TEX Support Programs

by
Phil Sherrod and Alan Wright
Vanderbilt University

The process of installing TEX on a computer system involves more work than just getting the TEX program itself running. It is also necessary to have support programs to convert the device *independent* output produced by TEX (the "DVI" files) into a device *dependent* form suitable for driving one or more output devices.

A DECSystem-10 computer is used at Vanderbilt which is very similar to the DECSystem-20 used at Stanford. This enabled us to get the SAIL version of TEX running with only a few days of effort. During the year that we have been using TEX we have found it to be remarkably reliable and bug-free and have had to devote very little time to its maintenance. Most of our effort has been directed toward improving the efficiency and convenience of the support programs used to drive the Versatec printer.

The Versatec printer is not connected directly to the DEC-10 but rather is connected by a parallel port to a Monolithic Systems Corp. Z80 micro-computer system which consists of a Z80 processor and 64Kb of memory. This system serves as an intelligent controller for the Versatec and unloads a significant amount of processing from the DEC-10. The Monolithic micro is connected to the DEC-10 through a 9600 baud RS232 serial line.

The support programs provided by Stanford for driving a Versatec consisted of two DEC-10 programs written in SAIL, DVIVER and VERSER, and an assembly language program for the Z80 called PAINT. We obtained a cross assembler for the Z80 from Stanford and wrote our own program to communicate with the ROM monitor in the Z80 and downline load program images.

The DVIVER and VERSER programs were moderately large (about 60K 36-bit words) and fairly slow, consuming a total of about 7 seconds of CPU time per page generated (TEX only uses about 1.5 seconds of CPU time in formatting each page).

The DVIVER program reads the DVI files produced by TEX, breaks tall characters into "parts" which will fit in the raster-scan buffer area in the Z80; sorts the parts on each page into top-to-bottom order and produces an intermediate work file that is read by VERSER. VERSER reads this work file and produces a file of commands and data to be transmitted to the Z80. The actual process of generating the raster scan pattern for each character and part takes place in the Z80. However, since the Z80 has no local storage, VERSER must downline load the raster pattern definition for each character as part of the command and data file it produces for a document. Once a raster pattern has been defined for a particular character the invocation of it requires only a simple and short command.

The total number of character pattern definitions that can be simultaneously held in the Z80 is constrained by the available memory space in the Z80. In the original implementation of the Z80 PAINT program, each character (or part) definition occupied a fixed size of 256 bytes in Z80 memory. The VERSER program kept track of which character definitions were currently defined in Z80 memory and used a round-robin technique to replace old character definitions with new ones if more characters were used than could be simultaneously accomodated by the Z80. This meant that any number of characters could be used in a document but a character definition might have to be reloaded several times.

Our work at Vanderbilt on the TEX support programs has been directed at improving their efficiency and convenience. The DVIVER and VERSER programs were combined into a single program called VERTEX which was written in BLISS for efficiency. VERTEX performs the same basic functions as DVIVER and VERSER but is more efficient because it does not have to generate the intermediate work file and can avoid having to reaccess font information that was used by both DVIVER and VERSER. Its memory management is also more efficient and it uses a least-recently-used technique to keep in memory information about fonts. The VERTEX program occupies 33K words of memory and uses about 5 seconds of CPU time per page generated.

The Z80 PAINT program was also rewritten. The memory space occupied by instruction code was reduced to 30 percent of its original size allowing more space to be used for character pattern definitions. The space allocated for each character pattern definition was reduced from 256 to 128 bytes. This resulted in more characters having to be broken into parts but substantially increased the total number of characters that could simultaneously be defined in Z80 memory. A least-recently-used (rather than round-robin) technique is used to select character definitions to be replaced when more characters are needed than can be defined simultaneously in Z80 memory. The command protocol used to communicate between the DEC-10 and the Z80 was also improved. A reinitialization command sequence was added that allows the Z80 to be restarted under DEC-10 control; status information about paper and toner supplies are sent from the Z80 to the

DEC-10.

The last part of our project was to integrate the TEX support programs into the spooling system on the DEC-10. We did this by adapting VERTEX to fit into the GALAXY spooling system which is part of the operating system on the DEC-10. A time-sharing user can now issue a "VPRINT dvi-file" command to queue a file for printing on the Versatec. The conversion of the DVI file to Z80 commands is performed by the spooler; the Z80 commands are transmitted directly to the micro-processor without creating any intermediate files.

As part of our rewrite of the PAINT program, we added commands to do general vector drawing. A DEC-10 program has been written to display on the Versatec plot files created in a format compatible with our Zeta pen plotter. We intend to modify TEX to add a command which will specify the name of a plot file and the amount of space to reserve for it. VERTEX will then merge the plot commands with the document as it is being processed for printing on the Versatec.