PN-AAT-028
ISN
40520
ARCHIVE
12/14/84

Working Paper # 12
November 1984

HOW TO USE KERMIT FOR TRANSFERRING FILES BETWEEN MICROCOMPUTERS: CP/M AND MSDOS SYSTEMS

Norman Weatherby, Ph.D.

Center for Population and Family Health
Faculty of Medicine
Columbia University
60 Haven Avenue
New York, NY 10032

000378
000013
S

PREFACE

The use of microcomputers is rapidly increasing throughout the world. For example, in the last three years, the Center for Population and Family Health (the Center) has moved from having no microcomputers to the support of installations in Bolivia, Haiti, Nigeria, Tanzania, Nepal, the Sudan, and our central offices. We see continued growth in the use of microcomputers for project management and administration, accounting, the writing and editing of reports, and data entry, editing and analysis.

Although microcomputers speed the processing of information, there are problems with the transfer of information among systems. The first problem is distance--diskettes must be carried from one installation to another, or the information must be sent across telephone lines that are sometimes too noisy and full of static to be used even for voice communications. The second problem is one of compatability of diskettes across systems. For example, diskettes from Apple II series, Apple Macintosh, Hewlett-Packard, TurboDos, CP/M, and IBM PC-type systems are not interchangable. Even within product lines, such as the IBM PC series, problems may occur if different versions of the disk operating system are used to produce the diskettes. Thus, information may not be successfully transferred even if diskettes are carried from one system to another.

One solution to these problems is to use programs such as KERMIT that allow information to be electronically transmitted between systems, either across telephone lines or through a direct connection between the computers. Such programs must be able to transfer information without error even across poor quality telephone lines. The program running on one system must have a specific protocol, or method of packaging the information, that is used by the other system. Thus, a consistent protocol, used in programs for different computers, should allow the error-free transfer of information.

This document explains to the novice user how to get KERMIT to transfer files between certain types of microcomputers. It is useful to those who are in the difficult situation of needing to send information between incompatible microcomputer systems, either directly or across telephone lines. For example, by following the procedures in this guide, Center staff in Haiti will be able to transfer information from a Corona system to an Osborne Executive. This ability to transfer information quickly and without error is an essential element of the continued success of our research and service efforts.

INDEX

TOPIC	PAGE
Introduction	1
Now, KERMIT	2
Getting ready to use KERMIT	3
Connecting the computers	4
Loading KERMIT into memory	5
Setting KERMIT's options	6
Setting the baud rate	7
Communicating between computers	8
Sending and receiving files	9
Checking what was sent	9
Figuring out why (if) KERMIT does not work	10
Summary	11
Appendix 1 Current KERMIT versions	13
Appendix 2 Known KERMIT versions	15

INTRODUCTION

KERMIT (like the frog, a registered trademark of Henson Associates, Inc., used by permission) is ideal for transferring ASCII and binary files between computers of all sizes. It is a protocol for transferring sequential files between computers of all sizes over ordinary asynchronous telecommunication lines using packets, checksums, and retransmission to promote data integrity. KERMIT is non-proprietary, thoroughly documented, well tested, and in wide use. The protocol and the original implementations were developed at Columbia University and have been shared with many other institutions, some of which have made significant contributions of their own. KERMIT programs have been written for a wide variety of microcomputers, minicomputers, and mainframes.

Which ones? At the end of this document is an extensive listing that was downloaded (using KERMIT) from a DEC mainframe at Columbia University. In addition, you will find the address to write for further documentation, and some information for those who would like to participate in the growth and acceptance of KERMIT as one of the major file-transfer protocols.

This guide was written so that people who have very little computer experience could use KERMIT to transfer files between CP/M micros or between CP/M and MSDOS (IBM PC-type) systems. It simply tells you how to work with KERMIT, without going into its features. However, it is not a substitute for the published documentation. And, as the real documentation states:

> No warranty of the software nor of the accuracy of the documentation surrounding it is expressed or implied, and neither the authors nor Columbia University acknowledges any liability resulting from program or documentation errors.

KERMIT is copyrighted, and it is not "public domain". Simply, it is a free program that is not to be sold as a product. As the distributors state, you are free to redistribute KERMIT on your own terms, and are encouraged to do so, with the following stipulations: KERMIT should not be sold for profit; credit should be given where it is due; and new material should be sent back to Columbia University at the address given in the summary so that they can maintain a definitive and comprehensive set of KERMIT implementations and documentation for further distribution.

Where do you get KERMIT? As stated below, Columbia University distributes versions of the program only on nine track magnetic tape. You can get the program from friends or bulletin board systems. Look around for the latest version for your micro, and be aware that the program is constantly improving.

NOW, KERMIT

Without going into the details of KERMIT, this guide shows you how to use the program to transfer files between microcomputers that have KERMIT. The sections include

1. Getting ready to use KERMIT
2. Connecting the computers, either directly or through modems
3. Loading KERMIT into memory
4. Setting KERMIT's options
5. Setting the baud rate
6. Communicating between computers
7. Sending and receiving files
8. Checking what was sent
9. Figuring out why (if) KERMIT does not work
10. Summary

GETTING READY TO USE KERMIT

You will need the following items in order to use KERMIT for file transfers. The only real problem is given last--determining what kind of cable(s) you will need for the systems.

1. Two microcomputers. It is best if each one has two or more disk drives. Each one should have a serial port that can be used with a modem.

   IBM PC-type systems should have a DTE ("T" for terminal) port on an asynchronous communications adaptor (the serial port) that is called COM1:. Perhaps there is a serial port, but it is used to send information to a serial printer. In this case, the port may be DCE ("C" for communications) port. Using a DCE port will not cause problems if you have the correct cable. Note that the IBM dot matrix printer, which is made by Epson, is a parallel printer and not a serial printer. Never connect a modem or other serial device to the parallel port used for the parallel printer, even though it appears that the cable will plug into the connection.

   CP/M systems typically come with one or more serial ports. One should be for a modem (DTE) and another for a printer (DCE). In the CONFIGUR or SETUP program that came with the system, the modem port is usually labeled RDR:/PUN: or AUXIN:/AUXOUT:. The printer port is labeled LST: or PRT:. The words "modem" or "printer" may be stamped on the computer's case, but it is best not to pay attention to this, since you use the CONFIGUR or SETUP program to set the port you are going to use as a modem port. For example, the

Osborne Executive's "modem" port is not standard. In order to use KERMIT, you have to use the Executive's SETUP program to label the "printer" port as AUXIN:/AUXOUT:. Although this printer port is DCE, it can then be used with a modem if you have the right cable.

2. Two full duplex modems if you are going to transfer files across telephone lines. You should be able to set at least one of the modems to answer mode. File transfers work well but slowly at 300 baud (around 30 characters per second), so it is best to have two 1200 baud modems.

3. Two diskettes with KERMIT, one for each computer. These diskettes should contain the operating system of your computer so that you can boot the system off of them. You may also want to have system utilities, such as the CONFIGUR or SETUP program, on the diskettes.

4. The diskette with the file you want to send from one computer, and a blank, formatted diskette for the computer that will receive the file.

5. The right cable(s). You should be able to plug in the ends of the cable(s) to the equipment that you are using. Pins 1 through 8 and 20 are connected. Do not have other pins connected unless you are using a modem that was sold under the same brand name as the manufacturer of your computer, since some manufacturers of computers put electricity on other pins to power their own modems. This electricity can damage your modem. In addition, do not short-circuit any pins in the cable. For example, do not use a cable that has pins 4 and 5 connected, or "jumpered", at one end.

   The main question is whether or not pins 2 and 3 have to be switched. "Switched" means that the signals from pin 2 on one end will go to pin 3 on the other end. Likewise, signals from pin 3 on one end will go to pin 2 on the other end.

   IF YOU ARE DIRECTLY CONNECTING THE COMPUTERS:

   a. Do not switch pins 2 and 3 if you are connecting a modem (DTE) port on one system to a printer (DCE) port on the other system.

   b. Switch pins 2 and 3 if you are connecting a modem (DTE) port on one system to a modem (DTE) port on the other system, or if you are connecting a printer (DCE) port on one system to a printer (DCE) port on the other system.

IF YOU ARE USING MODEMS TO COMMUNICATE THROUGH THE TELEPHONE LINES:

For each system,

a. Do not switch pins 2 and 3 if you are connecting a modem (DTE) port to the modem.

b. Switch pins 2 and 3 if you are connecting a printer (DCE) port to the modem.

By making sure that you have your modems, diskettes, and the right cable(s) together before attempting to use KERMIT, you will avoid problems later on.

CONNECTING THE COMPUTERS

IF YOU ARE GOING TO USE MODEMS, use the correct cable to hook up a

full duplex modem

to each computer's serial communications port. If you are working with acoustically coupled modems, set one modem to answer mode and the other modem to originate mode.

IF YOU ARE GOING TO DIRECTLY CONNECT THE COMPUTERS,

a. Directly connect the serial modem ports of the two machines.

OR

b. Directly connect the serial modem port of one computer to the serial (not parallel) printer port of the other computer.

One benefit of using a direct connection between the computers is that you can use high baud rates, such as 4800 baud, to transfer the files. Since you are not using modems and telephones, ignore the parts of this guide that concern use of the modems and dialing of the telephone.

LOADING KERMIT INTO MEMORY

Loading KERMIT into the memory of an Apple II or Atari computer is more difficult than loading it into CP/M or MSDOS (PCDOS) systems. Thus, readers should refer to other documentation and help files for assistance with those systems.

IF YOU HAVE A HARD DISK: Micros with hard disks, such as the IBM PC-XT, often refer to all or part of the hard disk as disk C. If you have a hard disk, copy your KERMIT program and associated utility, help, and documentation files from the KERMIT floppy to the hard disk (if someone else has not copied them over already). Then work off the hard disk. If you do not see the prompt for the hard disk on your screen, type the command

c:<cr>

Then, to load KERMIT into memory, type the name of your program, which is usually KERMIT. For example, at your operating system's prompt, type

kermit<cr>

Then skip to the section below about setting KERMIT's options.

IF YOU HAVE TWO FLOPPY DRIVES: Put the KERMIT diskette in drive A of each computer. Note that on a CP/M machine such as the Kaypro, H89, or Osborne 1, it is wise to have the operating system on the KERMIT disk (through the use of the SYSGEN utility) and to type the command

cntl C (without a carriage return)

where cntl is the control key and C can be upper or lower case, to make the computer realize that a new disk has been put in drive A. This "warm boot" command is not necessary on micros that use CP/M 3.0, MSDOS, or PCDOS operating systems.

Put a blank, formatted disk in drive B of the system which is to receive the file. On the sending system, put the disk with the file that you want to transfer in drive B. Go to drive B, and then get into KERMIT on each machine by typing a: and the name of the program, which is usually KERMIT. For example, type

a:kermit<cr>

Newer versions of KERMIT allow you to change the "logged disk drive," but it is wise to follow the above advice so that you are sure of the disk from which you are sending files or to which you are receiving files.

SETTING KERMIT'S OPTIONS

To see the options that are available in your version of KERMIT, type at the prompt the command

status<cr>

Note that the toggle command set ibm (either on or off) is NOT FOR IBM PC MICROS—it is a command for IBM mainframes.

SETTING THE PARITY OPTION: A computer "byte" is composed of eight bits, where each bit is a zero or one. All (english) printable letters, numbers, punctuation marks, and spaces between words can be represented by seven of the eight bits. The eighth bit is reserved for checking to make sure that the other seven bits are correct. However, some microcomputer software packages (such as WordStar) use the eighth bit for special characters that allow features such as right justification. In any event, if both computers allow the following command, issue it to allow the transmission of all eight bits:

set parity none<cr>

SETTING THE BAUD RATE OPTION: The baud rate controls the speed of the file transfer. It can be set from within KERMIT on some CP/M machines, such as the Osborne 1 and the Intertec Superbrain, and on MSDOS (PCDOS) computers. One really nice feature of KERMIT is that you do not have to cope with that poorly-documented mode command on the IBM PC-type micros.

If you can set the baud rate from within KERMIT, make the baud rates of both computers equal by typing at the prompt the command

set baud ? <cr>

and select the correct baud rate from a menu that is given on the screen. Directly connected computers can communicate at 4800 baud. Acoustically coupled modems operate at 300 baud, and some SmartModems allow 1200 baud to be used. Then go to the section of this guide that discusses communicating between computers.

If you get messages that the set baud command is not implemented, you will have to get out of KERMIT and set it from your operating system, as discussed below.

Note that a faster way to set the baud rate is to type

set baud xxxx<cr>

where xxxx is the baud rate you want.

SETTING THE BAUD RATE

If you cannot set the baud rate from within KERMIT, such as on Heath/Zenith 89 systems, then you can get back to your operating system by typing, at the prompt,

exit<cr>

Then use the computer's software to set the baud rate of the serial port to which the cable is attached. This is 300 or 1200 baud when modems are used and 4800 baud when a direct connection is made. The baud rate of a CP/M machine is set through the SETUP or CONFIGUR utilities supplied with the system.

While you are setting the baud rate, you can also check to make sure that the serial port is set up as the RDR:/PUN: or AUXIN:/AUXOUT: device.

Then load KERMIT back into the memory of your system and reset its options.

COMMUNICATING BETWEEN COMPUTERS

After KERMIT is loaded into the memory of each computer, and its options are set, type at the prompt the command

connect<cr>

on each computer. This will put you into the connect mode, which allows one computer operator to send messages to the other operator through the modems or the direct connection. Note that KERMIT replies with an "escape" message that tells you how to get back to the command state of the program. WRITE DOWN THIS MESSAGE, AS YOU WILL NEED IT LATER.

In fact, it is a good idea to test the command now. The way you get back to the KERMIT prompt varies by the type of system. Note that you may have to issue the command but add a C (upper or lower case) to it to get back to the KERMIT prompt. For example, on many CP/M machines that have the backslash character \ , the "escape" command is control \ , but you type

ctrl\c (without typing a return)

where ctrl is the control key. After some experimentation, you will see the KERMIT prompt, and you can again type the command connect<cr> .

IF YOU ARE USING MODEMS: From the originate modem, dial the answering modem. If you have an auto-dial modem, you can issue the dial command to it. For example, with a touch tone telephone and a Hayes SmartModem, the only command that is necessary for the answering modem is

ATS0=1<cr>

to make sure that the modem answers on the first ring.

The only command needed for the originate SmartModem is

ATDTxxxxxxx<cr>

where xxxxxxx is the telephone number to dial.

IF YOU ARE USING MODEMS OR A DIRECT CONNECTION: When the connection is made, each of you will be able to send and receive messages (for example, "HELLO") that serve to test the connection.

If you want to see what you are typing, get back to the KERMIT prompt by typing the "escape" command that you wrote down and tested above. Then issue the command

set loc on<cr>

to have a local echo, but you must remember to

set loc off<cr>

before attempting to transfer a file.

If the connection is not made, so that it is not possible to send AND receive messages, then you should check

the baud rates of the computers
the modems, and the wiring to the modems
or the wiring between the directly-connected computers.

Make sure that the computer is sending or receiving data to the correct port on the computer. In order to make these checks, you may want to return to the beginning of this guide and start over. This is necessary so that you will not omit any important step in the use of KERMIT, such as setting the baud rate. It is also helpful, as a last resort, to check whether or not the telephone died because you forgot to pay the bill (don't laugh...it once happened to us!)

SENDING AND RECEIVING FILES

When each computer operator is satisfied that a connection has been made, then both operators return to the KERMIT prompt, as explained above.

The operator who wants to receive a file types

receive<cr>

The file name will be sent from the other system.

The operator who wants to send a file types the command

send [drive:]filename.ext<cr>

For example, to send the file on disk B that is named myfile.txt , the sending operator would type

send b:myfile.txt<cr>

It is usually not necessary to type in the drive specification, since you should be using disk B (or C with a hard disk) as the "logged disk drive", but it is always safe to do so. If the receiving operator presses the return key after the sending operator presses return, it may be necessary for the receiving operator to enter another return so that the file will be sent.

KERMIT will wait for a few seconds, and then the operators will see on their screens that packets of information are being sent and received. KERMIT first sends the file name, and it is a good sign to the receiving operator when the file name appears on his or her screen. The number of packets will increment on both machines until the transfer is complete, and then each computer will return to the KERMIT prompt. At that time, the receiving operator can check the file or another file can be sent.

CHECKING WHAT WAS SENT

Once a connection has been established with KERMIT, it is not broken if one or both of the operators return to the operating system of their computers to check something such as the length or name of a file. Thus, when a transfer is complete, the receiving operator can get out of KERMIT by typing

exit<cr>

at the KERMIT prompt. The file that was received can be examined, usually by issuing the command

type [drive:]filename.ext<cr>

The file will scroll by on the screen, and you will see that it has been transferred without error!

The receiving operator can then return to KERMIT by reloading it into memory (discussed above). If you need to, reset the baud rate and the parity, and you will then be able to transfer another file or get into connect mode and send messages.

FIGURING OUT WHY (IF) KERMIT DOES NOT WORK

KERMIT has never, in our two years of experience with the program, made a mistake in sending and receiving files between CP/M computers or between CP/M computers and mainframes. It accurately transfers files even when the telephone line is unusable for voice transmissions because of static and noise.

Operators do, however, tend to make mistakes. One such problem is impatience. Please let KERMIT wait for a few seconds before you touch a carriage return on the receiving computer to "make it work." If KERMIT seems to be dead after about thirty seconds of waiting, then something is wrong. If you were able to send messages back and forth in connect mode, or if KERMIT fails after the first four or five packets, then probably the problem is that one of the operators set the local echo switch on but forgot to set it off. It is also possible that the file that you wanted to send does not exist, or the sending operator misspelled its name in the send command.

Occasionally, KERMIT will fail for reasons beyond its control. The transfer will fail if one of the computers involved fails. For example, mainframe computers or their communication ports often stop (that's why we are now using micros for most of our work, right?). A feature such as call waiting or intercom messages on a telephone line will stop the transmission if they happen during a transfer. These failures are advantages in that they show that KERMIT is smart enough to quit when there is a major problem.

We have also encountered a problem with the way BASICA and dBASE II on IBM PC-type computers mark the end of a file and save it. The solution is to load the file into a text editor or word processor and then resave it before sending it. For example, load a file into WordStar and then save it. An unreliable alternative is to copy the file using the /A parameter. The problem is not fully understood; it centers around the PCDOS and MSDOS software's use of buffered output when writing disk files.

Apple II versions of KERMIT will work with DOS 3.3. However, Apple IIs are slow, there are differences between the II, the II+ and the IIe, and there is no standard way to set up Apples for communications. If you plan to use Apple KERMIT, please check with other Apple KERMIT users before ordering your communications card and modem. Currently, the best cards may be the Apple Communications card, the Hayes Micromodem II, or the Super Serial Card. Actually, the best way may be to install a Z80 card in the Apple II and run the Apple CP/M version of KERMIT.

SUMMARY

KERMIT is a very powerful program, with extensive error checking, and thus it is ideal for transferring files between computers. The full documentation will show you many commands and options for the program. You may want to order the documentation, but the simple procedures in this guide may be adequate for your needs.

Extensive documentation is available for this program from

KERMIT Distribution
Columbia University Center for Computing Activities
7th Floor, Watson Laboratory
612 West 115th Street
New York, NY 10025

There are three publications: the User's Guide, the Protocol Manual, and the Source Listing for your system. Each costs $5.00, and you should make out your check to Columbia University Center for Computing Activities. Make sure and tell them what microcomputer and operating system you are using. Those who are interested in mainframe implementations (which currently cost $100 for the complete distribution) should write for ordering information. Note that the KERMIT distributors can provide the programs only on 9-track magnetic tape in a varity of formats. They do not provide the program on floppies.

When necessary, there are specific help files that are distributed with KERMIT to help with its use. For example, there is a help file with the Kaypro II version because of the non-standard way it handles tab stops on its screen.

There are also articles and announcements about KERMIT. For example, see the June and July, 1984, issues of BYTE.

Please do not call the Kermit Distribution Center for assistance with KERMIT for your micro. After all, it is a free program, and national support, sales, service, debugging,

revisions, everything else is being handled by only two or three part-time people. You can write, however, for the publications.

Finally, KERMIT is changing and improving constantly — versions are added for new systems, old versions are improved, and documentation is rewritten. You are encouraged to make your own contributions to "KERMIT culture." Make them available to your friends, and send them back to Columbia at the address given above, on 9-track magnetic tape or IBM PC single-sided, eight sector floppy disk. For example, we need KERMIT for the TurboDos operating system.

The most current versions of KERMIT are listed in Appendix 1. A brand-new MSDOS version that acts as a server for remote communications has been released. Appendix 2 lists the systems for which KERMIT is available. The codes shown with each group show the status of the versions.

Good luck with KERMIT, and watch for new releases and updates!

APPENDIX 1

CURRENT KERMIT VERSIONS
as of 1:41pm Monday, 29 October, 1984

Listed in reverse chronological order of installation at Columbia. "Code" is the prefix under which the files are stored in the KERMIT distribution area.

Code	Machine/OS	Language	Ver #	dd/mm/yy	Author or Contact
MS	TI Pro/MS-DOS	MASM	2.26	29/10/84	Joe Smith, CSM
HP3	HP 3000/MPE	SPL	1.0	25/10/84	POLARIS@USC-ISI
CPM	CompuproIF3/4,CPM	ASM	3.9A	20/10/84	PS1.YAAGP@CU20B
20	DEC-20/TOPS-20	MACRO-20	4.2(251)	18/10/84	SY.FDC@CU20B
CR	Cray-1,Cray-XMP	Fortran77	-	18/10/84	lfm@lanl
MC1	Apple Macintosh	C (SUMACC)	-	16/10/84	engel@harvard
H8	Harris 800/VOS	Pascal/Asm	-	11/10/84	STEVENS@MACCWISC.MAILNET
VX	VAX/VMS	Pascal/Fortran	1.1E	11/10/84	P.Murton, U Toronto
RT	PDP11/RT11	OMSI Pascal	2.2C	11/10/84	P.Murton, U Toronto
86/APC	NEC APC/CPM-86	ASM86	2.8	10/10/84	CONTEXT@WASHINGTON
86/RB	Rainbow/CPM-86	ASM86	2.8	10/10/84	CONTEXT@WASHINGTON
AP2	Apple II/DOS	AppleAssembler	2.?	9/10/84	Olaf Pors, U of Va.
UN	Sperry/Univac1100	NOSC Pascal	2.0	8/10/84	Butt@UMD2
MAC80	8080 Cross Asmblr	MACRO-10	10F(121)	8/10/84	CERRITOS@USC-ECL
HL6	Honeywell L6/10	MASM	1.20A	5/10/84	Carlin%pco@MIT-MULTICS
HG	HoneywellDPS8,66	C	3.0	5/10/84	Carlin%pco@MIT-MULTICS
MS	H/Z-100	MASM	2.26	21/09/84	knutson@ut-ngp
MU	Honeywell/MULTICS	PL/I	2.0h	20/09/84	P.Amaranth, Oakland U
UN	Sperry/Univac1100	Exec Assembler		17/09/84	STEVENS@MACCWISC.MAILNET
AOS	DG S/250/AOS	Fortran		14/09/84	John Lee, RCA Labs
RDOS	DG Nova/4/RDOS	Fortran		14/09/84	John Lee, RCA Labs
HPM	HP1000/RTE	Fortran	1.0	14/09/84	John Lee, RCA Labs
APO	Apollo/Aegis	Fortran	1.0	14/09/84	John Lee, RCA Labs
K11	Pro350/POS	MACRO-11	2.22	12/09/84	ATSBDN@UOFT01.BITNET
K11	PDP11/RSX11M,M+	MACRO-11	2.22	12/09/84	ATSBDN@UOFT01.BITNET
K11	PDP11/RSTS	MACRO-11	2.22	12/09/84	ATSBDN@UOFT01.BITNET
K11	PDP11/RT-11	MACRO-11	2.22	12/09/84	ATSBDN@UOFT01.BITNET
APP	Apple II/DOS	CROSS	2.1A	12/09/84	OC.BUSH@CU20B
VIC	Victor 9000	CI C-86	1.0	7/09/84	W.Hertha, Victor Canada
170	Cyber 170/NOS	Fortran77,asm	2.2	7/09/84	knutson@UT-NGP
MS	NEC APC	MASM	2.26	7/09/84	CONTEXT@WASHINGTON
TRS	TRS-80 I and III	M80	3.5	8/08/84	stan@RICE
UX	(various)	Unix	3.0	2/08/84	SY.FDC@CU20B
KUSER	"KERMIT User Guide"	English	5.1	27/07/84	SY.FDC@CU20B
MS	IBM PC,XT/P-DOS	MASM	2.26	26/07/84	SY.DAPHNE@CU20B
MS	Rainbow100/MS-DOS	MASM	2.26	26/07/84	SY.DAPHNE@CU20B
MS	HP-150/MS-DOS	MASM	2.26	26/07/84	SY.DAPHNE@CU20B
MS	Wang PC/MS-DOS	MASM	2.26	26/07/84	SY.DAPHNE@CU20B
MS	Generic MS-DOS	MASM	2.26	26/07/84	SY.DAPHNE@CU20B
TSO	IBM 370/MVS/TSO	IBM Asm	1.0	18/07/84	SYSRONR@UCHIVM1.BITNET

SIR	Sirius-1	MASM	1.20	12/06/84	Barry Devlin, Dublin
ST	HP3000/SoftwareTools/Ratfor		1n	18/02/84	kpd%HP-LABS@CSNET-Relay
ST	Univac/SoftwareTools/Ratfor		1n	11/06/84	kpd%HP-LABS@CSNET-Relay
CPM	Apple II/CPM-80	ASM	3.9A	6/06/84	SY.FDC@CU20B
CPM	DECmate II/CPM80	ASM	3.9A	6/06/84	SY.FDC@CU20B
CPM	Generic CPM80 2.2	ASM	3.9A	6/06/84	SY.FDC@CU20B
CPM	Generic CPM80 3.0	ASM	3.9A	6/06/84	CERRITOS@USC-ECL
CPM	H/Z-89/CPM-80	ASM	3.9A	6/06/84	SY.FDC@CU20B
CPM	H/Z-100/CPM-85	ASM	3.9A	6/06/84	SY.FDC@CU20B
CPM	Kaypro II/CPM-80	ASM	3.9A	6/06/84	SY.FDC@CU20B
CPM	Osborne 1/CPM-80	ASM	3.9A	6/06/84	SY.FDC@CU20B
CPM	Superbrain/CPM80	ASM	3.9A	6/06/84	SY.FDC@CU20B
CPM	TRS80-II/CPM80	ASM	3.9A	6/06/84	CERRITOS@USC-ECL
CPM	Vector Graphx/CPM	ASM	3.9A	6/06/84	SY.FDC@CU20B
CPM	VT180/CPM-80	ASM	3.9A	6/06/84	SY.FDC@CU20B
CPM	TelconZorba/CPM80	ASM	3.9A	6/06/84	OC.BUSH@CU20B
PRO	DEC Pro-350	Bliss	1.0	1/06/84	OC.BUSH@CU20B
VMS	VAX/VMS	Bliss	3.6.051	1/06/04	OC.BUSH@CU20B
K10	DEC-10/TOPS-10	MACRO/Bliss	3(124)	1/06/84	OC.BUSH@CU20B
UCI	IBM PC/p-System	UCSD Pascal	0.1	23/05/84	KMM@CORNELLA.BITNET
RBLC	Rainbow/MS-DOS	CI/86	2.14	23/05/84	LCAMPBELL@DEC-MARLBORO
800	ABC-800/ABCDOS	BASIC-II	2.2	8/05/84	Per_Lindberg_QZ@QZCOM
CPM	Morrow Decision I	ASM	3.9	26/04/84	SY.FDC@CU20B
CPM	Nokia MikroMikko	ASM	3.9	26/04/84	SY.FDC@CU20B
CPM	Ohio Sci/CPM-80	ASM	3.9	26/04/84	SY.FDC@CU20B
MP	PDP-11/MUMPS-11	MUMPS(1982)		11/04/84	KMM@CORNELLA.BITNET
UCT	Terak/p-System	UCSD Pascal		11/04/84	KMM@CORNELLA.BITNET
KPROTO	"KERMIT Protocol Manual"		5th Ed	2/04/84	SY.FDC@CU20B
TA2	Tandy 2000/MS DOS	MASM	1.20	16/02/84	smp@UTEXAS-11
PRI	Prime/PRIMOS	PL/P		10/02/84	L.Spira, The SOURCE
CMS	IBM 370/VM/CMS	IBM Asm		1/02/84	SY.DAPHNE@CU20B
MDS	Intel DevSys/ISIS	PL/M		27/01/84	Chris@COLUMBIA-20
HP9	HP-98xx/p-System	HP Pascal		20/01/84	Gallaher@RUTGERS
ATA	Atari/DOS	Action!		9/01/84	G.TANG@SU-SCORE
MTS	IBM 370/MTS	Asm, Pascal		6/01/84	"Gavin Eadie"b@UMich-MTS
PC	IBM PC etc/PC-DOS	MASM	1.20	30/11/83	SY.DAPHNE@CU20B
PC	H/Z-100	MASM	1.20	30/11/83	SY.DAPHNE@CU20B
VIC	Victor 9000/CPM86	ASM86	1.1	25/11/83	Barry Devlin, Dublin
SEE	Seequa/MS DOS	MASM	1.18	17/11/83	Glenn Everhart, RCA
VIC	Victor 9000/MSDOS	MASM	1.18	10/11/83	Bryan Peterson, ORNL
VIC	Victor 9000/MSDOS	MASM	1.1	3/06/83	Barry Devlin, Dublin
VX	VAX/VMS	VAX-11 C		24/04/83	Todd Little, DEC

APPENDIX 2

KNOWN KERMIT VERSIONS
as of 29 Oct 84

Status:
- D Done, no further development going on
- C Continuing; already released, but a new release is in preparation
- P Work in Progress on initial release
- G Good intentions (they said they were going to, but no further word)

Availability:
- A Available from Columbia
- W Columbia is waiting for it

Level (a very rough indication):
- . Basic (send & receive files, terminal emulation if micro)
- - Sub-Basic (works, but missing some features, like wildcard send)
- + Advanced (includes server functions &/or data compression, CRCs, ...)
- ? Unknown

Machine	OS	Language	Status	Who
DEC-10	TOPS-10	MACRO-10/Bliss	C A +	Stevens Institute of Technology
DEC-20	TOPS-20	MACRO-20	C A +	Columbia University
DEC VAX	VMS	MACRO-32/Bliss	C A +	Stevens Institute of Technology
DEC VAX	VMS	Fortran/Pascal	D A .	Bruce Pinn, U of Toronto
DEC PDP-11	RT-11	OMSI Pascal	D A .	U of Toronto
DEC PDP-11	RT-11	MACRO-11	C A +	Brian Nelson, U of Toledo
DEC PDP-11	RSTS/E	MACRO-11	C A +	Brian Nelson, U of Toledo
DEC PDP-11	RSX-11/M	MACRO-11	C A +	Brian Nelson, U of Toledo
DEC PDP-11	MUMPS	MUMPS	D A .	David Rossiter, Cornell U
DEC Pro-3xx	P/OS,RT-11	MACRO-11	P W +	Brian Nelson, U of Toledo
DEC Pro-3xx	P/OS	MACRO-11/Bliss	C A +	Stevens Institute of Technology
DEC Pro-3xx	P/OS	DECUS C	P W ?	Bob Denny, Alisa Systems
DEC PDP-8	OS8,RTS8	?	G W ?	R. Hippe, Bureau of Engraving Inc
IBM 370,etc	VM/CMS	IBM assembler	C A .	Columbia
IBM 370,etc	MVS/TSO	IBM assembler	D A .	Ron Rusnak, U of Chicago
IBM 370,etc	MVS/TSO	PL/I	P W +	Chuck Bacon, NIH
IBM 370,etc	MTS	Assembler	D A +	Gavin Eadie, U of Michigan
IBM 370,etc	MTS	Pascal/vs	D A .	Bill Hall, Math Reviews
IBM Series1	EDX	EDL?	G W ?	Milt Nussbaum, Citicorp Info Svcs
Burroughs B7900		?	D W ?	BJM Morselt, Eindhoven U of Tech
CDC Cyber	NOS	Fortran-77	C A +	Jim Knutson, U Texas, Austin
Cray-1,XMP	CTSS	Fortran-77	D A +	Leah Miller, LANL
Elxsi 6400	EMBOS	C	G W ?	David Spector, NYU
Honeywell	MULTICS	PL/I	C A +	Paul Amaranth, Oakland U, Mich.
Honeywell	CP-6	?	G W ?	Jim Williams, HIS LADC
Honeywell	DPS/GCOS	C	D A .	Terry Carlin, Honeywell
Univac 1100	EXEC	Assembler	D A .	Paul Stevens, U. of Wisconsin
Univac 1100	EXEC/ST	Ratfor	D A .	Tidwell & Cole, Univ. of Utah

Univac 1100	EXEC	NOSC Pascal	D A +	Edgar Butt, U of Md
DG S/250	AOS	Fortran	D A .	John Lee, RCA Labs
DG Nova/4	RDOS	Fortran	D A .	John Lee, RCA Labs
Harris 800	VOS	Pascal,Asm	D A ?	David Wilson, U of Wisconsin
HP1000,3000	Software	Tools / Ratfor	C A .	Ken Poulton, HP Labs
HP3000	MPE	SPL	D A +	Ed Eldridge, Polaris Inc.
HP1000	RTE	Fortran	D A .	John Lee, RCA Labs
Naked Mini	OS4?	?	G W ?	P.C. Hariharan, Johns Hopkins U
PRIME	PRIMOS	PL/I	C A +	Leslie Spira, The SOURCE
Various	UNIX	C	C A .	Columbia and others
Apollo	Aegis	Fortran	D A .	John Lee, RCA Labs
Apollo	Aegis	Pascal	D W ?	CDC
68000	Forth	Forth	D W ?	Jon Steer, Videotel Inc.
XORMacs 68K	VersaDOS	C?	G W ?	Hans Muller, AI&DS, Mt.View, CA
Apple Lisa			G W ?	Apple
Apple Macintosh			C A -	Harvard
Apple Macintosh			P W ?	Columbia, Rice, Brown, others
Xerox 1100	(InterLisp-D)		P W ?	Jon L White, Xerox-PARC
LispM	(Common Lisp)		P W ?	George Carrette, MIT
Symbolics 3600	Lisp Machine		P W ?	Bruce Nemnich, MIT
Various	UCSD p-System/Pascal			
Terak			D A .	Kate McGregor, Cornell U
HP-98x6		(HP Pascal)	D A .	Mike Gallaher, Rutgers U
IBM PC			D A .	Kate McGregor, Cornell U
Corvus Concept			G W ?	David Todd, Wesleyan U
Sage II & IV			D W ?	Gary Fostel, NCSU
Sage II			G W ?	Scott Miller, Honeywell
8088/8086	MS/PC-DOS	MASM	D A ./+	Columbia & others
IBM PC, XT		Zenith Z100		Tandy 2000
Rainbow 100		HP-150		Wang PC
TI Professional		Victor 9000/Sirius-1		Generic DOS
Seequa Chameleon				
IBM PCjr	PC DOS	MASM	P W +	Columbia
IBM PC/AT	PC DOS	MASM	P W +	Columbia
Honeywell MicroSystem 6/10		MSDOS	D A .	Terry Carlin, Honeywell
Victor 9000	MS DOS	CI C-86	D A .	W. Hertha, Victor Canada
Rainbow 100	CP/M-86	ASM86	C A +	Bill Catchings, CU, & others
NEC APC	CP/M-86	ASM86	C A +	Ron Blanford, U of Washington
Victor 9000	CP/M-86	ASM86	C A .	Barry Devlin, Univ College Dublin
IBM PC,XT	CP/M-86	ASM86	P W +	John Klensin, MIT
Intel MDS	ISIS	PL/M	D A -	Columbia University
Compupro	MP/M-86	C?	G W ?	Mark Pahlavan, UCB Coop Extension
Luxor ABC-800	ABCDOS	Basic-II	D A ?	Torbjorn Alm, ABC club, Stockholm
SWTPc 6809	FLEX	?	G W ?	John Guidi, Jackson Lab, Maine

Z80/8080	CP/M-80	8080 assembler	C	A +	Columbia, w/other contributors
DEC VT180		DECmate II (CP/M only)			Ohio Scientific
Heath/Zenith-89		Heath/Zenith-100 (8085 side)			Telcon Zorba
Kaypro II		Apple II (Z80 soft card)			"Generic" CP/M 2.2
TRS-80 II (CP/M only)		Osborne I			"Generic" CP/M 3.0
Intertec SuperBrain		Compupro Interfacer 3/4			Vector Graphics

(All these versions are built from a common source with conditional assembly)

TRS80 I&III	TRSDOS	M80	D	A +	Stan Barber, Rice University
TRS CoCo	os9	C	G	W ?	Bob Larson, USC-ECLB
Atari	DOS	Action!	D	A .	John Palevich, Atari
Apple II	DOS	Cross-10/20	C	A .	A. Mione, Stevens Inst of Tech
Commodore-64		C64 asm	P	W ?	Eric Lavitsky, Rutgers, et al.
Commodore-64		C64-Forth A	D	W ?	RW Detenbeck, U of Vermont