SLAC-PUB-1861
December 1976
(A)

A TOUCH PANEL SYSTEM FOR CONTROL APPLICATIONS*

K. Crook R. Johnson

Stanford Linear Accelerator Center
Stanford University, Stanford, California USA

The use of finger-sensitive touch panels in association with computer-generated displays for control and monitoring of the Stanford linear accelerator is discussed. This control concept has proven to be very effective. The hardware and software aspects of the Touch Panel portion of the control system are described.

INTRODUCTION

At the Stanford Linear Accelerator Center (SLAC), basic research in particle physics is carried out using a high energy beam of electrons or positrons generated by a 3.2 km linear accelerator (Fig. 1). The facility is operated by Stanford University for the Energy Research and Development Administration.

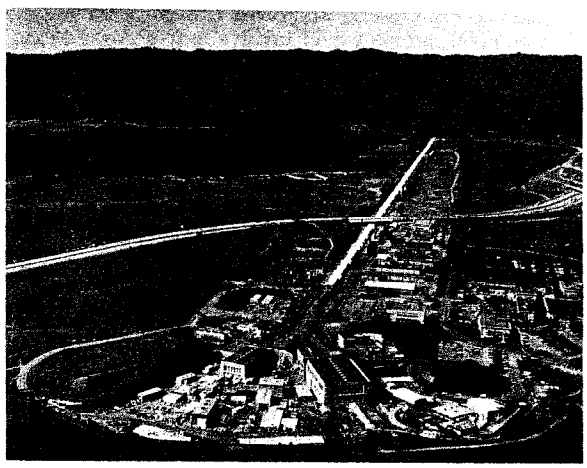

Fig. 1--Stanford Linear Accelerator.

The 3.2 km straight section of the accelerator performs the function of accelerating particles to energies in excess of 20 GeV. The particles are then transported through the Beam Switchyard to four major beam lines, and as many as eighteen different experimental locations. Up to eight different experiments can be run simultaneously.

The accelerator is controlled from the Main Control Center. Three operators and a supervisor monitor the operation of the machine and perform control functions such as magnet setup, energy and current adjustments, and repetition rate control. Operation is continuous, 24 hours a day, for periods up to six months.

*Work supported by the Energy Research and Development Administration.

There are thirty-one data collection and control stations along the accelerator. Equipment connected to each control station includes large power supplies, vacuum valve controllers, AC substations, cooling pumps, and modulator klystrons. Control commands from the operator to a remote device are entered directly on a display screen using a touch-sensitive panel.

THE ACCELERATOR CONTROL SYSTEM

The functional organization of the computer system is shown in Figure 2. Two central computers and eight

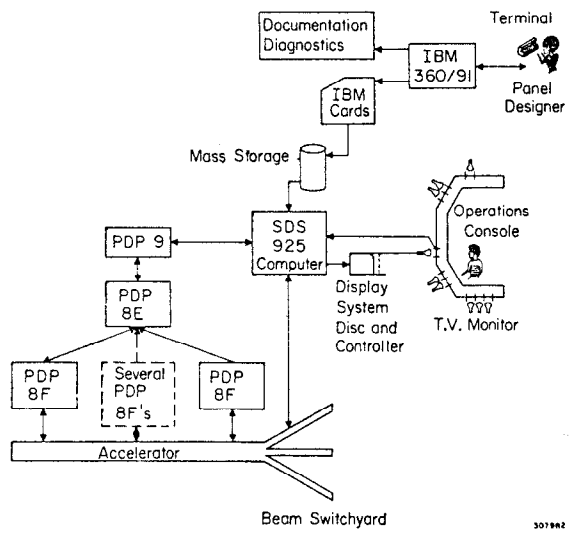

Fig. 2--Functional organization of computer system.

remote minicomputers are used to collect data, issue control commands, report status and out-of-tolerance conditions, and provide hard copy logs.

The Main Control Center houses an SDS-925 computer which controls and monitors the operation of all equipment in the Beam Switchyard. All computers are linked via serial data links operating at 3300 baud. The total volume of signals handled by the computer is quite large: approximately 4000 status

(Submitted to the 5th IFAC/IFIP International Conference on Digital Computer Applications to Process Control, The Hague, The Netherlands, June 14 - 17, 1977)

inputs, 650 analog inputs, and 1400 output controls.

The operating console comprises three almost identical operating positions. One such operating position is shown in Figure 3. Conventional switches and

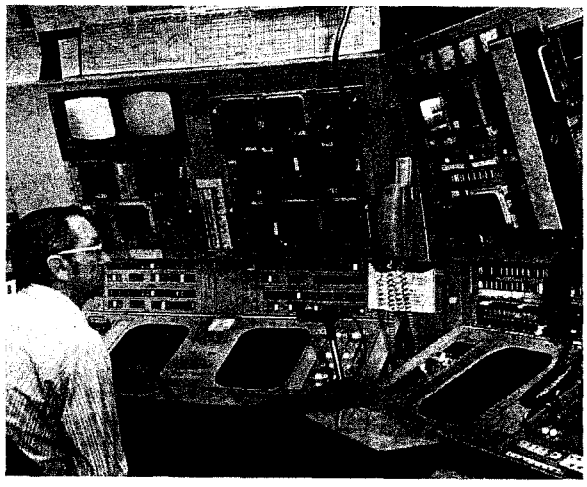

Fig. 3---Single console operating position.

lamp displays are used for certain functions such as emergency-off switches and critical annunciator displays, but the majority of status information and analog signals are presented on computer-generated displays. Similarly, the majority of control commands are issued through the touch panel system.

This approach, using computer-generated displays and touch panel controls with relatively few hardware control and status panels, has made it possible to keep the console size reasonably small, and to present data to the operators in a unified and coherent manner.

In order to establish and maintain the electron beams, the operators must have control and monitoring capability of a large number of devices and items of equipment, such as large power supplies, vacuum pumps, and beam collimators. While there are certain closed loop programs that are used to aid beam operators, such as the setting of power supplies to specified values, most controls to date are open loop.

During initial beam setup or when beam parameters are being changed, control room activity can be intense and highly interactive, with all three operators making frequent and sometimes continuous adjustments for long periods.

TOUCH PANEL SYSTEM

Considerations

It was decided that the type of system needed was one where the computer would provide a text plus graphics display having designated button areas with which the operator could interact directly by touching the display screen. The computer would thereby be able to respond to an operator because it would know where the operator was touching and what functional choice was being displayed at that location on the display. Furthermore, any one of an endless number of different displays could then be selected to provide ready access to information and permit control of any part of the accelerator from any terminal.

This type of system would also be flexible and, in effect, self-documenting. Connecting all information and controls to the computer system would involve a one-time effort of installation and documentation. Thereafter, control panels could be designed, generated, changed, or deleted in software with no time, expense, or materials needed to construct, install, and document new hardware panels. For example, the addition of a single additional on/off button to an existing "panel" would take only a program change and the new "panel" could be installed in a few minutes. A similar change to an existing hardware panel could result in materials expense, shop time, documentation time, etc., and could take several weeks. Because the panels would involve only software data changes, output listings would automatically provide immediate up-to-date documentation.

The choice of a suitable control surface to overlay the video screen was more difficult. A fundamental design constraint was that the surface must be curved to match the curvature of the TV tube to avoid excessive parallax. Flat screen displays would have been ideal, but the cost was prohibitive for our application. Several control surface (touch panel) designs were considered. A panel using the surface wave concept was constructed and performed well, but it was difficult to manufacture and was too costly for our application. A simple and relatively inexpensive system was adopted using crossed wires.

Track balls, joy sticks, light pens, and keyboards were also considered, but were rejected. For a system where there is a high degree of operator intervention, such as a rapid or continuous interaction with a variety of control functions (i.e., increase/decrease, on/off), we believe it is faster, more secure, and easier to push on a labeled location than to use other methods to select and activate these types of functions. We recognize, however, that other applications requiring the input of large quantities of text by an operator would necessitate an associated keyboard.

There is one control aspect where a track ball or thumb wheel was considered. That was to provide the control of analog values by rotating the ball or wheel until the desired reading was obtained. However, then the ball or wheel would have to be reassigned (or "attached") to an analog each time a different one was to be controlled. Again, in a highly interactive system, this would not be as quick or as easy to do. Alternatively, many balls/wheels would be needed if each were permanently attached to a given control function. This would add to the hardware expense and complexity.

The choice of design for the touch panel system

software was much easier to determine. One could design a system where each selectable display or "panel" would correspond to the selection or activation of an associated program. This program, utilizing standard subroutines, would perform all needed functions and generate the required displays. This approach, while seemingly simple and straightforward, in fact leads to potentially large and complex programs. Moreover, to add a new display (panel), or to change an existing one, requires a program change. This in turn requires a programmer and usually debugging time.

We chose a system design which is data- (table-) driven. There is a collection of programs which utilize data to provide a variety of functions which can be specified on a panel. Thus, once we have written and debugged these programs, there is no additional programming effort required to add or change panels.

Also, these tables can be generated by a compiler program using a source language as input. We thereby achieve a fast turnaround time for changes (usually less than 20 minutes), and new panel designs can be generated by nonprogrammers using a text editor.

Panel Hardware

The hardware design for the touch panel surface is shown in Figures 4 and 5. The bottom layer (A)

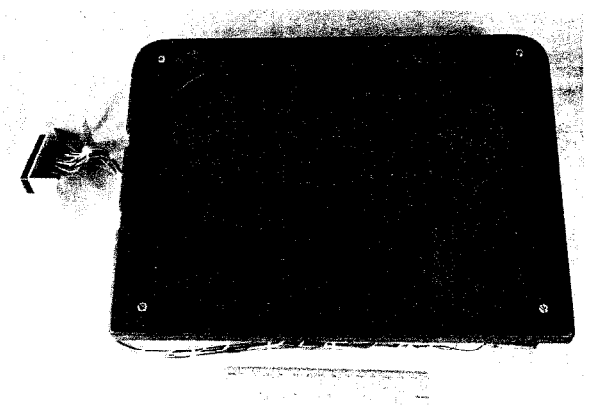

Fig. 4--Touch panel assembly.

consists of a plastic sheet which has been heat-formed in a vacuum chamber to exactly match the curvature of the video monitor surface. A grid of crossed wires is fastened to this base (B). We are presently using .150 mm diameter gold-plated spring steel wires held in tension by small springs at the edges. The wire intervals were chosen to provide an adequate spatial distribution to permit six-character labels in each button, and to prevent accidental contact closure in adjacent crossed wire locations. Using fourteen-inch (35 cm) diagonal monitors, this gave us a ten (horizontal) by thirteen (vertical) button

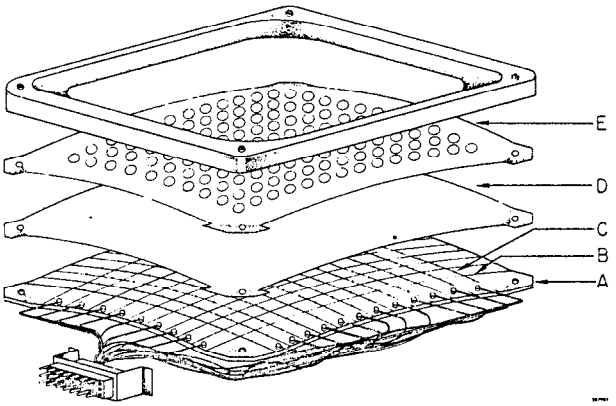

Fig. 5--Touch panel construction details.

matrix or a total of 130 button positions on a terminal.

The bottom layer of parallel horizontal wires is stretched above and in contact with the surface of the base plastic sheet. The parallel vertical wires orthogonal to these are stretched across spacers (C) to provide a vertical wire-to-wire spacing of one millimeter. Over the grid of crossed wires is a thin (.5 mm) sheet of plastic (D). This sheet provides a protective surface for the cross wires. The contacting of a pair of crossed wires produces a closed circuit of one X and one Y wire which can be detected by the computers' digital input interface.

For our particular application, we have added an additional thin (1 mm) sheet of plastic as an overlay (E) to the basic touch panel. This sheet has twelve-millimeter diameter holes punched in it which are centered on the wire cross points. Thus, an operator can readily maintain finger position over a single button or can move back and forth between adjacent buttons, activating them while looking at an oscilloscope rather than the touch panel.

A small LED was mounted in the frame to provide feedback information indicating that the software had detected a closed contact at any crossed wire position. It provides an immediate positive indication that the hardware portion of the touch panel system is functioning. Behind the touch panel assembly is mounted a small buzzer which is software-controlled. It is on while an increase or decrease function is being pushed. This provides audible feedback to the operator.

Panel Software

The software system is a distributed system where most functions are tasks which are executed under control of a Disk Operating System developed at SLAC. The touch panel portion of the system is depicted in the right-hand side of Fig. 2. The Touch Panel System software consists of a set of programs which run under the operating system utilizing various data bases in the SDS-925 computer. There are also other programs which run under the operating

system and utilize the same data bases. There may or may not be any interaction or communication between these programs and those of the Touch Panel System software. A generalized functional diagram of the flow of data and control within the software is shown in Fig. 6.

discussed below. The current available pool of panels (panel tables) is kept on a mass storage device and individual tables are read into the computer as needed. The panel table shown in Fig. 6 would be overlaid with the table which corresponds to the panel selected on the terminal for which the system is at that time

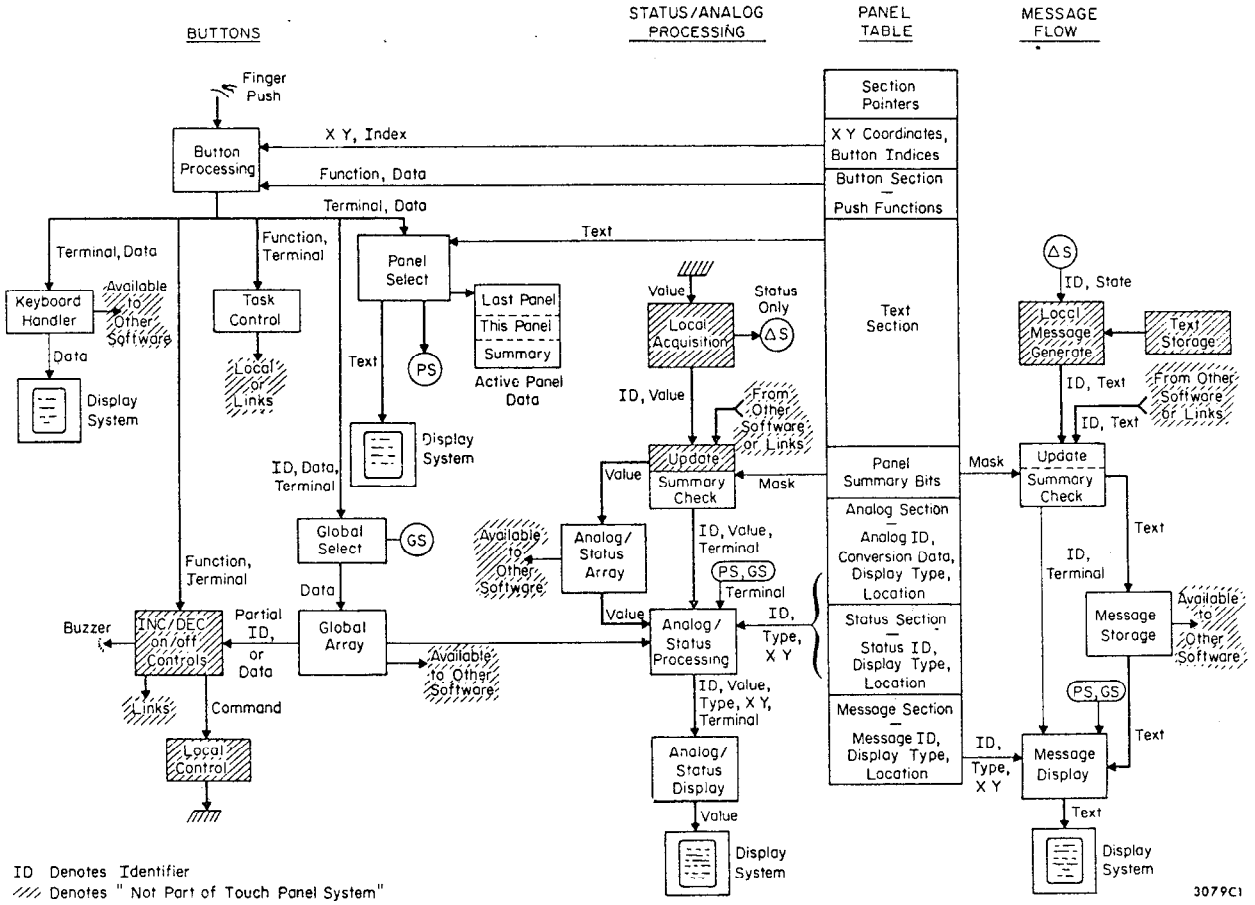

Fig. 6--Software data and control flow.

Data Bases

Central to the organization of the software system is the panel table. It, in effect, is the "panel". There is one separate table for each panel which has been entered into the system. It is a binary description of all the button functions specified for the panel; all of the analogs, status, and messages to be displayed; and the text which constitutes the basic picture of the display. The table is composed of separate sections for X and Y coordinates, buttons, analogs, status, and messages. There is a header section of pointers in order to locate these sections within the table. Also, there is one word of panel summary bits which indicate classes of analogs, status, or messages which have been included in the panel. These tables are generated by the Panel Generation Program, as processing information.

There are several other data bases which are a part of the system or are closely associated with it. For each terminal, there is in core memory a small data block consisting of "active panel data". When a panel is selected for use, the current panel identification (ID), the new panel ID, and the panel summary bits from the new panel are saved in this block. The current panel ID provides the information for subsequent loading of the panel table for use in the system. The last panel selected ID provides a means of having a button on a panel to reselect the previously selected panel on that terminal. The panel summary bits are used to prevent unnecessary overlaying (i.e., drum accesses) when analogs, status, or messages are being processed, if they have not been specified on a

currently selected panel.

Also associated with each terminal is a block of global values or data. This data is changed only by use of global select buttons on a panel. As each panel is selected, the global entries do not change. Therefore, the selection of parameters such as control rate, subdevice identification, control element number, etc., can be maintained as various panels are used.

Associated closely with the touch panel system are an analog array and a status array. The analog array is an array in core of current analog values indexed by device designation. The status array is also a core array of bit status (e.g., valve open/closed) which is also accessed by device designation. Text messages are stored consecutively in a data area of the mass storage device. There are several message categories, each having its own space allocation.

Buttons

There are five broad categories of functions with which buttons are associated as depicted in the "Buttons" column of Figure 6. All are initiated by the pushing of a button on a touch panel. The X and Y coordinates of the contact point are given to button processing software. The X and Y coordinates from the table of the panel selected on the terminal are searched for a match. If a match is found, the index associated with those coordinates is used to obtain the button data from the button section of the panel data. Processing is then passed to one of the five areas as specified in the button data.

One such function is the keyboard handler. This program, used for touch panel keyboards, saves incoming data and echoes it back via the display. Keyboard enable/disable, clear, and backspace are all available for use by the panel designer. Obviously, the keyboard display and the data entered from it could be numeric, alphanumeric, or any information set. Any program can ask the keyboard handler for the data accumulated. Data is kept for each terminal separately.

Task control is the initiation of tasks or the declaration of events for waiting tasks by button push. For example, to generate a log of all current set points, one could specify a button which initiates a set point logging program. Such a program is not actually a part of the Touch Panel System. These tasks can be in any of the computers in the control system.

Control functions use the button section of the table, which contains the necessary information to perform increase/decrease or on/off controls. This data might consist of hardware control channel or bit assignments; analog channel assignments; up, down, and read only information; rate of change (i.e., fast, medium, slow) information, etc. The data is passed to control processing software where the actual I/O is performed, or the commands are sent to another computer in the system.

Often the set of information supplied by the button section to specify the control function is not complete. Then missing data would be replaced by data specifying which global value or values are to be used. The global values are obtained from the global array (for that terminal) and are used to form a complete command.

These global values are changed via the global select button feature. In this case, a button push obtains a new global identifier (e.g., station, rate) and value from the button section of the panel table. A global processing program inserts the new value for that global in the global array for that terminal. Having replaced the global value, the program initiates analog and status tasks to update that panel display based on the new global value.

A panel select program generates the basic picture of a newly selected panel using the text data in the new panel table. It then updates the active panel data in core. Analog, status, and message tasks are initiated to initialize all of the data specified for display on the panel.

Status

Status processing begins in software which is not part of the Touch Panel System. State changes are detected in scanning programs, hardware scanners, or in other computers. Any change of state is sent with its ID to an update program which inserts the new state in the core status array for use by any program in the system (touch panel or not). For each change of state, the summary mask of each terminal is checked to see if that class of status is on display. For each terminal where a status of the class is displayed, the status section is searched for an ID match with the change-of-state ID. For each match found on each terminal, the terminal number, ID, state, and display information is passed to status display software. There the appropriate display is generated on the terminal where the status has been specified. Various types of standard displays are available: bit (0 or 1), box (■), or bar (▬). Also a special program code can be specified to provide special displays if necessary.

When a new panel or global value is selected, those programs initiate status update tasks. The display information from the status section of the panel and the current state of those status bits from the status array are individually passed to the status display software to initialize all status to be displayed on the panel.

Analogs

Analogs are processed in a completely analogous manner to the status changes with the value being stored in an analog array. The types of standard analog displays that are available include decimal conversion of the raw binary, decimal point shift, positive and negative offset, integer conversion, scaling, and horizontal or vertical bar graphs.

Text Messages

Text messages can originate anywhere in the system. All text is given one or more message types. The message-processing software updates the message files (on mass storage) by message type and then checks the summary bits for each terminal to see if a message section of that type is currently on display. For each terminal where the type is on display, the location and type of display is sent to the message display program, where the message section on the terminal is erased and replaced by a new set of text. The messages are displayed in scroll fashion, with the latest appearing at the bottom and old messages being eliminated from the top. The character size, line length, and the number of lines displayed can all be specified.

Panel Generation

Panels can be designed or changed by anyone (non-programmers) using a mnemonic language to specify functions desired as part of a panel. A list of source statements is written using an online text editor (or by keypunch). The source is used as data by a Panel Generation Program which is separate from the on-line portion of the system. It is a PL/I program which runs on the SLAC IBM Triplex System. Currently, the binary tables are loaded as files into mass storage of our control system computer for Touch Panel System use.

The Panel Generation Program consists of three subsections. There is a preprocessor which converts different forms of the source statements into one standard form or generates a series of "canned" statements from one source statement (e.g., a standard numeric keyboard). Processing continues in the compiler section, where each source statement is sequentially processed and the binary data in the various panel table sections generated. As each statement is processed, diagnostic output is generated for documentation and debugging purposes. After all statements have been processed, the table binary is assembled, listed, and punched, and a hard copy picture of the panel is generated. Several panels can be batch-processed, so a complete set of panels with documentation can be generated at any time. Two examples of panel displays are shown in Figures 7 and 8.

CONCLUSIONS AND OBSERVATIONS

We have found the use of touch panels for on/off controls and analog increase/decrease functions to be very effective. There has been no need to incorporate track balls or thumb wheels into our system. The use of an audible buzz or tone during an increase/decrease button push has been found to be highly desirable. Reliability has been good. Wire breakage at the cross points may occur after about 250,000 operations. Wire replacement and general panel maintenance can be readily done. Moreover, the cost of the panels is reasonable. Material cost is less than $70.00.

The basic design of the software has proven to be

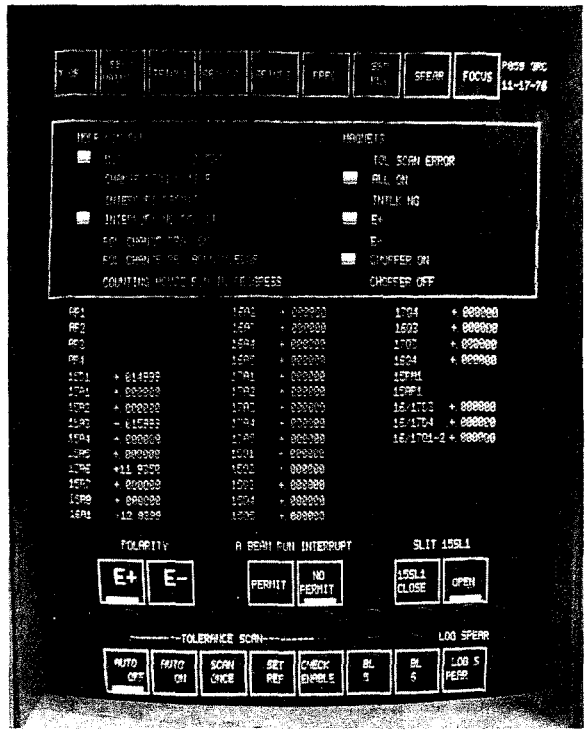

Fig. 7--Typical display.

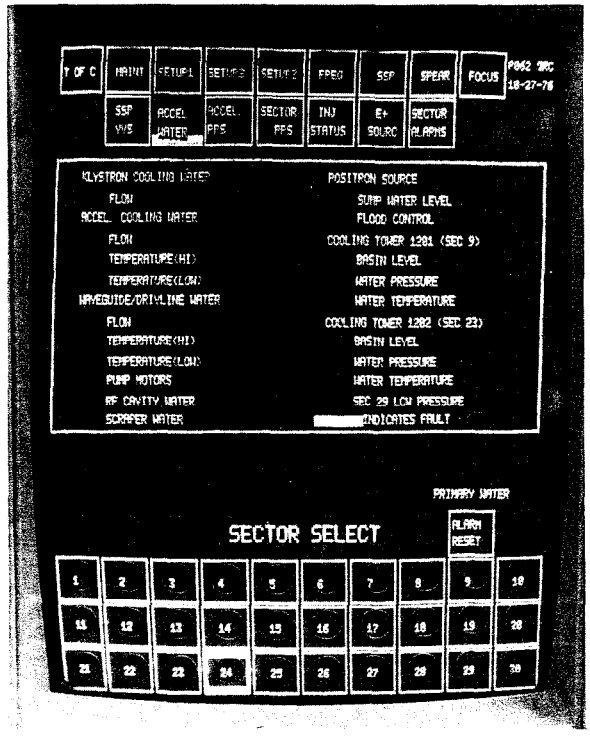

Fig. 8--Typical display.

extremely effective. After initial debugging of the touch panel software, the system has remained very stable. Panels are effectively and efficiently being designed and generated by accelerator operators rather than programmers. Errors in design have been easily detected using the diagnostics generated by the Panel Generation Program and changes quickly installed (usually in 20 to 60 minutes). We chose to implement the panel generation on the IBM Triplex System to take advantage of the text editor, mass storage, and PL/I features. However, an online panel generation system could be developed utilizing appropriate control panels selectable on the terminals.

Overall, we have found this system concept a desirable and practical system for highly interactive online control. All documentation is always current and automatically produced. It gives us a "generalized" terminal which is readily incorporated into a console or a remote location.

Although the accelerator structure at SLAC is physically different from other industrial plants, the type of operation, information, and control is not. The Touch Panel System we have developed could function equally well in other industries and applications.