Behavior Research Methods, Instruments, & Computers
1985, Vol. 17 (5), 556-564

A split-screen electronic messaging system for Apple II computers

DENIZ ERGENER and A. RODNEY WELLENS
University of Miami, Coral Gables, Florida

A method is described for configuring an Apple II+ or Apple IIe computer to create a multiuser, multiwindowed, electronic messaging system for use in computer-mediated communication research.

With the increased availability of local and long-distance computer networks, there has been an increasing amount of interest regarding the social psychological aspects of computer-mediated communication (Chapanis, 1972; Johansen, 1984; Johansen, Vallee, & Spangler, 1979; Short, Williams, & Christie, 1976). The effects of interactive text-based messaging upon group decision making, productivity, participation rates, leadership, coalition formation, and other social behaviors have been studied using a variety of multiuser systems. This paper describes a method by which a relatively inexpensive single-user microcomputer (an Apple II+ or Apple IIe) can be configured to emulate a multiwindowed, multiuser messaging system typically restricted to more expensive computer systems.

Kiesler, Siegel, and McGuire (1984) and Murrel (1983) described computer conferencing experiments that utilize multiuser systems whose software divides each user's terminal screen into several parts and allows messages from different people to appear simultaneously and scroll independently. The ability of users to monitor messages as they are being entered into the system (rather than waiting for completed messages to be delivered to the screen) allows participants to interrupt each other and may bring about a greater feeling of social "immediacy." Our goal was to create a similar system for studying two-person interactions using relatively inexpensive equipment immediately available to many experimenters. The software can be used either to support a stand-alone messaging system for computer-mediated communication research or to act as the front end of a more elaborate on-line, text-based data acquisition and analysis sytem. It is one of several subsystems recently added to a multimedia laboratory for social interaction research (Wellens, 1979).

HARDWARE COMPONENTS

The system consists of an Apple IIe or Apple II+ computer equipped with two Apple II Super Serial cards, one Videx Videoterm 80-column display card (replacing the standard 80-column card within the Apple IIe), four monochrome television monitors and two remote keyboard terminals (e.g., Synertek KTM-3 terminals) capable of serial communications. Three of the television monitors are connected in parallel to the video output pin of the Videx 80-column video display card placed in Slot 3 of the Apple. The Videx card is used to store character information and provide the split-screen video display viewed by users. Each keyboard is connected via an RS232 cable to one of the serial cards placed in Slots 2 and 4 of the Apple. One remote monitor and keyboard is placed before each subject, and the third monitor is used by the experimenter to follow the text exchanged between subjects. The fourth monitor is connected to the Apple's normal 40-column video output connector and can be used to monitor other programs that can be run simultaneously on the Apple.

This project was supported in part by a University of Miami Research and Sponsored Programs Summer Award in Business and the Social Sciences awarded to the second author. The author's mailing address is: Department of Psychology, University of Miami, P. O. Box 248185, Coral Gables, FL 33124.

SOFTWARE DESCRIPTION

An assembly language program was written that treats the Videx 80-column card as an external device for displaying text received from either of the keyboard terminals connected to the two serial ports. When an entry is made on either keyboard, the serial card to which it is connected generates an interrupt signal within the Apple. The assembly program detects the interrupt, polls the status register of each serial port to decide which card sent the interrupt, and then receives the character sent from the card's data buffer. The program then branches to a subroutine that POKEs the character in either the top or bottom half of the 2K video display buffer of the Videx card. The program also POKEs a flag to a memory location within the Apple, designating which port received the message, and POKEs the character sent to an adjacent memory location. This information can be used by other programs which can be run simultaneously to calculate response latencies, compute message lengths, or perform other real-time tasks.

The 80-column screen generated by the Videx card is divided in half by a horizontal line (created by using a graphics character available within the Videx character set) that remains at center screen. Up to 11 lines of text

may be displayed both above and below the center line. All text received from the keyboard connected to Slot 2 is displayed above the center line (upper window), and all text received from the keyboard connected to Slot 4 is displayed below the center line (lower window). Both windows have automatic wrap-around and line-feed capabilities. When messages exceed 11 lines, each window is capable of scrolling independently; the oldest line scrolls off the top of the window with new lines added to the bottom of the window. Each window has its own cursor that can be backspaced for making corrections by using either the "backspace" or "delete" key on the remote keyboard terminals. Line feeds are automatically generated whenever a "return" or "enter" key is struck.

A short BASIC program boots the system, sets the communication parameters on the serial cards (i.e., adjusts baud rate, word length, stop bits, and parity to be compatible with the remote terminals being used), loads and then runs the assembly language program. The BASIC program erases itself after performing these functions. The assembly program that is left running takes up less than 1K of memory space within the Apple and resides within memory locations reserved for the Apple's second high resolution graphics screen. It is written as a "background" program that remains essentially invisible to other programs that may be subsequently loaded and run simultaneously.

PROGRAM AVAILABILITY

An annotated source code listing of the assembly language program, the auto-boot BASIC program, and a short BASIC demonstration program is provided in the Appendix.

REFERENCES

CHAPANIS, A. (1972). Studies of interactive communication: The effects of four communication modes on the behavior of teams during cooperative problem-solving. *Human Factors*, **14**, 487-509.

JOHANSEN, R. (1984). *Teleconferencing and beyond: Communications in the office of the future*. New York: McGraw-Hill.

JOHANSEN, R., VALLEE, J., & SPANGLER, K. (1979). *Electronic meetings: Technical alternatives and social choices*. Reading, MA: Addison-Wesley.

KIESLER, S., SIEGEL, J., & MCGUIRE, T. (1984). Social psychological aspects of computer-mediated communication. *American Psychologist*, **39**, 1123-1134.

MURREL, S. (1983, October). Computer communication system design affects group communication. In A. R. Wellens (Chair), *Microcomputers in social interaction research*. Symposium conducted at the Society of Southeastern Social Psychologists, Winston-Salem, NC.

SHORT, J., WILLIAMS, E., & CHRISTIE, B. (1976). *The social psychology of telecommunications*. New York: Wiley & Sons.

WELLENS, A. R. (1979). An interactive television laboratory for the study of social interaction. *Journal of Nonverbal Behavior*, **4**, 119-122.

Appendix

THE FOLLOWING IS AN ASSEMBLY LANGUAGE PROGRAM THAT TREATS A VIDEX 80-COLUMN CARD AS AN EXTERNAL DEVICE FOR DISPLAYING TEXT RECEIVED FROM TWO KEYBOARD TERMINALS CONNECTED TO AN APPLE IIE OR II+ COMPUTER VIA TWO SUPER SERIAL CARDS. THE PROGRAM DISPLAYS TEXT RECEIVED FROM KEYBOARD ONE WITHIN THE TOP HALF OF THE 80-COLUMN SCREEN AND DISPLAYS TEXT RECEIVED FROM KEYBOARD TWO WITHIN THE BOTTOM HALF OF THE SCREEN. EACH HALF OF THE SCREEN SCROLLS INDEPENDENTLY. OTHER PROGRAMS MAY BE RUN SIMULTANEOUSLY WITH THIS PROGRAM TO MONITOR MESSAGING BEHAVIOR. THESE PROGRAMS CAN PEEK AT LOCATION 17017 TO DETERMINE WHICH KEYBOARD WAS THE LAST TO BE ACTIVE (1 OR 2) AND PEEK TO LOCATION 17018 TO DETERMINE THE LAST CHARACTER RECEIVED (ASCII 0-255).

SOURCE FILE: COMM IIE

```
C08B:                1 CONTROL    EQU $C08B
C08A:                2 COMMAND    EQU $C08A
C089:                3 STATUS     EQU $C089
C088:                4 DATA       EQU $C088
03FE:                5 IRQ1       EQU $3FE
03FF:                6 IRQ2       EQU $3FF
0020:                7 SLOT       EQU $20
0040:                8 SLOT2      EQU $40
0045:                9 ACC        EQU $45
0007:               10 CUR        EQU $07
0002:               11 SEPR       EQU $02
0020:               12 SP         EQU $20
04FB:               13 BASEH      EQU $4FB
057B:               14 HOR        EQU $57B
05FB                15 VER        EQU $5FB
0678:               16 BYTE       EQU $678
06F8:               17 NO         EQU $6F8
07F8:               18 MSLOT      EQU $7F8
```

NEXT OBJECT FILE NAME IS COMM IIE.OBJ0

```
4000:               19    ORG $4000
4000:               20    ********************************
4000:               21    * INITIALIZATION ROUTINE *
4000:               22    ********************************
```

```
4000:A9 36      23    LDA #54              SET CONTROL REGS OF
4002:8D AB C0   24    STA CONTROL+SLOT     SERIAL PORTS
4005:8D CB C0   25    STA CONTROL+SLOT2
4008:A9 69      26    LDA #105             SET COMMAND REGS OF
400A:8D AA C0   27    STA COMMAND+SLOT     SERIAL PORTS
400D:8D CA C0   28    STA COMMAND+SLOT2
4010:A9 4E      29    LDA #>START          LOAD ADDRESS OF INTERRUPT
4012:8D FE 03   30    STA IRQ1             SERVICE ROUTINE TO MASKABLE
4015:A9 40      31    LDA #<START          INTERRUPT VECTOR LOCATION
4017:8D FF 03   32    STA IRQ2
401A:58         33    CLI
401B:20 7B 40   34    JSR SETREGS          SET REGS OF 80-COL CARD
401E:20 09 C8   35    JSR $C809            INITIALIZE 80-COL CARD
4021:A9 0A      36    LDA #$0A
4023:8D B0 C0   37    STA $C0B0
4026:A9 20      38    LDA #$20
4028:8D B1 C0   39    STA $C0B1
402B:A9 07      40    LDA #CUR             LOAD CURSOR TO LOCATION
402D:AE 4D 41   41    LDX HOR1             SET BY HORIZONTAL & VERTICAL
4030:AC 4E 41   42    LDY VER1             POINTERS
4033:20 12 42   43    JSR OUTPUT           BRANCH TO OUTPUT ROUTINE
4036:AE 10 42   44    LDX HOR2
4039:AC 11 42   45    LDY VER2
403C:20 12 42   46    JSR OUTPUT
403F:A2 00      47    LDX #00              SET INDEX REGS TO
4041:A0 0C      48    LDY #12              SEPARATE TOP & BOTTOM OF SCREEN
4043:A9 02      49 SLOOP  LDA #SEPR        LOAD SEPR CHARACTER
4045:20 12 42   50    JSR OUTPUT
4048:E8         51    INX                  INCREMENT SEPR COUNTER
4049:E0 50      52    CPX #80              CHECK IF END
404B:D0 F6      53    BNE SLOOP            IF NO BRANCH TO FINISH SEPR
404D:60         54    RTS                  END OF INIT ROUTINE
404E:           55    ****************************************
404E:           56    * INTERRUPT SERVICE ROUTINE *
404E:           57    ****************************************
404E:AD A9 C0   58 START  LDA STATUS+SLOT  LOAD STATUS REG OF 1ST SERIAL PORT
4051:29 08      59    AND #$8              CHECK IF BUFFER FULL
4053:F0 0E      60    BEQ NSLOT            IF NOT CHECK 2ND SERIAL PORT
4055:A9 01      61    LDA #$01             SET KEYBOARD FLAG TO 1
4057:8D 79 42   62    STA FLAG
405A:AD A8 C0   63    LDA DATA+SLOT        GET DATA FROM BUFFER
405D:8D 7A 42   64    STA INDATA           SAVE INCOMING CHARACTER
4060:20 8C 40   65    JSR DISPLAY          JUMP TO TOP DISPLAY ROUTINE
4063:AD C9 C0   66 NSLOT  LDA STATUS+SLOT2 LOAD STATUS REG OF 2ND SERIAL PORT
4066:29 08      67    AND #$8              CHECK IF BUFFER FULL
4068:F0 0E      68    BEQ NOSLOT           IF NOT ESC FROM INTERRUPT ROUTINE
406A:A9 02      69    LDA #$02             SET KEYBOARD FLAG TO 2
406C:8D 79 42   70    STA FLAG
406F:AD C8 C0   71    LDA DATA+SLOT2       GET DATA FROM BUFFER
4072:8D 7A 42   72    STA INDATA           SAVE INCOMING CHARACTER
4075:20 4F 41   73    JSR DISPLA2          JUMP TO BOTTOM DISPLAY ROUTINE
4078:A5 45      74 NOSLOT LDA ACC          RESTORE ACCUMULATOR
407A:40         75    RTI
407B:           76    ****************************************
407B:           77    * SET REGISTERS FOR 80 COL CARD *
407B:           78    ****************************************
407B:8D FF CF   79 SETREGS STA $CFFF
407E:8D 00 C3   80    STA $C300
4081:A0 30      81    LDY #$30
4083:8C F8 06   82    STY NO
4086:A2 C3      83    LDX #$C3
4088:8E F8 07   84    STX MSLOT
408B:60         85    RTS
408C:           86    ****************************************
408C:           87    * DISPLAY FOR FIRST DEVICE *
408C:           88    ****************************************
408C:8D 7B 06   89 DISPLAY STA BYTE+3      SAVE ACCUMULATOR
408F:8A         90    TXA                  SAVE X-REG
4090:48         91    PHA
4091:98         92    TYA                  SAVE Y-REG
4092:48         93    PHA
4093:20 7B 40   94    JSR SETREGS          SET REGS FOR 80-COL CARD
4096:AD 7B 06   95    LDA BYTE+3           RESTORE ACCUMULATOR
4099:C9 0D      96    CMP #$0D             CHECK IF INCOME CHAR = RETURN?
409B:D0 4C      97    BNE VDOUT1           IF NO BRANCH VIDEO OUTPUT ROUTINE
409D:A9 20      98    LDA #SP              LOAD NULL CHARACTER TO ACC
409F:AE 4D 41   99    LDX HOR1             LOAD HORIZONTAL POINTER
```

```
40A2:AC 4E 41    100     LDY VER1          LOAD VERTICAL POINTER
40A5:20 12 42    101     JSR OUTPUT        BRANCH TO OUTPUT
40A8:20 AE 40    102     JSR CRLF1         GENERATE LINEFEED & CR
40AB:4C 13 41    103     JMP ENDDS1        JUMP END INTERRUPT SERV ROUTINE
40AE:            104     ********************************
40AE:            105     * GENERATE CR AND LF *
40AE:            106     ********************************
40AE:A2 00       107     CRLF1  LDX #0     RESET HORIZONTAL POINTER
40B0:8E 4D 41    108     STX HOR1
40B3:AC 4E 41    109     LDY VER1          INCREMENT VERTICAL POINTER
40B6:C8          110     INY
40B7:C0 0C       111     CPY #12           CHECK END OF PAGE
40B9:D0 25       112     BNE ENDVD1        IF NO THEN DON'T SCROLL
40BB:A2 00       113     LDX #0            SET POINTERS TO BEGINNING OF PAGE
40BD:A0 01       114     LDY #1
40BF:20 2E 42    115 MLOOP1 JSR INPUT      INPUT CHARACTER
40C2:20 12 42    116     JSR OUTPUT        OUTPUT CHARACTER
40C5:E8          117     INX               INCREMENT HORIZ POINTER
40C6:E0 50       118     CPX #80           CHECK FOR END OF LINE
40C8:D0 F5       119     BNE MLOOP1        IF NO THEN BRANCH BACK
40CA:A2 00       120     LDX #0            RESET HORIZ POINTER
40CC:C8          121     INY               INCREMENT VERT POINTER
40CD:C0 0B       122     CPY #11           CHECK END OF PAGE
40CF:D0 EE       123     BNE MLOOP1        IF NO BRANCH BACK
40D1:A9 20       124     LDA #$20          LOAD NULL CHARACTER
40D3:20 12 42    125 WLOOP1 JSR OUTPUT     CLEAR BOTTOM LINE
40D6:E8          126     INX               OF PAGE WITH WLOOP
40D7:E0 50       127     CPX #80
40D9:D0 F8       128     BNE WLOOP1
40DB:A2 00       129     LDX #0            RESET HORIZ POINTER
40DD:AC 4E 41    130     LDY VER1
40E0:8C 4E 41    131 ENDVD1 STY VER1
40E3:A9 07       132     LDA #CUR          PUT CURSOR TO NEXT LOCATION
40E5:20 12 42    133     JSR OUTPUT
40E8:60          134     RTS
40E9:            135     ************************************
40E9:            136     * VIDEO OUTPUT ROUTINE 1 *
40E9:            137     ************************************
40E9:C9 20       138 VDOUT1 CMP #$20       CHECK NULL CHARACTER
40EB:AE 4D 41    139     LDX HOR1          LOAD HORIZ POINTER
40EE:AC 4E 41    140     LDY VER1          LOAD VERT POINTER
40F1:20 12 42    141     JSR OUTPUT        JUMP TO OUTPUT ROUTINE
40F4:AD 7B 06    142     LDA BYTE+3        RESTORE CHARACTER AT ACC
40F7:C9 08       143     CMP #8            CHECK IF BACKSPACE
40F9:D0 03       144     BNE NOBS1         NO THEN BRANCH
40FB:4C 02 41    145     JMP YBS1          BRANCH TO BACKSPACE
40FE:C9 7F       146 NOBS1  CMP #127       CHECK IF DELETE
4100:D0 06       147     BNE GETCH1        NO THEN BRANCH
4102:20 36 41    148 YBS1   JSR DCRHOR1    BRANCH TO DECREMNT HORIZ POINTER
4105:4C 0E 41    149     JMP CUR1          JUMP TO CURSOR OUTPUT
4108:20 12 42    150 GETCH1 JSR OUTPUT     OUTPUT CHARACTER
410B:20 1B 41    151     JSR INRHOR1       INCREMENT HORIZ POINTER
410E:A9 07       152 CUR1   LDA #CUR       LOAD CUR CHAR TO ACC
4110:20 12 42    153     JSR OUTPUT
4113:68          154 ENDDS1 PLA            RESTORE REGISTERS
4114:A8          155     TAY
4115:68          156     PLA
4116:AA          157     TAX
4117:AD 7B 06    158     LDA BYTE+3
411A:60          159     RTS               ESCAPE FROM SERVICE ROUTINE
411B:            160     ************************************************
411B:            161     * INCREMENT HORIZONTAL POINTER *
411B:            162     ************************************************
411B:E0 4F       163 INRHOR1 CPX #79       CHECK IF END OF LINE
411D:F0 05       164     BEQ EQ79          YES THEN BRANCH
411F:E8          165     INX               NO INCREMENT HORIZ POINTER
4120:8E 4D 41    166     STX HOR1          SAVE HORIZ POINTER
4123:60          167     RTS               RETURN
4124:C0 0B       168 EQ79   CPY #11        IF END OF LINE CHECK END OF PAGE
4126:F0 0A       169     BEQ EQ12          IF YES THEN BRANCH
4128:A2 00       170     LDX #0            RESET HORIZ POINTER
412A:8E 4D 41    171     STX HOR1          SAVE HORIZ POINTER
412D:C8          172     INY               INCREMENT VERT POINTER
412E:8C 4E 41    173     STY VER1          SAVE VERT POINTER
4131:60          174     RTS               RETURN
4132:20 AE 40    175 EQ12   JSR CRLF1      IF END OF PAGE GENERATE LF & CR
4135:60          176     RTS               RETURN
4136:            177     ************************************************
4136:            178     * DECREMENT HORIZONTAL POINTER *
4136:            179     ************************************************
```

```
4136:E0 00      180 DCRHOR1 CPX #0      CHECK IF BEGINNING OF LINE
4138:F0 05      181     BEQ EQ00        IF YES BRANCH
413A:CA         182     DEX             NO THEN DECREMENT HORIZ POINTER
413B:8E 4D 41   183     STX HOR1        SAVE HORIZ POINTER
413E:60         184     RTS
413F:C0 01      185 EQ00  CPY #01       CHECK TOP OF PAGE
4141:F0 09      186     BEQ EQ01        IF YES THEN BRANCH
4143:A2 4F      187     LDX #79         NO THEN SET HORIZ POINTR END OF LINE
4145:8E 4D 41   188     STX HOR1        SAVE HORIZ POINTER
4148:88         189     DEY             DECREMENT VERT POINTER
4149:8C 4E 41   190     STY VER1        SAVE VERT POINTER
414C:60         191 EQ01  RTS
414D:           192     *
414D:           193     *
414D:           194     *
414D:00         195 HOR1  DFB 0
414E:01         196 VER1  DFB 1
414F:           197     ****************************************
414F:           198     * DISPLAY FOR SECOND DEVICE  *
414F:           199     ****************************************
414F:           200     *          FOR COMMENTS          *
414F:           201     *  SEE LIST FOR FIRST DEVICE     *
414F:           202     *                                *
414F:           203     ****************************************
414F:8D 7B 06   204 DISPLA2 STA  BYTE+3
4152:8A         205     TXA
4153:48         206     PHA
4154:98         207     TYA
4155:48         208     PHA
4156:20 7B 40   209     JSR SETREGS
4159:AD 7B 06   210     LDA BYTE+3
415C:C9 0D      211     CMP #$0D
415E:D0 4C      212     BNE VDOUT2
4160:A9 20      213     LDA #SP
4162:AE 10 42   214     LDX HOR2
4165:AC 11 42   215     LDY VER2
4168:20 12 42   216     JSR OUTPUT
416B:20 71 41   217     JSR CRLF2
416E:4C D6 41   218     JMP ENDDS2
4171:           219     ****************************
4171:           220     * GENERATE CR AND LF *
4171:           221     ****************************
4171:A2 00      222 CRLF2  LDX #0
4173:8E 10 42   223     STX HOR2
4176:AC 11 42   224     LDY VER2
4179:C8         225     INY
417A:C0 18      226     CPY #24
417C:D0 25      227     BNE ENDVD2
417E:A2 00      228     LDX #0
4180:A0 0D      229     LDY #13
4182:20 2E 42   230 MLOOP2 JSR INPUT
4185:20 12 42   231     JSR OUTPUT
4188:E8         232     INX
4189:E0 50      233     CPX #80
418B:D0 F5      234     BNE MLOOP2
418D:A2 00      235     LDX #0
418F:C8         236     INY
4190:C0 17      237     CPY #23
4192:D0 EE      238     BNE MLOOP2
4194:A9 20      239     LDA #$20
4196:20 12 42   240 WLOOP2 JSR OUTPUT
4199:E8         241     INX
419A:E0 50      242     CPX #80
419C:D0 F8      243     BNE WLOOP2
419E:A2 00      244     LDX #0
41A0:AC 11 42   245     LDY VER2
41A3:8C 11 42   246 ENDVD2 STY VER2
41A6:A9 07      247     LDA #CUR
41A8:20 12 42   248     JSR OUTPUT
41AB:60         249     RTS
41AC:           250     ************************************
41AC:           251     *VIDEO OUTPUT ROUTINE 2 *
41AC:           252     ************************************
41AC:C9 20      253 VDOUT2 CMP #$20
41AE:AE 10 42   254     LDX HOR2
41B1:AC 11 42   255     LDY VER2
41B4:20 12 42   256     JSR OUTPUT
41B7:AD 7B 06   257     LDA BYTE+3
41BA:C9 08      258     CMP #8
```

```
41BC:D0 03      259     BNE NOBS2
41BE:4C C5 41   260     JMP YBS2
41C1:C9 7F      261 NOBS2  CMP #127
41C3:D0 06      262     BNE GETCH2
41C5:20 F9 41   263 YBS2  JSR DCRHOR2
41C8:4C D1 41   264     JMP CUR2
41CB:20 12 42   265 GETCH2 JSR OUTPUT
41CE:20 DE 41   266     JSR INRHOR2
41D1:A9 07      267 CUR2  LDA #CUR
41D3:20 12 42   268     JSR OUTPUT
41D6:68         269 ENDDS2 PLA
41D7:A8         270     TAY
41D8:68         271     PLA
41D9:AA         272     TAX
41DA:AD 7B 06   273     LDA BYTE+3
41DD:60         274     RTS
41DE:           275     ********************************************
41DE:           276     * INCREMENT HORIZONTAL POINTER *
41DE:           277     ********************************************
41DE:E0 4F      278 INRHOR2 CPX #79
41E0:F0 05      279     BEQ EQ792
41E2:E8         280     INX
41E3:8E 10 42   281     STX HOR2
41E6:60         282     RTS
41E7:C0 17      283 EQ792  CPY #23
41E9:F0 0A      284     BEQ EQ23
41EB:A2 00      285     LDX #0
41ED:8E 10 42   286     STX HOR2
41F0:C8         287     INY
41F1:8C 11 42   288     STY VER2
41F4:60         289     RTS
41F5:20 71 41   290 EQ23  JSR CRLF2
41F8:60         291     RTS
41F9:           292     ********************************************
41F9:           293     * DECREMENT HORIZONTAL POINTER *
41F9:           294     ********************************************
41F9:E0 00      295 DCRHOR2 CPX #0
41FB:F0 05      296     BEQ EQ002
41FD:CA         297     DEX
41FE:8E 10 42   298     STX HOR2
4201:60         299     RTS
4202:C0 0D      300 EQ002  CPY #13
4204:F0 09      301     BEQ EQ13
4206:A2 4F      302     LDX #79
4208:8E 10 42   303     STX HOR2
420B:88         304     DEY
420C:8C 11 42   305     STY VER2
420F:60         306 EQ13  RTS
4210:           307     *
4210:           308     *
4210:           309     *
4210:00         310 HOR2  DFB 0
4211:0D         311 VER2  DFB 13
4212:           312     ********************************************
4212:           313     * OUTPUT A CHARACTER TO SCREEN *
4212:           314     *  POINTED BY X AND Y REGISTERS    *
4212:           315     ********************************************
4212:48         316 OUTPUT PHA              PUSH ACC
4213:8E 7B 05   317     STX HOR             SAVE HORIZ POINTER
4216:8C FB 05   318     STY VER             SAVE VERT POINTER
4219:20 4C 42   319     JSR VTAB            SET ADDRESSES FOR OUTPUT
421C:68         320     PLA                 POP ACC
421D:B0 05      321     BCS WRITE           IF CARRY SET WRITE TO
421F:9D 00 CC   322     STA $CC00,X         FIRST BANK OF 80-COL CARD
4222:90 03      323     BCC WSKIP           IF NOT WRITE TO
4224:9D 00 CD   324 WRITE  STA $CD00,X      SECOND BANK OF 80-COL CARD
4227:AE 7B 05   325 WSKIP  LDX HOR          RESTORE POINTERS
422A:AC FB 05   326     LDY VER
422D:60         327     RTS
422E:           328     ********************************************
422E:           329     * INPUT A CHARACTER FROM SCREEN *
422E:           330     *     POINTED BY X AND Y+1          *
422E:           331     ********************************************
422E:48         332 INPUT  PHA              SAVE ACC
422F:C8         333     INY                 INCREMENT VERT POINTER
4230:8E 7B 05   334     STX HOR             SAVE HORIZ POINTER
```

```
4233:8C FB 05      335     STY VER              SAVE VERT POINTER
4236:20 4C 42      336     JSR VTAB             CALCULATE ADDRESSES FOR OUTPUT
4239:68            337     PLA                  RESTORE ACC
423A:B0 05         338     BCS READ             IF CARRY SET READ FROM
423C:BD 00 CC      339     LDA $CC00,X          FIRST BANK OF 80-COL CARD
423F:90 03         340     BCC RSKIP            IF NOT READ FROM
4241:BD 00 CD      341 READ  LDA $CD00,X        SECOND BANK OF 80-COL CARD
4244:AE 7B 05      342 RSKIP LDX HOR            RESTORE POINTERS
4247:AC FB 05      343     LDY VER
424A:88            344     DEY
424B:60            345     RTS
424C:              346     ********************************************
424C:              347     *      THIS SUBROUTINE IS USED             *
424C:              348     *  TO SET ADDRESSES FOR OUTPUT             *
424C:              349     *  ALL THE CALCULATIONS BELOW              *
424C:              350     *  ARE RELATED TO THE INTERNAL             *
424C:              351     *     SETUP OF THE 80-COL CARD             *
424C:              352     *    THIS SUBROUTINE REQUIRES              *
424C:              353     *    HORIZONTAL AND VERTICAL               *
424C:              354     *    POINTERS TO ADJUST CARRY              *
424C:              355     *         FLAG AND X-REGISTER              *
424C:              356     ********************************************
424C:AD FB 05      357 VTAB  LDA VER
424F:0A            358     ASL A
4250:0A            359     ASL A
4251:18            360     CLC
4252:6D FB 05      361     ADC VER
4255:48            362     PHA
4256:4A            363     LSR A
4257:4A            364     LSR A
4258:4A            365     LSR A
4259:4A            366     LSR A
425A:8D FB 04      367     STA BASEH
425D:68            368     PLA
425E:0A            369     ASL A
425F:0A            370     ASL A
4260:0A            371     ASL A
4261:0A            372     ASL A
4262:18            373     CLC
4263:6D 7B 05      374     ADC HOR
4266:48            375     PHA
4267:A9 00         376     LDA #$00
4269:6D FB 04      377     ADC BASEH
426C:48            378     PHA
426D:0A            379     ASL A
426E:29 0C         380     AND #$0C
4270:AA            381     TAX
4271:BD B0 C0      382     LDA $C0B0,X
4274:68            383     PLA
4275:4A            384     LSR A
4276:68            385     PLA
4277:AA            386     TAX
4278:60            387     RTS
427A:00            389 INDATA DFB 0
```

SUCCESSFUL ASSEMBLY: NO ERRORS

```
   45 ACC          04FB BASEH      0678 BYTE       C08A COMMAND
C08B CONTROL       40AE CRLF1      4171 CRLF2         07 CUR
410E CUR1          41D1 CUR2        C088 DATA      4136 DCRHOR1
41F9 DCRHOR2       414F DISPLA2    408C DISPLAY    4113 ENDDS1
41D6 ENDDS2        40E0 ENDVD1     41A3 ENDVD2     413F EQ00
4202 EQ002         414C EQ01       4132 EQ12       420F EQ13
41F5 EQ23          4124 EQ79       41E7 EQ792      4279 FLAG
4108 GETCH1        41CB GETCH2     414D HOR1       057B HOR
4210 HOR2          427A INDATA     422E INPUT      411B INRHOR1
41DE INRHOR2       03FE IRQ1       03FF IRQ2       40BF MLOOP1
4182 MLOOP2        07F8 MSLOT      40FE NOBS1      41C1 NOBS2
4078 NOSLOT        06F8 NO         4063 NSLOT      4212 OUTPUT
4241 READ          4244 RSKIP        02 SEPR       407B SETREGS
4043 SLOOP           20 SLOT         40 SLOT2         20 SP
404E START         C089 STATUS     40E9 VDOUT1     41AC VDOUT2
05FB VER           414E VER1       4211 VER2       424C VTAB
40D3 WLOOP1        4196 WLOOP2     4224 WRITE      4227 WSKIP
4102 YBS1          41C5 YBS2
```

```
   02 SEPR          07 CUR          20 SLOT         20 SP
   40 SLOT2         45 ACC          03FE IRQ1       03FF IRQ2
04FB BASEH       057B HOR        05FB VER        0678 BYTE
4063 NSLOT       4078 NOSLOT     407B SETREGS    408C DISPLAY
40AE CRLF1       40BF MLOOP1     40D3 WLOOP1     40E0 ENDVD1
40E9 VDOUT1      40FE NOBS1      4102 YBS1       4108 GETCH1
410E CUR1        4113 ENDDS1     411B INRHOR1    4124 EQ79
4132 EQ12        4136 DCRHOR1    413F EQ00       414C EQ01
414D HOR1        414E VER1       414F DISPLA2    4171 CRLF2
4182 MLOOP2      4196 WLOOP2     41A3 ENDVD2     41AC VDOUT2
41C1 NOBS2       41C5 YBS2       41CB GETCH2     41D1 CUR2
41D6 ENDDS2      41DE INRHOR2    41E7 EQ792      41F5 EQ23
41F9 DCRHOR2     4202 EQ002      420F EQ13       4210 HOR2
4211 VER2        4212 OUTPUT     4224 WRITE      4227 WSKIP
422E INPUT       4241 READ       4244 RSKIP      424C VTAB
4279 FLAG        427A INDATA     C088 DATA       C089 STATUS
C08A COMMAND     C08B CONTROL
```

```
1 REM  THIS IS AN AUTOBOOT STARTUP PROGRAM. IT LOADS THE ASSEMBLY LANGUAGE
2 REM  PROGRAM AND ALLOWS THE USER TO CHANGE COMMUNICATION PARAMETERS.
3 REM  AFTER STARTING THE ASSEMBLY PROGRAM IT ERASES ITSELF TO MAKE ROOM
4 REM  FOR ANY OTHER BASIC PROGRAM THE USER WISHES TO LOAD.
5 REM  ************************************************************************
6 REM  ****** STARTUP ****** STARTUP ****** STARTUP ****** STARTUP ****** STARTUP ******
7 REM  ************************************************************************
10 CLEAR : HOME
20 PRINT  CHR$ (4) + "BLOAD COMMIIE.OBJ0"
30 VTAB 4
40 HTAB 5: PRINT "DEFAULT PARAMETERS FOR"
50 HTAB 5: PRINT "COMMUNICATION PROGRAM ARE:"
60 PRINT
70 HTAB 5: PRINT "BAUD.....300"
80 HTAB 5: PRINT "DATA BITS....7"90  HTAB 5: PRINT "STOP BITS....1"
100 HTAB 5: PRINT "PARITY.....E"
110 PRINT : PRINT : PRINT : HTAB 5
120 INPUT "ANY CHANGES? (Y/N) ";A$
130 IF A$ = "Y" THEN 170
140 IF A$ = "N" THEN 160
150 VTAB 14: HTAB 5: GOTO 120
160 CALL 16348
161 HOME : NEW : END
170 VTAB 16: HTAB 2: PRINT "VALID ENTRIES FOR ABOVE PARAMETERS ARE:"
171 HTAB 2: PRINT "BAUD.....300,600,1200,1800,2400,3600"
172 HTAB 13: PRINT "4800,7200,9600,19200"
173 HTAB 2: PRINT "DATA BITS..7,8"
174 HTAB 2: PRINT "STOP BITS..1,2"
175 HTAB 2: PRINT "PARITY.....E(VEN),O(DD),M(ARK)"
176 HTAB 13: PRINT "S(PACE),N(ONE)"
179 VTAB 7: HTAB 5
180 INPUT "BAUD.....";B$
190 BAUD = 0
200 IF B$ = "300" THEN BAUD = 6
210 IF B$ = "600" THEN BAUD = 7
220 IF B$ = "1200" THEN BAUD = 8
230 IF B$ = "1800" THEN BAUD = 9
240 IF B$ = "2400" THEN BAUD = 10
250 IF B$ = "3600" THEN BAUD = 11
260 IF B$ = "4800" THEN BAUD = 12
270 IF B$ = "7200" THEN BAUD = 13
280 IF B$ = "9600" THEN BAUD = 14
290 IF B$ = "19200" THEN BAUD = 15
300 IF BAUD = 0 THEN 170
310 CNTR = BAUD + 16
320 VTAB 8: HTAB 5
330 INPUT "DATA BITS....";D$
340 IF D$ = "7" THEN 370
350 IF D$ = "8" THEN 380
360 GOTO 320
370 CNTR = CNTR + 32
380 VTAB 9
390 HTAB 5: INPUT "STOP BITS....";S$
400 IF S$ = "1" THEN 440
410 IF S$ = "2" THEN 430
420 GOTO 380
430 CNTR = CNTR + 128
440 VTAB 10
450 IF S$ = "2" AND D$ = "8" THEN 540
460 HTAB 5: INPUT "PARITY.....";P$
```

```
461 CMND = 0
470 IF P$ = "E" THEN CMND = 105
480 IF P$ = "O" THEN CMND = 41
490 IF P$ = "M" THEN CMND = 169
500 IF P$ = "S" THEN CMND = 233
510 IF P$ = "N" THEN CMND = 9
520 IF CMND = 0 THEN 440
530 GOTO 560
540 HTAB 5: PRINT "PARITY.....N"
550 P$ = "N": GOTO 510
560 POKE 16385,CNTR: POKE 16393,CMND
570 GOTO 110
```

```
1 REM THIS IS AN EXAMPLE OF A BASIC PROGRAM THAT CAN BE RUN SIMULTANEOUSLY
2 REM WITH THE ASSEMBLY PROGRAM. IT SHOULD BE LOADED AND RUN AFTER THE
3 REM STARTUP PROGRAM. IT DISPLAYS THE CURRENT TIME AND DATE AND READS THE
4 REM DATA POKED BY THE ASSEMBLY PROGRAM AT LOCATIONS 17017 (KEYBOARD)
5 REM AND 17018 (ASCII CHARACTER)
6 REM **********
7 REM **** PEEK DEMO ***** PEEK DEMO ***** PEEK DEMO ***** PEEK DEMO***** PEEK DEMO ****
8 REM **********
15 SLOT = 5
20 D$ = CHR$ (4)
25 PRINT D$;"NOMONI,O,C"
30 HOME
34 REM READ THE CLOCK CARD (MOUNTAIN HARDWARE OR WESTSIDE) IN SLOT 5
40 PRINT D$;"IN#";SLOT
50 PRINT D$;"PR#";SLOT
60 VTAB 23
70 INPUT " ";T$
80 PRINT D$;"IN#0"
90 PRINT D$;"PR#0"
100 VTAB 12
105 REM DISPLAY CURRENT DATE & TIME
110 PRINT T$
112 VTAB 13: CALL - 868
115 REM DETERMINE & DISPLAY LAST KEYBOARD TO SEND MESSAGE (1 OR 2)
116 REM DETERMINE & DISPLAY LAST CHARACTER RECEIVED (0-255)
120 PRINT PEEK (17017), PEEK (17018)
125 REM TO DISPLAY ACTUAL CHARACTER INSTEAD OF DECIMAL EQUIVALENT USE CHR$
126 PRINT PEEK (17017), CHR$(PEEK(17018))
130 GOTO 40
140 END
```

(Manuscript received July 11, 1985;
revision accepted for publication October 28, 1985.)