J. Birnbaum
T. Kwap
M. Mikelsons
P. Summers
J. F. Schofield
F. Carrubba

An Interactive Graphics System for Nuclear Data Acquisition

Abstract: The graphics terminal described was developed for low-energy nuclear-physics data acquisition and control, and is currently in use at the Yale University Wright Nuclear Structure Laboratory as part of an IBM System/360 Model 44-based system. It is comprised of dual cathode ray tube displays, a light pen, and function keyboard, and includes character generation, display simulation, and photographic facilities. It is capable of plotting 200,000 points per second with variable intensity. The display programming structures, which support highly interactive communication between physicist and computer, are discussed in detail. A data acquisition programming system permits the creation and manipulation of self-describing global data and display entities. Examples of the resultant increased experimental sophistication and efficiency are presented.

Introduction

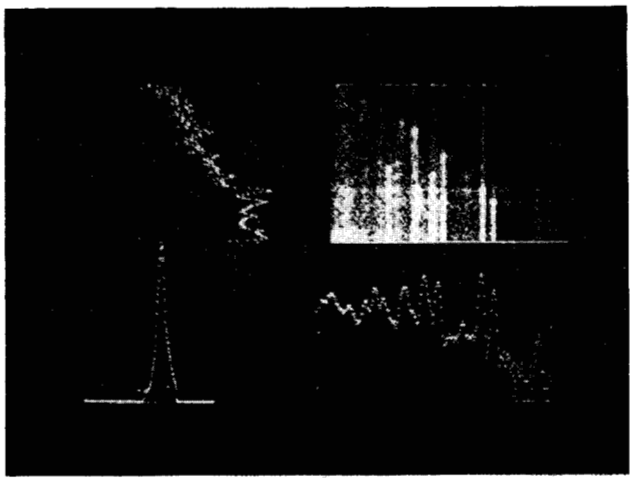
In the fall of 1965 the IBM Research Center, Yale University's Wright Nuclear Structure Laboratory, and the Atomic Energy Commission entered into a joint study whose purpose was the provision of on-line computing facilities at the Van de Graaff accelerator laboratory at Yale. The initial results of this collaboration have been published elsewhere[1,2] and will not be repeated here. Rather, this paper will treat those aspects of the project which deal with the interaction of the physicist and the experiment by means of a graphic terminal designed to meet the exacting requirements of on-line nuclear physics.

From the outset, a principal goal of the joint study has been to provide a computing system which would not only become an integral part of the experiment, but which would also serve as a powerful, yet flexible, extension of an experimenter's judgmental ability. This has led to the creation of a system in which simultaneous data acquisition, analysis, display and control are performed on a single computer,[3] an IBM System/360 Model 44. Emphasis has been placed upon making the full resources of the computer available while sparing the experimenter from concern with the detailed working of the hardware and programming system. Thus, data acquisition components were made modular and interface to the computer in a simple manner; all real-time programs may be written in a powerful, application-oriented dialect of FORTRAN; references to data are made symbolically; programs are independent of whether being used in real-time or not; and most experimental parameters are dynamically alterable at execution time without requiring either recompilation or advance provision.[4] A powerful multiprogrammed operating system, with full protection facilities, and dynamic storage allocation has been written to support these activities. A minimum core size of 64K bytes is required.

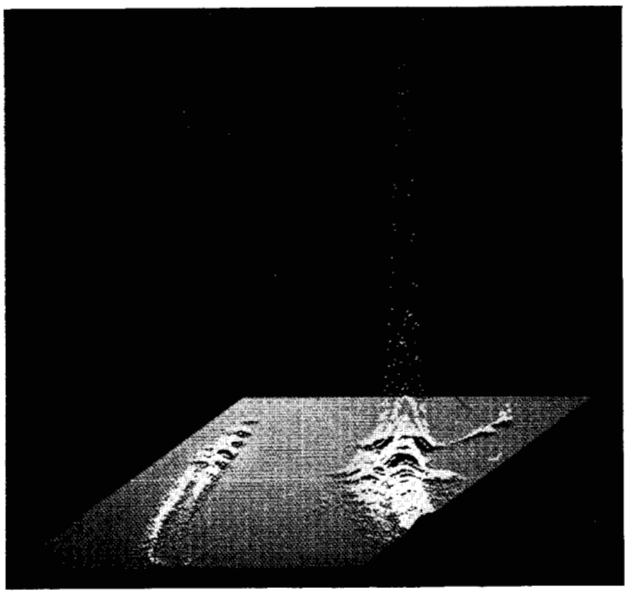
A basic requirement of such a system is that communication between experimenter and computer must be both flexible and rapid, and must be made directly in terms of the application rather than the system supporting it. This immediately suggests the use of graphic techniques: a display can be used not only for the visual presentation of data and intermediate results, but also as the medium for efficient choice of strategic alternatives during the course of an experiment. A further requirement is that the creation, combination, manipulation, deletion, and retention of such displays be accomplished simply, without special involvement on the user's part.

This paper describes the hardware that was designed to achieve these goals, and more importantly, the program-

Birnbaum, Kwap, Mikelsons, and Summers are at the Thomas J. Watson Research Center, Yorktown Heights, N. Y., and Schofield and Carrubba are with the Data Processing and Field Engineering Divisions, respectively, in New Haven, Conn.



(a)



(b)

(c)

Figure 1 (a) Display for particle-gamma angular correlation experiment. (b) Pseudo-isometric display of two-dimensional data resulting from heavy ion transfer reactions. The hyperbolic loci represent nuclear isotopes. (c) A contour map of the data in 1 (b). Only two levels of intensity have been used.

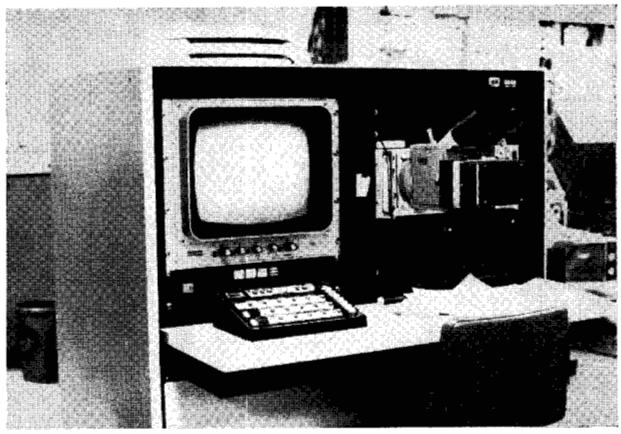


Figure 2 The Yale/IBM Graphics Terminal.

ming structures underlying it. Examples of the increased sophistication and efficiency achieved in the nuclear physics environment are presented. The techniques employed are of sufficient generality to warrant consideration in other applications, and some suggestions of potential extensions are included.

Graphic terminal

The cathode ray tube display has been a standard output device in nuclear physics laboratories for a number of years. Initially, the display was usually part of a fixed-wire pulse height analyzer,[5] but more recently it has also been driven by a digital computer which either supplements or replaces the fixed-wire analyzer.[6] The computer-based displays have also frequently used the CRT as an input medium, usually through the device of a light pen, which is generally used to select desired features of a display for further analysis or closer scrutiny. In general, the application calls for the display of points, frequently with intensity proportional to the contents of the data cell being displayed. Among the more common types of display are single-parameter spectra (histogram-like plots), contour maps, and isometric representations of multi-dimensional data. Figure 1 illustrates some typical displays; as can be seen, there is frequently a need to plot many thousands of points in a given display.

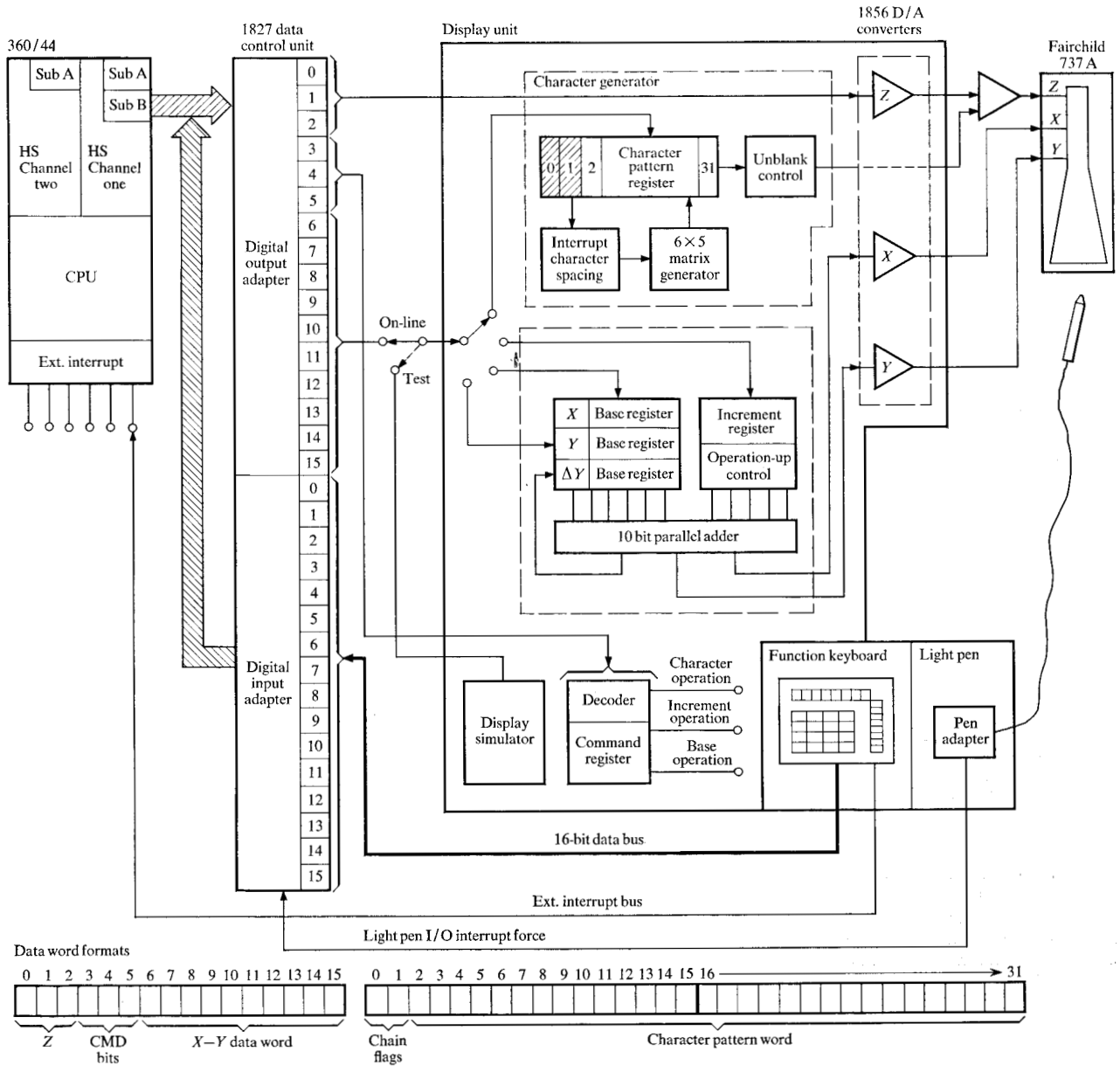
The display console designed and constructed for the IBM/Yale system is shown in Fig. 2. Its functional relationship to the computer is illustrated in Fig. 3. The coordinates of points to be displayed are stored in an array in the main core memory of the central processing unit (CPU) of the Model 44. Transfer of these points to the display is completely under control of the high-speed multiplexor channel, so that once a display list has been structured, CPU interference is limited to pre-empting memory cycles in the event of a storage access conflict

(commonly called cycle-stealing). (A design principle of the entire system is the relegation of input-output functions to the channels whenever possible to permit many simultaneous operations.) The channel program transmits the contents of the display buffer, through a 16-bit digital output register on the IBM 1827 Data Control Unit to a display control unit, which decodes this information and routes it to the appropriate digital-to-analog converters (DAC's). These are, in turn, connected to the deflection plates and intensity-modulation circuitry of the CRT. The control unit also contains circuitry for timing control across the demand-response interface, light pen control, a character generator, and a display simulator which is capable of independently producing standard patterns on the screen for calibration and diagnostic purposes.

The display can be operated in a random point plotting mode, or in raster mode, in which a point is taken to be in the same plane as its predecessor until a new planar coordinate is specified. Approximately 100K points/sec can be plotted randomly, with 200K points/sec being the limit in raster mode.

The system is essentially independent of the particular CRT used; at Yale the 17-in. Fairchild 737A, with 1-MHz bandwidth and 1024 addressable points in both directions

Figure 3 Functional diagram of the graphics terminal.



has been used. The CRT has a P7 phosphor, so that the light pen, which contains a filter, is sensitive only to the fast rise time blue component (response of the pen is less than 1 μsec), while the long-persistence yellow-green component minimizes the number of regenerations necessary for flicker-free performance. A second CRT, a 5-inch Fairchild 704A, is connected in parallel to the first, and is used for polaroid or 35mm photography.

A relocation register is provided in both the *x*- and *y*-directions so that the coordinates of a point on the screen are the sum of the coordinates transferred from the core buffer and the current contents of the corresponding relocation registers. These registers are analogous to the base registers in the System/360 architecture: they permit an entire display structure to be relocated on the screen by manipulation of the register contents. When the appropriate control bits are set, the two bytes of information are interpreted as a new base address for the specified register, and no point is plotted.

The control unit may be operated in either an absolute or incremental plotting mode, according to the setting of the command bits. In the former, the contents of the associated relocation registers remain unchanged until reset by the appropriate command. In the incremental mode, the relocation registers are updated with the coordinates of the last point displayed each time the scope is unblanked, i.e., the coordinates in the display buffer act as a set of increments relative to the last point plotted. When repetitive structures are involved in a display (such as grids), the incremental mode results in a great saving of core, and permits structuring a display as a family of linked image subroutines, as will be discussed later.

The creation of alphameric text on point-plotting displays is a cumbersome and core-consuming procedure, since a given character must be constructed from a dot matrix. The Yale unit contains a character generator to minimize this problem. The starting coordinates, intensity, and size of a given character are set with two two-byte transfers, with the appropriate command bits on to indicate this as a character operation. Then 32 bits of coded information are transferred, where each of the first 30 bits indicates whether the corresponding position of a 6 × 5 dot matrix is to be on or off. The character generator consists of circuitry which sweeps the DAC's through the 6 × 5 matrix, and blanks or unblanks the screen according to whether the corresponding bit is a ZERO or ONE. The last two bits may be used to generate automatic intercharacter spacing in a line of text; i.e., only the coordinates and specifications of the first point in a standard line need be specified. The sizes of the characters are specified as arbitrary multiples of 6 × 5 raster units. An advantage of this scheme for character generation is that it permits an unlimited variety of type fonts and sizes.

A function keyboard has been constructed to provide a simple and rapid means of communication with the system. In operation, an experimenter places a plastic template over the keyboard which defines a set of programs to be associated with key depressions on that template (e.g., "display," "analysis," "calibration," etc.). Depression of the desired key causes an interrupt to the computer which results in the corresponding program being located, loaded (if necessary), and executed, and also transfers eight bits of user-supplied information to the program. To protect against the danger of accidental multiple key depression, an electrical interlock is included: if more than one key is depressed simultaneously, the interrupt is inhibited, an error light is turned on, and the keyboard is disabled until an error reset key is depressed.

The system typewriter is adjacent to the graphic terminal and, if desired, may be used to enter information to augment that provided by the light pen and function keyboard.

A hardware display simulator in the display control unit produces horizontal, vertical, and 45° lines, as well as a full 1024 × 1024 raster of points, independently of the computer. A given pattern may be produced regeneratively or only once, according to the position of a switch. The simulator is useful for diagnostic and calibration purposes.

Display structures

Since each displayed point is represented in the computer memory by coordinate words which are repetitively written by the data channel, the creation and manipulation of a display is accomplished by operations on arrays of coordinates. The purpose of the programming system is to impose structure on this process by defining display entities as sets of coordinates contained in self-defining storage blocks and manipulated by a set of FORTRAN functions. This section describes the format and purpose of the various descriptor blocks and the basic facilities provided to operate on them.

• *Display descriptors*

The basic units of display, i.e., the plotted points represented by coordinate words in storage, are grouped together into display buffers. Each buffer has associated with it a WRITE command for the data channel. These commands are linked together by transfer-in-channel commands, as shown in Fig. 4, and are executed repetitively by the channel to regenerate the display. Thus, to add a buffer of points to an existing display, the associated WRITE command is inserted into the channel loop; to remove a buffer, the links are changed to bypass the associated WRITE command.

While the coordinate buffer represents a unit of display organization from the point of view of the data channel, the user deals with logical display entities identified by a positive integer called the *display number*. The display number is used by the system to index a table of pointers to

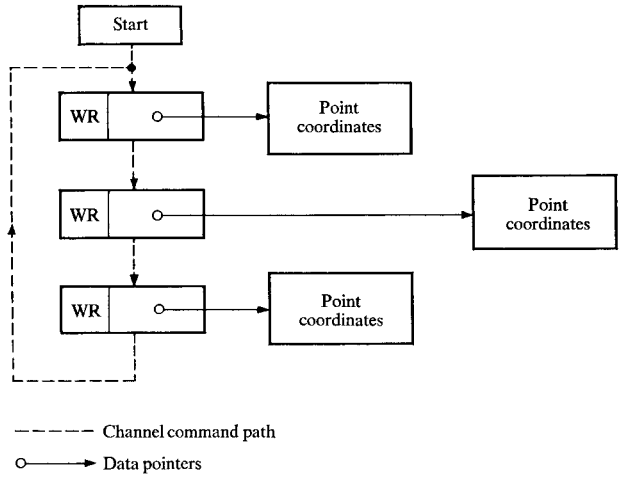
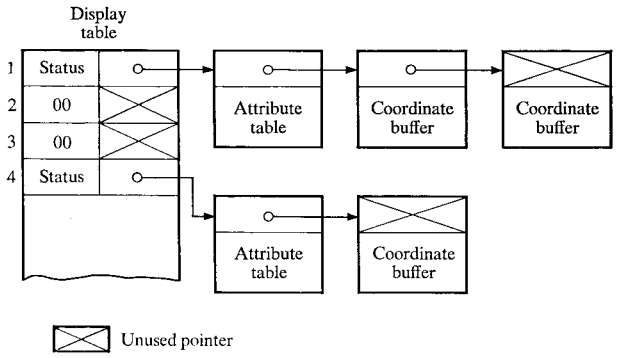


Figure 4 Display regeneration logical structure.

Figure 5 The display structure as seen by the user (applications programmer).



attribute tables (Fig. 5). Each display number is described by a two-part attribute table. The first part is a fixed system area containing pointers to coordinate buffers, the second is a variable length user area that contains parameters describing the logical contents of a display entity; these are needed in the re-creation and manipulation of displays. Any number of coordinate buffers may be associated with a display number and all operations performed on a display number affect all the buffers associated with it.

All the display descriptors reside in a contiguous block of storage defined at the beginning of a job by the data acquisition monitor.[3] The size and location of the display table is fixed for a job but space for attribute tables and coordinate buffers is allocated and released dynamically. To minimize fragmentation of the storage area a spatial condensation ("garbage collection") is performed whenever a contiguous block of desired size cannot be found. Since all user references to display entities are made by display number, descriptors can be moved whenever necessary as long as the addresses in the display table are updated. Translation of descriptors is facilitated by the common header format shown in Fig. 6. All blocks of storage contain a size field; unused blocks are identified by a zero display number, while the display number in an active block directly connects it to a display table entry.

• *Primitive functions*

Since a number of descriptors are involved in defining a display entity, several states of existence may be defined. These are briefly as follows:

NULL: The entity is undefined.
DORMANT: A display number has been assigned to the entity and an attribute table created.
AWAKE: One or more coordinate buffers have been created for the entity.
ACTIVE: The entity is currently being displayed on the CRT.

Operations on display entities can thus be viewed as transitions from one state to another. A set of function-type subroutines permits the user to perform the transitions shown in Fig. 7. Another set of functions permits the transfer of data to and from a display entity. The main argument of all the functions is an integer, the display number, which defines the entity upon which the operation is to be performed. The returned value of each function is equal to the display number if the operation was completed satisfactorily; otherwise, the value is a negative integer that indicates a particular error condition.

• *State transition primitives*

The transitions shown in Fig. 7 are performed from the null to the active state and back to null again by the functions ASSIGN, AWAKE, ACTIVE, ERASE, RELEAS, and UNASGN. All the functions have a display number as an argument and return the standard values. The ASSIGN and AWAKE functions have additional parameters to specify the size of the user area and coordinate buffer respectively. The ASSIGN function can be called with a display number of ZERO; in this case, the display table is searched for the first null entry and the transition is then performed on that display number. The function can thus be used to find an unused display entity. The meaning of the AWAKE-to-AWAKE state transition is that any number of coordinate buffers may be created for an entity by calling the AWAKE function. These are automatically concatenated by the system. The separation of the ACTIVE and AWAKE states permits the preparation of display lists to be independent of actual transfer to the display control unit. This permits display entities to be activated or erased according to an algorithm based on real-time, rather than one dependent, in part, on display creation time.

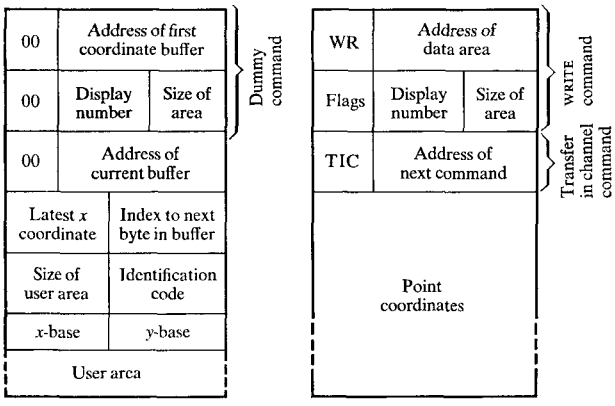
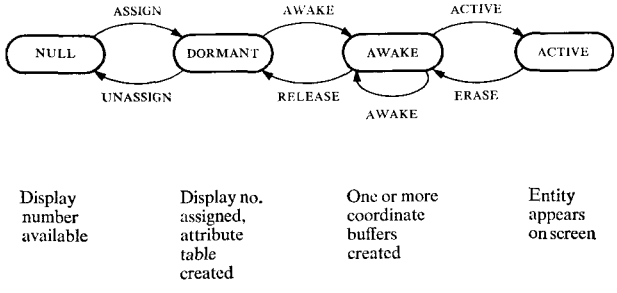


Figure 6 Formats of dynamically allocated descriptors.

Figure 7 Allowed transitions, display states.



• *Data transfer primitives*

The actual operation of plotting points, i.e., of storing coordinates in a buffer which will subsequently be made ACTIVE, is performed by the PUTPT (DN, *X*, *Y*, *Z*) function. DN is the display number; *X* and *Y* are the coordinates of the points in a 1024 × 1024 point frame of reference; *Z* is an integer from 1 to 7 specifying the intensity of the point. Coordinates are stored sequentially in the last created buffer; when no more space is available in a buffer, a new buffer can be created with the AWAKE function and plotting continued. The base-displacement and random-raster structures supported by the hardware are handled automatically by the routine in such a way as to make optimal use of buffer space.

TRACK (*X*, *Y*, *Z*, *T*) is a function used to retrieve values from a display by means of the light pen. When TRACK is called, processing is suspended until a light pen interrupt is detected. The routine then determines the coordinates of the selected point and the display number of the associated entity; if no interrupt is detected within the (user-variable) period of time *T*, control passes to the next programmed instruction.

Two functions, GETDAT and PUTDAT, are provided to manipulate groups of bits in the user area of the attribute table; the purpose of the user area is more fully described in a following section.

• *Special purpose primitives*

To make optimal use of the slave photographic scope in the terminal, the opening and closing of the camera shutter must be synchronized with the display regeneration cycles. The CAMERA function allows the user to stop a display, regenerate it for a given number of times and restart it, thereby eliminating the need for electronic shutter synchronization and guaranteeing a photograph in which intensities can be differentiated properly, since each point in the picture has been regenerated the same number of times. The number of regenerations necessary for proper exposure can be easily determined by using the simulator hardware in the single-cycle mode; once obtained, this value is independent of the number of points displayed.

Another special purpose primitive causes a "tracking pattern" to be inserted between each entity in an active display list. This procedure increases the probability of detection when the light pen is used for drawing. The CURSOR function initiates this process by modifying the channel command list; the NCURSE function stops it.

Display programming

The average system user is not concerned with the rather complex structures outlined above: in fact, most experimenters at Yale have used them successfully without any awareness of their internal detail. Since the data arrays which result from the data acquisition are also self-descriptive, information may be passed freely to and from the display routines on a purely symbolic basis. The purpose of this section is to indicate the manner in which the applications programmer, writing in FORTRAN, constructs these programs.

• *Program hierarchy*

The display primitives described above constitute only the lowest level of an *n*-level hierarchy of programs. As first level routines, they are machine- and configuration-dependent and are written as FORTRAN functions in System/360 assembly language for maximum efficiency. Second- and higher-level routines are generally written in FORTRAN and are functionally independent of machine type. The average experimenter programs only at the higher levels, if indeed he programs at all.

The creation of a second-level display program will typically involve the following simple steps:

1. A display number is obtained and an attribute table is created using the ASSIGN function.
2. A display coordinate buffer is created using AWAKE.
3. Data to be displayed is fetched, scaled to display coordinates, and inserted in the buffer using the PUTPT function.

4. If desired, information uniquely specifying the display (e.g., scaling factors, display origin) is placed in the user area of the attribute table using PUTDAT.
5. ACTIVE is invoked to initiate the display.

Although this is a simple and rapid procedure, it is seldom necessary. A large family of second-level routines, comprising all the standard displays, and display building blocks such as text production, axes, data scaling, etc., have been provided and the physicist can usually structure a specialized third-level routine from a combination of these second-level blocks. For example, a display program to produce a one-parameter spectrum with labelled axes and a title simply requires call statements to the three appropriate second-level routines. Information about the data to be displayed is passed to the routine symbolically through the global system linkages by simply naming the array or variable involved.

The function keyboard permits a user to invoke a program by depressing a key. This program, written by the user as a FORTRAN subroutine, has the full facilities of the system at its disposal; it may communicate with other programs by means of conventional common blocks or through global symbols and display numbers. A program is associated with a function key by means of a control card at the beginning of the job. In addition, program names can be grouped under a name associated with a selector key; selector names can be grouped under a keyboard name. The entire keyboard definition can be catalogued on disk and invoked by name.

An example of the types of display program often assigned to the function keyboard are the routines for display manipulation, which are a set of second-level routines written to perform certain transformations on created displays. Routines are provided in the display package for translating, rotating, changing the intensity or changing the size of a selected display. An experimenter may thus, by selecting appropriate function keys, manipulate his displayed experimental data. For example, an experimenter may have collected data that is being displayed as a spectrum on the CRT. Further, he may have a spectrum of theoretical data on file. The experimenter may select either or both spectra, select the desired display and, with both on the CRT at one time, he may, for example, move one spectrum to a position where direct visual comparison may be made, or he may choose to adjust the relative gains of the two displays.

Applications

• *On-line applications*

Virtually all the experiments performed at the Nuclear Structure Laboratory are so heavily reliant on the graphic terminal that their performance, at least at present levels of efficiency, would be impossible without it. Data are displayed in many forms, and often after one or more mathematical transformations have been performed. Usually the displays are created as a result of a function keyboard request and then are manipulated, erased, combined, stored or recalled as necessary. In many instances, parameters which determine the nature of the operation are entered directly from the display by means of the light pen or function keyboard.

The sequence of particle-gamma angular correlation experiments being performed at the laboratory is a fairly typical example of these procedures. These experiments attempt to determine basic properties of nuclei, such as the spin and parity of energy levels, by investigating the angular correlation between particles and gamma rays emitted in time coincidence after a nuclear bombardment. The four-part display illustrated in Fig. 1 is typical of the CRT presentation during the course of the experiment. The upper right quadrant represents a contour display of a slice through the two-dimensional energy space of coincident particles and gammas at the particular angle currently under investigation, with the intensity of the points proportional to the contents of the associated data cells. At the function keyboard, the user can change the contour thresholds, make the intensity proportional to the logarithm of the number of counts, change the data-compression ratios, and so forth. A display corresponding to any particular selection of parameters may be named, saved, and recalled from the disk at a later time (this is useful for comparing results at different angles). This quadrant of the display may be defined as an entity and a particular region of interest may be examined on the full screen at higher resolution. The contents of any cell, or group of cells, are printed when selected by the light pen. The displays adjacent to the contour display represent the total spectra in the associated dimension, although not necessarily to the same scale as the contour display. Again, these may be individually manipulated. Their principal purpose is to allow the user to specify limits of summation in the contour spectrum with the light pen.

The lower left quadrant is free to display any of a number of different quantities independently of the contents of the remainder of the screen. Shown in Fig. 1 is the spectrum representing the time distribution of the coincidence intervals as acquired by a time-to-amplitude converter. The physicist can set limits, using the light pen, which define those events which are to be included in the analysis. This is a simple example of how an on-line computer can significantly affect the quality of an experiment: if a hard-wired coincidence detector is used to tag those events which fall within its resolution the experiment accuracy can be only as good as the coincidence detector (typically tens of nanoseconds). However, by setting this gate digitally in the computer, resolutions of 1 nsec or less

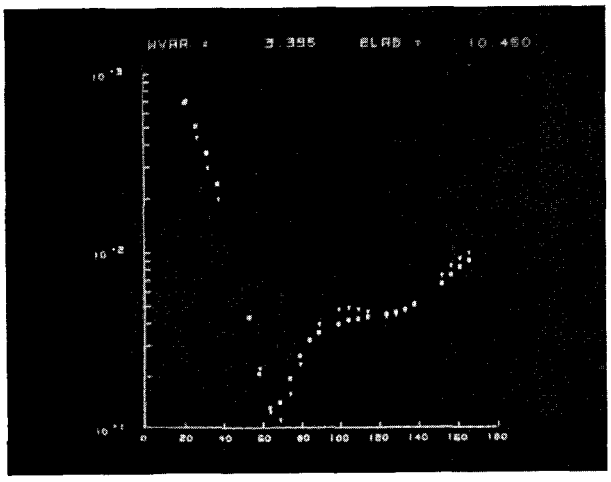
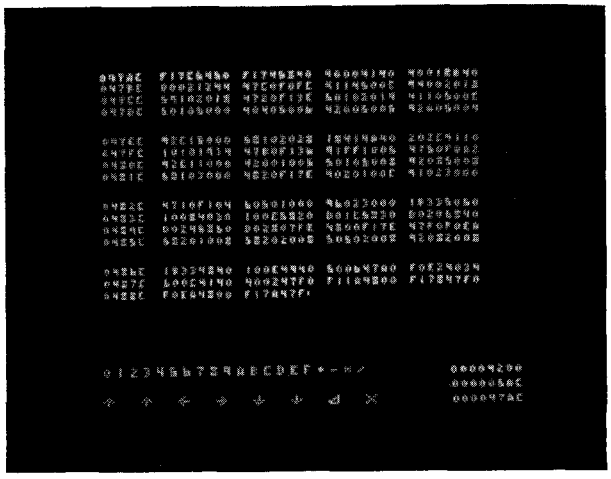


Figure 8 Typical display for phase shift analysis (nonlinear least squares fitting).

Figure 9 The display as used for dynamic memory dump. The light pen is used to manipulate the memory region displayed by selecting the appropriate operator at the bottom of the screen (e.g., up one line, down one page, etc.). The digits and operators serve as the input to a hexadecimal desk calculator at the lower right. The result of the calculation may be selected as the starting address for the current page, as shown.



are routinely obtained. Since all the data, regardless of interval, are simultaneously stored on tape, the physicist can replay them into the computer at a later time with any desired resolution. Because all programs, including those associated with the display, are independent of data source in this system,[1,3] all the on-line facilities are available to the physicist during the replay with no program modification required.

Several such experiments are currently in progress: while the nature of the particular displays involved differs considerably, the high degree of interaction is common to all.

- *Off-line*

In the particle-gamma experiments, further computer processing is done off-line after the entire angular distribution is obtained. Here again, the analysis is closely linked to the display. Such operations as background subtraction are readily accomplished with the light pen, typewriter, and function keyboard. A library of general analysis programs has been built up which uses the graphic terminal in an interactive way. For example, estimates of peak positions, heights, and half-widths are conveniently provided to a nonlinear least squares spectrum unfolding program using the light pen to select the relevant parameters. Many of these programs run in a conversational mode; that is, instructions as to the information required, or actions to be taken are displayed as text on the CRT screen. All have access to the display manipulation facilities, in the manner previously described. An example of the display for a phase shift analysis program is shown in Fig. 8.

The terminal has also proven useful as a debugging aid for the systems programmer. For example, a dynamic core dump is available as a keyboard program which displays a hexadecimal dump of a section of memory (Fig. 9). The light pen is used to select any region for examination; a hexadecimal desk calculator operated with the light pen permits easy computation of relocation factors and other pertinent information.

Conclusions

This paper has described a graphic terminal and associated programming support for a nuclear physics application in which interactive techniques are stressed. It should be apparent that the techniques developed are of sufficient generality to find application in other areas. The concepts are particularly relevant to situations in which high-speed point plotting is required, and in which complex relationships between data structures are being explored.

Acknowledgments

It is a pleasure to acknowledge the many contributions of Dr. H. L. Gelernter, who collaborated on the initial design of the hardware described, and was a vital force behind the entire project. Mssrs. K. Case and V. DiMilia performed the engineering design construction for an earlier version of the terminal described herein. Thanks are also due to H. B. Baskin and members of his Graphics Methodology group at IBM Research, many of whose ideas have been adapted in the system described. We are also indebted to members of the Yale Nuclear Structure

Laboratory, particularly Dr. M. W. Sachs, for many constructive suggestions; Dr. Sachs is also responsible for much of the applications programming described in this paper.

References

1. H. L. Gelernter, et al., *Nucl. Inst. and Meth.* **54,** 77 (1967).
2. M. W. Sachs, D. A. Bromley, J. Birnbaum, and H. L. Gelernter, *Bull. Am. Phys. Soc.* **12,** 1075 (1967); M. W. Sachs, et al., Yale Univ. Wright Nuclear Structure Laboratory, Internal Report No. 32, Jan. 23, 1968.
3. J. Birnbaum and H. L. Gelernter, *Proceedings of the IEEE Nuclear Science Symposium,* Los Angeles, 1967, *IEEE Trans. Nucl. Sci.*, **NS-15**, 109 (1967).
4. J. Birnbaum and M. W. Sachs, *Physics Today* **21,** 43 (1968).
5. See for example, *Proceedings of the Conference on Utilization of Multiparameter Analyzers in Nuclear Physics,* Grossinger, N. Y., 1962 (ed. by L. Lidofsky) Columbia Univ. Report NYO-10595.
6. See for example, *Proceedings of the Conference on Automatic Acquisition and Reduction of Nuclear Data,* Karlsruhe, Germany, 1964 (ed. by K. H. Beckurts, *et al.*), Gesellschaft für Kernforschung, Karlsruhe.

Received June 21, 1968