German Character Set Wanted: Tony Klinkert (PSC #1 - Box 4785, APO New York, 09633 (Residence Schiersteiner Str. 18, 6200 Wiesbaden, West Germany, 011-49-6121-84-1143 [home]) wants to upgrade his SPET to German. Needs information on where to find PaperClip 9000B and a character set ROM or an already-created German character set printer-file for 8300P (Diablo 630) printer. Write or call.

CLOSEOUT ON COM-MASTER FOR $25.00

ISPUG distributes COM-MASTER, a telecom program and terminal emulator which handles the ASCII or APL character sets, text or binary (PRG) files in upload or download, and lets you set any telecom parameter you care to. You may also save versions of the program, tailored as you want them, for specific applications. For more information, see list of SuperPET software in this issue.

Because of the discontinuance of the Gazette, Dan Jeffers (the author) is offering COM-MASTER for $25 until existing stocks of manuals run out (we have only 34 left in stock). Order from ISPUG. The last of the wine, the last of the lace, the last of the splendid Paris dresses...

Indexed by Courtesy of Marilyn

The index to Volume II of the Gazette, appended to this issue, was created by Marilyn Post of Dillon, Colorado, the only SPETter (apparently) who plays tennis and also is unaware of the old Army rule that you never, ever volunteer (a rule, we add, that the majority of ever-silent SPETters seem to know by heart...).

Stan Cook, a mathematician from British Columbia (now an Amigan) writes that "I feel quite guilty for not providing anything for the newsletter--it's just that I felt it would not measure up to all the great stuff from your other contributors." Well, Stan, here's a secret: Our job was to edit; the writer's job, to set down the facts--which often came in a jumble, written on soggy tablet, misspelt, littered with the ruins of English. Editors are consigned to this planet to convert such stuff into readable articles--and to enjoy the process!

INSTALLING AND USING THE HIGH-RES TECHNOLOGIES GRAPHICS BOARD

by
Nick Solimene
87 - 27 94th Street
Woodhaven, N.Y. 11421

[Ed. Last issue, we reprinted an article by Tom Stiff on a high resolution graphics board useful on SuperPET. Nick Solimene got a board early on. Following is his report on how he installed and used it.] The board works very well indeed and adds quite a lot to the performance of SPET. The display area of 1024 x 512 pixels is impressive. Unfortunately, SPET's screen provides a window to only 700 x 270 pixels. The full area can be seen, however, by panning the window over the entire display.

Documentation: The documentation is a sparse 11 pages, including the instructions on how to install the board and a demo disk. Installing it in my SuperPET (three boards) was not as simple as I would have desired. The board is plugged into the 40 pin socket on the lowest SP board and the cable normally attached there is plugged into the hi-res board.

Installation: The problem is that the ROM towers at UD11 and UD12 are too tall. The instructions suggest that you insert a 40 pin socket to raise the hi-res board so that it clears the ROMs. This was clearly inadequate on my machine.

In particular, one tower was much taller than the other. I rebuilt that tower by soldering a low profile header and socket together and resoldered the attached resistor and the wires from the UD11 switch to the new tower. [The use of heavy braided wire (#14 ?) is beyond me.] Now, both towers are about the same height and about as low as possible short of their removal and the consequent disabling of the UD11 and UD12 switches.

Even with two 40 pin sockets (albeit low-profile), the hi-res board rests on the ROMs and is is not firmly socketed. Moreover, the reduced space above the hi-res board, caused by the extra sockets, made it impossible to connect the CRT cable without first bending the connector pins from vertical to horizontal. Although the present installation works, it isn't satisfactory. As soon as I can get some taller spacers to raise the second board, I will use a third 40 pin socket so that the ROMs will be cleared and the board will be firmly socketed.

Comments on Hardware: The board seems well-constructed and has eight 64K bit chips for the graphics memory, some logic chips and the CRT controller. Display parameters are set by poking to the registers of the CRT controller. Graphics memory is accessed via a one-byte window after poking a two-byte address. The hardware doesn't provide for synchronizing the graphics display to the normal character display. This is done in software. You may display and scroll both characters and graphics independently. Either or both may be turned on or off.

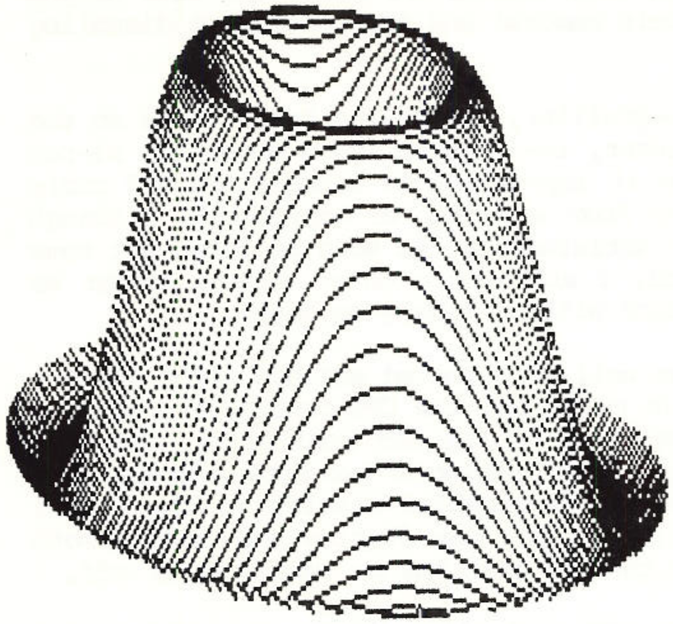
There seem to be some differences in the CRT controller parameters used on the two sides of the SuperPET and the documentation for the 6502 side. I wish more information had been provided on the CRT controller. It seems to have an interlace mode; I wonder if using it would double the number of pixels displayed to 700x512. Will write to Hi-Res Technologies for more information.

The Software: The software provided is rudimentary but sufficient to illustrate the capabilities of the hi-res board and how it may be programmed.

Some 6502 demos are provided. The ones I've looked at are in machine language with BASIC4 drivers. The main ML routine allows calls from BASIC4 using SYS 320xx to set window size and origin, fill screen (clears with 0), draw line between two points, reset parameters, adjust vertical or horizontal, or reverse field. The reverse seems to be missing in my copy or the instructions are wrong. The parameters are passed by using the locations of BASIC4 variables. It would have been nicer if these commands had been wedged into BASIC4. The source code isn't provided; the routines are partly described by giving BASIC4 equivalents using PEEKs and POKEs. There are some typos in the documentation; a critical OR is missing and a ')' was dropped. Looking at the 6809 source cleared this up.

For the 6809, there are some programs by Avy Moise. One group concerns the reconfiguration of OS/9 to use the graphics memory as a ram-disk. These I haven't looked at. There is also a graphics demo for use from the monitor, plus a demo in FORTRAN. The source code for the 6809 graphics demo is provided but it makes use of a macro named 'procedure' as well as one named 'call'. I have 'call' on one of the ISPUG disks but don't recall seeing 'procedure' anywhere previously. I'll try to get a copy. In any event, the .mod files are provided; you don't have to assemble to use demo.

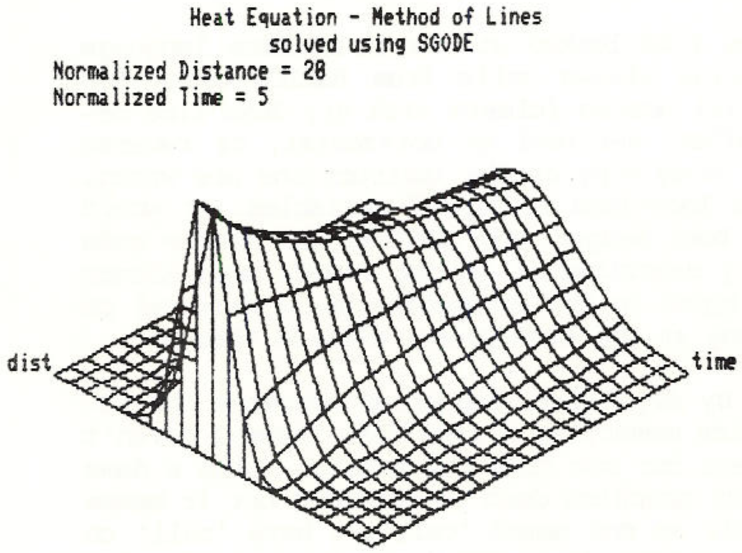
Except for trying some of the demos, I have done my graphics work using HALGOL, because of its speed and because the large memory of the 68K Grande board make the graphics most useful. I've added nine commands to HALGOL to handle graphics, which wasn't as difficult as I had expected.

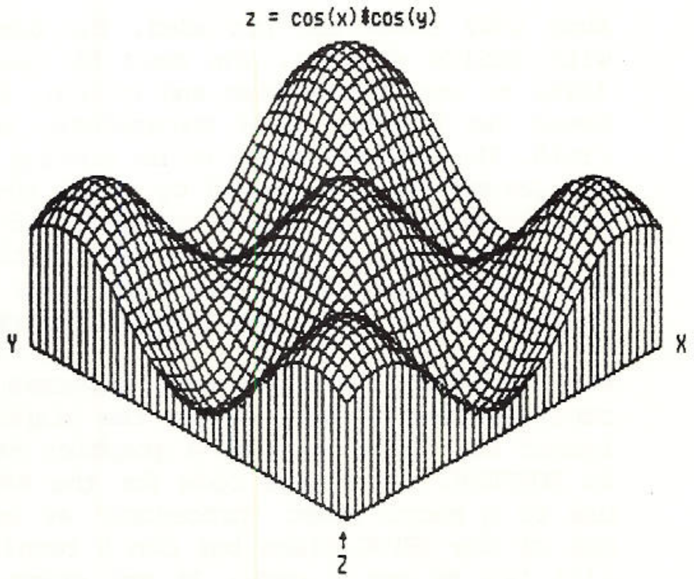


I converted the MTU hat program (see left) from an ancient ad by MTU to HALGOL and with some fiddling ran it in about 2.5 minutes; the original program was quite slow.

HALGOL does very well, and makes feasible display of 3D shapes of this variety (not necessarily figures of revolution) without having to wait forever.

The printer dump of the hat at left was done in BASIC4, using PEEKS and POKES of the graphics memory to build up data strings for my printer; the dumps are done sideways. Each graphics byte accounts for eight adjacent horizontal bits; dumping sideways therefore requires no bit manipulations. The dump rquires about six minutes. The demo disk has a ML dump for the 4022 printer, but it was far easier to use BASIC4 than to attempt to modify that program--especially without source available. Show below are two more plots as examples of what the board can do.





These were dumped directly from HALGOL by calling 6502 ML routines to manipulate the bits; run time is about two minutes instead of the six needed with BASIC4.

[Ed. If you're interested in the graphics board and require a tad of help, Nick says its okay to get in touch with him at the address above, but cautions that as he uses the graphics board, the DTACK GROUNDED board is needed as well as the Hi-Res Technology board, and that all his programs are written for modified HAL-GOL, now renamed as HBASIC.]

T H E   A P L   E X P R E S S   by REG BECK

Box 16, Glen Drive, Fox Mountain, RR#2, Williams Lake, B.C., Canada V2G 2P2

In the last column, we looked at direct disk access in APL but didn't have room for an application. The following functions will read in any sector on the disk, and produce both a hex and character dump. Unprintable characters are replaced by periods. The function SWAP lets us toggle back and forth between ASCII and APL character sets as we look at the character dump of the sector. SWAP 1 gives us ASCII, SWAP 2 yields APL. In Vol. II, No. 4, p. 104 we offered two functions for this purpose. These functions, APL and TXT, used the poke codes given in the Waterloo System Overview Manual and worked well except for differences in the punctuations between the character set we obtained and the normal ASCII set. We then read through back issues of the Gazette and found a note (Vol. I, No. 9, p. 115) which advised to SYS to location 45194 to get correct results. The arguments are 1 for ASCII and 2 for APL. READ was defined in the last column.

```
      DISPLAY 'DTH'
DTH:(16 48)ρ,((⍉('0123456789ABCDEF')[⎕IO+(2ρ16)⊤ω]),' ')
      DISPLAY 'CHARTODEC'
CHARTODEC:-⎕IO-⎕AVιω
      DISPLAY 'REMOVE'
REMOVE:¯1↓ω,ω[(ω∊⎕AV[1+(ι13),127,255])/ιρω]←'.':(⎕IO←1)=0:''
      DISPLAY 'SECTOR'
SECTOR:(DTH CHARTODEC READ ω),'|',(16 16)ρREMOVE READ ω
      DISPLAY 'SWAP'
SWAP:ω ⎕SYS 45194
⍝ITS NICE TO HAVE THESE IN DIRECT FUNCTION DEFINITION!  ⎕IO WAS MADE
⍝LOCAL ONLY IN 'REMOVE'.
```

We could have defined one function to convert directly from characters to hex but separated the two operations in the two functions DTH and CHARTODEC. REMOVE gets rid of the unprintable characters. All the functions are put together in SECTOR. The syntax used is SECTOR 18 1 to display sector 1 of track 18, for instance.

We have gone to some trouble in these functions to make them independent of the index origin. REMOVE was modified to make QUAD IO a local variable. More on this later in the column. In REMOVE, the conditional form was used to ensure an index origin of 1. The condition is looked at first which sets QUAD IO equal to 1. The condition of 1 = 0 can never be true so the alternate case will never result. A little trickery was used to make REMOVE result-returning since the form of replacement used (replacement of the unprintable characters with periods) modifies the vector but doesn't return it.

*   *   *

I recently came across an article by Howard Rotenberg in Commander, (Jan., 1983) titled Radix-50: Pack & Unpack. He discussed an old method used on the PDP-8 computer (8 instructions, 4K memory) for packing code. It converts three bytes into two bytes for a memory saving of one third. With only 4K available on the