UTILITY I/O MANAGER V4
for MC6800/MIKBUG Based Microcomputers

by D. R. Sentz

June 2015

Over the past year I developed this program for my homebrew MC6800-based microcomputer, to facilitate binary and text file transfers among the following equipment;

- Kleinschmidt AN/FGC-25 teletype set
- Audio Cassette Tape
- MacIntosh 512K Computer running MacTerminal 2.0
- Windows 7 PC running Tera Term 4.86, with TRENDNet TU-S9 USB-to-RS232C adapter

I developed the program as three "modules";

```
UTMGR4    -Text I/O between Mac and PC, Mac and audio tape,
           or PC and audio tape, or
          -Binary I/O between microcomputer and audio tape
TTY4       Adds output to teletype set from text buffer
READTTY    Adds input to text buffer from teletype set
```

You can load and run just UTMGR4, or UTMGR4 plus TTY4, or all three programs. TTY4 install patches to UTMGR4, and READTTY installs patches to both UTMGR4 and TTY4, therefore the programs must be loaded in the order listed above.

Memory requirements (in hexadecimal);

	Program	I/O buffer	
UTMGR4	1C00-1F71	0000-1BFF	binary (7 kbytes), or
		0100-1BFF	text (6¾ kbytes)
TTY4	1F72-1FF6	0100-1BFC	text (6¾ kbytes-2 bytes)
READTTY	1B00-1B8E	0100-1AFF	text (6½ kbytes)

If you load just UTMGR4 then you can read and write either text or binary data, in the range 0000-1BFF, from/to audio tape.

Just type MIKBUG "G" after loading the object code set, in the fllowing order;

UTMGR4.OBJ.txt,
TTY4.OBJ.txt, and
READTTY.OBJ.txt.

The size of the text buffer is 6½ kbytes when all three programs are resident.

I do not have a null modem for a direct connection between the Mac and the Win7 PC. To move text between the Mac and the PC I upload the text buffer from one, swap the MIKBUG console terminal cable to the other, and then download the text buffer. This worked fine for what I wanted to do, for text files up to 6½ kbytes. For example, I can print the resident assembler list file on the Apple Imagewriter printer even though I ran the assembler while using the PC as the MIKBUG console.

With just UTMGR4 loaded, I loaded from audio tape the binary program image of a floating point arithmetic subroutine package. Then I exited UTMGR4 using command "E" and downloaded the binary image to the Win7 PC console terminal, using the MIKBUG command "P". I did the same procedure for a scientific functions package and their demonstration drivers. So I can now load these math utilities to the microcomputer from the Win7 PC using MIKBUG command "L", as well as from the audio tape.

Using Command "E" and MIKBUG "G" provides seamless transfer of control back and forth between MIKBUG and UTMGR4.

An SWTPC6800 microcomputer should be able to load and run this program. See my other reports regarding

- my homebrew MC6800 microcomputer,
- audio cassette interface, and
- teletype current loop interface

The source and object code program listings that follow were generated by the paper tape version of the M6800 Resident Assembler MP-E that was sold by Southwest Technical Products Corp. (SWTPC). There are some leftover references in the comments to UTMGR3 and TTY3. Just read them as UTMGR4 and TTY4. UTMGR4 and TTY4 incorporated patches that I made to UTMGR3 and TTY3 during software testing.

PAGE 001   M6800 UT

```
00001                    NAM    M6800    UTILITY MANAGER
00002                    OPT    O,S,NOG  SEE ASSEMBLER DOC.

00004               * JUNE 13, 2014  BY DONALD R. SENTZ
00005               * 1ST TESTS JAN. 2015, CORRECTIONS 11 JAN 2015
00006               * 15 MAY 2015 CHANGED WTERM LINE TO LDX #$0100
00007               * THIS PROGRAM IS A USER INTERFACE FOR SWITCHING
00008               * AMONG MY HOMEBREW I/O UTILITIES
00009               *
00010               * THIS PROGRAM WAS PREPARED USING MACWRITE AND THE
00011               * RESIDENT ASSEMBLER MP-E.
00012               *
00013      E0C8     OUT4HS EQU    $E0C8    MIKBUG DISPLAY HEX SUBR.
00014      E1AC     INEEE  EQU    $E1AC    MIKBUG CHAR.INPUT TO A REG.
00015      E1D1     OUTEEE EQU    $E1D1    MIKBUG CHAR OUTPUT FROM A.
00016      E0D0     START  EQU    $E0D0    INIT (TO RESTORE ECHO-BACK)
00017      E0E3     CNTRL  EQU    $E0E3    MIKBUG RETURN POINT
00018      E07E     PDATA1 EQU    $E07E    MIKBUG CHAR STRING PRINTER
00019      8007     PIASB  EQU    $8007
00020      8010     ACIACT EQU    $8010    ACIA CONTROL REGISTER
00021      8011     ACIADT EQU    $8011    ACIA DATA REGISTER

00023 A002                    ORG    $A002
00024 A002 0002     BEGAD  RMB    2        START ADDRESS
00025 A004 0002     ENDAD  RMB    2        END ADDRESS
00026 A00C                    ORG    $A00C
00027 A00C 0001     XHI    RMB    1        X REG HIGH
00028 A00D 0001     XLOW   RMB    1        X REG LOW
00029 A042                    ORG    $A042    STACK POINTER
00030 A042 0001     STACK  RMB    1
00031 A048                    ORG    $A048    RESTART VECTOR
00032 A048 0002     RESTRT RMB    2

00034 1C00                    ORG    $1C00
00035 1C00 20 52    STRT   BRA    UTSTRT

00037               *RTERM- LOAD TEXT FILE FROM CONSOLE TERMINAL.
00038               *POINTER STOPS INCREMENTING AT END OF BUFFER.
00039               *USER MUST TYPE CTRL-D WHEN "SEND FILE" PROCESS
00040               *ENDS, I.E., AFTER BLINKING MACTERMINAL
00041               *CURSOR REAPPEARS.
00042               *
00043 1C02 86 3C    RTERM  LDA A  #$3C
00044 1C04 B7 8007         STA A  PIASB    DISABLE ECHO-BACK
00045 1C07 CE 0100         LDX    #$0100   START BUFFER AT ADDR $0100
00046 1C0A FF A002         STX    BEGAD
00047 1C0D BD E1AC  GET    JSR    INEEE    GET CHAR FROM CONSOLE
00048 1C10 A7 00    GET2   STA A  0,X      PUT CHAR TO TEXT BUFFER
00049 1C12 81 04           CMP A  #$4      WAS IT EOT (CTRL-D)?
00050 1C14 27 0E           BEQ    DONE
00051 1C16 8C 1BFF         CPX    #$1BFF   ARE WE AT END OF BUFFER?
00052 1C19 27 F2           BEQ    GET      IF SO, STAY THERE UNTIL EOT
00053 1C1B 08              INX             ELSE INCREMENT POINTER
00054 1C1C 81 0D           CMP A  #$D      WAS LAST CHAR A <CR>?
```

```
00055 1C1E 26 ED          BNE    GET      IF NOT GET NEXT CHAR
00056 1C20 86 0A          LDA A  #$A      ELSE INSERT <LF> AFTER <CR>
00057 1C22 20 EC          BRA    GET2
00058 1C24 86 34   DONE   LDA A  #$34     RESTORE ECHO-BACK
00059 1C26 B7 8007        STA A  PIASB
00060 1C29 FF A004        STX    ENDAD
00061 1C2C CE 1DB2        LDX    #MFOR    REPORT EOT LOCATION IN MEMORY
00062 1C2F BD E07E        JSR    PDATA1
00063 1C32 CE A004        LDX    #ENDAD
00064 1C35 BD E0C8        JSR    OUT4HS
00065 1C38 20 06          BRA    GOBACK

00067 1C3A CE 0100 WTERM  LDX    #$0100   TEXT BUFFER START ADDRESS
00068 1C3D BD E07E        JSR    PDATA1
00069 1C40 CE 1D9D GOBACK LDX    #MTHR    OPERATION COMPLETE
00070 1C43 20 12          BRA    REENTR

00072 1C45 BD 1D63 TAPOFF JSR    RESET
00073 1C48 20 F6          BRA    GOBACK

00075 1C4A CE 1E52 HELP   LDX    #MATE
00076 1C4D 20 08          BRA    REENTR

00078 1C4F CE 1D8B NOCMD  LDX    #MTWO    ELSE INVALID COMMAND
00079 1C52 20 03          BRA    REENTR

00081 1C54 CE 1D6D UTSTRT LDX    #MONE    <CR><LF>M6800 UTILITIES MANAG
00082 1C57 BD E07E REENTR JSR    PDATA1
00083 1C5A 8E A042 NOHEX  LDS    #STACK   CLEANUP STACK IF UTINHX BAILS
00084 1C5D CE 1D69        LDX    #MZRO    <CR><LF>?
00085 1C60 BD E07E        JSR    PDATA1
00086 1C63 BD E1AC        JSR    INEEE
00087 1C66 81 31          CMP A  #'1
00088 1C68 27 98          BEQ    RTERM    LOAD TEXT FILE FROM TERMINAL
00089 1C6A 81 32          CMP A  #'2
00090 1C6C 27 CC          BEQ    WTERM    SEND TEXT FILE TO TERMINAL
00091 1C6E 81 33          CMP A  #'3
00092 1C70 27 DD          BEQ    NOCMD    RESERVED FOR BEQ ASC2BD
00093 1C72 81 34          CMP A  #'4
00094 1C74 27 25          BEQ    WM2TAP   WRITE MEMORY TO TAPE
00095 1C76 81 35          CMP A  #'5
00096 1C78 27 2B          BEQ    RTAP2M   READ TAPE TO MEMORY
00097 1C7A 81 36          CMP A  #'6
00098 1C7C 27 C7          BEQ    TAPOFF   RESET ACIA (TAPE MOTOR OFF)
00099 1C7E 81 37          CMP A  #'7
00100 1C80 27 CD          BEQ    NOCMD    RESERVED
00101 1C82 81 38          CMP A  #'8
00102 1C84 27 C9          BEQ    NOCMD    RESERVED
00103 1C86 81 39          CMP A  #'9
00104 1C88 27 C5          BEQ    NOCMD    RESERVED
00105 1C8A 81 48          CMP A  #'H
00106 1C8C 27 BC          BEQ    HELP     PRINT COMMAND MENU
00107 1C8E 81 45          CMP A  #'E
00108 1C90 26 BD          BNE    NOCMD
```

```
00109 1C92 CE 1C00        LDX    #STRT    RESET PROG COUNTER IN STACK
00110 1C95 FF A048        STX    RESTRT
00111 1C98 7E E0D0        JMP    START    EXIT TO MIKBUG

00113 1C9B 8D 13   WM2TAP BSR    RWCOMN
00114 1C9D 81 41          CMP A  #'A
00115 1C9F 27 9F          BEQ    GOBACK
00116 1CA1 8D 69          BSR    WTAPE
00117 1CA3 20 9B          BRA    GOBACK
00118 1CA5 8D 09   RTAP2M BSR    RWCOMN
00119 1CA7 81 41          CMP A  #'A
00120 1CA9 27 95          BEQ    GOBACK
00121 1CAB BD 1D28        JSR    RTAPE
00122 1CAE 20 90          BRA    GOBACK

00124 1CB0 CE 1DC4 RWCOMN LDX    #MFIV
00125 1CB3 BD E07E        JSR    PDATA1
00126 1CB6 8D 22          BSR    BADDR
00127 1CB8 FF A002        STX    BEGAD
00128 1CBB CE 1DD8 RWCOM2 LDX    #MSIX
00129 1CBE BD E07E        JSR    PDATA1
00130 1CC1 8D 17          BSR    BADDR
00131 1CC3 FF A004        STX    ENDAD
00132 1CC6 CE 1DEC RETRY  LDX    #MSVN
00133 1CC9 BD E07E        JSR    PDATA1
00134 1CCC BD E1AC        JSR    INEEE
00135 1CCF 81 41          CMP A  #'A
00136 1CD1 27 06          BEQ    CONT
00137 1CD3 81 47          CMP A  #'G
00138 1CD5 27 02          BEQ    CONT
00139 1CD7 20 ED          BRA    RETRY    ASK AGAIN IF NO A OR G
00140 1CD9 39      CONT   RTS

00142 1CDA 8D 0C   BADDR  BSR    BYTE     THIS CODE COPIED FROM MIKBUG
00143 1CDC B7 A00C        STA A  XHI
00144 1CDF 8D 07          BSR    BYTE
00145 1CE1 B7 A00D        STA A  XLOW
00146 1CE4 FE A00C        LDX    XHI
00147 1CE7 39             RTS
00148 1CE8 8D 09   BYTE   BSR    UTINHX
00149 1CEA 48             ASL A
00150 1CEB 48             ASL A
00151 1CEC 48             ASL A
00152 1CED 48             ASL A
00153 1CEE 16             TAB
00154 1CEF 8D 02          BSR    UTINHX
00155 1CF1 1B             ABA
00156 1CF2 39      BYTERT RTS              SKIP MIKBUG CHECKSUM PART
00157 1CF3 BD E1AC UTINHX JSR    INEEE
00158 1CF6 80 30          SUB A  #$30
00159 1CF8 2B 0F          BMI    NOTHEX
00160 1CFA 81 09          CMP A  #$09
00161 1CFC 2F 0A          BLE    IN1HG
00162 1CFE 81 11          CMP A  #$11
```

```
00163 1D00 2B 07          BMI    NOTHEX
00164 1D02 81 16          CMP A  #$16
00165 1D04 2E 03          BGT    NOTHEX
00166 1D06 80 07          SUB A  #7
00167 1D08 39       IN1HG RTS
00168 1D09 7E 1C5A NOTHEX JMP    NOHEX

00170 1D0C CE 0802 WTAPE  LDX    #$0802   COND. MASKS FOR WRITE
00171 1D0F 8D 34          BSR    INIT
00172 1D11 A6 00   CONTW  LDA A  0,X      GET BYTE FROM MEMORY
00173 1D13 B7 8011        STA A  ACIADT   PUT IT TO I'FACE
00174 1D16 8D 42          BSR    LOOP2    WAIT UNTIL ACIA REG. "EMPTY"
00175 1D18 BC A004        CPX    ENDAD    WAS IT LAST BYTE TO WRITE?
00176 1D1B 27 03          BEQ    ENDCH
00177 1D1D 08             INX             POINT  TO NEXT MEM LOC.
00178 1D1E 20 F1          BRA    CONTW
00179               * ENDCH SENDS LAST BYTE TO ACIA A SECOND TIME, SO
00180               * THAT THE SERIAL OUTPUT OF ITS FIRST SENDING CAN
00181               * COMPLETE.
00182 1D20 B7 8011 ENDCH  STA A  ACIADT   PUT IT TO I'FACE AGAIN
00183 1D23 8D 35          BSR    LOOP2    WAIT FOR SERIAL OUT DONE
00184 1D25 8D 3C          BSR    RESET    NOW WE CAN TURN MOTOR OFF
00185 1D27 39             RTS

00187 1D28 CE 0401 RTAPE  LDX    #$0401   CODE MASKS FOR READ
00188 1D2B 8D 18          BSR    INIT
00189 1D2D C5 70   CONTR  BIT B  #$70     ANY ERROR FLAG SET?
00190 1D2F 27 03          BEQ    LOOP3    IF NOT, KEEP GOING
00191 1D31 8D 30   ENDOP  BSR    RESET    ELSE TURN OFF MOTOR
00192 1D33 39             RTS
00193 1D34 B6 8011 LOOP3  LDA A  ACIADT   GET BYTE JUST READ
00194 1D37 A7 00          STA A  0,X      STORE IT
00195 1D39 BC A004        CPX    ENDAD    ARE WE DONE?
00196 1D3C 27 F3          BEQ    ENDOP    IF SO, RETURN
00197 1D3E 08             INX             ELSE POINT TO NEXT LOC
00198 1D3F 8D 19          BSR    LOOP2    GET ANOTHER BYTE FROM TAPE
00199 1D41 20 EA          BRA    CONTR
00200               *SUBROUTINE ENTRY POINTS ARE INIT AND LOOP2
00201 1D43 0001    FLAG1  RMB    1        ACIA CONDITION CODE MASKS
00202 1D44 0001    FLAG2  RMB    1
00203 1D45 FF 1D43 INIT   STX    FLAG1    SAVE R/W CODE MASKS
00204 1D48 FE A002        LDX    BEGAD    GET STARTING ADDRESS
00205 1D4B 8D 16          BSR    RESET    GO INITIALIZE ACIA
00206 1D4D C6 1D          LDA B  #$1D     START MOTOR, DEFINE FORMAT
00207 1D4F F7 8010        STA B  ACIACT   8 BITS, ODD PARITY, 1 STOP
00208 1D52 F6 8010 LOOP1  LDA B  ACIACT
00209 1D55 F5 1D43        BIT B  FLAG1    TEST DCD (READ), CTS (WRITE)
00210 1D58 26 F8          BNE    LOOP1    WAIT FOR FLAG TO CLEAR
00211 1D5A F6 8010 LOOP2  LDA B  ACIACT
00212 1D5D F5 1D44        BIT B  FLAG2    TEST FULL (READ), EMPTY(WRITE
00213 1D60 27 F8          BEQ    LOOP2    WAIT UNTIL FLAG SET
00214 1D62 39             RTS
00215 1D63 C6 5F   RESET  LDA B  #$5F
00216 1D65 F7 8010        STA B  ACIACT
```

```
00217 1D68 39            RTS

00219              * MESSAGES
00220 1D69 0D      MZRO  FCB   $D,$A,'?,4
00221 1D6D 0D      MONE  FCB   $D,$A
00222 1D6F 4D            FCC   ^M6800 I/O UTILITIES MANAGER^
00223 1D8A 04            FCB   4
00224 1D8B 0D      MTWO  FCB   $D,$A
00225 1D8D 55            FCC   ^UNKNOWN COMMAND^
00226 1D9C 04            FCB   4
00227 1D9D 0D      MTHR  FCB   $D,$A
00228 1D9F 4F            FCC   ^OPERATION COMPLETE^
00229 1DB1 04            FCB   4
00230 1DB2 0D      MFOR  FCB   $D,$A
00231 1DB4 45            FCC   ^EOT LOCATION= $^
00232 1DC3 04            FCB   4
00233 1DC4 0D      MFIV  FCB   $D,$A
00234 1DC6 53            FCC   ^START ADDRESS=?_$^
00235 1DD7 04            FCB   4
00236 1DD8 0D      MSIX  FCB   $D,$A
00237 1DDA 20            FCC   ^  END ADDRESS=?_$^
00238 1DEB 04            FCB   4
00239 1DEC 0D      MSVN  FCB   $D,$A
00240 1DEE 50            FCC   ^POSITION TAPE, PRESS PLAY FOR READ^
00241 1E10 20            FCC   ^ OR RECORD FOR WRITE^
00242 1E24 0D            FCB   $D,$A
00243 1E26 57            FCC   ^WHEN READY, TYPE A TO ABORT, ^
00244 1E43 47            FCC   ^G TO PROCEED:_^
00245 1E51 04            FCB   4
00246 1E52 0D      MATE  FCB   $D,$A
00247 1E54 20            FCC   ^         COMMAND MENU^
00248 1E69 0D            FCB   $D,$A
00249 1E6B 4C            FCC   ^LOAD TEXT FILE FROM TERMINAL..1^
00250 1E8A 0D            FCB   $D,$A
00251 1E8C 53            FCC   ^SEND TEXT FILE TO TERMINAL....2^
00252 1EAB 0D            FCB   $D,$A
00253 1EAD 43            FCC   ^COMMAND CODE RESERVED.........3^
00254 1ECC 0D            FCB   $D,$A
00255 1ECE 57            FCC   ^WRITE MEMORY TO TAPE..........4^
00256 1EED 0D            FCB   $D,$A
00257 1EEF 52            FCC   ^READ TAPE TO MEMORY...........5^
00258 1F0E 0D            FCB   $D,$A
00259 1F10 54            FCC   ^TURN OFF TAPE MOTOR...........6^
00260 1F2F 0D            FCB   $D,$A
00261 1F31 50            FCC   ^PRINT COMMAND MENU............H^
00262 1F50 0D            FCB   $D,$A
00263 1F52 45            FCC   ^EXIT TO MIKBUG................E^
00264 1F71 04            FCB   4
00265              *
00266              * LOAD MIKBUG STACK PC WITH START ADDR
00267 A048               ORG   $A048
00268 A048 1C00          FDB   STRT
00269                    END

TOTAL ERRORS 00000
```

PAGE 001   TTY4

```
00001                    NAM    TTY4
00002                    OPT    O,S,NOG
00003               *
00004               * BY DONALD R. SENTZ JANUARY 2015
00005               *  MODS AND A CORRECTION APRIL 2015
00006               * ASCII TEXT BUFFER TO BAUDOT TTY OUTPUT
00007               * BASED ON FLOW CHARTS I DREW AND SUCCESSFUL
00008               * TEST THAT I PERFORMED IN 1978-79.
00009               * LOAD UTMGR3.OBJ BEFORE LOADING TTY4.OBJ
00010               *
00011      8008     PREGA  EQU    $8008    ADDR. OF MEK6800D1 PIA CHIP
00012      1C40     GOBACK EQU    $1C40    RETURN TO UTMGR3.OBJ
00013 0000                ORG    $0000
00014               *
00015               * HERE IS THE 7-BIT ASCII-TO-BAUDOT-CODE
00016               * LOOKUP TABLE I DESIGNED USING MS EXCEL 2010.
00017               * DETAILS IN MY EXCEL FILE ON TOSHIBA LAPTOP
00018               *
00019 0000 00             FCB    0,1,1,1,1,1,1,202,1,1,68,1,1,80,1,1
00020 0010 01             FCB    1,1,190,1,246,1,1,1,1,1,1,1,1,1,1,1
00021 0020 48             FCB    72,218,226,1,210,1,244,214
00022 0028 DE             FCB    222,228,1,1,216,198,248,250
00023 0030 EC             FCB    236,238,230,194,212,224,234,206
00024 0038 CC             FCB    204,240,220,252,1,1,1,242
00025 0040 01             FCB    1,134,178,156,146,130,154,180
00026 0048 A8             FCB    168,140,150,158,164,184,152,176
00027 0050 AC             FCB    172,174,148,138,160,142,188,166
00028 0058 BA             FCB    186,170,162,1,1,1,1,1
00029 0060 D6             FCB    214,134,178,156,146,130,154,180
00030 0068 A8             FCB    168,140,150,158,164,184,152,176
00031 0070 AC             FCB    172,174,148,138,160,142,188,166
00032 0078 BA             FCB    186,170,162,1,1,1,1,1
00033 0080 FE      LTRS   FCB    254
00034 0081 F6      FIGS   FCB    246
00035               *
00036               * WORKING REGISTERS
00037 0082 00      LKADR  FCB    0        MS BYTE OF 16 BIT ADDRESS
00038 0083 00      LKADR1 FCB    0        LS BYTE OF 16 BIT ADDRESS
00039 0084 00      LFFLAG FCB    0        INIT LFFLAG=0=LTRS MODE
00040 0085 8008    PIADR  FDB    PREGA    SAVES A FEW BYTES LATER
00041 0087 0002    BADDR  RMB    2        TEMP REG. FOR X
00042               * ADDING OUTPUT HEADER STRING. TO USE, CHANGE
00043               * INIT. OF BADDR FROM $0100 TO $00F6. THIS
00044               * HEADER FORCES PRINTER TO LTRS MODE
00045               * REGARDLESS OF CURRENT VALUE OF LFFLAG,
00046               * AND THEN SENDS <CR> AND <LF>
00047 00F6                ORG    $00F6
00048 00F6 00      HEADER FCB    0,0,0,0,0,$12,0,$12,$0D,$0A
00049               *
00050               *----------MAIN PROGRAM STARTS HERE--------
00051 1F72                ORG    $1F72
00052               *
00053               * THIS CODE PRINTS ASCII TEXT BUFFER TO TTY
00054               * INTERFACE UNTIL ASCII E-O-T ENCOUNTERED OR
```

PAGE 002 TTY4

```
00055                * POINTER IS AT END OF DEFINED BUFFER SPACE.
00056                *
00057 1F72 8D 22   START  BSR    INIT     CONFIGURE PIA
00058 1F74 C6 01          LDA B  #1       SYNC TO START OF 1MS
00059 1F76 8D 73          BSR    TIM1     TIM1 EXPECTS PIA ADDR. IN REG
00060 1F78 CE 00F6        LDX    #HEADER  INIT POINTER TO HEADER
00061 1F7B DF 87          STX    BADDR
00062                * HERE IS THE PRINTING LOOP
00063 1F7D DE 87   NEXT   LDX    BADDR    GET POINTER
00064 1F7F 8C 1BFD        CPX    #$1BFD   FAILSAFE TEST:END OF BUFFER+1
00065 1F82 26 03          BNE    BCONT    IF NOT END, CONTINUE
00066 1F84 7E 1C40 EXIT   JMP    GOBACK   ELSE PRINTOUT IS DONE
00067 1F87 A6 00   BCONT  LDA A  0,X      GET ASCII CHARACTER
00068 1F89 81 04          CMP A  #4       IS IT EOT?
00069 1F8B 26 02          BNE    BCONT2   IF NOT, CONTINUE
00070 1F8D 20 F5          BRA    EXIT
00071 1F8F 08      BCONT2 INX              INCREMENT BUFFER POINTER
00072 1F90 DF 87          STX    BADDR
00073 1F92 8D 14          BSR    PRCHR    GO PUT OUT CHAR. TO TTY
00074 1F94 20 E7          BRA    NEXT
00075                *
00076                * EXTERNALLY CALLABLE SUBR. TO INITIALIZE PIA
00077                * CONFIGURE PIA PORT SUCH THAT PIA0-B7 IS
00078                * AN INPUT, B6-B0 ARE OUTPUTS. ONLY B7 AND B0
00079                * ARE USED BY THE TTY INTERFACE BOARD.
00080                *
00081 1F96 DE 85   INIT   LDX    PIADR    GET PIA ADDRESS TO X
00082 1F98 4F             CLR A          ALL 0 TO CONTROL REG.
00083 1F99 A7 01          STA A  1,X      POINT PIADR TO DATA DIR. REG.
00084 1F9B 86 7F          LDA A  #$7F     CONF. B6-BO AS OUTPUTS
00085 1F9D A7 00          STA A  0,X      AND B7 AS INPUT.
00086 1F9F 86 04          LDA A  #4       CONTROL REG. B2 = 1
00087 1FA1 A7 01          STA A  1,X      POINT PIADR TO DATA REG.
00088 1FA3 86 01          LDA A  #1       NOW OUTPUT A MARK SIGNAL TO
00089 1FA5 A7 00          STA A  0,X      CLOSE THE TTY CURRENT LOOP
00090 1FA7 39             RTS
00091                *
00092                * EXTERNALLY CALLABLE SUBROUTINE TO CONVERT
00093                * 7-BIT ASCII CHAR. IN ACC. A TO BAUDOT CODE
00094                * AND OUTPUT IT TO TTY INTERFACE. THIS ROUTINE
00095                * CHECKS IF LTRS OR FIGS CODE NEEDS TO BE SENT
00096                * BEFORE SENDING THE CONVERTED CHARACTER, AND
00097                * DOES SO ONLY IF NECESSARY.
00098                *
00099 1FA8 84 7F   PRCHR  AND A  #$7F     ENSURE MSB=0 (FAILSAFE)
00100 1FAA 97 83          STA A  LKADR1   PREPARE LOOKUP ADDRESS
00101 1FAC DE 82          LDX    LKADR    GET TABLE LOOKUP ADDRESS
00102 1FAE A6 00          LDA A  0,X      GET BAUDOT CODEWORD
00103 1FB0 85 01          BIT A  #1       TEST: IS IT PRINTABLE?
00104 1FB2 26 20          BNE    DONE     IF LSB=1 THEN SKIP PRINT
00105 1FB4 36             PSH A          ELSE TEMPORARY SAVE CODEWORD
00106 1FB5 85 80          BIT A  #$80     IS IT CR, LF, OR SPACE?
00107 1FB7 27 18          BEQ    CONT2    IF SO, SKIP L/F CHECK
00108 1FB9 85 40          BIT A  #$40     ELSE IS IT LTR OR FIG?
```

```
00109 1FBB 26 09          BNE    FIG      IF L/F=1 THEN FIGS
00110                  * ELSE SEND LTRS CODE IF NECESSARY
00111 1FBD D6 84          LDA B  LFFLAG   CHECK LTRS/FIGS STATE
00112 1FBF 27 10          BEQ    CONT2    SKIP IF ALREADY LTRS MODE
00113 1FC1 5F             CLR B           ELSE CLEAR L/F FLAG
00114 1FC2 96 80          LDA A  LTRS     AND GET LTRS CODE
00115 1FC4 20 07          BRA    CONT3    AND SEND IT
00116                  * SEND FIGS CODE IF NECESSARY
00117 1FC6 D6 84   FIG    LDA B  LFFLAG   CHECK LTRS/FIGS STATE
00118 1FC8 26 07          BNE    CONT2    SKIP IF ALREADY FIGS MODE
00119 1FCA 53             COM B           ELSE SET L/F FLAG
00120 1FCB 96 81          LDA A  FIGS     AND GET FIGS CODE
00121                  *
00122 1FCD D7 84   CONT3  STA B  LFFLAG   UPDATE L/F STATE
00123 1FCF 8D 04          BSR    BAUDOT   SEND LTRS OR FIGS
00124 1FD1 32      CONT2  PUL A           GET CHAR. TO SEND
00125 1FD2 8D 01          BSR    BAUDOT   PRINT CHARACTER
00126 1FD4 39      DONE   RTS
00127                  *
00128                  * THIS SUBROUTINE OUTPUTS ACC.A B0-B6
00129                  * TO TTY INTERFACE.
00130                  * B0    = TTY START BIT = 0
00131                  * B1-B5 = BAUDOT CHARACTER
00132                  * B6    = TTY STOP BIT  = 1
00133                  *
00134 1FD5 DE 85   BAUDOT LDX    PIADR
00135 1FD7 C6 1F          LDA B  #31      HOLD STOP FOR 31 MS
00136 1FD9 8D 10          BSR    TIM1
00137 1FDB 8A 40          ORA A  #$40     PUT STOP BIT IN B6
00138 1FDD A7 00          STA A  0,X
00139                  * PUT START BIT OUT AT BEGINNING OF
00140                  * A 22 MS INTERVAL
00141 1FDF 86 06          LDA A  #6       INITIALIZE BIT SHIFT COUNTER
00142 1FE1 C6 16   SHIFT  LDA B  #22      SET TIMEOUT = 22 MS
00143 1FE3 8D 06          BSR    TIM1     HOLD FOR 22 MS
00144 1FE5 67 00          ASR    0,X      PLACE NEXT BIT ONTO OUTPUT
00145 1FE7 4A             DEC A
00146 1FE8 2E F7          BGT    SHIFT    KEEP SHIFTING UNTIL B6
00147 1FEA 39             RTS
00148                  *
00149                  * THIS ROUTINES COUNTS PERIODS OF THE
00150                  * 1 MS TIMER ON TTY INTERFACE BOARD.
00151                  *
00152 1FEB 6D 00   TIM1   TST    0,X      IS PIA0-B7 =1?
00153 1FED 2B FC          BMI    TIM1     YES, WAIT UNTIL B7=0
00154 1FEF 6D 00   TIM0   TST    0,X      IS PIA0-B7 =0?
00155 1FF1 2A FC          BPL    TIM0     YES, WAIT UNTIL B7=1
00156 1FF3 5A             DEC B
00157 1FF4 26 F5          BNE    TIM1     COUNT PERIODS UNTIL B=0
00158 1FF6 39             RTS

00160                  * INITIALIZE PRINT BUFFER WITH EOT
00161 0100                ORG    $0100
00162 0100 04             FCB    $04      PRELOAD ASCII E-O-T CODE
```

PAGE 004   TTY4

```
00164              * PATCHES TO UTMGR3. LOAD UTMGR3.OBJ
00165              * BEFORE LOADING TTY PRINT UTILITY.
00166              *
00167 1BFD                ORG    $1BFD
00168 1BFD 7E 1F72 TTY    JMP    START    LEAPFROG FROM UTMGR3
00169              *
00170              * PATCH END OF BUFFER ADDRESS
00171 1C17                ORG    $1C17
00172 1C17 1BFC           FDB    $1BFC    MAKE ROOM FOR "JMP START"
00173              *
00174              * PATCH COMMAND DECODER
00175 1C70                ORG    $1C70
00176 1C70 27 8B          BEQ    TTY      REPLACES BEQ NOCMD
00177              *
00178              * PATCH COMMAND MENU
00179 1EAD                ORG    $1EAD
00180 1EAD 50             FCC    ^PRINT TEXT BUFFER TO TTY......3^
00181                     END

TOTAL ERRORS 00000
```

PAGE 001 READTTY

```
00001                    NAM    READTTY
00002                    OPT    O,S,NOG
00003               *
00004               * JUNE 15, 2015  BY D.R. SENTZ
00005               *
00006               * UTMGR4.OBJ AND TTY4.OBJ MUST BE LOADED TO RUN THIS
00007               * PROGRAM. THIS PROGRAM READS SERIAL DATA INPUT ON T
00008               * L.S. BIT OF PIA REGISTER B OF THE MEK6800D1 CPU
00009               * BOARD, PERFORMS THE BAUDOT-TO-ASCII CHARACTER
00010               * CONVERSION, OUTPUTS THE ASCII CHARACTER TO THE
00011               * CONSOLE TERMINAL (MACTERMINAL 2.0 IN MY CASE), AND
00012               * STORES IT IN UTMGR4 TEXT BUFFER. THE CONSOLE
00013               * INTERFACE DATA RATE SHOULD BE 600 BAUD (300 MIGHT
00014               * WORK BUT I HAVEN'T TESTED IT). THE MEK6800D1
00015               * SUPPORTS 600 BAUD OK. USE THE LINE-BREAK SWITCH ON
00016               * THE TELEPRINTER TO EXIT THIS PROGRAM AND RETURN TO
00017               * THE UTILITY MANAGER COMMAND LINE.
00018               *
00019      0085     PIAADR EQU    $0085    TTY3 PUT PIA ADDRESS IN THIS
00020      0087     BADDR  EQU    $0087    LOC OF TEXT BUFFER POINTER
00021      1AFF     EOBADR EQU    $1AFF    NEW END OF UTMGR3 TEXT BUFFER
00022      1C29     RTNUTM EQU    $1C29    UTMGR3 REENTRY ON LINE BREAK
00023      1C4F     NOCMD  EQU    $1C4F    UTMGR3 REENTRY ON COMMAND ERR
00024      1F72     SEND   EQU    $1F72    TTY3 ENTRY POINT
00025      1F96     INIT   EQU    $1F96    SUBROUTINE TO INIT PIA REG. A
00026      1FEB     TIM1   EQU    $1FEB    TIMER SUBROUTINE IN TTY3.SRC
00027      E07E     PDATA1 EQU    $E07E    STRING OUTPUT SUBROUTINE
00028      E1AC     INEEE  EQU    $E1AC    CHARACTER INPUT SUBROUTINE
00029      E1D1     OUTEEE EQU    $E1D1    CHARACTER OUTPUT SUBROUTINE
00030               *
00031 0089                ORG    $0089
00032 0089 0001    BITCNT RMB    1        INPUT DATA BIT COUNTER
00033 008A 0001    OFFSET RMB    1        LTRS/FIGS ADDRESS OFFSET
00034               *
00035               *BAUDOT TO ASCII CONVERSION TABLE
00036               *
00037 008B 00      B2ASCI FCB    $00,$54,$0D,$4F,$20,$48,$4E,$4D
00038 0093 0A             FCB    $0A,$4C,$52,$47,$49,$50,$43,$56
00039 009B 45             FCB    $45,$5A,$44,$42,$53,$59,$46,$58
00040 00A3 41             FCB    $41,$57,$4A,$00,$55,$51,$4B,$00
00041 00AB 00      FIGS   FCB    $00,$35,$0D,$39,$20,$00,$2C,$2E
00042 00B3 0A             FCB    $0A,$29,$34,$26,$38,$30,$3A,$3B
00043 00BB 33             FCB    $33,$22,$24,$3F,$07,$36,$21,$2F
00044 00C3 2D             FCB    $2D,$32,$27,$00,$37,$31,$28,$00
00045 00CB 0D      MNINE  FCB    $0D,$0A
00046 00CD 53             FCC    ^SEND(S) OR RCV(R)?^
00047 00DF 04             FCB    $04
00048               *
00049 1B00                ORG    $1B00    TEXT BUFFER TO END AT $1AFF
00050               *
00051               * SOLICIT SEND OR RCV, IF RCV, INIT BADDR
00052               *
00053 1B00 CE 00CB SRTTY  LDX    #MNINE
00054 1B03 BD E07E        JSR    PDATA1   SOLICIT SEND OR RCV
```

```
00055 1B06 BD E1AC        JSR    INEEE    S=SEND, R=RECEIVE
00056 1B09 81 53          CMP A  #'S
00057 1B0B 26 03          BNE    CONT6
00058 1B0D 7E 1F72        JMP    SEND     OUTPUT TEXT BUFFER TO TELETYP
00059 1B10 81 52   CONT6  CMP A  #'R
00060 1B12 27 03          BEQ    RCV
00061 1B14 7E 1C4F        JMP    NOCMD    RETURN TO UTMGR IF NOT S OR R
00062 1B17 CE 0100 RCV    LDX    #$0100   INIT TEXT BUFFER POINTER
00063 1B1A DF 87          STX    BADDR
00064              *
00065              * CONFIGURE PIA A AND B, INIT. TABLE OFFSET.
00066              *
00067 1B1C BD 1F96        JSR    INIT     INIT PIA REG. A (TIMER INPUT)
00068 1B1F 4F             CLR A           NOW INIT PIA REG. B
00069 1B20 A7 03          STA A  3,X      POINT 2,X TO DDRB
00070 1B22 86 FE          LDA A  #$FE     REG.B B7-B1=OUTPUTS, BO=INPUT
00071 1B24 A7 02          STA A  2,X      PUT I/O DEF. TO DATA DIR. REG
00072 1B26 86 04          LDA A  #4       POINT 2,X TO I/O REG.B
00073 1B28 A7 03          STA A  3,X      3,X IS PIA CONTROL REG.B
00074 1B2A 7F 008A        CLR    OFFSET   INIT TO LETTERS MODE
00075              *
00076              * HERE IS THE TTY DATA RECEIVER
00077              *
00078              * AFTER COMPLETING TEXT ENTRY FROM TELETYPE SET,
00079              * OPERATE THE LINE-BREAK KEY TO RETURN TO UTILITY
00080              * MANAGER.
00081              *
00082 1B2D A6 02   CHKMRK LDA A  2,X      GET A SAMPLE OF THE LINE STAT
00083 1B2F 26 04          BNE    WAIT     IF MARK, THEN PROCEED
00084 1B31 DE 87          LDX    BADDR    IF BREAK, GET TEXT POINTER
00085 1B33 20 4D          BRA    EXIT     GO WRITE EOT AND WE'RE DONE
00086              *
00087 1B35 A6 02   WAIT   LDA A  2,X      SAMPLE MARK/SPACE STATE
00088 1B37 26 FC          BNE    WAIT     UNTIL MARK-TO-SPACE DETECTED
00089 1B39 C6 0B          LDA B  #11      WAIT 11MS (HALF BIT PERIOD)
00090 1B3B BD 1FEB        JSR    TIM1
00091 1B3E A6 02          LDA A  2,X      TEST FOR START BIT (SPACE)
00092 1B40 26 F3          BNE    WAIT     IF FALSE START, KEEP WAITING
00093 1B42 C6 05          LDA B  #5       ELSE INITIALIZE BITCNT
00094 1B44 D7 89          STA B  BITCNT
00095              *
00096              * ACC. A CONTAINS ALL ZEROES AT THIS POINT. READ
00097              * CHARACTER CODE FROM TELETYPE SET.
00098              *
00099 1B46 C6 16   LOAD5  LDA B  #22      WAIT ONE BIT PERIOD
00100 1B48 BD 1FEB        JSR    TIM1
00101 1B4B AA 02          ORA A  2,X      INSERT BIT TO LSB OF ACC. A
00102 1B4D 7A 0089        DEC    BITCNT   HAVE 5 BITS BEEN READ?
00103 1B50 27 03          BEQ    CEEONE   IF SO, PROCEED
00104 1B52 48             ASL A           ELSE LEFT SHIFT ACC.A
00105 1B53 20 F1          BRA    LOAD5    AND GET ANOTHER BIT
00106              *
00107              * CHECK LTRS OR FIGS, SET OFFSET ACCORDINGLY
00108              *
```

```
00109 1B55 81 1F   CEEONE CMP A  #$1F     WAS IT THE LTRS CODE?
00110 1B57 26 03          BNE    CONT2    IF NOT, CHECK FOR FIGS
00111 1B59 5F             CLR B           ELSE SET ACC. B =0
00112 1B5A 20 06          BRA    CONT3    AND GO UPDATE OFFSET
00113 1B5C 81 1B   CONT2  CMP A  #$1B     WAS IT THE FIGS CODE?
00114 1B5E 26 0B          BNE    CONT4    IF NOT, PROCEED TO LOOKUP
00115 1B60 C6 20          LDA B  #32      ELSE SET ACC. B =32
00116 1B62 D7 8A   CONT3  STA B  OFFSET   0=LTRS, 32=FIGS
00117 1B64 C6 16          LDA B  #22      DELAY TO 11/31THS OF STOP BIT
00118 1B66 BD 1FEB        JSR    TIM1
00119 1B69 20 C2          BRA    CHKMRK   GO TEST FOR STOP BIT OR BREAK
00120              *
00121              * HERE IS THE ASCII LOOKUP, OUTPUT TO MACTERMINAL,
00122              * AND WRITE TO TEXT BUFFER. I CHOSE TO USE SELF-
00123              * MODIFYING CODE HERE. THIS REMOVED A CONSTRAINT
00124              * ON THE START ADDRESS OF TABLE B2ASCI, AND SAVED
00125              * A FEW BYTES OF CODE.
00126              *
00127 1B6B 9B 8A   CONT4  ADD A  OFFSET   ADD OFFSET TO 5-BIT CODEWORD
00128 1B6D B7 1B74        STA A  EMBED+1  UDPATE INDEX IN LDAA BELOW
00129 1B70 CE 008B        LDX    #B2ASCI  START ADDRESS OF TABLE TO X
00130 1B73 A6 00   EMBED  LDA A  0,X      LOOK UP ASCII CODEWORD
00131              *
00132              * OUTEEE TAKES ABOUT 16.7MS TO EXECUTE AT 600
00133              * BAUD. THIS DELAYS NEXT INSTRUCTION UNTIL ABOUT
00134              * 6MS INTO THE 31MS STOP BIT PERIOD.
00135              *
00136 1B75 DE 87          LDX    BADDR    GET TEXT BUFFER POINTER
00137 1B77 A7 00          STA A  0,X      SAVE CHARACTER TO TEXT BUFFER
00138 1B79 BD E1D1        JSR    OUTEEE   OUTPUT CHARACTER TO CONSOLE
00139 1B7C 08             INX             UPDATE TEXT BUFFER POINTER
00140 1B7D 8C 1AFF        CPX    #EOBADR  HAS IT REACHED END OF BUFFER?
00141 1B80 26 07          BNE    CFIVE    IF NOT, THEN CONTINUE INPUT
00142 1B82 86 04   EXIT   LDA A  #$04     EOT ON BREAK OR END OF BUFFER
00143 1B84 A7 00          STA A  0,X      WRITE EOT TO TEXT BUFFER
00144 1B86 7E 1C29        JMP    RTNUTM   AND RETURN TO UTILITY MANAGER
00145 1B89 DF 87   CFIVE  STX    BADDR    SAVE UPDATED POINTER
00146 1B8B DE 85          LDX    PIAADR   RESTORE PIA ADDRESS TO X
00147 1B8D C6 0A          LDA B  #10      DELAY TO MIDDLE OF STOP BIT?
00148 1B8F BD 1FEB        JSR    TIM1
00149 1B92 20 99          BRA    CHKMRK   GO TEST FOR STOP BIT OR BREAK
00150              *
00151              *PATCH TTY3
00152 1BFD                ORG    $1BFD
00153 1BFD 7E 1B00        JMP    SRTTY    LEAPFROG ADDRESS
00154              *PATCH UTMGR3
00155 1C17                ORG    $1C17
00156 1C17 1AFF           FDB    EOBADR   END-OF-BUFFER ADDRESS
00157              *PATCH UTMGR3 COMMAND MENU
00158 1EAD                ORG    $1EAD
00159 1EAD 54             FCC    ^TELETYPE SEND OR RECEIVE......3^
00160              *PATCH TTY3
00161 1F80                ORG    $1F80
00162 1F80 1B00           FDB    EOBADR+1 END-OF-BUFFER PLUS 1
```

PAGE 004 READTTY

```
00163                    END
PIAADR 0085
BADDR  0087
EOBADR 1AFF
RTNUTM 1C29
NOCMD  1C4F
SEND   1F72
INIT   1F96
TIM1   1FEB
PDATA1 E07E
INEEE  E1AC
OUTEEE E1D1
BITCNT 0089
OFFSET 008A
B2ASCI 008B
FIGS   00AB
MNINE  00CB
SRTTY  1B00
CONT6  1B10
RCV    1B17
CHKMRK 1B2D
WAIT   1B35
LOAD5  1B46
CEEONE 1B55
CONT2  1B5C
CONT3  1B62
CONT4  1B6B
EMBED  1B73
EXIT   1B82
CFIVE  1B89

TOTAL ERRORS 00000
```

ALL THREE PROGRAM MODULES, UTMGR4,TTY4, AND READTTY IN ONE LOAD FILE

```
*UTMGR4.OBJ MAY 2015 BY D.R. SENTZ
INCORPORATES FIXES AND PATCHES TO UTMGR3.OBJ
S00B00004D3638303020555410
S11E1C002052863CB78007CE0100FFA002BDE1ACA7008104270E8C1BFF27F279
S11E1C1B08810D26ED860A20EC8634B78007FFA004CE1DB2BDE07ECEA004BDE3
S11E1C36E0C82006CE0100BDE07ECE1D9D2012BD1D6320F6CE1E522008CE1D79
S11E1C518B2003CE1D6DBDE07E8EA042CE1D69BDE07EBDE1AC81312798813206
S11E1C6C27CC813327DD813427258135272B813627C7813727CD813827C9812A
S11E1C873927C5814827BC814526BDCE1C00FFA0487EE0D08D138141279F8D10
S11E1CA269209B8D0981412795BD1D282090CE1DC4BDE07E8D22FFA002CE1D34
S11E1CBDD8BDE07E8D17FFA004CE1DECBDE07EBDE1AC81412706814727022092
S11E1CD8ED398D0CB7A00C8D07B7A00DFEA00C398D0948484848168D021B3941
S11E1CF3BDE1AC80302B0F81092F0A81112B0781162E038007397E1C5ACE08C5
S11E1D0E028D34A600B780118D42BCA00427030820F1B780118D358D3C39CEB9
S11D1D2904018D18C57027038D3039B68011A700BCA00427F3088D1920EA7D
S11E1D45FF1D43FEA0028D16C61DF78010F68010F51D4326F8F68010F51D449E
S11E1D6027F839C65FF78010390D0A3F040D0A4D3638303020492F4F205554EB
S11E1D7B494C4954494553204D414E41474552040D0A554E4B4E4F574E20436D
S11E1D964F4D4D414E44040D0A4F5045524154494F4E20434F4D504C4554452D
S11E1DB1040D0A454F54204C4F434154494F4E3D2024040D0A5354415254204D
S11E1DCC414444524553533D3F5F24040D0A2020454E4420414444524553539B
S11E1DE73D3F5F24040D0A504F534954494F4E20544150452C2050524553532B
S11E1E0220504C415920464F522052454144204F52205245434F524420464FD3
S11E1E1D522057524954450D0A5748454E2052454144592C20545950452041DC
S11E1E3820544F2041424F52542C204720544F2050524F434545443A5F040D0E
S11E1E530A2020202020202020434F4D4D414E44204D454E550D0A4C4F41FF
S11E1E6E44205445558542046494C452046524F4D205445524D494E414C2E2E40
S11E1E89310D0A53454E44205445558542046494C4520544F205445524D494E71
S11E1EA4414C2E2E2E2E320D0A434F4D4D414E4420434F4445205245534552B6
S11E1EBF5645442E2E2E2E2E2E2E2E2E330D0A5752495445204D454D4F5259B9
S11E1EDA20544F20544150452E2E2E2E2E2E2E2E2E2E340D0A52454144205435
S11E1EF541504520544F204D454D4F52592E2E2E2E2E2E2E2E2E2E350D0AF6
S11E1F105455524E204F46462054415045204D4F544F522E2E2E2E2E2E2E2E03
S11E1F2B2E2E2E360D0A5052494E5420434F4D4D414E44204D454E552E2E2E35
S11E1F462E2E2E2E2E2E2E2E2E480D0A4558495420544F204D494B4255472E75
S1141F612E2E2E2E2E2E2E2E2E2E2E2E2E2E2E2E450470
S105A0481C00F6

*TTY4.OBJ (ADD-ON TO UTILITY MANAGER)
INCORPORATES FIXES AND PATCHES TO TTY3.OBJ
SEE TTY4.LST.txt FOR PROGRAM LISTING
MAY 15, 2015 BY D.R. SENTZ
S00B0000545459342020203F
S11E00000010101010101CA0101440101500101010101BE01F6010101010101BA
S11E001B010101010148DAE201D201F4D6DEE40101D8C6F8FAECEEE6C2D4E095
S11E0036EACECCF0DCFC010101F20186B29C92829AB4A88C969EA4B898B0AC7B
S11E0051AE948AA08EBCA6BAAAA20101010101D686B29C92829AB4A88C969EB5
S11E006CA4B898B0ACAE948AA08EBCA6BAAAA20101010101FEF6000000800842
S10D00F600000000001200120D0AC1
S11E1F728D22C6018D73CE00F6DF87DE878C1BFD26037E1C40A6008104260251
S11E1F8D20F508DF878D1420E7DE854FA701867FA7008604A7018601A700396B
S11E1FA8847F9783DE82A60085012620368580271885402609D68427105F9637
S11E1FC3802007D6842607539681D7848D04328D0139DE85C61F8D108A40A72C
S11C1FDE008606C6168D0667004A2EF7396D002BFC6D002AFC5A26F53907
S104010004F6
S1061BFD7E1F72D2
S1051C171BFCB0
S1051C70278BBC
S11E1EAD5052494E54205445585420425546464552205 44F205454592E2E2EDC
S1071EC82E2E2E3355
```

```
*JUNE 15 2015 READTTY.OBJ (ADD-ON TO UTILITY MANAGER)
S00B000052454144545920B7
S11E008B00540D4F20484E4D0A4C524749504356455A44425359465841574A31
S11E00A60055514B0000350D3920002C2E0A29342638303A3B3322243F0736F6
S11E00C1212F2D322700373128000D0A53454E44285329204F522052435628E1
S10700DC52293F045E
S11E1B00CE00CBBDE07EBDE1AC815326037E1F72815227037E1C4FCE0100DF28
S11E1B1B87BD1F964FA70386FEA7028604A7037F008AA6022604DE87204DA605
S11E1B360226FCC60BBD1FEBA60226F3C605D789C616BD1FEBAA027A0089276F
S11E1B51034820F1811F26035F2006811B260BC620D78AC616BD1FEB20C29B97
S11E1B6C8AB71B74CE008BA600DE87A700BDE1D1088C1AFF26078604A7007E87
S1101B871C29DF87DE85C60ABD1FEB2099EF
S1061BFD7E1B0048
S1051C171AFFAE
S11E1EAD54454C45545950452053454E44204F522052454345495645 2E2E2EF2
S1071EC82E2E2E3355
S1051F801B0040
S9
```