Behavior Research Methods & Instrumentation
1982, Vol. 14 (2), 92-98

SESSION IV SMALL MICROCOMPUTERS IN THE LABORATORY

An inexpensive CRT terminal controller/computer

WILLIAM L. PALYA, JULIO ORTEGA, and JERRY B. WILSON
Jacksonville State University, Jacksonville, Alabama 36265

An inexpensive CRT terminal controller/computer is described. The board integrates virtually any keyboard and CRT monitor into a work station with firmware-governed intelligence. The system is designed to fill the cost/functionality niche below the Apple microcomputer. It can be configured as a dumb or smart terminal (approximately $100 plus keyboard and CRT monitor), as a self-contained computer that runs ROM-based software such as a small BASIC or an assembler/disassembler (an additional $100 in chips), or as a word processor if an audio cassette ($50) or dual digital cassettes with controller ($125) are added.

We developed a small microprocessor-based CRT terminal controller/computer (CRT) in response to a growing need for several types of work stations. Dumb terminals were needed to interact with a general-purpose computer. Microprocessor-based prototyping stations were needed for an instrumentation course and general hardware development. Simple real-time laboratory controllers were needed to replace relay racks. Small computers were needed to provide students an introductory experience with programming. And there was the inevitable demand for word processors.

We needed a system that fit the cost/performance niche below an Apple/TRS-80 class computer. Figure 1 roughly illustrates the cost/performance functions for various computers. For example, a PDP-11 has some initial cost, which includes hardware, software, and user time. It can be indicated by the point at which the PDP-11 line intersects the ordinate. Up to a point, the PDP-11 can perform tasks of increasing complexity with little or no increase in cost. However, as the tasks become increasingly complex, the cost increases, and eventually, it sharply increases. This general function is similar for computers that are either more or less expensive. The difference is that the sharply increasing portion of the cost/performance curve is encountered at different levels of functionality. Performance or functionality in this sense includes ease of use, speed, and capacity. The important point illustrated by the figure is that for tasks that are less complex, a less expensive computer is generally the more cost-effective solution.

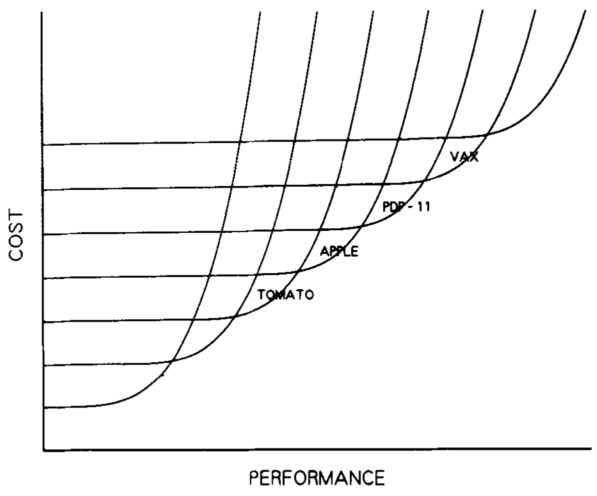

Figure 1. Hypothetical cost/performance functions for various computers. The point illustrated is that there are various cost/performance niches that are best filled by different machines.

It is well accepted that a PDP-11 provides more functionality than is necessary for simple tasks and is therefore not cost-effective when compared with an Apple-class computer in those situations. In our situation, the Apple-class computer provided more functionality than we could afford.

Several circumstances formed the context within

The authors gratefully acknowledge the contributions of Elizabeth Palya in all phases of producing this system and manuscript. P. Johnson noted an error. Requests for software or reprints should be sent to William L. Palya, Department of Psychology, Jacksonville State University, Jacksonville, Alabama 36265.

which our decisions were made. (1) Because the expertise, labor, and interest to build these work stations were available and the money to purchase commercial units was not, we chose to build our own. (2) The design is a compromise among the optimum systems for each of our needs. The board can have more than is needed for a dumb terminal, and the board provides less than ideal functionality as a machine to run BASIC. (3) We had earlier standardized on 6800-family microprocessors because of their similarity to our PDP-11; therefore, only 6800-based systems were considered. (4) We chose to sacrifice the ability to provide color displays because one-chip video controllers provide either high-density black and white (e.g., MC6845) or low-density color (e.g., MC6847) and because we felt that a higher density display capable of providing 80-character text lines was minimally acceptable. Also, we did not have enough color monitors or the money to purchase them. (5) The project had been precipitated by the availability of controllerless VT52 look-alikes for less than $100 each. As a result, the controller design conformed to the limits set by them. And finally, (6) we chose to implement the design as a printed circuit board because it would be less expensive when spread over 10 work stations.

The controller/computer evolved from a CRTC design by Melear and Browne (1979). We added serial and parallel I/O ports and the capacity for the microprocessor to function as a general-purpose computer. We also modified some commercial software to run on the work stations and to suit our needs better. The software consists of the MIKBUG monitor, a small integer BASIC, and a direct-execution assembler/disassembler.

COMMERCIAL HARDWARE

We used surplus Sander's 860 video terminals. These terminals are attractive and well-constructed. They contain an 18-MHz monitor (sufficient for 80 columns by 24 rows of characters), an 89-key keyboard, a 5-V, 5-A power supply, and enough additional space to mount an 8.5-in. disk drive. The keyboard is a Keytronic magnetic reed-switch full keyboard, with LEDs and sufficient additional keys to allow special-purpose function keys or a quasi-VT-100 emulation.

The keyboards provide 8-bit parallel ASCII output, but unfortunately, some keys were not well located. We rearranged the keyboard by cutting tracings and soldering jumpers to the appropriate code lines. The changes took approximately 2 h. This is a very simple operation and is well worth the effort to change any keyboard that is poorly designed. Most keyboards generate the appropriate ASCII code for a particular keypress by shorting a "vertical" or last n-bit line and a "horizontal" or first n-bit line with the contact closure. Figure 2 illustrates a typical key-decoding schematic. As can be seen, any key position can be made to provide any code by simply cutting the existing connections and then jumpering to the desired horizontal and vertical lines. If a schematic of the keyboard is not available, correct

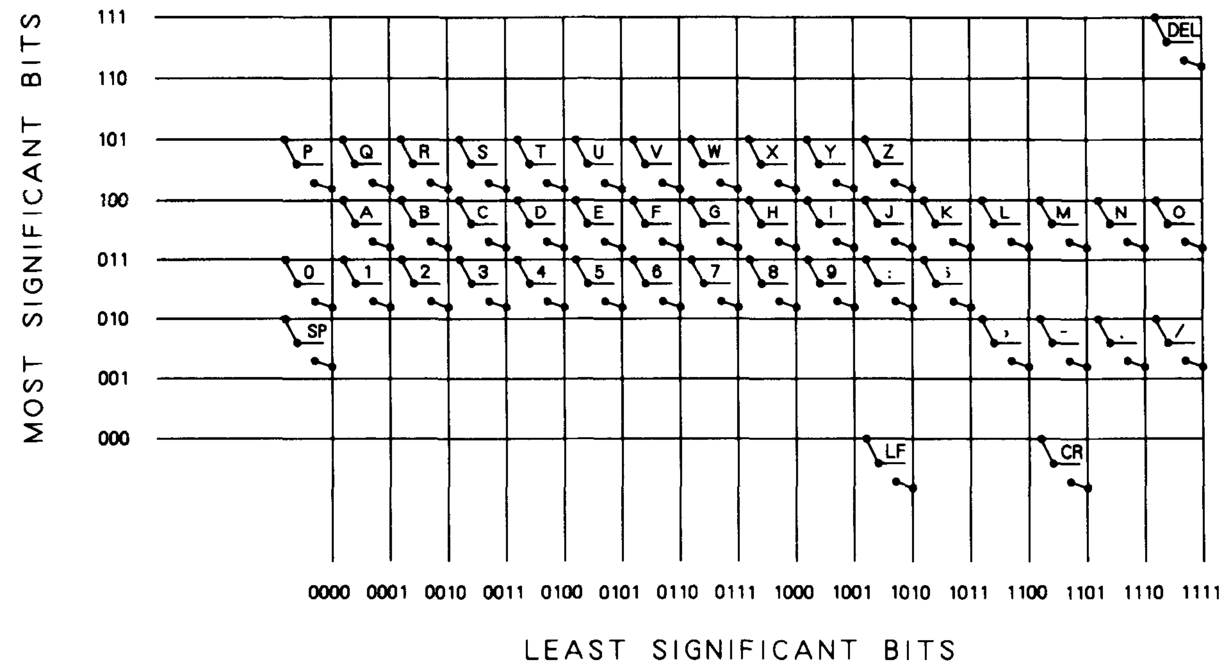

Figure 2. A typical key decoding schematic. Each key shorts a "most significant bits" line with a "least significant bits" line to generate an ASCII code.

decoding can usually be determined by observing which tracings are common to which key positions and then looking up the common element in their respective ASCII codes.

CUSTOM HARDWARE

Figure 3 provides a block diagram of the controller. It is a 6808 microprocessor with 12 KB of read-only memory (ROM) for firmware and 4 KB of user random-access memory (RAM). Two parallel (PIA) and two serial (ACIA) ports are available. The keyboard uses one PIA. The other can be used for general-purpose I/O or a parallel cassette interface. RS232 communication to the host computer uses one of the serial ports. The other ACIA can be used for a serial printer, a simultaneous link to a modem, or an audio cassette.

Characters can be inserted into the 2-KB (80 by 24 characters) display RAM by the 6808. These characters are then displayed on the screen by the CRTC. The CRTC completely handles the video display. It is a set of programmable counters and comparators, which correctly step through both the display memory and the character generator addresses, keeping track of programmed screen configurations and horizontal and vertical retraces. In addition, it provides vertical and horizontal sync signals, light-pen support, programmable hardware scrolling, various interlacing modes, and direct cursor addressing. Each of the display characteristics is under firmware control and can therefore be easily tailored to specific needs.

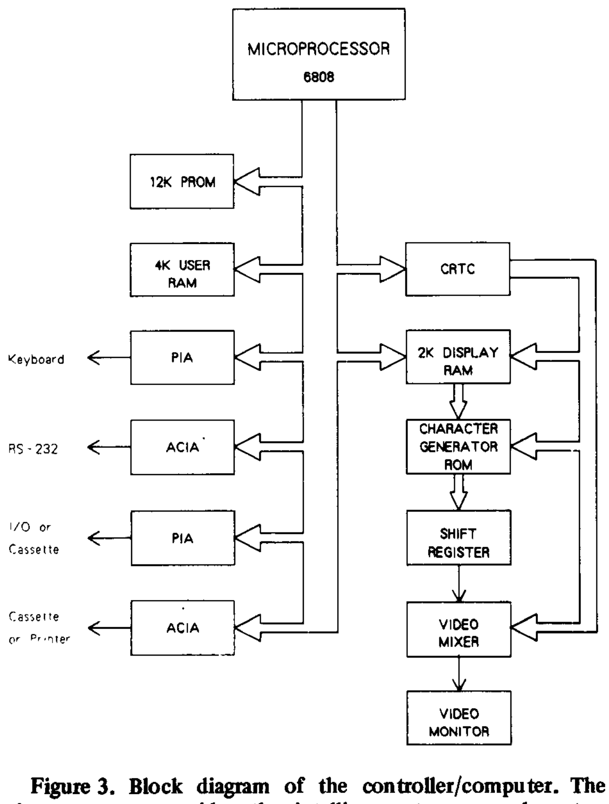

Figure 3. Block diagram of the controller/computer. The microprocessor provides the intelligence to move characters between the keyboard, the RS232 line, and the screen. It also provides for limited general-purpose computing.

The character generator produces the appropriate dot pattern for a character in parallel and passes it to a shift register, in which it is transformed into a serial bit stream appropriate for the dot pattern required for the screen. An analog switch adds the horizontal and vertical sync with this dot pattern and thereby produces a composite video signal for a standard monitor.

Figure 4 provides the schematic diagram of the controller board. It is laid out to roughly correspond with the photomask provided in Figure 5. The 6808 microprocessor and its address, data buffers, and crystal can be seen in the upper left of the figure. The PROMs appear to the right of the 6808 and the user RAM is in the upper right corner of the figure. A 74154 4-16 decoder (U15) and a 74139 dual 2-4 decoder (U16) are just below the PROMs. The 4-16 decoder segments memory into 16 4-KB partitions, and the 2-4 decoder segments the first 4-KB block into 4 1-KB segments to properly enable the 1-KB 2114 memory chips.

The video portion of the board appears in the lower right part of the figure. The 6845 video controller (U34) is responsible for correctly presenting on the CRT screen the ASCII characters that are stored in the display RAM. The 74157 multiplexers (U35, U36, and U37) located just below the CRTC allow either the 6808 or 6845 to have access to the display RAM address lines. Data communication between the CPU and the display RAM is gated through a 74645 transceiver (U41). The 2 KB of display RAM can be seen in the lower right corner of the figure. Characters to be output from the display RAM to the CRT screen are latched into a 74374 (U39). This information is then used as part of the address provided to the 6674 character generator (U38). The current row of the current character being displayed is determined by the 6845 and is used as the remainder of the address. The dot pattern appropriate for the row of that character is then transferred to a 74165 parallel-in/serial-out shift register (U23), which serializes the information. The cursor information is synchronized with the remainder of the video by U20. U18 shapes this summated video signal for input to the CD4066 analog switch (U17), which adds the vertical and horizontal sync signals with the video to produce the "composite video" signal required by CRT monitors.

The 6821 parallel ports (U24 and U25) can be seen in the lower left of the figure. The 6850 serial ports (U48 and U49), with their 1488 (U10) and 1489 (U11) RS232 drivers, can be seen directly above them. The baud-rate clock, divider, and baud-rate selector (U6 and U7) can be seen to the right of the serial port.

Figure 5 provides the component side and solder side photomask. The main bus passes across the top

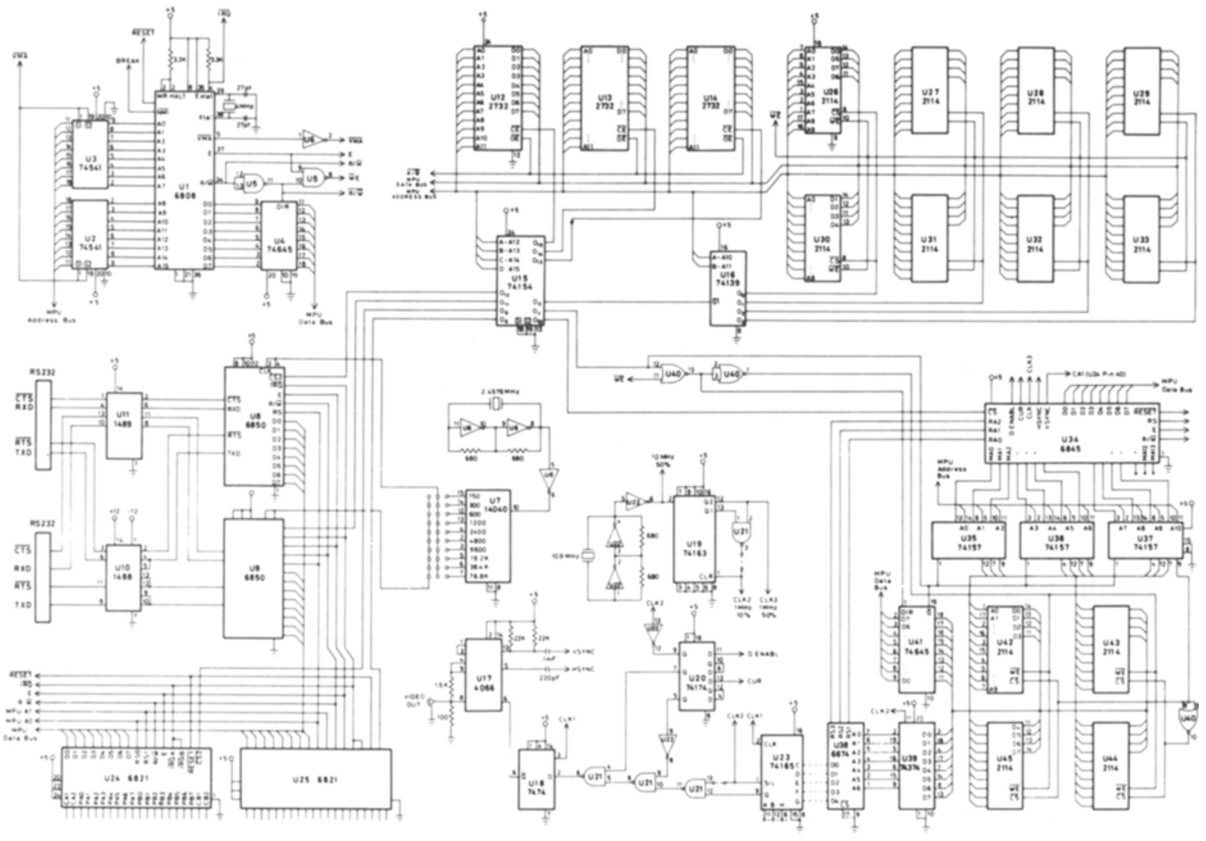
Figure 4. Schematic diagram of the controller. Chips are laid out to roughly correspond to their positions on the photomask.

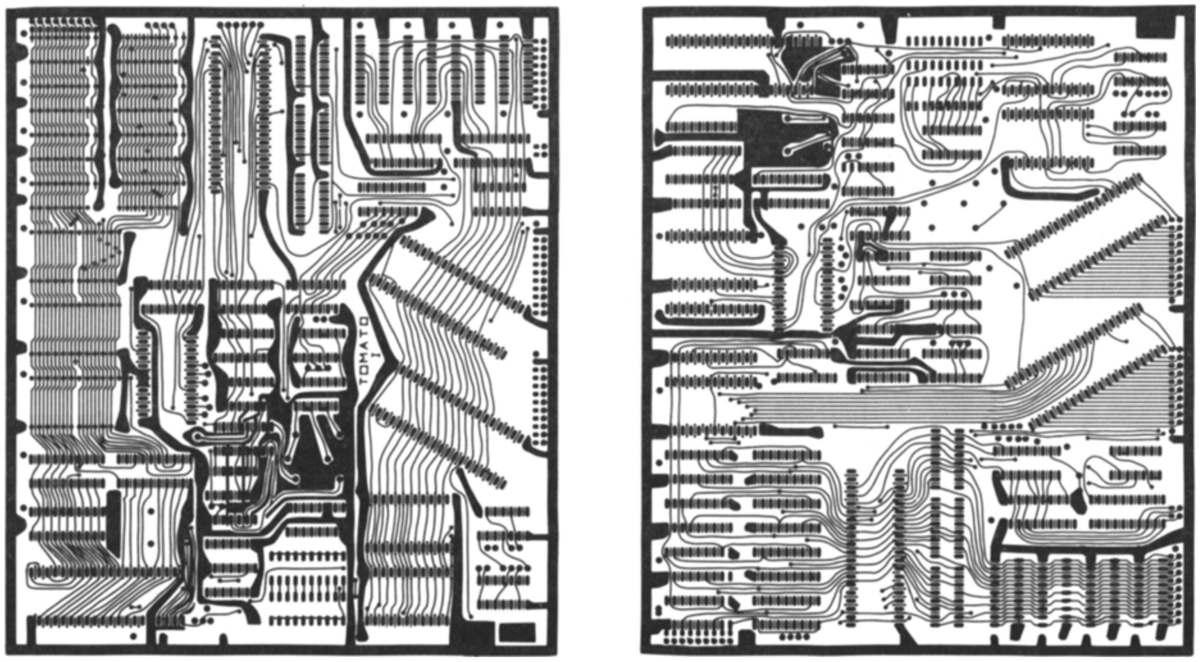
Figure 5. Components side and solder side photomasks of the controller board.

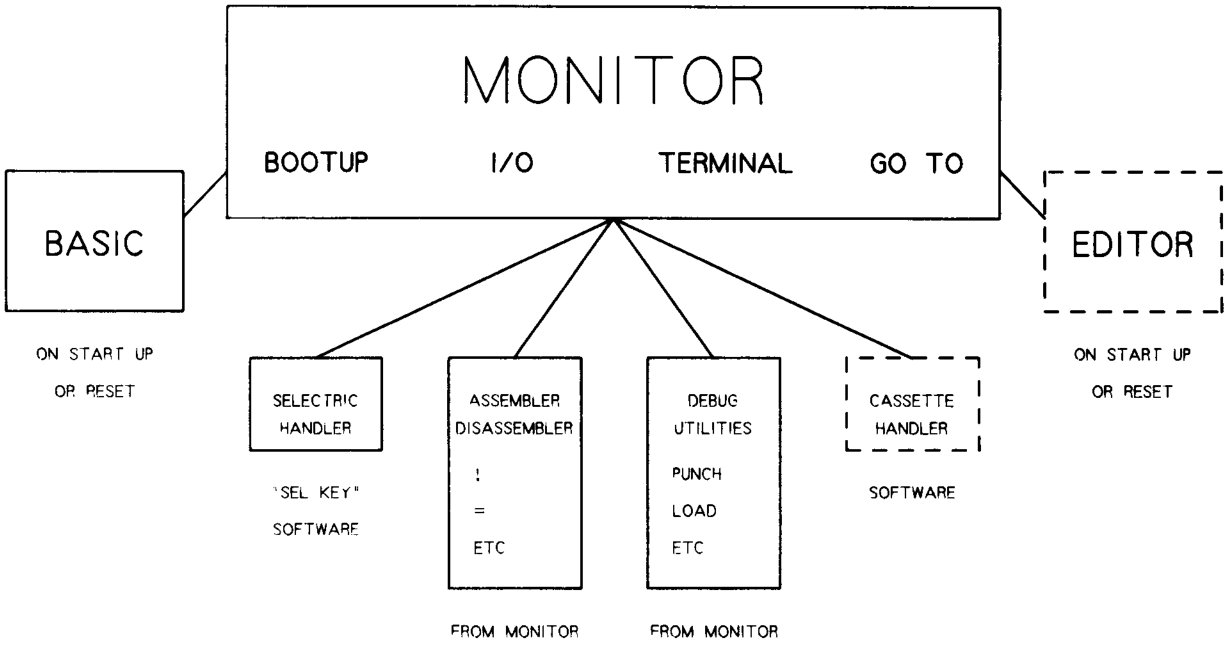

Figure 6. General architecture of the controller's software. On reset, BASIC (or the word processor) is invoked. The monitor can be called from BASIC. Subsequently either monitor debug commands or the assembler/disassembler can be used.

of the board on the component side from the 6808 (40 pin chip) on the left through buffers (three 20-pin chips) to the PROM (three 24-pin chips) and the user RAM (eight 18-pin chips) on the right. The address lines continue on the solder side. They descend on the right side of the board (which appears on the left side of the photomask) between the pins of the CRTC (40-pin chip) to the display RAM multiplexers (three 16-pin chips), and finally, to the display RAM (four 18-pin chips), which is located below the multiplexers. The control lines and data path to the serial (two 24-pin chips) and parallel (two 40-pin chips) ports located across the bottom of the board can be seen descending to the left (appears to right) of the 6845 on the solder-side photomask. Their interconnecting bus can be seen on the component side. The data path between the 6808 and the display RAM uses these same data lines. The remaining chips in the center of the board are the decode, video, clock, and other housekeeping circuits. Additional ground plane can be seen below the CPU, video, and baud rate crystals and around the video circuitry. This additional ground plane was added where possible to increase noise immunity.

A board can be fully populated in about 3 h. However, the board need not be fully populated. As a minimal system (such as a dumb terminal) the 6808 can be replaced by a 6802 microprocessor that contains 128 bytes of integral memory. This allows all of the user RAM to be deleted. A 1-KB 2758 PROM can be used in place of the more expensive 2732 PROMs, in that only a very short program is necessary to produce

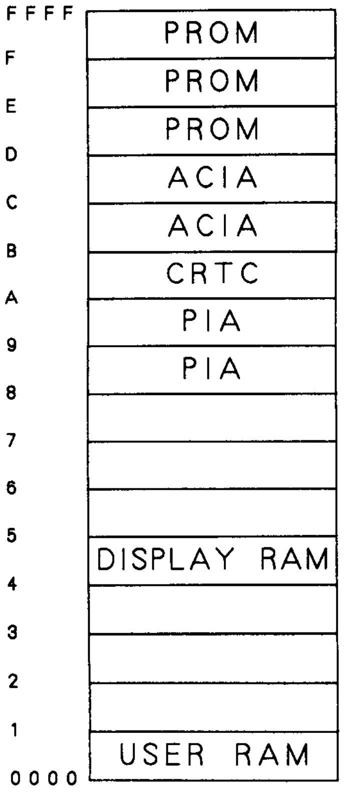

Figure 7. System memory map. The memory space is segmented into 16 partitions. User RAM is in the lowest and display RAM is in the fifth partition. The highest three segments are reserved for firmware, and the input/output controllers are located in the five lower partitions.

Table 1
TSC Micro BASIC Plus

Instructions	Variables	Commands	Functions	Relations	General
LET	ONE LETTER	SCRATCH	SGN	= <=	Multiple Statements per Line
READ/DATA/RESTORE	TWO SUBSCRIPTS < 98	RUN	ABS	< >=	Line Editing
INPUT	EXPONENTS < 99	LIST	TAB(X)	> <>	Integer Arithmetic Only
PRINT		MONITOR	SPC(X)		
GO TO		BREAK			
GOSUB/RETURN					
ON GOTO/ON GOSUB					
IF THEN					
FOR/NEXT					
DIM					
REM					
EXTERNAL					

a dumb terminal. The sockets, bus drivers, and one PIA and ACIA can also be deleted. This reduces the cost of the controller from about $200 to about $100. If desired, a Selectric typewriter can be directly interfaced for approximately $12 by incorporating the design described in Ortega and Palya (1982). Only the relay drivers and a PIA need be added to the minimal system. All remaining components including space in the 2758 for the firmware are already available.

SOFTWARE

The general architecture of the system is illustrated in Figure 6. The monitor is a modification of Motorola's MIKBUG and contains routines to boot up, implement I/O, provide a CRT terminal facility, and begin execution at specified locations. On start-up or reset, the monitor transfers control to BASIC (or the word processor). The Selectric printer software can be activated at any time by depressing the special function key "SEL" or through a software call. Each character subsequently output to the screen is printed to the Selectric. If a user wishes to use the assembler/disassembler or one of the prototyping debug instructions, the monitor is first entered from BASIC with the MONITOR command. Subsequently, the assembler/disassembler can be entered or one of the debug commands can be executed.

Figure 7 provides a system memory map. Memory is segmented into 4-KB partitions because it required only one chip to decode it that way. User RAM occupies the lowest 4 KB. Display RAM is the bottom half of the fifth 4-KB segment. It is not completely decoded, and therefore, it also appears again in the top half of this block. The PIAs, ACIAs, and CRTC are also incompletely decoded. Each echoes throughout an entire 4-KB block of memory. The top three 4-KB segments are occupied by the PROM firmware.

We obtained an inexpensive 4-KB integer BASIC with sources from Technical Systems Consultants. Table 1 provides the instructions and notable elements of this

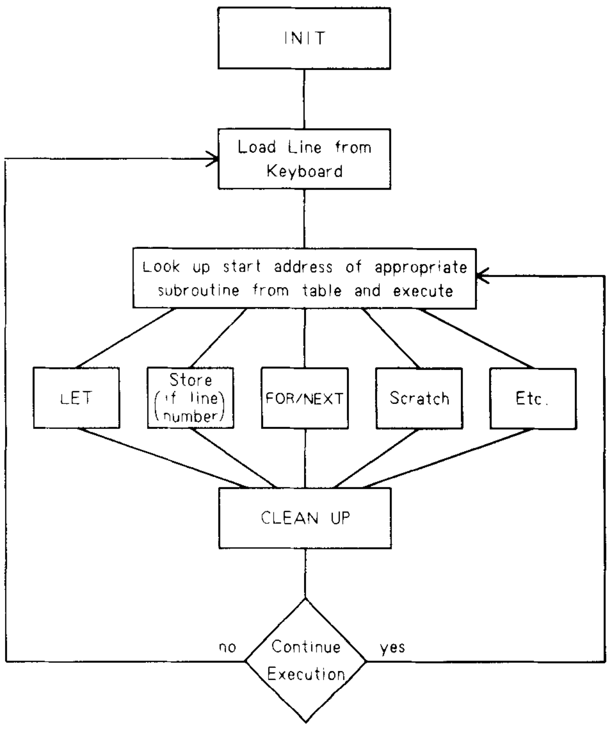

Figure 8. Global architecture of the BASIC implemented on the present system. Characters are loaded into a buffer from the keyboard. At carriage return, BASIC invokes the appropriate subroutine and either executes or stores the line. If only that line was to be executed, the program returns to the routine that loads characters from the keyboard; otherwise, execution continues with the next line in storage.

BASIC. It provides a reasonable first exposure to programming, as well as useful logic control for a prototyping system or process control.

Figure 8 provides the global architecture of BASIC and illustrates its simple design. Instructions can be added in two ways. BASIC can be reassembled after a subroutine is included to implement the desired function, and its name and start address is appended in the table of acceptable instructions. Alternatively, an external

Table 2
Micro Assembler/Disassembler

MA/D	Prompts with address of current location of memory pointer Advances pointer appropriately after each question Calculates relative branches (absolute location of destination is entered) Errors are reported immediately following a carriage return	
Prompt	User Response	Function
XXXX:	Carriage Return Only	Prints disassembled code
XXXX:	Instruction Mnemonic	Assembles, enters it
XXXX:	'A (Quote Character)	Enters ASCII value
XXXX:	"ABC (Double Quote String)	Enters ASCII string
XXXX:	$NNNN	Disassembles and prints NNNN lines
XXXX:	!	Begins execution at pointer
XXXX:	!NNNN	Begins execution at line NNNN
XXXX:	=NNNN	Dump NNNN bytes following pointer in hex
Editing		
XXXX:	&YYYY,NNNN	Move the NNNN bytes starting at YYYY to the locations following the current pointer

Table 3
Prototyping Debug Instructions

P	Punch in S1-S9 format
L	Load in S1-S9 format
D	Dump memory from XXXX to YYYY
M	Print and open a memory location for modification
R	Print registers
S	Single step: advance one instruction, print registers
B	Set a break point
U	Clear a break point
↑ A	Open A register for modification
↑ B	Open B register for modification
↑ X	Open X register for modification
↑ P	Open program counter for modification
↑ F	Open condition codes for modification

machine language subroutine can be called when the "external" instruction is executed. Control is passed back to BASIC by executing a return from subroutine instruction in the machine language subroutine.

We also use a 2-KB ROM-based immediate execution assembler/disassembler from AMI. It is very convenient for short debugging programs because instructions are entered directly without the need for compilation or downloading from a host computer. Table 2 provides the instructions and notable elements of the assembler/disassembler. It accepts standard 6800 mnemonics and assembles and enters them, as well as accepting and entering hex code or ASCII characters. It will also disassemble and list machine language as standard mnemonics. The contents of memory can be moved or dumped as formatted hex, and program execution can be conveniently started at any point.

The monitor provides debugging functions that are useful for solving problems with either hardware or software. Table 3 documents the available debugging or prototyping commands. The monitor provides support for up- or downloading programs with a host computer through an RS232 serial port using the Motorola 6800 S1-S9 protocol. It also provides for opening and modifying the registers and memory. Breakpoints can be set at any location, and by using the S command, programs can be executed one instruction at a time.

REFERENCES

Melear, C., & Browne, J. Simplify video-display design by using a versatile IC controller. *Electronic Design*, 1979, **13**, 94-102.

Ortega, J., & Palya, W. L. A simple IBM I/O Selectric typewriter controller. *Behavior Research Methods & Instrumentation*, 1982, **14**, 99-102.