XXXX	XXXX	XXXX

The hexadecimal velue indicating the defective EPROMs must be converted into BCD (binary-coded decimal) format. In this way, e binary velue of 16 bits is obtained (four bits for each hexadecimal figure). Each bit corresponds to a specific EPROM, and has the following meaning:

- Each zero bit indicates a working EPROM;

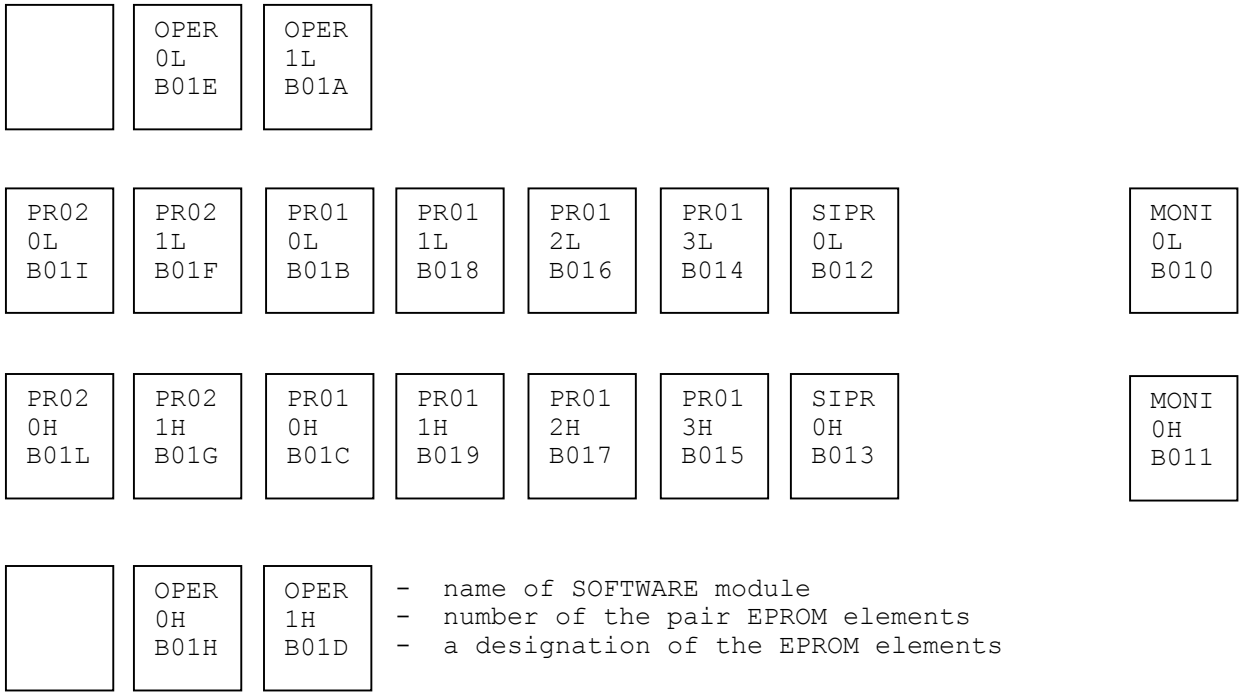
- Each one bit indicates a defective EPROM.

The binary value bits correspond to the codes indicating the individual EPROMs as shown in Table 4.3.

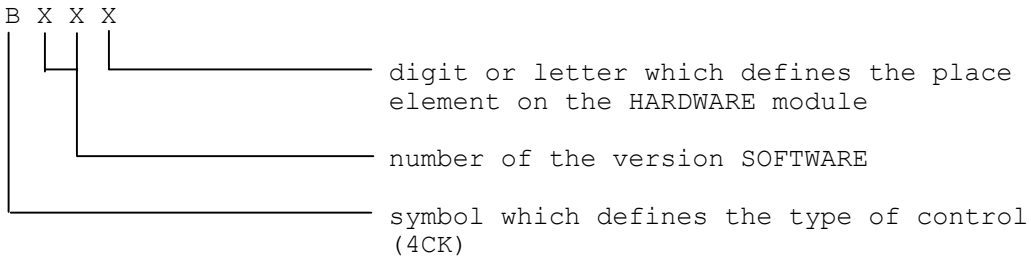
Table 4.3. - Decoding the Hexadecimal Codes

BIT	BIT VALUE	process1	process2	SIPROM	LOGICA
1	1 2 4 8	BxxB BxxC Bxx8 Bxx9	BxxI BxxL BxxF BxxG	Bxx2 Bxx3	BxxJ BxxK
2	1 2 4 8	Bxx6 Bxx7 Bxx4 Bxx5			

The allocation and designation of EPROM elements on the module M2001 and the allocation of SOFTWARE modules in the EPROM elements.



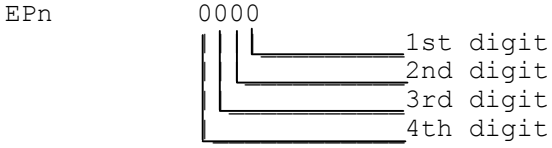
Where:



Where:

OPER	Operating system
PRO1	Process 1
PRO2	Process 2
SIPR	Siprom
MONI	Monitor

Values must be read from right to left:



Example:

Suppose that the following results of the EPROM diagnostics are shown on the display.

EPROM: 000A 0000 0000

???? ???? ????

This result indicates the following situation:

- Block EP1: defective EPROMs indicated by the hexadecimal value 000A;

- Block EP2: working EPROMs;

- Block EP3: working EPROMs;

- Block EP4: EPROMs not present;

- Block EP5: EPROMs not present;

- Block EP6: EPROMs not present.

The hexadecimal value given for block EP1, converted into BCD format is:

0000 0000 0000 1010

In this binary value, bits 2 and 4 are one, so it can be seen from Table 4.3. that the defective EPROMs are BxxC and Bxx9.

4.1.3. ERRORS IN THE FCRSYS FILE

If formal or logic errors are detected in the FCRSYS file records, the following message appears on the last line of the screen:

** INIT ABORTED **

In the FCRSYS file, there may be syntax arrors or logic errors in the declarations made. If this is the case, the initialization of the FCRSYS file is interrupted and the process software (section 3 of FCRSYS) is not activated.

To check the type of error, press key **PO** to move from the diagnostics screen to the JOB screen. The type of error detected in FCRSYS is indicated on the first two lines of the JOB screen.

To correct the error press **SEND** to move into editor mode and modify the file to eliminate the cause of the error.

4.2. AXCFIL, PGCFIL, IOCFIL FILE DIAGNOSTICS

4.2.1. DIAGNOSTICS SCREEN

If errors are detected in the process software characterization files, the system indicates the error and the file containing it on a specific diagnostics screen.

The diagnostics screen shows a series of lines which give the information required to identify the error. This screen is illustrated in Fig. 4.2.

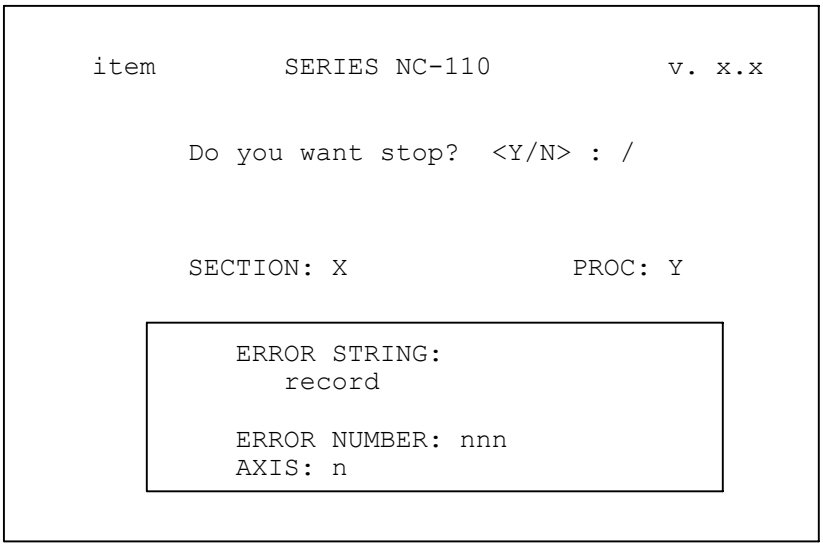


Fig. 4.2. Process Software File Diagnostics Screen

The parameters shown is bold type in the diagram have the following meaning:

item	Specifies the type of file containing the error. It may have the following meanings: AXIS = AXCFIL PROC = PGCFIL LOGIC = IOCFIL
x.x	Process software release number
X	Number of the file section containing the error
Y	Number of the process in which the error is present
record	Record containing the error
nnn	Error code: from 0 to 209. The error messages corresponding to the displayed code are listed in Appendix B
n	Name of the axis in whose subsection the error was detected during the check of section 2 of file AXCFIL

4.2.2. ERRORS IN THE AXCFIL, PGCFIL, IOCFIL FILES

Having displayed the diagnostics screen, the file test may be interrupted or reactivated as the user wishes.

If the user answers "Y" to the question "Do you want stop7", the file test is interrupted, the screen illustrated in Fig. 4.3. appears and the JOB partition is enabled for the correction of the error.

If the user answers "N", the record containing the error is not interpreted by the system, the file test continues and stops at the next error, if there is one.

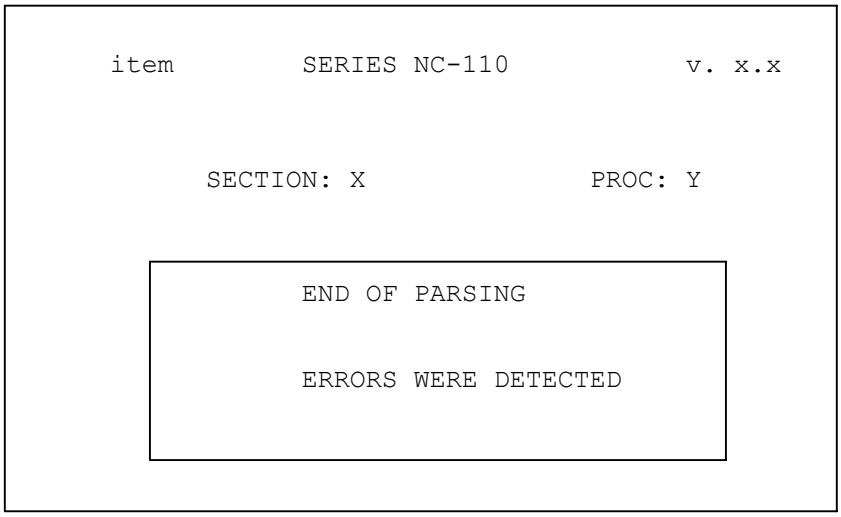


Fig. 4.3. Error Correction Screen

On this screen, the parameters shown in bold type ara the same as in Fig. 4.2.

The errors which may occur in the process software characterization files may be divided into four categories:

- Syntax and format errors comaon to the three files;
- Errors detected during the AXCFIL file test;
- Errors detected during the PGCFIL file test;
- Errors detectad during the IOCFIL file test.

The syntax and format errors with a code number from 1 to 100 are common to the three files. The errors with a code number from 101 to 209 are specific for the individual process characterization filas. The list of the error messages is given in Appendix B, "Characterization errors".

5. BASIC SYSTEM CHARACTERIZATION

5.1. THE BASIC SYSTEM

The basic system consists of a minicomputer which performs all the processing functions concerning the control of a process. The minicomputer consists mainly of the following modules:

- CPU;
- Memory;
- Peripheral interface.

These modules enable the system to handle the process software by communication with a hardware structure specifically designed for the process. Both the process software and the hardware used for the process have a modular structure, and the modules are declared when the system is being configured.

The basic system is configured using the FCRSYS file which contains all the information the basic system requires to handle the process software.

The modules which make up the process software are declared in file FCRSYS. The hardware modules used for the process are declared in the individual files which configure the process software (AXCFIL, PGCFIL, IOCFIL).

File FCRSYS can be installed in the system in the following two ways:

- Loading from an external unit;
- Direct creation of the file.

In either case a file with the name FCRSYS must be created in the MPO system memory erea.

5.2. FILE FCRSYS

File FCRSYS consists of eight sections, each one concerning the characterization of a particular module or component of the basic system. Each section consists of records with various structures. The structure of the records is described, section by section. The general structure of the file is described in the Section "File Structure" of Chapter 2.

All the informations contained in FCRSYS file are interpreted and activated during the system power on. Any modifications made to any section of the file are brought into effect the following time the system is switched on.

If the FCRSYS file is not yet installed in the system or if, due to errors in the media containing it, it is not accessible, the system indicates the error by sending the message FILMS1_20.

At power on, the FCRSYS test may be avoided by pressing the **HOLD** button at the same time as the **ON** button. In this way, the system moves directly to the debug utility menu.

At power on, the test on section 3 of FCRSYS may be avoided by pressing **CYCLE START** at the same time as the **ON** button. In this way, the initialization of the application software (process software) is not activated. In this case, the system sends the message FILMS1_21.

5.3. LOADING FCRSYS FROM AN EXTERNAL UNIT

The FCRSYS file can be installed in the system by transferring a file from another memory media. In this case, the devices and peripherals used for the transfer must be present, and the file must be copied in the MPO system memory.

5.3.1. MINIMUM FCRSYS

To load the file from an external unit, a minimum FCRSYS file must be created in any case. The FCRSYS file must contain the declaration of the hardware modules required for transfer and the driver of the peripheral used. To create this minimum file, the system must enter editor mode and open the FCRSYS file in the MPO memory area, using the following command:

```
EDI,FCRSYS/MPO
```

At this point the minimum FCRSYS file must be installed according to the peripheral used for loading.

The following sections describe the various methods of loading the file and the lists of the corresponding minimum FCRSYS files which must be created.

Loading from CMOS RAM

The FCRSYS file can be loaded from the CMOS memory to another control system with similar features.

This procedure requires the declaration of the CMOS modules (section 1) and the activation of the MP3 memory area driver (section 2).

minimum FCRSYS list:

```
*1
MPX=Y2Y2----
*2
XAMEP3,OC,3,4000,0100
*
```

Loading from Punched Paper Tap Using a Reader

This procedure requires the activation of the paper tape reader driver.

Write the following minimum FCRSYS:

```
*2
XAREAD,09,0,6
*
```

Loading from Punched Paper Tape Using a Teleprinter

This procedure requires the activation of the teleprinter driver.

minimum FCRSYS list:

```
*2
XAT485,07,0,4
*
```

Loading from a Magnetic Cassette

This procedure requires the activation of the magnetic cassette reader driver.

Write the following minimum FCRSYS:

```
*2
XATEAC,OB,0,3
*
```

5.3.2. COPYING FCRSYS

Having installed the minimum FCRSYS according to the media selected, leave editor mode by pressing **ESCAPE** twice, then:

- switch off the system;
- connect the peripheral used;
- switch on the system again.

The system runs the basic system diagnostics and the diagnostics screen appears on the display, it then tasts the minimum FCRSYS showing on the corresponding screen the message INIT ABORTED. Press **P0** to leave this screen and then press **SEND** to move into JOB mode.

At this point, the FCRSYS file resident on the selected media must be copiad in the MPO memory. The following table shows the commands the user must make, according to the media containing the FCRSYS file to be copied.

COMMAND	MEDIA
COP,FCRSYS/MP3,/MP0	CMOS
COP,/PRO,FCRSYS/MPO	Paper tape reader
COP,FCRSYS/CTO,FCRSYS/MPO	Magnetic cassette

Having sent the **COP** command, the message FILMS2_05 appears to indicate that the minimum FCRSYS file is already present in MP0, and confirmation is requested to copy the file. Press **Y** to copy the FCRSYS file in MPO and cancel the minimum FCRSYS.

The system indicates the end of the operation using the message FILMS2_03.

At this point, MPO contains the FCRSYS file which may be accessed for reading and writing in editor mode. In particolar, modifications may be made to the file so as to adapt it to the specific configuration of the system. This new file is activated the next time the system is switched on.

5.4. CREATING FCRSYS

The information in this section of the Chapter may be used both to create the FCRSYS file and to modify the FCRSYS file loaded from an external unit.

5.4.1. CREATION PROCEDURE

This section describes the procedures for testing, correcting and activating the FCRSYS file.

Move into editor mode and open the FCRSYS file using the command:

EDI,FCRSYS/MPO

Write the entire file, complete with all the sections, following the instructions given in the rest of the Chapter. Switch the system off and on to activate the file.

File Diagnostics

At power on, the system runs the FCRSYS test end indicates on the diagnostics screen any syntax or congruence errors made by the user. Eliminate the errors by leaving the diagnostics screen end enabling the JOB partition, as described in the Section "FCRSYS file diagnostics" in Chapter 4. Consult this section for all operations ievolved in testing and correcting the FCRSYS file.

Having corrected the file, switch the system off and on to bring into effect the modifications made and run the file diagnostics. If other errors are indicated, repeat the correction procedure, switch the system off and on again until no further syntax or record congruence errors are detected.

Initialization of the CMOS Area

When the diagnostic test does not detect further syntax or congruence errors, a PARITY ERROR is indicated. This is due to the fact that the CMOS memory areas have not yet been initialized.

Key in the following command, to initialize the MP3 memory area:

```
INI,MP3,/MP3
```

Opening Files Declared in Section 7

Section 7 of FCRSYS contains the process software characterization files (AXCFIL, PGCFIL, IOCFIL), the system message files (EDPERR, EDPMSG, IOERRM, PROERR, MESSAG) and the machine logic messages. In reality these files have not yet been created, so the system interrupts the FCRSYS test and displays the error messages in code (FILMS).

So that the system can complete the diagnostics, a few modifications must be made to anable the modules which are not yet present in the system.

For the characterization and message files, the commands which enable these files to be opened must be keyed in. This operation enables the FCRSYS diagnostic test to be completed, even though the files opened are not installed in the system. These files are installed once the FCRSYS file test has been completed.

Key in the following commands to open the files declared in section 7.

```
CRE,EDPERR/MP3,40,30
CRE,EDPMSG/MP3,40,9
CRE,IOERRM/MP3,40,44
CRE,PROERR/MP3,32,111
EDI,FORMAT/MP3
EDI,AXCFIL/MP3
EDI,PGCFIL/MP3
EDI,IOCFIL/MP3
EDI,PROTEC/MP3
```

As far as the machine logic is concerned, the character ";" must be inserted before the first character of the EP4 record in section 1. This character must be deleted when the first machine logic program is written. Insert the character ";" before the first character of the EPX record in Section 1 until the SIPROM is installed.

MP1 Memory area

The XAMEP1 record in section 2, which refers to the permanent memory area MP1, must be declared in full at the end of the process software characterization procedure, when the system has tested and interpreted all the characterization files.

When the FCRSYS file is being installed, the character ";" must be inserted before the first character of the XAMEP1 record. The last two parameters of the record must be replaced by a series of "?"s. The record must be declared in the following way:

```
;XAMEP1,OC,1,3FFF,????
```

At the end of the process software characterization procedure, when the system runs the bootstrap of all the characterization files, the following message is displayed:

```
WARNING: XAMEP1,OC,1,xxxx,YYYY
```

where:

xxxx	Initial address of the user memory which must be declared in the XAMEP1 record and replace the first group of "?"s. This address is also declared in the NBP record of AXCFIL
YYYY	Length, expressed in sectors, of the user memory which must be declared in the XAMEP1 record and take the place of the second group of "?"s.

To modify the XAMEP1 record, recall the FCRSYS file and insert the values indicated by the system message into the record.

For further information on how to declare the XAMEPl and NBP records, consult respectively "Section 2" of this Chapter and "Section 1" of Chapter 6.

5.4.2. STANDARD FCRSYS

The following is a standard FCRSYS file list which enables the basic system to be configured under standard operating conditions. This file can be installed in the system as such and only requires a few modifications to adapt it to the specific control system in the user's possession. The notes after the file list describe a few modifications to be made to the file, so that it may be used correctly.

If the user wishes to create a "personalized" file on the basis of his specific requirements, he can create an ex novo file, following the instructions give in the following Section "FCRSYS sections".

```
* 1
MPX=Y2Y2----
EP1=Y4 ,A000,0008
EP2=Y4 ,8000,0004
EP3=Y4,5000,0002        (only declare with M.L. on EPROM)
EP4=Y4,E000,0002        (only declare if SIPROM is present)
*2
XACONS,O4
XAGRAF,OD
XAMEP1,OC,1,3FFF,????   (declare at end of installation)
XAMEP2,OC,2,4800,0100
XAMEP3,OC,3,4000,0100
XAT485,07,0,4
*3
XMAINP
*4
/MP3
/DYO
ISO
*7
FILMS1,EDPERR/MP3
FILMS2,EDPMSG/MP3
FILMS3,IOERRM/MP3
FILMS4,PROERR/MP3
FILMS5,MESSAG/MP3
FORMAT,FORMAT/MP3
AXCONF,AXCFIL/MP3
IOCONF,IOCFIL/MP3
PGCONF,PGCFIL/MP3
;FILCMD,PROTEC/MP3
*8
SIPROM
DEBUG
PIMP
CKSUM
*
```

NOTES. When installing the standard file, these instructions must be followed:

- Section 1, EP3 record: if the machine logic is present, enable record EP3 by deleting character ";". If the machine logic consists of 2 EPROMs only, the last value declared on the RH side must be 0002. If the machine logic is stored in CMOS RAM, replace Y4 with N2;
- Section 2, XAMEP1 record: this record must be declared in full at the end of the process software characterization procedure only (see Section "MP1 memory area" of this Chapter).

5.5. NC-110 SYSTEM MEMORY MAP

The NC-110 system memory map consists of 16 memory segments with a capacity of 64 KB each. Each segment contains one or more system software modules and is identified by the start hexadecimal address. The segments are made up both of CMOS RAM and EPROM memory.

The following table gives the NC-110 system memory map, so that the configured memory areas in the FCRSYS file may be more easily identified.

INITIAL ADDR	FINISH ADDR	TYPE	CAPACITY	DESCRIPTION
00000	OFFFF	CMOS	64 K	Work area monitor + trap Work area SIPROM Work area diagnostics
10000	1FFFF	CMOS	64 K	-I/0 -Video -Transducer -A/D -D/A -Communication
20000	3FFFF	CMOS	128 K	MP0 Work area operating system Work area MC/TC process M01
40000	4FFFF	CMOS	64 K	MP2/MP3
50000	5FFFF	EPROM	64 K	Siprom object
60000	7FFFF	EPROM	128 K	Operating system
80000	8FFFF	EPROM	64 K	MC/TC PROCESS 2
90000	9FFFF	EPROM	64 K	AXIS MANAGER
A0000	DFFFF	EPROM	256 K	MC/TC PROCESS 1
E0000	FFFFF	EPROM	64 K	Siprom compiler
F0000	FFFFF	EPROM	64 K	Monitor and diagnostic

5.6. FCRSYS SECTIONS

5.6.1. SECTION 1

This section contains the configuration parameters for all the memory modules in the NC-110 system.

The information contained in this section can also be used to diagnose the declared memory modules.

If syntax errors relative to this section are detected, the system displays the message FILMS1_22.

If the declarations contained in this section are not consistent with the existing hardware, the system displays the message FILMS1_23.

If the directory of EPROM modules has not been created due to an error, the system sends the message FILMS1_24.

Each record of section 1 contains the configuration parameters for a particular system memory module.

Records Structure:

mod-code = char-par

where:

mod-code	Identification code of the type of memory module to be configured
char-par	Characteristic parameters of the memory module to be configured

The type of module is defined by an identification code made up of three letters which must always be entered in the first 3 characters of the record. The relation between symbolic code and type of module is shown in Table 5.1.

Table 5.l. - Memory Modules Identification Codes

SYMBOLIC CODE	TYPE OF MODULE
MBX	Bubble memory
DSK	Disk (hard disk, floppy disk)
MPX	Permanent CMOS memory
EPx	EPROM modules (x = 1-6)

SECTION 1 RECORDS:

RECORD MBX

This record declares the presence of the bubble memory modules. The system can support a maximum of two bubble memory boards with maximum capacity of 512 KB each (multibubbles).

MBX=aa

Where:

aa — Codes which indicate the presence of bubble memory boards set respectively as first and second board according to the position of the code. The codes can have the following values:
Y: the board is present and the user wishes to activate the diagnostic tests;
-: the board is not present.

Characteristics:

Table 5.2. provides the symbolic names for the bubble memory modules.

Table 5.2. – Symbolic Names for Bubble Memory Modules

MEMORY MODULE	SYMBOLIC NAME	NOTES
512 KB module (Submodule No. 1) Set as first board	/MBO	A single board with 4 bubble memory devices of 128 KB each
512 KB module (Submodule No. 2) Set as first board	/MB1	
512 KB module (Submodule No. 3) Set as first board	/MB2	
512 KB module /MB3 (Submodule No. 4) Set as first board		
512 KB module /MB4 (submodule No. 1) Set as second board		A single board with 4 bubble memory devices of 128 KB each
512 KB module /MB5 (Submodule No. 2) Set as second board		
512 KB module /MB6 (Submodule No. 3) Set as second board		
512 KB module /MB7 (Submodule No. 4) Set as second board		

P.S. The bubble memory not use now.

Example:

MBX=Y-

This record indicates that a 512 KB bubble memory board is present, and set as first board. The board must be tested.

RECORD DSK

This record declares the presence of a board which handles the magnetic disks, both hard disk and floppy disk.

DSK=a

Where:

a — Code indicating the presence of the board which handles the magnetic disks. The code can have the following values:
Y: the board is present and the user wishes to activate the diagnostic tests;
-: the board is not present.

RECORD MPX

This record contains the configuration parameters of the CMOS RAM permanent memory modules with battery backup.

MPX=anananan

Where:

a — Code indicating the presence of a specific permanent memory module. This parameter is associated with parameter n which follows it and identifies the position of a particular module. It can have the following values:
Y: the memory module is present and the user wishes to activate the diagnostic tests;
-: the memory module is not present. In this case, code n is of the "dummy" type.

n — Code which specifies the capacity of the permanent memory module, expressed in KB. This parameter is associated with the parameter that precedes it, and is only significant if a = Y. It can have the following values:
1: 32 KB memory module
2: 64 KB memory module

P.S. The magnetic disks not use now.

Characteristics:

The system memory segments, in which the areas of permanent memory corresponding to the declared modules are allocated, are identified by fixed addresses that the user must not modify. Table 5.3. shows the start addresses (in hexadecimal) of the segments in which the areas of permanent memory are allocated.

The memory segment reserved for the system is allocated at hex address 2000:0. This segment contains the system software working area and the characterization file FCRSYS. This is therefore always present, it must not be declared in FCRSYS and is automatically tested by the system.

Table 5.3. - Permanent Memory Allocation

MEMORY MODULE	ALLOCATION ADDRESS
MP1	3000:0
MP2 and/or M13	4000:0

Example:

MPX=Y2Y2----

This record defines three permanent memory modules. These modules correspond to memory areas allocated in segments 3000:0, 4000:0, with capacities of 64, and 64 KB respectively. The diagnostics are activated for all the modules.

RECORD EPx

This record contains the configuration parameters of the system software modules and of the EPROMs containing them.

EPx=an,start-add,ep-num

Where:

x — Identification number of the software module stored on EPROM: from 1 to 6.

a — Code which indicates the presence of the software module. The code can have the following values:
Y: the software module is present and the user wishes to Activate the diagnostics;
N: the software module is not present and the user does not want to activate the diagnostics.

n — Identification code for the type of EPROM installed. It can have values 1, 2, 4, 8 with the following meanings:
1 = EPROM type 2764
2 = EPROM type 27128
4 = EPROM type 27256
8 = EPROM type 27512

start-add	Start address of the system memory segment in which the software module is allocated. The address is made up of 4 hexadecimal digits.
ep-num	Number of EPROM elements which contain the declared software module. The number is made up of four hexadecimal digits. Up to 16 EPROM elements can be installed: from 0 to F hexadecimal.

Characteristics:

In addition to the software module reserved for the basic software, the system is able to handle other modules with a maximum size of 128 KB. The module reserved for the basic software is allocated at hex address 6000:0 and is A 256 KB module. This module is Always present in the system, it must not be declared in FCRSYS and is automatically tested by the system.

Example:

EP1=Y4, A000, 0008

This record declares a software module allocated at hex address A000:0. 8 EPROM elements type 27256 are installed, and the diagnostics are enabled to check that the checksum is correct.

5.6.2. SECTION 2

This section contains the configuration parameters for the drivers of the peripherals handled by the system.

If syntax errors occur in this section of FCRSYS, the system sends the message FILMS1_15.

If the driver is not present in EPROM, the system sends the message FILMS1_17.

If the logic number is not consistent with the driver, or is not between O-F hexadecimal, the system sends the message FILMS1_18.

Records Structure:

log-name,log-num[,par1][,par2][,par3]

Where:

log-name	Logic name of the driver to be configured. The name is composed of six alphanumeric characters, the first of which must be alphabetic. Table 5.4. shows the logic names of the drivers relative to the peripherals handled by the system.
log-num	Logic number which identifies the peripheral handled by the specified driver. The number is made up of two hexadecimal digits. Table 5.4. shows the logic numbers of the peripherals handled by the system.

parl...par3	Series of optional parameters which allow the following to be configured: - the serial connection line if the driver refers to one of the available peripherals; - the permanent memory if the driver is XAMEP.

The relation between logic name, logic number and peripheral is shown in Table 5.4., Table 5.5. shows the symbolic names of the peripherals.

Table 5.4. - Logic Names and Numbers

LOGIC NAME	LOGIC NUMBER	PERIPHERAL
XACONS	04	Console
XBUBBL	05	Bubble memories
XA5050	06	Printer
XAT485	07	Teleprinter TC 485
XADISK	08	Disks (hard disk and floppy disk)
XAREAD	09	Paper tape reader
XAPUNC	OA	Paper tape punch
XATEAC	OB	Magnetic tape cartridge TEAC
XAMEPx	OC	Permanent memory
XAGRAF	OD	Graphics display screen
XAPROM	OE	EPROM programmer ELIND RP600
XELANP	OE	EPROM programmer ELAN C41

Table 5.5. - Symbolic Names of Peripherals

PERIPHERAL	SYMBOLIC NAME
Console	---
Bubble Memories	/MBx
Printer	/LPO
Teleprinter TC 485	/TYO or /TY1
Disks (hard disk or floppy disk)	/HDO or /FDO
Paper tape reader	/PRO
Paper tape punch	/PPO
Magnetic tape cartridge	/CTO
Permanent memory	/MPx
Graphics display screen	---
EPROM programmer	/UPO

Examples:

1) XATEAC,OB,1,5

This record declares the driver to handle the magnetic tape cartridge. The tape cartridge is connected via a serial line to the display board connector and is set for a transmission speed of 4800 bauds.

2) XAGRAF, OD

This record declares the driver to handle the graphics display screen.

3) XAMEP3,OC,3,4000,0060

This record declares the driver to handle the area of battery backed up CMOS permanent memory MP3. The 12 KB memory area is allocated at physical address 4000:0.

SECTION 2 RECORDS:

RECORD XACONS

This record contains the configuration parameters of the console driver.

XACONS,O4

RECORD XBUBBL

This record contains the configuration parameters of the bubble memory driver.

XBUBBL,05

RECORD XA5050

This record contains the configuration parameters of the printer driver.

XA5050,06,connect,baud

Where:

connect = 0 = CPU board connector for the peripheral

baud Identification number of the serial line baud rate: from 0 to 6. Table 5.6. indicates the relation between number and baud rate.

RECORD XAT485

This record contains the configuration parameters of the TC 485 teleprinter driver.

XAT485,07,connect,baud

---

where:

connect = 0 = CPU board connector for the peripheral

baud Identification number of the serial line baud rate: from 0 to 6. Table 5.6. indicates the relation between number and baud rate.

RECORD XADISK

This record contains the configuration parameters of the disk unit driver.

XADISK,O8

RECORD XAREAD

This record contains the configuration parameters of the paper tape reader driver.

XAREAD,09,connect,baud

Where:

connect = 0 = CPU board connector for the peripheral

baud Identification number of the serial line baud rate: from 0 to 6. Table 5.6. indicates the relation between number and baud rate.

RECORD XAPUNC

This record contains the configuration parameters of the paper tape punch driver.

XAPUNC,OA,connect,baud

Where:

connect = 0 = CPU board connector for the peripheral

baud Identification number of the serial line baud rate: from 0 to 6. Table 5.6. indicates the relation between number and baud rate.

RECORD XATEAC

This record contains the configuration parameters of the magnetic tape cartridge TEAC driver.

XATEAC,OB,connect,baud

Where:

connect = 0 = CPU board connector for the peripheral

baud Identification number of the serial line baud rate: from 0 to 6. Table 5.6. indicates the relation between number and baud rate.

RECORD XAMEPx

This record contains the configuration parameters of the permanent memory driver.

XAMEPx,OC,dev-num,init-add,mem-len

Where:

x Identification number of the permanent memory area: from 1 to 3.

dev-num Identification number of the permanent memory area: from 1 to 3.

init-add Start address of the permanents memorys area. The address must be given in hexadecimal, with four digits and zero offset.

mem-len Length of the permanent memory area expressed in sectors. Each sector contains 128 bytes. The length must be given in hexadecimal, with four digits (max. value 0400).

RECORD XAGRAF

This record contains the configuration parameters of the display graphic partition driver.

XAGRAF,OD

RECORD XAPROM

This record contains the configuration parameters of the driver for EPROM programmer ELIND RP600.

XAPROM,OE,connect,baud

Where:

connect = 0 = CPU board connector for the peripheral

baud Identification number of the serial line baud rate: from 0 to 6. Table 5.6. indicates the relation between number and baud rate.

RECORD XELANP

This record contains the configuration parameters of the driver for EPROM programmer ELAN C41.

XELANP,OE,connect,baud

Where:

connect = 0 = CPU board connector for the peripheral

baud Identification number of the serial line baud rate: from 0 to 6. Table 5.6. indicates the relation between number and baud rate.

Table 5.6. - Serial Lines Baud Rate

BAUD RATE NUMBER	BAUD RATE VALUE
0	110
1	300
2	600
3	1200
4	2400
5	4800
6	9600

The peripherals with numbers between 00 and 03 are defined at system level and cannot be configured in this FCRSYS section. These peripherals are indicated in the following table:

Table 5.7. - System Peripherals

PERIPHERAL	LOGIC NUMBER	NOTES
Keyboard 1	00	Foreground partition
Keyboard 2	01	Background partition
Display 1	02	Foreground partition
Display 2	03	Background partition

5.6.3. SECTION 3

This section is used to characterize the process software modules which must be processed on system start up.

If this section is missing, the system displays the message FILMS1_25.

If syntax errors occur in this section, the system displays the message FILMS1_26.

If one of the files is not present in EPROM, the system displays the message FILMS1_28.

Records Structure:

program name

where:

program name — Logic name of a process software module. This name is composed of 6 alphanumeric characters. If the software module resides on EPROM, the first character must be the letter "X".

The file declared in the first record of the section is considered as "main". The "main" type file is the only module to which the monitor passes control at the end of the read of file FCRSYS. The name of this file must contain the expression "MAIN" following the character "X", to obtain the sequence of five characters XMAIN. The sixth character making up the file name is selected by the user.

On power up, the monitor performs the following operations:

1. It checks to see if other records exist, in addition to the first, which declare a "main" type file.

2. If no other "main" type files are declared, it checks the existence of the remaining records.

3. It analyses the first record containing the name of the "main" type file and, after reading FCRSYS, it passes control to the "main" type file for initialization.

Example:

```
*3
XMAINP
XPROTO
LOGAL/MP2
XFILMS
*4
```

This section of FCRSYS declares a process software made up of 4 programs. Of these programs, XMAINP is the "main" type program. After FCRSYS has been bootstrapped, the monitor passes control to program XMAINP.

5.6.4. SECTION 4

This section is used to configure the system default peripherals and to define the tape punch code.

Records Structure:

symbol-name

where:

symbol-name	Symbolic name of a peripheral or of the code used for the tape punch.

If this section is missing from FCRSYS, the system assumes the following default peripherals:

- Record 1: /MP1
- Record 2: /DY0
- Record 3: ASCII

SECTION 4 RECORDS:

RECORD 1

This record defines the default peripheral in input operations.

/dev-name

Where:

dev-name	Name of the default peripheral in input operations. The following names are allowed: - MPx (x = 0-3): permanent memory, units 0-3; - MBx (x = 0-7): bubble memory, units 0-6; - HD0: hard disk; - FD0: floppy disk.

Examples:

/MP1

This record defines the permanent memory MP1 as default device in input operations.

RECORD 2

This record defines the default peripheral in output operations.

/dev-name

Where:

dev-name	Name of the default peripheral in output operations. The following names are allowed: DY0: alphanumeric display partition; LP0: printer.

Examples:

/DY0

This record defines the alphanumeric display partition as default peripheral in output operations.

/LP0

This record defines the printer as default peripheral in output operations.

RECORD 3

This record defines the code used for the tape punch.

code

Where:

code	Tape punch code. The following codes are available: ISO EIA ASCII

Examples:

ISO

This record defines the standard ISO code for the tape punch.

ASCII

This record defines the standard ASCII code for the tape punch.

5.6.5. SECTION 5

This section defines the JCL (Job Control Language) command codes which can be used in the JOB partition. Each record of the section defines a specific command code according to its position. The order of the records is fixed and cannot be modified.

Table 5.8. shows the relation in position between the record number and the command code which is defined. The symbolic command codes can be defined in the required language.

As the relation between the record number and command code is according to position, no record can be omitted.

The JCL command codes are made up of an alphanumeric string of 3 characters.

Records Structure:

comm-code

where:

comm-code	JCL command code corresponding to the record entered. Any alphanumeric string of three characters is allowed.

If section 5 is not inserted in FCRSYS, the system assumes the default codes listed in table 5.8.

NOTE:

Table 5.8. - JCL Commands and Records

RECORD	DEFAULT CODE	COMMAND DESCRIPTION
1	RUN	Running program in EPROM
2	LTI	Insert logic name in logic names table
3	DIF	Files differences
4	CRE	Create file with fixed length record
5	COP	Copy file
6	ASS	Assign logic name to data set
7	DAS	Deassign logic name
8	RIS	- Reserved -
9	LTD	Delete logic name from logic names table
10	VLT	Display logic names table
11	RIS	- Reserved -
12	RIS	- Reserved -
13	MOU	Install driver
14	EDI	Editor
15	DIS	Display file
16	RIS	- Reserved -
17	RIS	- Reserved -
18	DIR	Display list of files
19	REN	Rename file
20	DEL	Delete file
21	INI	Initialize device
22	ATT	Assign protections to file
23	FOR	Create formatted file

5.6.6. SECTION 6

This section defines the directive codes and the editor environment messages. Each record of the section defines the position of a specific directive code or a particular message. The order of the records is fixed and cannot be modified.

Table 5.10. shows the relation in position between the record number and the directive code or the message which is defined. The symbolic directive codes can be defined in the required language.

As the relation between the record number and directive code is according to position, no record can be omitted.

By means of the records of this section it is possible to define the symbolic editor directive codes and the system messages in the required language.

If section 6 is not inserted in FCRSYS, the system assumes the directives and default messages listed in table 5.9.

Table 5.9. - Default Editor Directives and Messages

RECORD	DIRECTIVES AND MESSAGES	DESCRIPTION
1	RIM	"REPLACE" directive
1	INS	"INSERT" directive
1	CAN	"CANCEL" directive
2	EOF	"END OF FILE" message
2	BOF	"BEGINNING OF FILE" message
2	NEW	"NEW FILE" message
2	OLD	"OLD FILE" message
3	COMAND ERROR	message
4	MEMORY OVERFLOW	message
5	RECORD OVERFLOW	message
6	WRONG KEY	message
7	KEY OVERFLOW	message
8	WRONG DIRECTION	message
9	FORMAT ERROR	message
10	I/O ERROR	message

SECTION 6 RECORDS:

RECORD 1

This record defines the three-letter codes of the editor directives.

---

repl-code,ins-code,of the-code

Where:

repl-code — Three-letter code of the "REPLACE" directive. Any string of three letters is allowed.

ins-code — Three-letter code of the "INSERT" directive. Any string of three letters is allowed.

del-code — Three-letter code of the "DELETE" directive. Any string of three letters is allowed.

Characteristics:

The relation between code and directive within the record is fixed and positional: each code is associated with a specific directive according to the position it occupies within the record.

We recommend to declare the following record 1:

RIM,INS,CAN

This record defines the following editor directives:

- RIM = directive code "REPLACE";

- INS = directive code "INSERT";

- CAN = directive code "DELETE".

RECORD 2

This record defines the three-letter messages regarding the status and type of file.

---

end-code,beg-code,new-code,old-code

Where:

end-code — Three-letter code identifying the message with which the system communicates the end of a file.

beg-code — Three-letter code identifying the message with which the system communicates the beginning of a file.

new-code	Three-letter code identifying the message with which the system communicates the opening of a new file.
old-code	Three-letter code identifying the message with which the system communicates that the file already exists.

Characteristics:

The relation between code and message within the record is fixed and positional: each code is associated with a specific message according to the position it occupies within the record.

We recommend to declare the following record 2:

EOF,BOF,NEW,OLD

This record defines the following editor messages:

- EOF: communicates the end of a file;

- BOF: communicates the start of a file;

- NEW: communicates that a new file has been opened;

- OLD: communicates that an old file, i.e. a file that already exists, has been invoked.

RECORD 3

This record defines the message used by the system to communicate an error in the introduction of an editor directive.

ASCII-string

Where:

ASCII-string	ASCII string, composed of a maximum 16 characters, which contains the message with which the system communicates the following errors: - the syntax of the directive is not correct; - the directive does not exist.

We recommend to declare the following editor message:

COMMAND ERROR

This record defines the message "COMMAND ERROR". The system displays this message when an error in introducing an editor directive occurs.

RECORD 4

This record defines the message used by the system to communicate that the memory is insufficient to contain the current file in the editor phase.

ASCII-string

Where:

ASCII-string	ASCII string, composed of a maximum of 16 characters, containing the system message.

We recommend to declare the following editor message:

MEMORY OVERFLOW

This record defines the message "MEMORY OVERFLOW". The system displays this message when the memory is not large enough to contain the current file.

RECORD 5

This record defines the message used by the system to communicate that the record introduced exceeds the maximum length allowed for the current file. The message is only valid for files generated with the command CRE.

ASCII-string

Where:

ASCII-string	ASCII string, composed of a maximum of 16 characters, containing the system message.

Observations:

It should be remembered that the command CRE generates files with fixed length records.

We recommend to declare the following editor message:

RECORD OVERFLOW

This record defines the message "RECORD OVERFLOW". The system displays this message when an overflow error on a record is verified.

RECORD 6

This record defines the message used by the system to communicate an error in introducing a search key.

ASCII-string

Where:

ASCII-string	ASCII string, composed of a maximum of 16 characters, containing the message with which the system communicates the following errors: - Search sequence number. The number of records searched is not of the BCD type; - Search for key. The key pressed is not correct.

Observations:

For further details, consult the manuals: "NC-110 MC, Use and programming manual", "NC-110 TC, Use and programming manual" according to the type of NC-110 system used.

We recommend to declare the following editor message:

WRONG KEY

This record defines the message "WRONG KEY". The system displays this message when an error in introducing a search key is verified.

RECORD 7

This record defines the message used by the system to communicate that the sequence number searched for exceeds the handling capacity of the system.

ASCII-string

Where:

ASCII-string	ASCII string, composed of a maximum of 16 characters, containing the system message.

Observations:

Note that a file can contain up to 64,000 records.

We recommend to declare the following editor message:

KEY OVERFLOW

This record defines the message "KEY OVERFLOW". The system displays this message when an overflow error in the search for a record sequence is verified.

RECORD 8

This record defines the message used by the system to communicate that the direction command selected (**LINE SKIP** or **LINE BACK**) is not consistent with the current position of the pointer within the file.

ASCII-string

Where:

ASCII-string	ASCII string, composed of a maximum of 16 characters, containing the system message.

We recommend to declare the following editor message:

WRONG DIRECTION

This record defines the message "WRONG DIRECTION". The system displays this message when a direction error in the positioning of the pointer is verified.

RECORD 9

This record defines the message used by the system to communicate that the record format introduced (in the case of a formatted file) is not correct.

ASCII-string

Where:

ASCII-string	ASCII string, composed of a maximum of 16 characters, containing the system message.

We recommend to declare the following editor message:

FORMAT ERROR

This record defines the message "FORMAT ERROR". The system displays this message when a format error in a record is verified.

RECORD 10

This record defines the message used by the system to communicate that an I/O error has been verified.

ASCII-string

Where:

ASCII-string	ASCII string, composed of a maximum of 16 characters, containing the system message.

We recommend to declare the following editor message:

I/O ERROR

This record defines the message "I/O ERROR". The system displays this message when an I/O error in the JOB partition is verified.

5.6.7. SECTION 7

This section defines the permanent logic names of the data sets relative to system messages and to the configuration of he process software. The system can handle a maximum of 15 permanent logic names which cannot be modified by the user. Table 5.10. lists the logic names with the corresponding data sets.

A logic name is made up of an alphanumeric string of 6 characters, the first of which must be alphabetic.

The logic name of a data set is the name used by the system to address the data set regardless of data set allocation modes.

The purpose of the logic name is to make the code written to handle some machine functions independent of external variables (for example: hardware configuration, language selected for user/system communications, etc).

Records Structure:

log-name,file-name/device

where:

log-name	Logic name used by the system. The name is composed of 6 alphanumeric characters, the first of which is alphabetic
file-name	Name of the data set associated with the logic name. The name is composed of 6 alphanumeric characters, the first of which is alphabetic
device	Name of the peripheral on which the data set is stored.

Table 5.10. - Logic Names and Data Sets

LOGIC NAME	DATA SET
FILMS1	File EDPERR - JCL and FCRSYS errors
FIIMS2	File EDPMSG - JOB messages
FILMS3	File IOERRM - I/O errors in JOB
FIIMS4	File PROERR - process errors
FILMS5	File MESSAG - machine logic messages
FORMAT	File FORMAT - record formats of the formatted files
AXCONF	File AXCFIL - characterization
IOCONF	File IOCFIL - characterization
PGCONF	File PGCFIL - characterization
FILCMD	File PROTEC - part program names that can be enabled from K buffer

SECTION 7 RECORDS:

RECORD FILMS 1

This record defines the logic name of the file EDPERR which contains the EDP error messages detected in the JOB partition and during the FCRSYS diagnostics.

FILMS1, EDPERR/MP3

The messages must be stored in file EDPERR previously generated by the command **CRE** (see Section "Creation procedure" of this chapter). These messages can be introduced in file EDPERR in the following ways:

- Via editor, by introducing them individually on the keyboard;

- By copying messages which already exist on a specific memory device.

If the record FILMS1 is not declared, or if the file EDPERR associated with this record has not been generated, the system displays the following message in code:

FILMS1 nn

where nn is the code of the error message. The EDP error messages of the JOB partition and of file FCRSYS are listed in Appendix A of this manual.

RECORD FILMS2

This record defines the logic name of the file EDPMSG which contains the EDP messages of the JOB partition.

FILMS2, EDPMSG/MP3

The messages must be stored in file EDPMSG previously generated by the command CRE (see Section "Creation procedure" of this chapter). These messages can be introduced in file EDPMSG in the following ways:

- Via editor, by introducing them individually on the keyboard;

- By copying messages which already exist on a specific memory device.

If the record FILMS2 is not declared, or if the file EDPERR associated with this record has not been generated, the system displays the following message in code:

FILMS2 nn

where nn is the code of the EDP message. The EDP messages corresponding to these messages in code are listed in Appendix A of this manual.

RECORD FILMS3

This record defines the logic name of the file IOERRM which contains the error messages relative to I/O operations in the JOB partition.

FILMS3,IOERRM/MP3

The messages must be stored in file IOERRM previously generated by the command **CRE** (see Section "Creation procedure" of this chapter). These messages can be introduced in file IOERRM in the following ways:

- Via editor, by introducing them individually on the keyboard;

- By copying messages which already exist on a specific memory device.

If the record FILMS3 is not declared, or if the file EDPERR associated with this record has not been generated, the system displays the following message in code:

FILMS3 nn

where nn is the code of the I/O error message. The I/O error messages corresponding to these messages in code are listed in Appendix A of this manual.

RECORD FILMS4

This record defines the logic name of the file PROERR which contains the process error messages.

FILMS4,PROERR/MP3

The messages must be stored in file PROERR previously generated by the command **CRE** (see Section "Creation procedure" of this chapter). These messages can be introduced in file PROERR in the following ways:

- Via editor, by introducing them individually on the keyboard;

- By copying messages which already exist on a specific memory device.

If the record FILMS4 is not declared, or if the file EDPERR associated with this record has not been generated, the system displays the following message in code:

FILMS4 nn

where nn is the code of the error message. The error messages of the process corresponding to these messages in code are described in the following manuals: "NC-110 MC, Use and programming manual", "NC-110 TC, Use and programming manual".

RECORD FILMS5

This record defines the logic name of the file MESSAG which contains the messages used by the machine logic.

FILMS5,MESSAG/MP3

The messages must be stored in file MESSAG previously generated by the command **CRE** (see Section "Creation procedure" of this chapter). These messages can be introduced in file MESSAG in the following ways:

- Via editor, by introducing them individually on the keyboard;

- By copying messages which already exist on a specific memory device.

If the record FILMS5 is not declared, or if the file EDPERR associated with this record has not been generated, the system displays the following message in code:

FILMS5 nn

where nn is the code of the machine logic message. The machine logic messages corresponding to these codes are described in the Characterization Documents or in the Machine Logic Documentation.

RECORD FORMAT

This record defines the logic name of the file FORMAT.

FORMAT,FORMAT/MP3

The file FORMAT is composed of a single record of 40 characters containing the format of the single records of the formatted files. These files must have been generated previously by directive EDI.

Formatted files are files made up of records with prefixed formats. The format of the records is declared via the file FORMAT. Note that in this case the logic name coincideses with the name of the corresponding data set. For further details, see Chapter 7, Section "Generation of Formatted Files".

The formatted files are:

- origin files;

- corrector files;

- tool monitoring files.

The instructions for writing the FORMAT file and loading the origin, offset and tool monitoring files are described in manuals: "NC-110 MC, Use and Programming Manual" and "NC-110 TC, Use and Programming Manual".

RECORD AXCONF

This record defines the logic name of the characterization file AXCFIL.

AXCONF,AXCFIL/MP3

File AXCFIL contains the characterization parameters of the axis manager. See Chapter 6 "Axis manager characterization".

RECORD IOCONF

This record defines the logic name of the characterization file IOCFIL.

IOCONF,IOCFIL/MP3

File IOCFIL contains the characterization parameters of the machine logic software. See Chapter 8 "Machine logic characterization".

RECORD PGCONF

This record defines the logic name of the characterization file PGCFIL.

PGCONF,PGFIL/MP3

File PGCFIL contains the characterization parameters of the technological process currently working on the machine tool. See Chapter 7 "Technological process characterization".

RECORD FILCMD

This record defines the logic name of the file PROTEC which contains the names of the part programs which can be selected by the machine logic.

FILCMD,PROTEC/MP3

File PROTEC has records of varying lengths without a preallocated area (as a common part program).

5.6.8. SECTION 8

This section defines the names of the executable files resident on EPROM which the system can process. The system can handle a maximum of 10 executable files, thus section 8 is made up of 10 sections, each of which defines the name of an executable file.

Records Structure:

log-name

where:

log-name	Logic name of an executable file. The name is composed of an alphanumeric string with a maximum of 6 characters. The first character must be alphabetic.

Example:

```
*8
SIPROM
DEBUG
PIMP
CKSUM
BACKUP
*
```

6. AXIS MANAGER CHARACTERIZATION

6.1. AXIS MANAGER

The axis manager is a process software module which moves the axes according to the commands sent by the user programs (part programs). The movement of the axes is divided into two parts:

- Axis interpolation;

- Axis servo control.

Interpolation and servo control of the axes are the main activities on which numeric control is based. These two activities are performed by two specific interpolation and servo control routines, which communicate by means of a data exchange buffer.

Interpolation

Interpolation consists in calculating, on the basis of the programmed parameters, the coordinated movement of several axes so as to obtain a movement which, basically, may be of the following three types:

- Linear;

- Circular;

- Helical.

Servo Control

Servo control consists of all the activities which make sure that the axes do in fact follow the exact profile calculated by the interpolator.

The position and speed of the interpolated axes, also known as continuous axes, are continually controlled by the system, to make sure that the calculated path is followed. These axes differ from the point-to-point axes in that with the latter there is no guarantee that the path will be followed, just that the final point will be reached.

Interpolation and servo control of the axes are governed by the following factors:

- Each interpolator may handle up to 3 interpolated axes plus the spindle axis;

- The interpolated axes must have the same interpolator;

- The spindle axis must have the same interpolator as the continuous axes;

- The point-to-point axes to be moved in parallel must have different interpolators.

6.2. AXIS FEATURES

The axes controlled by the axis manager may be divided into various types according to the specific function they perform on the machine tool. The methods of controlling the axes differ according to the various types. In this section of the chapter, we shall discuss all the types of axis the axis manager can control.

6.2.1. TYPES OF AXIS

Section 2 of file AXCFIL indicates the type of axis to be configured. These types are listed and described below.

Coordinated Axis

This is a working axis which may be interpolated with other axes of the same type.

Point-to-point Axis

This axis is not interpolated with others, it is used for positioning from one point to another. This axis may be programmed as an indexing axis, as a tool magazine or more generally as a mechanical component fitted with a positioning transducer. It may be activated using either an analogic command or the hydraulic or electromagnetic ON/OFF command.

Rotary Axis

A rotary coordinated axis, which is programmed in degrees.

Switched Axis

A coordinated axis which is not coupled until it is programmed as an alternative to another axis. The switched axis is declared on the dyad of axes which must be switched. The name of the other axis on which switching takes place must be declared in the TPA record of each of the two axes.

Spindle Axis Without Transducer

The speed of this axis is controlled by means of function S. Its position cannot be controlled by the numeric control system.

A typical application of this type of axis consists in the "motorized" tool. This axis is caused to rotate by programming the 3-letter code: USS, spindle name ± revs. Remember that the "motorized" tool is declared in AXCFIL as a commnon spindle axis without a transducer.

Spindle Axis with Transducer

The speed of this spindle axis is controlled by function S and its angular position by machine logic. Its threading process may be entirely controlled by the system.

Diametrical Axis

This interpolated axis must be programmed and displayed with factor 2. A typical example of this is the X axis of a lathe or of a facing/boring head.

Axis with Set Point

A special interpolated axis for driving the spindle in coordinated motion, interpolated with the other axes. When it is to be activated by the logic, it uses the spindle transducer value as the set point. A typical example is the C axis in lathes.

Virtual Axis

This axis may be programmed even though the axis is not physically present on the machine. The motion is created by the combined movement of two other axes, defined during simulation. To defines a virtual axis remember that:

- the axis must be declared in AXCFIL with the same interpolator as the axes simulated;

- in the subsection of the axis, the following records must be declared: NAS, TPA, NTC.

Example: NAS=P; TPA=1O1; NTC=O,O

Absolute Axis

This axis does not require a home position microswitch because it only has one electrical zero for its entire travel.
E.g.: the optical lines or rotary transducers usesd within a revolution. To set this axis, search for thes marker (selector, home position microswitch search mode), and then the system moves automatically to the electrical zero.

Zero Cycle after Reactivation

This cycle is used for the axes which may be deactivated by the logic but when reactivated do not ensure correct transducer reading. In this case the bit on the corresponding K package is reset.

Spindle Axis with Tensioned Ramp

This feature provides an acceleration ramp which is controlled by the spindle rotation command. The spindle activator ramp must be eliminated to improve performance in the angular position.

Split Axis

The movement of this axis is determined by the movement of an axis known as the master axis with which it is used. In this way, there is a dyad of axes: a master axis and a slave axis. The split axis is the slave axis. For a split axis, the following records must be declared in file AXCFIL: NAS, TPA, NTC, GAS, MCZ, MFC, SKW.

The mechanical and electrical features of the split axis must be identical to those of the master axis, and, in record MCZ, the home position microswitch research speed and direction must be the same as those of the master axis. The mechanical pitches of the slave axis and the master axis may have a different sign. The difference between the two pitches is declared in record PAS.

Example: PAS=5000,1 (master axis); PAS=5000,-1 (slave axis)

Roll-Over Axis

This is used in a rotary interpolated axis when the position displayed in the 0-359,999 degree range is to be kept. This means that the entire movement of the axis is regulated by the position displayed in absolute and incremental programming.

Spindle without Servo Control

This axis must only be declared for spindles with an a.c. motor and no servo control. In this way the system, may set, in G84 and G86, the machine logic signals to request the inversion or interruption of the spindle at the bottom of the hole.

Remember that to control the a.c. spindle motor, the same criteria as for a d.c. spindle motor, with or without a transducer, must be used both for characterization and the machine logic signals.

6.2.2. DIGITAL/DIGITAL ELECTRONIC WHEEL HANDLING

The digital/digital wheel is controlled like any other axis and may be configured both in a single-process or multi-process environment.

In a multi-process environment it is possible to configure:
- a wheel for each process controlled by the system;
- a wheel common to all the processes controlled by the system.

In both cases, the calibrations indicated in the following example must be made in files AXCFIL and IOCFIL for each process.

Example:
In section 1 of file AXCFIL the record CAS must be declared as follows:

```
CAS=1,XZAS,1O     (A = name of wheel)
```

After the subsections for axes X, Y and Z, enter the following subsection for the electronic wheel:

```
NAS=A
TPA=1,
NTC=n,O           (n = 5 - number of encoder transducer)
PAS=400,1
SRV=0,0,0
GAS=O,0
POS=O,O
```

Records SRV, GAS, and POS are only required for problems concerning the checks made by the characterization interpreter.

In section 3 of file IOCFIL, record ADV must be declared as follows:

```
ADV=A
```

6.2.3. DIGITAL/ANALOG ELECTRONIC WHEEL HANDLING

The digital/analog wheel features an encoder transducer and is connected to the NC-110 analog input by means of the M2250 interface.

To characterize th d/a wheel, declare record ADV in section 3 of file IOCFIL as follows:

```
ADV=3
```

6.3. FILE AXCFIL

6.3.1. FEATURES

The axis manager is characterized using file AXCFIL. This file consists of three sections numbered from 1 to 3. Each section characterizes a particular component or group of components of the axis manager.

The file is installed in the system in editor mode, using the following command:

EDI,AXCFIL/MPO

All the sections contain the same number of subsections as there are declared processes, so certain records are repeated, within the section, for all the processes declared.

The general structure of the file is described in the "File Structure" section of Chapter 2. The general structure of the AXCFIL file records is described in the "Process software characterization" section of Chapter 2.

6.3.2. SECTIONS

Section 1 contains the configuring parameters for the CPU and the processes controlled by the NC-110 system.

Section 2 contains the configuring parameters for the axes. This section, within the subsections for the processes declared, contains a subsection for each type of axis declared.

Section 3 is used for setting the parameters for the correction of geometrical errors.

6.3.3. SECTION 1

This section configures the system CPU and the axis manager. The processes controlled by the system are declared.

This section consists of records: NBP, TIM, PRO, COM , INx, and CAS.

Records PRO, INx, and CAS must be repeated the same number of times as there are processes declared.

RECORDS OF SECTION 1:

NBP

This record is used for declaring the number of processes controlled by the system and establishing the limits to the user memory.

NBP=proc-num,start-add

Where:

proc-num	Number of processes controlled by the system. These processes are subsequently configured in files PGCFIL and IOCFIL. A maximum of 5 processes (from one to five) may be declared.
start-add	Initial user memory address. During characterization the conventional hexadecimal address 7FFF may be declared. Having completed characterization, the address indicated by the system message must be declared. See Chapter 5, Section "MP3 memory area".

Format: word, hexadecimal (four figures)

Characteristics:

The initial user memory address protects the memory. It makes sure that subsequent modifications made to the characterization files do not cancell the contents of the user memory.

Observations:

The initial user memory address must also be declared in record XAMEP of file FCRSYS (See the section "Section 2" of Chapter 5).

Example:

If the system displays the following message at the end of the characterization bootstrap:

WARNING: XAMEP1,OC,1,3AFO,OO

record NBP is written in the following way:

NBP=2,3AFF

TIM

This record is used for defining clock pulses of the CPU of the NC-110 system and the position of the respective board in the package.

TIM=cpu1-pulse,cpu2-pulse,cpu3-pulse,cpu4-pulse,cpu5-pulse

Where:

cpui-pulse	It is the clock pulse, expressed in ms, of the system and axis manager. The order in which the check pulses within the record are declared corresponds to the order in which the CPU board are positioned in the system board package.

Format: word, word, word, word, word

Observations:

The zero value for the parameter cpui-pulse indicates that there is no CPU present in the corresponding position in the board package.

Example:

A system CPU board in position 1.

The following record must be declared:

TIM=1O,0,0,0,0

PRO

This record is used for setting the number of the current process selected for characterization.

PRO=proc-num

Where:

proc-num	Number of the process required for characterization: from 1 to 5. This number must be the same as or smaller than the number of processes declared in record NBP of AXCFIL.

Format: integer

Observations:

Record PRO must be repeated for each of the processes declared.

COM

This record is for declaring the spindle axis of the current process also used in another specified process.

COM=subs-num,axis-name

Where:

subs-num	Number of the specified process in which the spindle axis is used.
axis-name	Name of the spindle axis used in the process specified in parameter subs-num. The name S is usually declared.

Format: word, ASCII string

Observations:

The spindle axes common to another processe are not to be declared further in record CAS of the current process. Do not declare the subsection of the common axis used for the process in which record COM is declared.

N.B.: The interpolators which control the spindle axis common to the processes must have the same clock pulse.

INx

This record contains the parameters which configure the interpolator.

INx=cpu-num,axis-name,spind-name,int-pulse,block-num

Where:

x	Alphanumeric character indicating the number of the interpolator. This character is also declared in section 5 of PGCFIL; in record NIP for the process in which the specified interpolator is used.
cpu-num	Position of the CPU in the board package containing the interpolator of the axes declared must be equal 1.
axis-name	List of the names of all the continuous and point-to-point axes which have a transducer. A slave axis configured as a split axis must not be declared. The names A, B, C, D, X, Y, Z, U, V, W, P, or Q may be used for the interpolated axes. All the letters of the alphabet may be used for the point-to-point axes.

spind-name	Name of the spindle axis. The name is only declared if the spindle axis has a transducer.
int-pulse	Interpolation clock pulse. This time, expressed in ms, must be equal to or a multiple of the servo control clock pulse and the CPU clock pulse on which the interpolator resides (record TIM).
block-num	Number of blocks processed, during the movement along the profile, by the precalculation for the interpolator. It may vary from 1 to 64. If a low number is declared, the machine may be blocked on the profile due to the fact that the processed points are too close together. If an excessively high number is declared the MP1 memory is reduced. Each block occupies 512 bytes. Always declare 1 for the interpolators of the point-to-point axes; for the interpolator of the coordinated axes, declare a minimum of 2.

Format: word, ASCII string, ASCII string, word, word

Example:

IN1=1,XYZ ,S,10,16

This record is for establishing the following sequence of axes:

X = first axis; Y = second axis; Z = third axis;
axis.

CAS

This record contains the parameters which configure the axis servo control.

CAS=cpu-num,axis-name,servo-pulse

Where:

cpu-num	Position of the axis manager CPU (containing the servo control of the axes) declared in the board package starting from the right. The system CPU must always be in position 1.

axis-name	Names of the axes controlled by the axis manager. Every component which has a transducer and/or requires an analogical reference from the control system is considered an axis. The names A, B, C, D, X, Y, Z, U, V, W, P, Q may be used for the interpolated axes. For the point-to-point axes all the letters in the alphabet may be used. S is used for the spindle axis. The names must be declared in the same order as the interpolators.
servo-pulse	Servo control clock pulse. It is expressed in milliseconds. It must be equal to or a submultiple of the interpolator clock pulse and equal to or a multiple of the servo control CPU clock pulse.

Format: word, ASCII string, word

Observations:

The order in which the axes are declared in the parameter axis-name, determines the order in which the subsections for the axes should be created in section 2 of AXCFIL.

Example: CAS=1,XYZS,1O

This record is for declaring the presence of the following components:

- CPU board in position 1;

- Interpolated axes: X, Y, Z;

- Spindle axis: S;

- Servo control clock pulse: 10 msec.

6.3.4. SECTION 2

Section 2 configures the axes controlled by the axis manager. This section is specific for each process, so the records it contains must be declared for each process declared in record PR0. Each axis declared in record CAS of section 1 must also be declared in the same order in record NAS of section 2.

Subsections for the Axes Declared

For each axis declared in record NAS, a subsection containing the records listed below must be defined after record NAS. The records must be entered in the same order in which they are listed.

Records containing the parameters of the axis:

TPA
NTC
RAP

After these three records, if the transducer for the declared axis is present, the following records must be entered:

PAS
GAS
SKW
POS
SRV

Finally a series of records subjected to various conditions may be entered:

MCZ
MAN
GMx
LOP
MFC
TSM
ASM
POM
ADP
ZNO
FBF

Measuring Units

The units for measuring the movement of the axes are set in record PAS. This record enables the transducer reading to be related to the physical dimensions of the movement of a specific type of axis. The relationship between a transducer revolution and the following movements is determined:

- The distance covered by the linear axes;

- The angle performed by the rotary axes;

- The stations covered by the point-to-point axes;

- The rotation of the spindle axes.

The measuring units declared in record PAS must also be used in the other records which use the quantity to which these units refer. It is advisable to use the following measuring units:

- Millimeters for the movement of the linear axes;

- Degrees for the rotations of the rotary axes;

- Stations for the movement of the point-to-point axes;

- Revolutions per minute for the rotation of the spindle axes.

RECORD OF SECTION 2:

PRO

This record is for setting the number of the current process selected for characterization.

PRO=proc-num

Where:

proc-num	Number of process to be selected for characterization: from 1 to 5. This number must be smaller than or equal to the number of processes declared in record NBP of AXCFIL.

Format: integer

Observations:

Record PRO must be repeated for each of the processes declared.

NAS

This record is used for defining the current axis selected for characterization.

NAS=axis-name

Where:

axis-name	Name of the axis to be selected for characterization. It is one of the names declared in records INx and CAS of section 1.

Format: ASCII string

Characteristics:

The name of the axis must be declared in the order indicated in record CAS of section 1. This record must be declared for each axis controlled by the axis manager.

TPA

This record is used for defining the type of current axis selected for characterization.

TPA=axis-type, switch-axis-name

Where:

axis-type	Hexadecimal code for the type of current axis. The hexadecimal code and corresponding type of axis may be found in the word configuration indicated in Table 6.4.
switch-axis	Name of the axis on which the curren axis is switched. If the axis is not switched write a comma only.

Format: hexadecimal (four figures), ASCII string

Table 6.4. - Hexadecimal Codes for Types of Axes

HEXADECIMAL CODE	BIT	MEANING
0001	0	1 = Coordinated axis
0002	1	1 = Point-to-point axis
0004	2	1 = Rotary axis
0008	3	1 = Switched axis
0010	4	1 = Spindle axis without transducer
0020	5	1 = Spindle axis with transducer
0040	6	1 = Diametrical axis
0080	7	1 = Axis with set point
0100	8	1 = Virtual axis
0200	9	1 = Absolute axis
0400	10	1 = Zero cycle after reactivation
0800	11	1 = Spindle axis with tension ramp
1000	12	1 = Split axis
2000	13	1 = Roll-over axis
4000	14	1 = Spindle without servo control

This table shows the hexadecimal value and corresponding 1 bit. Various combinations of bits may be used to select several types of axis-selectors or buttons. In this case the hexadecimal value to be declared is obtained by adding together the codes of the corresponding 1 bits.

Characteristics:

Various combinations of bits are possible. In this case the hexadecimal values corresponding to the bits must be summed up to obtain the hexadecimal value to be entered.

Example:

To declare a diametrical coordinated axis 41 must be keyed in. To declare a switched rotary axis OC must be keyed in.

NTC

This record is used for defining the type and number of the transducer and converter for the current axis.

NTC = trans, conv

Where:

trans — Type of board and number of the transducer.
The type of board is indicated using the following symbols:
+ = boards M2230, M2240
- = board M2260
The number of the transducer is calculated using the expression indicated in the characteristics. Type and number must be indicated in order, one after another.
If two boards of type M2230 and/or M2240 are in the rack, they must be indicated the following simbols:
1-st board (left) - 1,2,3,4
2-nd board (right) - 5,6,7,8
If two boards of type M2260 are in the rack, they must be indicated the following simbols:
1-st board (left) - -1
2-nd board (right) - -2

conv — Type of board and number of the converter.
The system can to have the boards of the connverters: M2250, M2260. Every board has 4 cannels of connverters.
If two boards of type M2250 and/or M2260 are in the rack, they must be indicated the following simbols:
1-st board (left) - 1,2,3,4
2-nd board (right) - 5,6,7,8
If two boards of type M2260 are in the rack, they must be indicated the following simbols:
1-st board (left) - -1
2-nd board (right) - -2

Format: real,real

Characteristics:

The system can control boards M2250, M2240, M2230, M2260. The non-intelligent boards may be calibrated from 0 to 15 and the intelligent boards from 0 to 7.

The number of the transducer or converter depends on the hardware calibration of the board and the position of the transducer or converter on the board. Note that two boards may not be calibrated with the same number, even if they are of different types.

The intelligent boards have 4 input connectors for the transducers, whereas the non-intelligent boards have 3.

The intelligent boards have 8 input connectors for the converters, whereas the non-intelligent boards have 4.

Calculation of the number of the transducer

a) boards M2240, M2230 (intelligent)

N = (Np x 4) + Nc

b) board M2250, M2260 (non-intelligent)

N = (Np x 3) + Nc

where:

N Number of the transducer to be declared.

Np Number of the board declared in the hardware calibrations.

Nc Number of the input connector on the transducer board.

Calculation of the number of the converter:

board M2250 (non-intelligent)

N = (Np x 4) + Nc

where:

N Number of the converter to be declared.

Np Number of the board declared in the hardware calibrations.

Nc Number of the input connector on the converter board.

If both intelligent and non-intelligent boards are to be used, the following calibrations must be made.

- boards M2240, M2230: numbers 0,1,2,3,4,5,6,7

- board M2250 numbers 8,9,10,11,12,13,14,15

Example:

Table 6.5. shows an example of the characterization of transducers and converters on an intelligent board with hardware calibration 0 and a non-intelligent board with hardware calibration 8. The table shows the number of the transducer or converter and the type of board which controls them.

Table 6.5. - Characterization of transducers and converters

TRANSDUCERS		CONVERTERS	
NUMBER	**TYPE OF BOARD**	**NUMBER**	**TYPE OF BOARD**
1	intelligent	1	intelligent
2	intelligent	2	intelligent
3	intelligent	3	intelligent
4	intelligent	5	intelligent
-25	non-intelligent	6	intelligent
-26	non-intelligent	7	intelligent
-27	non-intelligent	8	intelligent
		-33	non-intelligent
		-34	non-intelligent
		-35	non-intelligent
		-36	non-intelligent

RAP

This record is used for setting the speed and acceleration parameters for the current axis.

RAP = fast-speed, fast-accel

Where:

fast-speed — Fast speed of axis expressed in millimeters per minute. This parameter must not be declared for a spindle axis.

fast-accel — Fast acceleration: Represents fast acceleration expressed in:
- mm/(s*s) for linear axes;
- degrees/(s*s) for rotary axes;
- stations/(s*s) for point-to-point axes.

Format: real, real

Characteristics:

Record RAP may be used to optimize the positioning of the spindle. This features only applies for large-scale spindles with transducers.

To obtain this feature record RAP must be entered in the subsection of the spindle axis using the second parameter only. In this specific case the second parameter of the record represents the acceleration of the spindle expressed in revs/(s*s).

The feature is optional, so if the record is omitted or if the second parameter is set to zero, the spindle is oriented by moving it to the orientation point using the position error only. If the acceleration parameter is given a value greater than zero, the deceleration on the orientation point is controlled. The acceleration is only controlled if the spindle is configured with a ramp (TPA=820).

The acceleration value entered is not controlled, but it is not advisable to exceed the maximum value represented by the following expression:

$$accel = \frac{speed\ range}{time \times 60} \times \frac{volt}{KC}$$

where:

accel	Acceleration value calculated.
speed range	Main range speed.
volt	Value of the parameter "max-rif-inver" of record TSM.
time	Value of the parameter "time max inver" of record TSM.
KC	KC value.

Example:

Subsection for a spindle axis:

```
NAS=5
TPA=820
...
GM1=3000,7.5,10
TSM=5,15
...
RAP=,20
```

Acceleration calculation:

$$accel = \frac{3000}{5 \times 60} \times \frac{15}{7.5} = 20\ revs/sec^2$$

GAS

This record is used for defining the current axis action.

GAS=axis-bckl,dead-zone

Where:

axis-bckl	Axis backlash. Value of the backlash when inverting the direction of motion of the axis when the position is measured indirectly (e.g. resolver on screw or motor). The value must be expressed in millimeters if the axis is linear; in degrees if the axis is rotary.
dead-zone	Dead zone. This represents the zone within the positioning tolerance in which the control loop is not closed. It is expressed in millimeters.

Format: real, real

Example:

GAS = 0.15,0.005

PAS

This record is used for setting the pitches of the transducer with respect to the current axis.

PAS=el-pitch,mec-pitch

Where:

el-pitch	Electrical pitch. The value to be used for this parameter depends on the type of transducer (encoder, resolver, inductive) and the software version. See characteristics. This parameter must be positive.
mec-pitch	Mechanical pitch. This represents the relationship between the distance covered by the axis and the number of revolutions performed by the respective transducer. If an inductive transducer is used the mechanical pitch corresponds to the number of pulses every two millimeters and may be positive or negative. If a point-to-point axis is used, the pitch corresponds to the number of stations covered in a transducer revolution.

Format: word, real

Characteristics:

If the transducer is an encoder the electrical pitch consists of the number of impulses per revolution provided by the encoder, multiplied by 4.

The maximum electrical pitch is:

. 32768 with an intelligent interface.

If the transducer is a resolver the electrical pitch consists of the number of impulses for every resolver revolution. If the transducer is inductive the electrical pitch consists of the number of impulses emitted every two millimeters that the inductive transducer moves.

For both transducers, the maximum value for the electrical pitch is 32768.

If an optical line transducer is used, any number may be chosen for the mechanical pitch, whereas the electrical pitch must correspond to the number of impulses provided by the optical line multiplied by four and by the mechanical pitch. The mechanical pitch used is normally 1 mm and the electrical pitch is the number of impulses per millimeter multiplied by four.

Example:

Axis with encoder transducer, 1250 impulses/rev and 1/1 ratio with 10 mm screw (every encoder rev = 10 mm):

PAS=5000,10

If the transducer is a resolver write:

PAS=32768,10

SKW

Define the master axis used with the current axis.

SKW=master-axis,skew,kds,gain

Where:

master-axis	Name of master axis to be used with the current split axis so as to form the master/slave dyad of axes. The name of the axis must consist of one letter only and the subsection of the master axis must be declared before that of the slave axis.
skew	This determines the maximum skew accepted between the two axes. The value to be declared must be expressed in millimeters or fractions of millimeters for linerar axes, in degrees for rotary axes.

kds	This specifies the skew between the markers of the transducers of the two axes. The value to be declared must be expressed in transducer impulses. Remember that this record is only activated if a split axis is declared in record TPA of section 2.
gain	Gain constant for the recovery of the skew error. The recommended calibration range is form 0 to 2. If the 0 value is entered, the error is not recovered.

Format: ASCII string, real, real, real

Note:

To calculate the skew error use this formula:

$$Eskew = \frac{Emaster - Eslave}{2}$$

Besides, the skew gain affects the errors of both the master and the slave axes as follows:

Emaster = Emaster-1 + (Eskew * gain)

Eslave = Eslave-1 + (Eskew * gain)

The greater the gain the higher the axes control. Yet, if the gain is too great, oscillation will occur, whereas if the gain is too small, solidarity between the axes tends to decrease.

The advisable values range from 1 to 1.5. The skew gain should never reach 2.

MCZ

This record is used for setting the home position microswitch parameters for the current axis.

MCZ=zero-input,direction,speed

Where:

zero-input	Input machine logic signal which represents the home position microswitch input. Example: I19A6, U60K2.
Direction	Direction during the search for the zero position microswitch of the relevant axis: 0 = positive direction 1 = negative direction

speed	Speed of the axis during the electrical zero search function. The speed must be sufficiently low to make sure that no pitches are skipped. The speed must be expressed in millimeters per minute for linear axes; in degrees per minute for rotary axes.

Format: variable SIPROM, boolean (two figures), real

Observations:

As zero input parameter you can characterize one of the inputs used for parameters characterization in record MFC.

Example:

```
MCZ = I19A0,0,200
...
...
MFC = I19A0,I19A1
```

POS

This record is used for setting the current axis positioning tolerance parameters.

POS=pos-allow,wait-pos

Where:

pos-allow	Positioning allowance. This represents the distance from the theoretical positioning point within which the axis must be positioned for the movement to be considered concluded. The value must be expressed in millimeters for a linear axis; in degrees for a rotary axis and in revs for a spindle axis.
wait-pos	Position allowance waiting time. This is the maximum period of time the axis may take to move into the allowed position range after the calculation has been terminated. When this time is reached, the error is indicated and the machine tool is deactivated. The time is expressed in seconds. With a spindle axis, this parameter is not declared.

Characteristics:

If for parameter pos-allow a value lower than the minimum resolution of the axis (given by the ratio between the mechanical pitch and the electrical pitch) is used, the axis is activated but its positioning is not controlled.

SRV

This record is used for defining the various types of servo error on the current axis.

SRV=servo-error-stand-by,servo-error-vff,servo-error-no-vff

Where:

servo-error stand-by	Maximum error allowed when the axis is stationary. The value is expressed in millimeters.
servo-error vff	Maximum error allowed when the axis is in motion with VFF. The value is expressed in millimeters.
servo error no-vff	Maximum error allowed when the axis is in motion without VFF. The value is expressed in millimeters.

Format: real, real, real

Characteristics:

With a spindle axis or point-to-point axes, deactivate the servo error using the command SRV=0,0,0

MAN

This record is used for setting the speed and acceleration parameters for the manual movement of the current axis. This record is not declared for a spindle axis with a converter.

MAN=max-man-speed,max-accel

Where:

max-man-speed	Maximum manual speed for the axes, expressed in millimeters per minute for linear axes and degrees per minute for rotary axes.
max-accel	Maximum manual acceleration, expressed in millimeters per square second for linear axes; degrees per square minute for rotary axes. These two values must not exceed those declared in record RAP.

Format: real, real

Characteristics:

If RAP=1 is declared in file PGCFIL, the maximum manual speed corresponds to the speed during the home position microswitch search function.

GMx

This record is used for setting the maximum speed allowed for the specified range of the current axis. The record also defines the reference value and the gain constant.

GMx=max-speed,kc,kv

Where:

x	Number of the spindle axis range: from 1 to 4. If the axis is not a spindle axis, set n = 0 (declare record GM0).
max-speed	Maximum speed, expressed in millimeters per minute for linear axes, in degrees per minute for rotary axes. For a spindle axis, it represents the maximum angular speed of the spindle, expressed in revs per minute, allowed in the specified range.
kc	Value expressed in volts corresponding to the maximum speed of the axis. It is advisable to adjust the 7.5 value for problems concerning D/A converter linearity. For a spindle axis it represents the reference value corresponding to the maximum angular speed of the spindle. In this case it is advisable to adjust 7,5.
kv	Gain constant in a sample of spindle and non-spindle axes. It is an empirical value to be measured on the machine. Typically Kv = 20. The spindle gain is applied while the spindle is being positioned.

Format: real, real, real

Characteristics:

With a spindle axis, the same number of GMx records as there are spindle ranges must be declared (x = 1-4).

Parameter kc must be declared even in the case of a spindle axis without servo control (a.c. motor). In this case the standard value 7.5 must be declared.

With a spindle axis without a transducer the kv value is not used by the system. (Set kv = 0).

LOP

This record is used for setting the operating limits for the current axis.

LOP=pos-op-limit, neg-op-limit

Where:

pos-op-limit	Positive operating limit for the absolute home position microswitch. Value expressed in millimeters for linear axes and in degrees for rotary axes.
neg-op-limit	Negative operating limit for the absolute home position microswitch. Value expressed in millimeters for linear axes and in degrees for rotary axes.

Format: real, real

Characteristics:

For an absolute axis with a "double resolver" the operating limits for the axis are activated when the system is switched on. For a non-absolute axis, the operating limits are not activated until the axis has been reset.

If both parameters are set to zero, the operating limits are deactivated.

MFC

This record is used for setting the physical overtravel input for the current axis.

MFC=overtravel-pos-input,overtravel-neg-input

Where:

overtravel pos-input	Machine logic signal representing the positive overtravel input.
overtravel neg-input	Machine logic signal representing the negative overtravel input.

Format: variable SIPROM, variable SIPROM

TSM

This record is used for setting the parameters for inversion of the current spindle axis.

TSM=max-inver-time,max-inver-ref

Where:

max-inver time	Maximum spindle inversion time. It represents the time taken by the spindle to move from the clockwise limit to the anticlockwise limit. The time is expressed in seconds.
max-inver ref	Maximum reference range during inversion. It represents the reference corresponding to the maximum clockwise speed plus the reference corresponding to the maximum anticlockwise speed. It is basically the value of the spindle kc (record GMx) multiplied by 2.

Format: real, real

Characteristics:

Remember that the parameters entered in record TSM are used by the process to synchronize the inversion of the working axes, the rotation of the spindle in the fixed cycle G84 (only for spindle without a transducer) and the deactivation of the spindle in the fixed cycle G86. It is therefore important to measure with accuracy the value of the inversion time to be entered in record TSM.

Record TSM must be declared even in the case of a spindle axis without servo control (a.c. motor).

Example:

TSM=3,17

ASM

This record is used for defining the name of the master axis for the current slave axis.

ASM=master-axis

Where:

master-axis	Name of the master axis used with the current slave axis.

Format: ASCII string

Characteristics:

Record ASM must be declared in the subsection of the slave axes which may be coupled to the master axes to perform specific work cycles. At the moment the record must be declared in the following two cases:

- On lathes in the subsection of the spindle. In this case the master axis must be declared to obtain the correct constant cutting speed. Typical value ASM=X;

- In the subsections of the axes with set-points. In this case, the axis with which it is to be used must be declared. Typically the spindle axis: ASM=S.

POM

This record is used for setting the parameters for positioning the current spindle axis.

POM=spiaz,limit-speed

Where:

spiaz	Defines the displacement in revolutions between the transducer electricl zero and the oriented spindle position. The machine logic may request greater displacement than the set value.
limit-speed	Limit speed. It defines the angular speed of the spindle, expressed in revolutions per minute, below which the spindle orienting algorithm is activated. It is normally set at 100 to 200 revs per minute.

Format: real, real

Observations:

This record is only entered in the subsections of the spindle axes without transducers.

ADP

This record is for setting the parameters for the analogica converter used with the current axis.

ADP=board-num, volt

Where:

board-num	Number of the analogical converter associated to the displaying of the spindle power. Its value is positive if an intelligent board is used, or negative if a non-intelligent board is used.
volt	Voltage value, expressed in volts, measured at the analogical input to which 100% of the power is applied.

Format: integer, real

ZNO

This record is used for setting the parameters for positioning the axis with respect to the home position microswitch cycle.

ZNO=null-offset, zero-shift

Where:

null-offset	Value, expressed in millimeters or inches, at which the axis must be positioned after the home position microswitch cycle.
zero-shift	Numeric value which enables the interpolator to examine a possible CSI routine which causes a dynamic offset in real time on the current interpolated point: 0 = disables 1 = enables

FBF

This record defines the parameters used to signal a possible "Transducer Error".

FBF = space, time

Where:

space — Space, expressed in millimeters or inches, where no "Transducer Error" signal is generated.

time — Time, expressed in seconds, during which no "Transducer Error" signal is generated.

Format: real, real

Characteristics:

If this record is not declared, the following default values are assumed:
- Field space: the mechanical pitch value;
- Field time: 40 samplings.

If the space and/or time values declared in the record are equal to zero, the check algorithm of Transducer Errors is disabled.

6.3.5. SECTION 3

Section 3 is used for setting the parameters for correcting geometrical errors. This section is specific for each process and consists of the following records: PRO, NAS, PAS, Exx, NMO. The records of the section must be declared for each process declared in record PRO.

It is possible to relate the geometrical errors to all the axes for a maximum of 1000 points per axis. The correction points on an axis must be equidistant. For each axis with geometrical errors the following records must be declared: NAS, PAS, NMO.

The geometrical errors declared occupy the memory containing the servo control of the axis concerned.

If more correction points than the memory can handle are declared, during initialization the system indicates a memory overflow.

Compensation for geometrical errors has a field of application with well-determined limits. The error calculation table uses variables in integer format, whose values range from -32768 to +32767.

The values contained in the table must satisfy the following conditions:

a) $$-32768 \leq \left[\frac{cp}{E(n) - E(n-1)} \right] \leq +32767$$

b) $$-32768 \leq \left[\frac{ep}{mp} \times E(n) \right] \leq +32767$$

where:

cp = correction pitch

ep = electrical pitch

mp = mechanical pitch

E(n) = current correction point

E(n-1) = previous correction point

The minimum pitch accepted by the system depends on the fast speed of the axis and on the servo control clock pulse. The minimum pitch is calculated using the following expression:

$$\text{minimum pitch} = \frac{Fs \times clock}{60} \times 2$$

where, Fs is the fast speed expressed in meter per minute and clock is the servo control clock pulse expressed in milliseconds.

Example:

Fast speed = 12 m/min; servo control clock pulse = 10 ms

$$\text{Minimum calibrated pitch} = \frac{12000 \times 0,002}{60} \times 2 = 0,8 \text{ mm}$$

RECORDS OF SECTION 3:

PRO

This record is used for setting the number of the current process selected for characterization.

PRO=proc-num

Where:

proc-num	Number of the process to be selected for characterization: from 1 to 5. This number must be lower than or equal to the number of processes declared in record NBP of AXCFIL.

Format: integer

Observations:

Record PRO must be repeated for each of the processes declared.

NAS

This record is for declaring the current axis for correcting geometrical errors.

NAS=axis-name

Where:

axis-name	Name of the axis on which the geometrical errors are to be corrected.

Format: ASCII string

PAS

This record is used for setting the correct pitch for the current axis.

PAS=correct-pitch

Where:

correct-pitch	distance, expressed in millimeters, between two consecutive geometrical error correction points. This value must be constant.

Format: real

Exxx

This record defines the error between the tranducer reading and the real position of the current axis.

Exxx=error

Where:

xxx	Number of the corrector.
error	Error detected between the transducer reading and the real measurement. The value is expressed in millimeters with a sign (as detected) and must consist of three figures.

Format: real

Characteristics:

The numbers of the correctors declared must be in ascending but not necessarily consecutive order.

Example:

```
E100 = .004
E110 = -.009
E120 = .006
```

NMO

This record is used for defining the corrector used with the absolute home position microswitch of the current axis.

NMO=correct-num

Where:

correct-num	Number of the corrector which corresponds to the home position microswitch. A correction point must be made to correspond to the machine zero. The point corresponding to the zero, declared in Exxx, must be declared in record NMO. Example: if point E12=.001 it corresponds to the machine zero, declare NMO=E12.

Format: ASCII string

Observations:

The errors must be declared from the negative position to the positive position and the pitch must be positive.

6.4. EXAMPLE OF AXCFIL FILE SETTING

The following example refers to the control of two processes with the folowing features:

- Process 1:

• Two coordinated axes with encoder transducer;
• Spindle axis without a transducer.

- Process 2:

• Two coordinated axes with encoder transducer.

This is a list of an AXCFIL file for the features mentioned above.

```
;SECTION 1
;
*1
NBP=2,3FFF
TIM=12,0,0,0,0
;
; PROCESS 1
;
PRO=1
IN1=1,XZ,S,12,16
CAS=1,XZS,12
;
; PROCESS 2
;
PRO=2
IN2=1,XZ,,12,16
CAS=1,XZ,12
;
;SECTION 2
;
*2
; PROCESS 1
;
PRO=1
;
;COORDINATED AXIS
;
NAS=X
TPA=41,
NTC=1,1
RAP=10000,800
GAS=0,0
PAS=10000,10
MCZ=I13A0,0,200
POS=.01,5
SRV=.5,2,10
MAN=2000,200
```

```
GM0=10000,7.5,20
LOP=18,-480
MFC=I13A0,I13A1
;
;COORDINATED AXIS
;
NAS=Z
TPA=01,
NTC=2,2
RAP=10000,800
GAS=0,0
PAS=10000,10
MCZ=I13A2,0,200
POS=.01,5
SRV=.5,2,10
MAN=2000,200
GM0=10000,7.5,20
LOP=17,-580
MFC=I13A2,I13A3
;
;SPINDLE AXIS
;
NAS=S
TPA=820,
NTC=3,3
GAS=0,0
PAS=5000,1
SRV=0,0,0
TSM=8,15
POS=0,0
RAP=,12
GM1=200,7.5,15
GM2=600,7.5,15
GM3=1500,7.5,15
GM4=3000,7.5,15
ASM=Z
POM=.334,100
;
; PROCESS 2
;
PRO=2
;
;COORDINATED AXIS
;
NAS=X
TPA=1,
NTC=1,1
RAP=10000,800
GAS=0,0
PAS=10000,10
MCZ=I13A6,0,200
POS=.01,5
SRV=.5,2,10
MAN=2000,200
GM0=10000,7.5,20
LOP=18,-480
MFC=I13A6,I13A7
;
;COORDINATED AXIS
```

```
;
NAS=Z
TPA=01,
NTC=2,2
RAP=10000,800
GAS=0,0
PAS=10000,10
MCZ=I13A4,0,200
POS=.01,5
SRV=.5,2,10
MAN=2000,200
GM0=10000,7.5,20
LOP=17,-580
MFC=I13A4,I13A5
;
;SECTION 3
*3
; PROCESS 1
;
PRO=1
;
NAS=X
PAS=5
E040=0.02
E050=-.05
E060=-.088
E070=0.02
E080=.08
E090=-.05
E100=-.088
E110=0.02
E120=-.05
E130=-.088
E140=0.02
NM0=E060
;
NAS=Z
PAS=5
E010=.08
E020=-.05
E030=-.088
E040=0.02
E050=-.05
E060=-.088
E070=0.02
E080=.08
E090=-.05
E100=-.088
E110=0.02
E120=-.05
E130=-.088
E140=0.02
NM0=E020
;
; PROCESS 2
;
PRO=2
```

```
;
NAS=X
PAS=1
E040=0.02
E050=-.05
E060=-.088
E070=0.02
E080=.08
E090=-.05
E100=-.088
E110=0.02
E120=-.05
E130=-.088
E140=0.02
NMO=E070
;
NAS=Z
PAS=1
E010=.08
E020=-.05
E030=-.088
E040=0.02
E050=-.05
E060=-.088
E070=0.02
E080=.08
E090=-.05
E100=-.088
E110=0.02
E120=-.05
E130=-.088
E140=0.02
NMO=E040
```

7. TECHNOLOGICAL PROCESS CHARACTERIZATION

7.1. THE TECHNOLOGICAL PROCESS

The technological process consists of all the numeric control activities involved in the mechanical working on the machine tool. There are basically two ways to control the technological process.

- By program.

- From the console.

Control by program consists in the implementation of machining cycles by means of part programs. The part program, by means of a specific language, allows all the operations required for machining to be performed.

Control from the console allows intervention on the machining cycles programmed via part programs, from the operator console. This intervention is performed by means of potentiometers and selectors available on the operator console.

7.2. GENERATION OF FORMATTED FILES

Control of the technological process requires access to a series of files normally used for the handling of instruments and typical functions, such as tools, movement and the origins of axes.

These files are: files of origins, corrector files, tool monitoring files, random tool files, axis movement files. If these files are used, they must be declared in file PGCFIL via record FIL.

For each type of file and for the five processes which can be configured, the names shown below are recommended:

Files of origins: FI1EOR, FI2EOR, .FI3EOR, FI4EOR, FI5EOR

Corrector files: FI1COR, FI2COR, FI3COR, FI4COR, FI5COR

Tool monitoring files: GE1TOL, GE2TOL, GE3TOL, GE4TOL, GE5TOL

Random tool files: FI1RAN, FI2RAN, FI3RAN, FI4RAN, FI5RAN

Axis movement files: FI1MOV, FI2MOV, FI3MOV, FI4MOV, FI5MOV

If the system is used in a single process environment, a single file for each type can be defined (FI1EOR, FI1COR, GE1TOL, FI1RAN, FI1MOV).

7.2.1. FILES OF ORIGINS

These files contain the tables of the origins of the axes, and reside in the memory areas declared in file PGCFIL, record FIL.

Before a file of origins is generated, the format of the records contained in the file must be declared in the previously opened FORMAT file (see Chapter 5, "Section 7").

The file of origins for all the declared processes can then be generated, via the following command:

FOR,FInEOR/device,xx

where:

n	Number of the process to which the file of origins refers.
device	Memory device on which the specified file of origins reside. The same device as declared in record FIL of file PGCFIL must be declared.
xx	Number of origins to be assigned to the process (max. 99).

All the procedures relative to writing the FORMAT file and to loading the file of origins are fully described in the following manuals: "NC-110 MC, Use and programming manual" and "NC-110 TC, Use and programming manual".

7.2.2. CORRECTOR FILES

These files contain the tool corrector tables, and reside in the memory areas declared in file PGCFIL, record FIL.

Before a corrector file is generated, the format of the records contained in the file must be declared in the previously opened FORMAT file (see Chapter 5, "Section 7").

The corrector files for all the declared processes can then be generated, via the following command:

FOR,FInCOR/device,xx

where:

n	Number of the process to which the corrector file refers.
device	Memory device on which the specified corrector files reside. The same device as declared in record FIL of file PGCFIL must be declared.

xx	Number of correctors to be assigned to the process (max. 9999).

All the procedures relative to writing the FORMAT file and to loading the corrector files are fully described in the following manuals: "NC-110 MC, Use and programming manual" and "NC-110 TC, Use and programming manual".

7.2.3. TOOL MONITORING FILES

These files contain the tool life tables, and reside in the memory areas declared in file PGCFIL, record FIL.

Before a tool monitoring file is generated, the format of the records contained in the file must be declared in the previously opened FORMAT file (see Chapter 5, "Section 7").

The tool monitoring files for all the declared processes can then be generated, via the following command:

FOR,GEnTOL/device,xx

where:

n	Number of the process to which the tool monitoring file refers.
device	Memory device on which the specified tool monitoring files reside. The same device as declared in record FIL of file PGCFIL must be declared.
xx	Number of tools handled by the tool monitoring file (max. 9999).

All the procedures relative to writing the FORMAT file and to loading the tool monitoring files are fully described in the following manuals: "NC-110 MC, Use and programming manual" and "NC-110 TC, Use and programming manual".

7.2.4. RANDOM TOOL FILES

These files contain the tables of tools for handling RANDOM type tool changes, and reside in the memory areas declared in characterization in file PGCFIL, record FIL.

For this type of file it is not necessary to declare the format of the records in the FORMAT file. The random tool files are generated for each process declared with the following command:

CRE,FInRAN/device,4,xx

where:

n	Number of the process to which the random tool file refers.
device	Memory device on which the specified random tool files reside. The same device as declared in record FIL of file PGCFIL must be declared.
xx	Position number in the tool magazine.

When the open operation has been terminated, the random tool file is ready to be written via editor or loaded from an external device with the command COP. For further details, refer to manual: "Siprom - Interface Programming", section:
Auxiliary Functions, par.: T Function.

7.2.5. AXIS MOVEMENT FILES

The axis movement files contain the records of axis movement which can be launched by the sachine logic. These files reside in the memory areas declared in record FIL of file PGCFIL and can contain up to 255 records.

For this type of file it is not necessary to declare the format of the records in the FORMAT file.

The axis movement files are generated for each process declared with the following command:

EDI, FInMOV/device

where:

n	Number of the process to which the axis movement file refers.
device	Memory device on which the specified axis movement files reside. The same device as declared in record FIL of file PGCFIL must be declared.

In the axis movement records you must also specify the unit of measure in which coordinates and feedrate are expressed. This allows you to correctly execute a subprogram when the main program has been compiled in a different unit of measure.
Once the subprogram has been executed, the control re-establishes the unit of measure of the main program.

7.3. FILE PGCFIL

7.3.1. FEATURES

The process is characterized by means of file PGCFIL. This file is made up of six sections numbered from 1 to 6. Each section enables a particular element of the process to be characterized.

The general structure of the file is described in Chapter 2, Section "File Structure". The general structure of the file record is described in Chapter 2, Section "Process software characterization".

The file is introduced into the system via the editor, by sending the following command:

EDI,PGCFIL/MP3

One of the following instructions must be inserted at the top of the file: OLD, NEW.

If the instruction NEW is inserted, the system, durihg the characterization bootstrap phase, resets the memory reserved for the process. In this way the last version of PGCFIL is lost, with the parameter and variable values contained in it (for example, the name of the part program selected via SPG, the values of the variables, the function T stored in the spindle, etc).

When the system is bootstrapped, the file PGCFIL containing the instruction NEW is loaded. Thus, this instruction may only be used during the loading and testing of the characterization.

If the instruction OLD is inserted, the system, during the bootstrap phase, does not reset the memory reserved for the process, and accesses the previously stored file PGCFIL. In this way, the file PGCFIL containing the instruction OLD is not loaded into memory and is not interpreted by the system. The preexisting values of the variables and parameters are not lost. It is convenient to use this instruction when the file does not require any other modifications.

When the file PGCFIL is to be loaded or modified, the procedure set out below should be followed:

1. Load or modify the file by inserting the instruction NEW;

2. Switch the system off;

3. Switch the system on and replace the instruction NEW with the instruction OLD;

4. Switch the system off;

5. Switch the system on.

7.3.2. SECTIONS

If the system is used in a multiprocess environment, the various records for each process controlled must be declared in sections 2, 4, 5 and 6 of the file. Processes must not be declared in sections 1 and 3 because the records contained in these sections are common to all the processes.

Section 1 is optional and allows the characterization of the three-letter codes allowed by the process. This section is common to all the processes.

Section 2 is optional and allows the characterization of the system variables. This section is specific to each process.

Section 3 is optional and allows the characterization of the JCL three-letter codes. This section is common to all the processes.

Section 4 is optional and allows the characterization of the system variables, the part program libraries, the files of origins, the corrector, random tool, tool monitoring, axis movement files, communication files and machine logic message files. This section is specific to each process.

Section 5 allows the characterization of the machine tool equipment. This section is specific to each process.

Section 6 allows the characterization of the axes to be moved and the console potentiometers. This section is specific to each process.

7.3.3. SECTION 1

This section is used to characterize the three-letter codes of the part programs allowed by the system and is common to all the processes. If this section is omitted, the tables present in EPROM containing the default three-letter codes are considered valid. Table 7.1. shows the names and default sync codes of the part program three-letter codes. Section 1 is composed of record TRI only.

SECTION 1 RECORDS:

TRI

This record is used to modify the names of the part program three-letter codes.

TRI=old-name, new-name, synchro-code

Where:

old-name	Name of the three-letter code to be modified.
new-name	Name to be assigned to the three-letter code. The system replaces the existing name (parameter old-name) with this name. If the letter "D" is introduced, the existing name is deleted.
synchro-code	Default synchronization code of the specified three-letter code: 1 = sync without override capability 2 = sync with override capability 3 = no sync with override capability 4 = no sync without override capability If the code is not declared, the default code shown in Table 7.1. is assumed.

Format: ASCII string, ASCII string, hexadecimal (two digits)

Characteristics:

Synchronization allows a momentary suspension in the processing of the part program to synchronize the calculation of the movement with the movement itself. Processing resumes when the movement of the axes recuperates the delay with respect to the calculation.

Synchronization or non synchronization can be requested by means of two specific characters placed in front of the part program block. These characters are:

- # to request synchronization;

- & to request no synchronization.

Synchronization and non synchronization with override capability requested by a three-letter code can be ignored during the processing of a part program by inserting the characters "#" and "&" in front of the part program block in which the three-letter code is located.

Synchronization and non synchronization with no override capability requested by a three-letter code cannot be ignored during the processing of a part program even by inserting the characters "#" and "&" in front of the part program block in which the three-letter code is located.

Table 7.1. - Three-letter Codes and Default Sync Codes

THREE-LETTER CODE	SYNC CODE
ASC	3
BEQ	3
BGE	3
BGT	3
BLE	3
BLT	3
BNC	3
BNE	3
CAN	3
CLG	3
CLO	3
CLP	3
CLS	3
CRE	3
CTL	3
DAM	3
DCG	3
DEF	3
DER	3
DFP	3
DIS	3
DLO	3
DLY	3
DPI	1
DPT	3
DSA	3
DSC	3
DTL	1
EPF	3
EPP	3
ERP	3
EXE	2
FIL	3
GET	1
INP	3
MIR	3
OPN	3
OUT	3
PUT	1
RED	3
REL	1
RPT	3
RQO	1
RQP	3
RQU	3
SCF	1
SCR	3

Table 7.1. continued

THREE-LETTER CODE	SYNC CODE
SND	2
SOP	1
SPA	3
SPF	3
SPP	3
TGL	3
TOF	3
UAO	3
UAV	1
UCG	3
UIO	3
UOT	3
URT	3
USS	1
WAI	1
WRT	3

Example:

TRI=CLS,D,

This record is used to delete the name of the three-letter code CLS.

TRI=EXE,RUN,

This record replaces the name EXE with the name RUN. The sync code is the default code (2) and requires synchronization with override.

If the character "&" is inserted before the record EXE:

&EXE

the three-letter code sync request is ignored and is activated non synchronization request.

7.3.4. SECTION 2

Section 2 is optional and allows the characterization of the system variables contained in the symbol table. This section is specific to each process.

Each of these records must be repeated for each of the processes controlled by the system.

In particular, this section defines the default type and the types allowed for each variable.

The default type is the type automatically assumed by the system for a determined variable. The types allowed are any type that a determined variable can allow, other than the default type.

The types allowed for the system variables are:

- boolean;
- byte;
- integer;
- long integer;
- real;
- long real;
- ASCII.

The system variables that may be modified by means of this section are contained in the symbol table shown in Table 7.2. below.

Table 7.2. - Symbol Table

VARIABLE (1)	MAXIMUM NUMBER (2)	DEFAULT TYPE (3)	ALLOWED TYPE (4)	SYNC CODE (5)	MODIF. PARAMS.
E	40	6	62	3	2
O	5	0	0	3	2
P	15	0	0	3	2
L	15	0	0	3	2
C	15	0	0	3	2
TMR	1	6	32	3	1
UOV	1	6	32	3	1
JOG	1	6	32	3	1
RTA	1	6	8	3	1
RTO	1	6	8	3	1
ERF	1	6	32	3	1
MCD	1	6	32	3	1
USB	1	1	1	3	1
UVR	1	1	1	3	1
USO	1	1	1	3	1
URL	1	1	1	3	1
UCV	1	2	2	3	1
RAP	1	1	1	3	1
UAS	1	1	1	3	1
RMS	1	2	2	3	1
UEP	1	1	1	3	1
SA	1024	1	127	2	1
SK	1024	2	127	2	1
SYVAR	200	2	127	3	1

Table 7.2. continued

VARIABLE (1)	MAXIMUM NUMBER (2)	DEFAULT TYPE (3)	ALLOWED TYPE (4)	SYNC CODE (5)	MODIF. PARAMS.
TIM	7	4	16	3	1
TOT	7	4	16	3	1
SSL	1	6	32	4	1
ACP	1	6	32	4	1
VOL	1	1	1	3	1
ERR	1	1	1	3	1
IOSTA	1	2	2	3	1
MBR	1	1	1	3	1

The "Modifiable Parameters" column of the symbol table shows the numbers corresponding to the parameters which can be modified for each variable. These numbers are shown in brackets at the top of the relative columns of the variable parameters.

The geometric variables o, p, l, c cannot have not any other type of variable. For these variables, a zero is shown in the column relative to the default type.

For variables o, p, l, c, the maximum number of elements which can be configured is 256. For vsriable E, the maximum number of elements is 8192.

SECTION 2 RECORDS:

PRO

This record defines the number of the current process selected for characterization.

PRO=proc-num

Where:

proc-num	Number of the process to be selected for characterization: from 1 to 5. This number must be less than or equal to the number of processes declared via record NBP of AXCFIL.

Format: integer

Observations:

Record PRO must be repeated for each of the processes declared.

SIM

This record modifies the attributes of the specified variable, or defines new variables.

SIM=var-name,var-new,max-num,def-type,amm-type,synchro-code

Where:

var-name	Name of the variable with one or more attributes to be modified. If a new variable is to be defined, the expression "NEW" must be introduced.
var-new	New name of the variable, if the name of the variable specified in var-name is to be modified. Name of the new variable to be defined, if the var-name parameter is "NEW".
max-num	Maximum number of elements allowed for the specified variable. Table 7.3. shows the maximum number of elements for the new variables defined by the user.
def-type	Code of the specified variable default type. The code is in hexadecimal. The type must be allowed by the system. The allowed types with the relative codes are listed in Table 7.4.

amm-type	Numeric value which defines the types allowed for the specified variable. If the variable can have several types, the numeric value to declare must be the sum of the numeric values relative to the individual types. The types allowed are listed in Table 7.5. with the relative nmeric values.
synchro-code	Default sync code for the specified variable: 1 = sync without override capability 2 = sync with override capability 3 = no sync with override capability 4 = no sync without override capability

Format: ASCII string, ASCII string, word, hexadecimal (two digits), word, hexadecimal (two digits)

Observations:

The record SIM must be repeated for each of the processes declared.

In each process a maximum of 20 new variables of the symbol table can be declared.

Characteristics:

The numeric value which specifies the types allowed for a determined variable is obtained by summing the numeric values of the individual types. For example, if numeric value 54 is declared, obtained by the sum of values 32, 16, 4, 2 (see Table 7.5.) the following types are allowed: long real, real, integer, byte.

Table 7.3. - Maximum Number of Variable Elements

VARIABLE TYPE	MAX. NUMBER OF ELEMENTS
Boolean	65519
Byte	65519
Integer	32758
Long integer	16379
Real	16379
Long real	8189
ASCII	65519

Table 7.4. - Default Types and Codes

VARIABLE TYPE	CODE
Boolean	1
Byte	2
Integer	3
Long integer	4
Real	5
Long real	6
ASCII	7

Table 7.5. - Allowed Types and Numeric Values

VARIABLE TYPE	NUMERICAL VALUE
Boolean	1
Byte	2
Integer	4
Long integer	8
Real	16
Long real	32
ASCII	64

The validity of an argument is conditioned by the "override" byte of the variable. The "override" byte has not been introduced, in order to ensure that the user doesn't use the file PGCFIL parameters to modify variables essential to the system.

In the variables defined by the user, this byte is generated in such a way that the variables cannot be modified during characterization.

Example:

SIM=E,,200,,,

This record modifies the number of elements of variables E from 40 (default value) to 200.

7.3.5. SECTION 3

This section is optional and is used to characterize JCL three-letter codes. If this section is omitted, the tables present in EPROM containing the default JCL three-letter codes are considered valid. It is made up of the JCL record and is common to all the processes.

SECTION 3 RECORDS:

JCL

This record modifies the names of the JCL three-letter codes.

JCL=old-name, new-name, synchro-code

Where:

old-name	Name of the three-letter code to be modified.
new-name	Name to be assigned to the specified JCL three-letter code. The system replaces the existing name (parameter old-name) with this name. If the letter "D" is introduced, the existing name is deleted.
synchro-code	Default sync code of the specified three-letter code: 1 = sync without override capability 2 = sync with override capability 3 = no sync with override capability 4 = no sync without override capability. If the code is not declared, the default code shown in Table 7.6. is assumed.

Format: ASCII string, ASCII string, hexadecimal (two digits)

Observations:

The sync code should not be changed, as these commands are issued with the **SEND** key.

Examples:

JCL=CAO,D,

This record deletes the name of the three-letter code CAO.

JCL=VTU,UTV,

This record replaces the name VTU with the name UTV. The sync code is the variable default code, 4.

Table 7.6. - JCL Three-Letter Codes and Sync Codes

JCL THREE-LETTERALS	SYNC CODE
CAC	4
CAO	4
CLG	4
CTU	4
DBT	4
DCG	4
DIS	4
DPT	4
ERM	4
ESE	4
EVA	4
GSE	4
ORA	4
PTM	4
RCM	4
REL	4
RIF	4
SPG	4
UCA	4
UCG	4
URP	4
VIC	4
VOA	4
VTU	4

7.3.6. SECTION 4

This section is used to characterize the part program libraries and the files of origins, the corrector, random tool, tool monitoring and axis movement files. This section is specific for each process and is made up of the following records: PRO, ASS, NPL, NDD, PRF, FIL, STR, SER, CHN, SCR.

SECTION 4 RECORDS:

PRO

This record defines the number of the current process selected for characterization.

PRO=proc-num

Where:

proc-num	Number of the process to be selected for characterization: from 1 to 5. This number must be less than or equal to the number of processes declared by means of record NBP of AXCFIL.

Format: integer

Observations:

The record PRO must be repeated for each of the processes declared.

ASS

This record is used to assign a real type value to a system variable.

ASS=var-name,value

Where:

var-name	Name of the system variable to which the value is to be assigned.
value	Real type value which must be assigned to the specified variable.

Format: ASCII string, real

Characteristics:

The real value is converted into the default variable format.

If the variable is of the boolean type, there are two possibilities:

- The real value is other than zero. In this case the value one is assigned to the boolean variable;

- The real value is zero. In this case the value zero is assigned to the boolean variable.

Values can be assigned to all variables with the procedures described.

ASS=TMR,value
value: LONG REAL format
Defines the time interval at end of block in G04.

ASS=UOV,value
value: LONG REAL format
Defines the stock allowance value.

ASS=JOG,value
value: LONG REAL format
Defines the length of the block to execute in JOG state.

ASS=RTA,value
value: LONG REAL format
Defines the probe requalification value for the abscissa axis.

ASS=RTO,value
value: LONG REAL format
Defines the probe requalification value for the ordinate axis.

ASS=ERF,value
value: LONG REAL format
Defines the max. dynamic form error.

ASS=MCD,value
value: LONG REAL format
A pure number between 0 and 2. Defines the value of maximum deviation of the direction cosines between two consecutive elements; if this value is exceeded, a stop is forced.
The value defined in MCD is only active in continuous mode G27.
E.g.: ASS = MCD,1
is equal to a max. deviation of 90 degrees between two consecutive elements; if this value is exceeded, the movement out of the element occurs at zero speed. The outwards speed is greater than zero if the deviation between the two elements is from 0 to 90 degrees.

ASS=USB,value
value: BOOLEAN format
0 = enables the interpretation of the slashed blocks
1 = disables the interpretation of the slashed blocks

ASS=UVR,value
value: BOOLEAN format
0 = disables the function
1 = enables rapid speed on all the axes progrmed with working speed

ASS=USO,value
value: BOOLEAN format
0 = disables OPTIONAL STOP (M01)
1 = enables OPTIONAL STOP (M01)

ASS=URL,value
value: BOOLEAN format
0 = function disabled
1 = enables rapid speeds limited by the potentiometer for manual movements

ASS=UCV,value
value: BYTE format
0 = dispiays the calculated coordinates
1 = displays the coordinates read by the transducers
2 = displays the axes position error

ASS=RAP,value
value: BOOLEAN format
0 = disables function
1 = enables automatic return on the profile after arrest in HOLD and displacement and/or automatic search of the home position micro in the axis clearing manoeuvre.

ASS=UAS,value
value: BOOLEAN format
0 = connects the axes
1 = disconnects the axes for testing programs with stationary axes

ASS=RMS,value
value: BYTE format
Defines a percentage of speed variation in the return phase of the tapping cycle.

Example:

ASS=RMS,110 increases return speed by 10%

ASS=RMS,10 decreases return speed by 90%

ASS=UEP,value
value: BOOLEAN format
0 = anables use of VFF
1 = disablas use of VFF

ASS=VOL,value
value: BOOLEAN format
0 = disables wheel
1 = enables wheel

ASS=SSL,value
value: LONG REAL format
Defines the limit of the number of spindle revolutions.

ASS=ACP,value
value: LONG REAL format
Defines the limit of spindle power.

ASS=SK1023,value
value: BYTE format
Defines the value of the 1024th byte of K buffer, corresponding to W255K3.

ASS=MBR,value
value: BOOLEAN format
0 = Exit from retrace status
1 = Enable retrace movements

ASS=E25,value
value: REAL format

ASS=E30,value
value: LONG REAL format

NOTE. No error is indicated if the maximum values defined are exceeded.

NPL

This record defines the maximm number of part programs and labels which can be recalled in a work sequence.

NPL=prog-num,label-num

Where:

prog-num	Number of part programs which can be recalled in the selected work sequence. Maximum value is 255. If this parameter is omitted, the system assumes by default 10 programs.
label-num	Number of labels which can be recalled in the selected work sequence. Maximum value is 100. If this parameter is omitted, the system assumes by default 255 labels.

Format: integer, integer

Characteristics:

If one of the parameters in record NPL is set to zero, the default number of part programs or labels is assumed.

23 bytes of user memory are reserved for each label and 12 bytes for each part program present in the work sequence.

Record NPL is used to define a memory area in which the addresses of the labels of all the part programs selected with SPG are stored.

NDD

This record is used to define the default memory device for the part programs.

NDD=device

Where:

device — Name of the default memory device used by the part programs.

Format: ASCII string

Characteristics:

If the record is not declared, the device MP1 is assumed by default.

PRF

This record establishes the number of profiles and rough shaping blocks defined in the part programs.

PRF=prof-num,block-num

Where:

prof-num — Maximum number of profiles which can be defined (DFP). If this parameter is not declared, the system assumes eight profiles by default.

block-num — Maximum number of rough shaping blocks for each profile defined. If this parameter is not declared, the system assumes 16 rough shaping blocks by default.

Format: integer, integer

Characteristics:

Record PRF is used to define a memory area in which the addresses of the various profiles defined in the part programs and of the blocks making up these profiles are stored. This record is only used for lathes.

To accept the default parameters, the following must be written:

PRF= ,

FIL

This record defines the data set of specific files.

FIL=name1/dev1,name2/dev2,name3/dev3,name4/dev4,name5/dev5

Where:

name1	Name of file of origins. The name FInEOR is normally declared, n being the number of the current process.
dev1	Memory device on which the file of origins resides.
name2	Name of the corrector file. The name FInCOR is normally declared, n being the number of the current process.
dev2	Memory device on which the corrector file resides.
name3	Name of the tool monitoring file. The name GEnTOL is normally daclared, n being the number of the current process.
dev3	Memory device on which the tool monitoring file resides.
name4	Name of the random tool file. The name FInRAN is normally declared, n being the number of the current process.
dev4	Memory device on which the random tool file resides.
name5	Name of the axis movement file. The name FInMOV is normally declared, n being the number of the current process.
dev5	Memory davice on which the axis movement file resides.

Format: ASCII string, ASCII string, ASCII string, ASCII string, ASCII string

Where:

name1	Not used.
dev1	Not used.
name2	Not used.
dev2	Not used.
name3	Name of the data set of the file containing machine logic messages. It is normally FInMSG, where n is the number of the current process. The files that contain machine logic messages must be created with the CRE instruction. Prior to ereating one machine logic message file, the following record must be opened in section *7 of the FCRSYS file: FILMS5,MESSAG/MP0
dev3	Memory device that supports the machine logic message file.
name4	Not used.
dev4	Not used.

STR

This record defines the number of structures in ASSET lahguage.

STR=struct-num

Where:

struct-num	Number of structures to be allocated. A maximum of 255 structures can be defined.

Format: word

Characteristics:

This record acts on formatted files. These files are made up of records with predetermined fixed formats (integer, ASCII, real, etc.), thus the number of structures must be declared in order to access the records of the file.

The structures declared in the part programs must be of the same format as the records (e.g. file VITUT, etc.). Each field can have different meanings. For further information, refer to the manuals "NC-110 MC, Use and programming manual" and "NC-110 TC, Use and programming manual".

SER

This record defines the memory used to handle the user oriented serial line.

SER=byte-num

Where:

byte-num	Number of bytes used for handling the user oriented serial line.

Format: hexadecimal (two digits)

Example:

If the following is declared:

SER=0

the line is handled with a standard number of bytes (850 bytes).

If you declare a value other than zero (n), the reserved area will occupy 850 + n bytes. n can occupy as many as 64 kbytes.

SER is normally set to zero when only one driver is managed.

CHN

This record is used to define the number of channels for access to formatted files in ASSET language.

CHN=chan-num

Where:

chan-num	Number of channels to allocate for access to files via ASSET language.

Format: word

Characteristics:

This record acts on formatted files. If the value 1 is declared in the chan-num parameter, only one file at a time can be accessed.

The convenience of increasing the number of channels should be evaluated, as for each channel enabled, approx. 750 bytes of user memory are used.

The programmer should open a channel, define the structure or structures and access the specific records.

Remember that to access another file, the previously opened channel must be closed.

For further information, refer to the manuals "NC-110 MC, Use and programming manual" and "NC-110 TC, Use and programming manual".

SCR

This record defines the display items in ASSET language.

SCR=dim

Where:

dim	Size, in bytes, of the memory for the user display with ASSET language.

Format: word

Characteristics:

An item of minimum size is formed of 6 bytes distributed in the following way:

- one byte per row;
- one byte per column;
- one byte for underlining;
- two bytes (one word) for the length of the item;
- one byte for each ASCII character of the item.

For further information, refer to the manuals "NC-110 MC, Use and programming manual" and "NC-110 TC, Use and programming manual".

The screen display in ASSET leenguage is composed of 21 rows and 64 columns.

7.3.7. SECTION 5

Section 5 is used to characterize the machine tool equipment. This section is typical of each process and is composed of the following records: PRO, NIP, DPM, SMC, TOF, GXX, PRC, CWP, NAM, NPD, G70, MBR, TAS, INU.

SECTION 5 RECORDS:

PRO

This record defines the number of the current process selected for characterization.

PRO=proc-num

Where:

proc-num	Number of the process to be selected for characterization: from 1 to 5. This number must be less than or equal to the number of processes declared via record NBP of AXCFIL.

Format: integer

Observations:

The record PRO must be repeated for each of the processes declared.

NIP

This record defines the interpolator number.

NIP=int-num

Where:

int-num	Alphanumeric character which identifies the number of the interpolator of the coordinated axes already declared in file AXCFIL (record INx).

Format: ASCII string

Observations:

This record must be inserted after record PRO.

DPM

This record defines the operating measuring parameters of the probe.

DPM=appro-quote,safe-quote,mis-speed

Where:

appro-quote	Probe approach value expressed in millimeters.
safe-quote	Probe safety value expressed in millimeters.
mis-speed	Probe measuring speed expressed in meters per minute.

Format: real, real, real

Characteristics:

The probing resolution represents the smallest length which can be detected by the probe. This depends on the interpolator clok pulse and the measuring speed according to the following formula:

$$\text{Resolution} = \frac{\text{measuring speed}}{\text{samplings/sec. x 60}} \text{ millimeters}$$

Example:

Interpolator clock pulse = 10 ms
Measuring speed = 100 mm/min

Resolution = 100 / (100 x 60) = 0.016 mm

SMC

This record defines the maximum correction value of the tool corrector.

SMC=max-correct

Where:

max-correct	Maximum correction value for the tool correctors. 1 mm is assumed as default. It is active during tool requalification.

Format: real

TOF

This record defines the type of machine tool controlled by the NC-110 system.

TOF=mach-type

Where:

mach-type	Identifying code (hexadecimal) of the type of machine tool controlled. It can have the following values: 01 = mill only 02 = lathe only 05 = mill which can switch to lathe 06 = lathe which can switch to mill

Format = hexadecimal (two digits)

Characteristics:

If the record TOF=05 is declared and the three-letter code from part program CTL,T is used, it is possible to use the work technology of the lathe on the mill, such as the rough shaping macro, threading, constant cutting speed, feed in mm/giro.

If the record TOF=06 is declared and the three-letter code from part program CTL,F is used, it is possible to use the work technology of the mill on the lathe, that is, to use the parameters of the correctors as corrector length (Z) and tool size respectively.

On power up, if the NC-110 system is configured as MC, the G default functions are: G00, G80, G20, G40, G27, G90, G70, G17, G94, G97.

If the system is configured as TC, the G default functions are: G00, G80, G20, G40, G27, G90, G70, G17, G95, G96, G98.

The format of the corrector files is different for the NC-110 MC and NC-110 TC systems. However, the corrector files for the NC-110 TC may contain information typical of the NC-110 MC system. In this way file records of the NC-110 TC system can be configured with the same meaning as the correctors for the NC-110 MC.

For further information regarding the format of the corrector files, refer to the following manuals:

- "NC-110 MC, Use end programming manual";
- "NC-110 TC, Use and programming manual".

GXX

This record defines the G functions initialized by the syptem at power up.

GXX=class1,class2,class3,class4,class5,class6

Where:

class1 – 6	Code of the G function to be initialized at power up. This record must only be declared if you want the system to initialize a configuration other than the default one, characterized in the TOF record.

Format: decimal (two digits)

Observations:

1. The GXX record must be declared after TOF.
2. All the fields are mandatory.
3. If you declare the GXX record, you must omit G70.
4. You can declare a single G function for each class. The allowable codes are:

CLASS	CODE
1	00 - 01 - 02 - 03
2	27 - 28 - 29
3	90 - 91
4	70 - 71
5	93 - 94 - 95
6	96 - 97

Example:

GXX=01, , ,70,95,96

PRC

This record is used for defining the calculation accuracy.

PRC=so-pre

---

where:

so-pre is the calculation accuracy expressed in mm.

Format: real

Observations:

If this record is omitted, the default value is 0.01 mm.

CWP

This record defines the operating mode of pushbuttons and selectors in the various processes.

CWP=cw-console,offset

---

where:

cw-console is an hexadecimal code that allows you to:
. define whether the pushbuttons and selectors are active in other simultaneous processes;
. disable the subsystem screen;
. enable program blocks to be executed in "share" modality (ROUNDING PRIORITY).
The allowable values are listed in Table 7.7.
Each bit is associated to a selector or switch.

Offset the allowable values are:
0 - use length and radius offsets and GTL
1 - use neither offsets nor GTL

Format: hexadecimal (4 digits), boolean (2 digits)

Observations:

- ROUNDING PRIORITY: This feature permits to execute the part programs selected from different processes as follows:

lst part program block ----- process 1
lst part program block ----- process 2
lst part program block ----- process 3
lst part program block ----- process 4
lst part program block ----- process 5
2nd part program block ----- process 1

- cw-console: This parameter can have the following values:

Table 7.7. - How to set the cw-console parameter

HEX CODE	BIT	VALUE	MEANING
01	0	1	not used
02	1	1	Hold
04	2	1	Spindle speed override
08	3	1	Feedrate override
10	4	1	Manual rate
20	5	1	Selected mode
40	6	1	Cycle start
80	7	1	Disable screen
100	8	1	Rounding priority
200	9	1	not used
400	10	1	not used
800	11	1	not used
1000	12	1	not used
2000	13	1	not used
4000	14	1	not used
8000	15	1	not used

This table specifies the hexadecimal value with the corresponding bit set to 1. You can combine different bits, in which case the value to be declared is obtained by adding the weights of the bits set to 1.

NAM

This record defines the name of the axis parallel to the default spindle axis.

NAM=paral-spind

Where:

paral-spind	Name of the axis parallel to the default spindle axis.

Format: ASCII string

NPD

This record defines the default abscissa and ordinate axes of the machining plane.

NPD=abs-axis,ord-axis

Where:

abs-axis	Name of the default abscissa axis of the machining plane.
ord-axis	Name of the default ordinate axis of the machining plane.

Format: ASCII string, ASCII string

Observations:

If the process contains a single axis in the machining plane the same axis name must be indicated ln the two parameters.

G70

This record defines the unit of measurement used for the values.

G70=value

Where:

value	Code (boolean) which defines the units of measurement used for characterization: 0 = millimeters 1 = inches

Format: hexadecimal (two digits)

Observations:

If the record G70 is not declared, the system assumes millimeters as default.
Do not declare this record if you have already declared a GXX record.

MBR

This record defines the number of blocks which can be executed in retrace.

MBR=retrace

Where:

retrace	Maximum number of part programs blocks which can be processad in retraca in both auto and semiauto. This value can be from 1 to 64, if the size of the part program allows it.

Format: word

TAS

This record defines the functional parameters of the probe in the probing cycles G72 and/or G73.

TAS=probe-input,probe-status,path,type

Where:

probe-input	Input machine logic signal used to read the status of the probe in cycles G72 and/or G73.
probe-status	Status of the probe. This indicates the value of the input machine logic signal when the probe is released. It can be 0 or 1.
path	Preferential approach path on the various axes of the probe, it has the following format: Axis name (X,Y,Z,U,V,W,A,B) Direction (can be + or -)
Type	Type of probe: S = can be oriented N = cannot be oriented

Format: SIPROM variable, boolean, ASCII string, ASCII string

Observations:

Parameters path and type are not currently used.

INU

This record defines the functional parameters of the probe in probing cycle G74 (tool integrity cycle).

INU=probe-input,probe-status

Where:

probe-input	Input machine logic signal used to read the status of the probe in cycle G74 (measures the difference between the value measured and the theoretical value). Can also be the same input signal declared in record TAS.
probe-status	Status of the probe. This value indicates the value of the input machine logic signal when the probe is released. It can be 0 or 1.

Format: SIPROM variable, boolean

7.3.8. SECTION 6

Section 6 is optional and allows the characterization of the axes to be moved and of the console potentiometers. This section is typical of each process and is composed of the following records: PRO, MAS, FRO, SSO, FMO.

SECTION 6 RECORDS:

PRO

This record defines the number of the current process selected for characterization.

PRO=proc-num

Where:

proc-num	Number of the process to be selected for characterization: from 1 to 5. This number must be less than or equal to the number of processes declared via record NBP of AXCFIL.

Format: integer

Observations:

The record PRO must be repeated for each of the processes declared.

MAS

This record defines the axes that can be moved manually. It also allows the display of the values relative to the specified axes.

MAS=axis-name

Where:

axis-name	List of the names of the axes which can be moved manually. The values of these axes are displayed on the system video. A maximum of 8 axes can be declared.

Format: ASCII string

Observations:

The names of the axes must not be separated by commas.

Example:

MAS=XYZ

This record is used to move axes X, Y and Z manually. It also allows the display of the values relative to the three axes.

FRO

This record is used to modify the default values set by the feed rate override selector.

FRO=feed1,feed2,...,feed12

Where:

feed1...12	Numeric value which defines the feed rate override in correspondence with each of the 11 positions of the relative selector.

Format: real, real, ..., real

Characteristics:

All 12 values which can be selected must be defined, even if a single value only is to be modified.

If this record is not declared, the system assumes the default values indicated in the following table.

Table 7.8. - Feed Rate Dafault Values

POSITION	FEED RATE
1	0
2	0.125
3	0.25
4	0.375
5	0.5
6	0.625
7	0.75
8	0.875
9	1.
10	1.125
11	1.25
12	1.25

The values of positions 11 and 12 must be aqual, as the system considers 12 selector positions although there are only 11.

SSO

This record is used to modify the default values set by the spindle speed override selector.

SSO=speed1,speed2,...,speed12

Where:

speed1...12 — Numeric value which defines the spindle speed override in correspandence with each of the 11 positions of the relative selector.

Format: real, real, ..., real

Characteristics:

All 12 values which can be selected must be defined, even if a single value only is to be modified.

If this record is not declared, the system assumes the default values indicated in the following table.

Table 7.9. - Spindle Speed Override Default Values

POSITION	SPINDLE SPEED OVERRIDE
1	0.75
2	0.80
3	0.85
4	0.90
5	0.95
6	1.0
7	1.05
8	1.1
9	1.15
10	1.20
11	1.25
12	1.25

The values of pesitions 11 and 12 must be equal, as the system considers 12 selector positions although there are only 11.

FMO

This record is used to modify the default values set by the feed manual override selector.

FMO=manual1,manual2,...,manual12

Where:

manual1...12 — Numeric value which defines the feed manual override in correspondence with each of the 11 positions of the relative selector.

Format: real, real, ..., real

Characteristics:

All 12 values which can be selected must be defined, even if a single value only is to be modified.

If this record is not declared, the system assumes the default values indicated in the following table.

Table 7.10. - Feed Manual Override Default Values

POSITION	FEED MANUAL OVERRIDE
1	-1.00
2	-0.50
3	-0.20
4	-0.05
5	-0.01
6	0.00
7	0.01
8	0.05
9	0.20
10	0.50
11	1.00
12	1.00

The values of positions 11 and 12 must be equal, as the system considers 12 selector positions although there are only 11.

7.4. EXAMPLE OF PGCFIL FILE SETTING

The following example shows the list of a PGCFIL file relative to the handling of two processes.

```
NEW
*1
TRI=BNC,BRE,
TRI=UIO,UOI,
*2
PRO=1
SIM=UAS,USA,,,,
PRO=2
SIM=UCW,URR,,,,
*3
JCL=ORA,AXO,4
JCL=RCM,D,
*4
PRO=1
NDD=MP2
ASS=ERF,.01
ASS=MCD,1
FIL=FI1EOR/MP3,FI1COR/MP3,,,FI1MOV/MP3
FLC=,,FI1MSG/MP3,
PRO=2
NDD=MP2
ASS=ERF,.01
ASS=MCD,1
FIL=FI2EOR/MP3,FI2COR/MP3,,,FI2MOV/MP3
FLC=,,FI2MSG/MP3,
*5
PRO=1
NIP=1
DPM=10,12,500
TOF=6
CWP=1F,0
NAM=Z
MPD=Z,X
TAS=I15A26,1,,
PRO=2
NIP=3
TOF=2
CWP=1F,0
NAM=Z
NPD=Z,X
*6
PRO=1
MAS=XZ
PRO=2
MAS=XZ
```

8. MACHINE LOGIC CHARACTERIZATION

8.1. MACHINE LOGIC

Machine logic is a software interface which allows personalization of numeric control on a specific machine tool. This interface allows communication between the numeric control and the machine tool, and between the process software and the machine tool electrical cabinet in particular.

Communication is via specific variables (signals and words) which are updated by the system with a frequency, known as sampling, determined in the characterization phase of the system.

The implementation of the machine logic is the phase following the characterization phase described in this manual.

For further information regarding machine logic, refer to the following manual: "NC-110, SIPROM, Interface Programming".

8.2. FILE IOCFIL

8.2.1. FEATURES

The machine logic is characterized by file IOCFIL. File IOCFIL is made up of four sections numbered from 1 to 4. Each section characterizes a particular machine logic element or group of elements.

The general structure of the file is described in Chapter 2, Section "File Structure". The general structure of the file records is described in Chapter 2, Section "Process Software Characterization".

The file is introduced into the system via the editor, by sending the following command:

EDI,IOCFIL/MP3

8.2.2. SECTIONS

If the system is used in a multiprocess environment, the various records for each process controlled must be declared in sections 2 and 3 of file IOCFIL. The records declared in sections 1 and 4 are valid for all the processes.

Section 1 defines the procedures for allocation, synchronization and execution of the machine logic and of any user routines. This section is common to all the processes.

Section 2 is used to define the characteristics and execution procedures of the auxiliary functions. This section is specific to each process.

Section 3 is used to characterize the activities handled by the machine logic. This section is specific to each process.

Section 4 is used to define the variables of T rack. This section is common to all the processes.

8.2.3. SECTION 1

Section 1 defines the proceduras for allocation and execution of the machine logic. It is common to all the processes and is made up of the following records: ALM, INx, OUx, CLO, RUS.

SECTION 1 RECORDS:

ALM

This record defines the allocation address for the execution of the machine logic object.

ALM=obj-add

Where:

obj-add	Allocation address for the execution of the machine logic object. This address must coincide with the value declared in the "exec address" parameter of the AMBIENT.

Format: hexadecimal (four digits)

Characteristics:

If ALM=0 is declared, the machine logic is not enabled on system power up during the characterizatisn bootstrap.

INx

This record defines the physical input boards.

INx=board1,board2,...,board8

Where:

x	Numeric code which identifies the record declared: two INx records numbered from 0 to 1 can be declared. Each record allows eight input boards to be defined.
board1,... ,board8	Numeric code of hardware calibration identifying the input board. It can be from 0 to 5.

Characteristics:

The INx records allow a maximum of 5 input boards (two records) to be defined. This record is declared when the user wishes to access the system working memory area. This is possible if a virtual address is declared in SIPROM AMBIENT, in the "I/O load address" and "I/O exec address" parameters. By modifying this record, it is possible to force the values (0 or 1) of the physical inputs using the SIPROM debugger. If the board physical address (1000 hexadecimal) is declared in the "I/O load address" and "I/O exec address" parameters, this record must not be declared.

Format: word, word, ..., word

Example:

IN0 = 0,1,2,3,,,,,

This record declares the existence of 2 boards, each determined by a specific hardware setting: 0, 1.

OUx

This record defines the physical output boards.

OUx=board1,board2,...,board8

Where:

x	Numeric code which identifies the record declarad: two OUx records numbered from 0 to 1 can be declared. Each record allows eight output boards to be defined.
board1,... ,board8	Numeric code of hardware calibration identifying the output board. It can be from 0 to 5.

Characteristics:

The OUx records allow up to 5 input boards (two records) to be defined. This record is declared when the system work memory area is to be accessed. This is possible if a virtual address is declared in SIPROM AMBIENT, in the "I/O load address" and "I/O exec address" parameters. If however the board's physical address (1000 hexadecimal) is declared, this record must not be declared. In both cases [virtual address and physical address] values (0 and 1) of the physical inputs can be forced by using the SIPROM debugger.

Format: word, word, ..., word

Example:

OU0 = 4,5,,,,,

CLO

This record defines the machine logic sampling times.

CLO=tick-logic ,time-slow

Where:

tick-logic	Logic tick. This is the time, expressed in milliseconds, of fast and slow logic activation. This time must be equal to the CPU clock pulse, or an integer multiple of the CPU clock pulse.
time-slow	Portion of logic tick time which the system dedicates to the processing of slow logic only. This time is expressed in milliseconds and must be less than or equal to half the logic tick.

Format: word, word

Characteristics:

The time declared in SIPROM environment for tha execution of fast logic (parameter "fast time max") must be less than or equal to half the time declared in time-slow parameter, so that sufficient time is reserved for the execution of slow logic.

CLE

This record defines a data area reserved for the CSI that is reset during initialization.

CLE=sta-add,length

Where:

sta-add	is the starting address of the data area reserved for the CSI, to be reset every time the system is initialized. It normally corresponds to the last 4 Kbytes of segment 4000.
length	is the number of bytes occupied by the CSI area.

Format: hexadecimal (four digits),decimal

Example:

CLE=4FOO,4096	reset max. 4 Kbytes at the end of segment 4000

RUS

This record defines a user routine and its modes of synchronization.

RUS=user-rout,synchro-code,rout-time

Where:

user-rout	Name of the user routine. The name is composed of an alphanumeric string of six characters, the first two of which must be the characters RT.
synchro-code	Sync code of the user routine. It can have the following values: 1 = Routine executed at each cycle of fast logic. 2 = Routine executed at each cycle of slow logic. 3 = Routine executed on the basis of a time declared by the user.
time-rout	Activation time of the user routina expressed in milliseconds. It must be equal to an integer multiple of the CPU clock pulse. This time must only be provided if synchro-code = 3. If synchro-code = 1 or synchro-code = 2, the logic tick is assumed.

Format: ASCII string, word, word

Observations:

The logic tick is declared in record CLO of PGCFIL. The CPU clock pulse is declared in racord TIM of AXCFIL.

Example:

RUS=RTCSI1,2,

This example regards the declaration of a CSI routine synchronized with slow logic.

8.2.4. SECTION 2

Section 2 is used to define the characteristics and the modes of execution of the auxiliary functions. This section is specific to each process and is made up of the following records: PRO, Mxx, Hxx, GPS. The records of the section must be declared for each process declared via the record PRO.

SECTION 2 RECORDS:

PRO

This record defines the number of the current process selected for characterization.

PRO=proc-num

Where:

proc-num	Number of the process which is to be selected for characterization: from 1 to 5. This number must be less than or equal to the number of processes declared via record NBP of AXCFIL.

Format: integer

Observations:

The record PRO must be repeated for each of the processes declared.

Mxx

This record defines the number of M functions and their characteristics.

Mxx=m-type,m-request,class

Where:

xx	M function code: from 0 to 99.
m-type	Hexadecimal code which identifies the type of M function specified. The hexadecimal code and the corresponding type of function are obtained from the configuration of the byte indicated in Table 8.1.

m-ruquest	Hexadecimal code of the type of request effected by the specified M function. The hexadecimal code and the corresponding request are obtained from the configuration of the byte indicated in Table 8.2.
class	Hexadecimal code which identifies the display and stored search classes of the specified M function. The hexadecimal code and the corresponding classes are obtained from the configuration of the byte indicated in Table 8.3. 16 display classes and 16 stored search classes are available, numbered from 0 to 15.

Format: hexadecimal (two digits), hexadecimal (two digits), hexadecimal (two digits)

Chararacteristics:

Table 8.1. and Table 8.2. indicate the hexadecimal values with the corresponding bits at 1. The bits can be in various combinations to activate several characteristlcs. In this case the hexadecimal value to be declared is obtained by the sum of the weighted values of the corresponding bits at 1.

Two classes ara specified in the byte of Table 8.3. The display class is determined by the weighted values of the first four bits. The stored search class is determined via the weighted values of the last four bits.

The M functions requesting reset and compensation changes must also request calculation block.

An M function with stored search class 0 and display class 0 is not stored during the stored search of the part program.

The expedite type functions have the following characteristics:

- Can be programmed in continuous mode.

- The BCD code issued (WO3Rl) remains stored all the time that continuous mode is active. When continuous mode is axited, the code is no longer stored.

When the function has the same display and/or stored search classes, only the last M function inserted in the block or searched by the function RCM is displayed and/or issued.

At the end of a stored search, the M functions are issued in the following way:

- The function with the smallest class is issued first.

- If there are M and H functions belonging to the same class, the start motion declared functions are issued first, and then the end motion declared functions. See below Table 8.0.

Table 8.0. - Output Saquence of M and H Functions

COMBINATION	OUTPUT SEQUENCE
M motion start - H motion start	M - H
M motion end - H motion start	H - M
M motion start - H motion end	M - H
M motion end - H motion end	M - H

Table 8.1. - Type of M Function

HEXADECIMAL CODE	BIT	MEANING
0001	0	1 = Motion start function
0002	1	1 = End of motion function
0004	2	1 = Function acceptable in hold
0008	3	1 = Function not to be displayed
0010	4	1 = Expedite function
0020	5	1 = - Reservad -
0040	6	1 = Modal function
0080	7	1 = Function displayed after reset

Table 8.2. - M Function Requests

HEXADECIMAL CODE	BIT	MEANING
0001	0	1 = - Not significant -
0002	1	1 = Forces conditional semiauto
0004	2	1 = Calculation stop
0008	3	1 = Forces semiauto force
0010	4	1 = Compensation change request
0020	5	1 = Reset request at end of execution
0040	6	1 = - Not meaningful -
0080	7	1 = - Reserved -

Table 8.3. - M Function Class

BIT	MEANING
0-3	Hexadecimal code of the display
4-7	Hexadecimal code of the stored search class

Hxx

This record defines the number of H functions and their characteristics.

Hxx=h-type,class

Where:

xx	Function H code: from 0 to 99.
h-type	Hexadecimal code which characterizes the type of H function specified. The hexadecimal code and the corresponding type of function are obtained from the configuration of the byte indicated in Table 8.4.
class	Hexadecimal code which identifies the display and stored search classes of the specified H function. The hexadecimal code and the corresponding classes are obtained from the configuration of the byte indicated in Table 8.5. 16 display classes and 16 stored search classes are available, numbered from 0 to 15.

Format: hexadecimal (two digits), hexadacimal (two digits)

Characteristics:

Table 8.4. indicates the hexadecimal values with the corresponding bits at 1. The bits can be in various combinations to specify several types of H functions. In this case the hexadecimal value to be declared is obtained by the sum of the weighted values of the corresponding bits at 1.

Two classes are specified in the byte of Table 8.5. The display class is determined by the weighted values of the first four bits. The stored search class is determined via the weighted values of the last four bits.

An H function with stored search class 0 and display class 0 is not stored during the stored search of the part program. When the function has the same display and/or stored search classes, only the last H function inserted in the block or searched by the function RCM is displayed end/or issued.

At the end of a stored search, the M functions are issued in the following way:

- The function with the smallest class is issued first.

- If there are M and H functions belonging to the same class, the start motion declared functions are issued first, and then the end motion declared functions. See Table 8.0.

The expedite H functions are not stored.

Table 8.4. - Type of H Function

HEXAEDECIMAL CODE	BIT	MEANING
0001	0	1 = Motion start function
0002	1	1 = End of motion function
0004	2	1 = Function acceptable in hold
DOO8	3	1 = Function not to be displayed
0010	4	1 = Expedite function
0020	5	1 = - Reserved -
0040	6	1 = Modal function
0080	7	1 = Function displayed after reset

Table 8.5. - H Function Classes

BIT	MEANING
0-3	Hexadecimal code of the display class
4-7	Hexadecimal code of the stored search class

GPS

This record is used to define the type of auxiliary functions handling.

GPS-manage-code

Where:

manage-code — Code which identifies the type of handling of auxiliary functions S, T, M, H.
0 = serial handling
1 = parallel handling

Format: hexadecimal (two characters)

Characteristics:

Parallel handling consists in the contemporary issue of functions S, T and of the first M and H functions of start or end motion programmed in the block.

Serial handling consists in the sequential issue of the auxiliary functions in the following order: S, T, M start motion, H start motion, indexing axis. The indexing axis function is always issued last.

If the record GPS is not declared, serial handling is assumed by default.

8.2.5. SECTION 3

Section 3 allows the characterization of the activities handled by the machine logic. This section is specific to each process and is composed of the following records: PRO, ASM, TAx, ASx, UCDA, ADV, CWD. The records of this section must be declared for each process declared via record PRO.

SECTION 3 RECORDS:

PRO

This record defines the number of the current process selected for characterization.

PRO=proc-num

Where:

proc-num	Number of the process which is to be selected for characterization: from 1 to 5. This number must be less than or equal to the number of processes declared via record NBP of AXCFIL.

Format: integer

Observations:

The record PRO must be repeated for each of the processes daclared.

ASM

This record specifies the name of the current spindle axis.

ASM=spindle

Where:

Spindle	Name of the spindle axis.

Format: ASCII string

TAx

This record defines the characteristics of the point-to-point axes.

TAx=axis-name,axis-type,phase-disp,axis-pos,limit1,limit2

Where:

x	Number of the point-to-point axis. This number is read by the machine logic to move the specified point-to-point axis.
axis-name	Name of the point-to-point axis declared in record INx of AXCFIL.
axis-type	Hexadecimal code which identifies the type of point-to-point axis. The hexadecimal code and the corresponding type of axis are obtained from the configuration of the byte indicated in Table 8.6.
phase-disp	displacement between electrical zero and mechanical zero of the point-to-point axis. It is expressed in stations (fractions of stations are also considered). This value must be less than the mechanical pitch declared in record PAS of AXCFIL.
axis-pos	Number of the positions of the point-to-point axis.

The values to be characterized in fields limit 1 and limit 2 have different meanings according to the point-to-point axis used, namely:

a) Point-to-point axis without D/A converter

limit 1	First deceleration threshold of the point-to-point axis which is communicated to the machine logic during axis movement. The value is expressad in stations (fractions of stations are also considered).
limit 2	Second deceleration threshold of the point-to-point axis which, is communicatad to the machine logic during axis movement. The value is axpressed in stations (fractions of stations are also considered).

b) Absolute-linear point-to-point axis with resolver transducer and D/A converter

limit 1 — The field value must not be characterized.

limit 2 — This parameter is used for determining the position of the absolute axis after the control has been turned off and then on. To calculate the value of this parameter, use the following formula:

$$LIMIT2 = \frac{\text{value characterized in axis-pos}}{2 * n^{\circ} \text{ of "real" stations of p.p.axis}}$$

c) Point-to-point axis different from the types of steps (a) and (b)

The value of fields limit1 and limit2 must not be characterized.

Format: ASCII string, hexadecimal (two digits), real, real, real, real

Observations:

An absolute axis is an axis with a one to one ratio between transducer and point-to-point axis revolutions. In this case the home positlon micro search is not necessary.

Characteristics:

Table 8.6. - Type of Point-to-Point Axis

BIT	**MEANING**
0 = Point-to-point axis	without D/A converter 1 = Point-to-point axis with D/A converter
1	0 = Linear point-to-point axis 1 = Cyclic point-to-point axis
BIT 3|2	
0 0	Non-absolute axis with home position micro
0 1	Absolute axis with encoder
1 1	Absolute axis with resolver
BITS 4-7	- Not meaningful -

This table shows the meaning of the individual bits. The value to be declared in the axis-type parameter is obtained by converting into hexadecimal the sum of the weighted values of the individual bits at 1.

ASx

This record defines the characteristics of the indexing axes.

ASx=axis-name,moda,axis-type,pos-num

Where:

x	Number of the indexing axis: from 1 to 3
axis-name	Name with which the specified indexing axis is programmed.
mode	Code which indicates the modes of driving the specified indexing axis: 0 = absolute mode 1 = incremental mode
axis-type	Code which defines the type of indexing axis specified: 0 = rotary axis 1 = iinear axis
pos-num	Number of positions of the specified axis.

Format: ASCII string, hexadecimal (two digits), hexadecimal (two digits), word

Characteristics:

In absolute mode the digits relative to the position programmed which the indexing axis must reach are communicated to the machine logic. In incremental mode the digits relative to the number of positions (in clockwise or anticlockwise direction) that the axis must effect in incremental mode to reach the programmed position are communicated to the machine logic.

UCDA

This record defines the use of the D/A converter relative to the point-to-point axis.

UCDA=axis-name

Where:

axis-name	Name of the point-to-point axis on whose D/A converter the machine logic can force the analog reference. A maximum of 3 point-to-point axes can be defined. Driving can be on three different channels but only on one at a time.

Format: ASCII string

ADV

This record defines the operating procedures of the electronic wheel.

ADV=axis-vol

Where:

axis-vol	Name of the electronic wheel axis declared in racord NAS of AXCFIL (for the digital/digital wheel). Number of the input on the wheel D/A converter board (for the digital/analog wheel). This parameter sets as in the record NTC.

Format: ASCII string (D/D wheel)
sign followed by a number (D/A wheel)

Characteristics:

To activate and use the electronic wheel, the conditions set out below must be satisfied.

Activation:

- Mode selector set to "continuous manual" or to "manual jog".
- Program VOL=1
- The selected axis must have an enable request.
- The axes must not be disconnected.
- The cycle start pushbutton must not ba pressed.
- The feed hold must not be active.
- The control must not be in hold.
- The signal COMU must be at 1 (reversed).

Use:

When the electronic wheel is active (VOL=1), the selected axis moves if the following conditions are satisfied:

- the axis is present in the process axes table and is not virtual;

- the axis is not a spindle axis (e.g., NAS=S).

CWD

This record disables the activities handled by the machine logic.

CWD=ctrl-word

Where:

ctrl-word	Control word. A hexadecimal code which indicates which activity handled by the machine logic must be disabled. The hexadecimal code and the corresponding disabled activity are obtained from the configuration of the byte indicated in Table 8.7.

Format: haxadecimal (four digits)

Characteristics:

If certain activitias controlled by the machine logic are not used, they can be disablad by the control word, in order to make the system faster. The following table shows the structure of the control word. This table indicates the hexadecimal value with the corresponding bit at 1. The bits can be in various combinations to disable various activities. In this case the hexadecimal value to declare is obtained by the sum of the weighted values of the corresponding bits at 1.

Table 8.7. - Activities Handled by the Machine Logic

HEXADECIMAL CODE	BIT	MEANING
0001	0	1 = Spindle handling
0002	1	1 = Wheel handling
0004	2	1 = Display handling
0008	3	1 = Console handling
0010	4	1 = Auxiliary axis 1 handling
0020	5	1 = Auxiliary axis 2 handling
0040	6	1 = Balancing axes handling
0080	7	1 = Logic commands handling
0100	8	1 = D/A converter handling
0200	9	1 = - Reserved -
0400	10	1 = Axes enable handling
0800	11	1 = End of travel disable handling
1000	12	1 = - Reserved -
2000	13	1 = Fast logic signals handling
4000	14	1 = Axes movement handling
8000	15	1 = - Reserved--

8.2.6. SECTION 4

Section 4 is used to define the SIPROM variables of T rack, it is common to all the processes and is composed of record Txx only.

SECTION 4 RECORDS:

Txx

This record assigns a value to the SIPROM variables of T rack.

Txx=value

Where:

xx	Numeric code which identifies the current record in section 4: from 1 to 128.
value	Value to be assigned to the SIPROM word variable corresponding to the current record. The value must be provided in hexadecimal format. Values from 00 to FF hexadecimal (0-255 decimal) can be assigned.

Format: hexadecimal (two digits)

Observations: Up to 64 records can be declared, each corresponding to a variable of T rack.

Characteristics:

Up to 128 variables of the word type belonging to T rack can be declared in SIPROM programming. The machine logic can access these variables in both read and write modes. For further details refer to the following manual: "NC-110 MC/TC, SIPROM, Interface Programming".

The relation between record Txx and the words declared in T package to which a value is to be assigned is of the following type:

```
T01=W0T0
T02=W0T1
T03=WUT2
........
T63=W31T2
T64=W31T3
```

8.3. EXAMPLE OF IOCFIL FILE SETTING

The following list is an example of setting file IOCFIl relative to the control of two processes.

```
*1
ALM=5008
IN=0,1,2,,,,,
OU1=3,,,,,,,
CL0=10,2
RUS=RTVISU,3,1000
*2
PR0=1
M00=6,C,10
M01=2,6,10
M02=2,24,0
M03=45,0,21
M04=45,0,21
M05=6,0,21
M06=2,14,62
M07=45,0,44
M09=6,0,44
M10=45,0,77
M11=6,0,77
M12=45,0,77
M13=45,0,21
M14=45,0,21
M19=45,0,21
M30=2,24,0
M60=2,4,3
PR0=2
M00=6,C,10
M01=2,6,10
M02=2,24,0
M03=45,0,21
M04=45,0,21
M05=6,0,21
M06=2,14,62
M07=45,0,44
M09=6,0,44
M13=45,0,21
M14=45,0,21
M19=45,0,21
M30=2,24,0
M60=2,4,3
M66=2,14,62
*3
PR0=1
ASM=S
TIS=1
TA1=V,E,O,32,.7,.3
CWD=0040
```

```
ADV=A
PR0=2
ASM=S
TA1 =V ,3,.005,30,,
TA2=B,3,.0216,360,,
AS1 =B,0,0,360
UCDA=B
CWD=0040
*4
T01=10
T02=0
T03=10
T04=0
T05=15
T06=0
T07=2
T08=0
T09=9
T10=0
T11=2
T12=0
T13=68
T14=0
T15=30
T16=0
```

9. EDP AND LOGIC I/O ERRORS AND MESSAGES

This appendix lists the error messages relative to the JOB partition and to the I/O messages for this partition.

9.1. EDP ERRORS

These errors concern the JOB partition and the analysis of file FCRSYS. If the error message file EDPERR has not be characterized, these errors are indicated by the system via the messages "FILMSl nn", where nn is the error message code.

Note that the messages listed here can be characterized by the user via file FCRSYS and file EDPERR (see Chapter 4, "Section 7"). In this case, when the system verifies an error, it displays the error message defined by the user in the characterization phase.

The following list shows the messages for each code.

Table 9.1.

CODE	MESSAGE
FILMS1 01	LOGIC NAME TABLE IS FULL
FILMS1 02	LOGIC NAME ALREADY EXISTS
FILMS1 03	UNDEFINED LOGIC NAME
FILMS1 04	PERIPHERAL ERROR
FILMS1 05	FILE NOT FOUND IN BUBBLE OR CMOS
FILMS1 06	ILLEGAL RECORD TYPE
FILMS1 07	LORD-GO WITHOUT STAXT ADDRESS
FILMS1 08	INVALID COMMAND
FILMS1 09	FORMAT ERROR
FILMS1 10	SYNTAX ERROR
FILMS1 11	UTILITY NOT AVAILABLE
FILMS1 12	LINE ERROR
FILMS1 13	TIME-OUT ERROR
FILMS1 14	ILLEGAL REQUEST

continued table 9.1.

CODE	MESSAGE
FILMS1 15	FORMAT ERROR: SECTION 2 OF FCRSYS
FILMS1 16	BOOTSTRAP ERROR: SECTION 2 OF FCRSYS
FILMS1 17	DRIVER NOT PRESENT IN EPROM
FILMS1 19	EOF RECORD MISSING
FILMS1 20	FILE FCRSYS NOT FOUND
FILMS1 21	POWERED UP IN EMERGENCY MODE
FILMS1 22	FORMAT ERROR: SECTION 1 OF FCRSYS
FILMS1 23	FCRSYS AND HW CONFIGURATION DO NOT MATCH
FILMS1 24	EPROM DIRECTORY ERROR
FILMS1 25	SECTION 3 OF FCRSYS IS EMPTY
FILMS1 26	FORMAT ERROR: SECTION 3 OF FCRSYS
FILMS1 27	BOOTSTP ERROR: SECTION 3 OF FCRSYS
FILMS1 28	FILE NOT PRESENT IN EPROM
FILMS1 29	MEMORY 0VERFLOW
FILMS1 30	COMMAND NOT ALLOWED

9.2. EDP MESSAGES

These messages are displayed by the system in the JOB partition. If the EDP EDPMSG messages file has not been characterized, these messages are indicated by the system via the messages "FILMS2 nn", where nn is the EDP message code.

Note that the messages listed here can be characterized by the user via file FCRSYS and file EDPMSG (see Chapter 4, Section 7). In this case, the system displays the EDP message defined by the user in the characterization phase.

In some cases, the EDP messages ask a question to which the user must respond in order to be able to continue processing.

The following list shows the messages and their meaning for each code.

Table 9.2.

CODE	MESSAGE AND MEANING
FILMS2 01	Not used
FILMS2 02	CONFIRM? (Y/N) This message is displayed when the entire contents of the fill are to be deleted. If the user responds "Y", the file are deleted, if the response is "N", control is restored to the JOB.
FILMS2 03	C0MMAND EXECUTED This message is displayed when a command has terminated.
FILMS2 04	ALREADY EXISTS. UPDATE??? (Y/N) This message is displayed when trying to copy to a file with preallocated space and fixed length. If the user responds "Y", the records of the file are updated starting with the first. If the response is "N", the new file is not copied.
FILMS2 05	ALREADY EXISTS. DELETE??? (Y/N) This message is displayed when trying to copy a file which already exists on the specified medium. If the user responds "Y", the existing file is replaced with the new file, if the response is "N", the new file is not copied.
FILMS2 06	FILE COPIED This message is displayed after a file has been copied.
FILMS2 07	WAIT! JCL BUSY This message is displayed when a JCL command is issued and a command issued by the host is being executed.
FILMS2 08	PARITY ERROR This message is displayed when a parity error is detected during the read of a paper tape punch. The message warns the user to correct the file just loaded into memory. The characters not read correctly are replaced by the "?" character.
FILMS2 09	CONTINUE? (Y/N) This messaga is generated by utility DIF, when a difference between two files is detected. If the user responds "Y" the comparison is carried on, if the response is "N" the utility ends.

9.3. I/O ERRORS

These errors concern the I/O operations-relative to the JOB partition. If the IOERRM error messages file has not been characterized, these messages are indicated by the system via the massagas- "FILMS3 nn", where nn is the I/O error message code.

Nota that tha messages listad hare can be characterized by the user via file FCRSYS and fils IOERRM (see Chapter 4, Section 7). In this case, the system displays the I/O error message dafined by the user in the characterization phase.

The following list shows the messages and their meaning for each code.

Table 9.3.

CODE	MEANING
FILMS3 01	INVALID OPERATION
FILMS3 02	PARAMETERS DO NOT MATCH
FILMS3 03	INVALID DEVICE NAME
FILMS3 04	INVALID RECORD NUMBER
FILMS3 05	INVALID RECORD LENGTH
FILMS3 06	INVALID BUFFER LENGTH
FILMS3 07	INVALID LOGIC CHANNEL
FILMS3 08	INVALID FLAG NUMBER
FILMS3 09	INVALID FUNCTION CODE
FILMS3 10	UNDEFINED LOGIC NAME
FILMS3 11	CHANNEL ALREADY FREE
FILMS3 12	FILE ALREADY OPEN
FILMS3 13	FILE CLOSED
FILMS3 14	RECORD ALREADY WRITTEN
FILMS3 15	NO FREE SECTORS
FILMS3 16	FILE ALREADY EXISTS
FILMS3 17	MEMORY OVERFLOW

continued table 9.3.

CODE	MEANING
FILMS3 18	RECORD LOGICALLY DELETED
FILMS3 19	INVALID DATA SET ASSIGNMENT
FILMS3 20	INVALID OPERATION
FILMS3 21	INVALID ACCESS METHOD
FILMS3 22	LOGIC CHANNEL NOT AVAILABLE
FILMS3 23	RECORD NUMBER DOES NOT EXIST
FILMS3 24	FILE DOES NOT EXIST
FILMS3 25	PROTECTED FILE
FILMS3 26	WRITE PROTECTED FILE
FILMS3 27	END OF FILE
FILMS3 28	BEGINNING OF FILE
FILMS3 29	DEVICE ALREADY ALLOCATED
FILMS3 30	DEVICE NOT READY
FILMS3 31	WRITE PROTECTED DEVICE
FILMS3 32	PARITY ERROR
FILMS3 33	BUFFER OVERFLOW
FILMS3 34	DEVICE NOT FORMATTED
FILMS3 35	HARDWARE ERROR
FILMS3 36	INVISIBLE FILE
FILMS3 37	FORMAT ERROR
FILMS3 38	LINE ERROR
FILMS3 39	TRACK ZERO DESTROYED IN HD-FD FORMAT
FILMS3 40	TRACK DESTROYED IN R, W ON HD-FD
FILMS3 41	FLOPPY NOT INSERTED IN DRIVE
FILMS3 42	NOT USED
FILMS3 43	NOT USED
FILMS3 44	SEARCH FIELD NUMBER ERROR

10. CHARACTERIZATION ERRORS

This appendix provides a list of the error messages which may arise during the process software characterization procedure. All the error messages listed appear on the process software characterization diagnostics screen (see Chapter 4 "Characterization Diagnostics"). The message gives the error code and the file containing the error.

The following sections describe the error messages corresponding to the codes displayed on the diagnostics screen.

Characterization errors are divided into four groups:

- Syntax and format errors common to all the characterization files.

- Errors in file AXCFIL.

- Errors in file PGCFIL.

- Errors in file IOCFIL.

Codes numbers from 1 to 100 indicate syntax and format errors common to all the process software characterization files. Codes numbers of over 100 indicate errors in the individual characterization files.

The name of the file containing the errors is displayed in the top left-hand corner of the diagnostics screen. The names used are:

- AXIS for fila AXCFIL.

- PROC for flle PGCFIL.

- LOGIC for file IOCFIL.

10.1. ERRORS COMMON TO ALL CHARACTERIZATION FILES

CODE	TYPE OF ERROR AND COMMENT
1	Sections missing in the characterization file. This error is indicated when sections of the characterization file have been omitted.
2	Each line of the files must begin with the character "*", or with a letter, or with the character ";".
3	The section has not been specified. Should the characterization file begin without the section 1 specification (*1), this error is detected on each record until the Section 2 specification (*2) is found.
4	The section the system is waiting for is not the same as the one declared by the user. This error is indicated if a section is declared using an asterisk followed by a number which is not consecutive to the number used to declare the previous section. Example: if "*3" is declared after section 1.
5	CR command missing after the section number. The name of the section must be on a separate line (ending with CR LP commands).
6	Unauthorised operator name. The name of the operator is not included in the list of operators allowad access to the current section.
7	Operator name is too long. The maximum length of the operator name is five letters.
8	After the operator's name character '=' must be entered.

CODE	TYPE OF ERROR AND COMMENT
9	Character which may not be used in the operator name. The operator name must consist of upper or lower cases, it may not contain "blank" characters (but it may be followed by "blank" characters) and must end with the character "=".
10	Omission of parameters in a record. The number of parameters required by a record is fixed and must always be respected. If a parameter is not to be used, it is simply omitted. In this case, however, the character separating the parameters (the comma) must be entered.
11	Too many parameters in a record. The number of parameters required by a record is fixed and must always be respected.
13	Error in an integer parameter. This error is indicated when an integer parameter contains non-numerical characters.
14	Integer overflow. The integer value ranges from -32767 to +32.768.
15	Error in a word parameter. This error is indicated when a word parameter contains non-numerical characters.
16	Word overflow. A word value may vary from 0 to 65454.
17	The first character of the SIPROM code must be "I" or "U".

CODE	TYPE OF ERROR AND COMMENT
18	Error in the numeric value of the SIPROM code. This error is displayed if: - a K buffer connector number is greater then 255; - an A buffer connector number is greater then 31; - a T buffer connector number is graater then 15; - the last value of the SIPROM variables code is greater than 31. E.g.: U260K22, I33A10, I33T10, U40K45 are incorrect
19	The letter denoting the rack must be: - K = logic rack K; - A = physical rack A; - T = virtual rack T.
20	Error in a real parameter. This error is indicated when a real parameter contains non-numerical characters.
21	Error in a hexadecimal parameter. This error is indicated when a hexadecimal parameter contains non-numerical characters other then: A, B, C, D, E, F.
22	Too many characters in a hexadecimal parameter. Only hexadecimal values of two figures are allowed.
23	String too long. ASCII string parameters may contain any ASCII characters except for "," and ";" and must not contain more then ten characters.
24	Too many sections in tha fils. Too many sectisns have been definad in the characterization file.
25	The process characterization file must begin with NEW or OLD.

10.2. ERRORS IN FILE AXCFIL

These errors are indicated during the AXCFIL characterization file test.

CODE	TYPE OF ERROR AND COMMENT
101	Clock pulse too long for a CPU. The clock pulse is the period of time between subsequent scheduling within a CPU board. It is expressed in milliseconds and must be contained in one memory byte. The maximum is therefore 255 milliseconds.
102	The servo control does not respond. This error is indicated when, having executed the initialization command for the servo control, the AXIS MANAGER does not send a message indicating that Initialization has been done. The error may be a result of one of the following situations: - A CPU which is not present has been declared in record TIM, (check record TIM and the CPU present in the system). - Synchronization was lost in the initial communication between the system and the servo control. This may be caused by a power failure during the AXCFIL file test or if at power on the synchronization key was already written in the memory (dirty or non initialized memory). If this is the case, switch the control system off and then on again. - The CPU is present but faulty. For example, the EPROMs are installed in the wrong order, the CPU clock generator is faulty.
103	Incorrect servo control CPU slot specification. The servo control CPU are numbered 1. This error is indicated if, in record CAS, the number of CPUs declared is not within these limits.

CODE	TYPE OF ERROR AND COMMENT
104	Servo control clock pulse too long or zero. The servo control clock pulse is axpressed in milliseconds and ranges fro 1 to 255. This error is indicated when these limits are not respected in record CAS.
105	Incorrect interpolation CPU slot specification. The interpolation CPU are numered 1. This error is indicated when the limits are not respected in record INx.
106	Too many elements. The saquence of motion elements controlled by the interpolator varies from 1 to 64 elements. This error is indicated when these limits are not respected in record INx.
107	Interpolation clock pulse too long or zero. The interpolator clock pulse is axpressed in millisaconds and may vary from 1 to 255. This error is indicatad when these limits are not respected in record INx.
108	An interpolator can control one spindle only. The "spind-name" parameter of record INx must consist of one character only.
109	An interpolator cannot be present without axes.
110	Interpolator table overflow. At the moment, a maximum of 9 interpolators may be present in the system at one time.
111	Duplicata interpolator. The name of the interpolator contained in the current record has already been used to define a previous interpolator.
112	At least one axis must be declared in record CAS. At least one servo control must be declared in record CAS which defines the servo controls features.
113	Duplicate axes in an axis defining string. In the "axis-name" parameter of record CAS there are two axes with the same name.

CODE	TYPE OF ERROR AND COMMENT
114	AXis table overflow. A maximum of 4 axes may be present at one time. This error is indicated if more then 3 axes plus the spindle are declared.
115	Axis table overflow of an interpolator. At the moment, a maximm of 4 axes is allowed for each interpolator. This error is indicated if more then 3 axes excluding the spindle are declared.
116	Linear axis declared on several interpolators. An axis may be handled by one interpolator only.
118	Axis declared in several CAS racords of one process, or in several process in the case of a common axis.
119	Axis declared but not defined. This error is indicatad when an axis defined in an INx record is not defined in a CAS record.
120	Interpolation clock pulse which is not a multiple of the servo control clock pulse. The interpolator provides the servo control with data so that synchronization is maintained in the system. The interpolation clock pulse must be greater or equal to the servo control clok pulse. If it is greater, it must be multiple of the servo control clock pulse.
121	The TIM record in section 1 of file AXCFIL is missing.
122	Checksum error on one EPROM of the CPU. This is one of the errors detected by the monitor of the CPU. It is due to the loss of information stored in one of the EPROMs of the CPU.
123	Write-read error on one A of the CPU. This is one of the errors detected by the monitors of the CPU. This is due to an error in one of the RAMs of the CPU.

CODE	TYPE OF ERROR AND COMMENT
124	On a CPU with a clock pulse = 0 there must not be any servo controls or interpolators. If the clock pulse of a CPU board specified by the TIM record is zero, the CPU board is not present, so there cannot be any interpolators or servo controls on it.
125	Clock error in the CPU timer. This is one of the arrors detectad by the monitor of the CPU. It is due to a fault in the timer of tha CPU.
126	Servo controls of one CPU must have the same clock pulse. This condition is necessary as othervise problems regarding the synchronization of the axes may arise.
127	The servo control clock pulse is not equal to or a multiple of the CPU clock pulse.
128	The clock pulse of one of the interpolators is not equal to or a multiple of the CPU clock pulse.
129	80287 error on the CPU. This is one of the errors detected by the monitor of the CPU. It is due to a fault in the mathematical processor 8087.
131	Duplicate record. The current record is repeated in the current NAS subsection.
132	The TPA record is missing.
133	The NTC record is missing. The NTC record must always be declared.
134	The GAS record is missing (the transducer is present). The record must be specifiad in the current axis subsection only if the transducer used with the current axis is present, i.e. if the "trans" parameter of the NTC record is not zero.

CODE	TYPE OF ERROR AND COMMENT
135	The PAS record is missing (the transducer is present). See error 134.
137	The RAP record is missing. If the transducer and the converter are declared (record NTC) and the current axis is not a spindle, the RAP record must ba declared.
138	The MAN record is missing (the transducer is present). See error 137.
139	Manual speed greater than fast speed.
140	Manual acceleration greater than fast acceleration.
141	The POS record is missing (the transducer is present). If the transducer is present the positioning tolerance must be declared in the POS record.
142	The SRV record is missing (the transducer is present).
147	The GMO record must not be declarad if the axis is a spindle.
148	The GM1 record must be declared if the axis is a spindle.
149	The axis specified in record NAS has not been declared. This error is indicated if the specified axis has not been declared in section 1.
150	If the axis is not a spindle only the GMO record must be declared.
151	Record in section 2 of file AXCFIL without previous NAS record. Error indicated if the heading of the subsection is not specified at the beginning of section 2.
152	Records GMx, RAP, RAP, and POS have not been declared using an axis with an interpolator. The speed is expressed in millimetres per interpolator clock (mm/interpolator clock), so the interpolator must be declared.

CODE	TYPE OF ERROR AND COMMENT
153	The electrical pitch must not be zero.
154	The mechanical pitch must not be zero.
155	Spindle in section 2 but not in section 1. This error is indicated when a spindle axis has been defined in section 1 (in the INx record), but in section 2 this axis is not a spindle (in the TPA record).
157	A coordinated axis and a point-to-point axis may not be used together at the same time. A coordinated axis moves in coordination with another axis, and thus cannot be a point-to-point axis.
158	The name of the switched axis must consist of one character only. This condition is required so that the name corresponds to the axis names declared in section 1.
159	The axis declared as the switched axis is not present in record TPA. The axis specified as the swithced axis must be declared in section 1.
160	The transducer number ia incorrect. The transducer number may vary from - 48 to +32.
161	The converter number is incorrect. The converter number may vary from -64 to +127.
163	The home position microswitch research direction must be 0 or 1.
166	The name of the master axis must consist of one character only. This condition is required so that the name corresponds to the axis names declared in section 1.
167	The axis specified as the master axis is not present in record ASM. The axis specified as the master axis must be declared in section 1.

CODE	TYPE OF ERROR AND COMMENT
168	The switched axes have not been declared correctly. An axis declared as a switched axis in record TPA of another axis is not actually switched (in record TPA of the latter axis). For example if X is coordinated and switched on Y, for X: TPA=9,Y and for Y: TPA=9,X. This means that X is swiched on Y but also that Y is switched on X. If this is not the case, the error is indicated.
169	An axis has not been specified in section 2. All the axes defined in section 1 must have a corresponding NAS subsection in section 2.
170	An interpolator has not been initialized correctly. This error is detected by the monitor of the CPU if is unable to correctly initialize one of the interpolators
171	One of the servo controls has not been initialized correctly. This error ls detected by the monitor of the CPU if one of them is unable to correctly initialize one of the servo controls, for all the CPU and all the servo control.
172	In record GMx the "x" parameter may not exceed the value 4.
173	The NAS record or PAS record is missing. This error is indicated in the following cases: - If an Exxx record is declared before the NAS and PAS records have been declared. - If a PAS record is declared not preceded by the related NAS record. - If an NMO record is declared not preceded by the NAS and PAS records.
174	The Exxx records are not in ascending order.
175	At least two correctors must be declared. For the compensation table to be valid, at least two correctors must be declared.

CODE	TYPE OF ERROR AND COMMENT
176	Syntax error in the value of the NMO record. This error is indicated when the MMO record is different from: NMO=Exxx where xxx is a numerical value.
177	Number of compensation on the home position microswitch not present. This error is indicated when in the NMO record the corrector on the home position microswitch has not been declared in a previous Exxx record.
178	The type of axis declared is not conguent with the transducer/converter declared (record NTC). This error is indicated when a type of axis with a transducer is declared in record TPA and the number of the transducer is not declared in the corresponding NTC record. An error is also indicated if the inverse situation occurs.
179	The spindle axis cannot be declared in record TPA as having and not having a transducer at the same time. This error is indicated when there is no transducer on a spindle axis declared as having and not having a transducer (TPA=30).
180	In record TSM the parameters must not be zero. This error is indicated when in record TSM one of the two parameters is omitted or declared as zero.
181	Record TSM must only ba declared in the subsection of a spindle axis (TPA=20, TPA=10, TPA=820, TPA=810).
182	Record GMx must be declared after record NTC.
183	Too many processes declared in record NBP. This error is indicated when a number of processes exceeding the maximum allowed is defined (first parameter of record NBP).

CODE	TYPE OF ERROR AND COMMENT
184	In record PRO the operator has declared a process number axceeding the maximum number of processes declared in record NBP, or has selected a process which does not exceed the limit defined in NBP but which has not previously been configured.
185	Records CAS and INx have baen declared before record NBP. Records CAS and INx are not accepted if record NBP is not declared first.
186	Characterization memory overflow. This error is indicated when the maximum memory limit for characterization, (files AXCFIL, PGCFIL, IOCFIL), declared in record NBP of AXCFIL is exceeded.
187	At least one process must be declared. The process number has not be defined in record PRO.
188	Record POM inserted in the subsection of a nonspindle axis or the PAS record or TPA record are missing. If the axis is not a spindle axis record POM should not be inserted in the subsection. Record POM must be preceded by record PAS and record TPA.
189	Record POM is missing. If the axis is a spindle with a transducer, record POM must be inserted in its subsection.
190	The A/D parameter is out of range. ADP must be between -127 and + 127.
191	The parameter expressed in volts in record ADP must not be 0. In record ADP the second parameter must not be zero.
192	Common axis declared in a non-configurated process. You can only declare a "common" axis in record COM if you have previously declared thet axis in another process.
193	SKW declared in the subsection of a non-split axis. The skew error (SKW) must only be declared in the subsection of split axes (see TPA).

CODE	TYPE OF ERROR AND COMMENT
194	Master axis name longer than one character. The "master-axis" parameter in record SKW, as it is an axis name, must consist of one character only.
195	Master axis not present. A non-configured axis has been declared as the master asis in record SKW.
196	The axis declared as a split axis has already been associated with another axis. In the "master-axis" parameter of record SKW an axis which has not already been associated by means another SKW record must be indicated.
197	The split axis has already been associated with another axis. An axis already associated by means the SKW record in another NAS subsection has been declared as a split axis (see TPA).
198	Error in the declaration of a split axis. A split axis must only be declared in the following records: NAS, TPA, NTC, GAS, MCZ, MFC, SKW. The records not declared or any records declared other then those listed cause this error.
199	Axis associated with itself. A split axis (see TPA) cannot in record SKW be associated with itself.
200	A split axis cannot ba interpolatad. This error is indicated if a split axis is declared in a INx record.
201	The split axis must be declared after the master axis in section 2.
203	If an axis with a sat point is declared, the master axis must be declared (ASM).
209	Declaration of records ZNO or FBF without declaring record PAS first.

10.3. ERRORS IN THE CPUs

The errors listed below concern the servo control CPUs during characterization or operation. The error message may not be modified.

There are five errors, two of which are closely linked to the test run by the axis manager CPU while the servo controls are being configured; the other three may arise at any moment during the normal operation of the machine tool.

MEMORY OVERFLOW CPU n CHANNEL x

This error is displayed when there is not enough storage space in the axis manager to contain all the data required to handle the axis and/or interpolators configured in the CPU. In the message, character "n" indicates the CPU with a full memory and character "x" indicates the channel on which the error is present ("channel" means axis or interpolator).

I/O ERROR CPU n CHANNEL x

This error arises when a servo control is being configured if the input/output devices (converters, transducers, home position microswitches, negative and/or positive overtravel microswitches) ars not present in the system or if they do not pass the initial functional check. In the message, character "n" indicates the CPU which is to control the axis and character "x" indicates the axis (channel) on which the error was detected.

TIME SLICE OVERRUN CAS=n - TIME SLICE OVERRUN INx

These error messages are displayed if the sampling rate of servo controls and/or interpolators is insufficient for the characterization of the axis manager. If the message "TIME SLICE OVERRUN CAS=n" is displayed the character "n" indicates the servo control sampling rate; in the message "TIME SLICE OVERRUN INx" the character "x" indicates the interpolator which caused the error.

→ E8087 (BASE:OFFSET)

This error arises during normal machine, operation, when a calculation error is made by the arithmetical coprocessor 8087. Message "BASE:OFFSET" represents the hexadecimal address of the memory corresponding to the error detected.

NOTE. If these error messages are displayed it is advisable to write down all the messages provided by the system concerning the error detected on the CPU. If the error arose during the execution of a part-program write down the part of the program being executed when the error arose. This data will help the service technician to repair the machine quickly and efficiently.

10.4. ERRORS IN FILE PGCFIL

These errors are indicated during the PGCFIL characterization file test.

CODE	TYPE OF ERROR AND COMMENT
101	Duplicate record.
102	3-letter code not allowed. The 3-letter code specified in section 1 is not allowed.
103	3-letter code not allowed. The character "D" is required. The 3-letter code specified in section 1 is written incorrectly. If the "ASCII string" parameter in record TRI consists of one charater only, this character must be the letter "D".
104	The synchronization code must be more then one and less then four.
105	Symbol table variable index overflow. The index assigned to the variable specified in the record is not within its range.
106	Default type not allowed. The default type must be between 1 and 7.
107	Incorrect code of default types allowed. The code defining the types allowed must be less then 247.
108	Default synchronism not allowed. The default synchronism must be between 1 and 4.
109	New variable name not allowed. The new variable name is already present in the symbol table or is written incorrectly.

CODE	TYPE OF ERROR AND COMMENT
110	Label number or programs not allowed. A maximum of 255 programs and/or 255 labels may be defined.
112	The device name specified is written incorrectly. The device name in record NDD is written incorrectly. This name may consist of a maximum of 3 characters.
113	The interpolator number must be an alphanumeric character. In record NIP the parameter "int-num" must be an alphanumeric character.
114	The interpolator specified is not present, or it has less then 2 elements. In record NIP the paramater "int-num" which defines the interpolator is written incorrectly, or less then 2 alements were characterized for precalculations.
115	The axis name must consist of one character only. In records NAM and NPD the parameters which define the axis names must consist of one character only.
116	The axis specified is not present. In records NAM and/or NPD the name of the axis declared has not been declared in file AXCFIL.
117	The type of axis specifiad is not congruent with the type declared in file AXCFIL. In records NAM and/or NPD the axis declared is not congruent with the one declared in file AXCFIL. For example, an axis which has been declared as a spindle axis in AXCFIL cannot be declared as an interpolation plane axis.
118	Records NIP, NAM, NPD and TOF are missing in PGCFIL. These records are essential.
119	Symbol table variable overflow. A maximum of 20 new symbol table variables may be added by means section 2.
120	The probe status must be 0 or 1.

CODE	TYPE OF ERROR AND COMMENT
122	The preferential direction of approach must indicate the axis and the direction. The parametar "path" of record TAS must consist of two ASCII characters; the first is the axis name and the second must be either "+" or "-".
123	The probe type must be S or N.
124	Duplicate axes in an axis defining string.
125	Records NAM and NPD cannot be declared before declaring record NIP.
126	A maximum of 4 axes may be defined.
128	The parameter specifying the name of the variable must be declared. The error is indicated when one of the parameters specifying the name of the variable is omitted in record SIM and/or ASS.
129	Symbol table variable not found.
131	The switched axis of one of the axes interpolated by the process interpolator must be interpolated by the same interpolator. In record NIP it must be ensured that the switched axis is intarpolated by the process interpolator. If, for example, X is switched on A, both X and A must be interpolated by the same interpolator, othervise this error is indicated.
132	The axis name is not allowed. Not all characters may be used as axis names, a specific group of characters is used.
133	The value in record TOF is incorrect. The value of the "mach-type" parameter in record TOF may be one of the following: 1, 2, 5, 6.
134	Syntax error in the data set specifications in record FIL. A data set consists of: file-name/device-name. The "file-name" must consist of between 1 and 6 characters. The character "/" must be included. The "device-name" must consist of 3 characters.

CODE	TYPE OF ERROR AND COMMENT
135	Too many structures declared. In record STR a maximum of 255 structures may be requested.
136	Too many display descriptors declared. The display screens available to the user are 8 minus the number of processes declared (record NBP). If, declaring record SCR, this limit is exceeded the error is indicated.
137	More than 255 channels declared. In record CHN a maximum of 255 channels may be requested.
138	Records GXX and G70 have both been declared. These records cannot be declared together.
139	Record GXX has been declared before TOF.
140	While declaring GXX, the following errors have occurred: - the parameter is made up of either one or more then two characters; - the parameter contains letters; - the specified G is not allowable; - the specified G is allowable but the parameter is not correct.
184	In record PRO the operator has declared a process number exceeding the maximum number of processes declared in record NBP, or has selected a process which does not exceed the limit defined in NBP but which has not previously been configured.
186	Characterization memory overflow. This error arises when the maximum characterization memory (for files AXCFIL, PGCFIL, IOCFIL), declared in record NBP of AXCFIL has been exceeded.

10.5. ERRORS IN FILE IOCFIL

These errors are indicated during the IOCFIL characterization file test.

CODE	TYPE OF ERROR AND COMMENT
101	duplicate record.
102	The machine logic execution address is missing. The address specified in record ALM is incorrect. This address must correspond to the address declared in the "exec address" parameter of the AMBIENT SIPROM.
103	The fast logic tick is too long or zero. The fast logic tick is expressed in milliseconds and must be contained in one byte, so it cannot exceed 255 ms. This tick must not be zero.
104	The clock pulse of the system CPU which processes the machine logic must not be zero. To correct this error record TIM in file AXCFIL must be checked.
105	The fast logic tick is not a multiple of the CPU clock pulse. To correct this error, check that the fast logic tick is a multiple of the CPU clock pulse by checking the declaration of record CLO or of record TIM of file AXCFIL.
106	(Slow logic time slice) (Fast logic tick/2).
107	Record whith extension not allowed. In the current record, the numeric value must not exceed the maximum valua.
108	Input port already defined in another INx record. The same input port may not be defined in more then one INx record.

CODE	TYPE OF ERROR AND COMMENT
109	Input port already defined in another OUx record. It is not possible to define the same input port in more than one OUx record.
110	Output port already defined in another OUx record. It is not possible to define the same output port in more than one OUx record.
111	Output port already defined in an INx record. It is not possible to define the same output port already defined in the INx record.
113	The first parameter in record Mxx must not be zero.
114	The axis type must be 0 (absolute) or 1 (incremental). In record ASx the "axis-type" parameter must be 0 or 1.
115	The axis name must consist of one character only. In the current record the "axis-name" parameter must consist of one character only.
116	Record CLO is missing.
117	The specified axis is not present. An axis name which does not exist has been declared in the current record.
118	The specified axis is not a spindle. In record ASM an axis which is not a spindle axis has been daclared as a spindle axis.
120	For a point-to-point axis without a D/A converter, the first slow-down threshold must not be zero. If the point-to-point axis has not got a D/A converter, the first slow-down threshold must not be zero ("limit1" and "limit2" paramaters of record TAx).

CODE	TYPE OF ERROR AND COMMENT
121	Record NPx is missing or has a parameter set to zero. The number of stable positions for the point-to-point axis has not be defined in record NPx or the NPx valua is zero. This is only valid if the axis s a cyclic axis.
123	The number entered in record ADV is greater than 128.
124	The number of axes defined must not exceed 4. A maximum of 4 axes may be defined in record UCDA.
126	The I/O card declared is not prasent in the system. An input/output card which is not present in the system has been defined in racord INx.
127	The axis name is not one of the names allowed. Not all characters may be used for the axis names. A specific group of characters is used.
128	The machine logic address is out of the system memory. In record ALM, in which the address of the machine logic header is declared, a memory address not present in the system memory map has been declared.
129	In a rotary indexing axis, the number of displacaments must be specified. When the axis is a rotary indexing axis the number of displacements must be specified in record ASx (fourth parameter of the record).
130	The "manage-code" parameter in record GPS is not 1 or 0. In record GPS the parameter may be: - 0 = serial management of functions M and H; - 1 = parallel management of functions M and H. This error is indicated when this parameter in record GPS is different from the above values.
131	The name of the user routine declared in record RUS doas not consist of 6 characters and/or does not start with the characters "RT". The value of the second parameter is not 1, 2, or 3.

CODE	TYPE OF ERROR AND COMMENT
132	The routine sampling time is not equal to or multiple of the system CPU clock pulse. In record RUS the sampling time must be equal to or multiple of the system CPU clock pulse.
133	The routine declared in record RUS is not present in the memory. To correct this error, check that the memory containing the routine is installed in the system. Check that the routine name is coherent with the name declared in tha initial part of the routine. Check that the memory containing the routine is declared in FCRSYS.
184	In record PRO the operator has declared a process number exceeding the maximum number of processes declared in record NBP, or has selected a process which does not exceed the limit defined in NBP but which has not previously been configured.
186	Charactarization memory overflow. This error arises when the maximum characterization memory (for files AXCFIL, PGCFIL, IOCFIL), declared in record NBP of AXCFIL has been exceeded.