Kermit in ACTION!#

General Information
Author: John Howard Palevich
Language: ACTION!
Compiler/Interpreter: ACTION!

Kermit in ACTION!#

a Kermit implementation in ACTION!

How to install Kermit on your ATARI home computer.

RAM: 48K, or more RAM
Peripherals: At least one disk drive
ATARI 850 & a modem, or
ATARI 1030, or other communications device

---

1. Format a diskette and write a copy of DOS onto it.
2. Write the AUTORUN.SYS file for the type of modem that you are going to use. If you are using an 850, write the AUTORUN.SYS file that came with the DOS II Master Diskette.
3. Write all the K*.* files onto the diskette.
4. Insert an ACTION! cartridge into your ATARI computer, turn on your 850 (or 1030, or whatever) and power cycle your ATARI computer. After the DOS boots you should see the ACTION! editor screen.
5. Read in and edit the file "KERMIT.ACT". Change the line near the end of the file from "KCOM850.ACT" to whichever device you wish to use. Write out the "KERMIT.ACT" file when you are done. If you are trying to support a new modem type, create a new KCOM file and use its name here....
6. Clear the editor buffer and go to the ACTION! monitor. Type R "KERMIT.ACT" to compile and run the Kermit program.

That's it. Here are some commonly asked questions, with some off-the-cuff answers:

Q: Why do I need an ACTION! cartridge to run Kermit?

A: The people who developed ACTION! have provided a standalone runtime (Edited by Carsten Strotmann).

Q: Why do I have to re-compile the program every time I want to use it?

A: DOS II's menu program destroys the R: and T: device drivers, so you can't use the "L" menu option to run a pre-compiled ACTION program that depends on the R: or T: drivers.

If you are clever, you can append the ACTION! object code to the AUTORUN.SYS file to generate an auto-booting version of KERMIT.

If you have OS/A+, you can, indeed, save the compiled version of Kermit and execute it from the command line. Just make sure that you have loaded a device driver first!

Jack Palevich

KERMIT .PNS -- a sample phones file

```
SU-Score(300)#4153221570
SU-Score(1200)#4154970061
```

```
;D:KCOM1030 .ACT

;All the communications stuff:
;
;Opening, closing and dialing for
;the ATARI 1030 modem
;
; KERMIT protocol
; for Atari Home Computers
; version 1.1
; (C) 1983 John Howard Palevich
; to be distributed free of charge
;
;Started NOVEMBER 5, 1983

;Print a string which will identify,
;to the user, what hardware this
;COM file supports

PROC MODEMINIT()
PRINTE("for the Atari 1030 modem")
RETURN

;Return number of character in the
;input buffer

CARD FUNC NCIB()
BYTE INCNT = $400
RETURN(INCNT)

;Put a character out the modem

PROC PUTR(BYTE DATA)
PUTD(2, DATA)
RETURN

;Put out a byte as a modem command

PROC PUTCMD(BYTE CMD)
BYTE CMCMD = $0007
CMCMD = $FF
PUTD(2, 27)
PUTD(2, CMD)
CMCMD = 0
RETURN

;Temporarily Suspend Communications
;so that file I/O can take place

PROC StopR()
PUTCMD('Z)
RETURN
```

```
;Close down the modem channel

PROC CloseR()
PUTCMD('Y)
CLOSE(2)
RETURN

;Initialize communications

BYTE FUNC OpenR()
STRING fname = "##:"
BYTE T
Close(2)
fname(1) = 'T
fname(2) = '1
t = 12
Open(2, fname, t, 0)
T = MSTATUS(2)
IF T >= 128
THEN
PRINTF("Can't open %S, error %B%E",
fname, T)
CLOSE(2)
RETURN(T)
FI

RETURN(0)

PROC StartR()
PUTCMD('Y) ;Resume operation
PUTCMD('A)
PUTR($20)
PUTR('?) ;No Translation
PUTCMD('C)
PUTR(PARITY)
RETURN

;SubEQ(S, I, SS)
;
; Check if SS is = S(I..I+Len(SS)-1)

BYTE FUNC SUBEQ(STRING S BYTE I STRING SS)
INT J
IF S(0)-I+1 < SS(0) THEN RETURN(0) FI

FOR J = 1 TO SS(0) DO
IF S(I+J-1) <> SS(J) THEN
RETURN(0)
FI
OD

RETURN(1)


;Dial the number in string P
;return 0 if failure, 1 if OK

BYTE FUNC AutoDial(STRING P)
```

```
BYTE I, NN, C, DVSTAT1 = $2EB

NN = P(0)     ;LENGTH OF STRING

;This modem ignores baud rate

FOR C = 1 TO NN
DO
IF P(C) = '# THEN
DO
C ==+ 1
UNTIL
C > NN OR P(C) > 32
OD
EXIT
FI
OD
IF C > NN  THEN
PRINTE("No phone number in this entry!")
RETURN(0)
FI

PRINTE("Dialing...press any key to abort")
ERRORNUM = 0
STARTR()

IF dial = 0 THEN
PUTCMD('N)
ELSE
PUTCMD('O)
FI

PUTCMD('K)
FOR I = C TO NN
DO
PutR(P(I))
OD
PutR($9B)

;Wait for carrier
WHILE CH = $FF DO
MDEVSTAT(2)
IF (DVSTAT1 & $80) <> 0 THEN
RETURN(1)
FI
OD
PRINTE("User abort")
PUTCMD('M) ;Go on-hook
STOPR()
RETURN(0)

;Hang up the phone line

PROC HANGUP()
STARTR()
PUTCMD('M) ;Go on-hook
STOPR()
RETURN
```

```
; --- END OF D:KCOM1030.ACT ---

;D: KCOM850. ACT

;All the communications stuff:
;
; Opening, closing and
; DIALING
; (for the DC-Hayes Smartmodem)
; KERMIT protocol
; for Atari Home Computers
; version 1.1
; (C) 1983 John Howard Palevich
; to be distributed free of charge
;
;Started NOVEMBER 5, 1983

PROC MODEMINIT()
PRINTE("for the Atari 850 and the")
PRINTE("DC-Hayes Smartmodem")
RETURN

CARD FUNC NCIB()
CARD NC = 747,
INCNT = $400
BYTE I
MDEVSTAT(2)
I = MSTATUS(2)
IF I >= 128 THEN
PRINTF("R: device error: %D%E",
I)
RETURN(0)
FI
RETURN(NC)

PROC PUTR(BYTE DATA)
PUTD(2, DATA)
RETURN

;Temporarily Suspend Communications I/O

PROC StopR()
Close(2)
RETURN

PROC CloseR()
CLOSE(2)
RETURN

BYTE FUNC OpenR()
STRING fname = "##:"
BYTE T
Close(2)
fname(1) = 'R
fname(2) = dnum + '0
t = 13
Open(2, fname, t, 0)
T = MSTATUS(2)
IF T >= 128
THEN
```

```
PRINTF("Can't open %S, error %B%E",
fname, T)
CLOSE(2)
RETURN(T)
FI

CIOV(2, 34, 0, 0, 192+48, 0)
CIOV(2, 38, 0, 0, 32+PARITY*5, 0)
CIOV(2, 36, 0, 0, 8+baud, 0)
CIOV(2, 40, 0, 0, 0, 0)
RETURN(0)

PROC StartR()
OpenR()
RETURN

;SubEQ(S, I, SS)
;
; Check if SS is = S(I..I+Len(SS)-1)

BYTE FUNC SUBEQ(STRING S BYTE I STRING SS)
INT J
IF S(0)-I+1 < SS(0) THEN RETURN(0) FI

FOR J = 1 TO SS(0) DO
IF S(I+J-1) <> SS(J) THEN
RETURN(0)
FI
OD

RETURN(1)

;GetMack() - wait for reply from SM
PROC GetMack()
BYTE A, S
IF ERRORNUM >= 128 THEN RETURN
FI
S = 0
DO
IF CH <> $FF THEN
ERRORNUM = $FF
RETURN
FI
IF NCIB() > 0 THEN
A = GETD(2)
IF DEBUG = 1 THEN
PUT(27)
PUT(A)
FI
IF S = 0 THEN
IF A >= 32 THEN
S = 1
FI
ELSE
IF A = 10 THEN ;End of reply
RETURN
FI
FI
FI
```

```
OD

;PutMatch(c) - put a character out
; to R:, wait for a matching character
; or user's abort

PROC PutMatch(BYTE c)
BYTE A
PUTD(2, C)
IF ERRORNUM >= 128 THEN RETURN
FI
DO
IF CH <> $FF THEN
ERRORNUM = $FF
RETURN
FI
IF NCIB() > 0 THEN
A = GETD(2)
IF DEBUG = 1 THEN
PUT(27)
PUT(A)
FI
IF A = C THEN
RETURN
FI
FI
OD

;Dial the number in string P....

BYTE FUNC AUTODIAL(STRING P)
BYTE I, C, NN

NN = P(0)     ;LENGTH OF STRING

;See if Baud Rate Specified
FOR C = 1 TO NN
DO
IF P(C) = '( THEN
IF SUBEQ(P,C,"(300)") = 1 THEN
BAUD = 0
ELSEIF SUBEQ(P,C,"(1200)") = 1
THEN
BAUD = 2
FI
EXIT
FI
OD

FOR C = 1 TO NN
DO
IF P(C) = '# THEN
DO
C ==+ 1
UNTIL
C > NN OR P(C) > 32
OD
EXIT
FI
```

```
OD
IF C > NN  THEN
PRINTE("No phone number in this entry!")
RETURN(0)
FI

PRINTE("Dialing...press any key to abort")
ERRORNUM = 0
STARTR()
PutMatch(13) ;Establish baud Rate
PutMatch('A)
PutMatch('T)
PutMatch(13)
GetMack() ;Swallow Reply
PutMatch('A)
PutMatch('T)
PutMatch(' )
PutMatch('D)
IF dial = 0 THEN
PutMatch('P)
ELSE
PutMatch('T)
FI
FOR I = C TO P(0)
DO
PutMatch(P(I))
OD
PutMatch(13)
DO
IF ERRORNUM >= 128
OR CH <> $FF THEN
PRINTE("User Aborted")
PUTD(2, 13) ;to get out of wait-for-carrier mode
I = RTCLOCK+10
WHILE RTCLOCK <> I DO OD ;Drain
STOPR()
RETURN(0)
FI
IF NCIB() > 0 THEN
C = GetD(2)
IF DEBUG = 1 THEN
PUT(27)
PUT(C)
FI
IF C = 'C OR C = '1 THEN ;Connected
STOPR()
RETURN(1)
ELSEIF C >= 32 THEN
PrintF("Unexpected result '%C'%E", C)
STOPR()
RETURN(0)
FI
FI
OD

;CAUSE THE SMARTMODEM TO HANG UP

PROC HANGUP()
BYTE B
```

```
STARTR()
;As per page 9-2 of the Smart-
;modem manual.  Basicly, the
;escape sequence has to be pre-
;ceded by at least one character,
;and we can't count on the user
;having typed one, so we type one
;ourselves.

PUTR('+)
WAIT(100)
PUTR('+)
PUTR('+)
PUTR('+)
WAIT(200)
;Flush buffer
WHILE NCIB() > 0 DO
B = GETD(2)
IF DEBUG = 1 THEN
PUT(27)
PUT(B)
FI
OD
ERRORNUM = 0
PutMatch(13) ;Establish baud Rate
PutMatch('A)
PutMatch('T)
PutMatch(13)
GetMack() ;Swallow Reply
PUTMATCH('A)
PUTMATCH('T)
PUTMATCH(32)
PUTMATCH('H)
PUTMATCH('0)
PUTMATCH(13)
GETMACK()
STOPR()
RETURN

; --- END OF D:KCOM850.ACT ---

;D: KERMIT.ACT

; COMPILE THIS FILE
; KERMIT protocol
; for Atari Home Computers
; version 1.2
; (C) 1984 John Howard Palevich
; to be distributed free of charge
;
;Started September 24, 1983

;Start code above T: and/or R:
;by compiling while those devices
;are in RAM.  There ought to be a
;better way!

MODULE

DEFINE MAXPACK = "94"
```

```
BYTE ARRAY
RECPKT(MAXPACK),
PACKET(MAXPACK),
FILNAM,
SBUF(2050)

DEFINE
EOF = "-1",
SOH = "1",
CR = "13",

MAXTRY = "5",
MYQUOTE = "'#",
TRUE = "1",
FALSE = "0"

BYTE
LMARGN = $52,;OS LEFT MARGIN
CH = 764,    ;OS CH VARIABLE
RTCLOCK = 20,;OS CLOCK IN JIFFYS
CRSINH = $2F0, ;OS CURSOR INHIBIT FLAG
BACKS,       ;CHAR TO SEND FOR BACK S
baud,        ;baud rate variable
dial,        ;nz for tone dialing
DISKN,       ;DEFAULT DISK
DNUM,        ;port num
localecho,   ;local echo flag
PARITY,      ;communication parity
ERRORNUM,    ;ERROR NUMBER
debug,       ;debugging flag

STATE,
PADCHAR,
EOL,
QUOTE

INT
SIZE,
N,
RPSIZ,
SPSIZ,
PAD,
TIMINT,
NUMTRY,
OLDTRY,
FD,
REMFD,
IMAGE,
HOST

INCLUDE "D:KIO.ACT"

; This is where KCOM#.ACT is
;included.  Include the KCOM file
;which matches the comunications
;device and/or modem you wish to use.
;
; For an 850 and a Hayes SmartModem,
```

```
;include KCOM850.ACT
;
; For the ATARI 1050,
;include KCOM1050.ACT
;
; For any other set of devices, write
;your own KCOM functions, and include
;that file here.

INCLUDE "D:KCOM850.ACT"

INCLUDE "D:KFUNC.ACT"
INCLUDE "D:KPRO.ACT"
INCLUDE "D:KTTY.ACT"
INCLUDE "D:KMENU.ACT"

; --- END OF D:KERMIT.ACT ---

;D:KFUNC .ACT

; Utility functions for Kermit
; (C) 1983 John Howard Palevich
; to be distributed free of charge
;
;Started September 24, 1983

MODULE

CARD ARRAY bauds = [300 600 1200
1800 2400 4800
9600]

PROC SHOWBUF(STRING BUF, INT LEN)
INT I
FOR I = 0 TO LEN-1 DO
PUT(27)
PUT(BUF(I))
OD
RETURN

PROC MERROR(BYTE A,X,Y)
IF debug = 1 THEN
PRINTF("ERROR %B%E", y)
IF Y = 128 THEN
CLOSE(2)
CLOSE(3)
CLOSE(1)
BREAK()
FI
FI
ERRORNUM = Y
RETURN

CARD FUNC DecodeBaud(BYTE b)
STRING buf(6)
STRC(bauds(b), buf)
RETURN(buf)

CARD FUNC DecodeFlag(BYTE f)
IF f = 0 THEN
```

```
RETURN("off")
ELSE
RETURN("on")
FI

BYTE FUNC IsAlpha(BYTE c)
IF (c >= 'a AND c <= 'z) OR
(c >= 'A AND c <= 'Z)
THEN
RETURN(1)
ELSE
RETURN(0)
FI

BYTE FUNC ToUpper(BYTE c)
IF c >= 'a AND c <= 'z THEN
RETURN(c - 32)
ELSE
RETURN(c)
FI

;SPack()
;
; Send a Packet

PROC SPack(BYTE TY
INT NUM, LEN
STRING DATA)
INT I, BUFP
BYTE CHKSUM
STRING BUFFER(100)

IF DEBUG = 1 THEN
PRINTF("SPack('%C,%D,%D,",
TY, NUM, LEN)
PUT('")
SHOWBUF(DATA, LEN)
PRINTF("%C)%E", '")
ELSE
PUT('.)
FI

FOR I = 1 TO PAD
DO
PUTD(2, PADCHAR)
OD

BUFFER(0) = SOH
BUFFER(1) = 32 + LEN+3
BUFFER(2) = 32 + NUM
BUFFER(3) = TY

CHKSUM = BUFFER(1)+BUFFER(2)
+BUFFER(3)

FOR I = 0 TO LEN-1
DO
BUFFER(I+4) = DATA(I)
CHKSUM ==+ DATA(I)
```

```
OD

CHKSUM = (CHKSUM + ((CHKSUM & 192)
RSH 6)) & 63
BUFFER(LEN+4) = 32 + CHKSUM
BUFFER(LEN+5) = EOL
CIOV(2, 11, BUFFER, LEN+6, -1, -1)
RETURN

;GetRT
; Get a byte from R: with timeout
; and user-abort

BYTE FUNC GetRT(BYTE POINTER B)
CHAR FSC = 19, TIMER

TIMER = FSC+3
WHILE NCIB() = 0 DO
IF FSC = TIMER THEN
IF DEBUG = 1 THEN ;say timeout
PRINTE("(Timeout)")
FI
RETURN(0)
ELSEIF CH <> $FF THEN ;User abort
RETURN(0)
FI
OD
B^ = GETD(2)
RETURN(1)


; RPack()
;
; Read a Packet

INT FUNC RPack(INT POINTER LEN, NUM
STRING DATA)
INT I, DONE
CHAR CHKSUM, T, UT, TY

IF DEBUG = 1 THEN
PRINT("RPack")
FI

DO
IF GETRT(@T) = 0 THEN
RETURN(0)
FI
IF DEBUG = 1 AND T <> SOH THEN
PUT(27)
PUT(T)
FI
UNTIL
T = SOH
OD
DONE = FALSE
WHILE DONE = FALSE
DO
IF GETRT(@T) = 0 THEN
```

```
RETURN(0)
FI
IF IMAGE = FALSE
THEN
T ==& 127
FI
IF T <> SOH THEN   ;GOT LEN
CHKSUM = T
LEN^ = T-3-32

IF GETRT(@T) = 0 THEN
RETURN(0)
FI
IF IMAGE = FALSE
THEN
T ==& 127
FI
IF T <> SOH THEN ;GOT NUM
CHKSUM ==+ T
NUM^ = T - 32

IF GETRT(@T) = 0 THEN
RETURN(0)
FI
IF IMAGE = FALSE THEN T ==& 127 FI
IF T <> SOH THEN
CHKSUM ==+ T
TY = T

FOR I = 0 TO LEN^-1 DO
IF GETRT(@T) = 0 THEN
RETURN(0)
FI
IF IMAGE = FALSE THEN T ==& 127 FI
IF T = SOH THEN EXIT FI
CHKSUM ==+ T
DATA(I) = T
OD

IF T <> SOH THEN
IF GETRT(@T) = 0 THEN
RETURN(0)
FI
IF IMAGE <> TRUE THEN T ==& 127 FI
IF T <> SOH THEN
DONE = TRUE
FI
FI
FI
FI
FI
OD
CHKSUM = (CHKSUM +
((CHKSUM & 192) RSH 6)) & 63
UT = T - 32
IF CHKSUM <> UT THEN
IF DEBUG = 1 THEN
PRINTF("(Bad checksum: %D <> %D)%E",
CHKSUM, UT)
```

```
FI
RETURN(FALSE)
FI
IF DEBUG = 1 THEN ;give type
PRINTF("('%C%C,%D,%D,%C",
27, TY, NUM^, LEN^, '")
SHOWBUF(DATA, LEN^)
PRINTF("%C)%E", '")
FI
IF TY = 'E THEN
PRINT("Error: ")
SHOWBUF(DATA, LEN^)
PUTE()
FI
RETURN(TY)

;BuFill
;
;Get a bufferful of data from the
;file that's being sent.  Only
;control-quoting is done;  8-bit &
;repeat count prefixes arn't handled

INT FUNC BuFill(STRING BUFFER)
INT I
BYTE T,T7
STOPR()
I = 0
DO
T = GETD(3)
IF MStatus(3) >= 128 THEN
IF DEBUG = 1 THEN
PRINTE("End-of-file")
FI
EXIT
FI
IF IMAGE = TRUE THEN
T7 = T & 127
IF T7 < 32 OR T7 = 127 OR
T7 = QUOTE
THEN
BUFFER(I) = QUOTE
I ==+ 1
IF T7 <> QUOTE THEN
T ==! 64
FI
FI
ELSE
IF T <> 155 THEN T ==& 127 FI
IF T < 32 OR T = 127
OR T = QUOTE OR T = 155
THEN
IF T = 155 THEN
BUFFER(I) = QUOTE
BUFFER(I+1) = 13 ! 64
I ==+ 2
T = 10
FI
BUFFER(I) = QUOTE
```

```
I ==+ 1
IF T <> QUOTE THEN T==! 64 FI
FI
FI
BUFFER(I) = T
I ==+ 1
IF I >= SPSIZ-8 THEN
STARTR()
RETURN(I)
FI
OD
STARTR()
IF I = 0
THEN
RETURN(EOF)
ELSE
RETURN(I)
FI

;BufEmp
;
;Get data from an incomming packet
;into a file.

PROC BufEmp(STRING BUFFER
INT LEN)

INT I
BYTE T

STOPR()
FOR I = 0 TO LEN-1
DO
T = BUFFER(I)
IF T = MYQUOTE
THEN
I ==+ 1
T = BUFFER(I)
IF (T & 127) <> MYQUOTE
THEN
T ==! 64
FI
FI
IF IMAGE = TRUE THEN
PUTD(3, T)
ELSEIF T = CR THEN
PUTD(3, 155)
ELSEIF T <> 10 THEN
PUTD(3, T)
FI
OD
STARTR()
RETURN

;SPar()
;
;Fill the data array with my
;send-init parameters
```

```
PROC SPar(STRING DATA)
DATA(0) = 32 + MAXPACK
DATA(1) = 32 + 5
DATA(2) = 32 + 0
DATA(3) = 64 ! 0
DATA(4) = 32 + 13
DATA(5) = MYQUOTE
RETURN

;RPar()
;
;Get the other host's send-init
;parameters

PROC RPAR(STRING DATA)
SPSIZ = DATA(0) - 32
TIMINT = DATA(1) - 32
PAD = DATA(2) - 32
PADCHAR = DATA(3) ! 64
EOL = DATA(4) - 32
QUOTE = DATA(5)
RETURN

; --- END OF D:KFUNC.ACT ---

;D:KIO .ACT

; I/O routines for kermit
; (C) 1983 John Howard Palevich

DEFINE STRING = "BYTE ARRAY"

STRING iocb
CARD filenumber

STRING dname(20), fname(20)

;WAIT T 60THS OF A SECOND

PROC WAIT(INT T)
BYTE I
WHILE T > 255
DO
I = RTCLOCK-1
WHILE I <> RTCLOCK DO OD
T ==- 255
OD
I = RTCLOCK + T
WHILE I <> RTCLOCK DO OD
RETURN

PROC STRCPY(STRING A, B)
CARD I
FOR I = 1 TO B(0) DO
A(I) = B(I)
OD
A(0) = B(0)
RETURN

BYTE FUNC MStatus(BYTE ch)
```

```
iocb = $340 + ch LSH 4
RETURN (iocb(3))

PROC CIO=$E456(BYTE a, x)

PROC CIOV(BYTE ch, cmd
CARD adr, len
INT ax1, ax2)

iocb = $340 + ch LSH 4
iocb(2) = cmd
iocb(4) = adr
iocb(5) = adr RSH 8
iocb(8) = len
iocb(9) = len RSH 8
IF ax1 >= 0 THEN
iocb(10) = ax1
FI
IF ax2 >= 0 THEN
iocb(11) = ax2
FI

CIO(0, CH * 16)
RETURN

;Do a Get Status Command
BYTE FUNC MDevStat(BYTE ch
STRING adr)
CIOV( ch, $0D,
adr + 1, adr(0), -1, -1)
RETURN(iocb(3))

; -- file locking, unlocking, etc.
; -- directory hacking functions

;Returns 0 if EOF, else the file name
CARD FUNC GetNext(CHAR ch)
INT I, J
STRING DSPEC(20)
Close(ch)
Open(ch, dname, 6, 0)
IF mstatus(ch) >= 128
THEN
RETURN(0)
FI

FOR i = 0 TO filenumber
DO
INPUTMD(ch, DSPEC, 20)
IF mstatus(ch) >= 128 THEN
Close(ch)
RETURN(0)
FI
OD
IF DSPEC(0) <> 17 THEN RETURN(0) FI
filenumber ==+ 1
Close(ch)
;Convert dspec into file name
I = 1
```

```
DO
FNAME(I) = DNAME(I)
I ==+ 1
UNTIL
DNAME(I-1) = ':
OD

J = 3
DO
FNAME(I) = DSPEC(J)
I ==+ 1
J ==+ 1
UNTIL
J > 10 OR DSPEC(J) = 32
OD
FNAME(I) = '.
I ==+ 1
J = 11
WHILE
J <= 13 AND DSPEC(J) <> 32
DO
FNAME(I) = DSPEC(J)
I ==+ 1
J ==+ 1
OD

FNAME(0) = I-1
RETURN(fname)

;Get the first name

CARD FUNC GetFirst(BYTE ch
STRING name)

STRCPY(dname, NAME)
filenumber = 0
RETURN(GetNext(ch))

;FIND CHAR C IN STRING A

BYTE FUNC FindC(STRING a
BYTE c)
CARD i,l
l = a(0)
FOR i = 1 TO l DO
IF a(i) = c THEN
EXIT
FI
OD
RETURN(i)

;Normalize a file name string to Dn:<0..8>.<0..3>
;where n is the value of diskn
;name should be at least 3+8+1+3+2=17 bytes long
;returns 0 if not a valid name

BYTE FUNC Normalize(STRING name)
CARD i, len
BYTE C
```

```
len = name(0)
IF len = 0 THEN
RETURN(0)
FI

;first, check if <letter>(<number>):

i = FindC(name,':)
IF i > len THEN
FOR i = 1 TO len DO
name(len-i+4) = name(len-i+1)
OD
name(1) = 'D
name(2) = '0 + DISKN
name(3) = ':
len ==+ 3
FI

;fixup length
name(0) = len

;and convert to upper case

FOR i = 1 TO len DO
c = name(i)
IF c >= 'a AND c <= 'z THEN
name(i) = c - 32
FI
OD

RETURN(1)

BYTE FUNC INSET(BYTE C STRING S)
CARD I
FOR I = 1 TO S(0)
DO
IF C = S(I) THEN
RETURN(I)
FI
OD
RETURN(0)

; --- END OF D:KIO.ACT

;<<<D:KMENU.ACT>>>
; Menu functions of Kermit program

MODULE
DEFINE NUMWID = "38"

STRING PNFILE = "D:KERMIT.PNS"
STRING PARAMFILE = "D:KERMIT.OPT"

;Restore Phone Number Buffer

PROC RESTNUMS()
BYTE I, J
```

```
Close(3)
ERRORNUM = 0
OPEN(3, PNFILE, 4, 0)
IF ERRORNUM < 128 THEN
FOR I = 0 TO 19 DO
ERRORNUM = 0
InputMD(3,SBUF+I*NUMWID, 37)
IF ERRORNUM >= 128 THEN
EXIT
FI
OD
ELSE
I = 0 ;Couldn't find file
FI
CLOSE(3)

FOR J = I TO 19
DO
SBUF(NUMWID*J) = 0
OD
RETURN

;Display the editor screen

PROC DispES()
BYTE I

;Display Screen
CRSINH = 1
PUT(125)

PRINTE("Computer Name (baud rate) # 555-1212")
FOR I = 0 TO 19
DO
Put(32)
PRINTE(SBUF+NUMWID*I)
OD

PrintE("Use arrows, then RETURN to dial,")
PrintE("or ESC to quit. ^S Saves")
PRINT("SPACE modifies, ^R Restores")
Position(LMARGN, 0)
Put($1F)
CRSINH = 0
Put($1E)
RETURN

;Auto-Dial a number, return 1 if
;successful, 0 if failure
;
; Also has provisions for editing
; phone numbers.

BYTE FUNC EditDial()
BYTE I, NN, C, CY
BYTE POINTER P

RESTNUMS()
```

```
DISPES()
CY = 0

;Edit/Select Loop

DO
CRSINH = 1
POSITION(LMARGN, CY+1)
PUT(27)
PUT($1F)
C = GetD(1)
IF C = 32 THEN
;User wants to change this line
POSITION(LMARGN,CY+1)
CRSINH = 0
PUT('?)
InputMD(0,SBUF+CY*NUMWID, 37)
DISPES()

ELSEIF C = 27 THEN
Position(LMARGN, 23)
CRSINH = 0
PUT($9C)
PrintE("Not Dialing")
RETURN(0)

ELSEIF (C = $1C OR C = '-)
AND CY > 0 THEN
PUT($7E) ;Erase the arrow
CY ==- 1

ELSEIF (C = $1D OR C = '=)
AND CY < 19 THEN
PUT($7E) ;Erase the arrow
CY ==+ 1

ELSEIF C = 'S-'@ THEN ;^S
OPEN(3, PNFILE, 8, 0)
FOR I = 0 TO 19 DO
P = SBUF+I*NUMWID
IF P(0) > 0 THEN
PRINTDE(3, P)
FI
OD
CLOSE(3)
RESTNUMS()
DISPES() ;Just to inform user
CY = 0

ELSEIF C = 'R-'@ THEN ;^R
RESTNUMS()
DISPES()
CY = 0

ELSEIF C = $9B THEN ;RETURN
EXIT
FI
OD
```

```
;Dial the chosen number

CRSINH = 0
PUT(125)
P = SBUF+CY*NUMWID
PrintE(P)
C = AutoDial(P)
RETURN(C)

;Execute a DOS-type command

PROC DODOS(BYTE CMD
STRING FSPEC)
STRING FMSCOM = [0 $21 $23 $24 $FE]
STRING FILNAM(21)
BYTE I, CNF

IF FSPEC(0) = 0 AND CMD <> 'A THEN
RETURN
FI

IF CMD = 'A THEN     ;DIRECTORY
IF FSPEC(0) = 0 THEN
STRCPY(FSPEC, "D#:*.*")
FSPEC(2) = '0 + DISKN
FI

NORMALIZE(FSPEC)
CLOSE(6)
ERRORNUM = 0
OPEN(6, FSPEC, 6, 0)
DO
INPUTMD(6, FILNAM, 20)
IF ERRORNUM >= 128 THEN EXIT FI
PRINTE(FILNAM)
IF FILNAM(1) >= '0 AND
FILNAM(1) <= '9
THEN EXIT FI
OD
CLOSE(6)

ELSE         ;ALL OTHER COMMANDS
NORMALIZE(FSPEC)
I = INSET(CMD, "DFGI")
IF I = 0 THEN RETURN FI
IF CMD = 'I
THEN
PRINTF("Type 'Y' to format %S%E",
FSPEC)
CNF = GetD(1)
IF TOUPPER(CNF) <> 'Y
THEN
PRINTF("Aborted%E")
RETURN
ELSE
PRINT("Formatting. . .")
FI
FI
ERRORNUM = 0
```

```
XIO(6, 0, FMSCOM(I), 0, 0, FSPEC)
IF ERRORNUM >= 128
THEN
PRINTF("Disk I/O error %B%E",
ERRORNUM)
FI
FI
RETURN

PROC MICRODOS()
BYTE cmd
STRING fspec(21)
PUT(125)
DO
PRINTE("Micro-DOS:")
PRINTE(" A - Disk Directory")
PRINTE(" D - Delete File")
PRINTE(" F - Lock File")
PRINTE(" G - Unlock File")
PRINTE(" I - Format Diskette")
PRINTE(" Q - Quit (back to main menu)")
PRINTF("%ECommand -> ")
DO
cmd = GetD(1)
cmd = ToUpper(cmd)
UNTIL
INSET(CMD, "ADFGIQ") > 0
OD

PUT(CMD)
IF cmd = 'Q
THEN
PUTE()
RETURN
FI
PRINTF("%EFile spec -> ")
InputMD(0, fspec, 20)
DoDos(cmd, fspec)
OD

; SAVE PARAMETERS

PROC SaveParams()
ERRORNUM = 0
OPEN(3, PARAMFILE, 8, 0)
IF ERRORNUM < 128
THEN         ;Can write
PUTD(3, BACKS)
PUTD(3, BAUD)
PUTD(3, DISKN)
PUTD(3, DEBUG)
PUTD(3, IMAGE)
PUTD(3, LOCALECHO)
PUTD(3, LMARGN)
PUTD(3, PARITY)
PUTD(3, DNUM)
PUTD(3, dial)
FI
CLOSE(3)
```

```
RETURN


;RESTORE PARAMETERS

PROC RestoreParams()
CARD TEMP
CLOSE(3)
ERRORNUM = 0
OPEN(3, PARAMFILE, 4, 0)
IF ERRORNUM >= 128
THEN         ;Defaults
PRINTF("Couldn't open %S; error %D%E",
PARAMFILE, ERRORNUM)
BACKS = 127          ;RUB OUT
baud = 0             ;300 baud
DISKN = 1            ;D1:
debug = 0            ;debug off
IMAGE = 0            ;TEXT
localecho = 0        ;full
LMARGN = 2           ;2 CHARS
PARITY = 0           ;NO PARITY
DNUM = 1             ;PORT 1
dial = 0             ;Pulse
ELSE
BACKS = GETD(3)
BAUD = GETD(3)
DISKN = GETD(3)
DEBUG = GETD(3)
IMAGE = GETD(3)
LOCALECHO = GETD(3)
LMARGN = GETD(3)
PARITY = GETD(3)
DNUM = GETD(3)
DIAL = GETD(3)
FI
CLOSE(3)
RETURN

;SET PARAMETERS

PROC Params()
BYTE cmd
STRING ts

DO
Put(125)
PRINTE("Parameters are:")

IF BACKS = 8 THEN
TS = "control-H"
ELSE TS = "rub out"
FI
PRINTF(" A - Back S sends (%S)%E",
ts)

ts = DecodeBaud(baud)
PRINTF(" B - Baud rate (%S)%E",
TS)
```

```
IF IMAGE = 0 THEN
ts = "text"
ElSE
ts = "binary"
FI

PRINTF(" D - Default disk drive (D%D:)%E",
diskn)

PRINTF(" F - File type (%S)%E",
ts)

PRINTF(" I - I/O Port (%D)%E",
DNUM)

IF dial = 0 THEN
ts = "pulse"
ELSE
ts = "tone"
FI
PRINTF(" T - Dialing method (%S)%E",
ts)

ts = DecodeFlag(localecho)
PRINTF(" L - Local-Echo (%S)%E",
ts)

PRINTF(" M - Margin (%D)%E", LMARGN)


IF PARITY = 0 THEN
TS = "none"
ELSEIF PARITY = 1 THEN
TS = "odd"
ELSEIF PARITY = 2 THEN
TS = "even"
ELSEIF PARITY = 3 THEN
TS = "on"
FI
PRINTF(" P - Parity (%S)%E", ts)

PRINTE("^S - Save parameters")
PRINTE("^R - Restore paramters")

ts = DecodeFlag(debug)
PRINTF(" * - Debug Mode (%S)%E",
ts)

PRINTF(" Q - Quit (back to Commands)%E")

PRINTF("Parameter to change -> ")
cmd = GetD(1)
cmd = ToUpper(cmd)
IF IsAlpha(cmd) <> 0 THEN
Put(cmd)
FI

IF CMD = 'A THEN     ;BACK S
```

```
IF BACKS = 8 THEN
BACKS = 127
ELSE
BACKS = 8
FI

ELSEIF cmd = 'B THEN ;Baud-rate
baud ==+ 1
IF baud > 6 THEN baud = 0 FI

ELSEIF cmd = 'D THEN ;Disk number
diskn ==+ 1
IF diskn > 4 THEN diskn = 1 FI

ELSEIF cmd = '* THEN ;Debug
debug = 1-debug

ELSEIF cmd = 'Q THEN ;Quit
PRINTF("uit%E")
RETURN

ELSEIF cmd = 'F THEN ;File type
IMAGE = 1-IMAGE

ELSEIF cmd = 'L THEN ;local-echo
localecho ==+ 1
IF localecho > 1 THEN
LOCALECHO = 0
FI

ELSEIF cmd = 'T THEN ;dialing
DIAL ==+ 1
IF DIAL > 1 THEN
DIAL = 0
FI

ELSEIF CMD = 'M THEN ;Margin
LMARGN ==+ 1
IF LMARGN > 2 THEN
LMARGN = 0
FI

ELSEIF CMD = 'P THEN ;PARITY
PARITY ==+ 1
IF PARITY > 3 THEN
PARITY = 0
FI

ELSEIF cmd = 'I THEN ;Port #
dnum ==+ 1
IF dnum > 4 THEN dnum = 1 FI

ELSEIF cmd = 'S-'@ THEN ;Save Parameters
PRINTE("Saving")
SAVEPARAMS()

ELSEIF cmd = 'R-'@ THEN ;Restore parameters
PRINTE("Restoring")
RESTOREPARAMS()
```

```
ELSE
PUT(253)
FI
OD

PROC Main()
BYTE cmd, FLAG, I, BANK = $D500

BANK = 0

;SETUP MY ERROR ROUTINE
ERROR = MERROR

EOL = CR
QUOTE = MYQUOTE
PAD = 0
PADCHAR = 0
HOST = FALSE

FOR I = 1 TO 7 DO
CLOSE(I)
OD

PRINTE("Kermit for the Atari Home Computer")
PRINTE("v1.2 (c) 1984 John Howard Palevich")
MODEMINIT()
PRINTE("- Feel free to copy this program -")

RestoreParams()
Open(1, "K:", 4, 0)
IF OPENR() <> 0 THEN
PRINTE("PRESS ANY KEY TO EXIT")
CH = $FF
WHILE CH = $FF DO OD
CH = $FF
ELSE
STOPR()

DO
PRINTF("%E%ECommands are:%E")
PRINTE(" A - Auto-dial (then connect)")
PRINTE(" C - Connect (to remote computer)")
PRINTE(" D - Micro-DOS")
PRINTE(" F - Finish (remote server mode)")
PRINTE(" H - Hang up (the phone)")
PRINTE(" P - Parameters (inspect and change)")
PRINTE(" R - Receive (a file)")
PRINTE(" S - Send (a file)")
PRINTF(" Q - Quit (back to DOS)%E%E")
PRINTF("Command -> ")
DO
cmd = GetD(1)
cmd = ToUpper(cmd)
UNTIL INSET(CMD, "ACDFHPRSQ") <> 0
OD
Put(cmd)

IF CMD = 'A THEN
```

```
;Auto-dial
PRINTE("uto-dial")
IF EditDial() = 1 THEN
TTYMODE()
FI

ELSEIF cmd = 'C THEN ;connect
PRINTE("onnect")
TTYMODE()

ELSEIF cmd = 'F THEN ;Finish
PRINTE("inish")
Finish()

ELSEIF cmd = 'H THEN
;Hang up the phone
PRINTE("ang up")
HangUp()

ELSEIF cmd = 'D THEN ;MICRO-DOS
PRINTE("os")
MICRODOS()

ELSEIF cmd = 'Q THEN ;Quit
PRINTE("uit")
EXIT

ELSEIF cmd = 'P THEN ;Parameters
PRINTE("arameters")
Params()

ELSEIF cmd = 'S THEN ;Send
PRINTE("end")
SENDSW()

ELSEIF cmd = 'R THEN ;Recieve
PRINTE("ecieve")
RECSW()
FI
OD

CLOSER()
FI
CLOSE(1)
RETURN

;--- END OF D:KMENU.ACT ---

;D:KPRO .ACT

; KERMIT protocol section

; RInit()
;
; Receive Initialization

BYTE FUNC RINIT(STRING FSPEC)
INT LEN, NUM, T
IF DEBUG = 1 THEN
PRINTE("RInit")
```

```
FI

NUMTRY ==+ 1
IF NUMTRY > MAXTRY THEN
RETURN('A)
FI

IF FSPEC(0) > 0 THEN
FOR T = 1 TO FSPEC(0)
DO
PACKET(T-1) = FSPEC(T)
OD
SPACK('R, 0, T-1, PACKET)
FI

T = RPACK(@LEN, @NUM, PACKET)
IF T = 'S THEN
RPAR(PACKET)
SPAR(PACKET)
SPACK('Y, N, 6, PACKET)
OLDTRY = NUMTRY
NUMTRY = 0
N = (N + 1) MOD 64
RETURN('F)

ELSEIF T = FALSE THEN RETURN(STATE)
ELSE RETURN('A)
FI

; RFile()
;
; Receive File Header

BYTE FUNC RFile()
INT LEN, NUM, T
BYTE W
IF DEBUG = 1 THEN
PRINTF("RFile%E")
FI

NUMTRY ==+ 1
IF NUMTRY > MAXTRY THEN
RETURN('A)
FI

T = RPACK(@LEN, @NUM, PACKET+1)
PACKET(0) = LEN
IF T = 'S THEN
OLDTRY ==+ 1
IF OLDTRY > MAXTRY THEN RETURN('A) FI
IF (N = 0 AND NUM = 63)
OR (N <> 0 AND NUM = N-1)
THEN
SPACK('Y, NUM, 0, 0)
NUMTRY = 0
RETURN(STATE)
ELSE
RETURN('A)
FI
```

```
ELSEIF T = 'F THEN
IF NUM <> N THEN RETURN('A) FI
STOPR()
NORMALIZE(PACKET)
ERRORNUM = 0
OPEN(3, PACKET, 8, 0)
STARTR()
IF ERRORNUM >= 128
THEN
PRINTF("Couldn't create %S; error %D%E",
PACKET, ERRORNUM)
RETURN('A)
FI
PRINTF("Receiving %S%E",
PACKET)
SPACK('Y, N, 0, 0)
OLDTRY = NUMTRY
NUMTRY = 0
N = (N+1) MOD 64
RETURN('D)

ELSEIF T = 'B THEN
IF NUM <> N THEN RETURN('A) FI
SPACK('Y, N, 0, 0)
;WAIT 1 SECOND FOR ACK TO DRAIN
W = RTCLOCK+60
WHILE W <> RTCLOCK DO OD
RETURN('C)

ELSEIF T = FALSE THEN RETURN(STATE)
ELSE RETURN('A)
FI

; RData()
;
; Receive Data

BYTE FUNC RData()
INT NUM, LEN, T

IF DEBUG = 1 THEN
PRINTF("RData%E")
FI

NUMTRY ==+ 1
IF NUMTRY > MAXTRY THEN
RETURN('A)
FI

T = RPACK(@LEN, @NUM, PACKET)
IF T = 'D THEN
IF NUM <> N
THEN
OLDTRY ==+ 1
IF OLDTRY > MAXTRY THEN RETURN('A) FI
IF (N = 0 AND NUM = 63)
OR (N <> 0 AND NUM = N-1)
THEN
```

```
SPACK('Y, NUM, 0, 0)
NUMTRY = 0
RETURN(STATE)
ELSE
RETURN('A)
FI
FI

BUFEMP(PACKET, LEN)
SPACK('Y, N, 0, 0)
OLDTRY = NUMTRY
NUMTRY = 0
N = (N+1) MOD 64
RETURN('D)

ELSEIF T = 'F THEN
OLDTRY ==+ 1
IF OLDTRY > MAXTRY THEN
RETURN('A)
FI
IF (N = 0 AND NUM = 63)
OR (N <> 0 AND NUM = N-1)
THEN
SPACK('Y, NUM, 0, 0)
NUMTRY = 0
RETURN(STATE)
ELSE
RETURN('A)
FI

ELSEIF T = 'Z THEN
IF NUM <> N THEN RETURN('A) FI
IF DEBUG = 1 THEN
PRINTE("End-of-File")
FI
STOPR()
CLOSE(3)
STARTR()
SPACK('Y, N, 0, 0)
N = (N+1) MOD 64
RETURN('F)

ELSEIF T = FALSE THEN RETURN(STATE)
ELSE RETURN('A)
FI

; RecSw()
;
; This is the state table switcher
; for receiving files

PROC RECSW()
STRING FSPEC(20)
INT NUM, LEN, T

STARTR()
PUT(125)
PRINTE("Type the file to receive, or just")
PRINTE("RETURN if the other computer is not")
```

```
PRINTE("in Server mode.")
PUTE()
PRINT("File Spec -> ")
INPUTMD(0, FSPEC, 19)

PRINTE("Receiving File(s)")
PRINTE("type any key to abort")

STATE = 'R
N = 0
NUMTRY = 0
DO
IF CH <> 255 THEN
PRINTE("User Aborting")
CH = 255
EXIT
FI
IF STATE = 'D THEN STATE = RDATA()
ELSEIF STATE = 'F THEN STATE = RFILE()
ELSEIF STATE = 'R THEN STATE = RINIT(FSPEC)
ELSEIF STATE = 'A THEN
PRINTE("Aborting")
EXIT
ELSE
EXIT
FI
OD
STOPR()
Close(3)
RETURN

; SInit
;
; Send Initiate:
;  Send my parameters, get other
;  side's back

BYTE FUNC SINIT()
INT NUM, LEN
BYTE T

IF DEBUG <> 0 THEN
PRINTF("SInit%E")
FI

NUMTRY ==+ 1
IF NUMTRY > MAXTRY THEN
RETURN('A)
FI
SPAR(PACKET)
IF DEBUG <> 0 THEN
PRINTF("n = %D%E", N)
FI
;Clear out any junk in the input
;buffer
WHILE NCIB() > 0 DO GETD(2) OD

SPACK('S, N, 6, PACKET)
T = RPACK(@LEN, @NUM, RECPKT)
```

```
IF     T = 'N THEN RETURN(STATE)
ELSEIF T = 'Y THEN
IF N <> NUM THEN
RETURN(STATE)
FI
RPAR(RECPKT)
IF EOL = 0 THEN
EOL = 13
FI
IF QUOTE = 0 THEN
QUOTE = '#
FI
NUMTRY = 0
N = (N + 1) MOD 64
IF FILNAM = 0 THEN
RETURN('A)
FI
;Open a file
STOPR()
ERRORNUM = 0
Close(3)
OPEN(3, FILNAM, 4, 0)
STARTR()
IF ERRORNUM >= 128 THEN
PRINTF("Error %D; couldn't read %S",
ERRORNUM, FILNAM)
RETURN('A)
FI
PRINTF("Sending %S%E", FILNAM)
RETURN('F)

ELSEIF T = FALSE THEN RETURN(STATE)
ELSE RETURN('A)
FI

; SFile
;
; Send File Header


BYTE FUNC SFILE()
INT NUM, LEN, T, I
STRING STFNAME(20)
IF DEBUG = 1 THEN
PRINTE("SFile")
FI

NUMTRY ==+ 1
IF NUMTRY > MAXTRY THEN RETURN('A) FI

I = 1        ;STANDARD FILE NAMES DON'T HAVE D1:
WHILE FILNAM(I) <> ': DO I ==+ 1 OD
LEN = FILNAM(0)-I
FOR T = 0 TO LEN-1 DO
STFNAME(T) = FILNAM(I+T+1)
OD

SPACK('F, N, LEN, STFNAME)
T = RPACK(@LEN, @NUM, RECPKT)
```

```
IF T = 'N OR T = 'Y THEN
IF T = 'N
THEN
NUM ==- 1
IF NUM < 0 THEN NUM = 63 FI
FI

IF N <> NUM
THEN
RETURN(STATE)
FI
NUMTRY = 0
N = (N + 1) MOD 64
SIZE = BUFILL(PACKET)
IF SIZE = EOF THEN
RETURN('Z)
ELSE
RETURN('D)
FI
ELSEIF T = FALSE THEN RETURN(STATE)
ELSE RETURN('A)
FI

; SData
;
; Send File Data

BYTE FUNC SData()
INT NUM, LEN, T

NUMTRY ==+ 1
IF NUMTRY > MAXTRY THEN
RETURN('A)
FI
SPACK('D, N, SIZE, PACKET)
T = RPACK(@LEN, @NUM, RECPKT)
IF T = 'N OR T = 'Y THEN
IF T = 'N
THEN
NUM ==- 1
IF NUM < 0 THEN NUM = 63 FI
FI

IF N <> NUM
THEN
RETURN(STATE)
FI
NUMTRY = 0
N = (N + 1) MOD 64
SIZE = BUFILL(PACKET)
IF SIZE = EOF THEN
RETURN('Z)
FI
RETURN('D)
ELSEIF T = FALSE THEN
RETURN(STATE)
ELSE RETURN('A)
FI
```

```
; SEOF()
;
; Send End-Of-File

BYTE FUNC SEOF()
INT NUM, LEN, T

IF DEBUG = 1 THEN
PRINTF("SEOF%E")
FI

NUMTRY ==+ 1
IF NUMTRY > MAXTRY THEN
RETURN('A)
FI
SPACK('Z, N, 0, PACKET)

IF DEBUG = 1 THEN
PRINT("SEOF1 ")
FI

T = RPACK(@LEN, @NUM, RECPKT)
IF T = 'N OR T = 'Y THEN
IF T = 'N
THEN
NUM ==- 1
IF NUM < 0 THEN NUM = 63 FI
IF N <> NUM THEN RETURN(STATE) FI
FI

IF DEBUG = 1 THEN
PRINTF("SEOF2 ")
FI
IF N <> NUM
THEN
RETURN(STATE)
FI
NUMTRY = 0
N = (N + 1) MOD 64
IF DEBUG = 1 THEN
PRINTF("Closing %S%E", FILNAM)
FI
STOPR()
IF DEBUG = 1 THEN
PRINTF("getting next file%E")
FI
DO
FILNAM = GETNEXT(6)
IF FILNAM = 0 THEN EXIT FI
CLOSE(3)
ERRORNUM = 0
OPEN(3,FILNAM, 4, 0)
IF ERRORNUM < 128 THEN
EXIT
ELSE
PRINTF("Can't read %S; Error %D%E",
FILNAM, ERRORNUM)
FI
OD
```

```
STARTR()
IF FILNAM = 0 THEN
RETURN('B)
FI
PRINTE(FILNAM)
RETURN('F)
ELSEIF T = FALSE THEN RETURN(STATE)
ELSE RETURN('A)
FI

; SBreak()
;
; Send Break (End-of-Text)

BYTE FUNC SBreak()
INT NUM, LEN, T

IF DEBUG = 1 THEN
PRINTF("SBreak%E")
FI

NUMTRY ==+ 1
IF NUMTRY > MAXTRY THEN
RETURN('A)
FI
SPACK('B, N, 0, PACKET)

T = RPACK(@LEN, @NUM, RECPKT)
IF T = 'N OR T = 'Y THEN
IF T = 'N
THEN

NUM ==- 1
IF NUM < 0 THEN NUM = 63 FI
IF N <> NUM THEN
RETURN(STATE)
FI
FI

IF N <> NUM
THEN
RETURN(STATE)
FI

NUMTRY = 0
N = (N + 1) MOD 64
RETURN('C)

ELSEIF T = FALSE THEN RETURN(STATE)
ELSE RETURN('A)
FI

;MAIN SEND FILE ROUTINE

PROC SENDSW()
STRING FSpec(20)
DO
Print("File spec -> ")
```

```
INPUTMD(0, FSPEC, 19)
IF FSPEC(0) = 0 THEN RETURN FI
Normalize(FSPEC)
FILNAM = GETFIRST(6, FSPEC)
IF FILNAM = 0 THEN
PRINTE("Invalid file name")
FI
UNTIL
FILNAM <> 0
OD
Put(125)
PRINTF("Sending %S%E", FSpec)
PRINTE("Type any key to abort.")
STARTR()

STATE = 'S
N = 0
NUMTRY = 0
DO
IF CH <> 255 THEN
PRINTE("User Abort")
CH = 255
EXIT
FI
IF     STATE = 'D THEN STATE = SDATA()
ELSEIF STATE = 'F THEN STATE = SFILE()
ELSEIF STATE = 'Z THEN STATE = SEOF()
ELSEIF STATE = 'S THEN STATE = SINIT()
ELSEIF STATE = 'B THEN STATE = SBREAK()
ELSEIF STATE = 'A THEN
PRINTE("Aborting")
EXIT
ELSE EXIT
FI
OD
STOPR()
CLOSE(3)
RETURN

;Tell Server to quit

PROC Finish()
INT NUM, LEN, T

IF DEBUG = 1 THEN
PRINTE("Finish")
FI
STARTR()
FOR NUMTRY = 0 TO 3
DO
PACKET(0) = 'F
SPACK('G, 0, 1, PACKET)

T = RPACK(@LEN, @NUM, RECPKT)
IF T = 'N OR T = 'Y THEN
IF T = 'N
THEN
NUM ==- 1
IF NUM < 0 THEN NUM = 63 FI
```

```
IF 0 <> NUM THEN
EXIT
FI
FI

IF 0 = NUM
THEN
STOPR()
RETURN
FI
FI
OD

STOPR()
PRINTE("Server didn't respond")
RETURN

;--------------------------
;Kermit Protocol code ends here
;--------------------------

; --- END OF D:KPRO.ACT ---

;D:KTTY. ACT
; Terminal emulation for the masses
; Emulates a VT-52, Option quits,
; Start scrolls.


MODULE
CARD ARRAY LBASE(24)
BYTE ARRAY LCUR(24)

BYTE CX, CY, LMAR, DLTOGGLE,TSTATE,
consol = $D01F
CARD SDLST = $230,
SAVEDL, HELPLINE

;Create a display list and display it
;
; Uses: LBASE, LCUR, LMAR, SAVEDL,
; Modifies: DLTOGGLE, SCREEN MEMORY

PROC HACKDISPLAY()
BYTE ARRAY DBASE
BYTE I
CARD J, TBASE
DBASE = DLTOGGLE*85+SAVEDL+72
DLTOGGLE = 1 - DLTOGGLE
TBASE = DBASE
FOR I = 0 TO 2 DO
DBASE(I) = $70
OD
FOR I = 0 TO 23 DO
DBASE ==+ 3
DBASE(0) = $42
J = LCUR(I)
J = LBASE(J) + LMAR - LMARGN
DBASE(1) = J
DBASE(2) = J RSH 8
```

```
OD
DBASE(3) = $00
DBASE(4) = $42
DBASE(5) = HELPLINE
DBASE(6) = HELPLINE RSH 8
DBASE(7) = $41
DBASE(8) = TBASE
DBASE(9) = TBASE RSH 8
SDLST = TBASE
RETURN

PROC CFLIP()
BYTE POINTER M
BYTE I
I = LCUR(CY)
M = LBASE(I) + CX
M^ ==! $80
RETURN

PROC LCLEAR(BYTE LINE)
BYTE I
BYTE ARRAY T
I = LCUR(LINE)
T = LBASE(I)-2
FOR I = 0 TO 81 DO
T(I) = 0
OD
RETURN

PROC TINIT()
CARD I, J
;First, find 24 valid lines in
;Sbuf.  Valid lines don't cross 4K
J = SBUF
FOR I = 0 TO 23
DO
IF (J RSH 12) <>
((J + 81) RSH 12)
THEN
J = (J & $F000) + $1000
FI
LBASE(I) = J+2
J ==+ 82
LCUR(I) = I ;set up current line order
LCLEAR(I)
OD
;Now set up a display list
SAVEDL = SDLST
HELPLINE = SDLST+32
PUT(125)
PRINTE("OPTION quits, (SHIFT)+START scrolls")
DLTOGGLE = 0
TSTATE = 'N
CX = 0
CY = 0
LMAR = 0
CFLIP()
HACKDISPLAY()
RETURN
```

```
BYTE FUNC TPUTN(BYTE C)
BYTE I, TEMP
BYTE POINTER M
BYTE ARRAY TOSCR = [$40 $00 $20 $60]
CFLIP()
IF C < 32 THEN
IF C = 27 THEN
RETURN('E)
ELSEIF C = 10 THEN
IF CY < 23 THEN
CY ==+ 1
ELSE
LCLEAR(0)
TEMP = LCUR(0)
FOR I = 0 TO 22
DO
LCUR(I) = LCUR(I+1)
OD
LCUR(23) = TEMP
HACKDISPLAY()
FI

ELSEIF C = 13 THEN
CX = 0

ELSEIF C = 7 THEN ;BELL
SETCOLOR(4, 0, 14)
I = RTCLOCK + 2
WHILE I <> RTCLOCK DO OD
SETCOLOR(4, 0, 0)

ELSEIF C = 8 THEN ;BACKSPACE
IF CX > 0 THEN
CX ==- 1
FI

ELSEIF C = 9 THEN ;TAB
IF CX < 72 THEN
CX = (CX + 8) & $F8
FI

ELSEIF C = 12 THEN
FOR I = 0 TO 23 DO
LCLEAR(I)
OD
CX = 0
CY = 0

FI
ELSE         ;printing char
I = LCUR(CY)
M = LBASE(I) + CX
M^ = TOSCR((C & $60) RSH 5)
% (C & $9F)
IF CX < 79 THEN CX ==+ 1
FI
FI
CFLIP()
```

```
RETURN('N)

BYTE FUNC TPUTE(BYTE C)
BYTE TEMP, I
BYTE ARRAY M
IF C = 'A THEN
IF CY > 0 THEN
CY ==- 1
FI

ELSEIF C = 'B THEN
IF CY < 23 THEN
CY ==+ 1
FI

ELSEIF C = 'C THEN
IF CX < 79 THEN
CX ==+ 1
FI

ELSEIF C = 'D THEN
IF CX > 0 THEN
CX ==- 1
FI

ELSEIF C = 'H THEN
CX = 0
CY = 0

ELSEIF C = 'I THEN
IF CY > 0 THEN
CY ==- 1
ELSE
LCLEAR(23)
TEMP = LCUR(23)
FOR I = 0 TO 22 DO
LCUR(23-I) = LCUR(22-I)
OD
LCUR(0) = TEMP
HACKDISPLAY()
FI

ELSEIF C = 'J OR C = 'K THEN
I = LCUR(CY)
M = LBASE(I)
FOR I = CX TO 79 DO
M(I) = 0
OD
IF C = 'J THEN
FOR I = CY+1 TO 23 DO
LCLEAR(I)
OD
FI
ELSEIF C = 'Y THEN
RETURN('R)
ELSEIF C = 'Z THEN
PUTD(2, 27)
PUTD(2, '/)
PUTD(2, 'Z)
```

```
FI
CFLIP()
RETURN('N)

PROC TPUTSW(BYTE C)
IF TSTATE = 'N THEN
TSTATE = TPUTN(C)
ELSEIF TSTATE = 'E THEN
TSTATE = TPUTE(C)
ELSEIF TSTATE = 'R THEN
IF C < 32 THEN C = 32 FI
CY = C - 32
IF CY > 23 THEN CY = 23 FI
TSTATE = 'C
ELSEIF TSTATE = 'C THEN
IF C < 32 THEN C = 32 FI
CX = C - 32
IF CX > 79 THEN CX = 79 FI
CFLIP()
TSTATE = 'N
ELSE
TSTATE = 'N
FI
RETURN

PROC TQUIT()
SDLST = SAVEDL
PUT(125)
RETURN

PROC TTYMode()
BYTE c, SKSTAT = $D20F, OLDSCROLL

StartR()

TINIT()
OLDSCROLL = RTCLOCK - 1
DO
IF ch <> $FF THEN
c = GetD(1)
IF c = 155 THEN c = 13
ELSEIF c = 127 THEN c = 9
ELSEIF c = $7E THEN c = backs
FI
PutD(2, c)
IF localecho = 1 THEN
TPUTSW(c)
FI
FI

IF ncib() > 0 THEN
c = GetD(2) & $7F ;strip parity
TPUTSW(c)
FI

consol = 8
IF (consol & 4) = 0 THEN
EXIT
```

```
ELSEIF (CONSOL & 1) = 0
AND RTCLOCK <> OLDSCROLL THEN
;START - SHIFT LEFT & RIGHT
IF (SKSTAT & 8) = 0 THEN
IF LMAR > 0 THEN
LMAR ==- 1
FI
ELSE
IF LMAR < 40+LMARGN THEN
LMAR ==+ 1
FI
FI
HACKDISPLAY()
OLDSCROLL = RTCLOCK
FI
OD
TQUIT()
StopR()
RETURN

;End of D:KTTY.ACT
```