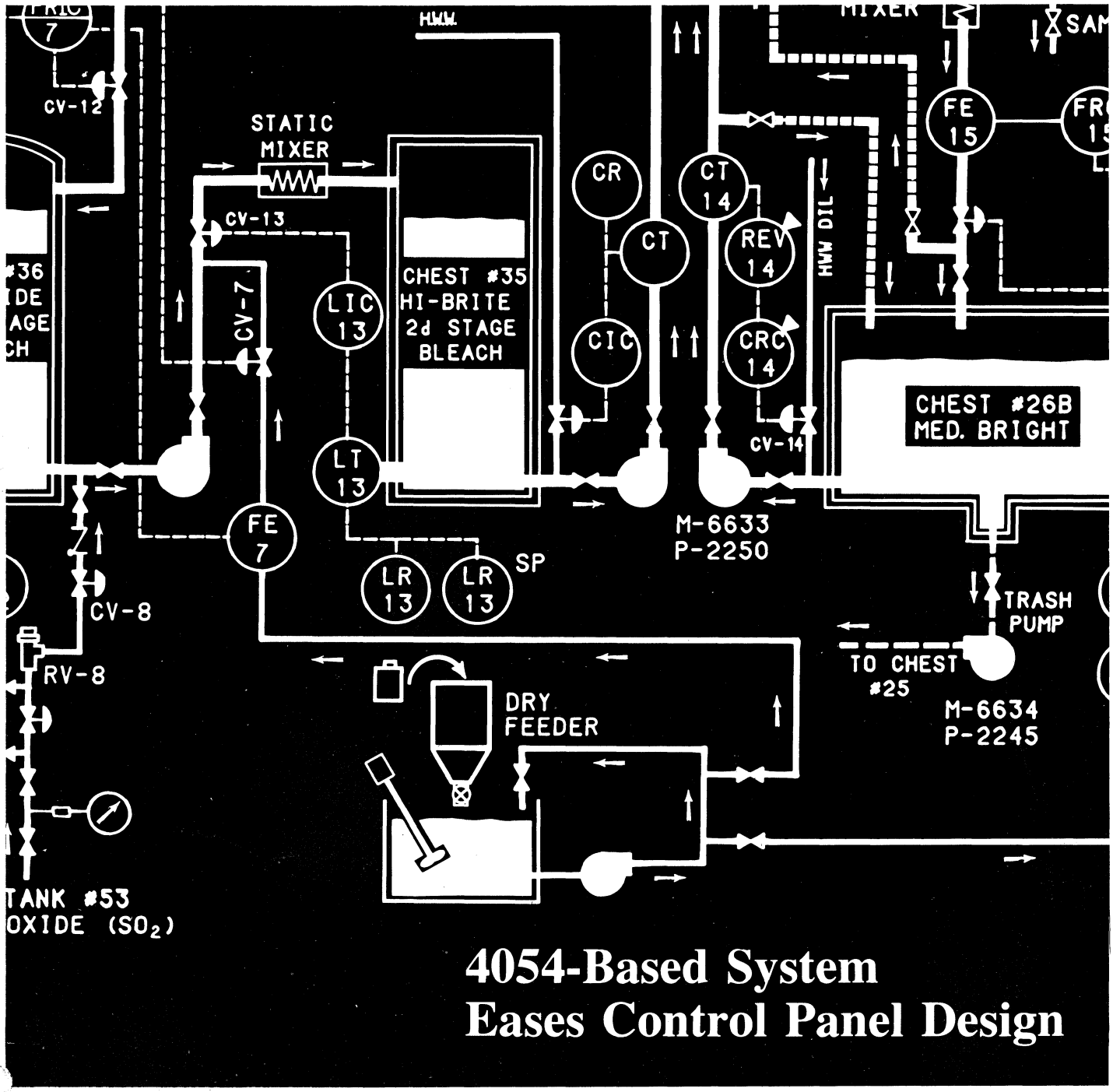
THE 4050 SERIES
APPLICATIONS LIBRARY NEWSLETTER

VOL. 5 NO. 2
MARCH-APRIL 1981

Tekniques



4054-Based System Eases Control Panel Design

Tektronix®
COMMITTED TO EXCELLENCE

Tekniques

In This Issue

TEKniques, the 4050 Series Applications Library Newsletter, is published by the Information Display Division of Tektronix, Inc., Group 451, P.O. Box 500, Beaverton, Oregon 97077. It is distributed to TEKTRONIX 4050 Series users and members of the 4050 Series Applications Library.

Publishing Manager	Ben Buisman
Managing Editor	Patricia Kelley
Editor	Terence Davis
Technical Editor	Dan Taylor
Graphic Design	John Ellis
Circulation	Rory Gugliotta

To submit articles to TEKniques or for information on reprinting articles, write to the above address. Changes of address should be sent to the 4050 Series Library serving your area (see Library addresses).

4054-Based System Eases Control Panel Design

by Terry Davis
TEKniques Staff

"One picture is worth more than ten thousand words," says the old proverb. And nowhere is this more true than in the communication of technical information. This is the driving force behind the growth of computer graphics. It is also the heart of an interesting application in the picturesque lakeshore town of Escanaba, Michigan.

There, at Interactive Graphic Controls Company, desktop computer graphics are being used to generate color graphic display or control panels and highly detailed training materials for use in complex process control environments. A TEKTRONIX 4054 Graphic Computing System with Dynamic Graphics is connected to four peripherals — a TEKTRONIX 4907 File Manager (three disks), a backlit Talos Graphic Tablet, a CalComp 1051 36″ four-pen Drum Plotter, and an NEC Spinwriter for printout.

The system is primarily used to prepare master drawings for the graphic panels and training materials. An added benefit is found in processing office work with the text processing capabilities of the Editor ROM Pack, and software developed for such things as accounting and invoicing. Then he put the system to work performing Computer-Aided Design, with a slant to instantaneous communication.

Richard Growden, a registered P.E. in Michigan with 21 years of engineering and production management experience in the paper industry, started the business 8 years ago as a consulting engineering firm. "Actually," Dick says "much more interest developed in the Graphic Panels and training materials (ie: slides, overheads, drawings, and full color printed matter) than in small engineering projects. The Graphic Panels and the training materials are highly specialized products that can be much more efficiently produced by a small company than by the large engineering firms utilized on major projects. These products are complimentary to each other in that either one may be produced by using the other as a guide. And when both types are ordered on a project, there is added economic value."

The business was renamed INTERACTIVE GRAPHIC CONTROLS to better describe the products and services being offered. The decision to install a computer graphics system to increase productivity and shorten delivery times was made when a self-contained, high resolution, graphic computer system that could support other types of equipment needed to produce the drawings required became economically available for small businesses.

The graphic display panels are color-coded, illuminated representations of process control systems (Fig. 1). They are

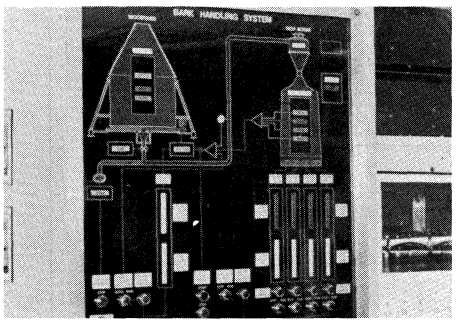


Fig. 1. At Mead Paper Co. in Escanaba, MI., the Bark Boiler panel is 36" x 42". It displays 38 pilot lights and 10 meters, in addition to 3 push-buttons, 6 selector switches, and 6 rheostats.

playing important parts in the operation of large process plants, such as the paper mills that Dick Growdon knows so well. But the same panel principles are valuable in any process control operation, from paper mills to nuclear or coal-fired power plants. The idea is to create easy-to-read panels that are graphic representations of major systems in the plant — processing fluids, for instance. The color coding separates different fluids, or system process lines, for quick identification and ease of following a line or process.

Panel Design

The heart of the design system is the 4054. Its high-resolution 19″ display allows precise screen previewing of the graphic panel layouts. The display, and the local processing, enables a detailed graphic representation of panel layouts,

so they can be perfected on the screen before plotting and production.

The creation process is a reversal of usual Computer-Aided Drafting (CAD) techniques. Where CAD typically generates finished working drawings, Dick Growdon starts with the working drawings for the plant that he's designing panels for. Copies of the plant construction drawings and instrumentation drawings are used as input to the panel design process. By digitizing the working drawings into the system, the finished panels become representations of the actual process system. This greatly improves understanding of the control panel when compared to rows of instruments and alarm areas with lots of lighted message boxes.

Lines of varying width can be created, in proportion to the line's importance to the system, or its carrying capacity (actual pipe size). Multiple fluid lines are color coded for easy separation; hidden line crossovers are used (Fig. 2). Standard

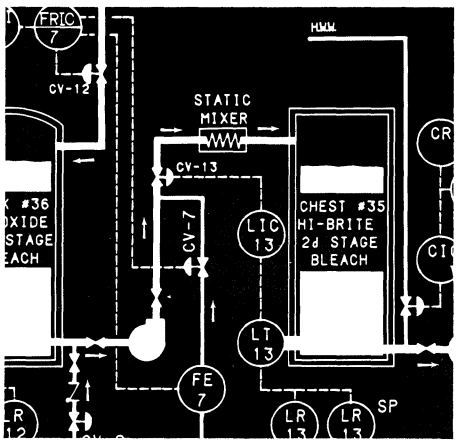


Fig. 2. Plots from the CalComp 1051 serve as artwork masters for the graphic panels.

components can be selected from a disk-based file of pumps, valves, and switches, and positioned in the drawing as desired. Special parts representations can be easily created.

Once input, the graphic layout can be adjusted on the 4054 display if necessary before it is plotted on the Calcomp 1051 Plotter. The output of the plotter is a very high resolution black and white positive drawing of the control panel display. This then becomes the master artwork for a photographic reproduction process. Photographed and reproduced as a 7 mil film negative, the heart of the final product is almost complete. Keyed color films are added to the back of the negative, and the colored print is sandwiched between two layers of plastic. The top layer is a hard, scratch-resistant plastic, while the backing layer is of flame-resistant polycarbonate Fig. 3).

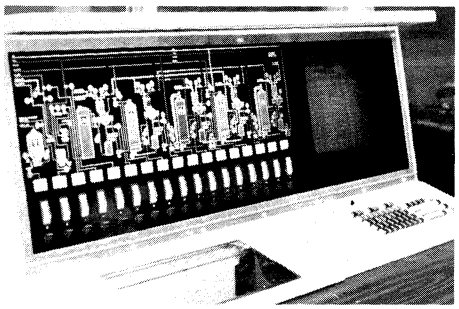


Fig. 3. This panel is in the pulp mill digester house at Great Southern Paper Co. in Cedar Springs, GA. It is 19" x 43", not including the crt display, and contains 342 lights.

The panel is completed by framing and adding lights. (Framing is often made to fit into the plant control consoles.) The end result in a process control plant is a set of color-keyed panels that represent the systems monitored by the plant control console (Fig. 4). They are very useful

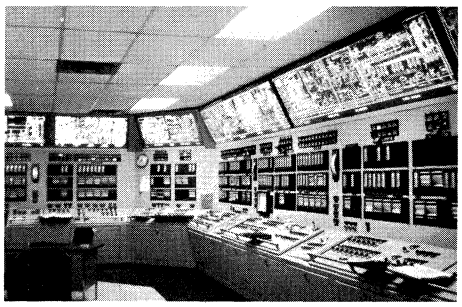


Fig. 4. At the Mead paperboard plant in Stevenson, ALA., the power and recovery room houses seven panels for a total length of 45 feet. These are static back-lighted displays.

during new employee orientation, as a refresher for vacation replacements in an area, or when promotions move people to new jobs. Some installations use simplified graphics to give visitors an easy-to-understand overview of a plant's entire operation.

Indicator lights are routinely added to the panels, for quick identification of a problem in the plant. When a panel light comes on, it's immediately next to the sensed problem area on the panel. And the panel represents, to approximate scale if possible, the actual plant layout, making location of the actual problem a snap.

Besides normal back illumination, and the installation of status indicator lights for control room panels, displays can be set up for special functions as well. Indicator lights can be complemented with switches or emergency controls, for instance. Or gauges for detailed readout can be mounted directly on the panels, next to the indicator light that signals a problem in the area monitored by the gauge. (See Fig. 5.)

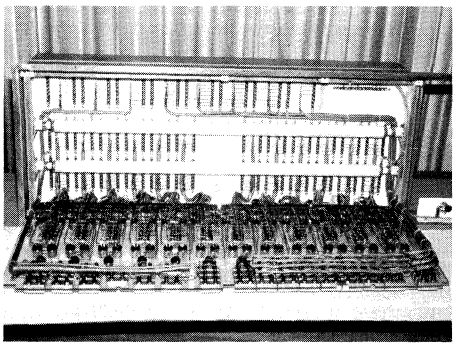


Fig. 5. At a Mead Paper Co. bleach plant, this 19" x 38" panel houses 17 meters and 17 push-buttons, built into the illuminated graphic panel.

With panels like these, the location of a problem in the plant systems can be pinpointed almost as quickly as it occurs. No grids or messages to read and decode to find a physical location. "The map-like layout is the key to understanding," says Growdon. "Messages and grids take time to decode, and time is precious in a problem situation."

Similar panels can be created by manual means, drawn on a drafting table and finished with tape and press-down symbols. But there are a number of reasons for choosing the 4054-based system; certainly speed, simplicity, and ease of use are high on the list. Consistent representation of parts is another reason, as the computer always generates the same drawing of a part. Yet the system is versatile enough to allow special parts design as needed. But perhaps the top reason is productivity. Interactive Graphic Controls is now able to accept very large jobs due to reduced lead times in meeting deliveries. This factor is important since final drawings are often unavailable until shortly before training programs are starting or production facilities begin. With the 4054, it is possible to do all of the design and office work with only one person, Dick Growdon, before calling for outside help to assemble panels.

4051 Graphics Play Intergral Part in Health Services Research and Planning

At West Virginia University Bill Detweiler (left) and Ed Bosanac (right) have instituted computer graphics to aid in health systems planning and development.

by Edward M. Bosanac, Dr. PH
and William Detweiler, Research Assistant
School of Medicine
Department of Community Medicine
West Virginia University
Morgantown, W.V.

The Office of Health Services Research (OHSR) provides health systems planning and data systems development. It is part of the Department of Community Medicine at West Virginia University in Morgantown. Although part of the University, we function very much like a public consulting firm.

For instance, the State of West Virginia has a hiring freeze. By contracting with OHSR, they may maintain control of a substantial part of the work without having the fiscal exposure of permanent staff. Many state agencies are not geared toward new development. While expert at maintaining and operating programs, new development is not one of their pursuits. Thus, they will contract with OHSR. And since OHSR is in the university setting, we have access to professionals, large systems, general planning methods and software development that would be outside of the individual agency's realm.

OHSR staff consists of two assistant professors, four other professional staff and five technicians. In the course of our work for others, we are at the same time committed to improving the statistical, analytical, and information processing capabilities of our health care researchers and planners. One of these areas is the way data is represented.

Computer Graphics Bypass Illustration Problems

It is OHSR's belief that graphic illustrations are an effective and often essential complement to tabular displays or verbal discussions of data. Charts and graphs provide the reader with "a point of view" that may have otherwise been difficult to obtain. These aids spare the reader the effort of mentally visualizing the data. In addition, maps of various kinds reveal the geographic distributions of resources and events. There is no substitute for a well designed map when conveying to the reader the idea of spatial trends.

Several adjectives characterize the value of graphic displays:

RAPID — trends in the data may be recognized at a glance;
FORCEFUL — they provide more emphasis than text or tables;
CONVINCING — specific points of interest may be clearly demonstrated;
REVEALING — relationships that may have otherwise gone unnoticed may be discovered; and
ATTRACTIVE — they offer a visual change of pace to text and tables.

Although graphics help those who publish data understand and communicate their findings more effectively, the mechanics of producing and revising an illustration is time consuming. Also, cartographic design requires specialized technical and statistical skills that may not be available to certain organizations. The staff at OHSR views computer graphics as a means of bypassing many of these problems. This is why an investment has been made in developing a system for producing charts, graphs, and maps.

Although development of the system continues, it is fully operational and is used on a daily basis. The system's design and operation enables any of the professional and clerical staff to produce a graphic display in just a few minutes.

The office's graphic system operates on several different computers. That is, the computational and graphics workloads are distributed among large mainframe computers and OHSR's Tektronix 4051 desktop graphic system. This configuration allows for interactive processing on both the mainframes and the 4051, batch submission of jobs on the large systems, and transfer of data between any of the machines. The use of more than one computer offers the user a wider range of programming options. When heavy computational power is required, the task is delegated to the mainframe. On the other hand the 4051 is used for lesser

tasks, eliminating the constraints of job management and downtime associated with the large system.

The 4051 desktop computer system performs data analysis and graphic manipulations on a stand-alone basis within the office. Moreover, because it's equipped with the RS-232 communications option, the 4051 serves as an intelligent terminal sharing data with the host computer.

One of the system's most essential peripherals is the 4662 Digital Plotter. The Plotter is not only able to accept commands from the 4051, but also from a host computer at another location. When operating in this manner there is no delay in having mainframe-generated graphics in hand since plots are not bound for output at the distant facility. In addition to the 4662's plotting capabilities, it is also used to digitize maps or drawings under the control of the 4051 or another computer.

We also have a 4907 File Manager with three flexible disk drives, and a Hard Copy Unit.

Various CRT and hard copy terminals are used at OHSR for submission of jobs to the host site. However, office personnel mostly favor our Tektronix 4025 for this task. Its capabilities for graphic display, text editing, and internal storage of information, and text editing make time online more productive.

OHSR is an affiliate of the West Virginia Network for Educational Telecomputing (WVNET). Through this organization access is provided to an ITEL AS/5 and an AMDAHL 470 V7/A. A ZETA drum plotter is available at the host site for large multicolor plots.

However OHSR has emphasized the stand-alone system, i.e., the 4051, in order that final form plots may be produced in a matter of minutes. It takes at least a working day to get a plot from the host site. The 4051 has the additional advantage of minimizing production costs. The current plotting charge by the network is $50 per hour.

The 4051 software library includes programs which we develop for the production of maps and special purpose graphs — such as population pyramids. We also rely on a multitude of programs obtained from the Applications Library. Our analytical library includes PLOT 50 Mathematics Library, PLOT 50 Statistics Library, PLOT 50 Business Planning and Analysis Vol. 1, PLOT 50 Graph Plot, and the newest PLOT 50 Statistics Disk-based Tests and Distributions.

The large system software consists of an extensive library maintained by WVNET. OHSR relies on these programs when specialized processing is required on large amounts of data. However some of these programs have been generalized and run on the 4051.

Computer Cartography Major Focus

Computer cartography, both on the 4051 and the larger systems, has been our major focus. We have several geographic base data files:

DIMECO — Dual Independent Map Encoding (county). This file was created by the U.S. Bureau of the Census and contains the boundary information for West Virginia's 55 counties.

MCD — Minor Civil Divisions. These are the largest subcounty administrative units used by the Bureau of the Census for a variety of tabulations. These are often townships or magisterial districts. There are 353 MCDs in West Virginia.

STAM — Statewide Traffic Assignment Model. The 781 statewide traffic zones in West Virginia are represented in this file. The zonal system was created by the State Department of Highways. Each zone is a functional unit based on existing traffic patterns, travel time, population, and natural boundaries.

HSR — Health Service Regions. The 37 health service regions in West Virginia were delineated by OHSR as part of a study of health care delivery patterns in the State.

TRAVEL TIME — This file is under development. It will allow travel time isochrons to be drawn about any location in West Virginia.

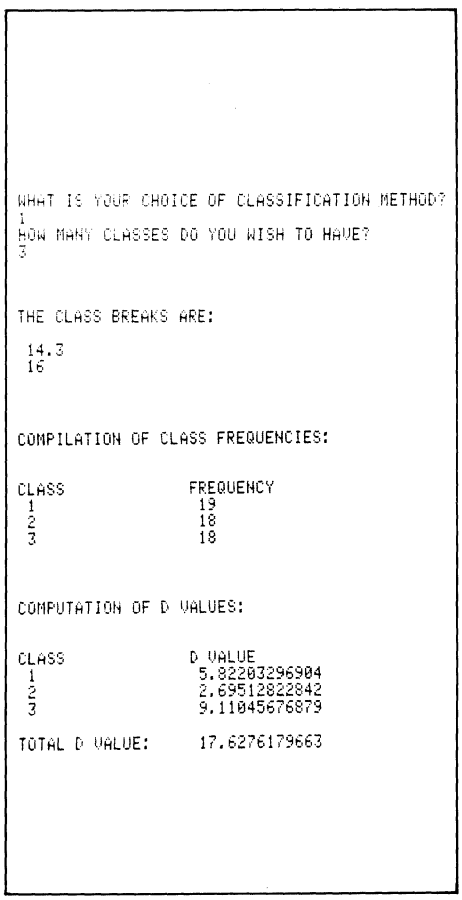
Two other files are currently under development: GBF/DIME — Standard Metropolitan Statistical Area DIME files, and ENUMERATION DISTRICT CENTROIDS.

The entire staff is versed in the use of OBS-WYLBUR and SAS. WYLBUR is a comprehensive system for manipulating text and for remote entry and retrieval of computer jobs. We enter our data on the 4051 and save it on the 4907 disk. At an opportune time we feed the data to WYLBUR and store it on the host disk.

SAS (acronym for the Statistical Analysis System and distributed and maintained by the SAS Institute) is used for data base management and summarization of large files. The output is often downloaded to the 4051 for charts and graphs.

POLYVERT, Harvard's Polygon Converter Program is the backbone of all of OHSR's geographic base files. This system allows us to input any file, such as the DIMECO, the CIA's World Data Bank of Countries, or user prepared geographic boundary files. These files are restructured to a format more suitable for plotting and POLYVRT also handles the filtering of detail. For instance, we can pull off the West Virginia portion of DIMECO and generalize the detail level of the file, reducing the number of points in the polygon so computations aren't so monumental.

We bring census and vital records data in from the host to the 4051 in tape communications mode, then ship it to the disk. To classify the data for mapping we took a class intervals program from one of the geography journals and adapted it for the 4051. Using this program we can group data by equal frequencies, or equal steps, geometric and arithmetic mean, nested mean and so on.



```
WHAT IS YOUR CHOICE OF CLASSIFICATION METHOD?
1
HOW MANY CLASSES DO YOU WISH TO HAVE?
3


THE CLASS BREAKS ARE:

 14.3
 16


COMPILATION OF CLASS FREQUENCIES:

CLASS              FREQUENCY
 1                  19
 2                  18
 3                  18


COMPUTATION OF D VALUES:

CLASS              D VALUE
 1                  5.82203296904
 2                  2.69512822842
 3                  9.11045676879

TOTAL D VALUE:     17.6276179663
```

Fig. 1. (a) A sorting program classifies data by equal frequencies

```
DO YOU WISH TO MAKE ANOTHER RUN OF THIS PROGRAM?
YES
DO YOU WANT TO USE THE SAME SET OF OBSERVED VALUES?
YES
WHAT IS YOUR CHOICE OF CLASSIFICATION METHOD?
2
HOW MANY CLASSES DO YOU WISH TO HAVE?
3

THE CLASS BREAKS ARE:

 14.5
 18.1

COMPILATION OF CLASS FREQUENCIES:

CLASS            FREQUENCY
 1                17
 2                30
 3                8

COMPUTATION OF D VALUES

CLASS            D VALUE
 1                0.0133068074364
 2                0.00372001882973
 3                0.00771823187795

TOTAL D VALUE:    0.0247450581441
```

Fig. 1 (b). A sorting program classifies data by equal steps

To put these files together, we took the shade routine published in TEKniques (Vol. 3 No. 1, "Shading Routine for Complex Shapes"), built a front end for the data, coded a routine to read the map and boundary file and another routine to handle the intervals, titles, and legends. We are now able to produce value maps, shade maps or three-dimensional maps through the 4051. The user has the option of using color rather than line patterns for shading. The output is invaluable.

For example, we graphed the spatial characteristics of the birth data in Fig. 2a. Notice the higher rates in the southern portion of the state. At this point the health services planner would take a closer look by getting a value map and looking at specific values (Fig. 2b). You simply wouldn't be able to easily pick up this information from tabular data.

In another application, we did a study of the prevalence of hypertension. When we graphed these estimates, we noticed that a higher proportion of the population from the central part of the state seemed to be at risk. You'd never pick that up in a table. At this point, other professionals could come in, look at their maps and immediately relate the hypertension locations to circumstances with which they are familiar, such as, socioeconomic data or population geography of the state.

We can rotate our three-dimensional maps. This allows relationships that may be obscured to be revealed by turning the map so many degrees.

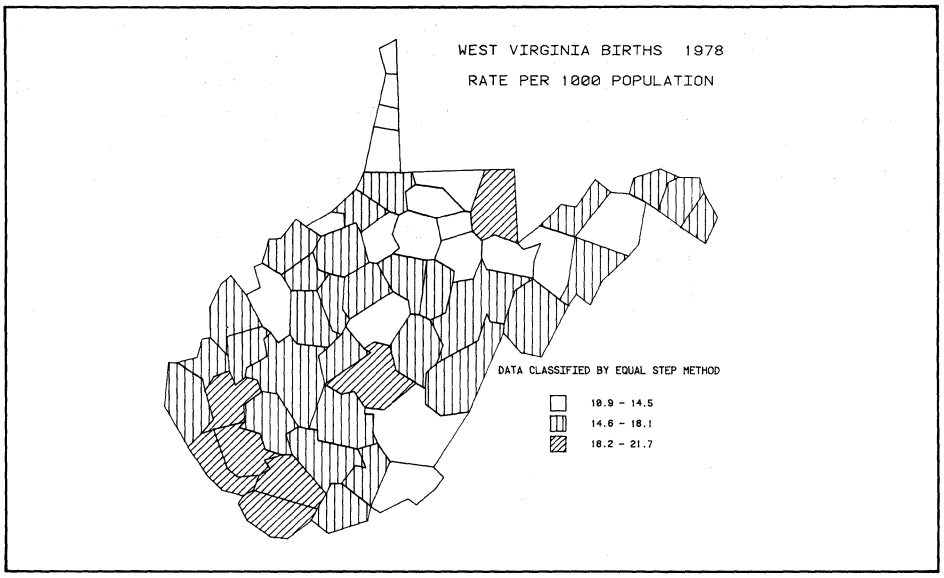


Fig. 2a. From this map, the higher birth rate in the southern part of the state is immediately apparent.

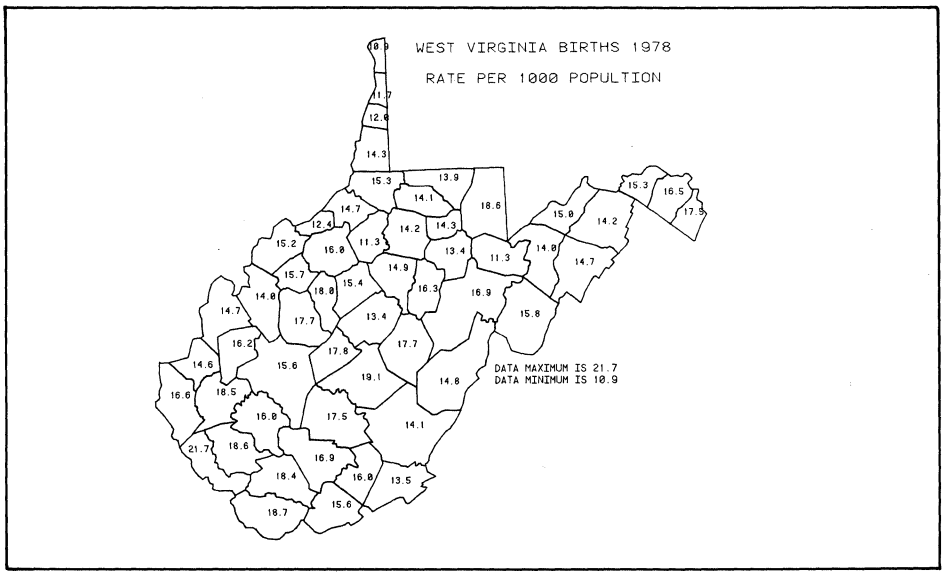


Fig. 2b. A value map translates the shaded map into hard figures.

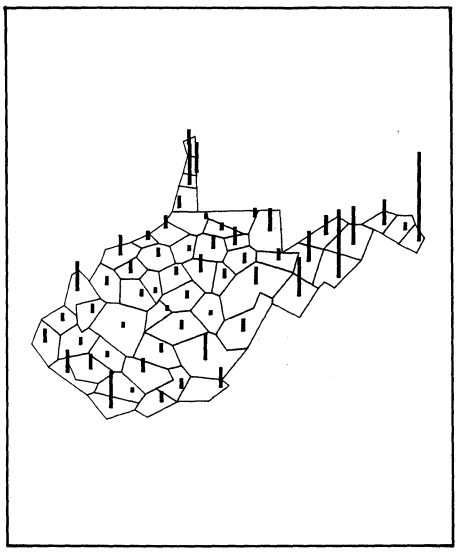


Fig. 3a. A three-dimensional data map links statistics to locale.

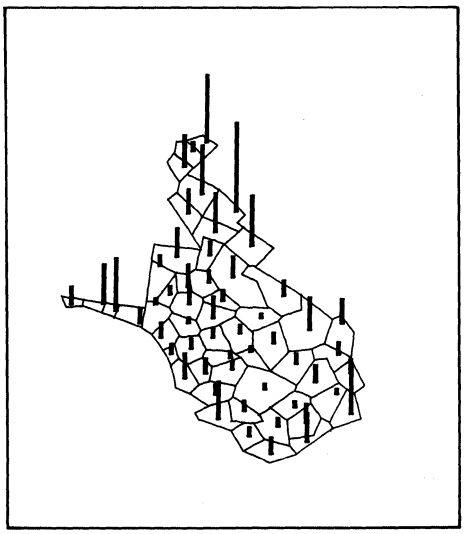


Fig. 3b. Rotating the map allows a clear look at the previously obscurred data magnitude in the north end of the state.

To produce the charts and graphs we needed, we modified an Applications Library program by adding a routine to produce logarithmic axes. The additional code overloaded memory so we divided it into three overlays. Our efforts paid off, however, in that we are now able to graphically compare transient data of different orders of magnitude. For example, it is difficult to relate rates of change in the number of U.S. medical school graduates with the number of U.S. medical schools (Fig. 4). By plotting these two sets of data on a logarithmic scale graph, we get a more precise picture of what the data really show.

The ability to change source code to fit our needs is one of the great benefits of the 4051. If we were operating strictly on a host, it would have been out of the question to manipulate the source code to tailor the graph for our purposes.

Several enhancements to our graphing system facilitate its use. Our technician, Powsiri Klinkhachorn, built a piece of hardware which allows us to transfer communication from 4025 to 4051 by the flip of a switch. This permits use of the 4662 with the 4025 online and, alternately, by simply pressing the switch, the 4662 with the 4051 online.

We will have the same sort of switching mechanism for our printer. Thus we'll have 4051 listings, or host listings through the 4025, by the flip of a switch.

4051 Streamlines Operations

The extra power of SAS for data management combined with the graphics of the 4051 has greatly relieved our staff. Before this system was in place, we cut press-on shading to highlight our maps. We had masters level people cutting and pasting on maps! With the 4051, everyone is getting their maps in a tenth of the time. The charts are more consistant, too.

Planners and researchers used to be afraid to ask for graphs and maps because they took so much time. We showed our system to them, and now they're asking for all kinds of things.

Another real advantage to the 4051 is our ability to keep working even though the host goes down. If the main system is down and we can't get a map from it, we can always turn to the 4051.

Before we took possession of our 4051, we (Dr. Ed Bosanac and Mr. Doug Nottingham, Research Associate) were charged with developing a presentation for the Charleston Area Medical Center's Residency Retreat. We had to graphically present the history and predicted future of eight selected medical specialties. We got the 4051 in at 2:00 in the afternoon along with a copy of Data Graphing (4050 Applications Library program # 51/00-9523/0). Doug and I were both familiar with the host so we started to write programs to display the data. After we wasted a few hours on that, we decided to plug in Data Graphing to see what would happen. We were delighted. It's tutorial nature and the friendliness of the 4051 led us through the graphing and we were immediately on our way. We worked all night and the report was completed the next morning, a 101 page volume of graphics!

The result of the work of the health system planners and researchers at OHSR guides the expenditure of tens of millions of dollars. And the 4051 plays an extremely heavy role.

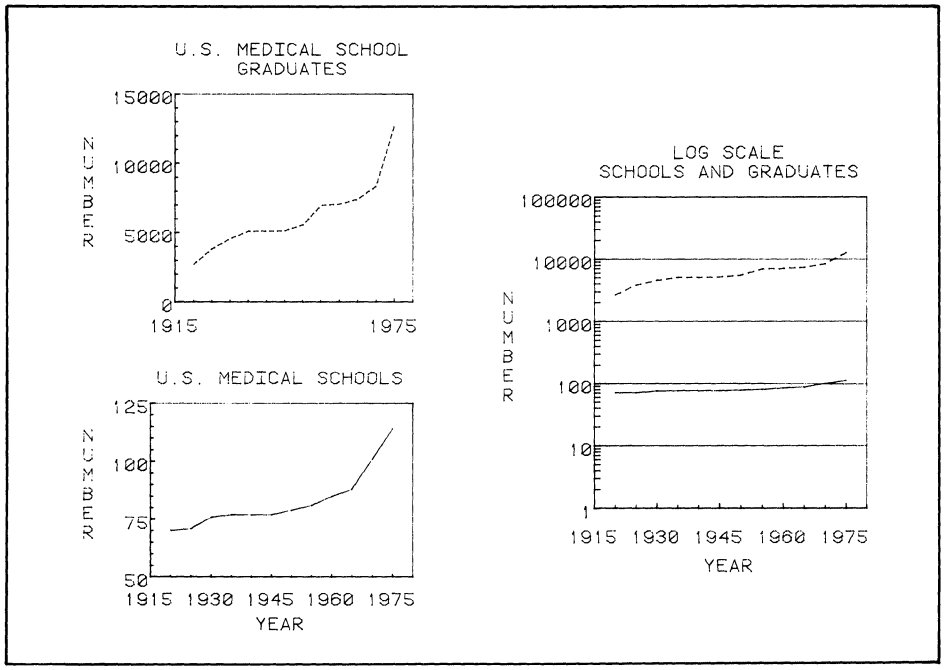


Fig. 4. Values of different orders of magnitude are easily compared using a program which overlays three graphs.

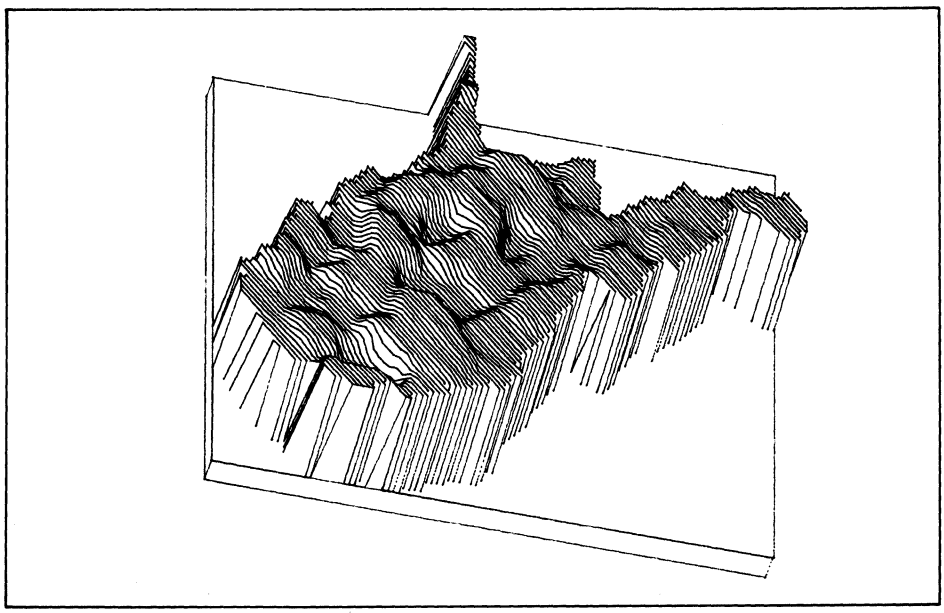
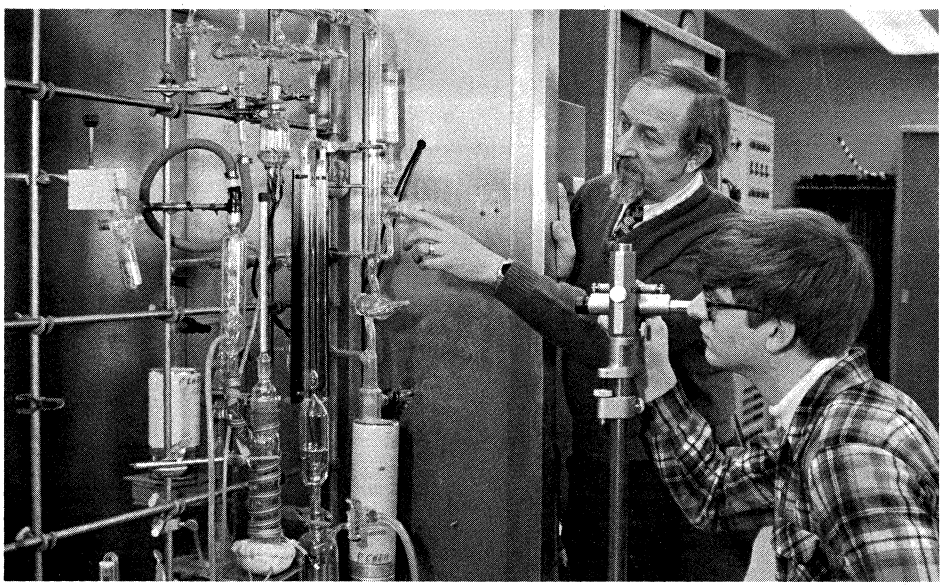


Fig. 5. Although a statewide three-dimensional value map is now available, a future project will segment it by jurisdictional polygons.

An Application of the PLOT 50 Gauss-Newton Non-Linear Regression Program in Physical Chemistry



At the University of Wisconsin-Oshkosh Dr. Gilbert Pollnow and student Marcus Raab measure vapor pressure with a typical mercury manometer. Minimizing the fixed absolute error inherent in readings from a mercury manometer (or the relative error in readings from a calibrated gauge) is easily handled by the Gauss-Newton program in PLOT 50 Statistics Vol. 4. Dr. Pollnow explains the procedure in the following article.

Gilbert F. Pollnow, Ph.D.
Department of Chemistry
University of Wisconsin-Oshkosh
Oshkosh, Wis.

Frequently, the two-constant thermodynamically derived Clausius-Clapeyron equation (2) is used to fit vapor pressure data over small temperature ranges, despite the fact that it does not take into account the decrease in the heat of vaporization with the increase in temperature:

$$\mathrm{Log_{10}}P_i = C_1 + C_2/T_i \qquad (2)$$

More satisfactory equations include the empirical and widely used Antoine equation (3):

$$\mathrm{Log_{10}}P_i = C_1 + C_2/(C_3 + t_i) \qquad (3)$$

and the so-called modified Kirchoff equation (4) which assumes a quadratic decrease in the heat of vaporization with temperature[1]:

$$\mathrm{Log_{10}}P_i = C_1 + C_2/T_i + C_3\mathrm{Log_{10}}T_i + C_4T_i \qquad (4)$$

In vapor pressure measurements, usually the temperature t_i or T_i (degrees Centigrade or Kelvin, respectively) is directly controlled as the independent variable with a known error, the pressure, P_i, being the dependent variable. Pressures measured with a typical mercury manometer have a fixed absolute error, e.g., ±0.01 mm Hg, whereas if a calibrated gauge is used, the error is usually a fixed percentage of the scale reading, e.g., 0.1%.

Clearly, the regression procedure should be chosen to minimize either the absolute or relative deviations depending on the method of experimental measurement. However, minimizing the absolute deviations in the pressures requires that the above equations be expressed in their exponential forms which are non-linear in the constants C_1, C_2, C_3, and C_4. In the event the pressure measurements have a fixed relative error and the errors are small, the least squares condition can be approximated as follows:

$$S = \Sigma((P_{io} - P_{ic})/P_{ic})^2 \cong \Sigma((d\ P_i)/P_i)^2 = \Sigma(d\ \ln P_i)^2 = \text{Minimum} = 2.303\ \Sigma(d\ \mathrm{Log_{10}}P_i)^2 \cong 2.303\ \Sigma(\mathrm{Log}\ P_{io} - \mathrm{Log_{10}}P_{ic})^2 \qquad (5)$$

Thus, minimizing the deviations in the logarithms of the pressures is for most practical purposes equivalent to minimizing the relative deviations in the pressures. Both problems are easily handled by the Gauss-Newton program #2 on the Statistics Volume 4 Tape.

In using the Gauss-Newton program, however, it is important to recognize that the input coordinates, X_i, Y_i must be in the form specified in the user's subroutine since they cannot be transformed within the Gauss-Newton Program. Thus, if the relative deviations are to be minimized, the actual logarithms of the pressures must be input as the Y_i. Also, the user must provide starting values for the constants which for the present case can be conveniently calculated from equation (2) using the conjugate coordinate extremes to determine C_1 and C_2. C_3 and C_4 can be set to zero, initially.

The following figures show the modified Kirchoff subroutines required and the graphical output with the constants superimposed via the editing keys, using data for trimethylamine[2]. The figures have all been labeled to be self-explanatory and unambiguous, except for the standard deviations which are in terms of the Y coordinates used by the program; hence, when the relative deviations are minimized the standard deviation is in terms of the $\mathrm{Log_{10}}P_i$.

In addition to the PLOT 50 references noted in the Statistics package, users may find the author's earlier article on this subject of some value since it shows in more detail how the derivatives enter a similar regression procedure.[3]

References Cited

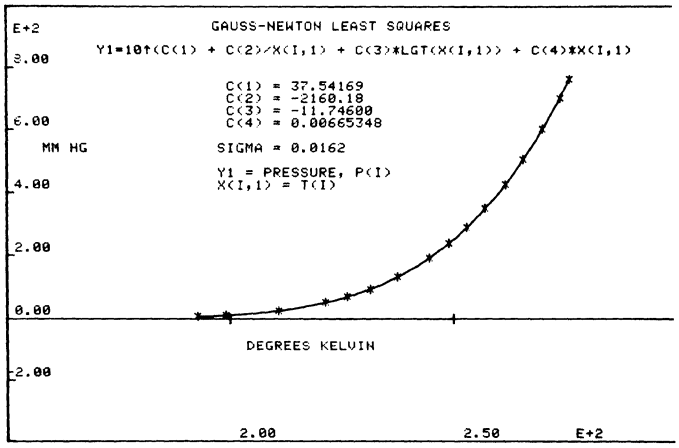
1. Vojtech Fried, Hendrik F. Hameka, and Uldis Blukis "Physical Chemistry" Macmillan Publishing Co., Inc., New York (1977) p. 165.

2. Aston, J.G. et al, J. Am. Chem. Soc. 66, 1174 (1944).

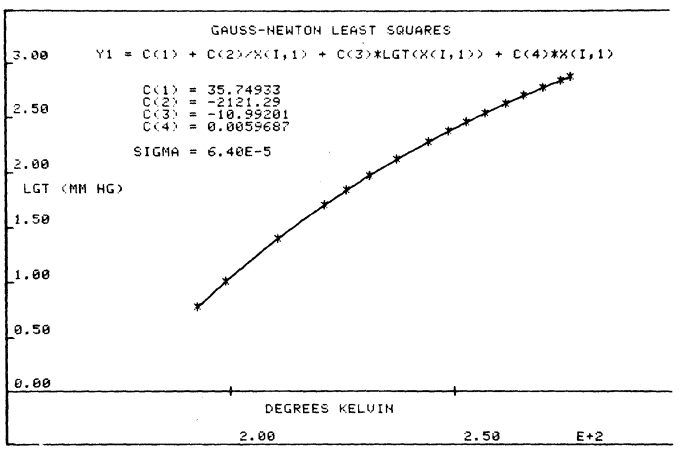
3. Pollnow, G.F., J. Chem. Ed. 48, 518 (1971).

Author's Note: In addition to the application described, I have found the same program to be very valuable for fitting polynomials which for theoretical reasons must pass through the origin, as in the case of freezing point depression experiments.

```
TERMINATION AFTER 4 ITERATIONS.
    FINAL SUM OF SQUARES = 0.00315662065504
                      DF = 12
                   SIGMA = 0.0162188692964

          J        PARAMETER          ERROR

          1        37.5416855698      0.869413214694
          2        -2160.18044601     19.8023992557
          3        -11.7459972148     0.361990590347
          4        0.00665347919572   3.113638332E-4

DO YOU WANT TO SEE THE INVERSE MATRIX: NO

DO YOU WANT THE PARTIAL CORRELATION MATRIX PRINTED: NO

***DONE***
     STARTING VALUES FOR C(1) AND C(2) WERE: 7.74788, -1343.45
GAUSS-NEWTON FIT TO EXPONENTIAL FORM OF KIRCHOFF EQUATION

DATA FOR TRIMETHYLAMINE (ASTON, J. G., ET AL, J. AM. CHEM. SOC.
     66, 1174 (1944)
```




```
TERMINATION AFTER 3 ITERATIONS.
    FINAL SUM OF SQUARES = 4.915616009E-8
                      DF = 12
                   SIGMA = 6.400270834E-5

          J        PARAMETER          ERROR

          1        35.749326111       0.520661233275
          2        -2121.29073343     11.0471896246
          3        -10.9920072355     0.220114670567
          4        0.00596868750938   2.053730342E-4

DO YOU WANT TO SEE THE INVERSE MATRIX: NO

DO YOU WANT THE PARTIAL CORRELATION MATRIX PRINTED: NO

***DONE***
     STARTING VALUES FOR C(1) AND C(2) WERE: 7.74788, -1343.45
GAUSS-NEWTON FIT TO LOGARITHMIC FORM OF KIRCHOFF EQUATION

DATA FOR TRIMETHYLAMINE (ASTON, J. G., ET AL, J. AM. CHEM. SOC.
     66, 1174 (1944)
```



Why Heats of Vaporization

All liquids (and solids at temperatures above absolute zero) spontaneously tend to evaporate as a result of the distribution of the kinetic energies of their molecules or atoms, i.e., to pass from the condensed state to the gaseous or vapor state. In a closed contained, where the vapor cannot escape, the liquid will come to co-exist in a dynamic equilibrium with its own vapor. The pressure exerted on the walls of the container by the gaseous component is called the vapor pressure and is the pressure at which no further net evaporation of the liquid will occur.

The vapor pressure is dependent upon the temperature of the liquid; as it increases, the vapor pressure and the rate of evaporation also increase. Measuring the vapor pressure of pure substances as a function of temperature suffices to determine the heat of vaporization through the Clapeyron equation (1) below and which is derived in most physical chemistry texts:

$$\Delta\bar{H}_v = T(dP/dT)(\bar{V}_v - \bar{V}_l) \qquad (1)$$

Thus, the molar heat of vaporization, $\Delta\bar{H}_v$, at any temperature, T, can be computed from the derivative of the fitted vapor pressure equation and the molar volumes of the vapor, $\bar{V}_v$, and the liquid, V_l, at the same temperature. The so-called Clausius-Clapeyron equation (2) results from the integration of equation (1) assuming the vapor to be ideal, $\bar{V}_v >> \bar{V}_l$, and that $\Delta\bar{H}_v$ is independent of temperature.

Heats of vaporization are very important in such industrial applications as refrigeration machines, internal combustion and steam engines, chemical drying processes, and others.

London Firm Develops Engineering Design Programs for 4050-Series

At Engineering Sciences Data Unit (ESDU) in London, England, qualified engineering staff produce evaluated design data for structural, aeronautical, mechanical and chemical engineers. Much of the work is painstaking, involving hundreds of time-consuming calculations; just the kind of thing for a computer application. But which computer, and which program?

Chris King is ESDU's Head of Mathematical Services and Computer Products, and he explained the way these and other questions were answered by building up a 4050 system. "In the mid 1970's we were spending a great deal of money with computer bureaus on such routine jobs as fitting a curve to a set of data points," he said. "It was the kind of thing that seemed ideal for a desk-top computer."

In 1974 ESDU had a look at the market, and after a three month survey they bought a Tektronix 4051.

"We were impressed with the system's graphics facilities," said King, "plus the fact that Tektronix had a lot of math software to back things up. Shortly after our initial purchase, we bought a 4662 Plotter, which was an instant success."

Since that time, ESDU has built the system by purchasing the 4052, the 4631 Hard Copy Unit and the 4907 File Manager. Working a steady 8-9 hours a day, five days a week, the ESDU installation deals with many thousands of individual calculations, producing for example, curves to help produce the company's Data Items.

These are sets of verified data, produced in a form that is suitable for direct use by a practicing design engineer who can, with complete confidence, take his or her desired data straight from the graphs which ESDU produce. A typical application is an aircraft noise project in which data from several microphones must be assembled and compared simultaneously.

"A COMpac is an interactive program for use on the 4050 Series systems. Based on the evaluated data in existing Data Items, each COMpac, together with its user documentation and related basis material is designed for professionally qualified users. However, it requires no special computer knowledge, since communication back and forth is in plain English.

According to King, the availability of the powerful small computer with graphics facilities and the continuing lowering of costs of this equipment have opened up many new applications of high speed computation to engineering design and analysis.

"Using their Tektronix-based COMpacs," he said, "engineers can make decisions as a program is executing, reducing the need to guess, with inadequate information, values of parameters that depend on engineering judgement. This guesswork is needed too with batch processing to avoid having to run the program several times."

Print-out and graphic display of intermediate results not only aid users at the various intermediate decision points but also make them more familiar with the technical background of the program and give them more confidence in it.

ESDU's COMpacs programs are designed to return control to the user from time to time, enabling him or her to apply (and further develop) engineering judgement.

"This also lets users leave the program for as long as they wish to seek the advice of colleagues on specific issues," said King. "In addition, these features make it easier to experiment with the effects of varying one or more design parameters, thus giving scope to exercise the art as well as the science of design."

COMpacs currently available are:

- No. 1018: The buckling of plates, clamped edges.
- No. 1010: The buckling of plates, two edges clamped and two edges simply supported.
- No. 1011: Buckling of plates, edges simply supported.
- No. 1013: Stresses and deflections in flat rectangular plates under normal pressure.
- No. 6015: Aerodynamic center of wing-fuselage combinations.
- No. 7004: Fatigue damage estimation under variable amplitude loading.
- No. 8018: Pressure change in the flow of two-phase mixtures.

A program of development to produce more COMpacs is under way at ESDU. Meanwhile, the company has produced a COMpac demonstration program. This is designed not only to show how a COMpac works, but also highlights salient features of the Tektronix 4050 Series computers (16K or bigger versions) and associated peripherals.

EDSU's address is:

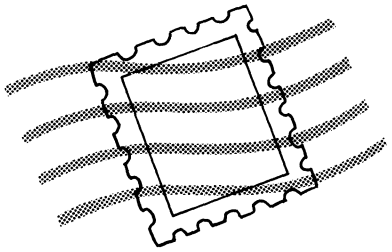
251-259 Regent Street London
WIR 7AD England
Telephone 01-437 4894

Their U.S. address is:

Suite 1037
733 15th Street N.W.
Washington, DC 20005
Telephone (202) 638-0055

Note: As a service, TEKniques may publish notices of software, hardware and services from suppliers other than Tektronix. No evaluation or endorsement by Tektronix, Inc. of these products or services is implied by such publication. Tektronix, Inc., expressly disclaims any obligation of warranty or support.

INPUT / OUTPUT



Interfacing is an area that generates a lot of questions, since there is often a variety of equipment involved as well as varying user needs. In this issue of TEKniques, two interfacing questions are addressed by Howard Sanders, Technical Support Speicalist at Tektronix, Wilsonville.

Multiple RS-232 Hookup via the GPIB

Can you make more than one RS-232 connection to a 4050 Graphic Computing System?

There are a number of reasons for wanting multiple RS-232 hookup. For instance, you might want to know if the host wants to talk without checking the Option 1 buffer for CR characters. If you have such a need, there is an answer. You can buy one of the many RS-232-to-GPIB couplers on the market and hook it to the GPIB port on the 4050.

This not only gives you an extra RS-232 port, but has the added advantage of asserting SRQ when the host is sending to you. Then, if you have ON SRQ set, your 4050 will do a GOSUB to the location specified in your ON SRQ statement, and handle the information from the host. Now you can have another program running until the host wants your 4050's attention; it will then automatically stop and listen to the Host.

The SRQ feature also works very well if you want to collect data from multiple sources, when those sources are equipped with RS-232 interfaces. The RS-232 interfaces can then be hooked to the couplers. Then, upon receiving an SRQ, the 4050 can simply poll to see which device wants attention, then go to the subroutine to handle it. In this way the 4050 can operate almost like a Host computer.

Secondary Addressing Advantages

What are the advantages of secondary addressing?

Speedier communication with peripherals is an important area that comes to mind. For instance, have you ever noticed how quickly you can get your 4050 series peripheral to respond to a command? That's a nice capability that you won't find on many machines, and it's simple because of GPIB secondary addressing. Many other machines use long strings or multiple commands to accomplish one simple operation. For example, the 4051 can command the plotter to draw with the secondary address command "PRINT @P,20:X(1),Y(1)." Other devices, ones that don't honor secondary addresses, might require the command string like "PRINT@P,32:"D",Z(1,1),Z(1,2)

The secondary address is the address that follows the primary address for the peripheral device. For instance, the command "PRINT @1,20:X,Y" tells the plotter (Primary Address 1) to do a draw (Secondary Address 20) to location X,Y. Peripherals such as the 4924 Tape Drive, the 4662 and 4663 Plotters, and the 4956 Graphic Tablet will all respond to Secondary Address commands. In addition to peripherals, this command scheme will work on the 4050 itself. In this way, using a program variable instead of a device number, you can easily change the device to which your program will output.

But not all companies use Secondary Addressing in their devices, so there is often a need to suppress the Secondary Address from automatically going out over the GPIB. Using a Secondary Address of 32 with an INPUT or PRINT statement will suppress Secondary Addressing over the GPIB.

Editor's Note

Catalogs Still Available

Once again, we'd like to remind you that 4050 Series Applications Library Catalogs are still available, and free for the asking. You should have received yours by now, but if you haven't, or if you need additional copies, just drop a line to the Applications Library office serving you. The addresses are listed at the back of each TEKniques issue.

Looking For Good Tips

Programming Tips are one of our most valued features, according to past surveys. Readers learn from them, and like the sharing of information. And contributors take pride in others learning from their tip, and get free programs as well. Do you have a good Programming Tip, or a helpful hint about programming? Let us know? We'd love to publish it as a Programming Tip or a BASIC Bit. And you get any one of 13 programs plus Programming Tips from the library for each of your published Tips or Bits. Details are listed at the end of the BASIC Bits column.

Don't Miss Out on Applications and Tips

Are you missing any TEKniques issues? Any of the issues from Vol. 4 are available by calling or writing the Applications Library office serving your area. Issues from Vol. 1 - 3 are no longer available, but the information has been combined into Application Reprints and a collection of Programming Tips. The following five categories of Application Reprints are currently available:

Engineering and Design	AX-4449
Mapping	AX-4460
Data Acquisition and Analysis	AX-4450
Business Graphing and Reporting	AX-4451
Peripherals and ROM Packs	AX-4452

If you need an article from one of the issues in Volumes 1 — 3, one of these reprint sets will likely fill your needs. To get your set, just contact your local Tektronix office, or the Applications Library serving you.

The Programming Tips collection combines 148 tips from the three volumes into one handbook, with a keyword index to help you find what you need. The handbook is available through the Applications Library. U.S. domestic price is $10.

Need Extra Copies?

Do you need more than one copy of TEKniques on hand, for students, occasional users, coworkers, or others? Just let us know how many you'd like to receive at each publication. Call (503) 685-3618.

Applications Library Changes

The Applications Library will soon be changing its software distribution methods. The Library has grown to include a substantial number of programs, with new ones being added all the time. So to simplify distribution, and to improve the cost-effectiveness for you, the software user, the following changes are in the works:

Individual programs presently in the library will be grouped into application area categories. Programs in the categories will then be combined into full tape sets, which will then be numbered, to be ordered as a set. The programs presently in the library will no longer be available on an individual basis. New programs will be available individually until they are included in an update tape, or a new volume of tapes.

Documentation for the programs will also be incorporated into sets, but will no longer include program listings. (Listings, of course, are always available from the program tape.) More information will be published in the next issue of TEKniques.

Input/Output — It's Your Column

The Input/Output column is included in the newsletter to answer any questions you might have, whether they're concerned with the 4050, peripherals, programming, the Library or TEKniques. Don't shy from asking a question. Whatever your question might be, let us know. We'll be happy to print your question and answer. (And, chances are, lots of others have similar questions and will benefit as well.)

Picture Transfer through GMX Files

by **Bob Parent**
Tektronix, Inc.
Wilsonville, OR
with Patricia Kelley
TEKniques Staff

TEKniques Vol. 4 No. 1 described the PLOT 50 Standard File, a specially formatted source data file which any PLOT 50 Standard File Compatible software package can access. The PLOT 50 Standard File was specifically designed to communicate data between analysis applications.

An adjunct to the Standard File are PLOT 50 Graphic Model Exchange Files (GMX), — specially formatted files to allow graphic data to be accessed by more than one PLOT 50 software package. There are two types of GMX files: Picture files and Font files. Picture files store picture definitions, and Font files store character font definitions.

PLOT 50 Picture Composition, described in TEKniques Vol. 5 No. 1, is the first PLOT 50 package implementing the GMX format for Picture files. Packages which output and input GMX files are identified as PLOT 50 GMX File compatible.

Care was taken to make the GMX format easy to understand and easy to generate. So users writing their own graphic applications can access GMX files or create GMX files, this article outlines the GMX Picture file format.

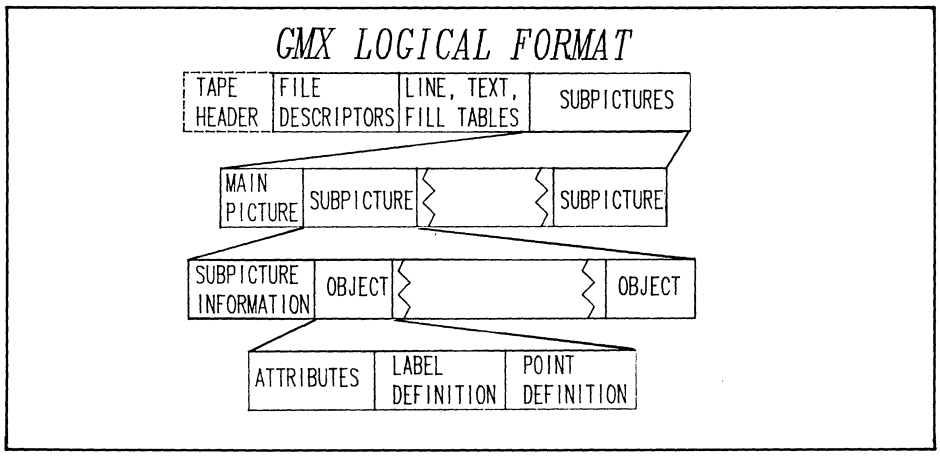
GMX Picture File Overview

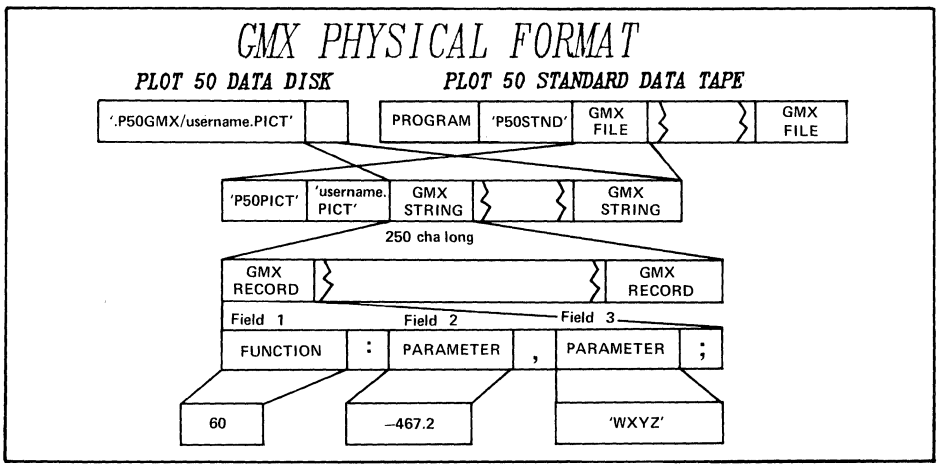
The basic structure of a GMX picture (or any picture) consists of small areas known as subpictures. And each subpicture is made up of graphic objects. For example, a layout of a house could consist of subpictures of the various rooms. These subpictures, in turn, would be composed of tables, chairs, and so forth. Or, a GMX picture may consist of only one subpicture.

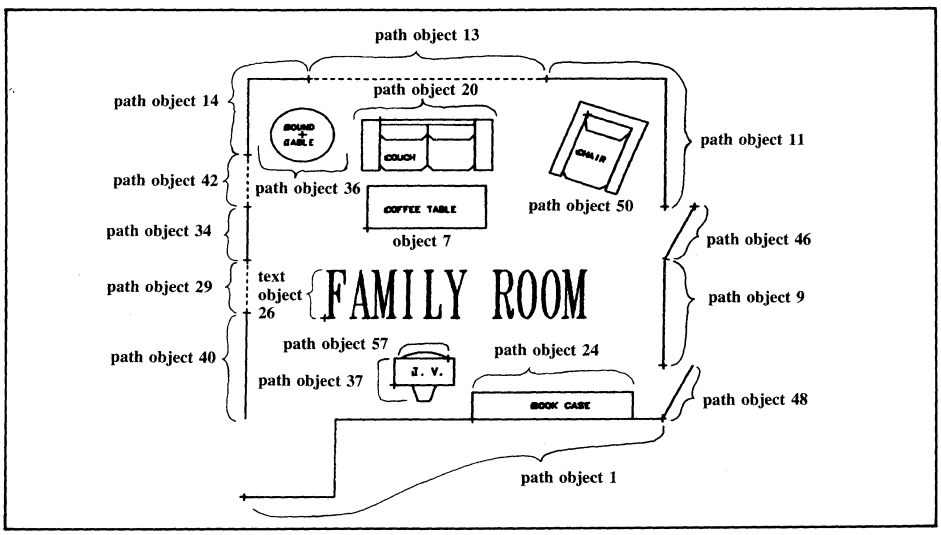
Thus, a GMX Picture file may be a series of subpictures (or one subpicture) which contain graphic objects. Put all the subpictures together and you have your master picture. But how are these graphic objects represented so different programs may access the file and display them?

Think of the GMX picture file(s) as a sequence of graphic "actions." These actions range from the most elementary steps which construct the graphic objects to an action which brings in an already-constructed subpicture. Each action has been assigned a **function code** which tells the program what to do with it and any parameters following.

Refer to the function codes and the file format illustrations while we take a look at a GMX Picture file.







The Graphic Model Exchange (GMX) file generated to create this drawing is listed at the end of this article. Annotations relate code to the room's components.

GMX Functions

GMX File Compatible packages support 42 functions (actions). The following describes each function and its parameters. The quantity, type and order of the parameters for each function is always the same. A dollar sign ($) after a parameter name indicates it is a string. An example is given for each function and parameter(s).

Function #	Function	Parameters	Example
10	**OPEN FILE**	**TYPE, NAME$**	**10:0, 'MYPIC';**
	This function always occurs at the beginning of the file. TYPE is the GMX file type, which is 0 for picture files. NAME$ is a user-defined name for the picture (this should not be confused with the name of the GMX file on the disk or tape).		
11	**CLOSE FILE**	**none**	**11;**
	This function occurs at the end of the GMX file.		
12	**FILE DESCRIPTOR**	**STRING$**	**12:'Data from 12-25-52';**
	This function allows strings of user data describing the file to be placed in it. The string may contain any printing characters (with ' replaced by ''). File descriptors must precede any tables or subpictures.		
20	**OPEN LINE TABLE**	**none**	**20;**
	This function starts the definitions of line styles. It must precede any subpictures; however, the three tables (line, text, and fill) may be in any order.		
22	**DEFINE LINE STYLE**	**INDEX, DASH, WIDTH**	**22:4,95,2.0;**
	This function defines the line style number specified by INDEX. DASH and WIDTH describe the appearance of the line. The semantics of DASH and WIDTH are not defined by GMX. This function can only be issued between opening and closing the line table.		
21	**CLOSE LINE TABLE**	**none**	**21;**
	This function closes definition of line styles.		
30	**OPEN TEXT TABLE**	**none**	**30;**
	This function begins definition of text styles, similarly to OPEN LINE TABLE.		
32	**DEFINE TEXT STYLE**	**INDEX,FONT$**	**32:5,'ITALICS';**
	This function defines the text style number specified by INDEX. FONT$ is the name of the character font. The set of legal names, and the appearance of the fonts are not specified by GMX.		
31	**CLOSE TEXT TABLE**	**none**	**31;**
	This function ends the text style definitions.		
40	**OPEN FILL TABLE**	**none**	**40;**
	This function begins definition of fill styles, similarly to OPEN LINE and OPEN TEXT TABLES.		
42	**DEFINE FILL STYLE**	**INDEX,P1. . .P5**	**42:1,2.4,13,3,3,2.5;**
	This function defines the fill style number specified by INDEX. The five parameters describe the fill style. The semantics of the fill parameters is not specified by GMX.		
41	**CLOSE FILL TABLE**	**none**	**41;**
	This function ends the fill style definition.		
50	**OPEN SUBPICTURE**	**NAME$**	**50:'';** **50:'HOUSE';**
	This function begins the definition of a subpicture. The first subpicture in the file has no name; thus, the function record looks like the first example. Note that the string parameter must still be present.		
51	**CLOSE SUBPICTURE**	**none**	**51;**
	This function terminates definition of the subpicture. It must be given before another subpicture can be opened.		
	The following functions may only be issued at the beginning of the subpicture (before any objects within the subpicture are defined).		
52	**SET EXTENT**	**LOWX, HIGHX, LOWY, HIGHY**	**52:10.5,35.2,1.5E3,1.8E3;**
	This function specifies the extent of the subpicture. This information can be of value to the receiving application for scaling the picture or subpicture.		
53	**DEFINE PICK POINT**	**X, Y**	**53:16.2,254.3;**
	This function defines a pick point for the subpicture. Pick points are used to point to a subpicture. It is often useful for subpictures to have the same pick point, i.e., center of subpicture relative to EXTENT (above).		
	The following functions are used to define the objects in a subpicture. They may only be used after any EXTENT or PICK POINT functions for the subpicture are given.		
59	**OPEN POINT OBJECT**	**ID,TYPE**	**59:37,1000;**
	This function begins definition of a POINT object. ID is a number which may be used to identify the object (all object ID numbers should be unique). TYPE is the type of point in the point object. The type of point is application dependent.		
60	**OPEN SUBPICTURE OBJECT**	**ID,X,Y,NAME$**	**60:3645,30.5,24.2,'RESISTOR';**
	This function begins definition of a SUBPICTURE REFERENCE object. ID is the object ID, as described above. X and Y give the location in picture space where the origin in subpicture space is mapped. NAME$ is the name of the subpicture (see OPEN SUBPICTURE).		
61	**OPEN TEXT OBJECT**	**ID,X,Y,STRING$**	**61:216,30.5,20.4,'65 ohms';**
	This function begins definition of a TEXT object. ID is the object ID. X and Y give the location of the lower left corner of the first character. STRING$ is the text string, which may be up to 72 characters long.		
62	**OPEN PATH OBJECT**	**ID**	**62:49761;**
	This function begins definition of a PATH object. ID is the object ID.		

Function #	Function	Parameters	Example
63	**OPEN POLYGON OBJECT**	**ID**	**63:49762;**
	This function begins definition of a POLYGON object. ID is the object ID.		
64	**CLOSE OBJECT**	**none**	**64;**
	This function ends the definition of an object.		
	The following object information functions, if used, must occur at the beginning of the object definition (i.e., before the labels and coordinates).		
65	**SET TRANSFORM**	**X1,X2,Y1,Y2**	**65:3,0,0,3;**
	This function defines the rotation and scaling of a TEXT or SUBPICTURE REFERENCE object. X1 and Y1 give the location in picture coordinates where the point (1,0) is mapped; X2 and Y2 give the location where the point (0,1) is mapped. SET TRANSFORM may only occur in a TEXT or SUBPICTURE object.		
66	**SET LINE STYLE**	**INDEX**	**66:3;**
	This function defines the line style for the object. INDEX is the number of the line style.		
67	**SET PEN NUMBER**	**INDEX**	**67:2;**
	This function sets the pen number for the object. INDEX is the desired pen number. This function can be used when plotting the picture to change pens.		
68.	**SET TEXT STYLE**	**INDEX**	**68:1;**
	This function defines the text style for the object. INDEX is the number of the text style.		
69	**SET FILL STYLE**	**INDEX**	**69:3;**
	This function defines the fill style for the object, INDEX is the number of the fill style.		
70	**SET OBJECT DESCRIPTOR**	**STRING$**	**70:'Monday';**
	This function allows a string of user-defined information to be stored in an object.		
71	**SET OBJECT EXTENT**	**LOWX, LOWY, HIGHX, HIGHY**	**71:10,15,13,17;**
	This function specifies the extent of the object.		
72	**SET PICK POINT**	**X,Y**	**72:37.542,13.6;**
	This function defines a pick point for the object. It is often useful for all occurrences of an object to have the same pick point relative to the subpicture.		
73	**SET LEVEL**	**INDEX**	**73:5;**
	This function assigns a level number to the object. Level numbers are useful for segregating different parts of the picture for manipulation. INDEX is the level number.		
	The following functions allow labels for an object to be defined.		
80	**OPEN LABEL DESCRIPTION**	**none**	**80:**
	This function begins the definition of the labels for an object.		
82	**SET LABEL STYLE**	**INDEX**	**82:1;**
	This function sets the character style for labels following. INDEX is the number of the character style.		
83	**SET LABEL TRANSFORM**	**X1,X2,Y1,Y2**	**83:1.5,1.3,—1.3,1.5;**
	This function sets the label transform for following labels (see SET TRANSFORM for description of mapping).		
84	**SET LABEL PEN**	**INDEX**	**84:2;**
	This function sets the pen number of the following labels. INDEX is the number of the pen.		
85	**ENTER LABEL**	**X,Y,STRING$**	**85:78.2,14.26,'Budget';**
	This function defines a label. The label will take on the characteristics set by the most recent label attributes in the current object.		
81	**CLOSE LABEL DESCRIPTION**	**none**	**81;**
	This function ends definition of labels.		
	The following functions describe the coordinate data for the object.		
90	**OPEN POINT DESCRIPTION**	**none**	**90;**
	This function begins definition of the coordinate data for the object. The point description must follow the label description if one is present. Point descriptions are only allowed in PATH, POINT, and POLYGON objects.		
92	**ENTER VERTEX**	**X,Y**	**92:56.23,1.5E02;**
	This function defines a vertex in the current object. The first and last points in the point description must always be vertices. The first and last points in the point description of a POLYGON object must have the same coordinates.		
93	**ENTER ARC POINT**	**X,Y**	**93:16.2,43.764;**
	This function defines an arc point in the current object. An arc point must be preceded and followed by vertices. Arc points are not allowed in POINT objects.		
94	**ENTER CURVE POINT**	**X,Y**	**94:57.87,12.43;**
	This function defines a curve point in the current object. A curve point must be preceded and followed by either curve points or vertices. Curve points are not allowed in POINT objects.		
91	**CLOSE POINT DESCRIPTION**	**none**	**91;**
	This function ends the point definition.		

GMX Picture Files on Tape

First, let's look at the preliminary requirements for storing a GMX picture file on tape.

GMX files may reside on any PLOT 50 Standard Data Tape. A PLOT 50 Standard Data Tape has the following format:

File 1. An ASCII program. When the tape is autoloaded, this file identifies the tape to the users as a PLOT 50 Standard Data Tape.

File 2. This is a binary data file. As its first item it must contain the string: P50STND. This string identifies the tape as a PLOT 50 Standard Data tape. Without this file, the tape cannot be read by a PLOT 50 Standard File Compatible/GMX File Compatible program. Further contents of this file are ignored by the program.

File 3 and following. The remainder of the files are PLOT 50 GMX, Standard or Package dependent Files. Some or all of these files may be GMX files. A GMX picture file on tape has two additional strings at the beginning which are not found on the disk. The first string is P50PICT which identifies the file as a GMX picture file. The second is the username for the file, which may be up to ten characters long with .PICT as the extension. When GMX files are put on or taken from a tape by a PLOT 50 GMX file Compatible program, they are referred to by name rather than number.

GMX Picture Files on Disk

The only requirement peculiar to the disk is that the file identifier of the GMX picture file must be:

@P50GMX/username.PICT

Thus, all GMX files on the disk reside in the library P50GMX. The extension .PICT is used to distinguish between picture files and other types of GMX graphic files. For example, .FONT would indicate a GMX font file.

Physical GMX Picture File

Physically, a GMX picture file is a sequential binary file, consisting only of strings, which describes the subpicture(s). Each string may be up to 250 characters long. Each string consists of one or more GMX records concatenated together.

Each record contains an action which defines or contributes towards defining one of the graphic objects which make up the subpicture. Thus, a GMX record consists of one or more fields which store a numeric function code, zero or more numeric parameters and/or zero or more string parameters. The quantity of numeric or string parameters for a given function code is always the same.

The function code specifies the type of action, e.g., OPEN TEXT OBJECT (Function 61), and the parameters in the record specify the other information necessary to fully define the action, i.e., the string and its location.

Each record is terminated by a semicolon (;). If the record consists of only a function code with no parameters, the semicolon immediately follows the function code, e.g., Function 11, CLOSE FILE. Otherwise, the function code is separated from the parameters by a colon (:). The parameters are separated from each other by a comma (,).

Spaces may be used between records and between fields within a record. A record may be broken across strings provided the break occurs between fields, i.e., after the field separator.

The form of a numeric parameter may be anything which is acceptable to the VAL function in BASIC, including signs, decimal points, and exponents. String parameters are enclosed in single quotes ('). A single quote in a string must be replaced by two single quotes (''). Note the examples of different formats included with the function codes.

GMX File Descriptor

A GMX File may contain a file descriptor. This is an application dependent character string which is placed at the beginning of the GMX file. It is used to store information which cannot be stored in GMX format in some other way.

GMX Tables

In order to improve portability between programs while still allowing a GMX picture to be sent to different types of devices, parameters of some object attributes are specified as integers. The meaning of these integer values can be established by the application producing the GMX file so that the application receiving the GMX file may reproduce the picture faithfully. The meaning of these integer values are described in a set of optional tables at the beginning of the GMX file (but following the File Descriptor, if any). The tables supported by GMX files are:

Line Style
Text Style
Fill Style

However, some PLOT 50 GMX File Compatible programs have a fixed set of line styles and character fonts, and don't use any fill style. Thus, these programs will not produce any tables preceding the GMX files they put out, and they will ignore any tables on files coming in.

GMX Subpictures

A GMX subpicture may begin with two attributes:

SET EXTENT
DEFINE PICK POINT

SET EXTENT provides the receiving program with the coordinates of the window of the subpicture, thus informing it of its scale.

By designating coordinates within a subpicture as its PICK POINT, a program can refer to this point in moving and placing subpictures.

Once the EXTENT and PICK POINT actions are defined, the objects within the subpicture are described.

GMX Graphic Objects

All GMX subpictures are constructed from five types of graphic objects:

PATH
POLYGON
POINT
TEXT
SUBPICTURE REFERENCE

While there are some differences in the format between these five types, they all follow the same general form.

The first record specifies which object will be defined. Next, there may be attributes which describe some of the characteristics of the object necessary for producing its image, e.g., line style. These need not be present, in which case a default value is implied.

After the attributes, there may be a section where any labels are defined. Each object may have its own labels. The labels themselves may have certain attributes, i.e., text style.

The last part of an object description is the point definition section. This section is only present for PATH, POLYGON and POINT objects. It contains the coordinates which describe the graphical part of the object.

Object Attributes

SET TRANSFORM may only occur in a TEXT or SUBPICTURE object. SET TRANSFORM allows text or a subpicture to be represented in the picture in a different size or orientation than their original definition. To do this, a linear transformation is applied to the text string or subpicture. To specify this transformation, you simply supply the coordinates in picture space of the locations where two points in the object's coordinate space are mapped. The two points of the TEXT object are lower right and upper left of the first character. The two points of the SUBPICTURE object are lower right and upper left of its window.

SET LINE STYLE is used with PATH or POLYGON objects.

SET PEN NUMBER may be used with any of the objects.

SET TEXT STYLE would be used with TEXT objects only.

SET FILL STYLE would be used to shade a POLYGON.

SET OBJECT DESCRIPTOR. The information stored here would not necessarily be printed. This function is application dependent.

SET OBJECT EXTENT. Like Function 52, this sets the window but of the object within the subpicture.

SET PICK POINT. The point set would be used to point to an object. It is often useful for all occurrences of an object to have the same pick point relative to the subpicture.

SET LEVEL. Using level numbers, a program can display only those objects contained on the same level. Very handy for manipulating parts of subpictures.

Object Labels

The Label Definition Section follows the Object Attribute section. GMX supports three attributes for labels:

Style
Transform
Pen

Style is self-explanatory. Refer to SET TRANSFORM in the Object Attributes for an explanation of Transform. Pen refers to the pen number.

Object Points

The Point Definition Section of the POINT object describes a set of individual locations. Each of these locations is called a vertex. No lines, curves, or arcs may be used within a POINT object.

For PATH or POLYGON objects, there may be three types of points: vertex, arc and curve. These locations are connected together according to certain conventions to create a contour. A POLYGON object describes a closed contour, while the contour described by a PATH is not necessarily closed.

When two vertex points occur together in a PATH or POLYGON, they are meant to be connected by a straight line. An arc point, which must always be immediately preceded and followed by vertices, implies that a circular arc be drawn through the three points. A sequence of curve points, which must always be preceded and followed by vertices, implies that a smooth curve is passed through the two vertices and the curve points. The net result is that a PATH or POLYGON defines a set of segments of various shapes which are connected end-to-end in an unbroken sequence.

Summary

This has given you an overview of the PLOT 50 Graphic Model Exchange File (GMX) structure. To summarize the commands and their sequence:

```
OPEN FILE (10)
  FILE DESCRIPTOR (opt) (12)

  OPEN LINE TABLE (opt) (20)
    DEFINE LINE STYLE (22)
  CLOSE LINE TABLE (21)

  OPEN TEXT TABLE (opt) (30)
    DEFINE TEXT STYLE (32)
  CLOSE TEXT TABLE (31)

  OPEN FILL TABLE (opt) (40)
    DEFINE FILL STYLE (42)
  CLOSE FILL TABLE (41)

  OPEN SUBPICTURE (50)
    SET EXTENT (opt) (52)
    DEFINE PICK POINT (opt) (53)

    OPEN POINT OBJECT (opt) (59)
      SET PEN NUMBER (opt) (67)
      SET OBJECT DESCRIPTOR (opt) (70)
      SET OBJECT EXTENT (opt) (71)
      SET PICK POINT (opt) (72)
      SET LEVEL (opt) (73)

      OPEN LABEL DESCRIPTION (opt) (80)
        SET LABEL STYLE (opt) (82)
        SET LABEL TRANSFORM (opt) (83)
        SET LABEL PEN (opt) (84)
        ENTER LABEL (85)
      CLOSE LABEL DESCRIPTION (81)

      OPEN POINT DESCRIPTION (90)
        ENTER VERTEX (92)
        .
        .
      CLOSE POINT DESCRIPTION (91)
    CLOSE OBJECT (64)

    OPEN SUBPICTURE OBJECT (opt) (60)
      SET TRANSFORM (opt) (65)
```

```
    SET LINE STYLE (opt) (66)
      (if any lines within the subpicture
      used the default, this command would
      override the default)
    SET PEN NUMBER (opt) (67)
      (override default)
    SET TEXT STYLE (opt) (68)
      (override default)
    SET FILL STYLE (opt) (69)
      (override default)
    SET OBJECT DESCRIPTOR (opt) (70)
    SET OBJECT EXTENT (opt) (71)
    SET PICK POINT (opt) (72)
    SET LEVEL (opt) (73)
    OPEN LABEL DESCRIPTION (opt) (80)
      SET LABEL STYLE (opt) (82)
      SET LABEL TRANSFORM (opt) (83)
      SET LABEL PEN (opt) (84)
      ENTER LABEL (85)
    CLOSE LABEL DESCRIPTION (81)
  CLOSE OBJECT (64)

  OPEN TEXT OBJECT (61)
    SET TRANSFORM (opt) (65)
    SET PEN NUMBER (opt) (67)
    SET TEXT STYLE (opt) (68)
    SET OBJECT DESCRIPTOR (opt) (70)
    SET OBJECT EXTENT (opt) (71)
    SET PICK POINT (opt) (72)
    SET LEVEL (opt) (73)
    OPEN LABEL DESCRIPTION (opt) (80)
      SET LABEL STYLE (opt) (82)
      SET LABEL TRANSFORM (opt) (83)
      SET LABEL PEN (opt) (84)
      ENTER LABEL (85)
    CLOSE LABEL DESCRIPTION (81)
  CLOSE OBJECT (64)

  OPEN PATH OBJECT (opt) (62)
    SET LINE STYLE (opt) (66)
    SET PEN NUMBER (opt) (67)
    SET OBJECT DESCRIPTOR (opt) (70)
    SET OBJECT EXTENT (opt) (71)
    SET PICK POINT (opt) (72)
    SET LEVEL (opt) (73)
    OPEN LABEL DESCRIPTION (opt) (80)
      SET LABEL STYLE (opt) (82)
      SET LABEL TRANSFORM (opt) (83)
      SET LABEL PEN (opt) (84)
      ENTER LABEL (85)
    CLOSE LABEL DESCRIPTION (81)
    OPEN POINT DESCRIPTION (90)
      ENTER VERTEX (92)
      ENTER ARC POINT (opt) (93)
      ENTER CURVE POINT (opt) (94)
      .
      .
      ENTER VERTEX (92)
    CLOSE POINT DESCRIPTION (91)
  CLOSE OBJECT (64)
  OPEN POLYGON OBJECT (opt) (63)
    SET LINE STYLE (opt) (66)
    SET PEN NUMBER (opt) (67)
    SET FILL STYLE (opt) (69)
    SET OBJECT DESCRIPTOR (opt) (70)
    SET OBJECT EXTENT (opt) (71)
    SET PICK POINT (opt) (72)
    SET LEVEL (opt) (73)
    OPEN LABEL DESCRIPTION (opt) (80)
      SET LABEL STYLE (opt) (82)
      SET LABEL TRANSFORM (opt) (83)
      SET LABEL PEN (opt) (84)
      ENTER LABEL (85)
    CLOSE LABEL DESCRIPTION (81)
    OPEN POINT DESCRIPTION (90)
      ENTER VERTEX (92)
      ENTER ARC POINT (opt) (93)
      ENTER VERTEX (92)
      ENTER CURVE POINT (opt) (94)
      ENTER VERTEX (92)
      |
      |
    CLOSE POINT DESCRIPTION (91)
  CLOSE OBJECT (64)
CLOSE SUBPICTURE (51)
CLOSE FILE (11)
```

For a working example, let's take a look at the following code to see exactly how the GMX picture file is constructed. This code was generated by the PUT utility of PLOT 50 Picture Composition software package.

```
P50PICT
FAMILYROOM.PICT 10:0,'FAMILYROOM.PICT';
 12:'SUBPIC:NO';
 50:'';
 52:0,100,0,100;
 53:50,50;
 62:1;
 71:20,90,15,30;
 72:20,15;
 90;
 92:20,15;
 92:35,15;
 92:35,30;
 92:90,30;
 91;
 64;
 62:9;
```

```
open file; username
file descriptor
   open subpicture (1st subpicture in a file has no name)
      set extent
      define pick point
      open path object (ID is #1)
         set extent
         set pick point
         open point description
            enter vertex (at coordinates 20,15)
              "     "
              "     "
              "     "
         close point description
      close object
      open path object (ID is #9--ID #'s don't have to be sequential)
```

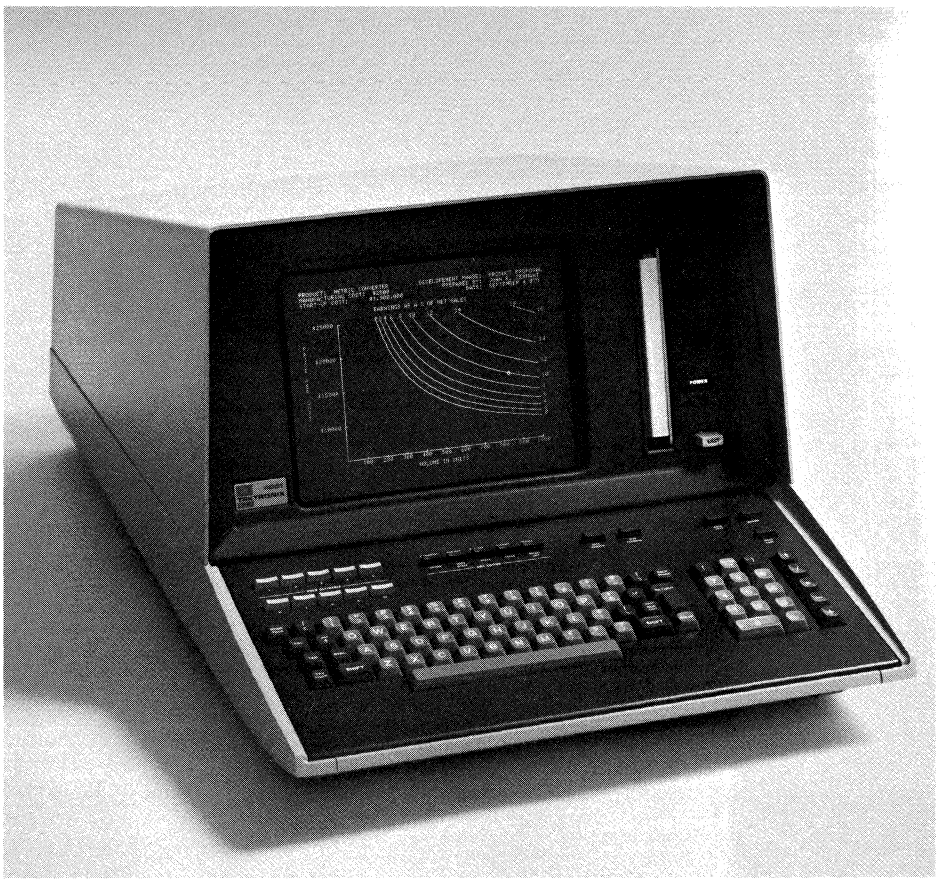
```
71:90,90,40,60;
72:90,40;
90;                                          open point description
92:90,40;
92:90,60;
91;                                          close point description
64;                                        close object
62:11;                                     open path object #11
71:70,90,70,95;
72:90,70;
90;                                          open point description
92:90,70;
92:90,95;
92:70,95;
91;                                          close point description
64;                                        close object
62:13;                                     open path object #13
71:30,70,95,95;
66:2;                                        set line style (for the object)
72:70,95;
90;                                          open point description
92:70,95;
92:30,95;
91;                                          close point description
64;                                        close object
62:14;                                     open path object #14
71:20,30,80,95;
72:30,95;
90;                                          open point description
92:30,95;
92:20,95;
92:20,80;
91;                                          close point description
64;                                        close object
62:34;                                     open path object #34
71:20,20,60,70;
72:20,70;
90;                                          open point description
92:20,70;
92:20,60;
91;                                          close point description
64;                                        close object
62:40;                                     open path object #40
71:20,20,30,50;
72:20,50;
90;                                          open point description
92:20,50;
92:20,30;
91;                                          close point description
64;                                        close object
62:42;                                     open path object #42
71:20,20,70,80;
66:2;                                        set line style
72:20,80;
90;                                          open point description
92:20,80;
92:20,70;
91;                                          close point description
64;                                        close object
62:46;                                     open path object #46
71:90,95,60,70;
72:95,70;
90;                                          open point description
92:95,70;
92:90,60;
91;                                          close point description
64;                                        close object
62:48;                                     open path object #48
71:90,95,30,40;
72:90,30;
90;                                          open point description
92:90,30;
92:95,40;
91;                                          close point description
64;                                        close object
62:50;                                     open path object #50
71:70.63,87.08,72.35,90.57;
72:77,88;
```

```
80;                                          open label description
82:4;                                           set label style (font)
83:0.94,-0.35,0.34,0.94;                        set label transform
85:75.18,80.15,'CHAIR';                         enter label
81;                                          close label description
90;                                          open point description
92:77,88;
92:72.51,75.8;
92:70.63,76.49;
92:75.81,90.57;
92:87.08,85.42;
92:81.89,72.35;
92:80.02,73.04;
92:84.51,85.24;
92:77,88;
92:76.65,87.06;
92:76.9,84.84;
92:82.53,82.77;
92:84.16,84.3;
92:80.36,73.98;
92:79.08,73.38;
92:73.45,75.45;
92:72.86,76.74;
91;                                             close point description
64;                                          close object
62:20;                                       open path object #20
71:39,61,77,87;
72:42,87;
80;                                             open label description
82:4;
83:1,0,0,1;
85:43,79,'COUCH';
81;                                             close label description
90;                                             open point description
92:42,87;
92:42,77;
92:39,77;
92:39,87;
92:61,87;
92:61,77;
92:58,77;
92:58,87;
92:58,86;
92:42,86;
92:42,84;
92:43,83;
92:49,83;
92:50,84;
92:50,86;
92:50,84;
92:51,83;
92:57,83;
92:58,84;
92:58,78;
92:57,77;
92:51,77;
92:50,78;
92:50,84;
92:50,78;
92:49,77;
92:43,77;
92:42,78;
91;                                             close point description
64;                                          close object
62:7;                                        open path object #7
71:40,60,66,74;
72:40,66;
80;                                             open label description
82:4;
83:1,0,0,1;
85:43,69,'COFFEE TABLE';
81;                                             close label description
90;                                             open point description
92:40,66;
92:60,66;
92:60,74;
92:40,74;
92:40,66;
91;                                             close point description
64;                                          close object
62:29;                                       open path object #29
```

```
71:20,20,50,60;                                    set line style
66:2;
72:20,60;                                          open point description
90;
92:20,60;
92:20,50;
91;                                                close point description
64;                                              close object
61:26,33,49,'FAMILY ROOM';                       open text object #26
71:33,77,49,58;                                    set extent
67:2;                                              set pen number
68:2;                                              set text style (font)
65:4,0,0,9;                                        set transform
72:33,49;                                          set pick point
64;                                              close object
62:36;                                           open path object #36
71:23.73,34.27,78.73,89.27;
72:29,84;
80;                                                open label description
82:4;
83:1,0,0,1;
85:26,85,'ROUND';                                  (places a two-word
83:1,0,0,1;                                        label on two lines)
85:26,82,'TABLE';
81;                                                close label description
90;                                                open point description
92:29,89.27;
93:29,78.73;                                       (arc)
92:29,89.27;
91;                                                close point description
64;                                              close object
63:24;                                           open polygon object #24
71:58,85,30,35;
72:58,30;
80;                                                open label description
82:4;
83:1.11,0,0,1;
85:68,32,'BOOK CASE';
81;                                                close label description
90;                                                open point description
92:58,30;
92:58,35;
92:85,35;
92:85,30;
91;                                                close point description
64;                                              close object
62:37;                                           open path object #37
71:44.8,54.8,33.38,41.38;
72:44.8,36.38;
80;                                                open label description
82:4;
83:1.05,0,0,1.28;
85:47.87,38.27,'T. V.';
81;                                                close label description
90;                                                open point description
92:44.8,36.38;
92:54.8,36.38;
92:54.8,41.38;
92:44.8,41.38;
92:44.8,36.38;
92:47.8,36.38;
92:48.8,33.38;
92:50.8,33.38;
92:51.8,36.38;
91;                                                close point description
64;                                              close object
62:57;                                           open path object #62
71:45.8,53.8,41.38,42.56;
72:53.8,41.38;
90;                                                open point description
92:53.8,41.38;
93:49.76,42.56;                                    (arc)
92:45.8,41.38;
91;                                                close point description
64;                                              close object
51;                                            close subpicture
11;                                          close file
```

4051 Redesigned for Added Value

by Dave Watts
Tektronix, Inc.
Wilsonville, OR



There are a number of reasons for redesigning an existing product. Sometimes technology changes, and sometimes supplied components change. Other times there is a desire to add features that weren't available when the product was introduced.

The 4051 has just undergone such a change, which has resulted in new standard features and added value. The basic 4051 now comes with 16K Bytes of standard memory. In addition, the Matrix ROM functions and Binary Program Loader functions are now standard.

The Added Value 4051 will allow first-time users a wider selection of application software due to the larger available memory. And the built-in matrix functions are a feature that's useful in solving linear regressions, statistical problems, and two- and three-dimensional transformations. The standard binary program loader in the backpack will load programs from tape to memory four to six times faster than ASCII stored programs.

In addition to containing the same matrix and binary program loader functions that are found in the 4052 and 4054, another change is found in the size of additional memory. There is a single memory increment of 16K Bytes available for the redesigned 4051. This means there are two memory sizes available — either 16K Bytes or 32K Bytes of user memory.

By adding these features, the 4051 will continue to serve many users as a high-resolution graphic desktop computer. Further information about the Added Value 4051 can be obtained from your local Tektronix Sales Engineer.

TEKniques Interview

This week the TEKniques reporter interviews the staff of the APPROACHING COMPUTER GRAPHICS SEMINAR, Lynn Normandin and Don Davis, Seminar Leaders, and Sam Dunne, Seminar Coordinator and Registrar.

T: I understand that TEK sponsors a computer graphics seminar. Can you tell us how it got started?

D: Yes. It began because management of our Information Display Division feels that managers in general have a difficult time finding out about promising new technology. As a leader in the field of computer graphics terminology, it seemed appropriate for us to address that problem. That led to the development of the Approaching Computer Graphics seminar program, about two years ago.

T: Is the program aimed only at managers?

L: It's presented from a manager's viewpoint. However, it has been found useful for a number of others.

T: What percentage of your attendees are managers?

L: Typically about 40% of the attendees are management people.

T: What about the other 60%?

D: That's really a tough question. Our attendees come from a wide variety of areas. Some of these include finance, sales and marketing, data processing, education, engineering, and architecture. We even have a graphic artist attending at about 80% of the seminars.

T: In a few words, what is the purpose of the seminar?

L: To provide managers and other non-computer professionals with a comprehensive introduction to computer graphics.

T: What do you mean by "non-computer" professionals?

D: Just that you don't have to be a computer expert to come to the seminar and benefit from it.

T: Does that mean that technical people shouldn't come?

D: Not at all. As a matter of fact, many of our attendees would be considered quite technical. However, they come to the seminar for the comprehensive overview we provide rather than for technical information.

T: When and where are the seminars held?

S: We have a regular schedule that takes us to major cities in the United States and Canada. Typically we will be in a different city each month, with two seminars in each location. During the next few months we'll be in Seattle, Atlanta, Washington D.C., and Houston. And of course attendees can and do come from all over to attend.

T: How do people find out about the seminar?

S: Some by word of mouth, some by trade publication articles. But most learn about it through our direct mail brochure, which is mailed to each location several weeks before the scheduled seminars.

T: It sure sounds interesting. From the outline in your brochure it looks like a pretty busy day.

D: Yes it is. We cover a lot of material, beginning with the history of computer graphics and ranging through hardware and software concepts, graphic display systems, CAD/CAM, graphing, and implementation of a computer graphics solution. We make extensive use of color visuals, using about 600 35mm slides. We also have two workshop sessions each day.

T: Are the workshops "hands-on" sessions using computer terminals?

L: No, the workshops are really management-solving sessions. We present case study problems and our attendees are broken into teams to develop possible solutions.

D: The workshops are a big help to many attendees, in that they are given an immediate opportunity to apply some of the things that they've learned.

L: The interactive discussion in the workshops is also a nice change of pace from the slide presentations.

T: Okay, let me review the seminar for our readers. It's a one-day introduction to computer graphics, aimed primarily at managers, but you don't have to be a manager to attend. Also, it's presented in non-technical terms, so you don't need a log of computer experience to benefit.

The seminar tuition is $150, which includes the seminar reference manual, lunch, and post-seminar refreshments. If any of our readers would like additional information, they should look at the partial brochure reprint in this issue, and/or call Sam on the seminar hotline.

S: That's right. The number is (503) 682-0771, and you can call collect. We have a recording device on the line, so you can call any time and leave a message if we're not there.

D: One other thing that I'd like to mention is that we also offer APPROACHING COMPUTER GRAPHICS SEMINAR at the customer's site.

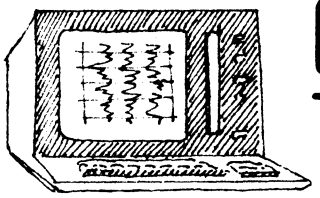
T: How does that work?

D: When there are several people in an organization who could benefit from this computer graphics overview, it's often more practical for us to bring the seminar to them. The cost savings are significant, particularly for those in remote locations.

These on-site seminars have proven to be especially useful to current users who want to introduce or promote computer graphics to other departments within their organization.

T: How would our readers go about getting more information on these on-site seminars?

S: Just call the seminar hotline number — (503) 682-0771. I'll be able to provide pricing and scheduling information, and answer any other questions that might arise.



PROGRAMMING TIPS

Correction to Programming Tip

TEKniques Vol. 4 No. 8 carried the programming tip "Repetitious Data Entry." Correct statement 170 to read:

170 GO TO 210

Our thanks to James Y. Hope, Systems Analyst at Shearwater Inc. in Stonington, CT, for pointing out this error. Our apologies to Michael Anderson for inadvertently changing a line of his code.

Convert PLOT 50: GRAPH PLOT to Disk

by Bill Detweiler
Office of Health Services Research
West Virginia University
Morgantown, W.V.

Persons who use TEKTRONIX Plot 50: Graph Plot (Tektronix product no. 4050A07) may find considerable advantage in storing the package's four programs on a 4907 File Manager disk. This implementation reduces overlay access time, and eliminates the task of constantly changing between the program tape and the data (or presentation) tape in the 4050 system tape drive.

To make this modification, follow these instructions:

1) Insert the Graph Plot tape in the system tape drive. Load file 1.
2) Delete lines 80 through 95, and lines 400 through 450.
3) Enter the following lines of code:

```
80 M=MEMORY
81 CALL "UNIT",0
82 GOSUB F1 OF 84,86,88,90
83 GO TO 96
84 APPEND "GRAPHPLOT/INIT";100
85 RETURN
86 APPEND "GRAPHPLOT/MANA";100
87 RETURN
88 APPEND "GRAPHPLOT/PRES";100
89 RETURN
90 APPEND "GRAPHPLOT/DISP";100
91 RETURN
400 GO TO 68
```

4) Now save the program on disk as "GRAPHPLOT/INIT".
5) Load the second file on the tape into the 4050 memory. If you do not have a binary program loader, load file 5 instead.
6) Convert lines 6830 and 6840 to 'REMARKs', either by entering the line number and then 'REMARK' or by inserting 'REM' in front of the existing statements.
7) Save this program on the disk as "GRAPHPLOT/MANA".
8) Load file 3 (file 6 for ASCII).
9) Convert lines 590 and 600 and 5720, 5730 to 'REMARKs' as explained in instruction 6.
10) Save this program on disk as "GRAPHPLOT/PRES".
11) Load file 4 (file 7 for ASCII).
12) Convert lines 6230 and 6240 to 'REMARKs' as explained in instruction 6.
13) Save this program on disk as "GRAPHPLOT/DISP".

To run the program, just enter: OLD "GRAPHPLOT/INIT" and then press return. You no longer have to worry about putting the program tape in the 4050 drive.

Storing Strings in Arrays

by Deedie Strandridge
Phillips Petroleum Company
Bartlseville, OK

In research, it is often necessary to refer to a sample by a name rather than a number. This routine allows you to store a character string in an array and call it back later.

The program converts a segmented character of a string variable to ASCII and stores that number in a two dimensioned array (number of samples x maximum number of letters). Later, the program reads the array, translating the ASCII representation back to a character. It can be used to print horizontal as well as vertical strings (with few modifications).

```
100 REM *** STORE A CHARACTER STRING IN AN ARRAY
110 INIT
120 DIM T(10,20)
130 PAGE
140 REM *** I = NUMBER OF SAMPLES
150 FOR I=1 TO 10
160 PRINT "ENTER A 20 CHARACTER DESCRIPTION OF THE SAMPLE."
170 INPUT A$
180 IF I=1 AND A$="" THEN 410
190 IF A$="" THEN 300
200 L=LEN(A$)
210 FOR J=1 TO 20
220 REM *** J = MAXIMUM NUMBER OF CHARACTERS
230 IF J<=L THEN 260
240 B$=" "
250 GO TO 270
260 B$=SEG(A$,J,1)
270 T(I,J)=ASC(B$)
280 NEXT J
290 NEXT I
300 K=I-1
310 PAGE
320 REM *** READ CHARACTER STRINGS FROM ARRAY AND PRINT
330 FOR I=1 TO K
340 A$=""
350 FOR J=1 TO 20
360 B$=CHR(T(I,J))
370 A$=A$&B$
380 NEXT J
390 PRINT A$
400 NEXT I
410 END
```

Use DELETE to Initialize Symbol Table

by Bill Finkenstadt
Tektronix, Inc.
Dayton, OH
and Dan Taylor
Tektronix, Inc.
Wilsonville, OR

One of the outstanding features of the 4050 System is its dynamic memory management. This means the memory space to store arrays or dimensioned strings is not allocated until the respective DIM command is executed.[1] And a DELETE n statement frees all space allocated to n array or n string. Such dynamic allocation allows you to use all available memory for your current data.

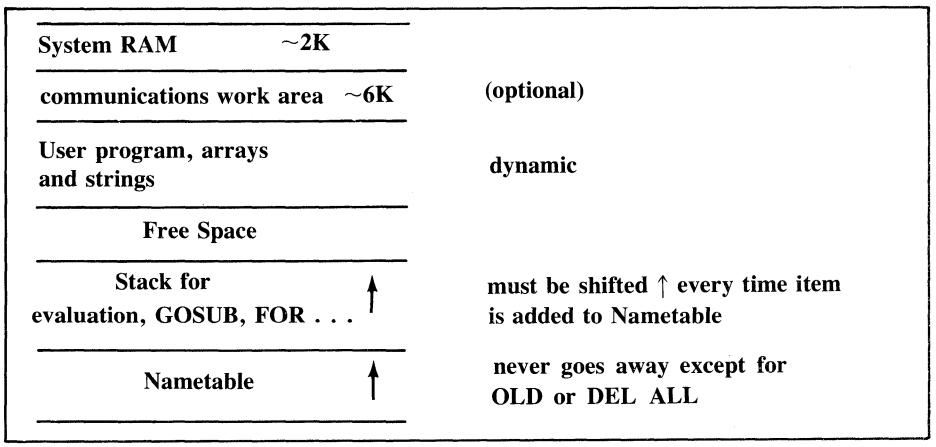
When a program statement including a variable, i.e., DIM A$(100) or PRI B, etc., is translated[2] by the 4050 System, the variable name is placed in the symbol table (if it isn't already there). When the program is executed, the symbol table has already been initialized, thus, no more table space will be required for variable names.[3]

However, suppose your program consists of several files which you'll be appending during the course of execution, and those files contain code using new variable names. If you are approaching a full memory with your program and data, you may want to include all variables from all files in your symbol table at the beginning. Thus, you will know what memory is available for data without worrying about symbol table space taken later for incoming local variables (see the diagram for memory allocation).[4]

How do you assign variable names to your symbol table without the statements which use them? Simply, include code which DELETEs them. The variable names in the DELETE statement will be assigned to the symbol table at translation time, and the code need never be executed.

```
1000 GOTO 2000
1010 DELETE T1, T2 . . .
1020 DELETE W1, W2
.
.
2000(use MEM to determine available space)
```

This helps avoid MEMORY full problems when those other files are APPENDed.



4050 Memory structure (conceptually)

[1]Execution occurs when a program is RUN.
[2]Translation occurs when a statement is keyed in (immediate mode), OLDed or APPENDed.
[3]See your 4050 Graphic System Reference manual for how much space is required for variable names and Programming Tips, p. 49.
[4]Be sure to leave space for the Status Stack (see Programming Tips pp. 21, 37, 45 and 83, and TEKniques Vol. 4 No. 6, p. 22).

Command Entry through POINTER Function

by Chester M. Nowicki, EE
Naval Air Development Center
Warminster, PA

The 4050 Series POINTER function lets you enter commands without pressing RETURN. This feature proves especially useful for implementing functions with a single keystroke.

The sample main program occupies statements 100 to 220. The subroutine beginning at statement 500 fetches up to five consecutive keystrokes, but retains only the first four. However, single key - strokes may cause direct action. For example, control-C causes a restart of the entry sequence (line 560). Also, control-L automatically assigns a "PAGE" command word to A$ and forces the subroutine to end (lines 600 to 620).

The routine fetches keystrokes with C$, builds commands with A$, counts entries with N and tracks the cursor print position with X and Y. Variables A and B serve only to fulfill the requirements of the POINTER instruction.

Statement 140 tests for an "END" command that terminates the program. Statement 150 accepts only commands that have four characters; unacceptable command lengths force the re-entry of a command. Statements 170 and 180 test for and act upon the "PAGE" sample command.

Alternate approaches to command decoding could employ indexed lists of command words (strings or data statements), whereby a computed GOSUB would effect the required action. The program could handle invalid commands by printing an appropriate message, then branching to line 130 for command re-entry. Of course, the instruction in statement 170 should branch to the statement for the command test routine (see remarks in statements 190 and 200), unless it's incorporated within that routine.

In case your program requires repeated entry and execution of commands, inserting GOTO 130 between statements 210 and 220 would provide a continuous loop. As written, the main program terminates after entry of any four character sequence, as well as the "END" and "PAGE" commands.

Of particular significance to the use of the POINTER command is that pressing RETURN assigns C$ the value of "" with zero length. If the program requires a carriage return, the programmer should provide this test:

```
. . . IF C$<>"" THEN  (line number of next instruction)
. . . C$=CHR$(13)
. . . (program continues)
```

Statement 630 tests for the zero length condition of C$ for completion of command entry as a consequence of pressing RETURN. This allows entry of commands shorter than four characters, e.g., "END"). Alternatively, character count N=5 automatically ends the subroutine and passes the previous four entries to the calling routine. Statement 620 effects the equivalent of a RETURN for the "PAGE" command, in response to the control-L keystroke.

```
100 DIM C$(1)
110 PAGE
120 REM *** GO FETCH COMMAND STRING
130 GOSUB 500
140 IF A$="END" THEN 220
150 IF LEN(A$)<>4 THEN 130
160 REM *** INTERPRET COMMAND STRING FOR ACTION
170 IF A$<>"PAGE" THEN 210
180 PAGE
190 REM *** IF COMMAND INVALID, PRINT MESSAGE; THEN
200 REM *** BRANCH TO 130 TO FETCH ANOTHER
210 PRINT
220 REM
230 END
480 REM *** THIS SUBROUTINE TAKES UP TO FIVE
490 REM *** KEY ENTRIES, BUT IGNORES THE LAST
500 PRINT USING "/,10T,""INPUT 4 CHARACTERS AND ANY 5TH ONE: "",S":
510 N=1
520 A$=""
530 GIN X,Y
540 POINTER A,B,C$
550 MOVE X,Y
560 IF C$="C" THEN 500
570 REM *** TEST ENTRIES TO EXCLUDE OR ACT UPON CONTROL CHARACTERS
580 REM *** BRANCH TO 540 TO IGNORE ENTRY, OR ASSIGN A PREDEFINED
590 REM *** COMMAND WORD TO A$, THEN SET C$=""
600 IF C$<>"L" THEN 630
610 A$="PAGE"
620 C$=""
630 IF N=5 OR LEN(C$)=0 THEN 680
640 N=N+1
650 A$=A$&C$
660 PRINT C$;
670 GO TO 530
680 RETURN
```

Transferring 468 Oscilloscope Waveforms from the 4924 to the 4050

by John Burgess and
Pat Adamosky
Tektronix, Inc.

The GPIB-compatible 4924 makes an excellent external storage medium for the 468 digital oscilloscope. But 468 data is output in binary format, so the tapes cannot be read directly into the 4050 Series Graphic Computing System from its internal tape drive. The solution is to read the tape from the 4924 into the 4050 through the GPIB. The following program will accomplish that task.

The file header on the 4924 tapes is changed from "ASCII LOG," which is generated by the 4924, to "BINARY DATA." The 4050 then does a WBYTE @64+D,110:, which sets up the 4924 to send binary data to the 4050. The 4050 will then display the transferred waveform.

NOTE: In the program, the primary address of the 4924 is designated as 4.

```
100 REM PROGRAM TO RETREIVE AND GRAPH A WAVEFORM FROM THE 468 AND 4924.
110 REM THE 468 HAS WRITTEN A WAVEFORM TO A TAPE IN THE 4924 IN TALK-
120 REM ONLY AND LISTEN-ONLY RESPECTIVELY.  A 4924 IS ALSO NECESSARY
130 REM HERE TO READ THE TAPE'S DATA.
140 REM
150 REM BY: JOHN BURGESS;    DATE: 8-19-80
160 REM LAST MOD: 8-25-80   BY: JHB
170 REM *--*--*--*  SECTION TO READ BINARY DATA FROM 4924  *--*--*--*
180 INIT
190 REM *--*--*--*--*  CHANGE HEADER TO "BINARY DATA"  *--*--*--*--*
200 REM THIS IS SO IT CAN BE READ BACK IN BINARY (TO FAKE OUT THE
210 REM "ASCII LOG" THAT THE 4924 WRITES AS ITS HEADER.
220 PRINT "What file number do you want: ";
230 INPUT F
240 PAGE
250 PRINT @4,@:@,@,1,1
260 FIND @4:F
270 INPUT @4:A$
280 A$=REP("BINARY  DATA",9,12)
290 FIND @4:F
300 C$=CHR(13)
310 A$=A$&C$
320 A$=A$&"@"
330 PRINT @4:A$
340 PRINT @4,@:@,@,@,@
350 REM *--*--*--*--*--*  NOW READ THE PREAMBLE  *--*--*--*--*--*
360 REM END OF PREAMBLE IS DENOTED BY A !%! (ASCII 37).
370 FIND @4:F
380 REM TELL 4924 TO SEND BINARY DATA FROM THE TAPE (MTA=4, SCG=14)
390 WBYTE @68,110:
400 RBYTE A
410 IF A>90 THEN 730
420 IF A=37 THEN 470
430 A$=CHR(A)
440 PRINT A$;
450 GO TO 400
460 REM NOW READ AND DECODE THE NUMBER OF POINTS TO BE SENT
470 RBYTE A
480 P=256*A
490 RBYTE A
500 P=P+A
510 PRINT
520 PRINT "␣␣THERE ARE ";P;" POINTS TO FOLLOW.  PRESS <CR> TO PLOT. "
530 PRINT "PRESS <CR> AFTER PLOT FOR NEXT WAVEFORM FROM TAPE FILE."
540 INPUT A$
550 REM NOW SET UP WINDOW AND PREPARE TO PLOT
560 WINDOW 1,P,0,255
570 REM MOVE TO FIRST POINT
580 RBYTE A
590 MOVE 1,A
600 FOR I=1 TO P-2
610 RBYTE A
620 DRAW I,A
630 NEXT I
640 REM READ AND DISCARD CHECKSUM BYTE
650 INPUT A$
660 PAGE
670 RBYTE A
680 REM: CHECK FOR END-OF-FILE
690 T=TYP(@)
700 REM: IF NEXT ITEM TYPE <> 1 THEN NOT EOF
710 IF T<>1 THEN 400
720 REM: IF END OF FILE, NO MORE WFMS, END PROGRAM
730 REM CLOSE TAPE FILE & RESET GPIB
740 PRINT "END-OF-FILE ENCOUNTERED. NO MORE WAVEFORMS ON TAPE FILE."
750 PRINT @4,2:
760 WBYTE @63,95:
770 HOME
780 END
```

Efficient Array Search

by Ramon L. Hershman

Navy Personnel Research and Development Center
San Diego, CA

Problem: A common problem in business or scientific computation is locating a target value T in a numeric array A. Assume that, at most, one of the N members of A is equal to T. We seek an efficient scheme either to determine that T does not exist or to extract its index in A. That is, the search should yield, say, S=0 if no A(I)=T or S=I where A(I) is the unique array member having the value T.

Two possible search methods are:

(A) Conventional Self-terminating Loop: Sequentially inspect each array member, terminating either when an A(I)=T is found or the loop ends without success. See lines 110-170 below.

(B) Scalar/Array Relational Comparison: This scheme exploits the powerful array operations of the 4050 Series. Lines 210-250 first execute a one-time-only load of the integers from 1 to N in the array W. Line 310 sets the array X either to 0 or to N-1 zeros with X(I)=1 where A(I)=T. Line 320 then sets X(I)=I if the target exists, and line 330 produces the index in S.

Methods (A) and (B) were inserted in FOR/NEXT routines on the 4051 and timed with a hand-held digital stopwatch. For the self-terminating loop, execution time was 16 msec times the number of array elements to be searched. If S=0 all N elements must be searched, and the results in column 2 of the table are obtained.

Execution times for Method (B) grow with slope ≈.0035 as N increases and are virtually independent of T's presence/absence or its location in A. Results appear in column 3 of the table. Thus for S=0 the scalar/array comparison scheme is approximately 4 times as fast as the conventional search for moderate to large N and is more than twice as fast even for N=5.

In the absence of other information, if T exists we expect S to average N/2. Method (B) is then approximately twice as fast for moderate to large N. It also has the virtue that, for fixed N, its response time has nearly zero variance which is desirable for interactive applications.

Users will note that the price of improved execution time is the added memory requirements of the arrays W and X. However, if frequent searches of large arrays are required, the tradeoff should prove attractive.

```
100 REM Method A
110 S=0
120 FOR I=1 TO N
130 IF A(I)<>T THEN 160
140 S=I
150 I=N
160 NEXT I
170 END

200 REM One-Time Load
210 DIM W(N),X(N)
220 FOR I=1 TO N
230 W(I)=I
240 NEXT I
250 END

300 REM Method B
310 X=A=T
320 X=X*W
330 S=SUM(X)
340 END
```

4051 Execution Times in Seconds

N	Method A S=0	Method B All S
5	0.08	0.03
10	0.16	0.05
25	0.40	0.10
50	0.80	0.19
100	1.60	0.37
200	3.20	0.71
400	6.40	1.37

Variable Format Field Modifier

by W.C. Wilkinson
Space Antenna Techniques
Princeton, NJ

In many cases an image statement is needed which will format the printing of a row of values in which the number of columns is not known beforehand. That is, the number of columns is a variable. The format statement,

```
IMAGE N(5D.2D)
```

does not allow the field modifier, n, to be a variable.

The following routine provides this capability in which M0 is the number of columns.

```
100 U$=STR(M0)
200 V$=U$&"(5D.2D)"
300 PRINT USING V$:C9
```

Now if M0 = 5, statement 300 is equivalent to:

```
300 PRINT USING "5(5D.2D)":C9
```

The attached program demonstrates the use.

```
100 REM PROGRAM TO DEMONSTRATE USE OF A VARIABLE
110 REM AS AN "n" FIELD MODIFIER
120 PRINT "ENTER NUMBER OF COLUMNS ";
130 INPUT M0
140 DIM C9(M0)
150 FOR I=1 TO M0
160 C9(I)=SQR(M0+I)
170 NEXT I
180 U$=STR(M0)
190 V$=U$&"(5D.2D)"
200 REM NOTE+ V$ IS NOW M0(5D.2D) AND M0 IS THE "n" MODIFIER
210 PRINT USING V$:C9
220 DELETE C9
230 END

RUN

ENTER NUMBER OF COLUMNS 5
    2.45    2.65    2.83    3.00    3.16
RUN

ENTER NUMBER OF COLUMNS 8
    3.00    3.16    3.32    3.46    3.61    3.74    3.87    4.00
RUN

ENTER NUMBER OF COLUMNS 15
    4.00    4.12    4.24    4.36    4.47    4.58    4.69    4.80    4.90
    5.00    5.10    5.20    5.29    5.39    5.48
```

Renumbering a Program

by Dick Browne
Tektronix, Inc.
Philadelphia, PA

TEKNIQUES Vol. 1 No. 7 described a system for renumbering a program so that subroutines began on even thousands. The following system would be simpler still. It assumes, like the earlier programming tip, that subroutines start on lines 230, 380, and 610, and the main program starts on line 100. Just use the following program: (In the example I use UDK #1, but any key will do.

```
1 GO TO 100
4 RENUMBER 1000,10,230
5 RENUMBER 2000,10,380
6 RENUMBER 3000,10,610
7 END
```

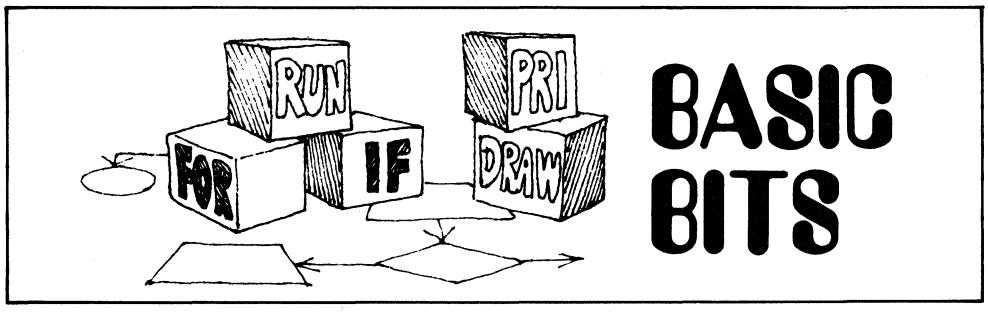
Pressing the UDK then renumbers everything in one step. With this system, a program "under construction" can be quickly and repeatedly renumbered without changing starting points of subroutines.

Going one step farther, instead of numbering 4,5,6,7, you could number the above program 4,8,12,13. Then, by pressing the appropriate key, you would only renumber the portion of the program you are working on, plus restoring following subroutines to their proper starting point.

Also, suppose you are working on the subroutine starting at step 2000. After the last REN statement, you might want to add

```
7 PAGE
8 LIST 2000,2999
9 END
```

to list the part of the program you are working on. Or, these steps could be assigned to another UDK.



Displaying Refreshed Questions

by Charles Otis Sweezey
Department of Theater & Dance
Southern Illinois University
Edwardsville, IL

There are times when you need questions to appear on the CRT while graphics are being displayed. The questions might relate to the graphics, for instance, and as they are answered more graphics are superimposed. A problem arises when you try to write a program to allow the questions to appear on the screen without having the screen "cleared" and the graphics lost when another question is needed.

This program makes use of the "Print @32,24:" command which allows a number, letter, or symbol to appear on the screen without storing on the display. When the program is enacted, the computer prints "Question: #" on the screen for a count of 25. At the end the "1" disappears. A row of "*" covers up the input. When another question is needed, "A" is given the value of "2" and a subroutine starting at line 500 causes the "Question: #" to blink with the "2" disappearing at the end.

Using a card to list which numbers relate to which questions the operator can run his program and answer needed questions. Question numbers disappear when the question is answered, and that answer is covered by "*'s" so that he isn't confused when his next question is asked. The program is shown below.

```
100 A=1
110 GOSUB 500
200 END
500 FOR D=1 TO 25
510 HOME
520 PRINT "QUESTION : #"A
530 MOVE 24,98
540 PRINT @32,24:""A;""
550 NEXT D
560 0,96
570 PRINT "********"
580 MOVE 0,96
590 INPUT A$
600 RETURN
```

Plot Rotation on the 4662 Plotter

by Jerry W. Anderson
Phillips Petroleum Co.
Bartlesville, Okla

Occasionally you may wish to rotate a 4662 plot from an existing plot program. Of special interest would be a 90° rotation so the long side of the graph paper may be used as a "vertical" direction. However, you can't make a rotation (other than 180°) by simply using the SET buttons on the plotter. But you may by inserting the following code in your program:

```
101 SET DEGREES
102 PRINT @D,25:90
105 REM D is device #
```

In addition, change all occurrences of the following statements:

FROM	TO
WINDOW X1,X2,Y1,Y2	WINDOW Y1,Y2,X1,X2
VIEWPORT X1,X2,Y1,Y2	VIEWPORT Y2,Y1,X1,X2

Also, all graphics commands must have their arguments switched. For example, MOVE @D:X,Y becomes MOVE @D:Y,X and RMOVE @D:0,1 becomes RMOVE @D:1,0. These changes may be done easily using the Editor ROM.

Programming Tip Exchange

51/00-0720/0	Machinery Cost Analysis
51/00-0902/0	Calendar Routines (7 Day)
51/00-1203/0	Planimeter
51/00-1605/0	Shear and Moment Diagrams for Determinate Beams
51/00-5506/0	3-D Function Plot
51/00-6102/0	Hewlett Packard I/F Package
51/00-6109/0	Inventory File System
51/00-8028/0	Change & List Program Variables
51/00-8032/0	Device Address Adding Program
51/00-8039/0	Tape File Header Expander
51/00-9516/0	Advanced Media Graphics
51/00-9527/0	3-D Transformation
51/00-9534/0	Q-Plot
plus	
51/00-7004/0	Programming Tips

4050 Series Applications Library Program Abstracts

Order

Documentation and program listings of each program are available for a nominal charge. Programs will be put on tape or disk for a small recording fee per program plus the charge for the tape cartridge or flexible disk. One tape/disk will hold several programs. Programs will be recorded on like media only, i.e., programs on tape cannot be sent on disk and vice versa unless so noted in the abstract.

(The program material contained herein is supplied without warranty or representation of any kind. Tektronix, Inc. assumes no responsibility and shall have no liability, consequential or otherwise, or any kind arising from the use of this program material or any part thereof.)

Domestic U.S. Prices:

Documentation and Listing only	*$25 per program*
Documentation, Listing and Recording Fee	*30 per program*
Tape Cartridge	*36 per tape*
Flexible Disk	*15 per disk*

Contribute

Contribute one program to the Library and receive three in exchange. Send in the membership card from your 4050 Series Graphic System Reference Manual to get the details. Or call us (503) 685 3618.

Forms

Please use the Applications Library Order Form. Order forms are included in the Membership Packet and are available from your local Tektronix Sales Engineer.

Outside U.S.

Program contributions or orders outside the U.S. must be processed through the local Tektronix sales office or sent to one of the Libraries serving your area. See Library Addresses section.

Applications Library Program Ordering, Distribution to be Simplified

The 4050-Series Applications Library now contains well over 200 user-supplied programs. Several hundered copies of these programs are requested by Application Library members each month.

We've noted a very distinct pattern in these orders — when a member orders one program in an application area, he usually orders several in that area.

Using this fact as a guide, we are grouping like-subject programs to fill single tapes or disks. We plan to bind together the appropriate documentation for each program on the tape or disk and offer the package at a savings over the price of the individual programs. You'll be able to order several programs by ordering one part number and save some money.

We're in the process of making the conversion now. We plan to distribute a new Library catalog in mid-summer that will detail the packages and the ordering procedure. Until then, you may order individual programs as before.

Correction to
ABSTRACT #51/00-1602/0

Title: Interactive Beam Analysis

Delete line 1295 in File 1

```
1295 IF ABS(K2(N))<1.0E-20 THEN 1480
```

Delete line 4270 in File 2

```
4270 IF ABS(K2(K))<1.0E-20 THEN 4275
```

Change File 1

```
Old   1298 PRINT "Enter Weight Moment of Inertia, WR↑2 (";K$;") ="
New   1297 PRINT "Enter Weight Moment of Inertia about a diameter of disk"
New   1298 PRINT "or element, W*Rad.Gyr.↑2(";K$;") = ";
Old   6390 T1(3,2)=H*I2(N)*W↑2+K2(N)
New   6390 T1(3,2)=H*I2(N)*W↑2/G+K2(N)
```

Change File 2

```
Old   4272 PRINT "Weight Moment of Inertia (WR↑2) about a"
Old   4273 PRINT "diameter of disk or element ........=";
New   4272 PRINT "Weight Moment of Inertia about a diameter"
New   4273 PRINT "of disk or element (W*Rad.Gyr↑2).........= ";
Old   4984 PRINT "Present Weight Moment of Inertia (WR↑2).....=";
New   4984 PRINT "Present Weight Moment of Inertia (W*Rad.Gyr.↑2)..= ";
Old   4986 PRINT "Enter new Weight Moment of Inertia (WR↑2)=";
New   4986 PRINT "Enter new Weight Moment of Inertia (W*Rad.Gyr.↑2)= ";
```

ABSTRACT #: 51/00-0604/0

Title: Loan Schedule for Budgets and Taxes

Author: Joe Flicek
W.R. Grace & Co.
New York, NY

Memory Requirement: 8K
Statements: 85
Files: 1 ASCII Program

This program computes a monthly loan schedule with totals for desired budget and/or tax periods. The loan schedule displays for each month of the period the following items:

- interest charge
- principal payment
- total loan payment
- ending loan balance

The display also includes the beginning and ending balances, and the total interest charges, total principal payments, and total loan payments for the period covered by the loan schedule.

User prompted input:

- beginning loan balance for budget/tax period
- interest rate of loan as a decimal
- monthly payment on loan
- number of months covered by period
- year of budget/tax period (final two digits)

For loans that extend over more than one budget or tax period, the program is rerun for each period. The ending loan balance of the first budget or tax period becomes the beginning loan balance of the second period and so forth.

```
LOAN SCHEDULE FOR BUDGETS AND TAXES
USER PROMPTED INPUT FOLLOWS BELOW:
INPUT BEGINNING LOAN BALANCE                    20000
INPUT INTEREST RATE AS DECIMAL                  .148
INPUT MONTHLY LOAN PAYMENT                      300
INPUT NUMBER OF MONTHS FOR PERIOD               12
INPUT YEAR OF LOAN SCHEDULE (TWO DIGITS)        81
```

```
LOAN SCHEDULE FOR 1981
----------------------

Beginning Loan Balance    $     20000.00
                          --------------

MONTH    Interest    Principal    LOAN        Ending
-YEAR    Charge      Payment      Payment     Balance
-----    --------    ---------    -------     -------

 1-81      246.67       53.33     300.00     19946.67
 2-81      246.01       53.99     300.00     19892.68
 3-81      245.34       54.66     300.00     19838.02
 4-81      244.67       55.33     300.00     19782.69
 5-81      243.99       56.01     300.00     19726.67
 6-81      243.30       56.70     300.00     19669.97
 7-81      242.60       57.40     300.00     19612.57
 8-81      241.89       58.11     300.00     19554.45
 9-81      241.17       58.83     300.00     19495.63
10-81      240.45       59.55     300.00     19436.07
11-81      239.71       60.29     300.00     19375.78
12-81      238.97       61.03     300.00     19314.75
           --------    ---------   -------
Total     2914.75      685.25     3600.00

Ending Loan Balance   $     19314.75
                      --------------
```

ABSTRACT #: 51/00-0605/0

Title: Mortgage Amortization

Author: Ronald C. Robinder
Tektronix, Inc.
Beaverton, OR

Memory Requirement: 8K
Files: 1 ASCII Program
Statements: 62

The program will calculate an amortization table for loans such as home mortgages.

You are prompted for the term of the loan, the principal amount, and the interest rate.

The program will return with the monthly payment. A pause allows you to interact for correction, copying, or terminating the execution.

After the pause, the program will display the repayment schedule in two year segments. A pause between each page allows you to copy, restart, quit or continue.

```
HOW LONG (IN YEARS) IS THE LOAN? 12
HOW MUCH IS BEING BORROWED? 20000
HOW HIGH IS THE INTEREST (IN %)? 14.8
YOUR MONTHLY PAYMENT WILL BE $ 297.61
```

```
MONTH   PRINCIPAL     INTEREST       BALANCE
  1     $   50.94     $  246.67     $ 19,949.06
  2     $   51.57     $  246.04     $ 19,897.49
  3     $   52.21     $  245.40     $ 19,845.28
  4     $   52.85     $  244.76     $ 19,792.43
  5     $   53.50     $  244.11     $ 19,738.93
  6     $   54.16     $  243.45     $ 19,684.77
  7     $   54.83     $  242.78     $ 19,629.94
  8     $   55.51     $  242.10     $ 19,574.43
  9     $   56.19     $  241.42     $ 19,518.24
 10     $   56.89     $  240.72     $ 19,461.35
 11     $   57.59     $  240.02     $ 19,403.76
 12     $   58.30     $  239.31     $ 19,345.46
 13     $   59.02     $  238.59     $ 19,286.44
 14     $   59.74     $  237.87     $ 19,226.70
 15     $   60.48     $  237.13     $ 19,166.22
 16     $   61.23     $  236.38     $ 19,104.99
 17     $   61.98     $  235.63     $ 19,043.01
 18     $   62.75     $  234.86     $ 18,980.26
 19     $   63.52     $  234.09     $ 18,916.74
 20     $   64.30     $  233.31     $ 18,852.44
 21     $   65.10     $  232.51     $ 18,787.34
 22     $   65.90     $  231.71     $ 18,721.44
 23     $   66.71     $  230.90     $ 18,654.73
 24     $   67.53     $  230.08     $ 18,587.20
```

ABSTRACT#: 51/07-0717/0

Title: 4907 MIPS — A Management Information Processing System

Author: Jim Dillion
Tektronix, Inc.
Santa Clara, CA
Modified by: L.C. Sheppard
Sheppard Software Company
Sunnyvale, CA

Memory Requirement: 32K
Peripherals: 4907 File Manager
Statements: 928
Files: 2 Program
Requires data disk

MIPS contains two programs: One for a 12 period fiscal year and one for a 13 period fiscal year. The following modifications have been made to the 12 period program:

1. Excess code removed to save memory.
2. Individual disk file setup changed to a single, large random access file where one record is equivalent to the old MIPS "file."
3. The fixed number of MIPS files was changed to allow from 1 through 300 with a default of 100.
4. The 4907 clock may be set.
5. Any disk unit may be used.
6. Tabular data may be sent to the screen, plotter or printer.
7. Graphic data may be sent to the screen, or plotter.
8. Prompts for data show a specific format if required.
9. The multicurve graph was altered to always put the fiscal year line definition in the upper left corner of the graph, instead of the lower right or upper left.

10. This version consistently reflects that it handles 12 accounting periods instead of 13.

11. Prompts for subfiles are more informative.

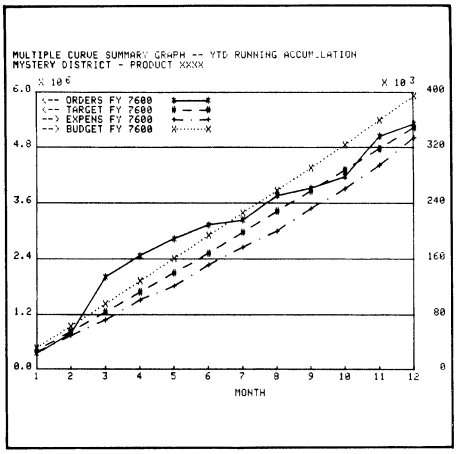
12. Changes in new disk creation.

13. MIPS will automatically store itself on each new disk.

14. The program may be sent to user on tape (no disk only restriction).

MYSTERY DISTRICT - PRODUCT XXX SALES ORGANIZATION
COMPARISON REPORT -- ORDERS VS TARGET

MONTH	F/Y7600 ORDERS	F/Y7600 TARGET	MONTH DIFF	MONTH %	7600YTD ORDERS	7600YTD TARGET	YTD DIFF	YTD %
1	351234	420000	-68766	83.63	351234	420000	-68766	83.63
2	451025	420000	31025	107.39	802259	840000	-37741	95.51
3	1205412	420000	785412	287.00	2007671	1260000	747671	159.34
4	456235	420000	36235	108.63	2463906	1680000	783906	146.66
5	365235	420000	-54765	86.96	2829141	2100000	729141	134.72
6	298565	420000	-121435	71.09	3127706	2520000	607706	124.12
7	95865	450000	-354135	21.30	3223571	2970000	253571	108.54
8	546213	450000	96213	121.38	3769784	3420000	349784	110.23
9	156235	450000	-293765	34.72	3926019	3870000	56019	101.45
10	254654	450000	-195346	56.59	4180673	4320000	-139327	96.77
11	852345	450000	402345	189.41	5033018	4770000	263018	105.51
12	265845	450000	-184155	59.08	5298863	5220000	78863	101.51



ABSTRACT #: 51/00-1408/0

Title: Failure Rate Analysis
Author: W. E. Price
Computer Sciences Corp.
Atlanta, GA
Memory Requirement: 32K
Peripherals: Optional — 4662 Plotter
4642 Printer
Files: 1 ASCII Program
Statements: 892

Failure Rate Analysis determines either a subsystem's or a subassembly's reliability and total mean time to failure (MTTF) during a specified operating time. You may choose from three types of analysis:

Parts Count Prediction
Part Stress Analysis with Redundant Circuits Only
Part Stress Analysis with Redundant Subassemblies and Circuits

Parts Count Prediction. In the early stages of development, parts count prediction can be used to produce "ball park" values. Data preparation is simple and the operations within the program are not complex.

Part Stress Analysis with Redundant Circuits Only. Most of the subassemblies are not redundant, therefore, this allows you to calculate the failure rates in a mode where you can only be concerned with redundant circuits.

Part Stress Analysis with Redundant Subassemblies and Circuits. When redundant subassemblies do exist, this section automatically calculates the subassembly failure rate as it does in Part Stress Analysis (above) and quantifies it before calculating the total subsystem data.

The program prompts you for general input:

```
                    FAILURE RATE ANALYSIS

Do you want the data printed on the
(0) SCREEN or the (1) PRINTER? 0

Enter the number of file in which you last stored data: 5

CHOOSE the TYPE of ANALYSIS:

0. PART STRESS ANALYSIS with Redundant Circuits only

1. PARTS COUNT PREDICTION

2. PART STRESS ANALYSIS with Redundant subassembly & circuits

 CHOOSE the procedure: 0

                    ***********

                    PART STRESS ANALYSIS

How many subassemblies will be calculated? 6


                  *** SUBASSEMBLY CALCULATION ***

Enter the number of locations at
which Subassembly #1 will exist: 13

Are theses initial calculations? y

           Subsystem: POWER

Assembly/Subassembly: 7.5 or 20 KVA UPS

INPUT Operating Hours: 476

                    *****

How many Circuits/LRU's are there? 25

Do you need to enter old component values? N

        Circuit#1: Rectifier Leg

How many components are there? 1

The Failure Rate for this circuit will be calculated using :

     1. INSPECTION FORMULAS
     2. REDUNDANCY FACTORS
     3. NO REDUNDANCY
     4. REPAIR RATE FORMULAS

CHOOSE the procedure: 3

Enter this circuit's quantity: 1

Is this a (0) MICROELECTRONIC or a

 (1) REGULAR ELECTRONIC DEVICE? 1

                    Component #1

                    NON MICROELECTRONIC DEVICE

Enter Q(Quantity): 1

Enter B0(Base Failure Rate): 4.042

Enter the number of Applicable Parameters for this device: 0

Enter (1) if you want a plot for this
circuit's data or enter a (0) if not! 1
```

Number of subassemblies
Number of locations per subassembly
Subsytem name
Assembly/subassembly name
Operating hours
Number of circuits/LRU's
Circuit name
Number of components per circuit
Procedure to use:
  Inspection formulae
  Redundancy factors
  No redundancy
  Repair rate formula
Circuit's quantity

The component calculation depends upon the type of analysis that has been selected. Parts Count Prediction determines the failure rate using this formula:

$F = \lambda_G \cdot \pi_Q \bullet Q$ (FAILURE RATE)

Where,

λ_G = Generic Failure Rate

π_Q = Quality Factor

Q = Quantity

Part Stress Analysis determines the failure rate for each part from one of these equations:

1. Non Microelectronic Parts

   $F = \lambda_B X (\pi\text{'s}) \times Q$ (FAILURE RATE)

   Where,

   λ_B = Base Failure Rate

   π's = Parameter Values (i.e. Quality and Stress Factors)

   Q = Quantity

2. Microelectronic Parts

   $F = \lambda_p \times Q$ (FAILURE RATE)

   Where,

   Q = Quantity

$\lambda_p = \pi_L \cdot \pi_Q (C_1 \pi_R + {}_2 \pi_E) \pi_P$

and

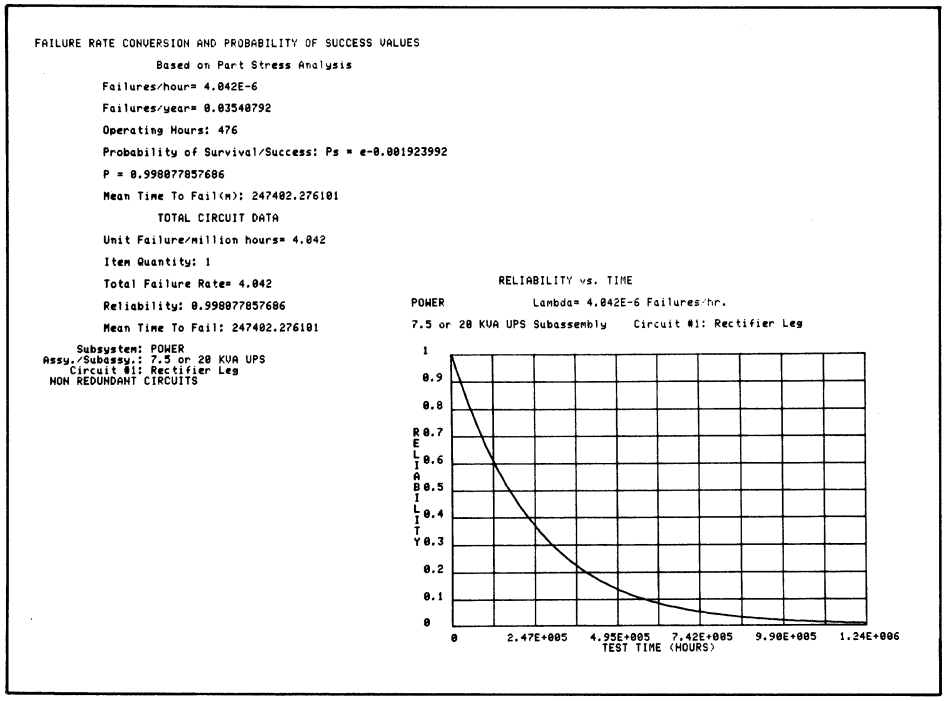
π_L: Learning Factor
π_Q: Quality Factor
π_E: Environmental Factor
π_p: Pin Factor (Nonlinear devices only)
C_1: Failure Rate #1
C_2: Failure Rate #2

If data is available for a partially complete circuit analysis, the program will prompt you for the total failures/million hours for the completed components so they won't have to be calculated again.

The circuit calculation is a collection process of all the component values. After each component's failure rate is calculated, it is added to a continuous sum of the failure rates. When the total failure rate is found for the circuit, it is used to find that circuit's reliability and MTTF.

A Reliability vs. Time plot may be had for every circuit analysis, and for a Part Stress Analysis after the total Subsystem data has been determined. A single term Poisson Distribution of a reliability may be asked for at the end of the analysis after the total subassembly or subsystem data has been calculated.

The Parts Count Prediction circuit data and subassembly data may be stored on tape. Part Stress Analysis subassembly data and the total subsystem data may also be stored on tape. You may also have stored data printed.



ABSTRACT #51/00-1607/0

Title: Beam Analysis II
Authors: Y.W. Luk
Larry D. Mitchell
Virginia Polytechnic Inst. & S.U.
Mechanical Engineering Dept.
Blacksburg, VA
Memory Requirement: 32K
Peripherals: 4051R05 Binary Program Loader
4051R01 Matrix ROM
Files: 6 Binary Programs
2 Binary Data (examples)
Statements: 1,661

One of the most frequently encountered engineering designs is beam because it can be used for modeling many structures. This program, using Transfer Matrix Method, provides the following analyses for an undamped beam system:

1. Free vibration (eignevalue — eigenvector)
2. Static Response
3. Forced dynamic response at a particular frequency
4. Frequency response at a point along the beam
5. Animation of beam motion in the dynamic case

This program computes and plots the curves of deflection, slope, moment and shear along the beam for all four analyses listed above. Beams of uniform or variable cross-section can be entered. Uniformly or linearly-varied distributed loads, concentrated point loads, applied moments or combinations of all three may be applied to static, forced dynamic or frequency response analysis. It also allows any combination of pinned, fixed, free or guided flexural boundary conditions, even normally kinematically unsta-

ble conditions can be handled if sufficient internal supports are provided. In-span support can be elastic springs and/or elastic moment spring. Static indeterminary to any order is handled. Modeling for dynamic responses uses lumped mass. In the dynamic case, the beam motion can be animated in time and space.

All the input data can be stored on the tape and recalled for future use. It also provides the option of using either English or SI units.

Program limitations

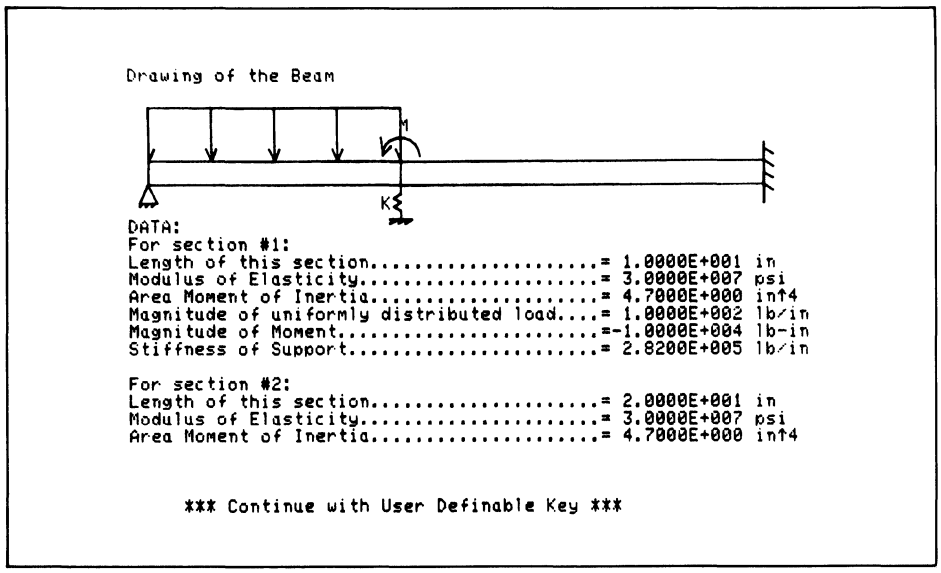
This program has the following limitations:

1. It handles only undamped beam system.
2. It will not handle rigid in-span indeterminate supports but they can be approximated by assigning a large value for the stiffness or rotary stiffness of the supports, e.g., 1 x 10^{10} lb/in or N/m, or 1 x 10^{10} lb-in/rad or N-m/rad. 1 x 10^{20} seems to be the limit for a beam having 5 sections before numerical difficulty is encountered.
3. The system must be loaded within the elastic range.

Conclusion

This interactive computer-aided analysis routine is almost unlimited in the complexity of multi-supported and load beams. It is much easier to use than other "canned" computer programs, especially those that are written for a batch environment; i.e., computer cards are used as input and line printer as output. It has error feedback, structure verification and graphical output. All these enable the user to enter the data correctly and get the correct result in a short time.

This program is much more powerful than Interactive Beam Analysis, application library program #51/00-1602/0; however, that program does a couple of things that Beam Analysis II doesn't.



ABSTRACT #: 51/00-1608/0

Title: Fatigue Analysis
Authors: Y.W. Luk
Graduate Student
L.D. Mitchell
Professor
Virginia Polytechnic Inst.
& S.U.
Mechanical Engineering Dept.
Blacksburg, VA

Memory Requirement: 22K
Files: 1 ASCII Program
Statements: 568

Machine members are often found to have failed under the action of repeated or fluctuating stresses, and yet an analysis reveals that the actual maximum stresses were below the yield strength. These failures usually are from stresses repeated for large number of times. This failure is called a fatigue failure.

There are many theories that predict the fatigue failure. The six most generally accepted are: modified Goodman fracture line, modified Goodman yield line, Soderberg line, Gerber line, Quadratic line, and Kececioglu line. These six failure lines are available in the program to size failure. Any equivalent stress theories are allowed. It can also compute the significant endurance limit with the theoretical stress concentration factor and other physical and environmental parameters supplied by the user.

Input Data Required. Units can be either in English or SI system. The following data are needed:

1. Ultimate tensile strength of the material, psi or Pa.
2. Yield strength of the material, psi or Pa.
3. Significant endurance limit, psi or Pa.
4. Moment causing alternating stress, lb-in or N-m.
5. Moment causing a steady stress, lb-in or N-m.
6. Alternating axial force, lb or N.
7. Steady axial force, lb or N.
8. Alternating torque, lb-in or N-m.
9. Steady torque, lb-in or N-m.
10. Safety factor, dimensionless.
11. Lower limit of the dimension, in or m.
12. Upper limit of the dimension, in or m.

If the significant endurance limit is not known, the following physical and environmental parameters are required for computing the significant endurance limit.

13. Type of surface finish.
14. Reliability, %.
15. Operating temperature, °F or °C.
16. Theoretical stress concentration factor.
17. Notch sensitivity factor (optional).
18. Notch radius (optional), in or m.
19. Type of material.
20. Type of loading.
21. Miscellaneous effect factor (optional).
22. Shape of cross section.
23. Endurance limit for R.R. Moore rotating beam specimen (optional), psi or Pa.
24. Number of cycles.

The following example concerns a rotating shaft of circular, machined steel. The input data details its parameters.

```
Do you want to use English Units? (Y or N)Y

We will use English Units throughout this routine.

Please enter the strength of the material to be used.

Tensile Strength, in psi, Su                = 200000
Yield Strength,   in psi, Sy                = 150000

Do you know the Significant Endurance Limit of the material? (Y or N)N
This section of the program will calculate the Significant
Endurance Limit.

Enter the # for the types of surface finish used.
#1 for polished finish.
#2 for ground finish.
#3 for machined or cold drawn.
#4 for hot rolled.
#5 for as forged.
3
What is the reliability in %?99

What is the operating temperature in Degree F ?190

Do you know the Theoretical Stress Concentration Factor (Kt)?Y
Theoretical Stress Concentration Factor = 2.5

Do you know Notch Sensitivity (q)? (Y or N)N
What is the notch radius in inches?0.02

Is the material steel? (Y or N)Y
Is it under bending or axial loading? (Y or N)Y

Is there any miscellaneous-effect factor? (Y or N)N
The S-N curve is used for this determination.
A Log-Log or a Log-Linear S-N curve will be used.
The finite life region is 0.9*Su @1E3 cycles to Se'' @1E6 cycles.
What method do you want to use for computing the
Significant Endurance Limit (Se''')?
Enter #1 for Log-Log method.
Enter #2 for Log-Linear method.
1
Is the cross section circular? (Y or N)Y

Do you know the Endurance Limit (Se') for rotating-beam
specimen? (Y or N)N
Is the design life infinite? (Y or N)N
Number of cycles = 500000
You will now be requested to supply the PARAMETRIC DESIGN STRESS
EQUATIONS for your design problem.
You must write these equations in BASIC. Use the following instructions.
If you are unfamiliar with the development of such equations,
see user's guide and its appendix for guidance.
Enter component loads starting with line 6000 incrementing by 10's.
 Use M1=Moment causing alternating stress (lb-in).
 Use M2=Moment causing a steady stress (lb-in).
 Use F1=An alternating axial force (lbf).
 Use F2=A steady axial force (lbf).
 Use T1=An alternating torque (lb-in).
 Use T2=A steady torque (lb-in).
 Use  N=Safety factor.
Enter the numeric value for each of the variables used in
your stress equations.
Do this before you enter your stress equations.
Enter your stress equations starting with line 6100
incrementing by 10's.
 Use A1=Alternating stress.
 Use A2=Mean stress.
 Use PI=Pi.
 Use D=Basic dimension. N.B., All dimensions should be given in terms of
 D. In case of rectangular component, use proportions.
After your stress equations are entered, type a numbered return
statement. Then type run 600.
EXAMPLE.....................................................
The following equations are in English Unit.
6000 T2=1200.
6010 M1=2400.
6020 N=1.8
6100 A1=N*M1/(PI*D↑3/32)
6110 A2=0.866*N*T2/(PI*D↑3/32)
6120 RETURN
RUN 600

6000 T2=1200
6010 M1=2400
6020 N=1.8
6100 A1=N*M1/(PI*D↑3/32)
6110 A2=0.866*N*T2/(PI*D↑3/32)
6120 RETURN
RUN 600

Select the fatigue failure line to be used in the design.

If Modified Goodman Fracture Line, enter MGF
If Modified Goodman Yield Line, enter MGY
If Soderberg Line, enter S
If Gerber Line, enter G
If Quadratic Line, enter Q
If Kececioglu Line, enter K
MGF
The Failure Line selected is Modified Goodman Fracture Line.
The failure equation is (Sa/(R2*Se''')↑m + (R1*Sm/Su)↑p = 1.
where M=1
where P=1
where R1=1
and where R2=1
Do you wish to change any parameters? (Y or N)N

The following entries will establish the limits on a Half
Interval Search for the solution to your problem.
What is the smallest basic dimension that you wish to try?
Give your answer in inches. Do not answer 0.0.  0.01
What is the largest dimension, in inches? 10
```

The Failure Line on which this chart is based is the Modified Goodman Fracture Line.

```
**************************************************************

Tensile Strength, Su.........................= 200,000 psi
Yield Strength, Sy...........................= 150,000 psi
Significant Endurance Limit, Se'''@500000 cyc.= 21,395 psi
Smallest dimension tried.....................= 0.01 inches
Largest dimension tried......................= 10.00 inches
Safety Factor, N.............................= 1.80
Moment causing alternating stress, M1........= 2,400.00 lb-in
Steady torque, T2............................= 1,200.00 lb-in

The following factors are used for computing Se''':
Surface factor (Ka)              = 0.64
Size and shape factor (Kb)       = 0.81
Reliability factor (Kc)          = 0.81
Temperature factor (Kd)          = 0.95
Fatigue Strength Reduction factor (Ke)= 0.42
Miscellaneous factor (Kf)        = 1.00
Endurance Limit for Rotating-beam Specimen (Se')= 100,000 psi
Significant Endurance Limit for infinite life (Se''')=16,960 psi

The Failure Line selected is Modified Goodman Fracture Line.
The failure equation is (Sa/(R2*Se''')↑m + (R1*Sm/Su)↑p= 1.
where M=1
where P=1
where R1=1
and where R2=1

The design dimension is 1.2911 inches

**************************************************************

Do you wish to convert the design dimensionto SI unit?(Y or N)Y

The design dimension is 0.0328 m
```

Using the same input data, the component is redesigned by the other five fatigue failure lines. A similar chart is displayed for each Failure Line. All six results are listed in the following table for comparison.

Type of Failure Line used	Design Diameter	
	in	m
Modified Goodman fracture line*	1.2911	0.0328
Modified Goodman yield line*	0.7491	0.0190
Soderberg line	1.2983	0.0330
Gerber line	1.2696	0.0322
Quadratic line	1.2691	0.0322
Kececioglu line (with b= 1.5)[1]	1.2693	0.0322

*If the user subscribes to the Modified Goodman theory, the largest dimension of the two Goodman solutions must be selected.

[1]This is an arbitrary choice of exponent used for this example only.

Normally, all these analyses are not carried out. Only the analysis that corresponds to the user's selected theory is computed. But the program provides the capability to redesign the component using other fatigue failure theories in order that the user can choose among alternatives.

From the results listed in the table, the smallest dimension is obtained by modified Goodman yield line. This result is not correct for a fatigue design because the modified Goodman theory demands that one considers the yield line and the fracture line. Thus, the largest dimension of the two Goodman solutions must be selected. The most conservative solution is obtained using Soderberg line. This is not always true. It depends upon the location of the load line. In this example, the solutions from modified Goodman line, Soderberg line, Gerber line, Quadratic line, and Kececioglu line are quite close. This is true only for this particular example because of the load line. Other examples may give more significant differences between these theories.

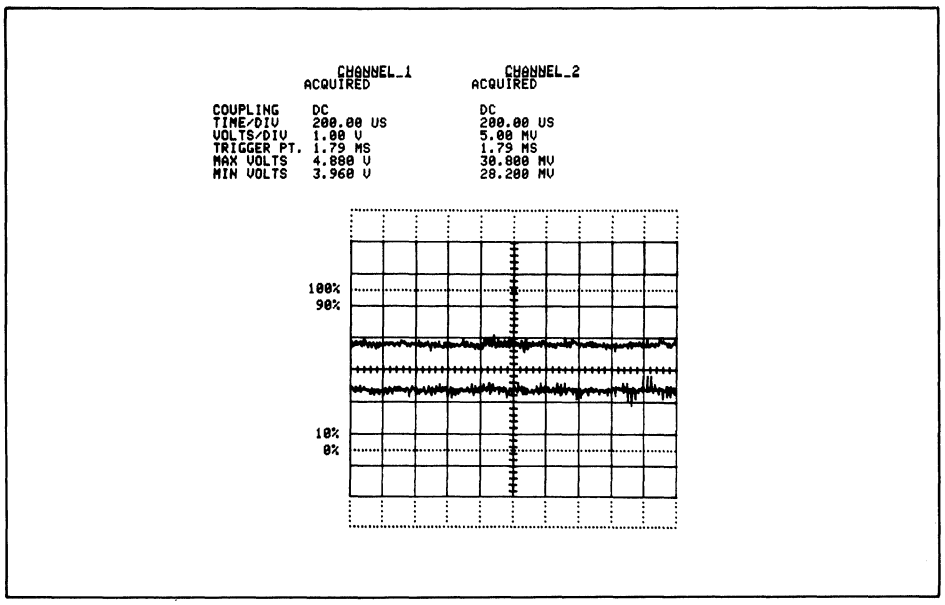
ABSTRACT #: 51/00-6125/0

Title: 4050/468

Author: Craig Bulmer
Tektronix, Inc.
Chicago, IL

Memory Requirement: 32K
Peripherals: Tektronix 468 Oscilloscope
Optional: 4662/3 Plotter
Files: 1 Binary Program
Statements: 463

This program will take waveforms from the 468 Oscilloscope and display the waveforms on the 4050 screen; with printed header information of Channel 1, 2 and/or Add; Volts/Div; Time/Div; Trigger Point; Max Volts; and Min Volts. Waveforms can be saved to tape and redisplayed from tape. Output to either screen or plotter with reference scope grid. Waveforms displayed from tape are displayed as dots.



ABSTRACT #: 51/00-8015/1

Title: Flow Diagrammer

Author: Keith S. Reid-Green
Education Testing Service
Princeton, NJ
Modified by: L.C. Sheppard
Sheppard Software Company
Sunnyvale, CA

Memory Requirement: Tape — 16K; Disk — 32K
Peripherals: 4662 Plotter
Optional — 4907 File Manager
Statements: Tape — 917
Disk — 911
Files: Tape — 5 ASCII Program
Requires 62 dedicated files
Disk — 1 Program

Flow Diagrammer was modified to run on disk with room for 200 diagrams per disk. The tape version will remain the same (see the abstract in the 4050 Applications Library Program Catalog). The modification for disk are detailed below:

A diskette for data storage is required.

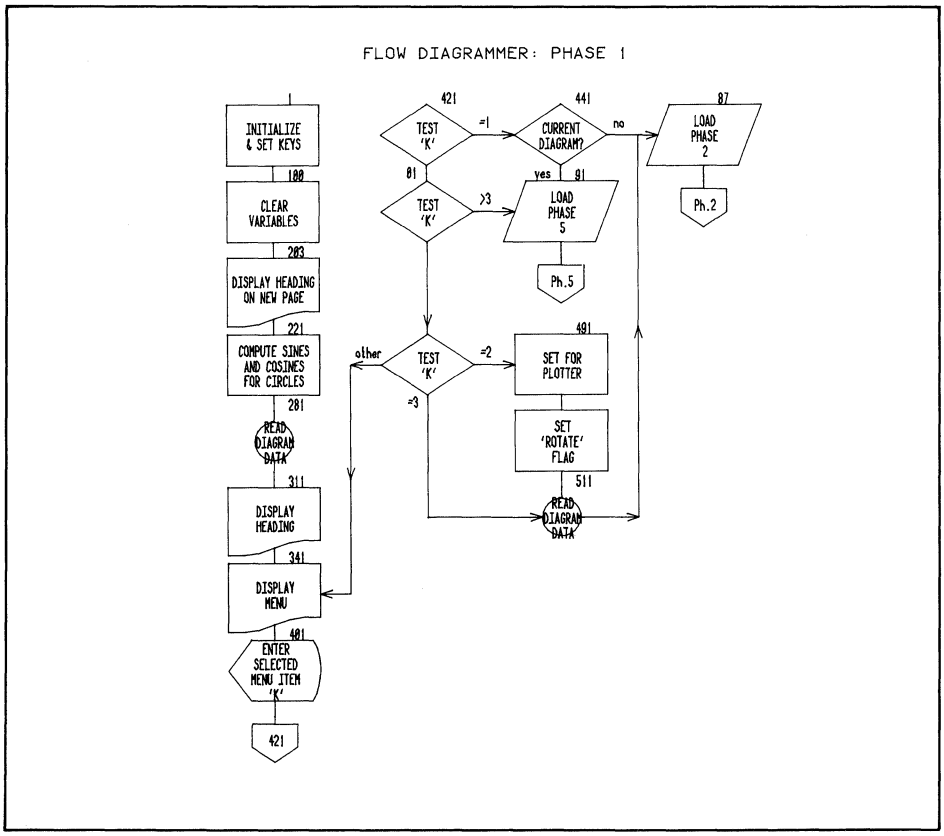
The five programs were combined into one for operational speed.

Any disk unit can be used for the data disk. 3 disk files replace 57 tape files.

The maximum amount of "box data" was doubled to 2000 characters.

The program will let you set the 4907 clock. It will also format the disk, if required, and create and format the data files on the disk.

Several changes were made in the program flow and data storage for increased speed and smoothness.



ABSTRACT #: 51/00-8018/1

Title: $FORMAT
Author: Bruce Clarkson
Science Applications, Inc.
Raleigh, NC
Revised by: John R. Carter, Sr.
Tektronix, Inc.
Santa Clara, CA
Memory Requirement: 16K
Peripherals: 4051R06 Text Editor ROM
or
A.L. Program #52/07-8047/0
Printer
Optional—4907 File
Manager
4924 Tape Drive
Files: 1 ASCII
Statements: 390

$FORMAT provides formatted text output similar to the University of Waterloo's SCRIPT program for the System/360. Commands are imbedded in the source text file at the time of creation — typically with the TEKTRONIX Editor ROM or with Applications Library program $EDIT.DOS (abstract #52/07-8047/0).

$FORMAT prompts for additional parameters relative to the page layout at execution time. The commands in the source file combined with these parameters give control over the format of the printed document.

Options include:
right margin justification (padding with blanks)
page size
page numbering
indentation
spacing
centering
and other refinements

The modifications to the original (Text Formatting) allow input from a 4907 File Manager in ASCII or Binary, and text file creation using $EDIT.DOS. Output may be to any peripheral or device, including the 4050 screen.

ABSTRACT #: 52/07-8047/0

Title: $EDIT.DOS
Author: John Carter
Tektronix, Inc.
Santa Clara, CA
Memory Requirement: 64K
Peripherals: Optional — 4907 File
Manager
— 4924 Tape Drive
— 4641 Printer
Files: 8 Binary Program
2 Binary Data
Requires dedicated tape or disk.
Statements: 1415
8,000 bytes binary data

$EDIT.DOS is a 4907 disk oriented screen editor for the 4052/54. However it may be used with cartridge tape. It creates free style, line oriented text enabling you to produce reports, manuscripts, form letters and so on. Its features are:

1. Two edit buffers for manipulating lines of text.
2. Complete INPUT and OUTPUT file handling for extra long text.
3. Fast and easy movement of the line pointer and character pointer back and forth through the text.
4. FIND and AGAIN which locates any string of characters in a line.
5. Five ways to do a REPLACE function.
6. Panic abort from long processes.
7. Single letter command entry.
8. Multiple commands per line of input plus REPEAT.
9. STATUS report on current editor environment.
10. LIST to any peripheral or device with or without line reference numbers.
11. Exit from program without disturbing open files or contents of buffers. Recover to $EDIT.DOS prompt.
12. Write over or append to the output file.

The $EDIT.DOS commands are:

A-gain	I-nsert	R-eplace
B-ottom	K-ill	S-ave
C-lear	L-ist	T-ag
D-own	M-acro	T-op
F-ind	O-ld	U-p
G-et	P-ut	V-erify
H-elp	Q-uit	X-change

Two special commands, "<" and ">", move the character pointer left or right through the current line. Another feature uses the vertical bar, "|", for repeating EDIT comands.

In addition to the keyboard commands there are 10 defined function keys for STATUS, RECOVERY, VERIFY ON/OFF, MERGE LINES, SPLIT LINES, PANIC STOP, RULER, CHANGE EDIT DELIMITER, REPEAT LAST EDIT COMMAND, and OLD "$FORMAT".

A complete HELP message file is included and should be sufficient for using the program at 95% efficiency on the first try.

Program 51/00-8018/0 has been modified to run with $EDIT.DOS to format and justify the edited text. It is available through the Applications Library.

```
*T↖L15
ELEF12;ELIN78;EJUSY;ETOP9;
↑
>  1:ELEF12;ELIN78;EJUSY;ETOP9;
   2:$EDIT.DOS
   3:EL12;EI0;6 APRIL, 1981;EA48;64K (4052/4054 only)
   4:E114;Ea60;(All optional)
   5:Eb;EI0;John R. Carter, Sr.Ea48;4907, 4924, 4641, Option 1.
   6:EA0;E120;Ei;This program creates free style, line oriented text.
   7:There are twenty-four EDIT commands that enable the user to genera
     te reports,
   8:manuscripts, form letters, or whatever.
   9:Es;Ei;A companion formatter program, $FORMAT.DOS, uses commands im
     bedded in
  10:the text for programmable page layout and text formatting.
  11:Es;Ei;The key features of this program are the double buffers for
     easy
  12:text manipulation, the direct access help messages for friendly us
     e, and the flexible
  13:use of many of the EDIT commands, to name just a few.
  14:Es;Ei;This program requires a 4052 or 4054 with 64K of memory.
  15:A 4907 disc drive is optional.
```

This text was created by $EDIT.DOS. Later $FORMAT.DOS. was used to produce the output. The asterisk is the EDIT prompt. T moves the pointer to the top. L15 says to list 15 lines. The "up arrow" shows where the current character point is and the > shows the location of the line pointer.

ABSTRACT #: 51/07-8048/0

Title: Disk-to-Tape Back/Restore Utilities

Author: John H. Grant
Tektronix, Inc.
Seattle, WA

Memory Requirement: 32K
Peripherals: 4907 File Manager
Files: 8
Statements:

These utilities allow a user of a 4050/4907 system to archive his disk files onto tape. Specific capability of the utilities include:

All file types supported by the 4907 File Manager may be archived and restored.

Multiple volume backups are supported, i.e., in the event that the information on the disk exceeds the capacity of a single tape, more than one tape will be used.

Directory editing for total backups is supported: files may be omitted from the backup process and file names which require passwords may (must) be modified to include the password.

Selective backups and restores are also supported.

Automatic hard copies of the archived and restored file names may be generated.

The disk-to-disk backup/recovery programs were designed specifically for owners of a single-drive 4907 who need to backup information from their disks.

However, it may also be used in multiple-drive configurations.

The disk-to-tape backup/restore utility programs are distributed to customers as eight tape files. The first of these is a program to transfer the remaining programs onto disk.

```
   Enter the unit number of the disk drive -- 1

   Do you want to backup or restore a disk (B/R)? B

   Selective or total (S/T)? T

   Insert the disk you would like to backup in drive 1
     and press RETURN:

   Insert your disk backup tape (WARNING: any data on
     your tape will be destroyed).  When you are ready
     to continue, press RETURN:

   Do you need to edit the directory (Y/N)? Y

   The following functions are available:
                  1   Directory listing
                  2   Directory listing interrupt
                  3   Delete directory item
                  4   Modify directory item
                  5   Edit complete

   Please make your selection via the user definable keys.
$DIRECTORY
                  LIB1/PROG2
                  LIB1/LIB2/PROG3
                  LIB1/LIB2/LIB3/PROG4
                  LIB1/LIB2/LIB3/LIB4/PROG5
                  SYSLIB/BACKUP
                  SYSLIB/BACKUP.B01
                  SYSLIB/BACKUP.B02
                  SYSLIB/BACKUP.B03
                  SYSLIB/BACKUP.R01
                  SYSLIB/BACKUP.R02
                  SYSLIB/BACKUP.R03
                  SCRATCHLIB/ASCRAN
                  SCRATCHLIB/PROG1
                  SCRATCHLIB/RESTRICTED
                  SCRATCHLIB/PROG.TEST
                  SCRATCHLIB/BINSEQ1
                  SCRATCHLIB/BINSEQ
                  SCRATCHLIB/BINRAN1
                  SCRATCHLIB/BINRAN
                  SCRATCHLIB/ASCSEQ1
                  SCRATCHLIB/ASCSEQ
                  SCRATCHLIB/ASCRAN1

   Please make your selection via the user definable keys.
$DELETE

   (To terminate DELETE mode, press RETURN when prompted for item.)

   Which item? PROG
   You want to delete LIB1/PROG2 (Y/N)? N
   You want to delete LIB1/LIB2/PROG3 (Y/N)? N
   You want to delete LIB1/LIB2/LIB3/PROG4 (Y/N)? N
   You want to delete LIB1/LIB2/LIB3/LIB4/PROG5 (Y/N)? N
   You want to delete SCRATCHLIB/PROG1 (Y/N)? N
   You want to delete SCRATCHLIB/PROG.TEST (Y/N)? N
     ***PROG not found***
   Which item? PROG1
   You want to delete SCRATCHLIB/PROG1 (Y/N)? Y
     ***DELETED***
   Which item?

   Please make your selection via the user definable keys.
$MODIFY

   (To terminate MODIFY mode, press RETURN when prompted for item.)

   Which item? RESTRICTED
   You want to modify SCRATCHLIB/RESTRICTED (Y/N)? Y
     Enter the correct name: SCRATCHLIB/RESTRICTED:PASSWORD
     ***MODIFIED***
   Which item?

Archived files --
                  @LIB1/PROG2
                  @LIB1/LIB2/PROG3
                  @LIB1/LIB2/LIB3/PROG4
                  @LIB1/LIB2/LIB3/LIB4/PROG5
                  @SYSLIB/BACKUP
                  @SYSLIB/BACKUP.B01
                  @SYSLIB/BACKUP.B02
                  @SYSLIB/BACKUP.B03
                  @SYSLIB/BACKUP.R01
                  @SYSLIB/BACKUP.R02
                  @SYSLIB/BACKUP.R03
                  @SCRATCHLIB/ASCRAN
                  @SCRATCHLIB/RESTRICTED:PASSWORD
                  @SCRATCHLIB/PROG.TEST
                  @SCRATCHLIB/BINSEQ1

 The end of the tape has been reached.  Please insert
   another tape and press RETURN to continue:
                  @SCRATCHLIB/BINSEQ
                  @SCRATCHLIB/BINRAN1

 The end of the tape has been reached.  Please insert
   another tape and press RETURN to continue:
                  @SCRATCHLIB/BINRAN
                  @SCRATCHLIB/ASCSEQ1
                  @SCRATCHLIB/ASCSEQ

 The end of the tape has been reached.  Please insert
   another tape and press RETURN to continue:
                  @SCRATCHLIB/ASCRAN1

   Disk backup completed.
```

4050 Series Applications Libraries

Africa, Europe, Middle East

Contact local sales office

Australia

4050 Series Applications Library
Tektronix Australia Pty. Limited
Sydney
80 Waterloo Road
North Ryde, N.S.W. 2113

Canada

4050 Series Applications Library
Tektronix Canada Ltd.
P.O. Box 6500
Barrie, Ontario
Canada L4M 4V3

Caribbean, Latin America and Far East (excl. Japan)

IDD Group
Export Marketing
Tektronix, Inc.
P.O. Box 500
Beaverton, Oregon 97077
U.S.A.

Japan

4050 Series Applications Library
Sony/Tektronix Corporation
9-31 Kitashinagawa-5
Tokyo 141 Japan

United States

4050 Series Applications Library
Tektronix, Inc.
Group 451
P.O. Box 500
Beaverton, Oregon 97077

Address Correction Requested — Forwarding and Return Postage Guaranteed.

TEKTRONIX, INC.
Information Display Division
Applications Library
Group 451
P.O. Box 500
Beaverton, Oregon 97077

Tektronix®
COMMITTED TO EXCELLENCE

BULK RATE
U.S. POSTAGE
PAID
TEKTRONIX, INC.