R2E
of
America

MICRAL V PORTABLE MICROCOMPUTER SYSTEM
OPERATOR'S MANUAL

TM-2001
Issued December 1978

3406 University Avenue S.E.
Minneapolis, Minnesota 55414
(612) 378-7060

MICRAL V PORTABLE MICROCOMPUTER SYSTEM
OPERATOR'S MANUAL

TM-2001
Issued December 1978

R2E OF AMERICA
47 Bedford St., S.E.
Minneapolis, Minnesota 55414

Printed in USA

PREFACE

PURPOSE

This manual provides all instructions and data necessary to operate the R2E MICRAL V Portable Microcomputer System using BAL (a superset of BASIC), or FORTRAN IV or Assembly Language, operating under the Sysmic Operating System.

CONVENTIONS USED IN THIS DOCUMENT

Most (but not all) of the operator commands for the system must be terminated by a carriage return. Whenever required, this is indicated in text by the symbol (cr).

Optional fields in instruction descriptions are enclosed by brackets [ ] (do not enter the brackets in the program). In general, spaces can be inserted as desired in command strings to improve readability.

REFERENCES

For a detailed description of the BAL Language (Business Application Language - a superset of BASIC), refer to the MICRAL BAL Reference Manual. This document explains all the programming conventions and formats and provides a detailed description of the command structure of the language, plus examples.

The BAL manual also contains a description of the file management system, describes the command structure and includes examples.

A detailed description of the FORTRAN IV Programming Language can be found in the FORT//80, FORTRAN IV Reference Manual.

Refer to the MICRAL Assembler Reference Manual for a description of programming formats and requirements for the Assembler.

REVISIONS TO THIS DOCUMENT

The descriptions and specifications herein are subject to change without notice. A comment sheet/mailer is bound into the back of this document. Use it to report comments/corrections to the manual, and/or to place your name on the mailing list for any future updates to this manual.

TABLE OF CONTENTS

CHAPTER 1. SYSTEM DESCRIPTION

1.1 GENERAL DESCRIPTION

The MICRAL V is a portable data processing system for business and industry.

The system is composed of an 8080 CPU, 32K of RAM (up to 64K total in 8K increments), a double density Shugart Minifloppy* drive providing 140K of usable on-line storage, a Panaplex 480 character alphanumeric display (12 lines, 40 characters per line), an ASCII keyboard with a separate 18 key numeric pad, a 32 character per line alphanumeric strip printer, and a power supply. The system includes an interface for a second minifloppy drive and a parallel printer interface.

The printer interface is compatable with various printers supplied by the user, including 30 to 60 cps Xerox-Diablo printers and 180 cps to 1000 lpm Centronics printers.

Figure 1 illustrates the system. It is mounted in a suitcase, measuring 22 inches (550 mm) wide, 28 inches (350 mm) deep and 6.7 inches (170 mm) high. It weighs about 18 kg (40 pounds).

The system (not including an external parallel printer) requires 110 Vac, 60 Hz power at about 5 amps, or it can be powered from a 220 Vac or 12 Vdc or 24 Vdc source.

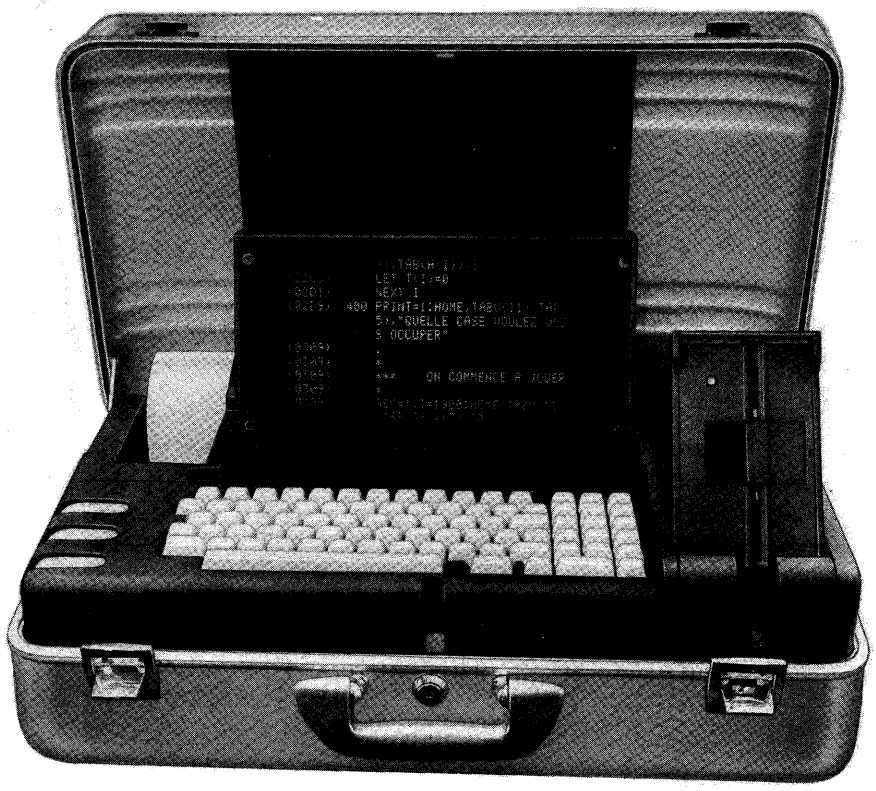

Figure 1. MICRAL V Portable Microcomputer System

*Registered trademark of Shugart Associates

1.2 MINIFLOPPY DISK DRIVE

The system uses a Shugart SA-400 double density minifloppy drive, mounted in the right side of the case. An optional second drive is supplied in its own mounting case. (A planned future product is a suitcase mounted system with two drives standard.) These are rugged mechanical drives using stepping motor actuators. The drives include a write protect feature.

In all operating instructions, the drive mounted in the right hand side of the case will be referred to as Unit 0; the externally mounted drive as Unit 1.

1.2.1 Minidiskettes

The minifloppy diskette, illustrated in Figure 2, stores up to 140K (K=1024) bytes of programs or data as formatted by R2E, using Shugart SA105 hard-sectored minidiskettes, or equivalent. Each 130 mm (5 1/2 in) square minidiskette will store up to 220K bytes, unformatted.

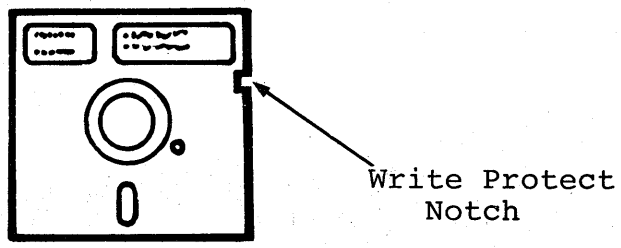

Figure 2. Minidiskette

Each minidiskette has 35 tracks, with 16 sectors of 256 characters per sector (double density). The SA105 is a hard-sectored diskette -- with 16 sector holes per diskette.

NOTE: Do not attempt to use 10 sector hard-sectored diskettes in this system. They will not operate properly with the system software.

The SA105 minidiskette has a write protect capability. A Write Protect Notch is located on the diskette jacket as shown in figure 2. When the notch is open, writing is allowed. When the notch is covered with a Write Protect Tab, writing is inhibited.

1.2.2 Proper Insertion of a Minidiskette

Figure 3 illustrates the proper insertion of a minidiskette into a drive. Note the orientation of the slot for reading and the orientation of the write protect notch. The label of the inserted diskette must face toward the door opening mechanism.

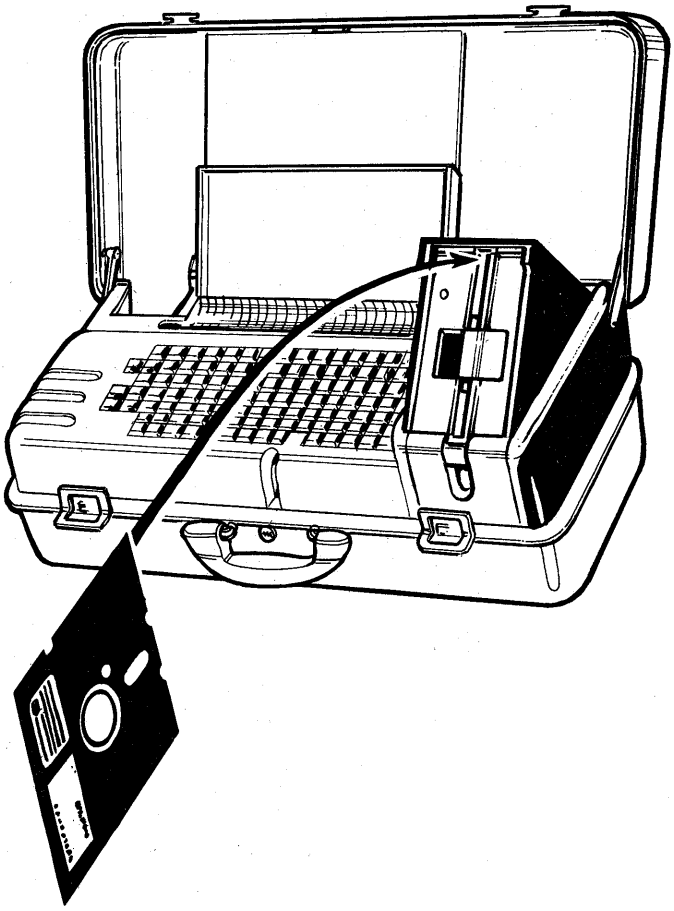

Figure 3. Inserting A Minidiskette

1.3 THE MICRAL V KEYBOARD

Figure 4 on the following page illustrates the MICRAL V keyboard. This is an upper case ASCII keyboard with an 18 key numeric pad for convenient entry of both decimal and hexadecimal numbers. A row of special function keys is also provided. The keyboard uses the standard 64 character upper case ASCII character set.

Most of the keys are self explanatory. The functions of a few special keys and the control keys are explained below the illustration.

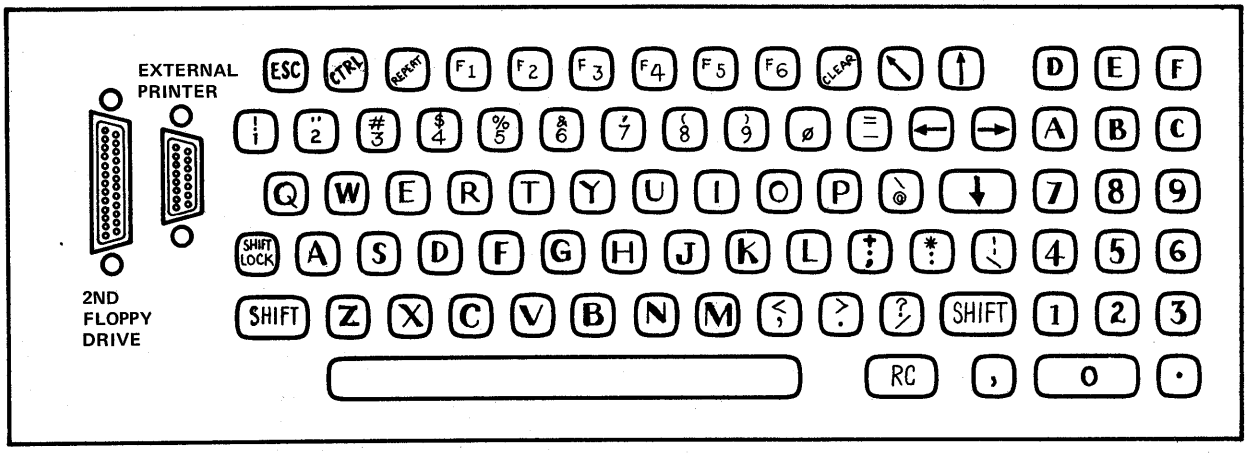

Figure 4. The MICRAL V Keyboard

CONTROL KEY	FUNCTION
ESC	Escape -- Used by BAL as an "Interruption" key to interrupt the current operation and return to some other point in the program.
CTRL	Simultaneously pressing CTRL and SHIFT clears the system, readying it for a system command. When CTRL-SHIFT is pressed the word "MICRAL.." and a blinking cursor appear in the upper left corner of the display.
REPEAT	Holding this key down while simultaneously pressing another key causes the repeated entry of that character or function.
F1-F6	These are function keys, whose state can be sensed under program control when using assembly language.
CLEAR	Clears the display screen and returns the cursor to the upper left corner position.
CARRIAGE RETURN	Used to terminate most keyboard entries. Provides both a carriage return (cursor moved to beginning of current line) and a line feed (cursor moved down one line) when pressed.

DOWN ARROW (LINE FEED)	Moves the cursor down one line, but does not return it to the beginning of the line.
UP, RIGHT & LEFT ARROWS	Moves cursor one character position in the direction indicated by the arrow. These keys are useful only in system dialog, not in response to a BAL ASK instruction.

1.4 POWER

The MICRAL V is generally powered from a 110 Vac, 60 Hz source. It can also operate on 220 Vac, 50 Hz; 12 Vdc; and 24 Vdc.

The computer has a short power cable with a 6 pin male connector. A separate mating power cable is provided for each power source. Each cable is wired to select the proper transformer taps for its input power source. The DC cables are designed to plug into a vehicle cigarette lighter.

To operate, plug the computer's power connector into the 6 pin connector on the appropriate power cable (the connectors are keyed to prevent incorrect insertion). Then plug the power cable into the appropriate power receptacle.

CHAPTER 2. OPERATING INSTRUCTIONS -- BAL

2.1 INTRODUCTION

This chapter contains complete descriptions of operating instructions for BAL, the Editor and the Utilities. Instructions for operation of the SYSMIC Operating System, the FORTRAN Compiler and the Assembler are found in Chapter 3.

2.2 TYPES OF DISKETTES

In the course of using the MICRAL V for various programs, you will use and/or generate the following types of diskettes containing programs and data:

1. System Diskettes -- The BAL System diskette contains the system programs and can contain your BAL source program and its translation. We suggest that you write protect the supplied System diskette and use it only for making copies of the operating system. You can then write and translate programs using the copies of your System diskette.

2. Source Program Diskette -- May contain edited copies of your source program, written in BAL. This source program must be re-written to the System diskette (starting on track 14) for translation.

3. File Diskettes -- These diskettes contain the files (as necessary) used by your user program(s). Each diskette may contain several files of the same or different types.

These different types of diskettes are discussed in more detail in following paragraphs.

2.3 BRINGING UP THE SYSTEM

Follow the steps below to power the system up and bring the BAL system on-line.

1. One end of the power cord has a 6 pin connector. Plug it in to the mating connector on the computer's power cable. Then plug the system power cord into the appropriate power source (110 Vac, 220 Vac, 12 Vdc, or 24 Vdc).

2. When the display has warmed up, the word "MICRAL.." and a blinking cursor will appear in the upper left corner of your display.

3. Insert the System diskette (normally will be write protected) into Unit 0 with the label side of the diskette facing the door closing mechanism and close the drive door. Respond to the display by typing B:0(cr). The computer then loads the BAL program.

4. Once the resident BAL routines are loaded, the system requests the current date and time. Respond as shown below. The user responses in

these and all examples in this manual are underlined.

YEAR.1978(cr) MONTH.08(cr) DAY.06(cr)
TIME.16(cr) MINUTE.06(cr) NOTE: Use 24 hour clock.

6. If you enter an illegal date or time (i.e., month 13), the display will return to YEAR.19. If you make a mistake, such as entering 06 when you really wanted 08, press the Escape (ESC) key to restart the operation.

7. When the above data is entered correctly, the system responds:

   BAL...

   It is now ready to accept BAL commands. The legal BAL commands and the paragraphs in which they are discussed in detail are listed below.

8. If using an external parallel line printer, make sure it is on-line and powered up.

BAL Command	Function	Section
R(cr)	RUN -- Begins execution of the translated program on the System diskette in Unit 0.	2.8
T(cr)	TRANSLATE -- Translates the source program on the diskette in Unit 0; recording the translated program on that same diskette.	2.6 & 2.7
U(cr)	UTILITIES -- Calls the utility program which includes the Editor.	2.9
D(cr)	DEBUG -- Loads the Debug package which is then used to trace program execution and debug the translated BAL program.	2.10
C(cr)	COMMAND -- Calls various utility programs such as Dump, Formatter, etc.	2.11

2.4 POWERING THE SYSTEM DOWN

To power down, remove any diskettes from the minifloppy drives and unplug the system from the power source.

2.5 COPYING AND FORMATTING

Before you begin to develop programs for the MICRAL V, you may wish to format your blank diskettes and make several copies of the System diskette. (We'd suggest reserving the supplied System diskette as a Master and making copies to use for all program development.).

See 2.5.2 for copying a diskette in a dual drive system; 2.5.3 for copying with a single drive system.

2.5.1 Formatting

Blank diskettes must be formatted prior to use, unless a complete formatted diskette is to be copied onto the blank diskette. The formatting operation initializes a diskette, programming it with the proper housekeeping data in the format required by BAL. Following this, it is ready for use with the system.

Format your diskettes as follows:

1. With the system operating (awaiting a BAL command), insert the System diskette into Unit 0 and close the door. Make sure that the diskette is write protected.

2. Type C(cr). This brings in the formatting program and the console display will be:

   COMMAND:

3. For safety, remove the System diskette from Unit 0 and insert a blank diskette (or a diskette to be re-formatted) into Unit 0 or Unit 1. Make sure that this diskette does not have a tab covering the write protect notch.

4. Respond to COMMAND by typing P(cr). You must then supply data for several conditions as indicated below.

   COMMAND :P(cr)
   OUTPUT :FLO(cr) UNIT :0 or 1(cr)

   This specifies that formatting is to occur on the floppy diskette (FLO) of Unit 0 or 1. Note that the System diskette will attempt to re-format itself if it is still in Unit 0 and you specify Unit 0.

5. The specified drive will step through all the tracks as the diskette is formatted. When formatting is complete (about a minute), a new message COMMAND :, appears on the display.

   If you note that the display does not change after a couple of minutes, check to be sure that the diskette is not write protected or try another diskette.

6. Repeat this operation from step 3 to format another diskette.

7. To terminate the operation, type M(cr) to return to BAL.

Once a diskette has been formatted and data has been recorded on it, you can erase the programs or data by copying new information on the diskette. It is generally not necessary to re-format a diskette.

2.5.2 Copying A Diskette - Dual Drive System

Prior to developing user programs, you will generally wish to make one or more copies of your System diskette. These will be used in developing and translating programs (see 2.7 for further information). Copy your System diskette as follows:

1. With the system operating, insert the System diskette into Unit 0.

2. Type C(cr). The copy program is loaded and the following is displayed on the screen:

   COMMAND:

3. Insert a blank diskette into Unit 1 and close the door. (The diskette need not have been previously formatted.) This diskette must not be write protected.

4. Respond to the display as follows:

   COMMAND :3(cr)

5. The drives will step through all the tracks as the information on the diskette in Unit 0 is copied onto the diskette in Unit 1. After the copying is complete (about 2 minutes), COMMAND : again appears on the screen.

   The diskette in Unit 1 is now a duplicate of the diskette in Unit 0, which is unchanged by this operation.

   If a read or write problem occurs, the system retries the operation. If the copy is not complete after 2-3 minutes, try another diskette.

6. To make another copy, repeat this procedure from step 3.

7. To terminate this operation, type M(cr) to return to BAL.

Note that the COMMAND function includes other utilities in addition to Copy. They are discussed in paragraph 2.11.

2.5.3 Copying A Diskette - Single Drive System

If you have a system with a single drive, you will need to write a short program to use in copying your diskettes. The steps you will need to use (programming in BAL) are:

1. Load the diskette to be copied (Diskette A).

2. Use the BAL BUFIN command to read the first n sectors of data into memory.

3. Remove Diskette A and load the diskette to be copied onto (Diskette B).

4. Use the BUFOUT command to output the first n sectors of data onto Diskette B.

5. Swap diskettes and follow the above procedure for the next n sectors. Continue until the entire diskette has been copied.

2.6 CREATING A SOURCE PROGRAM

MICRAL V allows the user to create a new source program and have it syntax-checked on a line-by-line basis as it is entered into the system. The BAL translator analyzes each instruction character-by-character and indicates any errors as soon as the next character has been entered.

When an error is found, the programmer can correct it immediately. See Appendix A for an explanation of the BAL error codes. See the MICRAL BAL Reference Manual for a complete description of the language.

Create a program as follows:

1. With the system running, insert a diskette containing the system programs into Unit 0.

2. After closing the drive doors, respond to the BAL prompt by typing T(cr). BAL...T is now on the screen.

3. The translator is loaded into memory and the following message is then displayed:

   NEW, OLD (N,O) :

4. To create a new program, type N(cr), then proceed to enter your BAL program from the keyboard according to the programming rules and formats defined in the MICRAL BAL Reference Manual.

   As the program is entered, it is analyzed and recorded on the system diskette, starting at track 14.

5. When you enter each ESEG statement (end of segment), the last source code for that segment is recorded on the diskette, the segment is translated, and the following is displayed for that segment:

   PROGRAM LENGTH XX (Will be 0 if this segment cannot be translated properly.)

   DATA LENGTH YY (Only at the end of Segment 0.)

6. When END is keyed as the last instruction of your program, the system completes the translation and returns to BAL, displaying the BAL... prompt.

7. You now have a system diskette in Unit 0 which contains the source program and the translated version of the program.

   If there are no errors, you can execute your translated program. If any errors are detected (one or more segments has a program length of 0), use the Editor to prepare a corrected Source program. See paragraph 2.9 for a description of the Editor.

2.7 TRANSLATING A PROGRAM

As you no doubt noted above, the Translator is used both to create a new program and to translate an existing program. The procedure for translating an existing program is:

1. With the system operating, insert a System diskette containing a source program (beginning at track 14) into Unit 0 and close the door. Note that this diskette may NOT have the write protect notch covered.

2. Respond to the BAL prompt by typing T(cr).

3. The Translator is loaded into memory, then several messages are displayed, requiring operator response as follows:

Message	Response	
NEW,OLD (N,O) :	N	This indicates a new program which is created as described in paragraph 2.6.
	O	An old program, already recorded on the System diskette, beginning at track 14. Additional information will now be requested.
	ESC	Return to BAL.
LIST (Y,N) :	Y	Yes; a program listing is to be output to the Display, to the strip printer, or to an external line printer.
	N	No; no listing required.
	ESC	Return to the New-Old question.
DBUG (Y,N) :	Y	Yes; the hexadecimal address of each line of the program is printed in parentheses on the listing along with the program source code.

If a listing is not selected, the hexadecimal addresses for lines with statement numbers will be printed in a table.

N No; no hex addresses are printed.

ESC Return to the New-Old question.

OUTPUT NUMBER

1 Output the program listing on Display panel.

2 Output listing on external parallel line printer.

5 Output listing on 32 column strip printer.

ESC Return to the New-Old question.

PART (Y,N) :

Y Yes; selects the partial translation of the program -- some number of segments to be translated.

The following additional information is required.

SEG NUMBER :N(cr) Respond to each request with the number of a segment to be translated. Respond with (cr) to begin the translation of the selected segments.

N No; translate the complete program. When segment 0 is modified, or if variables are changed, it is suggested that the complete program be re-translated.

ESC Return to the New-Old question.

4. The source program is read from track 14 of the diskette in drive 0 and translated, with the resultant program recorded back on the same diskette.

5. Follow each of the responses with a carriage return.

   As translation occurs, syntax errors may be found. When this occurs, the system indicates the error by displaying an error number on the Display and halting the listing output. (See the translation time error list in Appendix A.) The offending character will be enclosed in parentheses.

```
EXAMPLE:    25 DCL A, B2, CD
            *** (D)  ERROR 61   DBUG ADDRESS 0007
```

   You must press ESC to continue the translation.

6. You can stop the translation at any time by pressing the Escape (ESC)

key. When this occurs, program translation is stopped, and you can select one of the following options:

a. Continue the translation by again pressing ESC.

b. Change the conditions of the translation by pressing R(cr). You must then respond again to the LIST, DBUG, and OUTPUT NUMBER questions.

c. Abandon the translation by first pressing R, then responding to the list message by pressing ESC. When NEW,OLD appears on the display, again press ESC to return to BAL.

7. When the translation is complete, the system returns to BAL and displays BAL....

8. At this point your diskette contains both a source program and its executable translation (provided no fatal translation errors occured). The program can now be executed (see 2.8 below). Note that it is not necessary to re-translate a program after execution.

2.8 PROGRAM EXECUTION

Once your program has been translated, with no translation-time errors, that program can be executed. The steps are:

1. With the system operating, insert the translated diskette in Unit 0. If a file diskette is being used, insert it in Unit 1.

2. Close the doors and type R(cr).

3. The BAL run-time routines along with the translated program are loaded into memory and program execution begins.

4. During execution, if the program is semantically incorrect, a run-time error will occur. (See Appendix A for a list.) All run-time errors (except for file system errors controlled by the program) are fatal and cause execution to halt with the following message displayed.

   ERROR N IN SEGMENT X AT ADDRESS YY

   Where: N is the error number.
   X is the program segment number.
   YY is the Debug address within the segment.

If run-time errors are encountered, you may wish to run your program under Debug control to determine the problem. (See paragraph 2.10.)

2.9 EDITING A PROGRAM

To edit your program, you must provide:

1. The System diskette containing the source program, or a Source diskette.

   Note that, when it is created, your source program is written beginning on track 14 of the system diskette. Tracks 0-13 are used for system functions.

2. You may need a formatted diskette, either a blank or a diskette to be overwritten.

The steps for program editing are:

1. Insert the System diskette in Unit 0, close the door, and type U(cr).

2. The Utility package is loaded into memory, and the following message is displayed:

   FUNCTION :

3. Respond with E(cr) to call the editor.

4. Next respond to the following working messages, as shown:

```
INPUT  :FLO(cr)     UNIT  :0 or 1(cr)     TRACK  :YY(cr)
OUTPUT :FLO(cr)     UNIT  :0 or 1(cr)     TRACK  :ZZ(cr)
```

You can edit in any one of the following ways:

a. Edit from track 14 of the System diskette to some higher track on the same diskette (or vice versa). This method could be used for smaller programs. Note that each track of a diskette holds 4096 characters, counting blanks and all punctuation, including carriage return.

b. Edit from the System diskette to a Source diskette in Unit 1 (or vice versa). For example, edit from Unit 0, track 14 to Unit 1, track 0.

c. Edit from one source diskette to another (System diskette is removed once the Editor is read into memory).

Note that your edited source program must be written back onto track 14 of a System diskette before it can be translated.

7. The first instruction of the program is now displayed, along with a colon prompt, as shown below.

```
   PROGRAM "TEST"
:-
```

You must respond with one of the Editor commands. They are summarized below and explained in detail in the MICRAL BAL Reference Manual. An invalid command will be rejected by the system and will not appear on the display.

Command	Function
Space (press space bar)	Advance to next instruction.
M (modify)	Modify instruction presently displayed. The cursor is moved to the beginning of the instruction and you can begin to enter new characters, replacing the present characters. Pressing ESC at any point moves the cursor to the end of the instruction, retaining the characters passed over.
C (comment)	Similar to M, but moves the cursor to the end of the instruction, so it can be modified or comments can be added.
R (replace)	Delete the current instruction and replace it with one or more instructions which you now key in.
I (insert)	Insert one or more instructions following the instruction being displayed. This command is used to add new

lines to the source file. Pressing cr terminates each new line entered. Pressing cr before entering an instruction ends the insert function and the next sequential instruction in the source file is displayed.

Dn (delete) — Delete n lines (0 to 9), beginning with the line currently displayed.

+n — Advance the display down n instructions, where n is 1 to 9.

J(cr) — Insert source code from some other support device, defined by your response to the question "INPUT :" which is now displayed. This will generally be a source file from a diskette. The file to be input must terminate with an END statement (which will not be copied).

A — Advances to the end of file. "FILE END" is displayed.

,string(cr)
.string(cr)
* — Used to find the occurence of strings in text, beginning from the present line. The line containing the specified string is displayed.

,string -- Find the first occurrence of the specified string anywhere in text.
.string -- Find the first occurrence of the specified string which begins a line.
* -- Repeat previous string command.

Note that a string may be up to 15 characters long.

S(cr) — Abort the edit, cancelling all changes to the source program. The Utility requests the next FUNCTION.

E(cr) — Write the modified source code to the specified device. This command must be executed to properly terminate an edit and obtain the newly edited source code. This command can be entered at any point in the edit.

Note: Several keys are used for special functions in the Editor. They are:

Escape -- For the M and C commands, moves the cursor to the end of a line, retaining the information passed over.

Left Arrow -- Backs the cursor up one space without destroying the character passed over and allows retyping the previous character.

Right Arrow -- Steps the cursor one character to the right without

destroying a bypassed character.

@ -- Deletes the line being typed and returns the cursor to the beginning of the line for re-typing.

Note that the left and right arrows can be used in the M and C commands to make corrections within an instruction, then ESC can be pressed to move the cursor to the end of the line, retaining the information passed over.

8. When your edit is complete, type E(cr). The remainder of the original source program is copied into the new file, and the message "FUNCTION :" appears on the display.

9. You now have an edited Source program on the specified unit. You must copy the edited program back to track 14 of the System diskette prior to translation (if you were editing to some other track).

10. Type M(cr) to return to BAL. You now have an edited source program which must be translated before it can be executed.

2.10 THE DEBUG PACKAGE

Debug is a routine which allows you to execute your BAL program interactively. In normal execution of a BAL program, instructions are fetched, checked for run-time errors, and executed in sequence. Under Debug control, you can execute instructions in single step mode, insert breakpoints, examine and modify variables, etc. This is handy when you're trying to figure out why the program doesn't work the way you planned it.

Use Debug as follows:

1. With the system operating, insert a System diskette containing your translated program into Unit 0. If a file diskette is required by your program, insert it into Unit 1.

2. Type D(cr) in response to the BAL prompt. The Debug routine and your object program are loaded into memory and the following message is displayed:

   DBUG :

3. At this point, you must provide working instructions to the Debug program. A condensed table of these instructions is below. A more detailed description of these instructions can be found in the MICRAL BAL Reference Manual.

Command	Function
D variable(cr)	Displays the present contents of the selected variable.
M var=value(cr)	Replaces the contents of the selected variable

with "value".

B seg no,addr(cr)	A breakpoint is installed in program segment "seg no" at the specified address (addr). When instruction execution reaches that point in the program, control is returned to the user prior to executing the instruction. The breakpoint is removed once it has been reached.
A seg no,addr(cr)	A permanent breakpoint is installed at the specified address in "seg no". Program execution stops and control is returned to the user each time this instruction is reached. This condition is reset with the C command.
G(cr)	Continue program execution from the current point.
G address(cr)	Begin program execution from the indicated point. The address corresponds to one of the addresses printed on the source listing when the Debug option is selected during translation.
S(cr)	Selects single step instruction execution in which one instruction is executed each time the space bar is pressed.
C(cr)	Terminates single step mode and the permanent breakpoint settings.
Space	Pressing the space bar while in single step mode results in the execution of the next instruction in sequence.
E(cr)	Terminates execution of the program under Debug control and returns to BAL.

4. When a program is running under Debug, control is returned to the operator: 1) at the first instruction, 2) after each instruction in single step mode, 3) each time a breakpoint is reached, and 4) upon the detection of a run-time error. Control is also returned if you press ESC (if the program is hung up in a loop, for example).

2.11 THE UTILITY ROUTINES

There are two sets of utility routines available. They are described below.

2.11.1 Routines For Handling Source Files

These routines include the Text Editor and the Source Program Copy routine.

They are used as follows:

1. Insert your System diskette in Unit 0 and type U(cr).

2. The Utility package is loaded into memory and the system displays:

   FUNCTION :

3. Type E(cr) to call the Editor. The Editor is described in detail in paragraph 2.9.

4. Type S(cr) to call the Source Program Copy routine. Then respond to the display messages as shown below.

   a. To copy from one floppy disk to another

   ```
   INPUT  :FLO(cr)     UNIT  :0 or 1(cr)     TRACK  :0(cr)
   OUTPUT :FLO(cr)     UNIT  :0 or 1(cr)     TRACK  :0(cr)
   ```

   Naturally, you must have the proper diskettes in the drives for this operation. You can copy from a source diskette in Unit 0 to a formatted diskette in Unit 1 or vice versa.

   You will normally copy the source program from track 14 on one unit to track 14 on the other unit, because that's where the Translator expects to find it. You can copy to other tracks, but be careful that you don't overlay other valid data.

   b. To copy from floppy to the Display or Printer

   ```
   INPUT  :FLO(cr)     UNIT  :0 or 1(cr)     TRACK  :Y(cr)

   OUTPUT :STY(cr)                                  (To the display)
        or
   OUTPUT :IMP(cr)     OUTPUT NUMBER :Z(cr)   (To the printer)
   ```

   Where: Y = Track number (14 for source program)
   Z = 1 - Output to display
   2 - Output to external parallel printer
   5 - Output to strip printer

5. Type M(cr) to exit the Utility and return to BAL.

2.11.2 Routines For Handling Various Types of Files

These utilities include routines for copying files, dumping data and formatting.

1. With a System diskette in Unit 0, type C(cr)

2. The Command Utility is loaded into memory and the display is:

   COMMAND :

3. Respond with one of the following valid codes, after loading the required diskettes in the drives.

Code	Function
3(cr)	Copies the complete contents of the diskette in Unit 0 onto the diskette in Unit 1.
R(cr)	Copies the complete contents of one diskette onto another. You must supply system information as follows: INPUT :FLO(cr) UNIT :0 or 1(cr) OUTPUT :FLO(cr) UNIT :0 or 1(cr)
P(cr)	Formats a diskette. In response to command P, the system requires the following information: OUTPUT :FLO(cr) UNIT :0 or 1(cr) You can insert a diskette in either drive and format it using this command.
D(cr)	Dumps the selected sectors from a floppy to the Display or printer. To use this command, insert a diskette to be dumped into one drive and type D(cr). Then supply the system information as shown below. INPUT :FLO(cr) UNIT :X(cr) FIRST SECTOR :AA(cr) NB SECTORS :BB(cr) OUTPUT :STY(cr) (Dump to display) or OUTPUT :IMP(cr) OUTPUT NUMBER :Z(cr) (Dump to printer) Where: X = Unit 1 or 0 AA = Starting sector number BB = Number of sectors - decimal or hex (preceded by /) Z = 1 - Display 2 - External line printer 5 - Strip printer This command dumps the required number of sectors of data (BB) from Unit X, starting with sector AA onto the designated device. Data is dumped 16 characters per line, with the hexadecimal numbers on the left and their ASCII equivalents on the right. Unprintable ASCII characters are represented as a dot, ".".

Data is output sector by sector. If output is to the display, the system halts after each sector is output. Press any key (except ESC) to continue with the next sector. If you press ESC, the dump is aborted and you return to the COMMAND message.

L(cr) Free Format -- This command copies the specified files, no matter what format they are in, onto the specified output device.

The procedure for this command is identical with D, above.

M(cr) Return to BAL.

4. At any point, you can abandon your selection and return to "COMMAND" by pressing the Escape Key (ESC). The system will print ABORT, then return to COMMAND.

CHAPTER 3. OPERATING INSTRUCTIONS -- SYSMIC OPERATING SYSTEM (INCLUDES ASSEMBLER AND FORTRAN OPERATION)

3.1 INTRODUCTION

The MICRAL Assembler and the FORTRAN Compiler operate under control of the MICRAL Sysmic Operating System. This chapter provides a complete description of operating instructions for Sysmic, the Assembler and the FORT//80 FORTRAN Compiler.

For a complete description of the FORTRAN language, refer to the FORT//80 FORTRAN IV Language Manual.

Refer to the MICRAL Assembler Reference Manual for a description of Assembly Language Programming.

3.2 GENERAL DESCRIPTION

3.2.1 System Files

The Sysmic system is file oriented. The Sysmic diskette contains a number of system files, which are listed below and described in detail in following paragraphs.

/ - Requests a directory of files on the diskette.

UT - Utility Program. Includes routines to initialize a diskette and to create a catalog, create a volume, and create files.

CP - Copy. Copies from one device to another.

ED - Editor. Used to create and edit source files. Uses same commands as BAL Editor.

FTRN - FORTRAN Compiler. Compiles a FORTRAN source program.

ASM - Assembler. Produces an executable program from an assembly language source file.

In using the system, you must create and edit files as necessary, using the Utility and Editor programs. If you are creating source programs on a new diskette, you must use the UT program to initialize that diskette, create a volume and define files. This is described in detail in paragraph 3.5.

3.2.2 Definitions

Certain terminology is required in system commands, as follows:

Device Name	FL0 - Floppy Disk, Unit 0 FL1 - Floppy Disk, Unit 1 STY - Display ETY - Keyboard IMP - Printer
Output Device Codes	1 - Display Panel 2 - External Line Printer 5 - Strip Printer
Filename	All source files are referenced by filename. A filename can be a maximum of 6 characters.
Program name	An executable binary file is assigned a 6 character name of the same form as the source file.
File Specification (filespec)	Source files and executable binary files are specified differently in a command string. When used in a command string, a Source file is specified in the following general format: [Device name].filename When used in a command string, a Binary file is specified as: [device name.]filename The brackets indicate that the device name is optional. (If the device name is omitted, FL0 is assumed.) Note: that the period is required when a source file is specified; it is optional for a binary file.

3.2.3 General Command Syntax

The general form of a Sysmic command is:

[device name.]Program name, P1, P2,... Pn(cr)

Device name and program name are as defined above. P1 thru Pn are parameters which must be specified in the various command strings. They are discussed in detail in the individual command descriptions.

Note the following:

* Spaces may be inserted in the commands as desired to provide better readability.

* Numeric parameters are normally expressed in decimal notation. Hexadecimal notation can be used by preceding the hex number with a slash "/".

* Normally a complete command string is typed, followed by a carriage return. The system program is then loaded, and executed. If you wish, you can load the system program without executing it, change diskettes, then finish typing and executing the command. For example, the command "CP (cr) FL0,FL1 (cr)" will load the Copy routine from the system diskette, then halt. You can replace the system diskette with some other diskette to be copied from Unit 0 to 1, then enter the rest of the command and type carriage return. The copy will then occur.

3.3 LOADING THE SYSTEM

1. First plug the power cord into the appropriate receptacle. When the display has warmed up, the word "MICRAL.." and a blinking cursor appear in the upper left corner of the display.

2. Insert a Sysmic diskette in Unit 0, close the door, and type B:0,1. The system will be loaded and a double equal sign prompt (==) will be displayed. You can then enter one of the system commands, as described in following paragraphs.

3.4 DISPLAY DIRECTORY

This function prints (on the specified device) a directory of the files on the specified diskette.

Command format: [device1.] /n [,device2], L

Where:
- device1 - Specifies unit containing System diskette (0 is assumed if this parameter is omitted.)
- /n - / is the name of the directory display program; n specifies the number of lines on the display (12 on Micral V)
- device2 - Specifies optional output device. If this parameter is omitted, the display is assumed.
- L - Specifies print device. 1 - display; 2 - parallel printer; 5 - strip printer.

The directory is output in the following format:

```
.filename     T     P:xx     L:xx
```

Where:
- .filename - Specifies name of the file.
- T - Indicates one of three file types:
  - S - source
  - B - binary
  - L - free-format
- P:xx - Specifies the starting track of the file (in decimal)
- L:xx - Specifies the length of the file in sectors of 256 bytes (decimal)

3.5 UTILITY PROGRAM

The Utility program includes several functions: initialize a diskette, initialize a file, create a volume (directory), create a file, and delete a file.

3.5.1 Initialize A Diskette

This function is used to prepare a new diskette for use in the system. The operating system requires the use of part of the diskette for housekeeping data. This function writes the diskette into the proper format.

Command format: [device1.] UT,PM,device2(cr)

Where:

[device1.]UT	- Specifies the Utility program.
PM	- Specifies the initialization function.
device2	- Specifies the unit containing the diskette to be initialized.

3.5.2 Initialize A File

This function is used to erase a specified file and initialize the space reserved for that file.

Command format: [device1.] UT,PM,filespec(cr)

Where: [device1.] UT,PM have the same function as in 3.5.1, above.
filespec specifies the file to be initialized, such as FL0.TEST.

3.5.3 Create A Volume

This function is used to assign a diskette an identifying volume name and to create a directory. This function is necessary for the diskette to be accessable by Sysmic, and must be done prior to assigning file names.

Command format: [device1.] UT,CT,device2,vol-name[,nmax](cr)

Where:

[device1.] UT	- Specifies the Utility program.
CT	- Specifies "volume creation"
device2	- Specifies the unit containing the diskette being operated upon.
vol-name	- The name of the volume, 1-8 characters in length.
nmax	- Optional specification of the maximum number of files which may be contained on this diskette.

3.5.4 Declare (Create) A File

This function permits the user to define various named files on a diskette. A file must be defined before any data can be written into it. Note that files

are defined by assigning each a starting track and some length. A file may not be defined to overlap an area reserved for some other file.

Command format: [device1.] UT,CR,filespec,type,track[,length](cr)

Where:	[device1.] UT	- Specifies the Utility program.
	CR	- Specifies "create file"
	filespec	- The name of the new file, such as FL0.TEST or just .TEST
	type	- Defines file type as: S (source), B (binary), or L (free-format)
	track	- Beginning track on the diskette.
	length	- Optionally specifies the file length in segments of 256 bytes. If this parameter is omitted, the length is specified as 0.

When data is entered into a file (using the Editor for example), Sysmic will expand the file as necessary and assign the correct length (if greater than the defined length), provided that there is space available. For example, assume a file ".TEST" of length 0 declared on track 2 and a file ".NEXT" declared starting on track 4. When data is written into .TEST it can use up to 32 segments (2 tracks). If entry of data beyond this amount is attempted, .TEST would be expanded into the area reserved for .NEXT. Sysmic will not allow this, so an error message would be issued.

3.5.5 Delete A File

This function is used to delete a file from the directory.

Command format: [device1.] UT,SP,filespec(cr)

Where:	SP	- Specifies "delete file".
	filespec	- Specifies the file to be deleted.

This function deletes the filename from the directory, but does not affect the data itself. Thus you could use another file declaration (CR) to assign a new filename to the diskette area containing the data of the deleted file.

3.6 COPY PROGRAM

The copy routine allows the copying or listing of a single file or an entire diskette.

3.6.1 Copy A Diskette

Command format: [device1.] CP,unit-x,unit-y(cr)

Where:	[device1.] CP	- Specifies the copy program.
	unit-x	- Specifies the source device, such as FL0.
	unit-y	- Specifies the destination device, such as FL1.

This function copies the entire contents of the diskette in unit x onto the diskette in unit y. The destination diskette does not have to be previously formatted. The diskette in unit x is unchanged by this operation.

3.6.2 Copy A File

Command format:

```
                                (filespec1) [,(filespec2)[,type] ]
                [device1.] CP,     or       [    or             ](cr)
                                (device-x)  [,(device-y) [,type] ]
```

Where:

[device1.] CP	- Specifies the copy program.
filespec1	- Specifies the source file, such as FL0.ED.
filespec2	- Specifies the destination file, such as FL1.ED.
type	- Specifies type of file: source, binary, free.
device-x	- Used instead of filespec1 to specify a source device. In MICRAL V can be ETY (keyboard).
device-y	- Used instead of filespec2 to specify a destination device. Can be STY (display) or IMP (printer).

When the destination filename or device is omitted, the specified file is written to the display panel.

3.7 EDITOR

The MICRAL Editor allows you to create and/or modify source files which are cataloged on a diskette. The basic Editor commands are identical for both BAL and Sysmic and are described in paragraph 2.9.

Command format: [device1.] ED, filespec1 [,filespec2] [,S](cr)

Where:
- [device1.] ED - Specifies the editor.
- filespec1 - Specifies a source file which is to be created or edited.
- filespec2 - Specifies a temporary file for the edited program. It must be of the same type as source file.
- S - Specifies suppression of the second phase of the edit (see explanation below).

The Editor is executed in two phases. In phase one, each instruction from file 1 is presented to the user for editing, then stored in file 2. When all modifications have been made, the second phase of the editor recopies the edited program from file 2 back to file 1. If the optional S parameter was selected, the second phase does not occur and file 2 becomes the destination file.

If you do not specify the filespec for file 2, the system creates a temporary file with the same name as the source file, plus the suffix .S1. This file is created and cataloged on the same diskette as the source file.

In addition to the BASIC Editor commands described in paragraph 2.9, the following Editor commands are used in Sysmic:

@EOF Required to flag the end of a Sysmic source file. For example, it would be used as follows when creating a FORTRAN source file:

```
100    CONTINUE(cr)
       END 10H(cr)
       EOF(cr)
```

D.string Delete all text starting from the current line up to the first instruction beginning with the specified string (blanks are ignored).

D,string Similar to the above instruction, but deletes all text up to the first occurence of the specified string anywhere in text.

P1 Selects a "CARD FORMAT" for an Assembly Language program. Allows the use of the tabulation key (\ or ESC) with Sysmic. This command can be selected at any time during the entry of the Assembly program (at the beginning of the instruction line).

P2 Selects a "CARD FORMAT" for a FORTRAN program, allowing use of the tabulation key with Sysmic. This allows the protection of the "label field" (columns 2-5). This

command can be selected at any time during the entry of the FORTRAN program, but only at the beginning of an instruction.

J:filename — Insert the specified file. (This is only a syntax change to the basic Editor's J command.)

* — This repeats the previous string command, searching for a string if .string or ,string were specified; deleting text if D.string or D,string were specified.

3.8 FORTRAN COMPILER

The FORTRAN Compiler can be used to compile your FORTRAN source programs and produce executable binary files.

Command format: [device1.] FTRN,filespec,.B1[,NL][,NB][,LC][,LV](cr)

Where:
- [device1.] FTRN - Specifies the FORTRAN Compiler.
- filespec - Specifies a FORTRAN source file. See the FORT//80 Reference Manual for a description of FORTRAN programming requirements.
- B1 - Specifies creation of a binary file.
- NL - Specifies no listing.
- NB - Specifies no binary file.
- LC - Create listing and binary file.
- LV - Create listing on display or printer.

3.9 ASSEMBLER

The MICRAL Assembler runs under control of the Sysmic Operating System. Refer to the MICRAL Assembler Reference Manual for programming details.

Command format:

[device1.] ASM,filespec1 ,filespec2 [,NL] [,NS] [,NB] [,LV] (cr)

Where:
- [device1.] ASM - Specifies the Assembler program.
- filespec1 - Specifies the source file.
- filespec2 - Specifies the name of the binary file.
- NL - Specifies no listing.
- NS - Specifies no symbol table output
- NB - Specifies no binary file to be generated.
- LV - Listing to be output on the display screen.

Unless otherwise specified, the system generates a binary file and prints a listing and a symbol table.

In addition to the standard Assembly langauge commands, the following are used

when operating under control of Sysmic:

PAR	Pseudo instruction (no operand required) which causes ASCII characters to be generated with even parity. If this pseudo instruction is not specified, 7 bit ASCII characters are generated with the parity bit always set to zero.
END ETIQ	Pseudo instruction which requires an operand specifying the beginning address of the program.
DCN val	Pseudo instruction DCN which may require a one or two byte value.

3.10 MONITOR

The MICRAL MOMIC Monitor can be called when operating under Sysmic.

Command format: [device1.] MOMIC (cr)

The operation of MOMIC is described in detail in the MICRAL MOMIC Reference Manual.

APPENDIX A. MICRAL BAL ERROR LIST

A.1 GENERAL

When the translator detects an error, either during translation or the execution of a program, a message is displayed on the screen in the following format:

ERROR N IN SEGMENT XX AT ADDRESS YYYY

Where: N -- The error code as listed below.

XX -- The program segment which was being executed or translated.

YYYY -- The debug address of the instruction which is in error.

Note: The list of errors below is inclusive for all versions of BAL. Certain of these error codes are not applicable to the MICRAL V and will never appear on your display.

A.2 ERRORS FOUND DURING EXECUTION OF A PROGRAM

Error Code	Explanation
10	Specified segment not in user program
11	Data field is too small -- attempt to read non-existent data
12	Index is zero
13	Index is too large for the declared table size
14	Index is non-binary (table or string)
15	RETURN point in a program is incorrect
16	Overflow of a variable
17	The stack of the GOSUB or LDGO.SEG has overflowed
18	Arithmetic overflow
19	Undefined variable
20	Variable is not in floating point format (BCD)
21	Load segment (LDGO.SEG X) is incorrect
22	BUFIN or BUFOUT error -- BUFIN & BUFOUT applicable only to hard disk & floppy
23	Incorrect peripheral number specified (ASK or PRINT)
24	Format error
25	Common memory overflow
26	Lock number too large
27	Memory variable too large
28	Individual memory overflow
29	Common variable declaration incorrect
30	Common variable table overflow
31	Undefined operation code
32	Attempt to write a protected common variable

A.3 ERRORS FOUND DURING TRANSLATION OF A PROGRAM

Error Code	Explanation
50	Keyword incorrect (READ, PRINT, GOTO, etc.)
51	Error in the form of instruction (Incorr. label)
52	Value specified is not binary
53	Number of segments incorrect
54	The end of the instruction is incorrect
55	FOR/NEXT loop is incorrect
56	Syntax error
57	Incorrect operator (+, -, *, etc.)
58	Type of variable incorrect (string, BCD, binary)
59	Format error
60	Support variable incorrect (FIELD, EQUIVALENCE)
61	DCL (declaration) error
62	Binary code generated for this instruction is too large; instruction must be shortened
63	String is incorrect
64	BCD is incorrect
65	Overflow of stack for the nested FOR statement; number of nested loops must be reduced
66	Individual memory overflow
67	The area selected for writing already in use for a BAL program

A.4 ERRORS FOUND WHEN USING PERIPHERALS

Error Code	Explanation
01	The disk is not ready
02	Read or write error
03	Track positioning error
04	Parameter call error
05	Peripheral is write protected
06	Disk hardware error
08	File is not in binary format
40	Specified peripheral not in the system
41	Location of the volume unknown
44	Volume does not belong to the file system

NOTE: Certain errors do not result in the immediate abortion of the operation. In these cases, the system retries the operation and sounds a beeping tone to alert the operator of the retry. You may abandon these retry attempts by pressing the ESC key. This procedure is applicable to errors 1,2,3,5,6 and 25.

APPENDIX B. MICRAL FILE SYSTEM RESPONSES

B.1 TABLE OF RESPONSES

Hex	Decimal	Explanation
/0	0	Request correctly executed
/1	1	Peripheral not ready
/2	2	Read or write error
/3	3	Track positioning error
/4	4	Incorrect parameter call
/5	5	Peripheral is write protected
/6	6	Peripheral hardware error
/7	7	Start of file detected (tape)
/8	8	End of file detected (tape)
/9	9	Peripheral busy
/40	64	Peripheral or file system resource non-existent
/41	65	Local volume undefined
/42	66	Function undefined
/43	67	Volume unchangeable because it still contains files
/44	68	Volume does not belong to file system
/45	69	Volume not mounted
/46	70	Name of the file does not exist on the volume
/47	71	Incorrect file opening key
/48	72	File open
/49	73	Incorrect logical number
/50	80	Logical number not assigned
/51	81	Relative file not open for writing
/52	82	File name already exists on this volume
/53	83	The volume cannot contain a new file
/54	84	Unknown disk address
/55	85	Volume overflow
/56	86	Incoherant file
/57	87	Logical number already assigned
/58	88	File system cannot accept opening of a new file
/59	89	The volume is used, therefore cannot be replaced by another
/60	96	Incorrect file type
/61	97	File not open for writing
/62	98	Sequential file not open for writing
/63	99	Beginning of file
/64	100	End of file
/65	101	Loss of information on reading; your input buffer contains only the beginning of the recorded data
/66	102	Length of the key is incorrect
/67	103	Article found, but does not appear in any of the indexes requested
/68	104	Article not found
/69	105	No index
/70	112	Article already exists
/71	113	Article in course of modification

/72	114	Article not authorized for modification
/73	115	Writing or destruction of file not permitted because that file is not the last on the tape
/74	116	Attempt to open a file while it is in safeguard

APPENDIX C. SYSMIC ERROR CODES

01	Peripheral not ready (or floppy disk write protected)
02	Read/write error
03	Track positioning error
04	Incorrect parameter
05	Disk is write protected
06	Peripheral controller error
08	File not in binary format
20	Command syntax incorrect
21	Peripheral unknown
22	No directory found
23	Directory full
24	Specified disk area assigned to another file
25	Specified filename already exists
26	File already open
27	Too many files open simultaneously
28	File unknown
29	File number is incorrect
2A	Specified file type is incorrect
2B	Device name or filename syntax is incorrect

USER RESPONSE FORM

MICRAL V MICROCOMPUTER OPERATOR'S MANUAL

Please use this form to request future updates and/or to record your comments, suggestions, etc. concerning this document. Return to:

R2E of AMERICA
47 Bedford Street, S.E.
Minneapolis, Minn. 55414

DATE: _____

NAME: _____

COMPANY: _____

ADDRESS: _____

CITY: _____ STATE: _____ ZIP: _____

COMMENTS: