

ANZIO

Personal Computer Communications

USER MANUAL

Version 11.0 - January 1998

Copyright © 1987-1998 by Robert Rasmussen
All Rights Reserved



RASMUSSEN SOFTWARE, INC.

10240 SW Nimbus Ave., Suite L9, Portland, OR 97223 USA
(503) 624-0360

Copyright Notice

This software package and users manual are Copyright © 1987 - 1998 by Robert Rasmussen. All rights are reserved, worldwide. No part of this manual may be reproduced, transmitted, transcribed, or translated in any manner without the express written consent of Robert Rasmussen and Rasmussen Software, Inc., 10240 SW Nimbus Ave., Suite L9, Portland Oregon 97223.

The information contained within this document is subject to change without notice. No warranty of any kind is held in regards to this material.

Single PC License Agreement

The ANZIO product is sublicensed (not sold) to the original purchaser by Rasmussen Software, Inc., granting purchaser the right to use this software on one and only one personal computer. Backup copies of the software may be made solely for backup purposes in order to protect your investment.

The original nor any of the backup copies shall be sold, leased, gifted or otherwise be transferred or used by any other party, other than the purchaser, or on more than one machine at any given time. Purchaser may not under any circumstance tamper with any copyright protection scheme in use on the ANZIO distribution media.

The liability of Robert Rasmussen and Rasmussen Software, Inc. shall not exceed the cost of the software. Neither Robert Rasmussen nor Rasmussen Software, Inc. can be held responsible for any damages due to loss of data that may result in loss of dollars.

Acknowledgments

Windows is a registered trademark of Microsoft Corporation
Other product names are trademarks of their respective owners.

Printing History

Edition 8.6r March 1988
Edition 9.0 November 1988
Edition 9.0 Second printing December 1989
Edition 9.5 February 1990
Edition 9.7 March 1993
Edition 9.8 June 1993
Edition 10.9 July 1996
Edition 11.0 November 1996

Warning: This manual is undergoing revision in order to better describe differences between the various kinds of ANZIO, and their respective operating environments. In the meantime, warning messages will indicate when certain sections apply only to DOS, Windows, serial communication, network communication, etc.

i. TABLE OF CONTENTS

i. TABLE OF CONTENTSi

ii. INTRODUCTION 1

 ii.1 KEY FEATURES2

 ii.2 HOW THIS MANUAL IS ORGANIZED 4

PART I. NARRATIVE..... 6

1. USING ANZIO.....7

 1.1 INSTALLING ANZIO.....7

 1.1.1 INSTALLING THE FILES TO THE PC7

 1.1.2 INSTALLING THE TRANSFER PROGRAMS..... 8

 1.2 HOOKING UP THE PC 8

 1.2.1 WIRING THE CABLE9

 1.3 STARTING THE PROGRAM..... 11

 1.4 GETTING YOUR PARAMETERS STRAIGHT 11

 1.4.1 OPERATING SYSTEM.....12

 1.4.2 PORT.....12

 1.4.3 BAUD RATE12

 1.4.4 TERMINAL TYPE12

 1.5 GETTING ACQUAINTED 13

 1.6 LOGGING ON.....13

 1.7 USER PREFERENCE ITEMS.....15

 1.8 TERMINATING THE PROGRAM.....15

 1.9 MEMORY-RESIDENT OPERATION.....16

 1.9.1 RUN.....16

 1.9.2 STAY.....17

 1.9.3 MEMORY USAGE.....18

 1.10 CONFIGURING YOUR HOST SYSTEM.....18

 1.11 TERMINAL TYPES.....19





 1.11.1 DEALING WITH FUNCTION KEYS.....20

 1.11.2 USING A TERM TYPE OF "ANZIO"21

2. HELP WHEN YOU NEED IT 23

 2.1 THE **[F5]** HELP KEY AND SCREEN.....23


 2.2 THE HELP INDEX AND **[ALT] [H]**.....23

3. FUNCTION KEYS AND THE KEYBOARD	24
4. TALKING WITH ANZIO	26
4.1 ENTERING COMMANDS	26
4.2 EDITING COMMANDS	27
5. DEFINED KEYS (MACROS)	29
5.1 WHAT KEYS YOU CAN USE	29
5.2 DEFINING MACROS	30
5.3 EDITING MACROS	30
5.4 EXECUTING MACROS	31
5.5 OVERDEFINING SYSTEM FUNCTION KEYS	31
5.6 BLOCKING KEYS	31
5.7 SAVING AND RELOADING MACROS	32
5.8 SPECIAL FUNCTIONS INSIDE DEFINED KEYS	33
5.8.1 TABS	33
5.8.2 USING '↑' FOR 	33
5.8.3 PAUSES	34
5.8.4 LOCALLY DISPLAYED TEXT	34
5.8.5 EMBEDDED FUNCTIONS	35
5.8.5.1 EMBEDDING FUNCTIONS WITH 	36
5.8.5.2 LINE EDITOR WITH 	36
5.8.5.3 KEYSTROKES FOR FUNCTIONS	36
5.8.5.4 PREFIXING WITH '~'	37
5.8.5.5 STACKING FUNCTIONS	37
5.9 NESTING DEFINED KEYS	38
6. THE  KEY	39
6.1 USE AS A FREESTANDING OPERATION	39
6.2 IN A DEFINED KEY	40
6.3 INVOKED FROM THE HOST	41
6.4 THE EDITING KEYS	41
6.5 A SPECIAL USE: COMMAND LINE EDITING	43
7. USING A MODEM	44
7.1 THE DIAL COMMAND	44
7.1.1 MODEM RESPONSE CODES	45
7.2 THE WAIT COMMAND	46
7.3 SIGNING OFF	46
7.4 BAUD RATE SHIFT	47
7.5 FLOW CONTROL	48

8. HOW MANY CHARACTERS CAN I SEE?	49
8.1 SPECIAL SCREEN HARDWARE.....	49
8.2 132-COLUMN VIRTUAL SCREEN.....	50
9. PRINTING	52
9.1 HOW TO PRINT	52
9.1.1 PRINTING THE SCREEN	52
9.1.2 CAPTURE TO PRINTER	53
9.1.3 PASSTHROUGH PRINT.....	53
9.1.4 FILE TRANSFER TO PRINTER	53
9.2 THE WPRN PRINTER MODULE.....	53
9.2.1 PRINTER SETUP	54
9.2.2 CHOOSING A PRINTER	54
9.2.3 CHOOSING A FONT	54
9.2.4 THE FLUSH TIMER	55
9.2.5 LOW-LEVEL PRINT.....	55
9.2.6 THE PRINT WIZARD.....	56
9.2.6.1 AUTOMATIC TEXT HANDLING.....	56
9.2.6.2 PRINT WIZARD'S MARKUP LANGUAGE	56
10. USING REVIEW	57
PART II. DATA CAPTURE / FILE TRANSFER	59
11. DATA CAPTURE/FILE TRANSFER	60
11.1 DATA CAPTURE	60
11.1.1 ANZIO'S PICK COMMAND.....	60
11.1.2 THE KEEP COMMAND	60
11.1.3 PRINTING FROM THE SCREEN	61
11.2 FILE TRANSFER IN GENERAL	61
11.3 IMOS II FILE TRANSFER	62
11.4 IMOS III, IMOS V, IRX, ITX	62
11.4.1 I-SYSTEM SIMPLE UPLOAD.....	62
11.4.2 SEND-PC: I-SYSTEM DOWNLOAD.....	63
11.4.3 RECV-PC: I-SYSTEM UPLOAD.....	66
11.4.4 RECV-SPL: I-SYSTEM UPLOAD SPOOL FILE	67
11.4.5 KERMIT ON ITX	69
11.5 UNIX FILE TRANSFER	70
11.5.1 UNIX SIMPLE UPLOAD	70
11.5.2 SEND-PC.C: UNIX DOWNLOAD.....	71

11.5.3 DOWNLOAD: UNIX DOWNLOAD USING SHELL SCRIPT	71
11.5.4 RECV-PC.C: UNIX UPLOAD.....	72
11.5.5 KERMIT WITH UNIX	72
11.6 RM/COS.....	73
11.6.1 RM/COS SIMPLE UPLOAD.....	73
11.6.2 SEND-RM: RM/COS DOWNLOAD.....	74
11.6.3 SEND-L.RM: RM/COS LONG DOWNLOAD.....	75
11.6.4 RECV-PC.RM: RM/COS UPLOAD	75
11.7 VRX	76
11.8 PC-TO-PC.....	76
11.9 OTHER KINDS OF FILE TRANSFER.....	77
11.9.1 QUICKSEND	77
11.9.2 UFT	78
PART III. REFERENCE GUIDE	79
12. COMMANDS.....	80
12.1 COMMANDS BY TYPE.....	80
12.1.1 OPERATOR PREFERENCE ITEMS	80
12.1.2 COMMUNICATION PARAMETERS	80
12.1.3 FILE TRANSFER	81
12.1.4 LOCAL PROCESSING.....	82
12.2 COMMANDS ALPHABETICALLY	84
7E1	84
7E2	84
7N1.....	84
7N2.....	84
7O1.....	84
7O2.....	85
8E1	85
8E2	85
8N1.....	85
8N2.....	85
8O1.....	85
8O2.....	85
ANSWERBACK <string>.....	85
AUTO-LF [ON] AUTO-LF OFF.....	85
BACKSPACE 8 BACKSPACE 127.....	86
BAUD <nnnn>	86
BEEP [ON] BEEP SLOW BEEP OFF	86

BEEP IDLE [ON] BEEP IDLE OFF	86
BOX <x1> <x2> <y1> <y2> [<type>].....	86
BREAK.....	86
CALC	87
CALL <macro>	87
CAPTURE [ON] CAPTURE LONG CAPTURE OFF.....	87
CD [<unit>:]<dirname>.....	88
CHARSET <name>.....	88
CHOOSEPRINT.....	88
CLIP [<x1> <x2> <y1> <y2>].....	88
CLOSEI.....	88
CLOSEI/S.....	88
CLOSEO.....	89
CLOSEO/S.....	89
COLOR.....	89
COMMTYPE SERIAL COMMTYPE WINSOCK.....	89
COPY <filename> <newfile>.....	90
COPY/S <filename> <newfile>.....	90
CURSOR [BLINK] [ON] CURSOR [BLINK] OFF.....	90
DATA [BITS] 5 DATA [BITS] 6 DATA [BITS] 7 DATA [BITS] 8	90
DATE	90
DEFAULTS.....	90
DEFINE <x> <text>.....	90
DELAY <nnn>	91
DELAY/S	91
DELETE <filename>.....	91
DELETE/N <filename>.....	91
DELETE/S <filename>	91
DIAL <string> [<wait> [<retries>]].....	91
DIR [<pathname>].....	92
DIR/S [<pathname>]	92
DROPOUT	92
END.....	92
EJECT	92
ENV/S <variable>	92
F2	92
FILL <x1> <x2> <y1> <y2> <char>.....	93
FIND/S <filespec>.....	93
FINDNEXT/S.....	93
FLUSH	93

FLUSHTIMER <value>	93
FONT <size> FONT LARGER FONT SMALLER.....	93
FULL [DUP] HALF [DUP].....	94
GAUGE [ON] GAUGE OFF	94
HALF [DUP]	94
HELP	94
HELP <keyword>.....	94
HOLD [ON] HOLD OFF HOLD TOGGLE	95
HOSTNAME/S.....	95
HOTKEY <xxy>	95
IGNULL [ON] IGNULL OFF	95
IMOS	95
INTERPRET	96
INVOKE <macro>	96
IRQ <n>.....	96
ITX IRX IMOS RMCOS VRX UNIX.....	96
JUMP OFF JUMP MEDIUM JUMP FAST	96
KCOMMAND <command> [<parameters>].....	97
KEEP [<x1> <x2> <y1> <y2>].....	97
KEEP/N [<x1> <x2> <y1> <y2>].....	98
KEYS.....	98
KRECEIVE [<filespec> [AS <filespec>]]	98
KSEND <filespec> [AS <unixfilespec>].....	98
LAUNCH <program> [<parameters>]	99
LINE [DELAY] <nnn>.....	99
LOCK [ON] LOCK OFF 	99
LOG [<unit>:]<directory>.....	100
MENUBAR <x1> <x2> <y1> <y2> <x3> <x4> <off> <len> [<cols>].....	100
MERGE <filename>.....	100
MESSAGE <string>.....	100
MKDIR <dirname>	100
MKDIR/S <dirname>	100
MODE-132 <xx>.....	101
MONITOR [ON] MONITOR OFF.....	101
OPENI <filename>	101
OPENI/S <filename>.....	101
OPENO <filename>	102
OPENO/N <filename>.....	102
OPENO/S <filename>	102
PAN <n> PAN LEFT PAN RIGHT.....	103

PARITY EVEN PARITY ODD PARITY OFF	103
PASTE.....	103
PICK <x1> <x2> <y1> <y2> <type> [...]	103
PITCH <n>.....	103
PLAY NCR [ON] PLAY NCR OFF.....	103
PLAYSOUND <filename>.....	104
PORT <n>	104
PRINT	104
PRINT <x1> <x2> <y1> <y2>.....	104
PRINT/N <x1> <x2> <y1> <y2>.....	104
PRINTER <printer-name>.....	104
PRINTER-SETUP.....	105
PRINTER-SETUP <text>	105
PRINTFONT <SIZE>	106
PRINTFILE <FILENAME>.....	106
PRINTLOW [ON] OFF	106
PURGE.....	106
READ <filename>	107
RECEIVE QUIET [ON] RECEIVE QUIET OFF	107
RECEIVE CODED.....	107
RECONNECT [ON] RECONNECT OFF.....	107
RENAME <oldname> <newname>.....	107
RESET.....	107
RETRANSMIT.....	107
REVIEW	108
RMCOS	108
RTS-MODE 0 RTS-MODE 1 RTS-MODE 2.....	108
RUN [<program> [<parameters>]].....	108
RUN/N [<program> [<parameters>]].....	109
SAVE [<filename>].....	109
SCREENMODE <string>.....	110
SCREENMODE/S.....	110
SCROLL [ON] SCROLL OFF	111
SCROLL-LOCK [ON] SCROLL OFF	111
SEND <x1> <x2> <y1> <y2>.....	111
SETCOLOR <n> SETCOLOR NORMAL.....	111
SLEEP <hh> <mm> <ss>.....	111
STATUS [LINE] [ON] STATUS [LINE] OFF	111
STAY.....	111
STAY/G.....	112
STOP	112

STOP [BITS] 1 STOP [BITS] 1.5 STOP [BITS] 2	112
SYNC [ON] SYNC OFF SYNC FAST.....	112
TAB <i> <j> <k>	112
TAB CHARACTER <x>.....	113
TAB [ON] TAB OFF.....	113
TERM <termtype>	113
TERMNAME <name>	114
TIME	114
TIMEOUT <n>.....	114
TITLE <string>	114
TRACK-WINDOW [ON] TRACK-WINDOW OFF	114
TRANSMIT [ON] TRANSMIT OFF	115
TRANSMIT CRC.....	115
TRANSMIT LONG.....	115
TRANSMIT SINGLE.....	115
TRANSMIT TRAILER <string> TRANSMIT LONG TRAILER <string>.....	115
TTY	115
TYPE <filename>.....	115
UPPERCASE [ON] UPPERCASE OFF UPPERCASE TOGGLE.....	116
UNIX	116
VERSION.....	116
VERSION/S	116
VRX.....	116
WAIT <nnn>	116
WAITFOR <string> [<timeout>]	116
WF <string> [<timeout>]	117
WIDTH 132 WIDTH 80.....	117
WINDOW <x1> <x2> <y1> <y2> [FILL] [BOX DOUBLE]	117
WINDOWCLOSE.....	117
WINPRINT <filename>	117
WINSTART <filename>	118
WRITE <text>	118
XN	118
ZRECEIVE [<filename>].....	118
ZSEND [-a] <filename>	118
13. MORE ON STARTING ANZIO	119
13.1 PATHS AND SUBDIRECTORIES.....	119

13.2 COMMAND LINE PARAMETERS & DEFAULT	
FILES	119
13.2.1 PARAMETER FILE NAME.....	120
13.2.2 AUTO-START MACRO.....	121
13.2.3 /M: MEMORY	121
13.2.4 /H: HOST	121
13.2.5 /D: DEFINE.....	121
13.2.6 /C: CHOOSE	122
13.2.7 /T: COMMUNICATION TYPE.....	122
13.3 MEMORY USAGE.....	122
13.4 THE "SMALL" VERSION OF ANZIO.....	123
PART IV. TECHNICAL REFERENCE GUIDE.....	124
14. COMMUNICATION PROTOCOL.....	125
14.1 KEYBOARD LOCKING.....	125
14.2 TRANSMIT PRIORITIES.....	125
15. FILE TRANSFER PROTOCOLS.....	127
15.1 SIMPLE UPLOADS	127
15.2 TRANSMIT LONG	127
15.3 TRANSMIT CRC	127
15.4 RECEIVE.....	128
15.5 RECEIVE WITH CRC	128
15.6 CAPTURE AND CAPTURE LONG.....	128
15.7 PASS-THROUGH PRINT	129
16. SENDING COMMANDS FROM THE HOST	130
17. ESCAPE SEQUENCES AND STANDARD KEYCODES	132
17.1 NCR 7900 FUNCTIONS	132
17.2 ADDS VIEWPOINT (NCR 7901) FUNCTIONS.....	133
17.3 WYSE 60 FUNCTIONS	135
17.4 VT220 FUNCTIONS.....	137
17.5 ADDITIONAL FUNCTIONS.....	139
18. ERROR MESSAGES	141
18.1 COMMUNICATION CHIP ERRORS.....	141
18.2 TEXT MESSAGES.....	142
19. WORKING WITH OTHER PC SOFTWARE	146

19.1 FEEDING HOST DATA TO PC PROGRAMS	146
19.1.1 QUERY PROGRAMS.....	147
19.1.2 PICKING FOR LOTUS AND OTHERS	147
19.1.3 CUSTOM-PROGRAMMED EXTRACTION	149
19.2 NOTES ON OTHER PC UTILITIES	149
APPENDIX A. NOTES ON PARTICULAR HOST SYSTEMS.....	151
APPENDIX B. ADDITIONAL PROGRAMS	154
APPENDIX C. MIGRATION FROM EARLIER RELEASES.....	158
APPENDIX D. SERIAL COMMUNICATION PROBLEMS	159
APPENDIX E. DISTRIBUTION INFORMATION	161
APPENDIX F. SAMPLE DEFINED KEYS	166
APPENDIX G. ANZIO ON A NETWORK.....	170
INDEX	171

ii. INTRODUCTION

ANZIO is a family of programs that run on a personal computer (PC) to do terminal emulation, extended terminal functions, and file transfer. The family consists of several members, some running under DOS, some under Windows; some communicating via serial, some via networks such as TCP/IP. The six family members are:

ANZIO.EXE	for DOS serial connection (specifically referred to as Anzio for DOS).
ANZIOS.EXE	a small version of ANZIO, optimized for memory-resident operation.
ANZIO14.EXE	for DOS, using INT (interrupt) 14 to communicate through a redirector.
ANZIONET.EXE	for DOS, using INT 6B to communicate through a redirector.
ANZIOWIN.EXE	for Windows, using TCP/IP (WINSOCK), serial, PicLan, or WLIBSOCK.
ANZIOSCR.EXE	a limited-function version of ANZIOWIN, sold as Anzio Lite.

ANZIO makes your PC emulate a conversational, asynchronous terminal on a UNIX or VMS system (or similar); a system running NCR's ITX, or Ryan-McFarland's RMCOS operating system; or other operating systems. ANZIO does not do page mode or polling mode.

While ANZIO is running, the PC behaves very much like the terminal it is emulating. The host system thinks it has a terminal attached, and displays information accordingly. Keys you hit send characters to the host. Function keys and special keys such as **PG DN** are usually configured to send certain control-character sequences to the host system, although certain key combinations allow you to exercise control over the PC and ANZIO itself.

ANZIO generally uses the bottom line of the screen (or window) for special functions and messages. As a rule, anything that you type on the bottom line is not sent to the host.

TO GET THE PROGRAM RUNNING, see "Using ANZIO", section 1. When you want to terminate ANZIO, enter:

ALT X

This will return you to the operating system.

IF YOU ARE MIGRATING FROM AN EARLIER VERSION of ANZIO, check Appendix C.

ii.1 KEY FEATURES

Unlike many generic "communication" products, ANZIO is optimized for local connection to specific host systems, namely UNIX, VMS, and NCR I-systems. Much of the focus has been on making it work very fast. Much has been done also to go beyond simple emulation, thereby making the PC a better terminal than the one it is emulating. Windows versions of ANZIO are designed to be flexible in their use of screen space, while keeping the screen free from visual clutter.

Following are some of ANZIO's key features:

Screen scrollbar

ANZIO buffers any data that scrolls off the top of the screen, and (optionally) screen erasures. This data can be redisplayed easily with the REVIEW command.

Macro keys

An extensive list of macro key capabilities gives ANZIO a high degree of user programmability.

Host control

Programs running on the host system can control ANZIO and its PC to a high degree.

File transfer

Several file transfer protocols are included, as appropriate to the particular host system.

Configurability

The user can configure many "comfort" features, such as colors, non-blinking cursor, status/gauge line, and beep pitch and control.

Fonts (Windows)

The user can select any fixed-space font to use on the screen, including such character sets as Cyrillic and Greek.

Font sizing (Windows)

The user can scan through available font sizes, with the window following. Or, ANZIO can take over the whole screen, with font sizes chosen to fit.

Clean-screen approach (Windows)

Under Windows, the program adds as little as possible to the screen layout.

Alternate screen widths

DOS versions of ANZIO can deal with 132-column hardware capabilities, or can maintain a 132-column virtual screen on an 80-column actual screen. Windows versions intelligently handle 80 and 132 column screens, as well as custom screen sizes.

Memory-resident operation (DOS)

DOS versions of ANZIO can be pushed into the background, and then can pop up at the stroke of a hotkey. ANZIO can also shell to DOS.

Cut-and-paste

Columns of data on the screen can be picked up and written to a disk file in comma-separated values format, to be imported into spreadsheets and word processing programs. Under Windows, part or all of the Window can be copied to the clipboard (in both text and bitmap mode), and text from the clipboard can be pasted to the host program.

Printer control

From within ANZIO, the user can send the printer setup control characters. Also, ANZIO can deal with more than one printer connection. Windows versions can set printer font and size, or let Print Wizard handle the task automatically.

Mouse actions (Windows)

Several options are available for how ANZIO deals with various mouse-click combinations.

Calculator

A built-in calculator allows 4-function arithmetic in decimal or hex.

High-level screen commands

Certain commands can be sent from a host program written specifically to support ANZIO, allowing easy menus, boxes, windows, and area fills.

Online help

ANZIO contains a system of online help information on all commands.

Multiple emulations

ANZIO emulates the essential terminal types: VT220, SCO ANSI, AT386, Wyse 50, Wyse 60, Versys C332, ADDS Viewpoint/NCR 7901, and NCR 7900.

Local operations



ANZIO allows many local operations, such as DIR, RENAME, TYPE, LOG, etc.

ii.2 HOW THIS MANUAL IS ORGANIZED

This manual is organized in five parts. Part I is a narrative section describing the concepts and operation of the program. It is arranged in logical sequence as you might need to perform various operations. By reading only the first three subsections, you should be able to run the basic parts of the program. But humor us, and read all of Part I at least, before you call with questions.

Part II describes the mechanics of file transfer within ANZIO. This includes a brief description of the uses of the transfer programs provided for each operating system. Part III is a reference section, which explains all commands in detail. Part IV is for the more technically minded. This section gives you technical information on protocols used, etc., in case you're interested.

The last part is a group of appendices. Check through them for helpful information that might apply to your particular installation.

In the following instructions, the program will be referred to as ANZIO for simplicity. Comments that apply to only certain versions will so indicate. The personal computer will be called "the PC". The computer to which you are connected will be referred to as "the host". The symbol  is used to represent the "Carriage Return" character, generated by the "Enter" key. Function keys are noted as the appropriate keycap, such as  for function key 6.

PART I. NARRATIVE

1. USING ANZIO

In this section, you'll learn how to get the ANZIO program up and running. This involves making the correct physical hookup, starting the program, and doing a little configuration. In most cases, fortunately, the configuration is quite easy. The subsections will tell you how to save your settings so you won't have to go through that again.

1.1 INSTALLING ANZIO

ANZIO can be installed to and run from any disk accessible to your PC -- floppy, local hard disk, or networked disk. It cannot, however, be run from the distribution disk, because the files are compressed there.

The installation process consists of two phases. First, you will install the necessary files to your PC disk, using a simple procedure listed below. Then, depending on your host system type and your file transfer requirements, you may transfer certain programs from ANZIO to the host system.

1.1.1 INSTALLING THE FILES TO THE PC


If you are installing ANZIO onto a hard disk, you may want to put it in its own subdirectory. If you don't understand about subdirectories, ignore that for now and get a good book soon on hard disk management.

Put the distribution disk (disk 1) in the A drive (or any available diskette drive).

For DOS: Log to the disk drive that contains the original disk, such as:


A: 

Then enter the following command:

INSTALL <dest> 

where <dest> is the disk (and directory) where you want to put ANZIO, and <op-sys> is your target operating system as defined above. Do NOT put in the less-than or greater-than (" $<$ ", " $>$ ") symbols. Some examples:

INSTALL C:\ANZIODIR 

INSTALL D: 

The INSTALL batch will take you through the whole process. If something goes wrong in the installation process, there is no harm in trying it again. If you continue to have problems, please call us.

For Windows: Run the SETUP program on the diskette.

For more information on what is on the distribution diskette, and what you need in order to run ANZIO, see Appendix E.

1.1.2 INSTALLING THE TRANSFER PROGRAMS

Along with the ANZIO program on the distribution diskette, may be included a series of transfer programs to be loaded to the host computer. By following the procedures for a simple upload found in Part II (File Transfer) for your particular host operating system, it is easy to move the proper files from the distribution diskette to your host. These will then need to be compiled on the host in order for file transfer (other than a simple upload) to take place.

If your host system does not have the appropriate compiler (such as "cc" on a UNIX system), there may be other ways to install file transfer programs. First, check the file "READ.ME" on the distribution diskette. Then, if you still have questions, please call us.

1.2 HOOKING UP THE PC

Note: this section applies to a serial connection only.

The physical hookup between your PC and your host machine is usually very simple. Keep in mind that the host machine "thinks" that your PC is a terminal (a CRT). So the cable coming from the host is the same kind of cable that would otherwise go to a CRT. We recommend, in fact, that you unplug a line from a CRT to use initially, because that way you'll know that the line works and is properly configured, thus eliminating one potential source of problems.

1.2.1 WIRING THE CABLE

Note: this section applies to a serial connection only.

Your PC must have a "serial port". This is the device by which the PC talks to the outside world of host computers, modems, some printers (serial ones), and other devices. Some PCs have a serial port "on the mother board" as standard equipment. Other serial ports are contained on add-in boards. An internal ("integrated") modem has its own serial port, as well as the modem.

Each serial port (including internal modems) has an ADDRESS and an INTERRUPT VECTOR (also known as an IRQ). These are generally set by jumpers or switches on the board itself.

For Windows: Configuration settings in Windows tell it how your serial ports are configured. ANZIO then simply specifies a port number. Thus, if AnzioWin is set for PORT 2, it will tell Windows to open communication on port 2, and Windows must know how your port 2 is configured.

For DOS: ANZIO accesses the serial port at a very low hardware level, using the specified PORT and IRQ, as explained below.

The following port addresses are defined by the industry:

PORT	ADDRESS	IRQ
1	3F8-3FF	4
2	2F8-2FF	3

Other options that ANZIO defines are:

3	3E8-3EF	4
4	2E8-2EF	3
5	3220-3227	4
6	3228-322F	3

Finally, if your serial port is at an unusual location, you can store its address in a location reserved by the BIOS, and have ANZIO reference that location. This would be a very unusual situation. Please contact us for more information.

If your IRQ does not correspond to the PORT as above, ANZIO now lets you set it explicitly - see the IRQ command in the reference section.

It is VERY IMPORTANT that a) the vector correspond to the port, and b) you don't have two boards set to the same port. Having two serial ports share an IRQ is acceptable, provided they do not operate simultaneously. For instance, if you have a serial mouse on port 1, and are trying to communicate on port 3, you are asking for trouble. This is a restriction of the PC design, not something imposed by ANZIO.

WATCH YOUR GENDER! The serial port always has a MALE plug. A 25-pin female plug is probably a parallel printer port. If your cable from your host system ends in a male plug, you will need a 25-pin female-to-female gender converter (available from any PC supplier). This should NOT be a "null modem" (a special type of adapter)!

Your serial port may instead have a 9-pin male plug (because of space considerations). If this is the case, your best bet is to buy a standard 9-to-25 pin adapter or cable (such as a modem cable), and a gender converter if necessary.

If you are using an integrated modem, it is simply plugged into the phone jack. You may also be able to plug your phone into the modem.

PLEASE NOTE: We are glad to help with communication problems over the phone, but there comes a limit to what we can do from a couple of thousand miles away. Any reputable PC dealer should be able to install and test a serial port or internal modem.

The following is the standard 9-pin wiring connection:

1	CD (carrier detect)
2	SD (send data)
3	RD (received data)
4	DTR (data terminal ready)
5	GND (ground)
6	DSR (data set ready)
7	RTS (request to send)
8	CTS (clear to send)
9	RI (ring indicator)

The following is the standard 25-pin wiring connection:

2	SD (send data)
3	RD (received data)
4	RTS (request to send)
5	CTS (clear to send)
6	DSR (data set ready)
7	GND (ground)
8	CD (carrier detect)
20	DTR (data terminal ready)
22	RI (ring indicator)

Note that it is possible for ANZIO to run on as little as 3 wires: ground, SD, and RD. The host system, however, may need some jumpering in order to do this.

1.3 STARTING THE PROGRAM

Now that you've got it plugged in, you can get the program up and running.

For your initial venture into Anzio, just run the program.

For Windows: Start AnzioWin from its icon.

For DOS: Change to the appropriate directory, and run the program. For instance,

```
C:   
CD \ANZIODIR   
ANZIO 
```

Note that the name could be ANZIONET or ANZIO14.

1.4 GETTING YOUR PARAMETERS STRAIGHT

Warning: this area applies only to DOS.

The first thing ANZIO does is to try to read a "defaults" file called ANZIO.DEF (for DOS) or ANZIOWIN.DEF (for Windows). This file can contain all the system settings for such things as baud rate, tabs, etc. It will

not exist the first time you run the program, so the program will ask you four questions: a) operating system, b) port, c) baud rate, and d) terminal type. We'll look at each one closely.

1.4.1 OPERATING SYSTEM

ANZIO presents you with several options to tell it what the operating system of your host computer is. This information is important for ANZIO to make some educated guesses as to how to communicate and handshake with the host machine. Enter the number that corresponds to your answer.

1.4.2 PORT

ANZIO needs to know which serial port is connected to your host computer. For a definition of serial ports, see section 1.2.1.

Enter a number from 1 to 10.

1.4.3 BAUD RATE

Your PC and your host machine must communicate at the same speed, known as the baud rate. We'll assume that the host end is already set (either through software or with switches), and we must match it. If you don't know your baud rate, you might try to find out from someone who manages your system, or you might try guessing - you won't hurt anything.

If you are connected to a modem, you might need anything from 1200 to 38400. If not, it's probably 9600 or 19200. Enter the number.

1.4.4 TERMINAL TYPE

ANZIO can emulate several terminal types. It is important, of course, that the host system and the terminal (or PC, in this case) talk the same language. At this point, we'll assume your host is already set up for a particular terminal type. Enter the number that corresponds to the correct terminal type.

Section 1.11 provides more information on choosing a terminal type.

After you answer these questions, ANZIO will make some assumptions about several of its parameters. It will also try to read a sample macro key

file, named RMCOS.KYS (if you told it you were running with RMCOS) or SAMPLE.KYS (if you told it anything else). By hitting any key (or waiting 15 seconds), you will get beyond the banner screen to the "live screen".


If you get the message UNABLE TO INITIALIZE COMMUNICATION, you have probably told ANZIO to use a PORT that does not exist on your machine.



1.5 GETTING ACQUAINTED


By now you're probably looking at a nearly blank screen. That is, it should be blank except for the STATUS LINE at the bottom. This status line can show you several pieces of information: CAPS indicates you have the CAPS LOCK key on. NUM shows that NUM LOCK is on. LOCK shows you when your keyboard is locked, meaning nothing you type will go through to the host. You may also see communication errors from time to time indicated here. The status line can be turned off, but we'll leave that for later.

From the live screen, any "normal" key you press will go out through the communication line to the host computer. Function keys and special keys can either cause a special code to be sent to the host, or can invoke some function native to ANZIO. We'll get into that later. For now, let's see if we can communicate.


1.6 LOGGING ON

If you are connected to a UNIX system, but haven't seen a login prompt, just hit . You should see the UNIX login prompt. At this point you can enter your login name and password just as you would with a terminal.

If you are using an NCR I-system, your terminal must be logically attached before the host system will pay any attention. On IRX/ITX, this must be done from another terminal. On any IMOS system, you MAY be able to use   to self-attach, depending on how your system was SYSGENed; if not, use another terminal. To attach your terminal from another terminal enter


AT (n) 

where 'n' is the terminal number.

On IRX or ITX, you must log on. This is done the same as with a terminal, using either **CTRL** **C** and , or **CTRL** **BREAK**. If your login doesn't "take" the first time, you may need to try it again - hit **F4**¹ first to make sure your keyboard is unlocked. See Appendix F on sample defined keys for a couple of "login" keys.

On other systems, just proceed as though this were a normal terminal.

If you get "garbage" on the screen, chances are that your BAUD rate is incorrect. To try another baud rate, enter **ALT** **M** to get the "menu" screen, then enter:

BAUD 9600 

or whatever rate is appropriate. An  will get you back to the live screen, where you can try again.

Once you successfully log on, you will see the operating system's "banner" at the top of the screen, just as on a normal terminal. Now try running some of the host programs you are accustomed to running.

If something goes wrong at this point, you'll need to skip ahead in the manual. *Section 4 tells you how to enter commands. Section 12 details all commands.* We'll cover some common problems here.

If you can't see the characters you're typing, try²

HALF DUP 

If you see two of every character you type, try:²

FULL DUP 

If you see intelligible characters, but the positioning is all wrong, see TERM.

If you see "foreign" characters, you may need to change PARITY and DATA BITS.

¹ If you have redefined function keys to emulate a terminal, see section 3.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

If you have gotten no response at all, you have a problem with a) your PORT and/or BAUD setting, b) your serial port hardware, or c) your cabling.

1.7 USER PREFERENCE ITEMS

Now that you've gotten past the essentials, you may want to look at some of the "user preference items". These are parameters that affect the way the ANZIO "CRT" appears and sounds. We'll just mention some of the commands here that you might want to investigate in chapter 11: COLOR, CURSOR BLINK, BEEP, BEEP IDLE, PITCH, STATUS LINE, and GAUGE.

If your screen hardware (i.e., your EGA or VGA card) can display 132 columns at once, ANZIO would like to know. That way, ANZIO can respond to a command from your host computer to switch to 132-column mode. The problem is that there is no standard "video mode" number - each card has its own.

You'll find this information by checking the manual for that EGA/VGA card. If your board can display 132 columns, find out what "video mode" corresponds to 132 by 25. What we need here is a hex number. Once you have determined that number, use it with the MODE-132 command. For an example, see section 8.1.

1.8 TERMINATING THE PROGRAM

To exit from ANZIO just hit


ALT X

If you have made changes to either the settings (the defaults file information) or the defined keys (macros), and have not saved those changes, or if this is your initial run, ANZIO will ask you if you want to save them, and to what file names.

ANZIO can store your settings in two groups. One group is called "defined keys", which we haven't gotten to yet. This group is stored in a "key file". The other group is basically all of those parameters you saw on the HELP screen. This second group is stored in a "defaults file".

Whenever you start up ANZIO, it will look for a defaults file called ANZIO.DEF. If it finds one, it will load those settings. If it doesn't find one, as in the first time you ran the program, it will ask you for the basics. The defaults file can also tell the program to load a key file, thereby recreating your whole setup.

Note that it is possible to have multiple defaults files, as well as multiple key files. See "Command Line Parameters", section 13.2.

At this point, we suggest you simply enter "Y" to save each file, and enter  to accept each default file name.


1.9 MEMORY-RESIDENT OPERATION

Note: this section applies to DOS only.

ANZIO has two ways of staying in memory while you run something else. These use variations of the RUN and STAY command.

1.9.1 RUN

The RUN command tells ANZIO to stay in place and run some other program over the top of it. That is, if you tell ANZIO to

```
RUN WP.EXE 
```

it will load the indicated program. At the point that you are finished using WP.EXE and quit from it, you will be returned to ANZIO. ANZIO's screen will be intact, and your connection to the host should still be good, unless the program you ran messed with the serial port.

Note that it is possible using this method to have either a macro key in ANZIO or a command from the host computer cause the PC to execute a particular program.

If you just do a RUN with no program name, ANZIO will "shell to DOS". That is, you will end up at the DOS command line. From there you can run any series of programs. To return to ANZIO, simply enter at the DOS command line:

EXIT 

1.9.2 STAY

It is also possible to put ANZIO into a "TSR" or "memory-resident" mode where it can "pop up" over another program. When ANZIO is running, enter the command:²

STAY 

ANZIO will go to sleep, and you'll be back at the DOS command level. Now run some other program, such as:



WP 





Once you are into your word processing program, you can at any point enter

to pop up ANZIO (this keystroke can be changed with the HOTKEY command). When ANZIO pops up, it saves your entire screen and restores its own screen. Then when you do another STAY, you will be returned to your other application. In this way you can repeatedly and quickly switch back and forth between the two programs.

ANZIO can pop up on most video boards, over most programs that run in *text* mode. If you will need to pop up over a program in *graphics* mode (i.e., WP in preview mode, WORD in a graphics mode, 123 release 3, etc.), you must do a STAY/G rather than a STAY the *first* time, in order to save enough memory for graphics.

ANZIO can be unloaded at most times when nothing else is loaded on top of it, by doing the normal END or   command.

Notice that in the included sample key files,   is already defined to do a STAY/G. That means that you can switch in and out of ANZIO using the same key combination,  . Note, however, that you can't execute

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

that (or any) macro when the LOCK indicator shows on the status line. To remove it, hit **F4**¹.

1.9.3 MEMORY USAGE

ANZIO is a complex, and therefore large, program. You may not be able to run it and your other application(s), depending on memory requirements and availability. There are two ways you may be able to solve this problem. First, it is possible to reduce ANZIO's memory usage at the expense of REVIEW memory. See section 13.3. Second, there is a "stripped-down" version of ANZIO, called ANZIOS.EXE. See section 13.4.

ANZIO can NOT be loaded into high memory or extended memory, due to its size.

1.10 CONFIGURING YOUR HOST SYSTEM

In order for ANZIO and your host computer to communicate effectively, they must speak the same language. There are many aspects to this language, including BAUD rate, PARITY, TERM type, DUPLEX, and more. We have covered most of these.

On a UNIX system, these parameters are set up in three files. The file "/etc/inittab" or "/etc/ttytype" determines the assumed terminal type for each physical connection to the system. That file also references an entry in the file "/etc/gettydefs", which can set parity, echo, etc. It is also possible for each user to have a file (a shell script) named ".profile" in its home directory, which can do things such as query for terminal type and issue "stty" commands.

On an ITX system, all the parameters for each terminal are set in the SYSGEN entry for that terminal line.

For a description of things to watch out for on particular host systems, please refer to Appendix A.

¹ If you have redefined function keys to emulate a terminal, see section 3.

1.11 TERMINAL TYPES

ANZIO can emulate several common terminal types. The nitty-gritty is covered in section 17. Your host system or application software may lock you into a particular type. Or, you may be able to choose. Following is a description of some characteristics of each terminal, to assist in the decision.

The term "attribute" is used to describe the appearance of characters on the screen. This includes such characteristics as reverse video, blinking, underline, intensity, etc.

Be sure to see also section 1.11.1 on function keys.

The VT100/220 is the closest thing there is to a standard, and in fact is the basis of the ANSI standard. It is particularly adept at attributes/colors, in that each location on the screen can have its own, and the attributes don't occupy a position. ANZIO emulates both colors and attributes. The VT100 is limited, however, in special keys - it has only four function keys and a few special keys. The VT220 has more function keys, but they do not correspond well to the PC's keyboard.

ANZIO, when set in VT220 mode, responds to control sequences for the VT100, VT102, and VT220 (except for some obscure VT220 commands), as well as some additional sequences we have defined. This is ANZIO's most powerful operating mode.

The WYSE 50/60 is also a widely-supported terminal. The WYSE has many function keys, which have a "standard" definition, as well as several special keys. WYSE 50 supports two kinds of attributes. First, there are field attributes, which occupy a screen position. Second, there is a single attribute (per screen) which is used to indicate "protected" fields - this one does NOT take a position. Many software packages take advantage of the second type only. ANZIO supports both types, but it does NOT support protected fields as protected, nor does it support page mode.

The WYSE 60, on the other hand, allows character attributes, meaning each location can have its own attribute, and attributes do NOT occupy a position. This is a much preferred approach, and the one ANZIO uses. However, most software for WYSE 50 will also work with ANZIO.


The WYSE also has an advantage in that its arrow keys send single-byte control codes. Some software, notably "vi" under UNIX, seems to deal better with this.

The ADDS Viewpoint (NCR 7901) is a commonly-emulated terminal. The original model did not have function keys, so there is little consistency in emulating them. Like the WYSE, arrow keys are single bytes. The ADDS is limited in its attributes, using "tagged" attributes. Each location on the screen is either tagged or not tagged (one bit). Tagged locations show up in a specified alternate attribute. Thus, only one alternate attribute is available per screen.

The NCR 7900 is not widely supported outside the NCR community. Attributes are strictly field-level, occupying a screen position. Using a 7900 with some software requires disabling that software's use of attributes in order to be consistent.

In conclusion, if you have a choice, set ANZIO for VT220 mode, and set your host system for VT100, VT220, or ANZIO mode (see section 1.11.2).


1.11.1 DEALING WITH FUNCTION KEYS

Who owns the function keys? That is, if ANZIO is emulating a WYSE 60 (which has function keys), and you hit , does that cause a certain code to go to the host, or does it invoke ANZIO's HELP screen?

When you set ANZIO to emulate a certain terminal, it does NOT automatically adopt the function keys. This makes ANZIO easier to use if your host software never uses function keys.

Likewise, ANZIO does NOT respond to function key downloads from the host - this could cause much confusion.

If you would like to use function keys with your host software, you have two choices. First, you can load one of our macro key files, such as²

```
MERGE VT220.KYS 
```

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

The MERGE command reads the key file in, just like the READ command, but it leaves your assumed key file name the same as it was, and does not wipe out any macro keys already defined.

We have included several files of key definitions, each with a "KYS" extension. Each has a corresponding ".DOC" file explaining it. You can TYPE the ".DOC" file, read it with an editor or viewer, or print it out.

The second option is to define the keys yourself.

If you define **F5** to send a certain sequence to the host, you can still get ANZIO's HELP screen in two ways. First, you can use **ALT F5**, **SHIFT F5**, or **CTRL F5** if you haven't defined them also. Second, you can use **ALT M**.

When you use certain host software, this special key business can get quite confusing. For instance, let's take the spreadsheet package 20/20. The documentation for 20/20 will refer to the PAGE DOWN key. You must then look in an appendix to determine that for the VT100, for instance, a PAGE DOWN is equivalent to a **<PF1>↓**. Now if you are using ANZIO and you have loaded VT100.KYS, **F1** on your keyboard will generate the control code equivalent to the VT100's **<PF1>** key. So, you could accomplish the PAGE DOWN by hitting **F1↓**. For simplicity's sake, if you use 20/20 a lot, you might want to define your **PG DN** key to send the entire combination.

1.11.2 USING A TERM TYPE OF "ANZIO"

For ANZIO users with a UNIX host, we have provided some files to let UNIX know your terminal is actually ANZIO. This has several advantages: 1) many function keys and special keys are defined; 2) color is supported; and 3) ANZIO has some capabilities which go beyond those found in a standard VT220.

Note that the ANZIO.TIC file assumes a color capability on your PC. **If you are running a monochrome PC**, use "ANZIO-M" below instead of "ANZIO".

Following are the steps necessary to implement this approach.

First, set ANZIO itself to act as a VT220:²

```
TERM VT220 
```


Then, tell ANZIO to load the correct keys file:²

```
MERGE ANZIOTIC.KYS 
```

Next, upload the file "ANZIO.TIC" to your UNIX system, using the SIMPLE UPLOAD procedure listed in section 11.5.1. On UNIX, make the name lower-case ("anzio.tic"). This file is a terminfo source file.

For the next step, you will need to be super-user (root). If you don't understand that, you're in over your head here. Get help.


Now tell UNIX:

```
tic anzio.tic 
```

This tells UNIX to run the terminfo compiler "tic", reading the file "anzio.tic". YOU WILL PROBABLY GET WARNINGS, since this file has entries for both UNIX and AIX. Barring serious problems, however, your UNIX will now consider "anzio" or "ANZIO" a legal TERM type, for any program that uses terminfo. So set your TERM variable to "anzio":

```
TERM=anzio;export TERM 
```

Then tell UNIX to initialize the terminal:

```
tput init 
```




Note that if you have application software that uses a method other than terminfo (such as Word Perfect), it will not understand a TERM of "anzio". We are in the process of developing ANZIO definitions for popular software -- contact us.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

2. HELP WHEN YOU NEED IT

2.1 THE HELP KEY AND SCREEN

You have already seen that you can get some help just by hitting the   or the  key¹. This brings up the HELP screen, which shows all the settings and commands. It is, of course, brief. For more help, you can use the online help file. To get information on a certain command, enter at the "Function" prompt the word "HELP", followed by a space, followed by the first word of the command in question. For instance,²




```
HELP BAUD 
```

will bring up information on the BAUD command.

This information is pulled from a file named ANZIO.HLP. This file should be in the same directory as ANZIO.EXE.

Two special options have been added to the HELP file. First, HELP INDEX will produce a list of available topics (see below). Second, HELP ASCII will show you the ASCII character chart with hex equivalents.

2.2 THE HELP INDEX AND

At nearly any point in the program, you can hit   for help. This will bring up an index of topics on which you can get help. Simply move the highlight to the appropriate subject, using the arrow keys, and hit .

¹ If you have redefined function keys to emulate a terminal, see section 3.
² ANZIO commands must be entered at the "Func:" prompt; see section 4.

3. FUNCTION KEYS AND THE KEYBOARD

Several of the function keys have special meanings to ANZIO. These key assignments are displayed as part of the KEYS screen.

Note that it is possible to "overdefine" these keys. That is, you or someone who set up your system may have set `F3` (for example) to do something else. In that case, you could use `SHIFT F3`, `ALT F3`, or `CTRL F3`. If all the variations are redefined, use an `ALT` code shown further below.

`F1` is 'send defined key', explained below with the DEFINE command.

`F2` invokes the line editor. This allows you to edit a line (or field) and then send it to the host. See section 6.

`F3` is the FUNCTION PREFIX key, displaying 'Func:' on the 25th line. The next character entered is sent exactly as per the FUNCTION key on the 7900 and some other terminals, that is, a hex 02, then the keystroke, and then an `ENTER`. If just a `ENTER` is entered, no action is taken.

`F4` is used to temporarily unlock the keyboard -- see Keyboard Locking, section 14.1.

`ALT U` is the same as `F4`.

`F5` is the HELP key; it will list the commands, any open files, and the current status of all options (Baud rate, duplex mode, etc).

`ALT M` performs the same function as `F5`. That is, it brings up the HELP or MENU screen.

`F6` brings up the FUNCTION prompt, to give ANZIO commands from the live screen.

ALT F

performs the same as **F6**, above.

F9

is the PANIC BUTTON. It aborts a defined key in progress, turns TRANSMIT OFF, exits any endless loops, and empties the keyboard buffer.

ALT A

performs the same as **F9**, above.

F10

is the BREAK key. That is, it sends the BREAK signal to the host computer, just as the BREAK key on a terminal does. It will work at any time, and erases whatever is in the type-ahead buffer and unlocks the keyboard.

CTRL BREAK

can also be used as a break key.

ALT X

exits from ANZIO.

ALT H

brings up ANZIO's HELP INDEX.

4. TALKING WITH ANZIO

By now you may have noticed that, when entering functions, ANZIO lets you get away with some "sloppiness". Let's take a look at exactly what the program expects and needs at this point.

4.1 ENTERING COMMANDS

Commands to ANZIO, are entered in one of two places. First, you can hit **F6** or **ALT M** and enter the command at the bottom of the "live" screen. Second, by hitting **F5** or **ALT M** you get the "HELP" screen. It brings up a screen showing you all the commands available to you, as well as the current settings of all configuration options. At the bottom of this screen, you can also enter commands, and see them immediately be updated on the screen. When you are finished setting up, you can hit **ESC** to return to your "live" screen. Here is what the program will allow as far as entering commands:

1. Commands can be entered in upper or lower case, in any combination.
2. The first word, which identifies the command, does not need to be complete. It only needs to be long enough so the program can distinguish it from some other command. So if you can't remember how to spell "INTERPRET", use "INT". However, the command "DEFINE" can not be abbreviated.
3. Intermediate words can be shortened, and in many cases can be omitted., such as "FULL" instead of "FULL DUP", or "DATA 8" instead of "DATA BITS 8".
4. Leading, intermediate, and trailing spaces are ignored (except as part of defined keys).
5. The word "ON", such as in "GAUGE ON" is optional. Thus "GAUGE" is equivalent to "GAUGE ON".

6. Any command requiring an "ON" or "OFF" will also accept "SWITCH" or "TOGGLE", which will turn it on if it is off, and vice versa. So "GAUGE SWITCH" will alternately turn the gauge on and off.

4.2 EDITING COMMANDS

Some commands can get lengthy, so the program provides some editing capabilities when working on the bottom line. The keys perform functions as follows:

<BACKSPACE>

Deletes the character to the left of the cursor, and backs up the cursor.

, 

Move the cursor without changing the contents of the line.



Switches between "insert" and "overtyping" mode. An INS indicator shows at the right end of the command line.



Deletes the character at the cursor.



Takes the cursor to the beginning of the function entry.



Takes the cursor to the end of the line.






Erases all characters from the cursor to the end of the entry.



Function key 7 has a special purpose designed for use with the KEEP, PRINT, and PICK functions. It allows you to define a rectangular area (column) on the live screen, and return its coordinates to the command line. See section 11.

Control-P serves as a "prefix" key, allowing you to enter the actual keystrokes in this table. That is, to enter an "escape" character in the command, use   .



Aborts the command entry process.

When function keys are entered on line 25, they will show in reverse video with the function key name, preceded if necessary by an 's' meaning 'shift', a caret (^) meaning 'Control', or an 'a' meaning 'Alt'. Other special keys will show appropriate names, or hex codes.

The line-25 entry can be up to 255 bytes long. When you move too far to the right on the screen, the line will "slide" to accommodate you.

5. DEFINED KEYS (MACROS)

Defined keys are possibly the most useful function in ANZIO. Anywhere you need to repeatedly type something, you can assign that "something" to a single key. Then, rather than typing the entire sequence every time, you can cause ANZIO to type it for you. Several special functions, listed below, can be included within defined keys to further expand their usefulness.



To see the currently defined key sequences, enter ²






KEYS 

This will first display all system-assigned keys. Then it will show user-defined keys. The key that is defined is shown first in reverse half video (or its color equivalent), followed by its contents, followed by a space in reverse half video to show where the end of its contents is. Any non-ASCII character in the contents will be shown, in reverse video, either by name or as a hex number; e. g., 01 for a <control-A>.

If you are having problems with sending defined sequences ('System Overload', or dropping characters), see DELAY (section 12), as the character transmission rate may be too fast.

5.1 WHAT KEYS YOU CAN USE

There are three general categories of keys that can be DEFINED. The first consists of all printable characters, from space (hex 20) to tilde (hex 7E). The second consists of "special" keys, including function keys, named keys such as , and  keys. The third set is control codes, that is, hex 00 through hex 1F.

Function keys  through  can be defined four ways each: normal, with , with , and with . Also, if your systems supports keys <F11> and <F12>, ANZIO will support them also, both normal and shifted.


If all that confuses you, just do a little experimentation.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

Each defined key can contain up to 246 keystrokes. In addition, the available memory can limit the total number of keystrokes possible. Available memory can be checked in the HELP screen.

5.2 DEFINING MACROS

Keys are defined with the DEFINE command. The format is:²

```
DEFINE <x> <string> 
```

where <x> is the key to be defined, and <string> will become its contents. Note that the word DEFINE can not be abbreviated, and that there must be exactly one space after DEFINE and one space after the keystroke shown as <x>.

If you define a key that has already been defined, the old one will be deleted (without warning). If you define a key with no <string>, any definition in place will be deleted and nothing will replace it.

5.3 EDITING MACROS

Defined keys may be edited as they are being entered by making use of the command-line editing features explained in section 4.

A variation on this is also possible for a key that has already been defined. That is, you can edit it rather than reenter it. To begin this, enter:²

```
DEFINE <x>?
```

As soon as you enter the question mark (note that it does not have a space before it), ANZIO will erase it and display the existing contents of the key to the right. Now you can edit it as necessary, again using the command-line editing keys discussed in section 4.


To copy a string from one key to another, begin editing the first one, then back up and change the command. For instance, to copy the string assigned to X to Y, enter:²

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.


² ANZIO commands must be entered at the "Func:" prompt; see section 4.


```
DEFINE X?
```

Then back up and replace the X with Y, and hit .










5.4 EXECUTING MACROS






The manner in which a defined key is executed or initiated depends on the class of key (see section 5.1).

An ASCII key (a printable character) is initiated by entering ¹ followed by the key.





A special key, such as a function key, or a control character, is initiated by simply hitting that key; the  prefix is not used.

5.5 OVERDEFINING SYSTEM FUNCTION KEYS

The system-assigned function keys, such as  and , can be "overdefined". That is, you can assign your own meaning to them. Your definition always takes precedence. You can still use , , or  to bring up the help screen, for instance, unless you have over-defined ALL of those. You can also use  for , and  for .

We strongly advise you NOT redefine , , , , or .

5.6 BLOCKING KEYS

There may be some keystrokes that you don't want to allow the operator (or yourself) to inadvertently send to the host system. Examples might be , , , , and so forth. By defining these keys with a do-nothing string, you can accomplish that task:²

```
DEFINE   { } 
```


¹ If you have redefined function keys to emulate a terminal, see section 3.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

As you will see shortly, the curly braces have a special meaning, that in this case causes nothing to happen - the   will not be sent to the host.

5.7 SAVING AND RELOADING MACROS

After you have built a library of defined keys, you will probably want to save them on disk so they can be reused. Simply enter:²


```
SAVE <filename> 
```

This tells ANZIO to save all defined keys into a file named <filename>. Be careful with this filename, since any existing file of that name will be deleted in the process of saving these keys. Note that ALL user-defined keys are saved.

You can also tell ANZIO to save the keys to its current key file name (the file they were read from in the first place), by entering:²

```
SAVE 
```


To reload a set of defined keys, enter:²

```
READ <filename> 
```

This will read a set of defined keys into memory. Any defined keys already in memory will NOT be erased, but they MAY be overwritten, if the same key is defined in the file being loaded.

ANZIO keeps track of the name for the key file, in its "default" file, so it will find them automatically the next time ANZIO is run.

To give ANZIO a clean slate (no function keys defined), from the DOS prompt enter:

```
ANZIO NONE 
```

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

If you exit from ANZIO using END, STOP, STAY, or **ALT X**, and you have not saved alterations to your defined keys, ANZIO will ask you if you want to save them.


5.8 SPECIAL FUNCTIONS INSIDE DEFINED KEYS


Certain keys have special meaning when embedded in a defined key.


5.8.1 TABS





Tab characters may be included in the defined key by simply hitting the <TAB> key. This will show as a reverse video TAB on the screen during the DEFINE process, but will take up only one keystroke in the defined key. When the defined key is sent, the tab will be processed according to the current tab settings, just as though you had typed it.

5.8.2 USING '!' FOR

Anywhere a  is required, use the vertical bar character, '!'. For example, to create a defined key to get you into text editor on an I-system (see UNIX example later), you might enter the following:²

```
DEFINE E AS EWF EWF(3) |
      AS A TEXTFILE(2),OW |
      AS LO (LP) |EX $EDIT,RE | 
```

(this is all entered on one line). Once the E key has been defined like this, any time you enter E¹, the PC will send this to the host:²



```
AS EWF EWF(3) 
AS A TEXTFILE(2),OW 
AS LO (LP) 
EX $EDIT,RE 
```


² ANZIO commands must be entered at the "Func:" prompt; see section 4.


¹ If you have redefined function keys to emulate a terminal, see section 3.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.


5.8.3 PAUSES


To wait for operator input in a defined key sequence, use the number sign, '#'. When this character is found while transmitting a DEFINE string, the ANZIO program will stop transmission, accept your entry up to a  and send it (the  itself will not be sent). Then, transmission of the key string continues with the character following the '#'. For example, to create a key sequence to copy files on an I-system (see UNIX example later):²


```
DEFINE C AS A #(3) |  
      AS B #(1),NE,300,AP |  
      MOV A B | 
```



Then, when you enter  C, the PC will send to the host:

```
AS A
```

Then it will wait for you to enter text (a file name). Any key you enter will be sent, until (but not including) the . The "defined key" then takes over, and sends:

```
(3)   
AS B
```

and waits for another filename. Your filename is sent to the host. When you enter , the defined key continues with

```
(1),NE,300,AP   
MOV A B 
```


The PC is then finished transmitting its defined key sequence, and returns to normal text entry.

5.8.4 LOCALLY DISPLAYED TEXT

To display, but not send, text in a defined string, enclose it in curly braces, '{}'. When a left brace ({), is encountered during transmission, all characters following, up to the right brace (}), are displayed on the PC, but NOT sent to the host. The characters '|' and '#' have NO special meaning within braces; they are displayed on the screen.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.


Displayed text can be used to place prompts on the screen in conjunction with the operator input (#), above. For example:²

```
DEFINE C AS A
  {Source filename}#(3)|
  AS B {Destination name}#(1),
  NE,300,AP|
  MOVE A B| 
```

(entered on one line). Note that it is also possible to include cursor movement characters in locally displayed text.

As a UNIX example, consider the following:²

```
DEFINE v vi
  {ENTER FILE TO EDIT}#| 
```

(entered on one line). Now, if you enter¹  v, ANZIO will issue to UNIX:





```
vi 
```

and then will display on the screen:

```
ENTER FILE TO EDIT
```

When you enter a file name, it will be sent to the host to complete the command.

5.8.5 EMBEDDED FUNCTIONS

Two system-defined function keys can be included inside a defined key. These are , to initiate a local function, and  to invoke the line editor. Overdefining function keys has no effect on this use of  and .

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

¹ If you have redefined function keys to emulate a terminal, see section 3.

5.8.5.1 EMBEDDING FUNCTIONS WITH *Cj*

It is possible, and very useful, to embed a function in a defined key. In this way local operations can be "programmed", making it easier on the operator. For instance, if every day you had to open a file named UPFILE and send it to the host, you could enter:²

```
DEFINE S [F6] OPENI UPFILE |  
[F6] TRANSMIT | [Enter]
```

The vertical bars (|) shown are necessary to terminate the embedded [F6] functions.

Some functions would not, of course, be logical to embed, e.g. DEFINE itself. It IS possible to put a pause (#) inside a function inside a defined key. So, to expand on the example immediately above, you could have the operator enter the actual name of the file to be transmitted:²

```
DEFINE S [F6] OPENI # |  
[F6] TRANSMIT | [Enter]
```

5.8.5.2 LINE EDITOR WITH *m*

An [F2] embedded in a defined key will initiate the line editor. If there are keystrokes in the defined key following the [F2], these will be processed by the [F2] editor until a vertical bar (|) is reached. If there are NO keys following the [F2], or a number sign (#) is encountered, the operator will be given control. When the operator is finished with the line editor and hits [Enter], the line editor will process any remaining keys in the defined key (up to a |), send the edited line to the host, and control will resume with the rest of the defined key.

5.8.5.3 KEYSTROKES FOR FUNCTIONS



Several of ANZIO's functions can also take keystrokes from a macro (defined) key. So for instance:²

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.


```
DEFINE   REVIEW | 
```

will define the  key such that it will first take you into the REVIEW routine, then feed that routine a , telling it to show the prior screenful.

Obviously, things can get quite out of hand. A little experimentation does wonders.

5.8.5.4 PREFIXING WITH '~'


Because you may have a need to send one of the characters that cause special actions in defined keys, we have given you a way to do that. Simply prefix it with a tilde (~). Any character immediately following a tilde will be transmitted, not interpreted. So, to program a key to send '{A}', you would do:²

```
DEFINE a ~{A} 
```

The tilde here causes the left curly-bracket to be sent, rather than cause the 'A' to be displayed locally.

5.8.5.5 STACKING FUNCTIONS


Two of ANZIO's commands have been set up so that they can feed their results to other commands. These are DIR/S and MENUBAR. For instance, if you want to have a macro key open a file for transmission, but you want to display to the operator a list of possible files, you can define a macro to do:

```
OPENI  DIR/S *.DAT|#| |
```

The DIR/S command would present a directory on screen of all files matching "*.DAT". It would then allow the user to move a highlight around to indicate a file to be selected. When the operator does so, the DIR/S command will feed the file name to the OPENI command. The '#' allows the operator to give input to the DIR/S. All the '|' indicators are necessary.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

Likewise, suppose that there is a list of file names on the screen (generated by the host), and you want to receive one with Kermit. You can set up a macro to do:

```
KRECEIVE  MENUBAR 1 13 1 23 0 0 1 13|#||
```

The MENUBAR command would place a menubar over the existing file names, and allow the user to select one. The selected name would then be used to complete the KRECEIVE command.


5.9 NESTING DEFINED KEYS



For more complex macro key requirements, you can nest or branch macro keys, using the CALL and INVOKE commands.

If a macro key contains the ANZIO command to CALL another key, that key will be started. When the called key is finished, ANZIO will resume processing the first (calling) macro.

The INVOKE command is similar, but it will NOT return to the calling key.



6. THE KEY




The  key¹ invokes a powerful and flexible line/field editor that is usable in most host systems. It can be a function in and of itself, or it can be part of a more complex function built into a defined key.

In a nutshell, the  editor "picks up" the screen line on which the cursor is positioned and reads it into a special buffer, then allows you to manipulate it, and then sends the entire line or field back to the host computer. If the displayed field uses video attributes (reverse, underline, etc.), then the  editor will deal with a field (as wide as the current attribute). Otherwise, it will deal with an entire line. When dealing with the entire line, it assumes that columns 73-80 are a COBOL I-D area, and treats them specially, as explained below.

Manipulation options include the ability to move the cursor either direction, or to either end of the line; typeover, character insertion and deletion, tabbing both directions, and partial erasures. Many of these operations duplicate those used in editing function entries on line 25.


6.1 USE AS A FREESTANDING OPERATION

The  editor can be invoked at the beginning, or in the middle of, a line of entry on the screen. Ordinarily, it makes sense only when there is data on the screen to be edited, but that is not always true. In order for you to judge when it is appropriate, consider the steps that the  editor goes through.

First, the program looks at the position of the cursor on the screen. If you wish to select another line from the live screen, simply use the  and  keys to select the desired line. The POSITION of the cursor on the line is also important. If you have entered characters immediately BEFORE invoking , those characters have already been sent to the host computer, and it is waiting for more. Let's call this spot the CURSOR START POSITION.

Second, the program reads the screen memory for the entire field or line in question into the editor buffer.



¹ If you have redefined function keys to emulate a terminal, see section 3.

Then, you the operator are given a chance to modify the line's or field's contents. This proceeds until a  is entered. Key functions for editing are explained below.

Finally, the program sends the edited line or field to the host. If the cursor start position (as explained above) was 1, the entire line/field is sent. If it was not 1, and you never edited the contents of the line to the LEFT of the cursor start position, that part of the line/field from the cursor start position to the end of the line/field will be sent. If you BACKED UP, that is moved to the left of the cursor start position and changed something there, ANZIO will send the necessary BACKSPACES, then the rest of the line/field.

It is all rather difficult to envision - just play with it a bit.

6.2 IN A DEFINED KEY

The functions of the  key are the smallest logical unit. Most of the time the use of  requires some additional jumping around, so it is embedded into a defined key sequence.


As an example, consider the \$EDIT editor in the I-series operating systems. In order to change the current line, we must tell the editor


CH 

This causes the editor to redisplay the line, then move the cursor to the next line and allow us to reenter it. Finally the editor responds by showing us the changed line again. Now let's see how a defined key might make this easier (this sample is included in SAMPLE.KYS, see Appendix F).


We want to define a key to contain:


CH | {  } 



The 'CH|' goes through to the editor and says 'CH_↑↓'. After the editor displays the line and positions the cursor, the defined key takes over again. The brackets ('{' and '}') mean that whatever is between them happens on the screen only, without letting the host computer know about it. The  shown here represents the cursor-up arrow key. So the cursor-up arrow in brackets will move the cursor up a line, unbeknownst to the host.

Finally the  invokes the line editor, using the line where the cursor is now positioned.


6.3 INVOKED FROM THE HOST

The  editor can also be invoked by a command from a program running on the host computer. This is particularly effective when video attributes are being used. This technique allows field-level editing. The procedure works like this:

1. The host program displays a field of data in reverse video, or some other attribute.
2. The host program positions the cursor to the beginning of the field.
3. The host program issues a <hex-1C>F2<hex-1D>. This puts ANZIO into field editing mode.
4. The host program goes into input mode ("ACCEPT").
5. The user can manipulate the data on the screen, inserting characters, etc.
6. When the user hits , the host program receives the edited input.

If the user terminates the editing with an  or , ANZIO will tack that keycode onto the end of edited field. The host program must deal with that, of course.

6.4 THE EDITING KEYS

Following are descriptions of the special editing keys used in the  line editor. Any key that represents a normal character will either TYPE OVER the present character or be INSERTED, according to the INS indicator. Any control or function key not listed below will be ignored.

If you are in "line mode", that is, there are NO video attributes on the line, the line is treated as having two "sections", positions 1 to 72 and 73 to 80. If you are in field mode, you are dealing with only one "section".

<BACKSPACE>

Deletes the character to the left of the cursor, and backs up the cursor. In line mode, if the cursor position is less than 73, the deletion will not affect columns 73-80.



Move the cursor without changing the contents of the line.



If the editor was invoked from the PC, grab data from some other line on the screen. If invoked from the host, send the edited data terminated by the arrow key's code sequence.



Switches back and forth between INSERT and OVERTYPE modes.



Deletes the character at the cursor. Again, if the position is less than 73 (and you are in line mode), the I-D columns will not be changed.



Moves the cursor to column 1 or column 73, depending on which section it is currently in.



Moves the cursor to the position just beyond the last non-space character in the section; that is, it will not jump into positions 73-80 if in line mode.



Clears to spaces all positions from the current cursor position to the end of the section.

<TAB>

Moves the cursor right to the next tab stop.

<BACKTAB>

Moves the cursor left to the next tab stop.







Escapes from the line editor - it does not send back the edited line.



Terminates the process and sends the results back to the host, followed by a <return>.

6.5 A SPECIAL USE: COMMAND LINE EDITING

Have you ever entered a series of commands to the host, or an exceptionally long line, only to discover you misspelled something? One of the features provided with the  key is to allow you to return to a prior line on the screen, grab it, fix it and re-send it to the host.

When you wish to edit a prior line, first hit the . Then hit  one or more times. Each time, the contents of a higher line will appear on your line. When you have grabbed the right line, fix it. When you're done, simply hit the  key and the line you just edited is sent to the host. Try it, you'll be surprised at how often you use it.

7. USING A MODEM

You may need to have your PC talk to your host computer via a modem, over a telephone line. The modem at the PC end can either be an "internal" modem, that is a modem on a card inside the PC, or an "external" modem, connected to the PC's serial port. The following comments apply to both setups.

On the surface, working through a modem is no different from working locally, except that it is probably slower. However, you first have to make the connection to the host machine. To do this, you can either give commands to the modem directly, or use ANZIO's DIAL command.


When you first bring the system up, you will be talking to the modem, rather than to the host computer. Make sure you are in "LOCK OFF" mode, or you won't get far. If appropriate, switch back to "LOCK ON" after you connect. Also, be sure you work in upper case, as the modem may not recognize lower case.

All commands to the modem start with "AT" for "ATTENTION". These commands are NOT entered on line 25, but on the live screen. Remember that ANZIO does not know it is talking to a modem instead of a computer.

The usual procedure is to a) set any necessary parameters in the modem, b) have the modem dial, c) the modem responds when it connects, and puts you on line to the computer, d) you do your work on the computer as needed, and e) you sign off.


7.1 THE DIAL COMMAND

The "DIAL" command was implemented because of the problem of an unanswered line. Without it, it is very difficult for a defined key to know when or if a call was completed. The syntax is:²


```
DIAL <string> <wait>  
      <retries> 
```

or for example:


² ANZIO commands must be entered at the "Func:" prompt; see section 4.


```
DIAL 5551234 100 6 
```

This command tells ANZIO to dial the modem using <string> as the phone number, if the connection is not made wait a certain amount of time (<wait>, in 1/10 seconds), and then try again <retries> times. So the example would send the string "5551234" to the modem (prefixed by certain characters needed to tell the modem that this is a number to be dialed), the modem would dial the number and respond with a response code. If the response indicates that a connection was made, the command is completed. If a connection is not made, the PC would wait 10 seconds, and try 6 more times (always waiting 10 seconds). If the connection was still not made, an error message would appear, requiring an operator response. The <wait> and <retry> parameters are optional.

If something goes wrong in the DIAL process, you can abort the operation by hitting ¹, the panic button.

The parameter <string> can contain, prior to the phone number, any other modem setup characters necessary, such as "T" for touch-tone. These may be entered into a defined key also, such as:²

```
DEFINE D DIAL T05552478 , , , , 123456789 
```

By then pressing "D", ANZIO tells the modem to dial touch-tone, the number 0-555-2478, wait for four counts (a delay to wait for the telephone company's computer to accept a credit card number) and then enter the credit card number 1-2345-67-89.

Each parameter <wait> and <retries> can be anything from 0 to 32767. If either is omitted it is assumed to be 0.

7.1.1 MODEM RESPONSE CODES

The DIAL command determines, by means of a response code from the modem, whether the modem has detected a connection, no answer, etc. It uses the modem's text response codes for this, and it expects a successful connection to include "CONNECT" in its response. Some modems,


¹ If you have redefined function keys to emulate a terminal, see section 3.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.


though, can give a wide variety of response codes, which are not standardized. If ANZIO gives you "BAD MODEM STATUS", you must configure your modem to respond with only the basic response codes. Refer to your modem manual.

7.2 THE WAIT COMMAND

The WAIT command also has been added to facilitate certain types of modem operations. In some applications, after the modems have made their connection a pause is required before proceeding. This command is used for that. The syntax is:²

WAIT <nnn> 


where <nnn> is a number representing tenths of seconds.

The wait can be interrupted by . The number <nnn> can be up to 32767, representing 3276.7 seconds (about 53 minutes).


7.3 SIGNING OFF

Signing off is the process of disconnecting the modems, and "hanging up" the phone line. Either modem can initiate the process, depending on the host operating system. When one modem hangs up, the other will detect that and hang itself up, notifying the computer to which it is connected.

The general convention for making the host end hang up is to enter the command "exit" or "BYE" to the host. This tells the host that the session is finished, and the host tells its modem to hang up (this may not work on all operating systems). When the modem on your end detects the hangup ("carrier drop"), it will give you a response code, and place you back into the mode in which you are talking to the modem.




To hang up the modem from your end, you must regain control of the modem. On the Hayes compatibles, the standard method for this is to enter "+++ " with no . That is, the modem must see these characters with a pause both before and after. The modem will then drop into "command" mode, but the phone line will still be "off hook". Then tell the modem

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

ATH0 

This tells it to go "on hook", or hang up the phone. If all else fails, you can turn off the modem and turn it back on again.

This may be entered into a defined key also, such as:²

```
DEFINE H  WAIT 30 |  
+++  WAIT 50 | ATH | 
```

See Appendix F, SAMPLE DEFINED KEYS.

7.4 BAUD RATE SHIFT

Many modems will do a "baud rate shift", which means the baud rate between PC and modem is different from the baud rate over the phone line (modem to modem). This introduces a chance for problems.

Assume you have a 9600 baud modem, and your ANZIO is set to 9600 baud. You dial into a remote host that has only a 2400 baud modem. Generally, your modem will "sync up" with the remote modem, and start communicating with it at 2400 baud. Your modem (if configured such) can tell you that it connected at 2400 baud rather than 9600.

But now you modem must make a decision: at what rate will it talk to you? First, it can assume you want to talk at the modem-to-modem rate (2400), and start doing so. Or, it can continue to talk to you at 9600, but internally buffer data and do a baud rate shift. The modem makes this decision based on one of its parameters.

If your modem responds the first way (by changing its baud rate to you), there will be problems. ANZIO has no way to recognize this shift, and will from then on fail to communicate with the modem. So, your modem MUST be configured the second way, to do a baud rate shift.

Note that the same problem exists on the host end - if your host end modem changes its baud rate, the host system may no longer be able to communicate.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

7.5 FLOW CONTROL

When a modem is doing a baud rate shift as above, and in other situations as well, flow control is necessary. There are many nuances to flow control: hardware vs. software, whether the software codes are passed through, etc. We are working on this area. Please contact us if you have problems you believe are related to flow control.

8. HOW MANY CHARACTERS CAN I SEE?

ANZIO is very flexible in terms of the number of characters on a line and lines on a screen. If your video driver (the board in your PC that drives the monitor) can support more than the standard 80 by 25 character layout, ANZIO can work with it. Or, if the hardware supports only 80 columns, ANZIO can keep a 132-column "virtual screen" in memory and show you any 80 out of the 132. We will deal with each of these separately.


8.1 SPECIAL SCREEN HARDWARE

Many video drivers support extended character-mode resolutions, such as 80 by 43, 80 by 50, 132 by 25, and 132 by 43 or 50. The most useful of these is 132 by 25, but we'll allow you to work with whatever you want.

Note that there is no STANDARD available to know how to set video boards into these extended modes. So you can't tell ANZIO to set your screen to 132 by 25 without doing some setup first. We'll get to that in a moment. There is a standard, however, by which ANZIO can determine the current number of columns and lines. So at appropriate times, ANZIO will check those settings and see if it needs to readjust itself.

There are three ways to change the video mode. First, you can set it before you run the program. ANZIO will then initialize itself according to that setting. Second, you can use the RUN (or RUN/N) command in ANZIO to run a mode-setting program that was provided with your video board. On return from any RUN, ANZIO checks its parameters and readjusts as necessary. Third, ANZIO has a SCREENMODE command which MAY be able to set your hardware to the mode you want.

Most boards will have a "BIOS mode number" that corresponds to 132 by 25. This information would come from the documentation on your board. Let's say for purposes of demonstration that it's a hex 55. You can then tell ANZIO to set up that mode by doing:²

```
SCREENMODE BIOS-55 
```


You could then switch back to normal color mode (80 by 25) by doing:²

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

```
SCREENMODE COLOR 
```

You can also tell ANZIO what the correct mode number is for 132 by 25 operation, by doing:²

```
MODE-132 xx 
```

where <xx> would in this case be 55. Once ANZIO knows this information, it can switch to 132-column mode in response to a control-code sequence from the host. You could also then switch it manually using:²

```
SCREENMODE 132 
```

Note that the software on the **host** system may not work particularly well with other than the standard screen size. It may assume, for instance, that moving the cursor past line 24 position 80 will scroll the screen. We give you the tools, you figure out if you can use 'em...

8.2 132-COLUMN VIRTUAL SCREEN

ANZIO is equipped to behave as though the PC has a 132-column screen, even when it doesn't. The video driver card dictates how many characters actually show on the screen at one time. This is the "physical" width. If the physical width is 80, however, you can still tell ANZIO to use a 132-column "logical" or "virtual" width. That means that ANZIO will store 132 columns of data internally, and show you any 80 of those 132 columns.

The virtual screen width is set using the WIDTH command. Note that you can set the WIDTH greater than the physical width, but you can NOT have the WIDTH less than the physical width. That is, if your video hardware is showing 132 characters, WIDTH must be 132. For the following discussion, then, we will assume that the screen shows you 80 columns, and WIDTH is 132.

In this mode, you can think of your screen as providing a window to your data. Once there is data outside the normal 80-byte limit, you can see it in one of three ways.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

The PAN command moves your window back and forth horizontally. By entering the following, you can move the ANZIO screen left, right, or so it begins in a certain column:²

```
PAN LEFT
PAN RIGHT
PAN <n>
```

where <n> is the column number.

You can also see this data in REVIEW mode, by scrolling in all four directions. Note also that the `HOME` and `END` key are active in REVIEW mode.

The third way is to PICK from anywhere in the 132-column screen by simply moving the cursor to the left and right, as well as up and down.

A program has been included to demonstrate these abilities, on an NCR I-system. This is a COBOL source program which allows you to go browsing through spool files. The program is called SPOOLCRT.CBL. It must be moved up to the host system and compiled. Then assign A to the spool file and execute the program. It will tell you its commands.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

9. PRINTING

ANZIO can do several kinds of printing, including 1) printing the screen (window); 2) printing text as it is displayed; 3) passthrough print; and 4) file transfer directly to the printer.

In DOS versions of ANZIO, the destination can be any DOS device name, such as “PRN” or “LPT2”. The destination is selected using the PRINTER command. If network software is being used, that software may redirect a particular device name to a shared printer; this is transparent to ANZIO. Note that in DOS, ANZIO knows nothing about the type of printer being used, and so ANZIO can not do such things as changing fonts or line spacing, unless you program such operations into macro keys using printer-specific escape sequences.

In Windows, although DOS device names are an option, ANZIO offers a more powerful and flexible printer, referred to as “WPRN”. The WPRN module takes advantage of Windows’ printer interface, and so it can more precisely control output independent of printer type. So in Windows, the PRINTER variable will ordinarily be set to “WPRN”.

9.1 HOW TO PRINT

9.1.1 PRINTING THE SCREEN

ANZIO can print the currently active screen at any time. This can be the “live” screen, showing the emulation session to the host, or any part of it. Use the PRINT command for this. It can also be an internal ANZIO screen, such as the HELP, INTERPRET, KEYS, or any screenful of the REVIEW buffer, by variously entering “P” or “PRINT” from the keyboard.

Printing the screen does not automatically eject the page -- see the explanation of the FLUSH TIMER below.

WIN only

In Windows, you can also print the screen using the menu item File:Print Screen, or its accelerator **ALT** **P**. Note that the menu item always goes to the WPRN module, regardless of the setting of PRINTER. As such, it is subject to the settings for printer selection, font, Print Wizard, etc.

9.1.2 CAPTURE TO PRINTER

ANZIO can capture to the printer any text coming to the screen (this text will also show on the screen). This gives the user a way to start and stop printing without altering what happens on the host.

In DOS, enable print capture with `OPENO LST:` followed by `CAPTURE ON`; turn it off with `CAPTURE OFF` and then `CLOSEO`¹.

In Windows, enable print capture with the menu item `Transfer:Capture to Printer`, or issue the command `CAPTURE WPRN`.

9.1.3 PASSTHROUGH PRINT

Passthrough print is also known as “transparent print” or “slave print”. It is controlled from the host system, using escape sequences that are dependent on the terminal type being emulated. It is transparent in the sense that the text is not shown on the screen.

Because ANZIO supports the accepted protocols for passthrough print for all the terminal types it can emulate, it will work very well where the host system already supports passthrough print. If you would like to do passthrough print but do not have host support for it, contact Rasmussen Software for more information.

9.1.4 FILE TRANSFER TO PRINTER

For those who are creative, you can do file transfer (using Kermit, etc.), specifying an output file named “LPT1”, “LST:”, or “WPRN” as appropriate.

9.2 THE WPRN PRINTER MODULE

This section applies to WINDOWS only.

Windows versions of ANZIO contain a module referred to as “WPRN”, which translates various streams of text characters to be printed into calls to Windows’ printer application programming interface (API). In addition to

¹ ANZIO commands must be entered at the “Func:” prompt; see section 4.

serving as a bridge between two approaches to printing, it provides access to the flexibility and features provided by Windows.

9.2.1 PRINTER SETUP

The File:Printer Setup menu in ANZIO brings up a standard Windows dialog box, that allows you to make certain settings that affect printing, such as “print quality”. Also, you can select “Setup...” to bring up an additional box with additional options. From there, you can go to “Options” and set printer-specific options.

All these settings will be set into place, and used the next time you have ANZIO print something. In addition, they will be tracked between sessions in ANZIO’s settings file.

9.2.2 CHOOSING A PRINTER

Because a Windows PC often has access to more than one printer, there is a way in ANZIO to select a printer. You can direct ANZIO either to use a specific printer, or to use whatever printer is the default. To choose the printer, select File:Printer Setup, then choose “Setup”.

9.2.3 CHOOSING A FONT

Use the menu item File:Printer Font to choose a font and size for your printing.

All fonts are available to you, but some will work better than others. Fonts that are fixed-space will do a better job of lining up columns. TrueType fonts will give you more options for size. OEM fonts will do a better job of printing line-drawing characters. A good choice meeting all these requirements is “MS Linedraw” from Microsoft, available with many of their software packages and also available for download. Another alternative is “Courier New”, although it does not contain line-drawing characters and so ANZIO will use “+” and “-”.

The font size you choose will directly affect horizontal spacing (therefore characters per line) and vertical spacing (therefore lines per page), UNLESS you are using the Print Wizard feature (see below). So if you switch from printing an 80-column page to a 132-column page, you will

probably want to change your font size from 12 point (10 cpi) to about 7 point.

9.2.4 THE FLUSH TIMER

What is the end of a print job? At what point are you finished printing, and ready to close the job, letting it work its way through the Windows spooler and out to the printer? In a passthrough print situation, nothing from the host indicates the end of a print job. So ANZIO has the FLUSH TIMER.

You can set the FLUSH TIMER to a certain number of seconds. The default is 5. That means that if something has been sent to the printer, and 5 seconds go by without anything more being sent to the printer, ANZIO assumes you are done printing, and it flushes the job. So 5 seconds after you do a screen print, or 5 seconds after your last passthrough print data comes in, the information is printed on the printer.

If passthrough print data arrives sporadically from the host, with pauses in between, you may need to increase your FLUSH TIMER. You may want to disable this feature (by setting FLUSH TIMER to zero) -- in this case, the job will be closed only when you do a File:Eject (menu item) or an EJECT command, or when you quit from ANZIO.

9.2.5 LOW-LEVEL PRINT

The Windows model for printing, used in the WPRN module, is to “draw” text at various places on the page. If the data coming from the host computer in a passthrough print operation contains control codes, these will be drawn on the page instead of being obeyed. Likewise, if a host program (e.g., Word Perfect for UNIX) is generating PostScript code and doing passthrough print, the PostScript code will be printed on the page, rather than obeyed. That’s not what you want.

So ANZIO offers you Low-level Print, in the File menu. If this is turned on, ANZIO will send characters to the printer at a lower level, similar to writing to a device in DOS, although spooling still happens.

If the data from the host contains escape code or PostScript code, turn Low-level Print on.

9.2.6 THE PRINT WIZARD

The Print Wizard is an attempt to make ANZIO's print handler intelligently deal with a variety of kinds of printer data. This is an issue primarily for passthrough print.

When the Print Wizard is turned on, it will store and analyze data to be printed, and automatically deal with such things as 80-column versus 132-column reports, backspace-underlining, embedded escape codes, and line wrap.

Print Wizard also gives you a unique ability to control the printer, making it possible for a host-based program to specify everything from paper orientation to inclusion of bitmaps, in a device-independent manner.

9.2.6.1 AUTOMATIC TEXT HANDLING

ANZIO's Print Wizard feature is intended to solve a problem. The problem is that print data arriving at its passthrough print channel can be structured many different ways. There can be many unstated assumptions as to how the "printer" will react. The Print Wizard will analyze the print data, figure out those assumptions, make some decisions based on reasonableness and convention, and print out the data. The goal is to have an acceptable printout in a wide range of circumstances.

9.2.6.2 PRINT WIZARD'S MARKUP LANGUAGE

The second aspect of Print Wizard provides the programmer a way to control ANZIO's printing. This ranges from job-level changes, such as setting the paper orientation and point size, to precisely-specified items such as bitmaps and rectangles.

The approach ANZIO uses for this is a markup language based on HTML, the language used for web page design. We can't say that it **is** HTML, because a) HTML is changing too fast, and b) ANZIO extends HTML in certain areas, notably positioning items precisely on actual pages of paper.

Separate documents available from Rasmussen Software explain in detail ANZIO's markup language.

10. USING REVIEW

ANZIO has the ability to track prior screens of information that have scrolled by. These prior screens are kept in review memory and are accessible through the REVIEW command. ANZIO attempts to allocate up to 64KB to use for this purpose. Each character position takes two bytes, so that means you can have a maximum of 402 lines in 80-column mode, and 244 lines in 132-column mode.



Ordinarily, ANZIO will store in its review memory any lines that are scrolled off the top of the screen. In addition, if you turn SCROLL ON, then any screens that are ERASED will be "scrolled" instead.

If you have WIDTH set to 132 (on an 80-column screen), you can also use REVIEW to look to the right (or left) at characters that are out there in the virtual screen.

To initiate REVIEW mode, enter²

REVIEW 

Once in REVIEW mode, only the following keys will work:

 or 

restores the working screen and exits REVIEW mode.



moves your window up



moves your window down

 , 

move the window up or down one screenful

 , 

move the window back and forth one column, if possible

 , 

move the window to the far left and far right, respectively





P


prints the current screen

REVIEW can also perform a split-screen function. That is, you can go in to REVIEW, scroll back a ways, find something you want to keep on the

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

screen, and lock it in. That becomes the "REVIEW window", which covers up anything you might have in that area of the screen. The following keys, used in REVIEW mode, manipulate the REVIEW window:

-  opens the REVIEW window, locking it onto the screen
-  closes the current REVIEW window locked on the screen, and returns you to normal REVIEW mode
-  shrinks the size of REVIEW window on the screen
-  expands the size of the REVIEW window on the screen

Once a review window is opened over the top of the original screen, you must re-enter REVIEW in order to close that window with the  key.

PART II. DATA CAPTURE / FILE TRANSFER

11. DATA CAPTURE/FILE TRANSFER

Utilizing the PC means more than just terminal emulation. People want the power to capture data directly from the host, either right off of the live screen or through file transfer. ANZIO allows you to pick data off the screen, or transfer ASCII sequential files, including spool files, to and from the host.

11.1 DATA CAPTURE


ANZIO can capture what is on the screen in a number of ways. One of the most popular ways is with the PICK command, which allows you to highlight an area of the screen (columns) and write it to an output file on the PC in a Comma Separated Values format (CSV). This is usable by LOTUS 123, SuperCalc and others. If you prefer unformatted screen data, the KEEP command will write out the entire screen or a highlighted portion of the screen similar to the PICK command. It is also possible to print the screen or any portion of it directly on the PC's printer.

11.1.1 ANZIO'S PICK COMMAND

A full description of the ANZIO's PICK command is given in section 19.1.2. Refer also to the description of Q-GRAPH in Appendix B.

11.1.2 THE KEEP COMMAND

This command works the same as the PICK command, except data is not formatted on the PC; the data will appear in the file as it was displayed by the host software. Using ANZIO just as a terminal, run the program to display the information. Open an output file with the old familiar:²

```
OPENO <filename> 
```

Now use the KEEP command to either highlight a column,²

```
KEEP 
```

or to pull the entire screen²

² ANZIO commands must be entered at the "Func:" prompt; see section 4.


² ANZIO commands must be entered at the "Func:" prompt; see section 4.

KEEP 

11.1.3 PRINTING FROM THE SCREEN

The PRINT command allows the user to print what is currently displayed. Only the first "live" lines are printed, not the status line. This is done on the command line:²

PRINT 

It is also possible to print part of the screen, using the  method as described above.

11.2 FILE TRANSFER IN GENERAL

In the following discussion, transfer of data from PC to host is called "upstream" or "upload", and the opposite is called "downstream" or "download".

In many cases we can transmit data directly into existing software on the host system, and it will think we are typing it. In some cases the host will limit us to 80-byte records. In nearly all cases we are limited to passing ASCII data only (no packed numerics, etc.).

The case of transmitting sequential, ASCII records, up to 80 characters, upstream we shall refer to as "simple upload". Other types of file transfer, such as downstream, or involving records longer than 80 bytes, require a special piece of software on the host end, working in conjunction with ANZIO. The operation of these modules is detailed below. Note that we have not covered all the possibilities, every kind of file transfer. Section 11.9 mentions some of these.

You will notice that the only software distribution medium we use is diskette. When a module is needed on the host, we have included it in source form on the distribution diskette. You will need to transmit it up to the host, using the methods detailed in SIMPLE UPLOAD, and then compile it. We assume you know how to compile a program; if not, contact us, or

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

get some help from a software house in your area. IF YOU DO NOT HAVE A COMPILER, please contact us.

11.3 IMOS II FILE TRANSFER

This system has very few remaining users, so we have a couple of holes in our coverage. Contact us if you need to work in the IMOS-II environment.

11.4 IMOS III, IMOS V, IRX, ITX

11.4.1 I-SYSTEM SIMPLE UPLOAD

The simple upload is accomplished by having the PC transmit into either \$EDIT or SYSIN (IRX/ITX only). Here is the first method (issue these commands to the host through ANZIO's live screen, not to ANZIO itself):

```
AS A <filename>(<unit>),  
    NEW,<size>,AP  
AS EWF EWF(<unit>),OW  
EX $EDIT  
IN*
```

Then tell ANZIO:²

```
OPENI <pcfilename>  
TRANSMIT TRAILER \
```

When the transmission finishes, tell the HOST:

```
SA A  
QU
```



Here is the second method (for ITX only). Tell ITX:

```
AS A <filename>,NEW,<size>,AP  
MOV SYSIN A
```

Then tell ANZIO:²

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.




```
OPENI <filename>   
TRANSMIT TRAILER END$ 
```


And away it goes.

As mentioned, this method is used to transfer the source modules necessary for other types of file transfer. The simple upload is very fast, and is usually the preferred method of uploading when record length is not a problem.

IF YOU HAVE PROBLEMS with missing characters, or with the process locking up, you need to adjust DELAY and/or LINE DELAY. See section 12.

To compile a COBOL source on an I-system:










```
AS SI <sourcename>(<unit>)   
AS BO <objectname>(<unit>),  
    NEW,<sectors>,AP   
EX $COBOL 
```




The "\$COBOL" above might alternatively be "\$COBOL9" or "\$COBOL85". If you get a message indicating a printer is not assigned, just hit .

11.4.2 SEND-PC: I-SYSTEM DOWNLOAD

The program on the diskette as SEND-PC.CBL is a very flexible download program. It is designed to read a variety of file types and transfer those files to the PC. It is also customizable - if the file you need to transfer is not covered, you can put in your own specs - see comments in the source file itself.

To reiterate, this source must be uploaded to the host system and compiled. Following is a complete example, placing it on unit 3:

```
AS A SEND-PC-S(3),NEW,200,AP   
MOV SYSIN A   
  OPENI SEND-PC.CBL   
  TRANSMIT TRAILER END$   
AS SI SEND-PC-S(3) 
```

AS BO SEND-PC(3),NEW,200,AP 
EX \$COBOL 


This download process will produce a TEXT FILE on the PC. That is, each record will become a "line", with a carriage-return/linefeed after it (unless switch 3 is ON), on the PC. If there are non-ASCII characters in the file, such as packed numerics, results are unpredictable. If the file being read is a spool file, the program will convert page positioning information into the appropriate line feeds and form feeds.

The following file specs are supported by SEND-PC:

BLOCK	FIX/VAR	RECORD
512	V	80
512	V	510
480	F	80
512	F	128
512	F	256
512	V	132
512	V	134(spool file)
512	F	512
512	V	256
470	F	94
512	V	87(spool file)
512	V	90(spool file)
495	F	95

Several switch settings affect SEND-PC's behavior.

SWITCH 1

ON tells the program to use "alternate codes" (see below).

SWITCH 2

ON tells it to strip trailing spaces off each line before sending it.

SWITCH 3

ON tells the program NOT to put a CR/LF on the end of each record.

SWITCH 7




ON tells the program it is running on an early version of ITX (before 4.0)

SWITCH 8

ON tells the program to filter control codes out of the data. This slows the transfer down considerably.


SWITCH 1, as stated above, implements alternate codes. Use this if you are using modems or multiplexors that interfere with passage of XON and XOFF characters (hex 11 and 13)

The SEND-PC program takes advantages of ANZIO's ability to receive commands from the host. SEND-PC does all the OPENs and CLOSEs necessary. To run the program, follow these steps (issue these commands to the host through ANZIO's live screen, not to ANZIO itself):

```
SET SWITCH 2, 3 OFF   
AS A <filename>(<unit>)   
EX SEND-PC(<unit>) 
```

The program will ask you for the name of the file to be created on the PC. Answer that question - you may include a full path if needed. If the file named already exists, the program will ask if you want to delete it. If you answer "N", the program will terminate without transferring the file. To transfer to the printer, use a file name of "LST:".

Now all you have to do is sit back and watch. If you don't want to see it work, and want it to run somewhat faster, you can enter this command to ANZIO:²

```
RECEIVE QUIET ON 
```

If you get a file status 95 or 98, it means that the file you are trying to transmit doesn't match one of the file specs of the SEND-PC program.

It is also possible with SEND-PC to send a group of files, by assigning a parameter file containing the names of the files to be transferred. Create a file on the host system, using \$EDIT, that contains one line for each file to be transferred. At the very least, each line must contain the I-system name, with disk unit, in columns 1 through 30. The following fields are optional:

SHORTEN, col. 31

Tells whether to strip trailing spaces. "Y" for YES, "N" for NO, space indicates follow the switch setting.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

CRLF, col. 32

Tells whether to put carriage-return/linefeeds on the PC file. Enter "Y" or "N", or space to default to switch setting.

PCNAME, cols. 33-72

The path and name of the file to be created on the PC.

When used with a parameter file, SEND-PC checks three additional switches:

SWITCH 4

ON means to delete an existing file on the PC.



SWITCH 5

ON means to ignore an existing file on the PC. Switch 4 takes precedence over 5. If neither is ON, and a collision occurs, you will be asked what you want to do.


SWITCH 6

ON means to bypass a host file name that is not found, and go on to transfer other files in the parameter file. OFF would cause the program to stop and ask whether to continue.

To use the parameter file with SEND-PC, tell the host system:

```
AS PRM <prmfilename>(<unit>)   
EX SEND-PC(<unit>) 
```



Finally, it is possible to tell SEND-PC from the ITX command line what file to transfer:

```
EX SEND-PC(unit) <itxname>(<unit>)  
    <pcfilename> 
```

(This is entered all on one line).

11.4.3 RECV-PC: I-SYSTEM UPLOAD

This program is designed to upload records that may be longer than 80 bytes. It is set up for a variety of file specs, and others can be added as per comments in the source file. Operation is very similar to SEND-PC (issue these commands to the host through ANZIO's live screen, not to ANZIO itself):

```
AS A <filename>( <unit> ),NEW,  
    <size>,AP   
EX RECV-PC( <unit> ) 
```

The program will ask you for the name of the file on the PC to be uploaded. Answer the question, using a unit designator if needed.

The program will then ask you what type of file you want to create, listing several options. Select one.

If the file on your PC does NOT have carriage-return/linefeeds between records, it is still possible to upload it, if RECV-PC is told to create fixed-length records of exactly the right size.



RECV-PC at this point does not have options to a) use a parameter file, or b) process command-line parameters.

11.4.4 RECV-SPL: I-SYSTEM UPLOAD SPOOL FILE



RECV-SPL allows a print file on the PC to be uploaded to the host computer and printed on the speedier host printer.

The source code for RECV-SPL is included on the distribution diskette and should be transferred to the host using ANZIO, and compiled as RECV-SPL-O, similar to the other file transfer programs discussed here. The program is designed to upload records for printing, up to 132 characters (132 column width paper). The operational procedure is similar to RECV-PC (section 11.4.3):

To upload a PC file and print it directly on the I-system's printer (issue these commands to the host through ANZIO's live screen, not to ANZIO itself):


```
AS LO (unit,LP)   
EX RECV-SPL-O( <unit> ) 
```

To print out to a spool file for later printing:

```
AS LO <spoolfile>,NE,<size>,SP,AP   
EX RECV-SPL-O( <unit> ) 
```

The program will ask for the name of the print file on the PC to be uploaded. Answer the question, using the unit designator and directory name as needed. At the end of the transfer the program will tell the number of lines received by the host and the number of lines it had to truncate (those over 132 bytes). The program assumes a standard print file on the NCR host of 132 characters in length. However, several PC products handle print files of greater length. RECV-SPL will truncate these records to 132 characters.

As the program transfers the lines, they will appear on the screen and scroll up, if RECEIVE QUIET is OFF. If you do not wish to see these lines and want the transfer to run somewhat faster, you can enter this command to ANZIO:²

RECEIVE QUIET ON 

A note of warning is due here regarding PC print file controls. Certain word processors and spreadsheets allow the user to build files with special characters to control printer spacing, fonts and other options. This could present numerous problems in printing on a host printer, if that printer does not understand the escape (control) sequences being sent to it. Also double printing may be a problem, and may show up as two separate, but duplicate lines, instead of overprinting. This program is not printer specific, but could easily be modified by your own in-house programmers.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

11.4.5 KERMIT ON ITX

Beginning with ITX release 6, a module was included called \$IKERMIT (although it is not officially supported outside of ITX Windows), which implements the Kermit file transfer protocol under ITX. Beginning in ANZIO release 9.5, Kermit is also included.

\$IKERMIT is a server-only implementation of Kermit. To implement it, just tell ITX:


```
EX $IKERMIT 
```

At this point, ITX will only accept Kermit server commands. That means you cannot simply type commands to it to tell it what to do, but must use ANZIO's Kermit commands, such as KSEND, KRECEIVE, and KCOMMAND FINISH (see section 12.2).

\$IKERMIT assumes you want to transfer to and from disk unit 0. There are two ways to override this. On startup, you can tell it to work with unit 3, for instance, by telling ITX:

```
EX $IKERMIT 3 
```

Or, once \$IKERMIT is running, you can change its working disk by using its CWD (Change Working Directory) command. To do this, tell ANZIO:²

```
KCOMMAND CWD 3 
```

where "3" is the disk unit you need.


\$IKERMIT also assumes you are dealing with a TEXT file, as opposed to a BINARY file. To override this assumption, invoke it with the "B" parameter:

```
EX $IKERMIT B 
```

To terminate \$IKERMIT, tell ANZIO:²

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

KCOMMAND FINISH 


We are still exploring \$IKERMIT's operation, and we may be putting together more information. Feel free to contact us.

11.5 UNIX FILE TRANSFER





There are several; approaches to file transfer between ANZIO and UNIX, as detailed below. Note, however, that the COBOL programs (such as SEND-PC.CBL) are NOT intended for UNIX, and will probably not work.

11.5.1 UNIX SIMPLE UPLOAD

The following steps will upload a file into UNIX. Note that this procedure is effectively limited to records less than 255 bytes. First, enter this command to UNIX:

```
cp /dev/tty <filename> 
```



Then, enter these commands to ANZIO:²

```
OPENI <pcfilename>   
TRANSMIT TRAILER   
```

the sample macro key "u" in SAMPLE.KYS, as listed in Appendix F.

If UNIX loses some characters, you may need to boost ANZIO's DELAY setting. The DELAY is necessary because no handshaking is being used.





As a further example, following are the steps to transfer the SEND-PC.C program to UNIX, and compile it. First, tell UNIX:

```
cd /usr   
cp /dev/tty send-pc.c 
```


Then, tell ANZIO:²

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

```
OPENI SEND-PC.C   
TRANSMIT TRAILER   
```

Wait until the transfer is finished, then enter (to UNIX):


```
cc send-pc.c -o send-pc 
```

The last line tells UNIX to compile a "c" program (cc) named "send-pc.c", and put its compiled output (-o) in a file named "send-pc".

If you do not have a "C" compiler ("cc"), we may have provided you with objects (executable programs) for your particular UNIX system. Check the README file on the distribution diskette.

11.5.2 SEND-PC.C: UNIX DOWNLOAD


This C program included will transfer virtually any UNIX text file down to the PC (do not try to use it on an RM/COBOL indexed file). Upload it and compile it as explained above. To use it simply enter:

```
send-pc <unixfilename> <pcfilename>  

```


The program will delete an old <pcfilename>, create a new one, and send the file down.

11.5.3 DOWNLOAD: UNIX DOWNLOAD USING SHELL SCRIPT

As an alternative means of downloading a file, or a group of files, from UNIX, especially for those who don't have a C compiler, we have included the shell script DOWNLOAD. First, move it to the UNIX system using the simple upload procedure above. Then, make it executable by issuing this command to UNIX:

```
chmod +x download 
```

DOWNLOAD will take any file or list of files and download it (them) to ANZIO, using the same file name for the PC as on the host. EXISTING FILES OF THE SAME NAME WILL BE DESTROYED! As an example, the following command (to UNIX) would download the specified COBOL files:

download *.cbl 

11.5.4 RECV-PC.C: UNIX UPLOAD

The program RECV-PC.C is included because the simple upload procedure is limited to 255-byte records. RECV-PC has no limitation.

Transfer the program up and compile it, as explained above. Then, to transfer a file, just enter to UNIX:

recv-pc <unixname> <pcname> 

11.5.5 KERMIT WITH UNIX


The Kermit file transfer protocol is available for UNIX, for a small distribution charge. Kermit allows transfer of many kinds of files between many kinds of systems, with error checking, data compression, and more. Contact us for information on obtaining Kermit for your UNIX machine.

ANZIO contains several commands used to communicate in a Kermit mode: KSEND, KRECEIVE, and KCOMMAND. These are listed in section 12.2.


Kermit on UNIX should come with instructions, but because it is not especially easy to use, we'll give some pointers here:

PARITY

If your UNIX system is set up for even parity, as many are, you will need to set up Kermit that way too. To do so on startup, tell UNIX:

kermit -p e 

Or, once Kermit is running, you can give it the command:

set parity even 

Finally, it is possible to put the "set parity even" command in a file named ".kermrc" in your user home directory.

TEXT vs. BINARY

UNIX Kermit must know whether files being transferred are TEXT (which translates between **linefeed** for end-of-line on UNIX and **return-linefeed** for end-of-line on DOS) or BINARY (which does no translation). To see its current setting, tell UNIX Kermit:

```
sh 
```

To set it, tell UNIX Kermit:

```
set file type <ftp> 
```

where <ftp> is text or binary.

SERVER MODE


For less confusion, we recommend putting UNIX Kermit in "server" mode, by telling it:

```
server 
```

Then, give it commands using ANZIO's Kermit commands. See section 12.2 for explanations of the commands KSEND, KRECEIVE, and KCOMMAND. To terminate the server mode, tell ANZIO:²

```
KCOMMAND FINISH 
```

To terminate Kermit itself, tell Kermit:






```
quit 
```

11.6 RM/COS

11.6.1 RM/COS SIMPLE UPLOAD

For uploading, we will use the ability of the RMCOS system to copy from the terminal to a file. First, tell RM/COS:

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

CR 
(for LOGICAL NAME, enter )
(answer following questions as appropriate)
FC 
(for SOURCE, enter "ME ")
(for DESTINATION, enter the pathname of the created
file, then )

Then tell ANZIO:²

OPENI <pcfilename> 
TRANSMIT 





Wait until the transfer is finished, and tell RM/COS:


EXIT 





If data errors occur, you may need to boost the DELAY factor.

11.6.2 SEND-RM: RM/COS DOWNLOAD

This program will read an 80-byte variable file and transfer it using the CAPTURE procedure. Upload the program and compile it, as explained above. To use it, tell COS:

(assign the object file)
(assign "A" to the file to be transferred)
EXEC 
SEND   

Then wait for the program to pause, and tell ANZIO:²

OPENO <pcfilename> 
CAPTURE ON 
LOCK ON 


² ANZIO commands must be entered at the "Func:" prompt; see section 4.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

Wait for completion, then tell ANZIO:²

CLOSEO 
CAPTURE OFF 
LOCK OFF 


and tell COS:

X
REL  

This is not as cumbersome as it looks, if you set up some defined keys.

11.6.3 SEND-L.RM: RM/COS LONG DOWNLOAD

This program will read a 510-byte variable file and transfer it using the CAPTURE LONG procedure. This means the program will break the record into pieces to send it, and ANZIO will put it back together. The procedure is exactly the same as for SEND-RM, above, except that you must specify





CAPTURE LONG 

instead of CAPTURE ON.

11.6.4 RECV-PC.RM: RM/COS UPLOAD




This program allows you to upload records up to 510 bytes long into RMCOS. It produces a file that is 510 bytes variable, although every record is 510 bytes long (padded with spaces). As with the other programs discussed here, it can be adjusted by a programmer to create files with other specifications.

Follow these steps. First tell COS:


(create the destination file using "A" as the logical file name)
(assign the object program)
EXEC 
RECV-PC   

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

Then tell ANZIO:²

```
OPENI <filename>   
LOCK ON   
TRANSMIT LONG  
TRAILER END$$$ 
```

Wait for completion, then tell ANZIO:²

```
LOCK OFF 
```

and tell COS:

```
REL  
```

You may need to adjust your DELAY factor, if you get errors in your received file.

11.7 VRX

Simple upload should work going into VRX's OLPD editor, or other simple input programs. We do not support file transfer with VRX at this point - that is up to the user. However, the programs are simple to adapt to any VRX system for your own specific needs. Each program only requires minimum changes.

11.8 PC-TO-PC

I'll be honest with you, this was an afterthought.

It is possible to connect the serial ports of two PCs together, either directly (with some wire jumpering) or through a modem, and move ASCII files between them. But no guarantees.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

On the receiving PC:²

```
OPENO <filename>   
LOCK ON 
```

On the sending PC:²

```
OPENI <filename>   
PLAY NCR ON   
LOCK ON   
TRANSMIT 
```

When finished, close the file on each PC.

Any file can be moved from one PC to another with little or no trouble when there is no common media. Set up the sending PC the same as above, except use "TRANSMIT CRC" instead of "TRANSMIT". On the receiving PC, use "RECEIVE CODED". Again, no guarantees.

11.9 OTHER KINDS OF FILE TRANSFER

There are, as mentioned, some variations on file transfer we have not provided. The most common of these requirements involve either a) using host-generated data with word-processing and spreadsheet programs on the PC, and b) moving more complex kinds of files between PCs and hosts. More information on the first is given in section 19. If you would like to discuss specific requirements, please give us a call. We do have two programs available which may be of help to you.

11.9.1 QUICKSEND

A popular use of ANZIO is to bring data from the host's files down to the PC in order to use it in a spreadsheet or word-processing system. In order to do that, some extraction and data conversion must generally take place on the host. In other words, you don't want to bring your entire file down to the PC, only some selected data.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

QuickSend is designed for this purpose. It is a low-end report generator program, with a special verb "SEND" that causes extracted data to be sent to the PC, in conjunction with ANZIO.

QuickSend can also be useful for migration purposes, where the destination system is NOT in COBOL, and therefore the data needs to be in ASCII unpacked format, either in fixed fields or in comma-separated-values format.

QuickSend is extremely quick to set up and get operational, and will handle many of the tasks necessary for this kind of data link. It is available for ITX, IRX, and IMOS V.

11.9.2 UFT

UFT stands for Universal File Transfer. Two versions of UFT are available, one for I-systems and one for UNIX.

UFT for I-systems runs on an IMOS III, IMOS V, IRX, ITX, or RMCOS system, in conjunction with ANZIO on the PC. It deals on the host side with any file: indexed, relative, or sequential, even object programs. It also deals with packed-numeric and other eight-bit non-ASCII data (which is transferred transparently). On the PC side, it deals with text files as well as binary files and RM/COBOL-compatible binary sequential files. It also does error checking during transfer.

UFT for UNIX runs on a UNIX system, in conjunction with ANZIO on a PC. This combination provides an easy and fast way of moving text or binary data, 8-bit data, with compression, error checking, and wild-carding.

With UFT on different host systems, you can use the PC as a stepping stone to move files from one system to another. When you put all the parts together, you obtain the ability to transfer:

- I-system to I-system
- I-system to/from RM/COS
- I-system to/from RM/COBOL on PC
- RM/COS to/from RM/COBOL on PC
- I-system to/from UNIX

PART III. REFERENCE GUIDE

12. COMMANDS

Listed below are all the text commands that ANZIO can process. These can be 1) entered manually; 2) part of a macro; or 3) sent from the host computer.

Note that some commands apply only to DOS, Windows, serial, or network, as indicated. Also, many commands are not in the Lite version.

12.1 COMMANDS BY TYPE


Following is a list of ANZIO's commands by type. See section 12.2 for an explanation of each command.

12.1.1 OPERATOR PREFERENCE ITEMS

BEEP	Tells ANZIO whether to beep
BEEP IDLE	Beep when idle
COLOR	Set operating colors/attributes
CURSOR BLINK	Allows non-blinking cursor
FONT	Sets the screen font size
GAUGE	Gauge line at bottom of screen
JUMP	Set jump scrolling
PITCH	Set pitch for beeps
PRINTFONT	Sets printer font size
PRINTLOW	Sets low-level print
SCREENMODE	Various hardware screen modes
SCREENMODE/S	Report screen info to host
STATUS LINE	Status line on/off
TITLE	Sets window title
TRACK-WINDOW	Should Anzio save its window position?

12.1.2 COMMUNICATION PARAMETERS

7E1, 8N1, etc.	Set data bits, parity, stop bits
ANSWERBACK	Sets the terminal's answerback string
AUTO-LF	Auto-linefeed
BACKSPACE	Configure <Backspace> key
BAUD	Set baud rate
BREAK	Send a break
CHARSET	Sets character set translation
COMMTYPE	Set to SERIAL or WINSOCK

DATA BITS	Set data bits
DELAY	Delay between characters
DELAY/S	Reports DELAY settings
FULL DUP	Set duplex
HALF DUP	Set duplex
HOLD	Suspends host output
IGNULL	Ignore null characters
IMOS	Set host operating system
INTERPRET	Communication diagnostics
IRQ	Set interrupt vector
IRX	Set host operating system
ITX	Set host operating system
LINE DELAY	Delay on line turnaround
LOCK	Keyboard locking protocol
MONITOR	Communication diagnostics
PARITY	Set parity
PORT	Set communication port
RECONNECT	Should we reconnect on line drop?
RESET	Reset terminal
RMCOS	Set host operating system
RTS-MODE	For use with odd comm equipment
SCROLL-LOCK	Does the  key suspend host output?
STOP BITS	Set stop bits
SYNC	Remove screen "snow"
TAB	Set tab key handling
TERM	Set terminal emulation type
TERMNAME	Sets term type to be reported to host
TTY	Set host operating system
UNIX	Set host operating system
UPPERCASE	Forces keystrokes to upper case
VRX	Set host operating system

12.1.3 FILE TRANSFER

CAPTURE	Capture data coming to screen
CLOSEI	Close input file
CLOSEI/S	Close input file, report to host
CLOSEO	Close output file
CLOSEO/S	Close output file, report to host
DELETE/S	Delete file, report to host
FIND/S	Find a PC file name
FINDNEXT/S	Find next PC file name

KCOMMAND	Send Kermit command
KEEP	Copy part of screen to a file
KEEP/N	Copy part of screen, without terminators
KRECEIVE	Receive a file with Kermit
KSEND	Send a file with Kermit
OPENI	Open an input file
OPENI/S	Open an input file, report to host
OPENO	Open an output file
OPENO/N	Open an output file, even if it exists
OPENO/S	Open an output file, report to host
PICK	Pick screen columns for spreadsheet
PLAY NCR	PC-to-PC communication
PURGE	Clear file transfer buffer
RECEIVE CODED	For special kinds of file transfer
RECEIVE QUIET	Don't display file transfer
RETRANSMIT	Retransmit last element
TIMEOUT	For use with UFT
TRANSMIT	Simple file upload
XN	Send next record
ZRECEIVE	Receive a file using Zmodem
ZSEND	Send a file using Zmodem

12.1.4 LOCAL PROCESSING

BOX	Draw a box on the screen
CALC	Invoke the calculator
CALL	Macro key "subroutine"
CD	Change logged disk/directory
CHOOSEPRINT	Opens Printer Setup dialog box
CLIP	Copy to clipboard
COPY	Copy a file
COPY/S	Copy a file, result to host
DATE	Send date to host
DEFAULTS	Save settings
DEFINE	Make a macro
DELETE	Delete a file on PC
DELETE/N	Delete a file, ignore "not found"
DIAL	Dial a modem
DIR	PC disk directory
DIR/S	PC disk directory, result to host
DROPOUT	Exit from ANZIO without resetting port
EJECT	Eject a page

END	Quit
ENV/S	Sends environment variable to host
F2	Line/field editor
FILL	Fill screen area
FLUSH	Release a print job
FLUSHTIMER	Set timed release of print jobs
HELP	Get help on ANZIO
HOSTNAME/S	Sends host name to host
HOTKEY	Set hotkey for TSR use
INVOKE	Start a macro key
KEYS	Show special and macro keys
LAUNCH	Start another program in Windows
LOG	Change logged disk/directory
MENUBAR	Create a menu
MERGE	Merge in auxiliary macro file
MESSAGE	Displays a message box
MKDIR	Make a directory
MKDIR/S	Make a directory, result to host
MODE-132	Set mode for 132 by 25
PAN	Move right/left in virtual screen
PASTE	Paste Windows clipboard text
PLAYSOUND	Plays a WAV file
PRINT	Print all or part of screen
PRINT/N	As PRINT, but without CR/LF
PRINTER	Which printer to use
PRINTER-SETUP	Configure PC's printer
PRINTFILE	Print a text file
READ	Read a key (macro) file
RENAME	Rename a PC file
REVIEW	Scroll back data from top of screen
RUN	Shell out to DOS
RUN/N	Shell to DOS with no exit keystroke
SAVE	Save keys (macro) file
SCROLL	Save erased data
SEND	Transmit (part) of the screen to the host
SETCOLOR	Reset current color
SLEEP	Wait until certain time
STAY	Go to TSR mode
STAY/G	Go to TSR mode, space for graphics
STOP	Quit
TIME	Send time to host

TYPE	Show a PC file on screen
VERSION	What version of ANZIO?
VERSION/S	Tell host what version of ANZIO
WAIT	Wait a specified time
WAITFOR (WF)	Wait for specific text on the screen
WIDTH	Allow 132-column virtual screen
WINDOW	Open a window
WINDOWCLOSE	Close a window
WINPRINT	Print a file in Windows
WINSTART	Start a Windows program
WRITE	Write data to file

12.2 COMMANDS ALPHABETICALLY

This section lists all the commands in ANZIO. The command listing shows the exact form (syntax) of the command. A vertical bar (|) indicates alternate commands in the same group. Square brackets ([]) indicate optional parameters. Anything in angle brackets (<>) must be replaced by your appropriate entry.


- 7E1
Shortcut equivalent to DATA BITS 7, PARITY EVEN, and STOP BITS 1.
- 7E2
Shortcut equivalent to DATA BITS 7, PARITY EVEN, and STOP BITS 2.
- 7N1
Shortcut equivalent to DATA BITS 7, PARITY OFF, and STOP BITS 1.
- 7N2
Shortcut equivalent to DATA BITS 7, PARITY OFF, and STOP BITS 2.
- 7O1
Shortcut equivalent to DATA BITS 7, PARITY ODD, and STOP BITS 1.

- 7O2
Shortcut equivalent to DATA BITS 7, PARITY ODD, and STOP BITS 2.
- 8E1
Shortcut equivalent to DATA BITS 8, PARITY EVEN, and STOP BITS 1.
- 8E2
Shortcut equivalent to DATA BITS 8, PARITY EVEN, and STOP BITS 2.
- 8N1
Shortcut equivalent to DATA BITS 8, PARITY OFF, and STOP BITS 1.
- 8N2
Shortcut equivalent to DATA BITS 8, PARITY OFF, and STOP BITS 2.
- 8O1
Shortcut equivalent to DATA BITS 8, PARITY ODD, and STOP BITS 1.
- 8O2
Shortcut equivalent to DATA BITS 8, PARITY ODD, and STOP BITS 2.

ANSWERBACK <string>

Sets the ANSWERBACK to <string>. The ANSWERBACK can be sent to the host, on host request, in certain terminal emulations. If the ANSWERBACK requires a <RETURN>, code it as “\r”.


AUTO-LF [ON] | AUTO -LF OFF

Sets ANZIO to automatically insert a line-feed when a return () is encountered. This may be used when communicating with non-NCR host equipment.

BACKSPACE 8 | BACKSPACE 127

Different host systems require different codes for backspace. ANZIO can send either a DEL (decimal 127, hex 7F) or a BS (decimal 08, hex 08, ctrl-H) for backspace (or any other decimal value). This command sets that value.

BAUD <nnnn>

Sets the Baud rate of the connection. All the standard rates (50, 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 9600, 19200, 28800, 38400, 57600) are available, should you need to use them. The current baud rate is displayed by the  (HELP) key.

BEEP [ON] | BEEP SLOW | BEEP OFF

These control the beeper on the PC. BEEP SLOW is the default, sounding the beep only when the host issues a beep and unlocks the keyboard, and there is nothing to be sent (e.g. a defined key). BEEP ON sets the beeper to sound whenever a beep is sent from the host. BEEP OFF silences the PC.

BEEP IDLE [ON] | BEEP IDLE OFF

BEEP IDLE ON sounds the PC beep once per minute, when the host is idle and expecting input; for example, when online and at the command level. BEEP IDLE OFF turns this feature off.

BOX <x1> <x2> <y1> <y2> [<type>]

This command draws a box on the screen from (<x1>, <y1>) to (<x2>, <y2>). As <type>, use SINGLE or DOUBLE (default is SINGLE).

Any or all of the parameters can be cursor-relative, rather than absolute, such as BOX .-1 .-5 10 20 .



BREAK



The BREAK command sends a break to the host computer.


For ITX users, this opens up the possibility of having a defined key do an ITX logon. (The break is held for 300 milliseconds.) For an example, see Appendix F, the "B" key.

CALC

This command brings up a four-function calculator on line 25. This allows you to do some arithmetic without grabbing a calculator. The calculator can work in decimal or hex mode.

The calculator is a floating-point model similar to most desktop units in function. Numbers are entered in free format, with no assumed decimal points. The  key functions as a '+' key. The  key causes the 'clear' function, clearing first the entry and then the total. '*' is used for multiplication and '/' for division.

When you are finished calculating, hit the  key. This will return you to your live screen, while leaving the result on line 25. Or, you can hit the  key, which will exit the calculator and send the RESULT (the contents of the accumulator) to the host computer. The result will be retained in the calculator the next time you enter it.

The 'H' key puts you into hexadecimal mode, as indicated by 'HEX' on the line. 'H' again puts you back in decimal mode. Changing modes always converts the present total. In hex mode, you can add, subtract, multiply, and divide. If you hit , the result is always sent to the host in the current mode.

CALL <macro>

This command allows one macro to start another macro. When the second macro finishes, the first (calling) macro will resume. See also INVOKE.

CAPTURE [ON] | CAPTURE LONG | CAPTURE OFF

The CAPTURE feature is a means of saving an image, usually on disk, of all displayable data that comes from the host. When CAPTURE is ON, each time the cursor moves to a new line, the screen line that it was on will be stored as a record in the receive buffer. This is the same buffer used for normal downward data transfer, so it is not practical to do CAPTURE while doing file transfer.

At any time that there is data in the receive buffer, and an output file is open, the captured data will be written out to the output file.

CAPTURE is also used as the primary means of file transfer in an RMCOS environment. CAPTURE LONG is a variation of this, used when records longer than 80 bytes are being transferred.

WIN only

CAPTURE WPRN captures the received data to the Windows printer driver.

CD [*<unit>*:]*<dirname>*

Same as LOG.

CHARSET *<name>*

Sets the National Replacement Character (NRC) set to *<name>*. If *<name>* is "INTERNATIONAL", no NRC translation will be done. Other possible values are: 'UK', 'FINNISH', 'FRENCH', 'CANADIAN', 'GERMAN', 'ITALIAN', 'SWEDISH', 'SPANISH', 'NORWEGIAN', and 'SWISS'.

CHOOSEPRINT

WIN only

Brings up the Windows Printer Setup dialog box, so the user can select (and setup) a printer.

CLIP [*<x1>* *<x2>* *<y1>* *<y2>*]

WIN only

Copies the indicated rectangle (or the whole window) to the Windows Clipboard.

CLOSEI

Closes the input file. See OPENI.

CLOSEI/S

This command also closes the input file, but sends a response code to the host indicating completion. Possible results are:

00 : Completed successfully

01 : Error occurred

CLOSEO

Closes the output file. See OPENO.

CLOSEO/S

This command also closes the output file, but sends a response code to the host indicating completion. Possible results are:


- 00 : Completed successfully
- 01 : Error occurred

COLOR

This command allows the user to specify which colors/attributes are to be used, both for normal text and for highlights and so forth. The terminal which ANZIO is emulating can have four video attributes: reverse video, half intensity, underline, and blink. These may be used in any combination, making 16 combinations.

PCs generally do not have all these options. In color PCs, in fact, some of these options are replaced by different colors.

ANZIO has a "translation table" which translates each combination of emulated attributes into a parameter byte that is used at the lowest level on the PC. The COLOR screen allows you to manipulate that translation table.

In the COLOR screen, just hit the key for the attribute you want to change (0 through 9, A through F), followed by the two-byte color/attribute code you want, selected from the table of all possible values shown at the right. Repeat this as necessary, then hit  to exit.

You can also reset all colors to our defaults, as indicated.

COMMTYPE SERIAL | COMMTYPE WINSOCK

WIN only

Tells ANZIO whether to communication with the host system using a serial connection or a WINSOCK (Windows Sockets) connection. Changing this value will cause an existing connection to be dropped.

COPY <filename> <newfile>

Similar to DOS' "copy" command - copies a file. There is NO check for whether <newfile> already exists; if it does, it will be overwritten.

COPY/S <filename> <newfile>

Functions like COPY, but returns a status code to the host. Possible variations are:

00 : Completed successfully

01 : Error occurred

CURSOR [BLINK] [ON] |

CURSOR [BLINK] OFF


Tells ANZIO whether you want the standard blinking cursor, or the non-blinking pseudo-cursor.

DATA [BITS] 5 | DATA [BITS] 6 |

DATA [BITS] 7 | DATA [BITS] 8

Sets the number of data bits in the communication protocol. Generally 7. Notice that this bit count does NOT include the parity bit, if PARITY is EVEN or ODD.


DATE

Tells the program to send the current PC date to the host. This allows a defined key to be set up for automatic log-on with entry of date and time. The date is sent in format YY/MM/DD .

DEFAULTS

Tells the program that you want to save the current parameter settings. You will be prompted for the name of the file to save into. See section 1.8.

DEFINE <x> <text>

The Define command associates key <x> with <text>, so that whenever you enter x, the <text> is sent to the host. For more information refer to section 5.

DELAY <nnn>

Sets the time delay between characters sent to the host. This is necessary on some systems to prevent overloading the communication hardware on the host. <nnnn> is a number from 0 to 65535; larger numbers cause more delay; zero is no delay. Unlike in earlier versions of ANZIO, DELAY is now in units of 10 microseconds, regardless of PC speed.


See also LINE DELAY.

DELAY/S

Causes ANZIO to report to a host program what its DELAY and LINE DELAY settings are.

DELETE <filename>

Deletes the specified file on the PC. The filename may include a disk identifier, such as:

```
DELETE B:ANYFILE.XYZ 
```

If ANZIO can not find the file indicated, an error message will be generated.

DELETE/N <filename>

This command is the same as DELETE, except that an error message is not generated in the event that the indicated file is not found.

DELETE/S <filename>

This variation of DELETE returns a status code to the host. Possible results are:


- 00 : Completed successfully
- 01 : Error occurred

DIAL <string> [<wait> [<retries>]]

Instructs ANZIO to dial out on a modem, using <string> as the phone number, and if no connection is made to wait a period of time indicated by <wait> in tenths of seconds, and retry <retries> times. See section 7.

DIR [<pathname>]

Displays a directory of a disk unit, subdirectory, wild-card directory, etc. Format is the same as at the DOS prompt.

Once a screenful of files is shown, you can move around the highlight, to focus on a particular file. You will then see that file's size and creation date and time at the bottom of the display. If you position the highlight on a subdirectory entry and hit , you will move into that subdirectory, and ANZIO will display it.

Note that with the directory information on the screen, you can type in PRINT, and have it printed.

DIR/S [<pathname>]

Behaves as DIR, above, but allows you to select a file name and send it either to the host or to another function. See section 5.8.5.5.

DROPOUT

Exits from ANZIO, just like END or STOP, except the program does not restore the communication port to its original state. This is for use in unusual situations only.

END

Causes the program to terminate, and return to the operating system. Same as STOP. Can be abbreviated to "E".

EJECT

Ejects the page on the printer.

ENV/S <variable>

Looks up <variable> in the DOS environment, and sends its value to the host, without any terminator.

F2

Invokes the line editor. section 6.

FILL <x1> <x2> <y1> <y2> <char>

Fills a rectangular area of the screen with the character in <char>, or SPACE if no <char> is given. Any or all of the first four parameters can be relative to the cursor position, such as FILL .-3 .+3 etc.

FIND/S <filespec>

Causes ANZIO to report to the host the first file name matching the wildcard entry in <filespec>

FINDNEXT/S

Causes ANZIO to report to the host the next file name matching the wildcard entry in the last FIND/S command.

FLUSH

Causes ANZIO to flush all data to the PRINTER, by closing and reopening the printer file. If spooling is being used, on a network printer and/or through Windows, this will cause the print job to be released so it can be printed.

FLUSHTIMER <value>

Sets the flush timer to <value>, in seconds. Default is 5 seconds. When this number of seconds has elapsed since Anzio sent something to the printer, Anzio will execute a FLUSH command automatically, thereby allowing the data to be printed.

To disable this feature, set to zero. You might want to do this if you wanted to print two screen dumps per page, or if your pass-through print data was getting split up into multiple jobs, for instance.

FONT <size> | FONT LARGER | FONT SMALLER

WIN only

Changes the Windows screen font size. Anzio will change the window size to follow the font size. You can specify a <size> as either <height> or <height>x<width>. Note that Anzio will not necessarily find the indicated size available.

FULL [DUP] | HALF [DUP]

The DUPLEX setting tells the program either that a keystroke sent to the host should be also placed on the screen by ANZIO (HALF DUP), or that ANZIO should allow the host to echo back the keystroke to the screen (FULL DUP). If you are getting double characters, you need to set for FULL DUP. If you are not getting characters on the screen at all, you need to set for HALF DUP.


GAUGE [ON] | GAUGE OFF




This controls the gauge line on the bottom of the screen. The gauge line displays '....|..*.1.*..|....2...' etc. to help determine columns, useful in text entry. The gauge line also shows the cursor line and position.

HALF [DUP]

Sets Half Duplex mode. See FULL DUP.

HELP


This will display many available commands, any open files, and the current option settings. You may also use the  key.

At the end of the HELP screen, you may either enter  to return to your "working" screen, or you may enter another command, just as though you had hit  (do NOT hit  in this case). After this command is processed, either the HELP screen will be redrawn, or you will return to the working screen.

Note that the PRINT command at this location will print the HELP screen, not the live screen.

HELP <keyword>

Tells the program to give you some information on a particular command (those listed in this section). In this case the <command> must be spelled out completely - it can not be abbreviated. See section 2.

If you do HELP INDEX, ANZIO will present you with an index of help topics. You can then select one and hit  to see the available information.

HELP ASCII presents a chart of ASCII characters.

If your initial disk is no longer in the system, the help file will not be accessible.

HOLD [ON] | HOLD OFF | HOLD TOGGLE

When HOLD is ON, output from the host to the screen is suspended. The HOLD TOGGLE command can be attached to a macro key. See also SCROLL-LOCK.

HOSTNAME/S

Causes ANZIO to send its hostname to the host.

HOTKEY <xyy>

DOS only

Specifies what hotkey combination will be used to pop up ANZIO after the STAY command. The <xyy> parameter is a keycode combination, where x indicates shift modes and yy is a keyboard scan code (all in hex). Shift modes are:

8 = alt
4 = ctrl
2 = left shift
1 = right shift

For keyboard scan codes, do a HELP HOTKEY. ANZIO's standard HOTKEY is **[ALT] [F1]**, coded as 83B. Another alternative would be <ctrl-leftshift-A>, coded as 61E.

IGNULL [ON] | IGNULL OFF

Tells Anzio whether to ignore nulls (hex 00) in the incoming data stream. This item should be OFF unless instructed otherwise by Rasmussen Software.

IMOS

Sets IMOS mode. See ITX.

INTERPRET

Displays the 2048-character input buffer, that is, the last 2048 characters that were received from the host. Non-printable characters appear as reverse-video hex numbers. This display is of value when something strange has occurred at the terminal, as it shows the input conditions.

You can print what you see by entering "PRINT".

INVOKE <macro>

Tells ANZIO to start the indicated macro. This allows one macro to start another. Control does NOT return to the first macro in this case. See also CALL.

IRQ <n>

DOS only

Tells ANZIO what interrupt level (IRQ) the current PORT is set for. Note that if you do a PORT command, ANZIO will change your IRQ automatically to the standard value for that port. Therefore, if you are using a non-standard setup, you will need to then do an IRQ command.

ITX | IRX | IMOS | RMCOS |
VRX | UNIX

These commands tell ANZIO which operating system is running on the host. This primarily has to do with keyboard locking (See section 14.1). The HELP key, **F5**, will show the current operating system. The DEFAULTS procedure will save this setting for future use.

JUMP OFF | JUMP MEDIUM | JUMP FAST

WIN only

Tells ANZIO how to deal treat rapid incoming data that scrolls the screen. JUMP OFF shows every line. JUMP MEDIUM repaints the screen when the screen is one half screenful behind. JUMP FAST repaints only when the screen is a complete screenful behind.

KCOMMAND <command> [<parameters>]


Used to have ANZIO send a Kermit command to a host that is in Kermit server mode. The results of the command will depend on the host Kermit implementation. ANZIO supports the following commands:



- KERMIT <command>
- CWD <newdir>
- DIRECTORY <spec>
- ERASE <spec>
- FINISH <spec>
- HELP <spec>
- LOGIN <spec>
- JOURNAL <spec>
- COPY <file1> <file2>
- LOGOUT <spec>
- MESSAGE <spec>
- PROGRAM <spec>
- QUERY
- RENAME <file1> <file2>
- TYPE <filename>
- USAGE <param>
- VARIABLE
- WHO

KEEP [<x1> <x2> <y1> <y2>]

Sends whatever is currently on the screen to the output file. This file must have been opened with OPENO, below. If you OPENO LST:, the printer will be your output file.

If you just enter KEEP, the entire live screen will be copied to the output file. You can also specify a part of the screen, by giving the coordinates of the corners of a rectangle. This is, of course, not too easy, so we have given you a shortcut. Enter

KEEP 


You now have a rectangle that occupies one character location in the center of the screen. Use the arrow keys to move one corner of this highlight so that it covers the data you want to keep. Hit  again to nail that end down and pick up the other end. Move that corner around. When you have highlighted your area, hit , and the coordinates will appear in the command line.

Parameters can also be relative to the cursor position, as in BOX.

KEEP/N [*<x1>* *<x2>* *<y1>* *<y2>*]

Same as KEEP, except that line(s) written out are not terminated with *<return><linefeed>*.





KEYS

Shows all currently defined keys and their text (see DEFINE above). Also shows the system-defined function keys, and any not-obvious editing keys for use with  and with the command line editor.

This information can be printed by typing "PRINT" at the bottom of the screen.



KRECEIVE [*<filespec>* [*AS <filespec>*]]



When connected to a host system running Kermit, tells ANZIO to receive files. The following command formats are acceptable:

```
KRECEIVE <file name> 
KRECEIVE <filename>
  AS <filename> 
KRECEIVE <wildcard> 
KRECEIVE <wildcard>
  AS <wildcard> 
```

KSEND *<filespec>* [*AS <unixfilespec>*]

Tells ANZIO to transmit one or more files to a host running Kermit. The following command formats are acceptable:

```
KSEND <filename> 
KSEND <filename>
  AS <filename> 
```

KSEND <wildcard> 
KSEND <wildcard>
AS <wildcard> 

LAUNCH <program> [<parameters>]

WIN only

Starts another program, similar to RUN, but ANZIO does not wait for the other program to complete.


LINE [DELAY] <nnn>


This parameter allows a delay to be introduced after "unlock" (what constitutes "unlock" depends on the host operating system) is received from the host, before ANZIO is allowed to transmit. That is, it is a line turnaround delay.

<nnn> is a number from 0 to 65535. Unlike in earlier versions of ANZIO, this now represents units of 10 microseconds, regardless of PC speed.

If you are experiencing lockups, or problems with file transfer, particularly with I-systems, you will need to experiment with LINE DELAY and DELAY settings.

See also DELAY.

LOCK [ON] | LOCK OFF | 

Determines when to use normal NCR protocol, which 'locks' the keyboard when the host is not expecting entry. LOCK ON mode is the normal for I-systems; some programs, like free-standing utilities and modems, require LOCK OFF mode, which sends all characters as they are entered, never locking the keyboard. To temporarily unlock the keyboard, enter . See Keyboard Locking below, section 14.1.

For UNIX, use LOCK OFF.

LOG [`<unit>`:]`<directory>`

Allows you to change the "logged" or default disk unit and subdirectory. The logged unit is assumed whenever no unit and/or directory is specified for local operations such as DIR and OPENO.

The disk unit must be followed by a colon, to distinguish it from a subdirectory entry. Syntax for subdirectory entries we will leave for general books on MSDOS.

MENUBAR `<x1>` `<x2>` `<y1>` `<y2>` `<x3>` `<x4>`
`<off>` `<len>` [`<cols>`]

Superimposes a sliding menu bar system over the data presently on the screen, in one or more rectangles: (`<x1>`, `<y1>`) to (`<x2>`, `<y2>`) and (`<x3>`, `<y1>`) to (`<x4>`, `<y2>`). If a single-column menu is needed, just make `<x3>` equal `<x1>` and `<x4>` equal `<x2>`. If more than two columns are needed, use the `<cols>` parameters. When the user has selected an entry, ANZIO will send back the contents of the screen at offset `<off>` and length `<len>` of the item selected. See Appendix B.

Cursor-relative parameters are allowed - see BOX

MERGE `<filename>`

Reads in a file of macro keys, just as READ does, but does not change the default key file name. Thus, a subsequent SAVE will save to the prior file name. Also, MERGE will leave your existing macros defined (unless they are superseded by some in the new file).

MESSAGE `<string>`

Pops up a box containing `<string>`. When the box is acknowledged, it is removed.

MKDIR `<dirname>`

Creates a directory, similar to the DOS command of the same name.

MKDIR/S `<dirname>`

Creates a directory, similar to the DOS command of the same name. Reports its success to the host as:

00 : Operation successful
01 : Failed

MODE-132 <xx>

DOS only

Tells ANZIO what BIOS mode on your particular screen hardware corresponds to 132 by 25 characters. <xx> must be in hex. This allows ANZIO to set itself into 132-column mode in response to a command from the host. See section 8.1.

MONITOR [ON] | MONITOR OFF

MONITOR allows ANZIO to emulate the monitor command found in many CRTs. This gives the you the chance to see exactly what is coming down the line from the host. Generally, you must also turn LOCK OFF in order to work in this mode. All control characters are displayed in hex notation.

OPENI <filename>

Opens the specified file on the PC's disk for INPUT, to send to the host (see File Transfer, section 11). The <filename> can optionally include a unit designator and/or a subdirectory designator. If none is included, the logged drive and subdirectory will be used (see LOG, above).

Only one input (and one output) file can be open at a time.

OPENI/S <filename>

This command is intended to be sent from the host computer. It behaves the same as OPENI, above, except that a result status is sent to the host. Possible statuses are:

00 : Operation successful
01 : File not found
02 : Input file already open

OPENO <filename>

Opens the specified file on the PC for OUTPUT. <filename> may optionally include a unit and/or a subdirectory designator, otherwise the logged unit will be assumed.

Only one output (and one input) file can be open at a time.

If the file already exists on the PC, you will get a message to that effect. You may want to use DELETE to delete that file, or see OPENO/N below.

Output files are used for a) file transfer (see section 11), b) CAPTURE, c) KEEP, d) PICK, e) WRITE, and f) pass-through printing.

If you OPENO LST:, the printer will be your output file. In this way you can do file transfer from a host computer directly to the PC's printer. The logical printer name that will be used by ANZIO in this case is determined by the PRINTER setting as explained below. You can also open as output any DOS device, such as COM2 or LPT2.

OPENO/N <filename>

This command performs the same as OPENO, except that IF a file already exists with the name specified, it will be DELETED WITHOUT A MESSAGE before the new one is created.

OPENO/S <filename>

This variation of the OPENO command is intended to be sent from a host computer. Instead of giving error messages to the operator, it sends a result status to the host, as below:

- 00 : Operation successful
- 01 : File already exists
- 02 : An output file is already open
- 03 : File or subdirectory error

PAN <n> | PAN LEFT | PAN RIGHT

The PAN command within ANZIO works in conjunction with the WIDTH command and REVIEW (see sections on the REVIEW and WIDTH command and section 10). Once there is data outside the 80-byte limit, you can see it using the PAN command to move your window horizontally to the left or right limit or to column <n>.

PARITY EVEN | PARITY ODD |
PARITY OFF

Sets the parity for communication. Most NCR systems use EVEN.

PASTE

WIN only

Tells ANZIO to paste clipboard text data to the host.

PICK <x1> <x2> <y1> <y2> <type> [...]

The PICK command is a powerful and easy to use way of capturing host system data for use with LOTUS 123 and other tools on the PC. It allows you to capture one or more columns of numbers or labels off the screen, after that information has been put there by any program running on the host, and write it to the current output file in a format that LOTUS can read in directly. For more information, see section 19.1.2.

Parameters can be cursor-relative; see BOX.

PITCH <n>

Tells ANZIO what pitch (frequency) to beep at. Try pitches in the range 200 to 2000. Now, if you have several PCs in the same room, you can tell which one is beeping by having them beep at different pitches.

PLAY NCR [ON] | PLAY NCR OFF

Setting the ANZIO program to PLAY NCR ON causes it to act like the NCR host, to send a file to another PC. See section 11.8.

PLAYSOUND <filename>

WIN only

Causes Windows to play a WAV sound file.

PORT <n>


Specifies which hardware port is to be used for communication.

In Anzio for DOS, changing the PORT causes ANZIO to automatically change the IRQ (if appropriate). If you are using a non-standard IRQ, you will need to issue an IRQ command after changing the PORT.

Refer to section 1.2.

PRINT

Prints the current screen display on the printer. Only the lines of the "live" screen are printed, and video attributes are ignored. The logical printer name is that defined by the PRINTER command below.

It is also possible to print the screen showing during certain "local" operations. While in the middle of REVIEW, just type "P". When prompted at the bottom of INTERPRET, DIR, HELP, KEYS, or TYPE, enter the word "PRINT" followed by .

PRINT <x1> <x2> <y1> <y2>

Prints a portion of the screen to the PC's printer. The portion printed is in the rectangle indicated by the parameters. See KEEP.

Parameters can be cursor-relative; see BOX.

PRINT/N <x1> <x2> <y1> <y2>

Same as PRINT of a rectangular region, but ANZIO will not write out a <return><linefeed> after each line.

PRINTER <printer-name>

This parameter tells the program what logical device name is to be used as the standard printer. This is usually PRN, LPT1, or some derivative of one of them, depending on the PC.




The standard printer is used for the PRINT and PRINTER-SETUP commands.

It is also possible to set up a disk file name in lieu of a printer, in which case printer output would go to that file. One important point must be observed, however: no checking is done to avoid deleting an existing file of the name specified.

WIN only:



Setting PRINTER to WPRN tells ANZIO to use its internal Windows printer driver.






PRINTER-SETUP

This function allows you to type anything and have it go directly to the PC's printer. You can, in effect, use the system as a typewriter. You will need to follow each  with a <linefeed>, which can be generated using the  key. You can also enter the necessary control codes to set a printer into compressed pitch, etc. Any non-printable characters will show on the PC screen as their hex equivalents. This function is terminated by hitting the  key.

PRINTER-SETUP <text>

This command lets you send any text or control characters to the printer. It is particularly useful inside a defined key, as a way to set the printer into special modes (compressed pitch, for instance).

Remember that when entering a function, you can embed any special code by prefixing it with a . So, to send  <ctrl-N> to your printer (codes vary by printer, of course), you would enter:²

PRINTER-SETUP 
 
 

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

As another example, suppose you want to define the <F12> key to print the screen, and then eject a page. You could do:²

```
DEFINE <F12>  PRINT |  
 PRINTER-SETUP   | 
```

PRINTFONT <SIZE>

WIN only

Tells ANZIO's "WPRN" printer driver what character size to print in. The <size> can be in the form of <height> or <height>x<width>.

PRINTFILE <FILENAME>

Causes ANZIO to print the indicated text file, by copying it to the printer named in the PRINTER setting.

WIN only:

If the specified printer is "WPRN", and PRINT WIZARD is turned on, the Print Wizard will inspect the data to be printed, and will set the line spacing, character size, and margins automatically to make the document fit well on the page.

PRINTLOW [ON] | OFF

WIN only

Tells ANZIO's "WPRN" printer driver whether to do low-level print.

PURGE

This command causes ANZIO to delete all data stored in its "received file" buffer. Data is put into that buffer by a) CAPTURE and b) file transfer. Ordinarily, if data is building up in the buffer (as shown on the HELP screen), it is because an output file has not been opened to place the data into.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

READ <filename>

Reads in a previously SAVED file of defined keys. This new data REPLACES any keys you already have defined. The specified <filename> becomes the new assumed key file name, so if you do a SAVE without specifying a name, it will be used. See also MERGE.

RECEIVE QUIET [ON] |
RECEIVE QUIET OFF

Tells ANZIO that when it is receiving or sending data during file transfer, that data should not be displayed on the screen. Thus it is not "quiet" per se, but "invisible". Downloading with this parameter on speeds up the operation somewhat. This setting also affects functions sent from the host.

RECEIVE CODED

Tells ANZIO that the file being received is coded and that all information should be written out to disk as a coded file. This has its use in transmitting from one PC to another (see section 11.8).

RECONNECT [ON] | RECONNECT OFF

NET only

This setting tells ANZIO how to behave when the host system drops its end of the connection, such as when you log off of a UNIX host. If RECONNECT is ON, ANZIO will immediately try to establish a new connection (leading to a new login prompt). If OFF, ANZIO will quit.

RENAME <oldname> <newname>

Renames a file on disk from <oldname> to <newname>.

RESET

Resets several aspects of terminal emulation: character sets, attributes, and wrap mode.

RETRANSMIT

Causes the PC to resend the last file record it transmitted (during file transfer only), or the last function response code.

REVIEW

Allows the user to see what has scrolled away off the top of the screen. See section 10.


RMCOS

Sets the program to talk to an I-TOWER. See ITX above.

RTS-MODE 0 | RTS-MODE 1 |
RTS-MODE 2

DOS only

This communication parameter should be set only by those familiar with communication protocol. It governs the behavior of the "request-to-send" line which is part of the RS232 communication protocol, and generally can be ignored except when working with certain types of half-duplex modems or other special communication environments. The settings are:

- 0 : RTS stays high always (standard mode)
- 1 : RTS stays high until a  is sent
- 2 : RTS goes on then off with each character

RUN [<program> [<parameters>]]

This special command causes ANZIO to initiate another program. That is, ANZIO will stay in place (in memory), and another program (an EXE or COM program) will be run on the PC. When that other program is finished, control will return to ANZIO, with the screen intact. Thus you can go from terminal emulation to word processing, then return to terminal emulation, for instance.

For DOS: ANZIO will stay in memory, and another program will be run if there is adequate memory.

For Windows: ANZIO will start the other program, and then go to an inactive state until the other program finishes, for compatibility with DOS versions of ANZIO. See LAUNCH for an alternative.

The default for the RUN command is the command processor (such as C:\COMMAND.COM). If you just enter RUN, ANZIO will return you directly into DOS. To re-enter ANZIO, simply enter EXIT within DOS and the screen you were working with on the HOST is returned.

Do NOT run any program which takes over control of the serial port, or any terminate-stay-resident (TSR) programs.

For many programs, you will need to LOG to the disk that contains the program before running it.

It is most efficient if you enter the entire program name including the ".EXE" or ".COM". Otherwise, ANZIO will call COMMAND.COM and let IT find the program, somewhere in the PATH. Note that you can use this feature to run DOS commands, such as²

```
RUN COPY MYFILE PRN 
```

Note that if you have transferred data from the host to the PC with ANZIO, the output file must be closed before that data can be accessed. The HELP screen will show you if you have an output file open.

When you return to ANZIO, it will ask you to hit a key before it restores the screen.

RUN/N [<program> [<parameters>]]

Same as RUN, except that on return to ANZIO, no keystroke is needed.

SAVE [<filename>]

Saves all currently defined keys to the indicated file on the PC. Any existing file of the same name will be deleted. If no <filename> is given, ANZIO will save to the defaults file which was read at startup. If none was used, ANZIO will use the default "ANZIO.KYS".

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

SCREENMODE <string>

This command allows you to make ANZIO switch screen drivers.

DOS only

Possible options are:

COLOR	Standard color 80x25
MONO	Standard monochrome 80x25
MONO/C	Monochrome screen on color driver
BW80	(same)
43X80	80 columns by 43 (50) lines on EGA (VGA)
80X43	(same)
132	132-column mode - requires that MODE-132 has been set
80	COLOR or MONO as appropriate
BIOS-xx	BIOS mode xx (in hex).

The last option can be used to set your screen into a non-standard mode, if it is supported by the normal BIOS interrupt convention.

WIN only

Possible options are

132	132-columns wide
80	80 columns wide

<cols>x<rows>Sets the window to the indicated size. In a network connection, will send the new size to the host if it can accept it.

SCREENMODE/S

ANZIO will report to the host computer several parameters about the screen. This will be returned to the host as one line with several fields:

TYPE	1 character, C for color, M for mono, B for BW, blank if unknown
MODE	2 characters, hex mode as per BIOS
LINES	3 characters, number of lines on screen
COLS	3 characters, number of physical columns
WIDTH	3 characters, number of logical columns

TEXT The mode shown in SCREENMODE on the
 HELP screen

SCROLL [ON] | SCROLL OFF

The REVIEW memory is nice, but often the host software does a "screen erase" instead of scrolling, and so that data is lost. However, if SCROLL ON is used, ANZIO will scroll up the live lines whenever the host says to "erase screen".

SCROLL-LOCK [ON] | SCROLL OFF

If SCROLL-LOCK is on, then  key acts to suspend screen display of output from the host. See also HOLD.



SEND <x1> <x2> <y1> <y2>

This causes ANZIO to send back to the screen the data from the screen at the indicated positions. Note that <y1> MUST EQUAL <y2>, that is, this will send data from one line only. The format is done this way for consistency with other commands.

SETCOLOR <n> | SETCOLOR NORMAL

Sets the present working color to a particular decimal value, or hex value represented as 'xxH'. Does not affect the attribute/color table (see COLOR). Use SETCOLOR NORMAL if you accidentally get set into a strange color.

SLEEP <hh> <mm> <ss>

The SLEEP command tells ANZIO to go to sleep until a certain clock time is reached by the PC. This gives ANZIO the ability to wait until a certain time, then "wake up" and dial a remote computer. Hitting   will interrupt the SLEEP function.

STATUS [LINE] [ON] | STATUS [LINE] OFF

Tells ANZIO whether to put status information on the bottom line of the screen. This includes CAPS lock, NUM lock, keyboard LOCK, and communication errors.

STAY

DOS only

Causes ANZIO to go memory-resident (TSR). Does NOT save enough memory to pop up over graphics-mode programs. See section 1.9.2.

STAY/G

DOS only

Causes ANZIO to go memory-resident (TSR), saving enough memory to allow it to popup over graphics-mode programs. See section 1.9.2.

STOP

Stops the ANZIO program and returns to the operating system. Same as END.

STOP [BITS] 1 | STOP [BITS] 1.5 |
STOP [BITS] 2

Sets the number of stop bits in the communication protocol. Usually 1.

SYNC [ON] | SYNC OFF | SYNC FAST

DOS only

Some video adapters (notably CGA) produce "snow" if screen updating is not synchronized with horizontal retrace. Try SYNC OFF. If you see snow during operation, then use either SYNC ON or SYNC FAST. SYNC FAST is similar to SYNC ON, but it blanks the screen temporarily during scrolling, resulting in faster scrolling but more flicker.

TAB <i> <j> <k> ...

Sets the tab stops to columns <i>, <j>, ... The column numbers are entered in order, separated by any non-numeric character. There are ten tab stops; those not specified are not used. Tab stops can also be set with control codes from the host computer.

When the host system sends a control code that represents TAB or BACKTAB, ANZIO moves the cursor forward or backward to the next tab stop.

There is another use of tabs, intended for NCR I-systems. When the TAB key is pressed, and TAB is ON, the TAB CHARACTER (usually space) is sent enough times to move the cursor over to the next tab column. The tab character is initially a space, and the stops are set for COBOL source entry, that is, the initial state is equivalent to

TAB 8 12 15 18 21 24 27 30 33 73

TAB CHARACTER <x>

This allows you to override the usual tab character, which is space. Thus if you enter **F6** TAB CHARACTER Q, whenever you hit a TAB, the program will send one or more Q characters to the host.

TAB [ON] | TAB OFF

Turning the tab function off means the tab key itself will go through to the host, rather than being interpreted as above.

TERM <termtyp>

Tells ANZIO what type of terminal to emulate. Current options are:

N7900	NCR 7900 model 1
N7901	NCR 7901
VIEWPOINT	ADDS Viewpoint (same as N7901)
VT100	DEC VT100
VT220	DEC VT220
WYSE50	Wyse 50
WYSE60	Wyse 60
C332	Versyss C332
SCOANSI	Console of SCO UNIX
AT386	Console of various AT&T derived UNIX systems
ANZIO	ANZIO's own definition, based on VT220


Make sure ANZIO's TERM setting agrees with the host computer's assumption of terminal type.

TERMNAME <name>

NET only

Sets a TERM variable that will be sent to the host as part of the telnet protocol, during the initial phases of the connection. If none is specified, ANZIO will send the name set by the TERM command.

TIME

TIME allows ANZIO to send the time according to the PC's clock to the host (format HH:MM:SS ). Along with DATE, this provides the user with a means of setting up a defined key to log-on and enter the correct date and time.

TIMEOUT <n>

The timeout time (in 1/10 seconds) is used primarily during file transfer. If ANZIO sits locked for the specified time, it will a) beep, b) unlock itself, and c) RETRANSMIT. This can get a file transfer going again after certain communication errors. Primarily used with UFT.

TITLE <string>

WIN only

Specifies the title for the window.

TRACK-WINDOW [ON] | TRACK-WINDOW OFF

WIN only

If TRACK-WINDOW is on, ANZIO will remember the position of its window on the screen, and will automatically readjust to that position the next time it is started. This works, however, only if you save parameters between sessions. Some users prefer to turn this feature off to prevent ANZIO prompting to save parameters on exit.

TRANSMIT [ON] | TRANSMIT OFF

After opening a file for transmission (OPENI above), TRANSMIT ON sends it to the host, one line at a time. TRANSMIT OFF ends transmission; it is set automatically upon reaching the end of the input file. See File Transfer, section 11, for further information.

TRANSMIT CRC

Same as TRANSMIT ON, but causes ANZIO to use a CRC protocol on transmission. For use with special software on the host computer.

TRANSMIT LONG

This is a variation on TRANSMIT ON. It initiates file transfer to the host, but uses a special protocol that takes long records apart and sends them in pieces to the host. The host, of course, must have software (such as RECV-PC.CBL) that understands this protocol and puts the records back together.

TRANSMIT SINGLE

After opening a file for transmission (OPENI above), TRANSMIT SINGLE sends one line at a time to the host, waiting for the host to respond. See the XN command, below.

TRANSMIT TRAILER <string> | TRANSMIT LONG
TRAILER <string>

By specifying a TRAILER, you instruct ANZIO that when it is done transmitting a file to the host, it should send a certain "end-of-file" indicator that the host software will recognize, such as "END\$" or "\". To specify a **DEL** as the trailer, for UNIX, enter it either as **CTRL P DEL** or **ALT <127>**.

TTY

Tells ANZIO that you are working with a bulletin board system or some other unspecified host. See ITX.

TYPE <filename>

Causes a PC file's contents to be displayed on the screen. This is particularly useful for verifying file transmission. Note that if you have just finished a file transmission to the PC, and the output file is still open, the TYPE command will not work properly. In that case do a CLOSEO first.

If the file being TYPed has control codes in it, these will be shown as reverse video hex codes.

The information on the screen during a TYPE operation can be printed by entering "PRINT" at the bottom of the screen.

UPPERCASE [ON] | UPPERCASE OFF | UPPERCASE
TOGGLE

If UPPERCASE is on, keystrokes you hit will be forced to upper case before being sent to the host.

UNIX

Tells ANZIO that the host system is running UNIX. See ITX, above.

VERSION

Tells you which version of ANZIO you are running.

VERSION/S

Tells the host which version of ANZIO you are running.

VRX

Tells the program that the NCR host is running the VRX operating system. See ITX.

WAIT <nnn>

This command simply tells ANZIO to wait a certain amount of time before sending anything else to the host. The <nnn> parameter is in units of approximately 1/10 second. See section 7.2.

WAITFOR <string> [<timeout>]

This command, which is most often done inside a macro key, causes ANZIO to wait for a certain string of characters to appear on the screen, or for <timeout> seconds to elapse. If <timeout> is not specified, it is assumed to be 24 hours. The <string> value can be in quotes if it contains a space, for instance.

The WAITFOR command, which can be abbreviated to "WF" can be interrupted with **ALT** **A**.

WF <string> [<timeout>]

See WAITFOR, above.

WIDTH 132 | WIDTH 80

ANZIO has a "virtual screen" option, behaving as though it has a 132-column screen, with 80 columns visible at a time. The primary advantage of this is the ability to see an entire spool file. This works in conjunction with the PAN command.

This feature is no longer supported, and may not exist in future versions of ANZIO. The ability to have 132 columns on screen under both DOS and Windows makes this unnecessary.

See also section 8.

WINDOW <x1> <x2> <y1> <y2> [FILL] [BOX |
DOUBLE]

Opens a "window" at the indicated rectangle. This causes ANZIO to store the contents of that area in memory, so it can be replaced later by the WINDOWCLOSE command. If FILL is specified, the area is filled with spaces. If BOX is specified, a single-line box is drawn. If DOUBLE is specified, a double-line box is drawn. The MENUDEMO program demonstrates this feature - see Appendix B.

Parameters can be cursor-relative; see BOX.

WINDOWCLOSE

Brings the most recent WINDOW's old data back to the screen. See WINDOW.

WINPRINT <filename>

WIN only

Causes Windows to print the indicated file using the program associated with its file extension (in the same way the File Manager/Explorer does).

WINSTART <filename>

WIN only

Causes Windows to “open” the indicated file. This can be a program or a data file. If it is a data file, it is opened using the program associated with its file extension (in the same way the File Manager/Explorer does).

WRITE <text>

Writes the specified text to the current output file.

XN

Used in conjunction with TRANSMIT SINGLE, this tells ANZIO to send one more line to the host.

ZRECEIVE [<filename>]

Tells ANZIO to receive a file using the Zmodem protocol. If <filename> is specified, the received file will be stored with that name; if not, the received file will be stored with the same name as on the sending machine.

ZSEND [-a] <filename>

Sends <filename> to the host using the Zmodem protocol. The “-a” option specifies that this is a text file; that information is passed to the receiver, which should convert it by stripping carriage returns from it.

13. MORE ON STARTING ANZIO

13.1 PATHS AND SUBDIRECTORIES

Warning: this section applies only to DOS. When you start up ANZIO, the operating system (DOS) must be able to find the program (ANZIO.EXE). There are three ways this can happen. ANZIO has been written to be able to run in any of these modes, while still being able to find its associated files.

First, you can log to the disk/directory that contains ANZIO, such as:

```
C:   
CD\ANZIODIR   
ANZIO 
```

Second, you can tell DOS where ANZIO is:

```
C:ANZIO 
```

or:

```
\ANZIODIR\ANZIO 
```

Third, the directory that contains ANZIO can be somewhere in your PATH. If you have a hard disk, and are not using PATH, may we suggest you check out a good book on hard disk management.

13.2 COMMAND LINE PARAMETERS & DEFAULT FILES

Command line parameters provide a way to affect ANZIO's operation, such as a) bring up ANZIO with an alternate default file, b) cause ANZIO to automatically invoke a certain defined key, and/or c) restrict ANZIO's memory usage.

For DOS: Command line parameters are typed after the program name.

For Windows: Command line parameters are coded into the program icon.

If no command line parameters are present, ANZIO will try to load its standard defaults file (ANZIO.DEF or ANZIOWIN.DEF). This file, if present, will set ANZIO for all the correct user-settable parameters. In addition, it will tell the program what defined-key file, if any, ought to be loaded.

When ANZIO looks for a defaults file, it looks in up to three places. First, it looks in the present directory. Failing that, it looks in the next higher parent directory. Failing that, it looks in the directory where the ANZIO program is located.

If one or more command line parameters are present, ANZIO decides how to treat them by the length of the "word", and its starting character. A one-character entry will be understood to be a startup macro. A word with more than one letter (character) in it, but NOT starting with a slash, will be understood as the name of a defaults file to be loaded. A parameter starting with a slash is dependent on the next character, as listed below.


13.2.1 PARAMETER FILE NAME

A parameter that a) is more than one character long, and b) does NOT start with a slash, is understood to be the name of a parameter (settings) file to be used INSTEAD of ANZIO.DEF or ANZIOWIN.DEF. Thus:

```
C:\>ANZIO REMOTE 
```

would initiate the program and cause it to attempt to read a defaults file called "remote". So you could have one defaults file with the correct settings for local operations, and one with the correct settings for remote (modem) operations.

A special variation of this command line parameter is

```
A>ANZIO NONE 
```

which loads the program and tells it NOT to look for ANY defaults file. If your defaults file has been corrupted, this is a way to load the program in a "virgin" state, in which case it will ask you some questions about baud rate and so forth, just as at initial installation. This also tells ANZIO not to load any macro file.

13.2.2 AUTO-START MACRO

A command line parameter consisting of only one character will be understood as a defined key (macro) to execute immediately upon startup. This must, of course, be an ASCII key (not a function key). Thus:

```
C:\>ANZIO S 
```

will load ANZIO, which will load its standard defaults file, which will tell it which defined key file to load, which will then be loaded, and finally the "S" defined key (case sensitive) will be automatically initiated.

13.2.3 /M: MEMORY

DOS only

This parameter, of the form:

```
/M: <nnn>
```

where <nnn> is a number, tells ANZIO to limit the amount of memory it uses for scrollback and macro keys. See section 13.3 below.

13.2.4 /H: HOST

NET only

For network connections, this parameter specifies the host system to connect to, overriding the one contained in the parameter file. The format is:

```
/H: <hostname>
```

If you need to specify a host port, add it to the end, after a colon:

```
/H: <hostname> : <port>
```

13.2.5 /D: DEFINE

This parameter allows you to specify a key definition (macro) at runtime. The format is:

/D<key><space><string>

where <key> is the key to be defined (an ASCII key), and <string> is the definition.

13.2.6 /C: CHOOSE

WIN only

This parameter, format simply

/C

causes ANZIO to prompt the user for which “.DEF” file to use.

13.2.7 /T: COMMUNICATION TYPE

WIN only

For Windows versions of ANZIO, this series of parameters lets you specify at runtime which type of communication to use. Parameters are:

/Tt for TCP/IP
/Ts for serial
/Tn for Novell WLIBSOCK
/Tp for PicLan

13.3 MEMORY USAGE


DOS only:

ANZIO uses between approximately 165k and 222k of memory.


If there is not enough memory available, ANZIO will be limited in its use of a) REVIEW memory, b) space for DEFINEd keys, and c) space for received data (during file transfer).

If ANZIO has access to more memory than it needs, it will release that memory when it does a RUN or STAY.

If you wish to restrict ANZIO's memory usage, in order to free more memory for RUN or STAY (memory-resident, TSR) operation, you can tell it how much memory it is allowed to use for the review buffer:

```
ANZIO /M:<nnn> 
```

where <nnn> is a number of bytes between 16384 and 65535. If the number is outside this range, ANZIO will round it to the closer number. So for minimum memory usage, start ANZIO with the command:

```
ANZIO /M:1 
```

13.4 THE "SMALL" VERSION OF ANZIO

DOS only

There is a special "small" version of ANZIO now included as ANZIOS.EXE. This program has been created by removing from ANZIO many features, such as file transfer, REVIEW, and so forth.

We recommend that you use the standard version (ANZIO.EXE) to get everything set up and running. Then exit from ANZIO, saving your defaults.

Now, if you run ANZIOS, you will have a smaller, but limited function, option.

Note that ANZIOS automatically does the equivalent of "/m:1", as explained in 12.3.

PART IV. TECHNICAL REFERENCE GUIDE

14. COMMUNICATION PROTOCOL

14.1 KEYBOARD LOCKING

Note: This section generally applies only to users of the ITX operating system.

For ANZIO to work properly on an NCR I-system, it must depend on certain conventions which NCR uses in its operating systems. The IMOS systems (II, III, and V) issue a keyboard lock after each input, and an unlock when a new input is to be allowed. All I-systems (IRX, ITX and all IMOS) issue a BELL code as an audio prompt for input. ANZIO must therefore know which system it is operating under. This is accomplished by entering **F6** followed by the name of the operating system (IMOS, IRX, ITX, VRX, UNIX, or RMCOS).

ANZIO has a keyboard lock indicator which shows on the status line, unless you have turned STATUS LINE OFF.

VRX, UNIX, and RMCOS do not follow this convention. Therefore, when ANZIO is communicating with one of those systems, the lock feature should be disabled, except during some types of file transfer. Even with the I-systems, in some cases you may wish to disable the locking mode, such as when working with a smart modem.

Both the status of the LOCK mode and the operating system setting can be determined by displaying the HELP screen **F5**.

Occasionally, circumstances may leave you locked when you shouldn't be. An example is in ITX following certain system messages. Should this occur, use function key **F4**¹. This will temporarily unlock the keyboard.

14.2 TRANSMIT PRIORITIES

Whenever the keyboard is unlocked, ANZIO must decide what characters, if any, to send to the host. Because these characters can come from several sources, it may be important at times to know the priorities which ANZIO

¹ If you have redefined function keys to emulate a terminal, see section 3.

gives to these sources. The following list gives highest priority transmission first:

1. IMMEDIATE FUNCTION KEYS: The function keys for BREAK **F10**, UNLOCK **F4**, PANIC **F9**, and FUNCTION **F8** are all processed immediately, regardless of keyboard lock status.
2. RETRANSMIT: If the program has been requested to retransmit the last line from a file, that is the highest priority.
3. FILE TRANSMISSION: Normal file transmission happens next.
4. CALCULATOR RETURN: If the operator has exited from CALC using **F1**, the result is sent.
5. STATUS RETURN: The result status from commands such as OPENO/S is sent.
6. RESPONSE FROM FILE RECEPTION: The handshaking that is part of file reception is sent.
7. DEFINED KEY: The next keystroke in a defined key in progress will be dealt with.
8. BUFFERED KEYSTROKES: Any characters waiting in the 128-character keyboard buffer are sent last.

15. FILE TRANSFER PROTOCOLS

This section contains the nitty-gritty of file transfer protocols, and is intended only as a reference for those who are interested in that sort of thing. Only the operation of ANZIO is explained here - the other side can be determined from the various source programs that are included.

15.1 SIMPLE UPLOADS

During a simple TRANSMIT operation, ANZIO simply transmits each line of the input file as a single entry. ANZIO itself will read the entire record (line), regardless of length, and transmit it. Most host operating systems will only allow 80-byte ACCEPTs, however, so this method is generally used only for 80-byte text files.

Normal handshaking occurs, assuming LOCK ON is in effect. That is, ANZIO waits for an unlock, then sends a line and locks itself, and so on until the end of the file is reached. The file is then closed automatically.

15.2 TRANSMIT LONG

This method is provided to allow transfer of records longer than 80 bytes from PC to host. This is done by splitting each record into pieces during transmission; the program on the host must put it back together.

The entire line on the PC file will be read, regardless of its length. Each segment transmitted includes 3 characters of control followed by up to 77 characters of file data. The first two characters in the transmission indicate the number of characters of file data (01 to 77) in ASCII form. The third character is an ASCII "Y" or "N" indicating whether this segment is the last in a record.

15.3 TRANSMIT CRC

TRANSMIT CRC uses a special protocol for error checking, and is intended only for use with a special program, called UFT, from Rasmussen Software, Inc.

15.4 RECEIVE

File reception in ANZIO is based on techniques used to drive a printer "slaved" to a terminal.

Whatever is received by ANZIO bracketed by a DC2 on the front and a DC4 on the end is directed to the receive buffer. The DC2 and DC4, of course, do not go to that buffer. The receive buffer will be dynamically expanded as needed, to the point where it uses all dynamic memory available. This buffer is also used for CAPTURE, so CAPTURE and reception should never be operating simultaneously.


At some point when a) there is data in the receive buffer, b) an output file is open, AND c) the program is "unlocked", then ANZIO will do three steps. First, it will write the receive buffer out to the output file. Second, it will clear the receive buffer. Third, it will respond to the host with a <return>, indicating that the host can proceed with the next transmission.

Note that in order to end up with "records" on the PC file, the CR and LF codes must also be sent from the host, bracketed by DC2 and DC4.

15.5 RECEIVE WITH CRC

This is a special variation on file reception, that includes an error checking method. It is intended only for use with UFT from Rasmussen Software, Inc.

15.6 CAPTURE AND CAPTURE LONG

CAPTURE takes each line off the screen, as soon as the cursor moves to another line. It only captures incoming characters. The captured information is then written to the output file. If ANZIO is running in an RMCOS environment, it then responds with a , because it assumes in that case that CAPTURE is being used for file transfer.

CAPTURE LONG is a variation of capture that allows lines longer than 80 bytes to be transferred. It uses the same line protocol as is used in TRANSMIT LONG.

15.7 PASS-THROUGH PRINT

Pass-through print in ANZIO emulates that in the various terminals, with some handshaking added.


When ANZIO receives an escape code signifying pass-through print, it goes in to pass-through print (if it gets stuck there, you can use **F9** to get out¹). All following characters will go to the output location (see below) until the terminating code is received.


In determining the output location for pass-through print, ANZIO uses this logic: if an output file has been opened (via OPENO), the data will be written there. Otherwise, it will go to the file named in PRINTER. This approach lets you use the pass-through method to do file transfer to the PC's disk.

If ANZIO is running under RMCOS or UNIX, it will issue XON/XOFF handshaking to the host system, so that ANZIO will not be overrun with data to be printed (by the slower printer).

¹ If you have redefined function keys to emulate a terminal, see section 3.

16. SENDING COMMANDS FROM THE HOST

A powerful feature of ANZIO is the ability to control certain functions on the PC from the host computer. Although virtually any  command (function) can be instituted from the host, the most common use is to automate file transfer operations, removing some of the operator's responsibility (and chance for error).

When ANZIO receives a hex 1C (octal 034) from the host computer, it processes anything following that, up to but not including a hex 1D (octal 035), as an  command. So for instance, if the host sent a line that contained

```
<hex-1C>OPENO DOWNFILE<hex-1D>
```

it would process it just as though you had entered

```
OPENO DOWNFILE 
```

Note that if everything goes as planned, ANZIO will open-output a DOWNFILE. But if that file already exists, an error message will be generated TO THE SCREEN, requiring operator action.

In other words, although a function is initiated from the host, all displays, error messages, and subsequent input will be from/to the PC's screen and keyboard. So some functions are just not practical to be initiated in this way.

Several commands, though, have been specifically designed for this approach. These commands contain "/S", indicating that the status of the command (or a result code) should be sent to the host rather than generate a message to the screen. Section 12.2 explains the response codes these commands generate.

Note that in UNIX, it is possible to code ANZIO commands in an "echo" shell command, such as

```
echo "\0034OPENO DOWNFILE\0035" 
```

This works because the UNIX shell understands "\0034" to mean octal 34, and "\0023" to mean octal 23.

17. ESCAPE SEQUENCES AND STANDARD KEY-CODES

Warning: the following tables are incomplete.

The ANZIO program is first and foremost trying to make your PC act as a terminal does. This means that certain control codes, when received from the host, have special meanings. Also, ANZIO will respond to some additional codes, making it more useful than the terminal it is emulating.

ANZIO emulates several common terminals, as detailed below. Emulation has two parts. First, ANZIO must respond correctly to control sequences coming from the host. Second, ANZIO must issue codes to the host in response to certain special keys, such as arrow keys and function keys. Function keys, however, cause a problem, in that ANZIO has its own use for many of them. For an explanation of how we deal with function keys, see section 1.11.1.

Note that ANZIO does not necessarily respond to ALL codes that the corresponding terminal does. Some we have judged either of little value to our customers, or of great difficulty given the hardware platform that is the PC.

17.1 NCR 7900 FUNCTIONS

Some of the codes here actually emulate the NCR 7930 in 7900M1+ mode, including line-drawing.

Received codes:

<u>Code</u>	<u>Hex</u>	<u>Action</u>
SOH	01	Home (go to top of screen)
STX	02	Unlock keyboard
EOT	04	Lock keyboard
ACK	06	Move cursor right
BEL	07	Ring bell (see BEEP commands, section 12)
BS	08	Same as NAK
LF	0A	Linefeed (cursor down, scroll at bottom)
VT	0B	Vertical position, indicated by next character
FF	0C	Clear screen and cursor home
CR	0D	Return
DLE	10	Horizontal position, indicated by next character
DC1	11	XON or received command

DC2	12	Printer on - following characters up to DC4 go to output file
DC3	13	XOFF or end of received command
DC4	14	Printer off
NAK	15	Backspace (non-destructive cursor left)
SUB	1A	Cursor up
FS	1C	Initiate ANZIO command
GS	1D	Terminate ANZIO command
RS	1E	Initiate ANZIO command
US	1F	Terminate ANZIO command
ESC K	1B 4B	Erase to end-of-line
ESC k	1B 6B	Erase to end-of-screen
ESC l	1B 31	Alternate method for cursor positioning, indicated by next two characters
ESC Y	1B 59	Alternate method for cursor positioning, indicated by next two characters
ESC a	1B 61	Alternate method for cursor positioning, indicated by following ASCII numbers
ESC 0	1B 30	Video attribute field indicated by next character.
ESC 3	1B 33	Begin pass-through print.
ESC 4	1B 34	End pass-through print.
ESC M	1B 4D	Insert line
ESC l	1B 6C	Delete line
ESC g	1B 67	Download function key - IGNORED
ESC (1B 28	Graphics mode ON
ESC)	1B 29	Graphics mode OFF
ESC ';	1B 60 3B	Go to 132-column mode
ESC ':	1B 60 3A	Go to 80-column mode
ESC E	1B 45	Delete character
ESC F	1B 46	Insert character on/off

Keystrokes

Key	Hex	Code
<Home>	01	<ctrl-A>
<Up>	1A	<ctrl-Z>
<Left>	15	<ctrl-U>
<Right>	06	<ctrl-F>
<Down>	0A	<ctrl-J>
	7F	
<Backspace>	as determined by BACKSPACE command	

17.2 ADDS VIEWPOINT (NCR 7901) FUNCTIONS

Some of the codes here actually emulate the NCR 7930 in 7901M1+ mode, including line-drawing.

Received codes:

Code	Hex	Action
SOH	01	Home (go to top of screen)
STX	02	Unlock keyboard
EOT	04	Lock keyboard
ACK	06	Move cursor right
BEL	07	Ring bell (see BEEP commands, section 12)
BS	08	Same as NAK
LF	0A	Linefeed (cursor down, scroll at bottom)
VT	0B	Vertical position, indicated by next character
FF	0C	Clear screen and cursor home
CR	0D	Return
SO	0E	Turn on tagged attribute
SI	0F	Turn off tagged attribute
DLE	10	Horizontal position, indicated by next character
DC1	11	XON or received command
DC2	12	Printer on - following characters up to DC4 go to output file
DC3	13	XOFF or end of received command
DC4	14	Printer off
NAK	15	Backspace (non-destructive cursor left)
CAN	18	This "cursor-on" code from RMCOS is IGNORED.
SUB	1A	Cursor up
FS	1C	Initiate ANZIO command
GS	1D	Terminate ANZIO command
RS	1E	Initiate ANZIO command
US	1F	Terminate ANZIO command
ESC K	1B 4B	Erase to end-of-line
ESC k	1B 6B	Erase to end-of-screen
ESC Y	1B 59	Alternate method for cursor positioning, indicated by next two characters
ESC 0	1B 30	Video attribute indicated by next character will be tagged attribute
ESC 3	1B 33	Begin pass-through print.
ESC 4	1B 34	End pass-through print.
ESC M	1B 4D	Insert line
ESC I	1B 6C	Delete line
ESC 5	1B 35	Lock keyboard
ESC 6	1B 36	Unlock keyboard
ESC I	1B 49	Backtab
ESC g	1B 67	Download function key - IGNORED
ESC ';	1B 60 3B	Go to 132-column mode
ESC ':	1B 60 3A	Go to 80-column mode
ESC e	1B 65	Insert to EOP - IGNORED
ESC L	1B 4C	Duplicate line - IGNORED
ESC G	1B 47	Erase variable - IGNORED
ESC E	1B 45	Delete character
ESC F	1B 46	Insert character on/off
ESC H	1B 48	Next character is graphic

ESC f	1B 66	Insert character to EOS - IGNORED
ESC t	1B 74	Local - IGNORED
ESC T	1B 54	Online - IGNORED
ESC @	1B 40	Unknown - IGNORED

Keystrokes

Key	Hex	Code
<Home>	01	<ctrl-A>
<Up>	1A	<ctrl-Z>
<Left>	15	<ctrl-U>
<Right>	06	<ctrl-F>
<Down>	0A	<ctrl-J>
	7F	
<Backspace>	as determined by BACKSPACE command	

17.3 WYSE 60 FUNCTIONS

The Wyse 60 emulation does not include protected field operations.

Received codes:

Code	Hex	Action
BEL	07	Ring bell (see BEEP commands, section 12)
BS	08	Move cursor left
TAB	09	Tab
LF	0A	Move cursor down
VT	0B	Move cursor up
FF	0C	Move cursor right
CR	0D	Return
SO	0E	Unlock keyboard
SI	0F	Lock keyboard
DC1	11	XON or received command
DC2	12	Printer on - following characters up to DC4 go to output file
DC3	13	XOFF or end of received command
DC4	14	Printer off
CAN	18	Transparent print ON
SUB	1A	Clear screen
FS	1C	Initiate ANZIO command
GS	1D	Terminate ANZIO command
RS	1E	Home cursor
US	1F	Return plus linefeed
ESC =	1B 3D	Cursor positioning
ESC a	1B 61	Cursor positioning
ESC T	1B 54	Clear to EOL
ESC Y	1B 59	Clear to EOS
ESC y	1B 79	Clear to EOS
ESC #	1B 23	Lock

ESC "	1B 22	Unlock
ESC {	1B 7B	Home
ESC E	1B 45	Insert line
ESC R	1B 52	Delete line
ESC G	1B 47	Set field attribute
ESC H STX	1B 48 02	Graphics on
ESC H ETX	1B 48 03	Graphics off
ESC H	1B 48	Display next character as graphic
ESC (1B 28	Write protect off
ESC)	1B 29	Write protect on
ESC !	1B 21	Set attribute for tag
ESC 0	1B 30	Clear all tabs
ESC 1	1B 31	Set tab
ESC 2	1B 32	Clear tab
ESC i	1B 69	Tab
ESC j	1B 6A	Reverse index
ESC I	1B 4A	Backtab
ESC +	1B 2B	Clear, write protect off
ESC *	1B 2A	Clear, write protect off
ESC ,	1B 2C	Clear, write protect off
ESC &	1B 26	Write protect on, scroll off
ESC '	1B 27	Write protect off, scroll on
ESC D	1B 44	Set duplex
ESC N	1B 4E	Scroll off
ESC O	1B 4F	Scroll on
ESC t	1B 74	Clear to EOL
ESC 3	1B 33	Transparent print on
ESC 4	1B 34	Transparent print off
ESC ';	1B 60 3B	132-column mode
ESC ':	1B 60 3A	80-column mode
ESC 'A	1B 60 41	Set tag attribute as normal
ESC '6	1B 60 36	Set tag attribute as reverse
ESC '7	1b 60 37	Set tag attribute as dim
ESC '	1B 60	Other screen feature - ignored
ESC q	1B 71	Insert mode on
ESC r	1B 72	Insert mode off
ESC Q	1B 51	Insert a space
ESC W	1B 57	Delete character
ESC e 0	1B 65 30	Propagate on
ESC e 1	1B 65 31	Propagate off
ESC c	1B 63	Set bank
ESC d .	1B 64 2E	Wrap mode off
ESC d /	1B 64 2F	Wrap mode on
ESC d #	1B 64 23	Transparent print on
ESC d '	1B 64 27	ignored
ESC w	1B 77	Go to page
ESC Z	1B 5A	Define key - ignored
ESC z	1B 7A	Define function key - ignored
ESC c G	1B 63 47	Draw box

ESC c N	1B 63 4E	Draw box relative
ESC c F	1B 63 46	Erase rectangle
ESC c H	1B 63 48	Erase rectangle
ESC ;	1B 3B	Erase screen
ESC :	1B 3A	Erase screen
ESC ?	1B 3F	Report cursor position
ESC b	1B 62	Report cursor position

Keystrokes:

Key	Hex	Code
<Home>	1E	<ctrl-N>
<End>	1B 54	ESC T
<Up>	0B	<ctrl-K>
<Left>	08	<ctrl-H>
<Right>	0C	<ctrl-L>
<Down>	0A	<ctrl-J>
	7F	
<PgDn>	1B 4B	ESC K
<PgUp>	1B 4A	ESC J
<Insert>	1B 71	ESC q
<Backtab>	1B 49	ESC I
<ctrl-Home>	1B 7B	ESC {
<Backspace>	as determined by BACKSPACE command	

17.4 VT220 FUNCTIONS

Received codes:

Code	Hex	Action
BEL	07	Ring bell (see BEEP commands, section 12)
BS	08	Cursor left
TAB	09	Tab
LF	0A	Cursor down
CR	0D	Return
SO	0E	Graphics on
SI	0F	Graphics off
DLE	10	Print control character
DC1	11	XON or received command
DC2	12	Printer on - following characters up to DC4 go to output file
DC3	13	XOFF or end of received command
DC4	14	Printer off
SUB	1A	Cursor up
FS	1C	Initiate ANZIO command
GS	1D	Terminate ANZIO command
RS	1E	Initiate ANZIO command
US	1F	Terminate ANZIO command
ESC [..H	1B 5B .. 48	Cursor movement
ESC [..F	1B 5B .. 46	Cursor movement

ESC [1 K	1B 5B 31 4B	Clear to beginning of line
ESC [2 K	1B 5B 32 4B	Clear to EOL
ESC [2J	1B 5B 32 4A	Clear screen
ESC [J	1B 5B 4A	Clear to EOS
ESC [..D	1B 5B .. 44	Cursor left
ESC [..C	1B 5B .. 43	Cursor right
ESC [..A	1B 5B .. 41	Cursor up
ESC [..B	1B 5B .. 42	Cursor down
ESC [..m	1B 5B .. 6D	Set attribute/color
ESC [..r	1B 5B .. 72	Set scroll region
ESC 7	1B 37	Save cursor
ESC 8	1B 38	Restore cursor
ESC [?7h	1B 5B 3F 37 68	Wrap next character
ESC [?7l	1B 5B 3F 37 6C	Wrap off
ESC [?1h	1B 5B 3F 31 68	Set alt cursor mode (smkx)
ESC [?1l	1B 5B 3F 31 6C	Set normal cursor mode (rmkx)
ESC [?3h	1B 5B 3F 33 68	Set to 132-column mode
ESC [?3l	1B 5B 3F 33 6C	Set to 80-column mode
ESC [?.h	1B 5B 3F .. 68	Misc. settings - IGNORED
ESC [?.l	1B 5B 3F .. 6C	Misc. settings - IGNORED
ESC =	1B 3D	Set keypad mode - IGNORED
ESC >	1B 3E	Set keypad mode - IGNORED
ESC (0	1B 28 30	Set g0 graphics
ESC (?	1B 28 3F	Set g0 upper PC graphics
ESC (B	1B 28 42	Set g0 ASCII
ESC)0	1B 29 30	Set g1 graphics
ESC)?	1B 29 3F	Set g1 upper PC graphics
ESC)B	1B 29 42	Set g1 ASCII
ESC D	1B 44	Cursor down
ESC E	1B 45	Return + linefeed
ESC [E	1B 5B 45	Return + linefeed
ESC [F	1B 5B 46	Return and up
ESC H	1B 48	Set tab
ESC M	1B 4D	Reverse index
ESC [3g	1B 5B 33 67	Clear all tabs
ESC [0g	1B 5B 30 67	Clear tab
ESC 3	1B 33	Transparent print on
ESC 4	1B 34	Transparent print off
ESC [5i	1B 5B 35 69	Transparent print on
ESC [4i	1B 5B 34 69	Transparent print off
ESC [4h	1B 5B 34 68	Insert on

ESC [4l	1B 5B 34 6C	Insert off
ESC [L	1B 5B 4C	Insert line
ESC [Z	1B 5B 5A	Backtab
ESC [I	1B 5B 49	Tab
ESC [P	1B 5B 50	Delete character
ESC [M	1B 5B 4D	Delete line
ESC [@	1B 5B 40	Insert space
ESC [S	1B 5B 53	Scroll forward
ESC [T	1B 5B 54	Scroll reverse
ESC [d	1B 5B 64	Go to line
ESC [G	1B 5B 47	Go to position
ESC [X	1B 5B 58	Erase characters
ESC [b	1B 5B 62	Repeat character
ESC [6n	1B 5B 36 6E	Report cursor position
ESC n	1B 6E	Latin characters
ESC o	1B 6F	Latin characters
ESC N	1B 4E	Latin next character
ESC O	1B 4F	Latin next character

Keystrokes: ANZIO supports the arrow keys in "cursor" mode and in "application" mode. ANZIO does NOT support the numeric pad in "alternate keypad: mode, except by use of "defined keys" (see section 5).

Keystrokes in cursor mode (after ESC [?1l):

Key	Hex	Code
<Home>	1B 5B 48	ESC [H
<Up>	1B 5B 41	ESC [A
<Left>	1B 5B 44	ESC [D
<Right>	1B 5B 43	ESC [C
<Down>	1B 5B 42	ESC [B
	7F	
<END>	1B 5B 4B	ESC [K

Keystrokes application mode (after ESC [?1h):

Key	Hex	Code
<Home>	1B 4F 48	ESC O H
<Up>	1B 4F 41	ESC O A
<Left>	1B 4F 44	ESC O D
<Right>	1B 4F 43	ESC O C
<Down>	1B 4F 42	ESC O B
	7F	
<END>	1B 4F 4B	ESC O K

17.5 ADDITIONAL FUNCTIONS

These functions are specific to ANZIO.

<u>Code</u>	<u>Hex</u>	<u>Action</u>
DC1	11	Received function. The characters following, up to a DC3, can contain virtually any function. See section 16.
DC3	13	Terminates a received function. Otherwise ignored.
ESC P	1B 50	Multi-screen memory. ANZIO stores 8 screens full of information. The character following the 'P' is stripped to its low 3 bits, and this determines a page number (0 to 7). The program then places that page on the screen (very quickly). This feature can be used with special host software, to reduce the I-O time. Note that this screen memory can also be viewed by the REVIEW command.
ETB	17	Indicates that the two characters following contain an error-checking code (CRC). Ignored under RMCOS or UNIX, where it may mean "cursor off".

18. ERROR MESSAGES

Error indicators in ANZIO fall into two categories. The first consists of temporary indicators on the status line, showing communication errors. The second consists of textual error messages which are indicated usually on line 25.

18.1 COMMUNICATION CHIP ERRORS

If any of the words PARITY, FRAMING, or OVERRUN appear on your status line, these indicate that the communication chip in your PC has reported an error. (Note that other communication software may LOOK like it's not getting communication errors, but that is because it doesn't report them as ANZIO does.) Let's look at each of these:

PARITY

This means that the chip has detected a parity error. It may mean that your parity is not set correctly, and so your host system is sending its characters with a different parity than you are expecting. Correct your parity setting.

This message can also appear if your BAUD rate is wrong.

This may also be the result of telephone line noise, if you are working over a modem. You may be able just to ignore it, but if you're doing file transfer over a noisy line, data loss is a distinct possibility. Error-correcting modems or error-proof file transfer are in order.

FRAMING

This means that the chip is not getting the correct number of bits in its characters. Again, if this is intermittent, it may be the result of phone noise, as described above. It is also possible that your BAUD, PARITY, DATA BITS, and/or STOP BITS are set wrong.

Note that in ANZIO's terminology, DATA BITS refers to the number of bits for data only. Thus if you have DATA BITS 7 and PARITY OFF, you are dealing with a seven-bit character; turning PARITY ON will then yield an 8-bit character.

OVERRUN

This one is a little tougher. When a character comes in to your PC's serial port, it generates an interrupt. This causes the PC to interrupt whatever it is doing, go fetch the character, and put it into ANZIO's buffer. If another character comes in before the first is processed, an overrun has occurred.

Thus when you see OVERRUN, it indicates that something in your PC is preventing interrupts from being processed properly. Under DOS 5 and 6, this can usually be eliminated by 1) removing EMM386; 2) setting SMARTDRV to NOT do "write-back caching"; 3) removing SMARTDRV; or 4) Upgrading your serial port to a 16550. Please contact us for more information, or see ANZIO's READ.ME file.

Another possibility is poorly-written memory-resident software, such as print-spoolers, on-screen clocks, and so forth. To analyze for this problem, try running ANZIO with NO special software preloaded.

18.2 TEXT MESSAGES

AN INPUT FILE IS OPEN

Only one input file may be open at once. You will need to close your input file (CLOSEI) before you can open another.

AN OUTPUT FILE IS OPEN

Only one output file may be open at once. You will need to close your output file (CLOSEO) before you can open another.

BAD I/O STATUS: <nn> ON <filename>

An unexpected file error was encountered.

BAD MODEM STATUS

The DIAL command received a code from the modem that it couldn't understand.

BAD TAB FORMAT

The program could not understand the format of your TAB command.

DEMO VERSION TERMINATED

You are running a demo version of ANZIO, and your time is up.

DISK ERROR ON <command>

A disk error occurred while doing the command specified.

DISK OR DIRECTORY FULL

During some operation that writes to disk, you filled up either the disk or the disk's directory.

FILE EXISTS

The file name you are trying to OPENO already exists on the disk. Either delete it or use a different name.

FILE OR SUBDIRECTORY ERROR

Usually encountered when you are making incorrect use of a subdirectory name.

FILE NOT FOUND: <filename>

The specified file was not found on the disk.

INSUFFICIENT MEMORY

The dynamic memory space available to the program has been filled with a) defined keys and/or b) received data.

INVALID FUNCTION

Your function specification does not fit one of the functions available.

NO OUTPUT FILE OPEN

The operation you are attempting requires an output file to be open, and there is none.

OVERFLOW ON RECEIVE

A communication chip overflowed during a file reception operation. Your receive file is corrupted.

RECEIVE BUFFER OVERFLOW

File reception was unable to grab more dynamic memory. Your file is corrupt.

RESTRICTED COMMAND

You are using a restricted version of ANZIO, and the command you have selected is not available to you.

UNABLE TO DELETE: <filename>

The file indicated is not on the disk, so ANZIO can not delete it.

UNABLE TO INITIALIZE COMMUNICATION

You have selected an illegal communication parameter, and the program can not set up its communications. Examples include illegal baud rates, etc. Or, you may have told ANZIO to use a PORT that doesn't exist on your machine.

UNABLE TO READ KEY FILE: <filename>

The defined key file specified was not found.

UNABLE TO TRANSMIT

The program is unable to transmit, because of wiring or other communication problems.

UNABLE TO WRITE TO OUTPUT FILE

The output device is not ready (such as a printer), or the output file has filled the disk.

19. WORKING WITH OTHER PC SOFTWARE

19.1 FEEDING HOST DATA TO PC PROGRAMS

Warning: information in this section is somewhat dated, but still generally accurate.

One of the most popular uses of ANZIO involves transferring data from the host system to the PC in such a way that it can be used by any of the various tools available on the PCs, particularly spreadsheet and word processing systems. For instance, you might wish to capture your general ledger balances from the host and bring them into LOTUS 123 to create a financial statement. Or you might want to extract from your accounts receivable system the names and addresses of any customers who are past due, pass that information to the PC, and merge it into a standard dunning letter, using a word processing package.

The first point to understand is that "data" on one system is not necessarily "data" on the other. That is, the various means of storing information on the host system and on the PC may not be compatible. The second point is that more is required than simply doing file transfer. In the second case above, you would not want to transfer the entire A/R file - it could take hours, might not fit, and you'd still have to do the extraction.

The particular format that is used by the PC software varies quite a bit. There has been some effort made at establishing standards, but the standards are a long way from standard. In general, these programs deal with "fields" of data. A field can be either numeric (a number), or alphanumeric (any combination of characters, such as a line of an address). There may be a requirement for some kind of indication of whether a field is numeric or alphanumeric, such as quotes around alphanumeric fields. And there must be agreement on "delimiters", which are special characters and/or carriage-returns (CRs) used to indicate where one field ends and another begins.

It is up to you to know the particular requirements of your system. The following discussion will focus on LOTUS, which uses a format called CSV (comma separated values). Quite simply, this means that alphanumerics are in quotes, numbers are not, there are commas between fields, and a carriage-return linefeed indicates the end of a line. A file constructed

like this can be read into a rectangular area of LOTUS with the File Import Number command.

In summary, there are three things that must go on: 1) extraction, 2) reformatting, and 3) data transfer. There are three general approaches to this problem. The first involves your programmer writing the code to generate the file to be transferred. The second involves using special software on the host end to do the extraction and reformatting. The third uses your existing programs on the host and ANZIO's ability to capture and reformat columns of data. We will look at each one individually.

19.1.1 QUERY PROGRAMS


In this approach, a "query program" or "report generator" is used on the host. This is utility software that, once installed for a particular file or set of files, allows the end user to specify what information is to be printed, displayed, or extracted. The extract option produces a file which is of a format that can be transferred via ANZIO, and read into your PC software. That is, the query program must be coordinated to some degree with both ANZIO and the PC software.

19.1.2 PICKING FOR LOTUS AND OTHERS

Many of our customers approached us with a simple question: "The information I want is right there on the screen, why can't I just grab it and stick it into LOTUS?" On reflection, it seemed a reasonable request. So we provided it, in the form of the PICK command.


To start with, you must have on your host system software that produces a screen display of information, preferably in columnar form. Then using ANZIO just as a terminal, run that program to display the information. Your task now is to tell ANZIO which columns of information you want to pick, what their formats are, and where to put the data.




First, you must have an output file open. This is done with the old familiar:


```
OPENO <filename> 
```

Now use the PICK command. Enter²

² ANZIO commands must be entered at the "Func:" prompt; see section 4.



PICK 


The  key initiates a "highlight" function that allows you to indicate what column you want to deal with (we'll do this one column at a time). Your highlight appears in the center of the screen. Use the cursor arrows to move one corner. Each time you hit , you take control of the opposite corner, and can move that one. So, between the arrow keys and the  key, you can place the highlight so that it covers the column of data you want to pick. This includes working with a screen width of 132 columns, such as picking from a spool file display (see Appendix B on SPOOLCRT).

When you hit , your command line will suddenly show four numbers, representing the row and column of each of two corners. Thus you have defined your column. Now you must give a "type" indicator. These are:

A = Alphanumeric
N = Number
- = Number negated

The minus indicator works as a number, but ANZIO will reverse the sign before writing it out, so for instance liability accounts which ordinarily carry a negative balance can be inverted.

If you wish to define more than one column, enter a  and hit  again. You can continue this way until you run out of room in the command string (total of 255 characters). Note that "PICK" is entered only once, and that each column must have its four coordinates and its type indicator.

When you hit , the action begins. ANZIO reads the screen, and for each screen line, determines which columns intersect that line, in the order in which they were defined (not the order they appear on the screen). ANZIO then builds and writes out a record (a line), in CSV format, for any screen line with defined columns. Numbers are rearranged into a standard floating-point format, with leading minus signs as needed. Any number which on the screen contains a minus sign, a parenthesis, a "D", "B", "C", or "R", or any combination, will be considered negative by ANZIO. Dollar signs and commas will be removed.

Once the PICK is completed, you can continue to generate other screens and do other PICKs as needed. When you are finished, be sure to do a²

CLOSEO 

to close your output file.

Like almost anything else in ANZIO, this operation can be put into a defined key, if it is something you'll do more than once.

19.1.3 CUSTOM-PROGRAMMED EXTRACTION

This approach requires a programmer who is familiar with your particular installation. That person can write a program to extract specific data from a file or set of files, format it properly, and write it into a sequential file. Using ANZIO and the included file transfer programs, then, this file can be passed down to the PC.

The primary problem with this approach is that it lacks flexibility. End users who are accustomed to working with the various PC tools have come to expect flexibility, but this approach requires the programmer to "hard-code" every particular extraction needed.

19.2 NOTES ON OTHER PC UTILITIES

Warning: this information is dated, and applies only to DOS.

Whenever ANZIO shares control of the PC with another program, there is the possibility for conflicts. This sharing can occur in three contexts. First, a memory-resident program, such as SIDEKICK, might be loaded before ANZIO. Second, after you have done a RUN or RUN/N command in ANZIO, you can invoke another program. Third, when ANZIO is resident as a result of a STAY or STAY/G command, you can invoke another program.

The most common cause for conflict is use of the serial port. If another program sharing control with ANZIO manipulates the same serial port, such as it might if it had an auto-dial capability, ANZIO may lose control


² ANZIO commands must be entered at the "Func:" prompt; see section 4.

of that port and be unable to communicate. It is best to disable any auto-dial feature in this situation, or configure it to use another port.

APPENDIX A. NOTES ON PARTICULAR HOST SYSTEMS

Most of the instructions in this manual apply to all host operating systems. This section details some variations necessary for particular host systems.


When working with UNIX, be sure its TERM setting agrees with ANZIO's TERM setting. You can verify UNIX's setting by using the "set" command. To change UNIX's setting, tell UNIX:

```
TERM=VT220; export TERM 
```

for instance. Note that certain application software on UNIX may assume you are using a certain TERM type, regardless of the current setting.


If you use RM/COBOL under UNIX, and configure ANZIO (and UNIX) for the NCR 7900, you will find that they don't get along too well, because of the 7900's use of propagating attributes (attributes take a space on the screen). You can either modify the terminfo to disable attributes, or use a different terminal type.

It is also important that UNIX be configured properly for backspace and "kill". If these are not set correctly, the characters '@' and/or '#' may not be processed correctly (especially during PC-to-UNIX file transfer). To make sure, tell UNIX that "erase" is <backspace> and "kill" is <ctrl-U>:

```
stty erase '^h',kill '^u' 
```

This command can be made "permanent" by placing it where it will always be executed on startup. For the command to apply to an individual user account, place the above line in the file ".profile" in the user's home directory. For a command that will apply to every user, place the above line in the file "/etc/profile".

ANZIO will issue XON/XOFF handshaking when necessary, and will also respond to it coming from UNIX. To make UNIX work correctly with XON/XOFF, do:

```
stty -ixany 
```

Finally, set ANZIO for UNIX, LOCK OFF, FULL DUP, BACKSPACE 8, and probably TAB OFF.

When working with one of these I-systems, be sure your parameters are set as follows:

LOCK

LOCK should be ON at most times with these systems, in order for keyboard buffering to work. Exceptions are 1) when talking to a local modem, 2) during bootup of some systems, and 3) newer releases of SCLEDIT and VIEW under ITX, which do not beep.

TERM

The I-systems were originally designed for the 7900 terminal, so you'll probably want to set TERM to N7900. This is changing even as we write this, however, with later releases of ITX including support for other terminal types. Just be sure that TERM in ANZIO agrees with your SYSGEN setting.

Following are some notes on running under RM/COS:

CABLING

We have received reports of problems with RMCOS locking up the terminal line, effective RMCOS release 2.7. We have also heard that an alternative wiring scheme for "intelligent terminals", suggested by NCR support, eliminates the problem. This new cabling scheme is embodied in NCR cables with part numbers 1308-C045 (for 9-pin) and 1308-C046 (for 15-pin). Contact us or your NCR FE for more information.

TERMINAL TYPE AND PASS-THROUGH PRINTING

ANZIO will work if COS thinks it is an N7900, N7901, or ADDSVPT. Only the ADDS Viewpoint supports pass-through printing, however, so we suggest using that setting. Pass-through printing also requires that you define (in the SYSDEFIL) a printer slaved to your CRT.

DUPLEX

Always use FULL DUP.

LOCK MODE

You should generally work in LOCK OFF mode, except when doing file transfer. This is because the operating system does not ordinarily send a BELL code with each prompt.

TAB

Some programs under RM/COS are designed to make use of the TAB key. In order to send the tab code (hex 09) to the host, instead of translating it into spaces, set TAB OFF.

DEFINED KEYS

We suggest you set up a function key for "acknowledge" (control-G A), and one for "interrupt" (control-X). See the sample key file (RMCOS.KYS), as described in Appendix F.

FILE TRANSFER

As mentioned, file transfer works differently on RM/COS. Be sure to check section 11.6.

ANZIO has a "dumb" mode, called TTY. Use this mode when you are using ANZIO to talk to a bulletin board system or some other foreign system (or, you may want to use VT220 mode). In TTY mode, ANZIO does not respond to most of the control codes listed in this manual.

APPENDIX B. ADDITIONAL PROGRAMS

This section explains additional programs included on the ANZIO diskette.

Q-GRAPH is a PC program that takes a small Comma Separated Values (CSV) file, such as might be produced with the PICK command, and creates a graph from it. It currently supports LINE graphs of one or more variables, on the following types of video adapters: Hercules, EGA, CGA, VGA, MCGA, AT&T, and 3270. It does NOT support NCR's so-called NGA adapter (on the PC4i and others) in high resolution, but will drive it at CGA resolution.

When you PICK data from the screen, you create a file in CSV format. That means each line in the file represents a line in a spreadsheet, fields are separated by commas, and there are quotes around alpha fields. Q-GRAPH can read this file in to a mini-spreadsheet, make some decisions about it, and quickly place a graph on the screen.

The format that Q-GRAPH expects is this:

```
GRAPHTITLE
      TITLE1  TITLE2...
LABEL1      y1      z1...
LABEL2      y2      z2...
LABEL3      y3      z3...
(etc.)
```

GRAPHTITLE is an optional line. If it exists, it will show as a title at the top of the screen.

TITLE1, TITLE2, etc. are optional titles for each variable. They are presently ignored.

LABEL1, LABEL2, etc. are optional X-axis labels, and must be alpha.






The numeric values y1, y2, etc. are plotted as the first line. If additional columns exist (z1, z2, etc.), they will be plotted as additional lines.

Consider the following example, taken from the MENUDEMO program. The screen shows:

```
Software sales by month
```

	1987	1988
Jan	\$5211.25	\$8764.22
Feb	115.11-	9222.55
Mar	1015.22	10500.00
Apr	9015.87	6123.26
May	7555.23	12532.99
Jun	6123.88	14663.55
Jul	8523.41	13585.96
Aug	4123.88	10589.36
Sep	9800.52	19324.38
Oct	3549.54	8562.55
Nov	8432.93	6843.58
Dec	7315.82	9132.85

We would create our file as follows:²


```
OPENO PICKFILE.TST   
PICK 1 40 1 1A   
PICK 12 15 2 2A 25 28 2 2A  
  
PICK 1 3 3 14A 6 17 3 14N  
20 31 3 14N   
CLOSEO 
```

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

This would produce a file (PICKFILE.TST) that looks like this:

```
"Software sales by month"  
"1987" , "1988"  
"Jan" , 5211.25 , 8764.22  
"Feb" , -115.11 , 9222.55  
"Mar" , 1015.22 , 10500.00  
"Apr" , 9015.87 , 6123.26  
"May" , 7555.23 , 12532.99  
"Jun" , 6123.88 , 14663.55  
"Jul" , 8523.41 , 13585.96  
"Aug" , 4123.88 , 10589.36  
"Sep" , 9800.52 , 19324.38  
"Oct" , 3549.54 , 8562.55  
"Nov" , 8432.93 , 6843.58  
"Dec" , 7315.82 , 9132.85
```

We would then tell ANZIO to run the Q-GRAPH program:²

```
RUN/N Q-GRAPH.EXE STD LINE  
PICKFILE.TST 
```

and the graph would appear on the screen. Where you see STD in this example, we are telling Q-GRAPH to use its "standard" graph mode, which means it will attempt to determine what video hardware it is running on and use the appropriate mode. You can also override this, by substituting one of the following:

```
CGA  
EGA  
VGA  
HERC (Hercules)  
6 (monochrome on a CGA)
```

Hit any key to return to ANZIO.

The MENUDEMO program is a COBOL source program for use with I-systems (IMOS III, IMOS V, IRX, and ITX). It demonstrates some of the newer features of ANZIO, under control of the host program.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

Transfer MENUDEMO.CBL up to the host, using the "simple upload" instructions for your particular system. Then simply compile it and run it.

The MENUDEMO program demonstrates the following features:

- Commands sent from the host computer
- Control of color
- MENUBAR
- WINDOW
- FILL
- RUN - dropping out to the DOS level
- RUN program - host initiates a PC program
- PICK and Q-GRAPH.EXE




SPOOLCRT is a program for IMOS III, IMOS V, IRX, and ITX to read and display a manual spool file on the screen. It allows you to move forward and backward in the file by lines, screens, or pages.

SPOOLCRT will work with either an 80-column or a 132-column display, and as such is a good way of demonstrating ANZIO's 132-column virtual screen capability. Of course SPOOLCRT has to know which type of screen it has. You can tell it you have a 132-column screen either by setting switch 1 on before executing it, or by giving it the command "132W" once it is running.

The program does not restrict itself to ANZIO as a CRT. In fact, it contains the necessary code to switch a WYSE terminal into and out of 132-column mode.

SPOOLCRT is distributed on floppy, and must be uploaded and compiled before execution.

As a sample session, enter the following commands to the host:

```
SET SWITCH 1 ON   
AS A MYSPOOL(3)   
EX SPOOLCRT(5) 
```

Once loaded, SPOOLCRT will give you a list of its command options.


APPENDIX C. MIGRATION FROM EARLIER RELEASES

ANZIO is constantly evolving to meet the needs of its users. If you are installing a new release, you will need to check on a few things.

Information on what has been changed from release to release, as well as specific instructions for migrating, are included in the READ.ME file on the distribution diskette. Check it over carefully.

We try to maintain as much upward compatibility as possible. Old command forms are usually still operable, even when they have been superseded.




If you have problems with a new release, start ANZIO with a command line parameter of "NONE". That is, for Anzio for DOS, issue the following command to start ANZIO:

```
ANZIO NONE 
```

This will give ANZIO a clean slate for startup.

APPENDIX D. SERIAL COMMUNICATION PROBLEMS

When ANZIO completely fails to communicate, it can be difficult to determine the cause. Following are some things to try:

1. First, make sure ANZIO has its LOCK mode turned off by doing:²
LOCK OFF 
2. If a LOCK indicator continues to show on the bottom line, hit  .
3. Try setting ANZIO for different BAUD rates.
4. Try setting ANZIO for different PORTs.
5. Make sure your cable from the host system is plugged into the serial port (I know it's obvious, but . . .).
6. Try a CRT on the cable, to try to isolate the problem.
7. Open the PC, and check the jumpers or switches on the serial port. If at all possible, make the port and IRQ match one of the standards, shown in section 1.2.1. Otherwise, you may need to use ANZIO's IRQ command to set it for a non-standard situation.
8. Do a "loopback test", as follows. Disconnect the cable from the host. Using a 9-pin to 25-pin adapter (if necessary), and a female-to-female adapter (if necessary), configure the PC connection so that it ends in a 25-pin female plug. Then bend a small paper clip, and insert one end into hole 2 and one end into hole 3 on the exposed plug. Now, if you type on ANZIO's keyboard, the keystrokes should be echoed back to you (if you are in FULL DUP), or appear double (if you are in HALF DUP). If they do not, your serial port is not working properly.

² ANZIO commands must be entered at the "Func:" prompt; see section 4.

If the loopback test fails, you probably have a) two serial ports mapped to the same comm port number; b) two devices sharing an interrupt (IRQ); or c) a defective serial port.

APPENDIX E. DISTRIBUTION INFORMATION

Warning: outdated information.

ANZIO is distributed on a floppy diskette. This diskette is your master copy and should be guarded against accidental damage. The distribution diskette contains the following files:

ANZIO.EXE

This is the actual ANZIO program. Might instead be ANZIOD.EXE.

ANZIOS.EXE

This is a special SMALL version of the ANZIO program. It is designed for situations where memory is limited. It does not have many of the advanced features of ANZIO.

ANZIO.HLP

This is the help file used by ANZIO.

SEND-PC.CBL

A COBOL source program for file transfer of various file types from IMOS III, IMOS V, IRX, or ITX, to a PC (Section 11.4).

SEND-PC.C

A "C" source program for file transfer of any UNIX text file to a PC (Section 11.5).

SEND-RM

A source program for file transfer of 80-byte variable files using ANZIO's CAPTURE procedure under the RMCOS operating system (Section 11.6).

SEND-L.RM

A source program for file transfer of 510-byte variable files using ANZIO's CAPTURE LONG procedure under the RMCOS operating system (Section 11.6).

RECV-PC.CBL

A COBOL source program for file transfer of any file larger than 80-byte variable files to an IMOS III, IMOS V, ITX, or IRX machine from a PC (Section 11.4).

RECV-PC.RM

A COBOL source program for file transfer of any file larger than 80-byte variable files to an RMCOS machine from a PC. Currently it is set for 510-byte files (Section 11.6).

RECV-SPL.CBL

A COBOL source program for file transfer of a PC print file to an IMOS III, IMOS V, IRX or ITX spool file or printer (Section 11.4).

SPOOLCRT.CBL

A COBOL source program for the I-SYSTEMS, to display spool files on a CRT or PC (Appendix B).

Q-GRAPH.EXE

A quick graph program to produce line graphs from a simple CSV formatted file on the PC (Appendix B).

MENUDEMO.CBL

A COBOL source program for I-systems, demonstrating several ANZIO features. See Appendix B.

SAMPLE.KYS

A sample key file for all systems except RMCOS. See Appendix F.

RMCOS.KYS

A sample key file for RMCOS. See Appendix F.

VERSION.EXE


A program used in the installation process, to create the correct version (IMOS V, ITX, etc.) of various source programs.

INSTALL.BAT

The batch string used to install ANZIO.

READ.ME

A file of last-minute notes, etc. To see them, enter:

TYPE READ.ME 

SET-EGA.EXE

A PC program that allows you to configure an EGA or VGA video adapter to allow underlining, etc.

SET-EGA.DOC

A file that describes how to use SET-EGA.EXE.

DOWNLOAD

A UNIX shell script for downloading files to ANZIO.

RECV-PC.C

A UNIX source program that allows upload of longer records into UNIX.

VT100.KYS

Sample function keys for VT100 emulation.

VT100.DOC

A file that explains the keys used in VT100.KYS

VT220.KYS

Sample function keys for VT220 emulation.

VT220.DOC

A file that explains the keys used in VT220.KYS

VT220S.KYS

Sample function keys for VT220 emulation. Uses a different approach to translating PC function keys to VT220 function keys than does VT220.KYS.

VT220S.DOC

A file that explains the keys used in VT220S.KYS

WYSE60.KYS

Sample function keys for WYSE60 emulation.

WYSE60.DOC

A file that explains the keys used in WYSE60.KYS

KERMIT.KYS

A file of function key definitions to work with Word Perfect 5.0 on a UNIX system, when WPTERM is set for KERMIT.

KERMIT.DOC

A file that explains the keys used in KERMIT.KYS

ANZIO.TIC

A terminfo file for UNIX, that will enable you to set your TERM variable to ANZIO. See section 1.11.2.

ANZIO-M.TIC

A terminfo file for UNIX, that will enable you to set your TERM variable to ANZIO-M, for monochrome PCs. See section 1.11.2.

ANZIOTIC.KYS

A file of function key definitions that is synchronized with ANZIO.TIC (and ANZIO-M.TIC).

ANZIOTIC.DOC

A file that explains the keys used in ANZIOTIC.KYS.

NCR3000/SEND-PC.ECH

A file for installing "send-pc" on a UNIX V.4 system that doesn't have a C compiler. See READ.ME for more information.

NCR3000/SEND-PC

A file for installing "send-pc" on a UNIX V.4 system that doesn't have a C compiler. See READ.ME for more information.

NCR3000/RECV-PC.ECH

A file for installing "recv-pc" on a UNIX V.4 system that doesn't have a C compiler. See READ.ME for more information.

NCR3000/RECV-PC

A file for installing "recv-pc" on a UNIX V.4 system that doesn't have a C compiler. See READ.ME for more information.

NCRTOWER/SEND-PC.ECH

A file for installing "send-pc" on an NCR Tower system that doesn't have a C compiler. See READ.ME for more information.

NCRTOWER/SEND-PC

A file for installing "send-pc" on an NCR Tower system that doesn't have a C compiler. See READ.ME for more information.

NCRTOWER/RECV-PC.ECH

A file for installing "recv-pc" on an NCR Tower system that doesn't have a C compiler. See READ.ME for more information.

NCRTOWER/RECV-PC

A file for installing "recv-pc" on an NCR Tower system that doesn't have a C compiler. See READ.ME for more information.

APPENDIX F. SAMPLE DEFINED KEYS

As an aid in getting started in both using ANZIO and setting up and using defined keys, we have included two sample key files, one for RMCOS and one for everything else.

Note: Applies generally only to users of the ITX operating system.

The sample keys defined in this file are:

- 1
"IN*D" to begin INSERT in \$EDIT (on I-systems)

- B
For ITX, this is a special BREAK/SUSPEND key. It captures the entire screen as a WINDOW, does a BREAK and S for SUSPEND, thereby saving the old screen. See "b".

- b
Undoes the "B" key, above. Does a RET to exit the current process, then restores the screen saved above.

- C
Takes you to the COLOR screen.

- D
Sends today's DATE.

- T
Sends the current TIME.

- H
Hangs up a modem.


- L
Logs on to an ITX system, with a series of BREAKs and R, so that you should end up at the command level regardless of what state ITX was in when you started.



c
Logs on as "L" above, but using <ctrl-C>. For IRX and some ITX.





R
RUN/N - takes you to the DOS level.



t
Prompts for a file name, opens it, and transmits it with trailer END\$.



u
Does a UNIX simple file upload, with prompts.



 Takes you to CALCulator.

 or 
Takes you to REVIEW.


Causes line editing under \$EDIT (on I-systems), with
 CH | {  } 

 
Does a PAN LEFT.

 
Does a PAN RIGHT.

 
Does a STAY/G. Causes ANZIO to go memory-resident.

Note: applies only to users of the RMCOS operating system.

The sample keys defined in this file are:

C
Takes you to COLOR screen.

H

Hangs up a modem.

R

RUN/N - takes you to the DOS level.

F7

Takes you to CALCulator.

F8 or **PG UP**

Takes you to REVIEW.

D

<ctrl-G>Z for DELETE LINE

I

<ctrl-G>N for INSERT LINE

SHIFT <TAB>

<ctrl-G>B for BACKTAB.

HOME

<ctrl-G>C for ERASE FIELD

PG UP

<ctrl-G>A for ACKNOWLEDGE

END

<ctrl-G>R for ERASE RIGHT

PG DN

<ctrl-G>Q for COMMAND

INS

<ctrl-G>I for INSERT CHARACTER

SHIFT **F1** through **SHIFT** **F10**

<ctrl-G>1 through <ctrl-G>0 for FUNCTION 1 through
FUNCTION 10.

ALT F1

Does a STAY/G. Causes ANZIO to go memory-resident.

Note also that the following keys work properly with RMCOS: <TAB> (if TAB OFF), **DEL** for DELETE CHAR, and the arrow keys.

APPENDIX G. ANZIO ON A NETWORK

The question often comes to us, "Can ANZIO work on a network?" But this question means several things. Here is what you can and can't do with ANZIO.

You CAN load ANZIO on a PC that is connected to a network, but communicate with the host via a serial connection.

You CAN access files on a network server with ANZIO.

You CAN put the ANZIO program itself on a network (but please don't violate licensing restrictions).

You CAN talk to the host computer via the network, with AnzioWin, Anzio Lite, AnzioNet, or Anzio14.

ANZIO DOES NOT provide the underlying network software. That must come from other sources. For AnzioWin or Anzio Lite to communicate via the network, you need a WINSOCK.DLL (for TCP/IP) or support software for PicLan or WLIBSOCK. For AnzioNet or Anzio14, you need DOS-level network software.

INDEX

- #
 - as special key in UNIX 153
 - pause in macro key 34, 35, 36, 37
- \$EDIT 33, 40, 63, 66, 168, 169
- @ as special key in UNIX 153
- ^ (caret indicating CTRL) 28
- <alt-A> 25, 31, 119
- <alt-F> 25, 31
- <alt-F1> 169, 171
- <alt-H> 23, 25
- <alt-M> 14, 21, 23, 24, 26, 31
- <alt-U> 31, 161
- <alt-X> 2, 15, 17, 25, 31, 33
- <BACKSPACE> 27, 42
- <BACKTAB> 42
- <Control-C> 14
- <Ctrl-Break> 25
 - see also BREAK: 7
- <ctrl-LEFT> 169
- <ctrl-P> 27
- <ctrl-RIGHT> 169
- <DC2> 130
- <DC4> 130
- <DELETE> 27, 42
- <DOWN-ARROW> 42
- <END> 27, 42, 170
- <ESC> 28, 42
- <F1> 24, 27, 31, 42
- <F10> 13, 24, 128
- <F11> 29
- <F12> 29
- <F2> 24, 43
 - In a defined key 40
 - Invoked from the host 41
- <F2> 36, 39
- <F3> 24
- <F4> 24, 128
- <F5> 23, 24, 26, 96, 127
- <F6> 24, 128
 - in a defined key 36
- <F6> 26
- <F7> 27, 99, 169, 170
- <F7> 61, 149
- <F8> 169, 170
- <F9> 24, 45, 46, 128, 131
- <HOME> 27, 42, 169, 170
- <INS> 170
- <INSERT> 27, 42
- <LEFT-ARROW> 27, 42
- <PgDn> 170
- <PgUp> 169, 170
- <RETURN> 43
- <RIGHT-ARROW> 27, 42
- <shift-TAB> 170
- <TAB> 42
- <UP-ARROW> 42
- 132-Column display 15, 49, 50
 - see also MODE-132: 7
- 132-Column Virtual screen 50
 - PAN LEFT 51
 - PAN RIGHT 51
 - SPOOLCRT.CBL 51
- 7900 1, 20, 24, 115, 134, 153, 154, 155
- 7901 1, 20, 115, 135, 155
- 7E1 86
- 7E2 86
- 7N1 86
- 7N2 86
- 7O1 86
- 7O2 87
- 8E1 87
- 8E2 87
- 8N1 87
- 8N2 87
- 8O1 87
- 8O2 87

ADDS Viewpoint 1, 153
 see also 7901: 7
 Alt 28, 31
 Alternate codes 66
 ANSWERBACK 87
 ANZIO.DEF 15
 ANZIO.EXE 163
 ANZIO.HLP 23, 163
 ANZIO.TIC 166
 ANZIOD.EXE 1, 163
 ANZIO-M.TIC 166
 ANZIONCR.EXE 1
 ANZIOS.EXE 18, 125, 163
 ANZIOTIC.DOC 166
 ANZIOTIC.KYS 166
 ASCII chart 23
 AT386 116
 Attributes
 see Video Attributes: 7
 Auto-dial programs 152
 AUTO-LF 87
 BACKSPACE 153
 BACKSPACE 88
 BAUD 161
 BAUD 88
 BBS (bulletin board system)
 118
 BEEP 88
 Bell 127
 Blocking keys 31
 BOX 88
 BREAK 13, 25, 128, 168
 BREAK 25, 88
 Buffer, input 98
 buffer, keyboard 1, 25, 128, 154
 see also LOCK: 7
 Bulletin board systems 118, 155
 C332 115
 Cabling 9
 25-Pin connection 10
 9-Pin connection 10
 CALC 89
 CALL 38
 CALL 89
 CAPS 13
 CAPTURE 75, 89, 104, 130
 CAPTURE 76, 130
 CAPTURE LONG 76, 89, 163
 CAPTURE LONG 130
 Caret 28
 CD 90
 CGA 158
 Characters, number on screen
 49
 CHARSET 90
 CHOOSEPRINT 90
 CLIP 90
 CLOSEI 90
 CLOSEO 91
 Color 159
 COLOR 91
 Columns, number on screen 49
 Command line parameters 121
 COMMAND.COM 111
 Commands 13, 26, 82
 Commands sent from the host
 computer 159
 COMMTYPE 91
 Communications protocol
 Keyboard locking 127
 Transmit priorities 127
 Communications protocol 127
 Control 28, 31
 COPY (file) 92
 COPY/S (file) 92
 CRC 129, 130
 CSV files 156
 CURSOR [BLINK] 92
 Custom-programmed extraction
 151
 DATA BITS 92
 Data Capture 61

DATE 92
 DEFAULTS 92
 Defaults file 15, 32, 121
 DEFINE 30
 DEFINE 92
 Defined keys
 <Enters> 33
 <F2> 36
 <F6> 36
 automatic execution 123
 copying 30
 embedded functions 35
 locally displayed text 34
 nesting 38
 pauses 34
 prefixing with '~' 37
 reloading them 32
 saving them 32
 special functions 33
 stacking 37
 tabs 33
 Defined keys 29
 DELAY 71, 75
 DELAY 93
 DELAY/S 93
 DELETE 93
 demo program 1
 DIAL 44
 DIAL 93
 DIR 106
 DIR 94
 DIR/S 94
 DOWNLOAD 165
 DROPOUT 94
 Editing defined keys 30
 EGA 158
 EJECT 94
 END 94
 Ending ANZIO 15
 Entering commands 26
 ENV/S 94
 Error diagnosis 161
 Error messages
 Communication chip errors
 143
 Text messages 144
 Error messages 143
 Escape sequences
 7900 134
 Additional functions 142
 Viewpoint/7901 135
 VT220 139
 Wyse 60 137
 Escape sequences 134
 F2 94
 File transfer 61, 128
 IMOS II 63
 IMOS III, IMOS V, IRX,
 ITX
 Kermit 70
 RECV-PC 67
 RECV-SPL 68
 SEND-PC 64
 Simple upload 63
 IMOS III, IMOS V, IRX,
 ITX 63
 Other kinds 78
 PC-to-PC 77
 QuickSend 78
 RM/COS 75
 Receive PC 76
 Send PC 75, 76
 Simple upload 75
 UFT 79
 UNIX
 DOWNLOAD shell
 script 72
 Kermit 73
 RECV-PC.C 73
 SEND-PC 72
 Simple upload 71
 UNIX 71

VRX 77
 File Transfer 62
 File transfer protocol 129
 FILL 95, 159
 FIND/S 95
 FINDNEXT/S 95
 Flicker 114
 FLUSH 95
 FLUSHTIMER 95
 FONT 95
 Framing Errors 143
 FULL 96
 Function Keys 24, 134
 FUNCTION PREFIX key 24
 GAUGE 96
 HALF 96
 Hanging up the phone 46
 HELP 24, 26, 96, 106, 127
 HELP <keyword> 96
 HELP ASCII 97
 HELP index 23, 96
 Help key 23
 Hercules 158
 HOLD 97
 Hookup 9
 HOSTNAME/S 97
 HOTKEY 97
 see also Memory-resident
 operation 7
 I-D area 39
 IGNUll 97
 IMOS 127
 IMOS 97
 IMOS II 63
 IMOS III 63
 IMOS V 63
 INSTALL.BAT 165
 Installing ANZIO 7
 Installing transfer programs 8
 INTERPRET 106
 INTERPRET 98
 Interrupt 10
 INVOKE 38
 INVOKE 98
 IRQ 98, 106, 161
 IRQ 98
 IRX 63, 98, 127
 ITX 63, 98, 127
 ITX | IRX | IMOS | RMCOS |
 VRX | UNIX 98
 JUMP 98
 KCOMMAND 99
 KEEP 104
 KEEP 61, 99
 KEEP/N 100
 Kermit 38, 70, 73, 74, 99, 100
 KERMIT.DOC 166
 KERMIT.KYS 166
 Key files 13
 Keyboard locking (see also
 LOCK) 127
 KEYS 29, 106
 KEYS 100
 Keys, Defined
 see Defined keys: 7
 Keys, Function
 see Function keys or
 individual key: 7
 Keys, Macro
 see Defined keys: 7
 kill (under UNIX) 153
 KRECEIVE 100
 KSEND 100
 LAUNCH 101
 LINE DELAY 101
 LOCK 13, 24, 44, 127, 129,
 153, 155, 161
 LOCK 101
 LOG 102
 Logging on 13, 168, 169
 Loopback test 161
 LOTUS 123 148, 149

LST: 66
Memory usage 58, 124, 125
Memory-resident operation 17,
114, 125
MENUBAR 102, 159
MENUDEMO.CBL 158, 164
MERGE 102
MESSAGE 102
Migration 160
MKDIR 102
MKDIR/S 102
MODE-132 103
Modem 44, 93
DIAL 44
Signing off 46
WAIT 46
MONITOR 103
Network operation 172
NUM 13
OPENI 103
OPENO 61
OPENO 104
Overdefining function keys 24,
31
Overrun errors 144
page mode 1
PAN 51, 105
Panic button 25, 45, 46, 128
Parameters (in command line)
121
PARITY 92
PARITY 105
Parity errors 143
Pass-through print - see Printing,
pass-through: 7
PASTE 105
Paths 11
PC software 148
PICK 51, 104, 150, 156, 159
PICK 61, 105, 149
Picking for LOTUS 149
PITCH 105
PLAY NCR 78, 105
PLAYSOUND 106
polling 1
PORT 10, 161
PORT 106
PRINT 94, 96, 98, 100, 107,
118
PRINT 62, 106
PRINT/N 106
PRINTER 66, 68, 99, 106, 131
PRINTER 107
PRINTER-SETUP 107
PRINTER-SETUP 107
PRINTFONT 108
Printing
pass-through 104, 131, 153
PRINTING 52
PRINTLOW 108
PURGE 109
Q-GRAPH.EXE 156, 159, 164
Query programs 149
QuickSend 78
Quitting ANZIO 15
READ 32, 102
READ 109
READ.ME 165
RECEIVE CODED 109
RECEIVE QUIET 69
RECEIVE QUIET 109
RECONNECT 109
RECV-PC.C 73, 165
RECV-PC.CBL 67, 117, 164
RECV-PC.RM 164
RECV-PC.RM 76
RECV-SPL.CBL 68, 164
RENAME 109
Report generators 149
RESET 110
RETRANSMIT 128
RETRANSMIT 110

REVIEW 105, 113
 REVIEW 51, 58, 110
 RM/COBOL 153
 RM/COS 75, 98
 RM/COS 154
 RMCOS.KYS 13, 164, 169
 RTS-MODE 110
 RUN 49, 110, 152, 159
 RUN/N 111
 Running ANZIO 11
 Sample defined keys 168
 SAMPLE.KYS 13, 164, 168
 SAVE 32, 102, 112
 SAVE 32
 Saving your default file 15
 SCOANSI 115
 SCREENMODE 49
 SCREENMODE 112
 SCREENMODE/S 113
 SCROLL 113
 SCROLL-LOCK 113
 SEND 113
 Sending commands from the host
 132
 SEND-L.RM 163
 SEND-L.RM 76
 SEND-PC.C 72, 163
 SEND-PC.CBL 64, 163
 SEND-PC.ECH 166, 167
 SEND-RM 163
 SEND-RM 75
 Serial Port 9
 SETCOLOR 113
 SET-EGA.DOC 165
 SET-EGA.EXE 165
 Shift 28, 31
 Sidekick 152
 SLEEP 113
 Snow (on screen) 114
 Split screen 59
 Spool file 65, 68
 SPOOLCRT.CBL 51, 164
 Spreadsheet 78
 Starting ANZIO 11, 121
 Status line 13
 STATUS LINE 114
 STAY 152
 STAY 114
 STAY/G 114
 STOP 114
 STOP BITS 114
 stty 153
 SUBDIRECTORIES 7, 11, 102
 SYNC 114
 TAB 1, 153, 155
 TAB 114
 TAB CHARACTER 115
 TERM 115
 Termcap 153
 Terminating ANZIO 15
 TERMNAME 116
 TIME 116
 TIMEOUT 116
 TITLE 116
 TRACK-WINDOW 116
 TRANSMIT 129
 TRANSMIT 63, 77, 117
 TRANSMIT CRC 117, 129
 TRANSMIT SINGLE 117
 TSR operation
 see Memory-resident
 operation 7
 TSR programs 111
 TTY 155
 TTY 118
 TYPE 106
 TYPE 118
 UFT 79
 Underline 39, 91, 165
 UNIX 98, 118, 153
 UNIX 71
 UNLOCK 128

UPPERCASE 118
Vector 10
VERSION 118
VERSION.EXE 164
VGA 158
Video attributes 19, 20, 39, 41,
42, 91, 106, 153
Viewpoint 1, 115, 153
VRX 98
VRX 77
VT100 19, 115
VT100.DOC 165
VT100.KYS 165
VT220 115, 139
VT220.DOC 165
VT220.KYS 165
VT220S.DOC 165
VT220S.KYS 165
WAIT 46, 118
WAITFOR 119
WF 119
WIDTH 50, 58, 105
WIDTH 58, 119
WINDOW 119, 159
WINDOWCLOSE 119
WINPRINT 120
WINSTART 120
Wiring 9
Word processing 78
WRITE 120
Wyse 60 19, 115, 137
Wyse50 115
WYSE60.DOC 166
WYSE60.KYS 166
XN 120
XON/XOFF 66, 131, 153
ZRECEIVE 120
ZSEND 120