A SATELLITE DISPLAY CONSOLE SYSTEM FOR A MULTI-ACCESS CENTRAL COMPUTER

WILLIAM H. NINKE
Bell Telephone Laboratories, Inc.
Murray Hill, N. J., USA

Abstract: A new system is described for attaching many responsive graphical display consoles to a single multi-access central computer. The key component in this system is a new console, the GRAPHIC-2. The console includes a small general purpose computer and a specially designed display processor. Only low bandwidth communication is required between the central computer and a console so a console can be located remotely and connected via a voice-grade line. Combined hardware-software solutions to problems of the console system are presented.

INTRODUCTION

Two pioneering systems, DAC-I (1) and Sketchpad (2), have demonstrated the effectiveness and desirability of graphical input-output computer communication via a cathode-ray tube display console. Since the advent of these single console systems, work has been going on to design configurations that allow an increased number of consoles to be attached to a large central computer, minimize the hardware and operating costs of each console, and allow remote location of consoles from the computer. The constraints generally applied to this work have been that a user at each console should possess at least the problem-solving power and response times to input device manipulations provided by the early systems. Under these constraints configuration design has indeed been difficult.

An early configuration was the attachment of several consoles to a large time-shared computer (3, 4). The CRT's were refreshed and console input devices were monitored by the computer. The direct connection of even a single console, however, caused excessive operating expense for picture refreshing. The time-sharing system provided slow responses to console input device manipulations. During console usage, degradation of service to other computer system users also resulted (5).

In a later configuration, multiple consoles were attached through a controller to a computer. The controller stored pictures received from the computer on a drum memory and refreshed pictures without further computer aid. Console input devices were monitored by the computer. Dynamic subroutine loading and memory management were used to allow several consoles to be served concurrently. With this organization, three consoles were successfully attached (6). However, for this number, total dedication of the computer was required, and actions at one console interfered with response times at others. Remote console location beyond a few hundred feet from the computer was not possible.

A third approach was to interpose a small general purpose computer between the central computer and the display console (7, 8). The small computer, located at the display console, provided rapid responses to console input device manipulations and refreshed the picture out of its memory. It also accummulated work it could not perform for subsequent action by the central computer and supervised communications back and forth between the console and the central computer. The central computers were operated in batch mode with console access between batch jobs. The size of the problem that could be solved at a console was usually limited by what could be contained in the console memory at one time. For larger problems, the slow response to requests to update the console-contained piece of the problem made problem solving difficult. For these satellite systems, the central computer was totally dedicated during the batch slot allocated to a console so that high data transfer rates between the console and the central computer were desirable. Thus, these systems had to be located close to the central computer to achieve reasonable communications costs.

THE GRAPHIC-2 SYSTEM ORGANIZATION

Based on experience with the GRAPHIC-1 console system (7) (which is of the type described in the previous paragraph) and on a recognition of the problems of the other systems just outlined, a new display console system has been created. Specifically, the system consists of several satellite GRAPHIC-2 consoles attached to a large multi-access computer. Satisfactory performance is achieved with only voice-grade communication service (2000 baud) between the large computer and each console. Each console contains a small computer and a display; the configuration is shown in Fig. 1. The multi-access central computer currently is a GE 645 operating under the GECOS-II system.

The overall cooperative operation of a GRAPHIC-2 console and the central computer can be summarized as follows: The central computer contains a structured data base which describes a problem. The console computer also has a structured data base, the two data bases being similar though not identical. For example, greater precision in the storage of numbers and different linking conventions are used centrally. Separate programs reside in the central and console

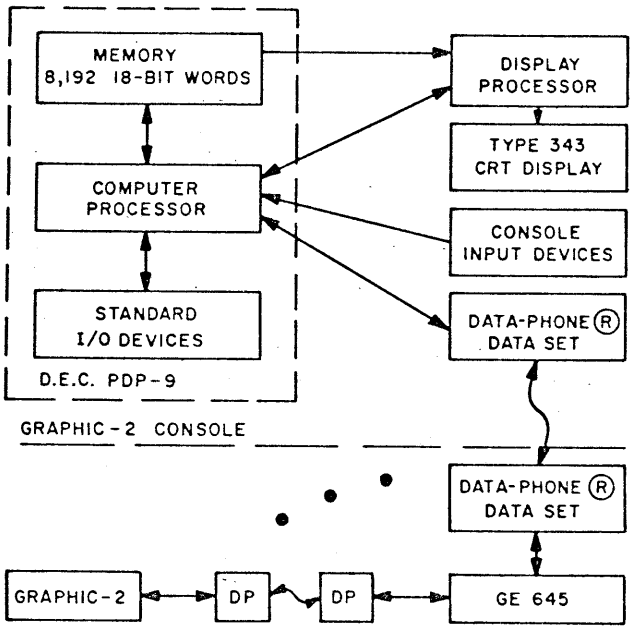

Fig. 1 GRAPHIC-2 SYSTEM ORGANIZATION

computers to deal with this structured information. These programs have been assembled from a single problem description. The programs differ in detail but cooperate to execute the common problem. The operation of these programs is carefully interlocked.

A user starts a problem by employing a bootstrap program to call for data and programs to be loaded from the central computer into the console core memory. Of course, if a problem is being formulated little or no initial data may exist. The user then employs the console input devices to work on his problem. Through an interrupt-driven monitor, the console computer acts on signals received from the devices to direct control to appropriate servicing programs which quickly perform most manipulations on the data. The display immediately reflects these manipulations so that a user has rapid visual feedback of his actions.

A history of console actions is transmitted to the central computer where the manipulations specified by the history are duplicated by the central computer program on the central data base. Because of this duplication, the history is highly encoded, i. e., it contains only end-point information of involved operations. A user has already received rapid response from the console computer, so there is generally no need for the central computer to act rapidly on this history. It can be transmitted to and processed by the central computer in small blocks as a problem is being worked on. Hence, much less information with less stringent response time demands passes from a GRAPHIC-2 console to the central computer as compared with a directly coupled console to its supporting computer and low bandwidth transmission to the central computer is satisfactory.

While the composition and editing of problem data can be easily handled by the console computer, large scale processing cannot. Large processing tasks, such as computing the frequency or time response of a circuit composed at the console, are directed to the central computer. The results of central computations are transmitted to the console for display and/or further work. Since the large scale computations usually take significant time, the extra delay caused by transmitting it over low bandwidth lines is not noticed by a console user (9). Therefore, low bandwidth communication is also satisfactory for transmission to the console. Hence, a console can be located remotely and connected to the central computer via an economical dialed-up voice-grade line.

The size of the memory in the GRAPHIC-2 is limited. Therefore, provision is made to deal with an appropriate piece of a problem if the problem is too large to fit totally into the console at one time (10). The console thus represents a dynamic window on a large problem surface. This dynamic quality is essential because, in man-computer problem solving, the course of action is man-directed (11). This course cannot be anticipated, so provision must be made for the problem to follow the man's directions.

In review, the GRAPHIC-2 system operation is, in most cases, one of duplicate processing. Tasks requiring rapid response are done whenever possible by the console computer. These same tasks are performed in the central computer, along with more complex tasks. So, remote console location with low bandwidth connection has been achieved at the expense of the duplicate processing. However, the console real-time tasks are treated in the central computer as blocked inputs without stringent response requirements. Multiple consoles are thus easily and efficiently handled by a central computer with a minimum of interference to other central computer users.

Let us now look at some problems which have required special consideration in the GRAPHIC-2 operation. As previously mentioned, both the local computer and the central computer contain structured data bases. The ability to structure picture information is an absolute necessity in any display console. If names and hierarchical interrelations cannot be represented and manipulated, the console user is severely limited in the problems he can attack. It might be mentioned also that structured information goes hand-in-hand with organized user thought processes. Manipulating and displaying structured information for a large problem is particularly difficult in a satellite console since memory space is limited.

A semi-interpetive mode of operation of the GRAPHIC-2 console which allows the direct display of the structured information is dependent upon special hardware features of the console. Before this mode is explained,

therefore, the console hardware will be described.

THE GRAPHIC-2 CONSOLE

The GRAPHIC-2 console is a shared memory multi-processor. One processor is that of a small general purpose computer; the other, a specially designed display processor. The computer, a D. E. C. PDP-9 (12), houses the 8, 192 18-bit word memory which serves both processors. The PDP-9 has as standard input devices a teletypewriter, a paper tape reader and punch, and a 60 Hz interrupt clock. The interrupt clock is used during light pen tracking and for synchronizing frame repetitions. The other standard devices are used only for bootstrapping, maintenance, and debugging. Options included on the computer are an Extended Arithmetic Element, a Direct Memory Access Channel Multiplexor, and an Automatic Priority Interrupt (API) system.

The display processor controls a slightly modified D. E. C. Type 343 Slave CRT Display. The method of picture formation is incremental point plotting on a 1024×1024 matrix. Lines and characters are formed from closely spaced points. Thus, the display can be classed as a dot scope, as opposed to a stroke-vector scope in which lines are swept out.

Console input devices are a light pen, an ASCII-code keyboard (full seven-bit code), and eight pushbuttons. The selection of this set is based upon experience obtained on the GRAPHIC-1 console. The small number of devices is possible because, with a good programming system, the light pen is extremely flexible. The display screen can be both a control and a display surface (13). Thus, the light pen can serve both to indicate the flow of control when pointed at light buttons (control functions on the screen) or to supply arguments to any activated subprogram. A console is connected via a DATA-PHONE® Data Set 201A to the GE 645 system.

The processor of the PDP-9 responds to input device signals to activate programs in the shared memory. In executing these programs, words are accessed from the memory, the operation code and arguments are interpreted, and the appropriate action such as an add, shift, etc. is performed. The display processor, through the memory multiplexor, has "cycle stealing" access to the memory with higher priority than the computer processor. It also accesses words from the memory, interprets the operation code and arguments, and performs the appropriate action. For the display processor these actions include displaying points, lines, and characters on the console CRT.

The PDP-9 processor and the display processor can operate independently of one another. the PDP-9 processor has control over the display, i. e., it can start and stop the display processor. One computer command loads the contents of the accumulator into the display address register of the display processor and starts the display. Under internal controls, the display processor uses the display address register to sequence through display words from the shared memory. As each word is executed, the display address register is incremented and the next word is fetched. Thus, the display address register performs the function of the program counter in a normal computer. The execution of sequential words from memory is performed until the computer issues a stop command or until specially coded display words (display trap words) stop the cycling and signal the computer. This latter situation will be discussed in detail shortly.

In addition to the information channel from the shared memory to the display processor, there is a path from the computer accumulator to the display processor. This path is reminiscent of the only data path in some early display scopes attached to computers. The instruction codes which are used in the direct memory access path are also used here. The accumulator-to-display path is under single step control, i. e., the display halts and a signal is passed back to the computer when the accumulator-provided word has been completely executed. The display address register is not changed during such actions. So, at the end of a string of words from the accumulator, a single command can restore normal cycling.

The accumulator data path may be used in setting and restoring the display status. It may also be used in performing interpretive function generation in which a display trap word stops normal cycling, interrupts the computer and provides functions generation arguments. The computer then uses the accumulator path to generate the function. Circles, arcs, or other curves can be generated in this way with normal display cycling being resumed upon completion of the function. A data path from the display processor back to the computer accumulator is used to pass display processor status such as X and Y coordinates back to the computer.

The display processor codes are shown in Fig. 2. In some previous consoles, a bit configuration could mean different things depending on the display mode. This problem has been avoided by designing the display code set with a separate operation code in each word. Thus, each display word is treated independently and one can tell by merely examining a single word what operation it will perform.

A leading "0" bit categorizes the word as a display primitive. The primitives control the setting of display parameters, and the plotting of points, lines, and characters. Characters are formed utilizing a dot matrix approach. A specially designed character generator uses the computer memory to store a character font. The character code is combined with a pointer to address a dispatch table in the computer memory. The dispatch

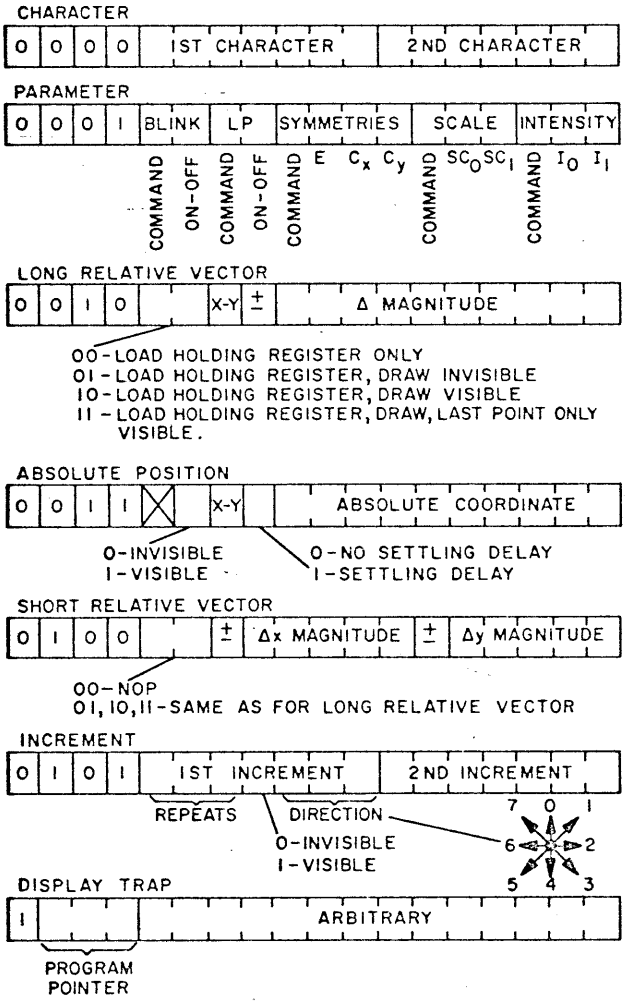

Fig. 2. DISPLAY PROCESSOR CODES

table entry is then used to access increment mode-like words which describe the character. A separate channel of the direct memory access multiplexor is employed. Double word buffering is used to minimize memory access waiting time. A user can compose and use any 128 character font he desires. No basic character grid is enforced although a 5×7 grid is normally used. The ASCII font is available as a programmed standard. The core space required by the ASCII font and dispatch table is 456 words.

Parameter, absolute position, and increment word functions are fairly standard. One exception is the parameter word symmetry feature which transforms the up, down, right, and left movement commands from the vector, character, and increment generators. The first bit, E, exchanges the axes. If E is set, a right command becomes an up command, a left command becomes a down command, and vice versa. The second bit, C_X, complements the X direction movements. If set, it makes a right command into a left command and vice versa. The C_Y stands for complement Y, and performs analogous functions to the C_X bit. Combinations of exchange and complement can be used to produce any of the symmetries of a square.

Except for symmetry, the condition bits in the parameter word indicate whether the corresponding current parameter value is changed or remains the same. For the symmetry feature, a "1" in the condition bit here functions as for the other parameters. A "0" condition bit, however, means that the indicated symmetry transformation is concatenated with the current transformation to form a new symmetry transformation. For example, if a +90° rotation is already set, concatenation with a 180° rotation produces a +270° rotation. Thus, the symmetry feature functions properly for nested graphical subroutines.

Several options are provided in the drawing of vectors. The holding register for the specified component may be loaded only. The holding register may be loaded and the vector, as specified by the current contents of both holding registers, drawn with or without beam intensification; both holding registers are cleared after vector execution. Alternately, the vector may be drawn with only the last point of the vector intensified. This option gives a relative point capability.

If the leading bit of a display word is "1", the word is a display trap. Upon encountering a display trap word, the display processor stops and signals the console computer via the automatic priority interrupt. The second, third, and fourth bits of the display trap determine which of eight transfer pointers in an interrupt transfer vector is used to direct control to a program. Thus, eight programs can be quickly evoked by display trap words. Other pointers in the interrupt transfer vector are activated by signals from the light pen, pushbuttons, keyboard, data phone, and display processor.

The display processor logic is presently running at 1 μsec per intensified dot in a line or character and 1/2 μsec per unintensified dot, or off screen dot. Design improvements should allow reductions to 500 and 200 nsec, respectively. The present random point plot time of 35 μsec will also be reduced.

EDGE VIOLATIONS

A description of a large problem can be contained at a console. During work on the problem, a console user may want to have only a small portion of the total picture fill the screen. He may want to move this area of interest, or window, smoothly about. During such actions, there may be picture pieces which are only partially contained on the CRT. It is particularly important in a satellite console that edge violations from window movement be easily and appropriately handled. Extra space for a separate display list and

cropping program and the processing of the total data base for every minute movement should not be necessary. Instead, edge violations should be handled dynamically as they occur.

The dynamic handling of physical edge violations is generally called "scissoring." The common approach to achieve scissoring is to represent picture parts by incrementally specified lines, points, and characters. Then if these incremental movement commands cause a scope boundary to be crossed, the CRT beam is blanked until the boundary is crossed in the opposite direction, at which time the picture again is on the screen. Extra bits in the X and in the Y coordinate registers are used to detect scope edge violations (3, 14, 15). Dot intensification does not take place unless the extra bits in both coordinate registers are all zero. A picture thus "wraps around" when only the least significant bits of the coordinate registers are considered but is intensified normally or totally blanked under control of the extra bits. Overflows or underflows of the extra bits generate computer interrupts. Programs are then used to handle these situations.

The edge-handling scheme used in GRAPHIC-2 is kindred to these techniques. Instead of extra bits in the coordinate registers, however, a program settable switch determines whether the display is blanked or normal beam intensification takes place. The computer is signalled every time an edge is violated and sets the status of this override switch before resuming the display. This approach is less expensive in hardware than having extra register bits. Even with the extra bits approach, programming must be present for underflows or overflows of these. Always using this underflow-overflow type of programming makes for a consistent symmetrical approach.

Two software registers are used to accumulate the edge violations, i. e., one register is incremented for each right edge violation and decremented for each left edge violation, and another register is used similarly for top and bottom edge violations. The edge-handling routine determines if both registers are zero, and if so, the current point, line, or character being generated should appear on the screen. If not, the picture is off the screen. So the scope hardware registers serve as the low-order ten bits of X and Y coordinate registers, the high-order bits of which are contained in software registers of any desired size. In practice, one 18-bit register for each axis has been set aside for the extra bits, allowing scissoring over a $2^{18} \times 2^{18}$ area.

INTERPRETIVE OPERATION OF THE CONSOLE

As previously emphasized, it is important to be able to display structured information locally. This is achieved in the GRAPHIC-2 system through intimate cooperation of the PDP-9 processor and the display processor. The display processor performs the primitive operation such as drawing points, lines, characters, and setting display parameters. The display trap words call upon the PDP-9 to perform the control operations analogous to direct and subroutine transfers.

One might argue that such extensive demands should not be placed upon the PDP-9 in running the display. The computer should be free for handling other tasks. Also specialized hardware is generally faster than interpretive software so that more material might be displayed flicker-free if hardware transfer were used. The specialized hardware approach is being taken with some current displays (14, 15).

Experience with the GRAPHIC-1 console has shown, however, that if not involved in running the display, the computer processor really is idle. On the other hand, using the computer processor to help in running the display reduces the complexity and cost of the display electronics, allows a hierarchical data structure to be displayed directly, and allows a pushdown stack of display structure to be maintained dynamically, when appropriate, so that light pen strikes are easily serviced.

The amount of flicker-free picture that can be displayed is not significantly affected by interpretive operation since operation of the PDP-9 processor and the display processor are overlapped. To see how the structure level is dynamically traced in overlap operation, let us begin by considering Fig. 3.

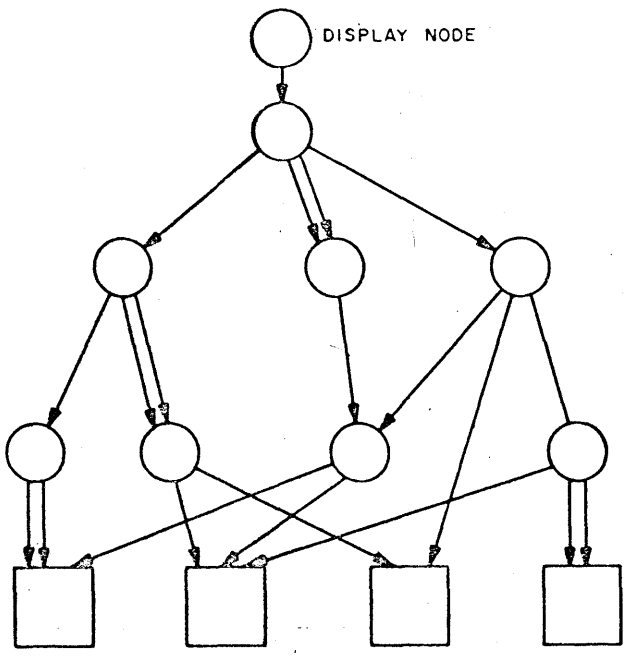

Fig. 3. PICTURE STRUCTURE

Here, a typical picture structure as used in GRAPHIC-2 is illustrated (10). This is a

directed graph structure in which there are several types of entities. The circles represent nodes; the straight lines with arrows, branches; and the squares, terminal nodes or leaves. The nodes represent particular subparts of a picture. The branches are references to lower-level subparts of the picture. By convention, only the leaves contain displayable material which is represented in display processor code formats. The related data structure in the central computer also conforms to the constraint of having displayable material only in the leaves, although this information is not represented by display codes. So nodes and branches represent structure; leaves, picture elements. Display is accomplished as follows. The console computer traces down from the display node which heads each picture until it reaches a leaf. The display processor is started on displaying the leaf. In the meantime, the console computer traces through the structure to find the next leaf. Position offsets between leaves are added up digitally during the structure tracing. When the display processor hits the display trap word marking the end of a leaf, the computer processor senses this and sets the display to the starting position of the leaf, gives the display processor the place in memory where the new leaf is located, and restarts the display. The console computer now traces through the structure to locate the next leaf, etc. If a light-pen hit occurs during a display frame, the console computer ceases overlap operation during the next frame and instead dynamically maintains a pushdown list of the path structure, updating this list between leaves. When a light-pen hit occurs in this frame, the complete structure path name leading to that leaf is immediately available. Overlap operation is resumed on the next frame.

The light pen can be used to point at a particular leaf in the picture structure. Programming techniques are used to blink the selected leaf or higher level structures containing the leaf. The place in the data structure of the currently considered leaf is thus made apparent to a user. This type of visual feedback has proven valuable, if not essential, both in the GRAPHIC-2 and elsewhere (16).

During overlap operation, the structure tracing for a new leaf is often finished well before the display of the previous leaf. Therefore, a minimum of time is needed to start the display processor on a new leaf. Extra time is required only during a light-pen interpretation frame. Contrast the interpretive mode to the operation of fixed hardware which must take the time for all the structure tracing and addition of position offsets in series with display time each frame. For heavily structured displays, the gains of overlapped interpretive operations can be significant.

Aside from handling leaf ends, display trap programs are used to perform data-structure manipulations every display regeneration cycle. Examples of this type of program are a "move" program to cause an object on the CRT to follow the light pen, and a "rubber-band line" drawing program. Another use is function generation using the accumulator path as outlined earlier. Still other uses are to change the character table pointer so that several character fonts appear in a picture and to implement the carriage return character function.

COMMENTS

Although in the GRAPHIC-2 software system the display handling proceeds without the exact software duplication of current hardware techniques, there is nothing to prevent such use. The display trap programs could easily perform conventional direct transfers and subroutine transfers and return (with status saving and restoring) or any other operations. Other software systems might employ the flexibility and power of display trap techniques in as yet unconceived ways. The ability provided by display trap words to cause console computer programs to be executed at specified places in the display cycle is another step in the continuing development of more powerful display techniques. These techniques began with displays run from lists through the computer accumulator, moved to linear lists out of a channel, then to hardware subroutining, and now to extensive intermixed programming.

The satellite console organization seems to be finding increasing favor. Older console systems are being modified (17), the large suppliers are joining some standard pieces to form such a system (18, 19), and entirely new consoles (20) and systems (21) are being designed. It is hoped that the ideas being pioneered by the GRAPHIC-2 system will be useful in these new systems.

ACKNOWLEDGMENTS

The author is indebted to C. Christensen, A. D. Hause, H. S. McDonald, E. N. Pinson, and L. Rosler for their many contributions in system organization and software aspects. The design efforts of P. E. Rosenfeld on I/0 interfacing and H. S. Magnuski on the character generator are acknowledged.

REFERENCES

(1) E. L. Jacks, "A Laboratory for the Study of Graphical Man-Machine Communication," Proc. of the FJCC, 1964, pp. 343-350.

(2) I. E. Sutherland, "Sketchpad: A Man-Machine Graphical Communication System," Proc. of the SJCC, 1963, pp. 305-329.

(3) R. H. Stotz and J. E. Ward, "Operating Manual for the ESL Display Console," Electronic Systems Laboratory, M.I.T., Cambridge, Massachusetts, ESL 9442-M-129, March 9, 1965.

(4) C. A. Lang, "New B-Core System for Programming the ESL Display Console, "Electronic Systems Laboratory, M.I.T., Cambridge, Massachusetts, ESL 9442-M-122, April 30, 1965.
(5) J. Katzenelson, et al., "A Program for On-Line Analysis of Electronic Circuits," IEEE International Convention Digest, March 1967.
(6) J. R. Kennedy, "A System for Time-Sharing Graphic Consoles," Proc. of the FJCC, 1966, pp. 211-222.
(7) W. H. Ninke "GRAPHIC-1 - A Remote Graphical Display Console System," Proc. of the FJCC, 1965, pp. 834-846.
(8) N. A. Ball, et al., "A Shared Memory Computer Display System," IEEE Transactions on Electronic Computers, Vol. EC-15, October 1966, pp. 750-756.
(9) H. C. So, "OLCA: An On-Line Circuit Analysis System," IEEE International Convention, March 1967.
(10) C. Christensen, E. N. Pinson, "Multi-Function Graphics for a Large Computer System," Proc. of the FJCC, 1967, pp. 697-711.
(11) E. M. Thomas, "System Considerations for Graphic Data Processing," Computers and Automation, November 1967, pp. 16-19.
(12) PDP-9 User Handbook, Digital Equipment Corporation.
(13) W. H. Ninke, "Man-Computer Graphical Communication," in F. F. Kuo and J. F. Kaiser (Eds.) System Analysis by Digital Computer, John Wiley and Sons, Inc., New York, 1966, pp. 395-432.
(14) "Programmed Buffered Display Type 338", Digital Equipment Corporation.
(15) "Elliot 4280 Graphical Display System", Elliot Automation Computers, Ltd.
(16) J. D. Joyce, M. J. Cianciolo, "Reactive Displays: Improving Man-Machine Graphical Communication," Proc. of the FJCC, 1967, pp. 713-721.
(17) Investigations in Computer-Aided Design for Numerically Controlled Production, Electronic Systems Laboratory, M.I.T., Report ESL-IR-278, August 1966, pp. 43-45.
(18) "The 1700/274 DIGIGRAPHICS System," Control Data Corp.
(19) "1130-2250 Model Component Description," IBM Corp.
(20) "The New IDIIOM, "Information Displays, Inc.
(21) D. T. Ross, et al., "The Design and Programming of a Display Interface System Integrating Multi-Access and Satellite Computers," ACM/SHARE 4th Annual Design Automation Workshop, Los Angeles, California, June 26-28, 1967.