Albany NY
Aug 86

UPSTATE UPSTATE UPSTATE UPSTATE UPSTATE UPSTATE UPSTATE UPSTATE UPSTATE UPSTATE

OUR NEXT MEETING will be on Thursday NO AUGUST MEETING

SEPTEMBER 18, 1986 AT 7:30 p.m.

PLACE: CAPITAL DISTRICT PSYCHIATRIC CENTER

New Scotland Ave. Next to Albany Medical Center

THERE WILL BE NO MEETING IN AUGUST BUT THERE WILL BE IN SEPTEMBER.
Future topics will include updates on PRBASE and a new fareware package called CREATIVE FILING SYSTEM.
New hardware for the TI and demos of PASCAL programs
** Short Note: Anybody have a Corcomp Disk Controller Card for sale? If so call Art at (518) 370-5215.

A NOTE to other Users Groups: The articles printed in the Upstate Newsletter may be reprinted if proper credit is given to the author and to the Upstate New York 99/4 Users Group.

ALBANY TEXAS INSTRUMENTS BULLETIN BOARD SERVICE (TIBBS)-out of service

UPSTATE 99/4A USERS GROUP
P.O. BOX 13522
ALBANY, N.Y. 12212

VOL IV, NO. 7 AUGUST 1986 Allison Smith, EDITOR 439-4860

```
*                                ANALYSIS OF DSR PROGRAM
*
       DEF  DSR              This puts DSR on the REF/DEF table to be accessed by
*                            other programs.
       REF  DSRLNK,VMBW,VMBR,VSBW,KSCAN  This puts on the REF/DEF table, for
*                            access by this program, the following utilities:
*                            DSRLNK:  Links to Device Service Routines (inclu-
*                            ding printer and disk drive files.  Must be fol-
*                            lowed by data 8 for DSR or data 16 for subprogram
*                            and requires the construction of a PAB (see PDATA
*                            in 9 program steps.
*                            VMBW:  Writes multiple bytes to VDP RAM--# bytes
*                            must be given in R2, start of VDP RAM buffer where
*                            bytes are to be written in R0, start of buffer
*                            from which bytes are to be read in R1.
*                            VMBR:  Reads #bytes in R2 from VDP RAM buffer
*                            starting  at R0 to CPU buffer starting at R1.
*                            VSBW:   Writes the most significant byte from R1
*                            to the address given in the VDP RAM given in R0.
*                            KSCAN:  Scans the computer input keyboard device
*                            indicated by byte placed at >8374 (>00=entire key-
*                            board).  If a key has been pressed for the first
*                            time the STATUS byte byte (>837C) has bit #2 set,
*                            and the ASC code of the key is put in >8375 which
*                            otherwise holds >FF or ASC of last previous key.
*                            apparently disregarded  by assembler (for looks only.)
PABBUF EQU  >1000            Now PABBUF means the number >1000 to assembler.
PAB    EQU  >F80             Sets PAB=>F80 for AS.  (Abbreviation for assembler.)
*
STATUS EQU  >837C            See REF KSCAN for reason this assignment was made.
PNTR   EQU  >8356            >8356 is a location in the DSR stack routine area.
*
SAVRTN DATA 0                AS advances to next even (word) boundary,  places
*                            0 in any  byte skipped.  It then assigns the label
*                            (SAVRTN) to that location and puts 0 in it.
PDATA  DATA >0004,PABBUF,>5000,>0000,>0009  Stores the PAB data at a word
*                            boundary and calls its location ""PDATA"".  When
*                            interpreted as a PAB, entries mean (p. 293, E/A Man.):
*                            E/A Manual)
*                            >00 (word 0):  I/O code--0=open.
*                            >04 (word 1):  Flag/Status 4=00000100; first 3 0's
*                            are error code (0=no error), next 0 says "fixed
*                            length", next 0 "display" (vs. "internal"), next
*                            10="input", last 0 "sequential" (vs. "relative").
*                            PABBUF (>1000, words 2,3):  Data buffer address
*                            used in R0 in VMBW,VSBW,VMBR,VSBR.
*                            >50 (word 4):  Maximum length of record in file.
*                            >00  The number of characters actually to be
*                            transferred for a READ or WRITE I/O code.
*                            0 (words 6,7):  Record number (not needed for
*                            sequential files).
*                            0 (word 8):  Screen offset (use >60 if for cassette).
*                            >09 (word 9):  Length of file descriptor (next step).
       TEXT 'DSK1.DATA'      Places "DSK1.DATA" (9 bytes) at PAB+10 location.
       EVEN                  Makes sure AS puts next datum at a word boundary.
READ   BYTE >02              Puts 2 (the "read" code) in a location it calls READ.
CLOSE  BYTE >01              Same as above for close code.
*
```

```
MYREG   BSS   >20           Advances location counter 32 (=>20) spaces and
*                           Creates an empty space starting with "MYREG".
BUFFER  BSS   80           Same as above for "BUFFER" (can use base 10 also).
*
DSR     MOV   R11,@SAVRTN  SAVE RETURN ADDRESS.  (Since this is the entry
*                           point of the program its location, automatically
*                           stored in R11, is MOVed to permanent location SAVRTN.
*
        LWPI  MYREG         Moves workspace pointer to MYREG to load registers.
        LI    R0,PAB        Puts PAB (>F80) in R0.
        LI    R1,PDATA      Puts PDATA, location of the PAB, in R1.
        LI    R2,>20        Puts >20 in R2, the number of bytes to be trans-
*                           ferred by VMBW, the next step.
        BLWP  @VMBW        Moves PAB data into PAB in VDP RAM.
*
        LI    R6,PAB+9      Moves pointer to name length (loads PAB+9 into R6).
        MOV   R6,@PNTR     Stores pointer to name length in >8356.
*
        BLWP  @DSRLNK      Opens file (note:  pointer to text has been stored
*                           in >8356, PAB placed in VDP RAM.
        DATA  8             See REF step for explanation.
*
        MOVB  @READ,R1     Puts >02 in R1.
        LI    R0,PAB        Puts PAB in R0.
        BLWP  @VSBW        Changes I/O code to read (2 in first byte of PAB).
*
        MOV   R6,@PNTR     Restores pointer to name.
        BLWP  @DSRLNK      Reads one record (because I/O code is now 2).
        DATA  8
*
        LI    R0,PABBUF     Puts >1000 in R0 (the VDP RAM source buffer).
        LI    R1,BUFFER     Gives address of 80 blanks.
        LI    R2,80         Puts 80, # of bytes to be moved, in R2.
        BLWP  @VMBR        Moves CPU to buffer.
*
        LI    R0,>FE        Specifies beginning location (all locations before
*                           >300 are on screen).
        LI    R1,BUFFER     Location of bytes to be read.
        LI    R2,80         # of bytes to be read.
        BLWP  @VMBW        Moves  line to screen.
*
LOOP
*                           The above is a reference point to go back to.
        BLWP  @KSCAN       Waits for key press.  (What's in >8374?)
        MOVB  @STATUS,R0   Puts most significant byte in STATUS in R0.
        JEQ   LOOP          As long as it's 0, goes back to LOOP.
*
OVER    MOVB  @CLOSE,R1    Puts 1 (I/O for close) into R1.
        LI    R0,PAB        Puts PAB address into R0.
        BLWP  @VSBW        Changes I/O op-code to CLOSE (since 1 is now in PAB).
*
        MOV   R6,@PNTR     Restores pointer to name.
        BLWP  @DSRLNK      Close file.
        DATA  8
*
        CLR   R0            Fills R0 with 0's.
        MOVB  R0,@STATUS   So that no error is reported (cover-up?).
        MOV   @SAVRTN,R11  Moves return address to R11.
```

PRESIDENTS COMPUTER CORNER

The following TI Writer tips appeared in the Mid-Hudson UG newsletter in late 1985:

TI-WRITER Tips

Don't like loosing your FILENAME?

When you want to insert a file into the text you are working on, you don't have to overwrite the original filename! When accessing the LoadF, PrintF or SaveF editor commands, instead of typing over the filename, just press Insert (FCTN-2) and type in the filename of the desired file and then at least one space. This will "push" the original filename to the right, while keeping it intact (unless you push it past the edge).

When you go back to the SaveF, etc., simply Delete the inserted filename and you can then use the original filename(s) again without typing them in again, or better yet, trying to remember what it was!
Brett Kropf

Here's another TI-WRITER tip, this one's from Rich Lane of Upstate NY 99/4 U.G.

Formatter Command Reminders

When entering formatter commands in TI-WRITER there are some rules to follow. Except commands pertaining to text such as "^" (Required Space), "&" (Underline), etc., format commands may not appear on a line with text.

Generally, multiple format commands may be strung together on a single line, but must be separated by a semicolon (;) and a period must be the first character on a format command line, example: .FI;LM 4;RM 75;AD.

There are some special rules which must be followed, for example there are four commands that can only be placed at the end of a string of commands or else they must be placed on a separate line. These are the DP (define prompt), FO (footer), HE (header), and TL (transliterate) commands. The CO (comment) command must always be on it's own line. If an indent command is used it will be nullified if it was preceeded by a NF (no fill).

Would you believe, one more TI-WRITER tip?...
This is modified from the original article by George Lambert which appeared in the Spring 1985 TI BUG SOUTH newsletter...

Formatter Page Lengths

When you print files using the Formatter, you've probably noticed that there are lines reserved for a HEader and/or a FOoter. The original version used 5 lines at the top of a page for the header and 3 at the bottom for the footer. (The top and bottom lines of this newsletter use these features.)

The new version of TI-WRITER (the one that doesn't begin with a paper wasting form feed), fails to generate the two blank lines (line 1 and 2) on the first page and generates only one of them on all subsequent pages. This causes your text to begin on the fourth line on page one and on the fifth line on all others.

By adding the following procedure to your file you can regain the first blank line and gain the second for an additional line of text:

The asterisk (*) in the first .HE line should be replaced with a line feed character by keying in this sequence: CTRL U, SHIFT J, CTRL U.