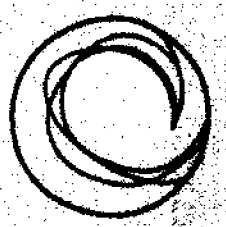
# MICROCOMPUTERS IN THE BOATYARD:

## Microcomputers and Design Aids

Robert M. Scher

August, 1984





MICHU-SG-84-604

This publication is a result of work sponsored by the Michigan Sea Grant College Program, Project number E/CCD-4, with a grant, NA-80AA-D-00072 from the Office of Sea Grant, National Oceanic and Atmospheric Administration (NOAA), U.S. Department of Commerce, and funds from the State of Michigan.

# Microcomputers and Design Aids

Robert M. Scher

Department of Naval Architecture and Marine Engineering
The University of Michigan

August, 1984



MICHU-SG-84-604
$5.00

Michigan Sea Grant College Program - Publications Office
2200 Bonisteel Boulevard - Ann Arbor, Michigan 48109

TABLE OF CONTENTS

## LIST OF FIGURES

# INTRODUCTION

The applicability of the computer to everyday design tasks has come about through two parallel developments. First, of course, the nature and general accessibility of the computer itself has changed almost beyond recognition since the age of the dinosaurs, less than forty years ago. To maintain some perspective on this time span it may be useful to remember that the first "higher-level" programming language, FORTRAN I, was introduced by IBM in 1956. Prior to that, anyone who used a computer had to be more or less initiated into the mathematics and mystery of how the machine worked. Today, by contrast, the microcomputer has become a fairly common household appliance, and grade-school children are routinely taught both computer use and programming.

Second, and equally important from the designer's point of view, the application of the computer to the design process has depended on the development of computer graphics. To a great extent, this development has been manifested in the form of peripherals, mainly input and output devices suitable for handling information in graphic form, and the software to use them.

Not much more than twenty years ago computers were still regarded, even by many knowledgeable designers and engineers, as somewhat esoteric things, an attitude that is difficult to recall at this point. Any substantial computing capability required an extravagantly large and expensive installation, which placed the use of computers primarily in the hands of major scientific research, the defense establishment, large manufacturing and industrial firms, and banks. For the most part, the computer was viewed as a tool of high-speed mathematical analysis or large-scale data processing. The high price of

computing limited the applications to a small set of large problems.

Apart from the restrictions on computer use due to unavailability and expense, the typical designer faced another limitation of the machine: its relative blindness to pictorial information. While the computer was already well established as a tool for high-speed calculation and data storage and retrieval, the ability of the machine to deal with drawings was not yet well developed. This was especially true with regard to input functions: since the only common input device was the keyboard (either through a teletype or a keypunch), data entry for graphic work involved typing in large sets of numbers even for relatively simple drawings. This was laborious, to say the least. It was also, quite often, an exercise in producing errors, and therefore not worth the effort. In addition, the storage requirements for graphic data were large, placing an added expense on system owners and users of commercial time-sharing services. Peripherals for graphic output were still expensive, although the technology was fairly well-known.

In 1963, a major development was introduced at MIT, namely, the first practical interactive graphics system. This system, SKETCHPAD, included a cathode-ray tube on which graphic information could be displayed, just as on an oscilloscope. Moreover, graphic information could be entered on the screen by drawing with a light-pen. This was the decisive ingredient: the designer and the computer were now in a position to communicate using the designer's own principal language, the drawing. For a time, graphic systems based on the SKETCHPAD concept remained expensive and exotic, due to their high demands on computer storage and processing-time resources. This was particularly the case since the computers of the era (both mainframes and minicomputers) were quite slow by today's standards.

During the remainder of the 1960's the major factor in bringing the computer to a larger number of designers was the development of more capable machines with price tags in the $100,000 range. While this expansion of the minicomputer market obviously could not bring a computer into every design office, it did place more computing power at the disposal of moderately sized companies, putting them in a position to consider the possible advantages of computer-aided design, among other applications. There was a secondary benefit as well for mainframe and time-sharing users: the minicomputer could be applied to controlling graphic input and output functions, thus saving the more expensive storage and processing time of the main computer for high-speed calculation and data handling.

A number of early software packages directly related to computer-aided design were developed during the 1960's. However, many of these were motivated by the introduction of numerically controlled flame cutting equipment, rather than by the potential use of similar programs to control and aid the drawing process. Thus, in a sense, computer-aided manufacture was the parent of computer-aided design, rather than the reverse.

With a broader market for computer applications, including design, rapid advances took place in the technology of graphic display and input devices. A number of distinct types of visual display units were developed, along with improved plotters and electrostatic hard copy units: the digitizer was introduced for graphic input. Thus, the electromechanical devices that were most vital for computer-aided design were already in a suitable form when the "microcomputer explosion" hit, beginning sometime in the latter part of the 1970's. Conceptually, at least, the software arrangements for microcomputer-based CAD also existed by the mid-70's.

The microcomputer, of course, has been Time Magazine's machine of the year, a distinction never conferred on mainframe computers or minis. We are now living in the midst of computers. The cost of serious professional computing power is within the reach of even modest businesses, consultants, and home users. A number of microcomputer systems incorporate substantial graphics capability, and are compatible with a variety of graphic input and output devices. System costs for a microcomputer-based CAD capability, including the processor, disk drives or other bulk storage devices, all required input and output peripherals, and software, may range from $5,000 to $50,000, the cost depending primarily on software requirements and the number and type of peripherals. Storage capacity, which was once the main determinant of system expense, is at last relatively cheap, and getting cheaper.

Again, history repeats itself. The result of increased availability and decreased cost has been to expand the applications market to a broader spectrum of designers. In turn, the larger market has stimulated the development of more capable systems, in both the hardware and software fields. With the advent of the next generation, the 32-bit microprocessor, we will undoubtedly see more computing power at the disposal of more users.

So, what are the uses of increased computing power? First, increased storage capacity and speed permit multi-user systems to be based on a micro. Eventually, accounting, engineering, and production may be able to use the system simultaneously. Second, larger and more detailed databases are accessible more rapidly. This has a marked effect on design work, and particularly on graphics. Third, more challenging analytical design methods are becoming feasible on the microcomputer. These methods may ultimately

include algorithms that are now confined to mainframe installations by reason of large storage or processing-time requirements. Analyses that were previously performed either on commercial time-sharing systems or by outside consulting firms will eventually be done in-house. Fourth, larger program memory capacity will permit software writers to create more "user-friendly" design systems and application programs, with better error protection and recovery, greater generality and convenience, and more transparency. This will be reflected in shorter learning times for a given application or system, fewer mistakes, and happier designer-users.

There is another kind of "computing power," however, that is not measured in terms of memory size, cycle time, or the capability and convenience of offline storage and input/output devices. This is the power of the users themselves. At the moment, the microcomputer market is still in revolution. With technological advances chasing one another there is always a real problem of rapid obsolescence, which is confusing enough by itself. In addition, however, with a large segment of new and relatively inexperienced users, frustration has been perhaps more prevalent than it should have been. As the market matures and stabilizes, which it seems to be showing signs of doing, users will finally be in a position to deal with suppliers of both hardware and software on a more nearly equal basis.

In addition to commercial houses, there will always be a cadre of earnest "do-it-yourself" types among computer users in our own field, people who are not necessarily fascinated by the machine for its own sake, but who know enough about systems and programming techniques to see how something might be done better for their own specific needs. In a field as compact as the marine industry, such people (and their companies) should be able to profit from a

free exchange of information, methods, and support. Among the majority of users, such healthy exchange has already become a "tradition," an established part of the microcomputer culture. (This seems to be true of most novel technical interests before the practitioners get too commercial to cooperate with each other.) In the case of our own industry, in particular, it might be a good thing to remember.

* * * * *

With the increasing familiarity and availability of computers, and particularly micros, the computer has become an essential part of many designers' equipment, in almost every field. Obviously, computer-aided design and computer-aided manufacture are now important and expanding disciplines, with a widening circle of participants. As a result of this rapid and extensive growth, however, the phrase "computer-aided design" has acquired a certain "buzz-word" stature, so that it is now used as a convenient short-hand expression for <u>many</u> of the computer applications that prove useful to designers in general. In effect, CAD has come to mean many things to many people: the words may therefore have lost some of their original precision of meaning, to the point of indicating some vague and mysterious communion of the designer and his computer.

Part of the reason for this is that "design" is an extremely generalized process, while the computer is itself an extremely generalized tool. To establish a framework for looking at the computer in terms of what the designer can <u>do</u> with it, let us agree that the computer represents an extension of three familiar tools of the designer:

(1) The computer is a calculator.

(2) The computer is a notebook or file.

(3) The computer is a sketchpad or a drafting table, with instruments.

To these three basic roles we might add a fourth: the computer is a communication device, an extension of the stamped envelope or the telephone.

These are very mundane definitions, of course, yet there are a few facts that make these simple roles into a powerful tool for the designer. First, there is the fact that a single machine (with the appropriate set of electromechanical attachments and operating instructions) can be used for mathematical analysis, storage and recall of information, and graphic presentation of the results of design. The computer is a generalized machine doing a generalized job. (Note the use of the word information, rather than data. The computer can store and recall not only externally meaningful numbers, such as weights, weld-lengths, costs, and the density of water, but it can also store and recall the procedures that a designer might use to relate these numbers to the design of a vessel.)

Second, there is the fact that a computer does best what human beings usually find difficult: the routine, dull, nitpicking business of repeated calculations, filing and sorting, and drawing accurate lines on a sheet of mylar. (The computer, unlike the human, is not subject to boredom, wandering attention, stray thoughts, cramped fingers, or eyestrain.) Conversely, the human being does best exactly what the computer does poorly, or not at all: recognition of a pattern or a problem for which there are no predefined solutions, intuition, and general learning. (The man, unlike the computer, is not bound within the circle of a mere algorithm.)

Third, there is the fact that what the computer does, it does fast.

Finally, the computer provides a means by which the design and manufacturing processes can be "integrated," a word which has also acquired more than a hint of buzz over the past few years. It means this: design is the creation of a specific idea (often a very complex idea) of a product, from a mass of data, requirements, experience, and even intuition. Manufacture is the creation of a physical product from this idea, plus material, plant equipment, and labor. The essence of "integration" is that these two processes are so closely related (or should be) that many of their elements can be shared by both: calculation and analysis, storage and recall of information, the production of drawings and other forms of communication, and even the control of the building process. This sharing of information between the design and production processes, if it is done right, is at the heart of modern industrial aims. It is particularly applicable to the creation of large, complex products, such as commercial vessels, where quality of the product and profitability of the creative processes are the two related basic objectives.

It is a truism that no computer can <u>do</u> design. Design is an act of creation, and while the machine may manifest "intelligence" (according to definitions largely constructed by computer scientists), no pure machine system has yet shown signs of creativity. However, the system composed of the designer and the computer is a creative partnership, in which each member contributes effort according to the kinds of work which he (or it) performs most efficiently. It should go without saying that in a properly designed system it is the man who uses the computer, and not vice versa, although the system "uses" them both. Here endeth the lesson.

* * * * *

The objective of this brief overview is to provide some preliminary insight into the areas where computer aids can be applied to design tasks in the small yard. The details of both hardware and software operation will be left to the relevant owner's manuals and program documentation. Here, we will introduce some of the underlying concepts, survey the most important application areas, and provide, if possible, some general guidance for the selection and assessment of available systems and programs.

In particular, we will be looking at these concepts from the point of view of the average user, the designer who needs results, rather than the computer scientist who needs to know what is happening inside the machine or inside the program. Furthermore, the application of microcomputers will be stressed, in the belief that these ubiquitous and relatively inexpensive systems will offer the best value in most small shipyards.

## ROUTINE CALCULATIONS IN THE SHIP DESIGN PROCESS: PROGRAMS AND SYSTEMS

In this section we will be discussing the use of the microcomputer as an aid in the analytical parts of design work, as distinguished from graphics. That is, we will be considering the use of the computer as a combined calculator and filing system. Rather than deal with the specifics of a large number of individual applications programs, we will attempt to see what features may combine to make a given program (or package of programs) more or less useful in everyday design work.

As in most engineering fields, ship design is characterized by a number of tasks involving lengthy and repetitious calculations: preliminary lines development, fairing, hydrostatics, and performance prediction, to name but a few. In most cases, these tasks involve not only repeated calculations, which may be quite simple individually, but which entail a high degree of organization and bookkeeping detail. For many of these tasks, one of the most important feature of any method, whether it is manual or computerized, is the degree to which the method allows a convenient organization of the task itself. For this reason, the structure of the procedure or program is very important.

As an example of program organization, consider the case of hydrostatic calculations. The procedure can be viewed as consisting of three distinct parts: input, computation, and output. Formally, input to the program consists of a few underlying "constants," such as the definition of units, and the entry of water density, plus a geometric description of the hull form. Usually, this description consists of the entry of certain stations, giving their location along the length of the hull, and the definition of a number of points on each station, each point being defined by a height relative to some

arbitrary baseline, and an offset, the transverse distance of the point from the centerplane of the vessel. In addition, for purposes of calculation, some particulars of the vessel may also be required: length, beam, and design draft, for example. The desired waterlines and trims for calculations will also be entered, as the designer sees fit.

The method of entry for offsets may be manual, by measuring from a body plan, or it may be aided by the use of digitizer, which will described in greater detail subsequently. Furthermore, the input process may be direct from the designer's notes or calculations into the computer, via the keyboard, in which case the user must note and use the exact order in which the program requires the input, or the program may provide prompts for the user to enter certain data at certain times during the input process. The input data may also be transferred from the memory into a disk file for subsequent checking or modification, and because the offsets will be used at several other stages of the design process.

Regardless of the details, once the input data is available to the computer, the computation of hydrostatic results can proceed. In most application programs, computations may be performed to yield a number of different sets of results: displacements, moments and centers, form coefficients, moments of inertia, etc. The algorithms used for these calculations vary. Some are merely computerizations of the standard tabular forms used by naval architects for decades, while others may first require the computer to calculate interpolated stations and offsets before performing the usual numerical integrations.

Finally, after the computations have been performed, the results may be displayed on the screen, printed out on hard copy, or stored in a results file

on disk for further reference and work.

The single-purpose program, such as a hydrostatics program, may be either interactive or non-interactive, the major difference being whether the program prompts the user through the input, calculation, and output phases, and whether the program permits the user to specify exactly what calculations he wants to have performed during the execution of the program. From the point of view of the user, it is often more convenient to deal with an interactive program, since he is then reminded by the program of exactly what kinds of data must be entered, and the required order of data entry is in effect remembered by the program itself. Furthermore, the user may save time by deciding whether he can omit certain calculations, or add new waterlines and trims for computation, as the results develop from previous conditions. On the other hand, interactive programs are inherently larger and more complex, since they must contain the material to generate prompts, read and interpret the user's responses, and transfer to the requested computational parts of the program.

Single-purpose design programs, as the name implies, are those that handle one particular aspect of the design analysis, and nothing else. Whether they are interactive or not, these programs must involve the three basic elements: input, computation, and output. However, it may be obvious that certain classes of programs require similar input items. For example, hydrostatics, lines fairing, transverse stability, floodable length, and launching calculations all require the input of hull offsets. For this reason, a significant saving of the designer's time and effort can be realized by constructing a package of programs so that one input session can establish the required information for a number of single-purpose programs.

This concept involves linking a number of programs into a unified "design system," in which various subprograms, or modules, are related by common interactive commands, directing the user through an entire session of input, computations of various kinds, and output of results. The structure of a typical design package, or system, is shown in Fig. 1. The components of the system are described in greater detail below.

## The Design Executive

In most systems, the linking of the various computational and utility modules is accomplished by providing a single program through which the user can gain access to each module as he desires. This program, which serves as a central "control program" for the design session, is often called the design executive, and may be conveniently named DEX. The DEX program itself performs no calculations, and in fact it need not perform any input or output functions except for those prompts and response evaluations that allow the user to transfer control to other modules. The DEX program must perform the following functions:

(1) Display prompts that ask the user to state what module he is interested in using next.

(2) Interpret the user's responses, which are normally the names of other modules in the system.

(3) Check the user's responses to see that he is asking for modules that are defined and available (usually in a file on one of the disk drives).

(4) Load the required module. (Often, when a specific application module is loaded, the DEX program is unloaded, temporarily, in order to free additional RAM capacity for input data, results, and the computational algorithm for the application. When the computations are completed, or at the user's command, the application module will cause the DEX program to be reloaded.

(5) Direct the transfer of data (whether input or results) to and from data files.

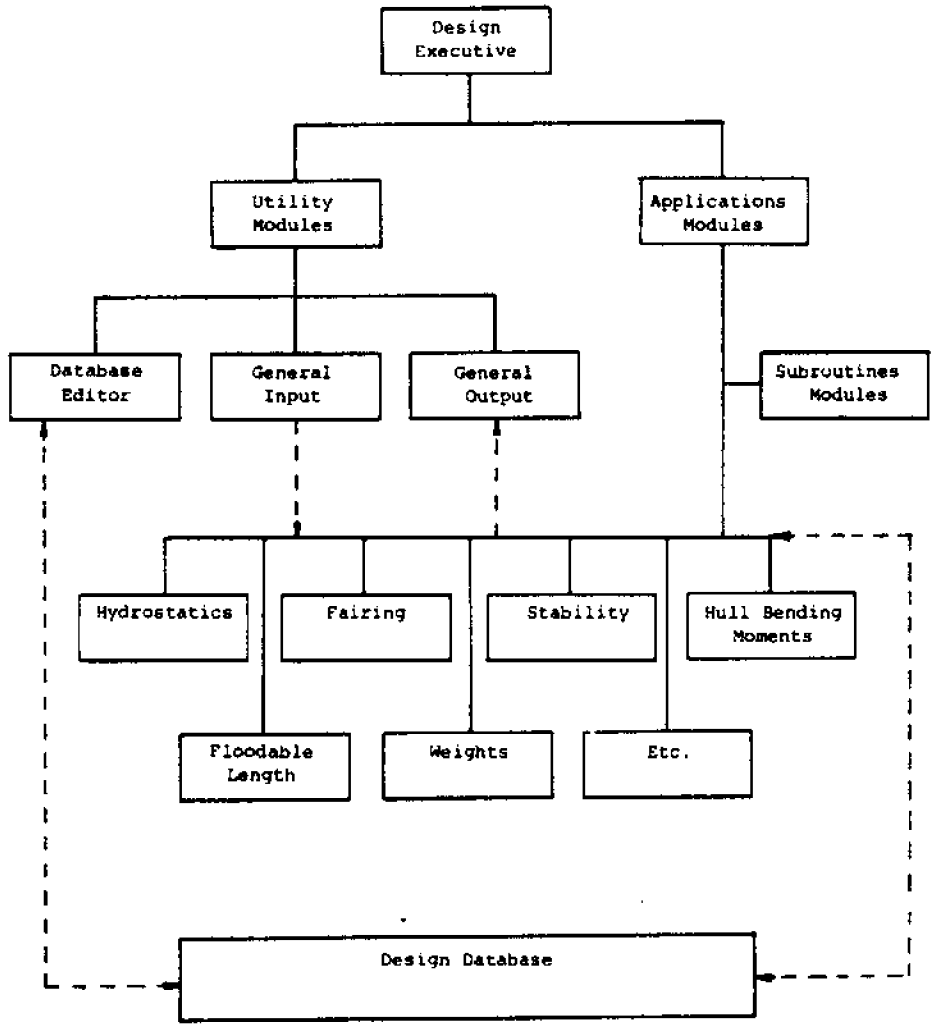



Figure 1. Structure of a typical design system. In this diagram, control functions are shown by solid lines, while dotted lines indicate the flow of design data. Each individual module shown by a single block is generally loaded into RAM separately. Thus, the over-all size and complexity of the system is limited by disk-storage considerations rather than by RAM capacity. However, the complexity of each individual module is constrained by RAM.

The DEX program generally contains calls to various subroutines of the machine's operating system to perform these functions, as well as statements to prompt and respond to the user's commands.

## Computational or Applications Modules

Each application module is, in effect, a single-purpose program. However, because of the structure of the design system, it is often possible to economize on the writing of subprograms by combining certain computational algorithms that are used by a number of modules. (For example, many of the modules may use numerical integration or curve-fitting algorithms.) Therefore, the individual applications modules may be loaded together with common algorithm material stored in another file. Thus, although each module may act in exactly the same way as an isolated, single-purpose program, it need not follow that the contents of an application program disk file (operating as part of a design system) include all the statements that would be required in a "stand-alone" program.

The applications modules may also contain interactive prompts for specialized input and computational direction, and for direction of output to the screen and/or the printer. Finally, each applications module must be able to return control to the DEX program before terminating.

## Design System Utility Modules

In most design systems, a large common body of input information (for example, ship particulars, units, water characteristics, building-material characteristics, and hull definition) may be shared by many of the applications modules. To facilitate the input of this data, and its subsequent editing, modification, and correction, it is often convenient to provide specific modules for handling these functions independent of any specific computational tasks. These subprograms are called system utility modules, and often include general input and output routines, and editors.

In some systems, applications and utility modules are arranged to make use of the concept of a "design database," a special data file configured to store all relevant design data, whether the specific values for data items are supplied by the user directly (as input) or are computed by one of the other applications modules.

## The Design Database

The design database is usually nothing more than a specific data file, intended to store and retrieve the relevant design parameters for a particular vessel. In its simplest form, the DB houses an array of numbers, indexed by position, each position corresponding to a particular design variable, such as length, beam, or draft, etc. In more sophisticated systems, access to various items in the DB can be obtained by name, rather than by location. For example, the DB may contain an item named LBP, the length between perpendiculars. In such an advanced DB, the value of LBP may be stored at any position in the file. The user, or an application or utility module, accesses the information by referring to the name LBP. At present, most design systems

for microcomputers use a rather simple form of database, in which access to information is done by location in the storage array. (This is primarily because the use of access by name entails a great deal of storage-capacity overhead, storage which is usually available only on larger machines.)

In many cases, it is advantageous to maintain two or more separate files as sub-parts of the design DB. This permits the separate storage of large collections of closely related data, such as offsets, without requiring the loading of offset information to modules that don't require it.

In addition, for a very detailed design database, it is useful to segregate various "levels" of the database: e.g., ship particulars and miscellaneous problem "constants" (units and water characteristics), offsets, and other design details, such as block, assembly, and even part descriptions.

## Common Design Algorithm Modules

As mentioned above, various applications modules may use common algorithm segments, such as integrators, spline-curve programs or other forms of interpolators, and frequently, certain tabular "lookups." To save space on disk storage, such commonly used algorithm segments may be stored in files separate from the programs that actually make use of them. In order to load and unload these algorithms during the running of other modules, special operating system commands must be used, or the submodules themselves can be designed to reload the programs that have called them at the proper time during execution.

## ADVANTAGES OF THE DESIGN SYSTEM APPROACH

To a great extent, the advantage of a unified design system over a collection of "stand-alone" design application programs depends on the convenience and "user-friendliness" of the design executive, together with any interactive features found in the applications and utility modules. Recall that the purely computational features of a program have very little to do with how much effort is involved in using the program. Besides, an efficient and accurate algorithm can be made a part of either a stand-alone or a system: the computation will proceed the same way in either case, once it gets underway.

The question that arises is this: what makes an interactive program in general, or a design executive in particular, convenient and user-friendly? The answers are largely subjective, depending on personal preference as much as anything else. Nevertheless, there are a few general considerations that would meet wide agreement. Let us examine a few "proverbs" relating to these elements.

(1) The user must see at least one of two things before he has even a decent chance of mastering a new system: either good documentation, or very good prompts. If the user hopes to learn a new system without violence, then he must either have a fairly clear picture of the over-all structure of the system (from documentation), or the program itself must take him by the hand and lead him through it. My own personal preference is for documentation, and I have lots of reasons. First, good documentation, unlike top-heavy "educational" prompting material, doesn't take up program storage. The storage that is available can be better used for good interactive prompts, error checking, and error recovery. Second, and more important, if I have

a good user's manual in my lap, I can see what I'm supposed to do while I'm actually looking at the problem, not at a screen full of explanations that I'll have to erase (and probably forget) before I get a chance to push the right button. Third, the printed page happens to be easier on the eye than the cathode ray tube, and I'd rather read extensive instructions on paper. Fourth, excessive explanations on the screen waste my time, and that isn't what I want to use a computer for. Finally, I would rather turn pages than "press return to continue." It is much easier to flip backward than it is to interrupt an entire program just to see the contents of the last screen.

(2) Interactive prompts should be clear, but short, and they should appear at times that make sense. The prompts should say exactly what you have to enter, whether it be a command, an item from a "menu," or a data entry. But they should not give you too many details about what the entry means; that should be in documentation. If the prompt is to enter an item from a menu, and there is more than one menu in the program, the prompt should tell you which menu you're on. Similarly, if there are several levels of commands, then something in the prompt should tell you which level you're on.

The best order of prompts is hard to judge until you've used a system a couple of times, at least. The ogre to avoid is the situation where you have to guess what to do next. Often enough, this happens when a system doesn't prompt you when you expect it to. Usually, it means that you will have to do something wrong in order to shock the design executive into giving you an error message.

(3) Errors in commands, menu items, or even some bad data, should result in an error message and a return to the previous prompt, rather than a system interruption or, even worse, transfer to an unexpected place. This should be

too obvious to require elaboration, but it sometimes happens.

(4) Modules should be able to be restarted after you realize you've put in bad data, without going all the way through or terminating the entire run. Again, this is a very obvious advantage that many systems do not provide.

(5) The commands or menu items should be very alphanumeric and transparent to English. If, however, menu entries do consist merely of numbers from a list, as is sometimes the case, then the list must be displayed along with the prompt to enter an item. Otherwise, it's hopeless.

(6) The program's responses to all commands or menu entries should obey the principle of least amazement. Similarly, bad data in the run should cause only module interruption, with a return to DEX perhaps, and not a system interruption that involves the loss of all computed results. Unfortunately, this is not always the case: for example, consider the calculation of block coefficient when you have forgotten to enter either length, beam, or draft, and therefore one of them is defined as zero. In calculations like this, it is fairly straightforward to write in protection against zero denominators, which should be considered as a part of error checking.

As a final note, the main advantages of the system approach are convenience and speed of operation by the user. However, these advantages can be more than negated by poor interactive features in the modules, or even worse, a dumb DEX. Systems that don't give satisfaction on most of the criteria listed above can make for some very long days.

## DESIGN AIDS IN NAVAL ARCHITECTURE AND SHIP PRODUCTION

The general utility of computers derives from the fact that any procedure that can be described in sufficient detail can be programmed. In the marine field, most of the supporting calculations for design are quite repetitious, and are therefore excellent candidates for computerization.

In a larger sense, the entire process of ship design is also one of repeated calculation, although in this greater context the "repetition" implies the continual refinement and modification of a conceptual vessel until the design requirements have been met to the designer's satisfaction. This process of successive refinements and alterations constitutes the "design spiral." In a manual design process, the ship designer customarily uses computing aids (such as a desk calculator), together with design data and rational or semi-rational methods (usually from a file system, old design notebooks, or publications). During the design process, he also maintains his own records of the successive steps in the spiral by hand, in the new design notebook.

As mentioned previously, the computer can be used to combine the functions of the computational aid, the design data file, the store of procedures, and the new design notebook, all within the framework of a single machine. Obviously, with the use of a computer, the design-system approach allows the user to proceed around the design spiral far more rapidly than he could by hand. This is true not only because of the high computational speed offered by the machine, but also because the whole process of design decision making, as embodied in the design spiral, can be compressed. With much of the required information available at a single work station, on a single display, the designer can move from calculation to calculation efficiently and quickly.

To support this process, however, the designer must have at his disposal the required applications programs, whether these are in the form of stand-alones or are incorporated as modules within a design system. The following review of application areas is not intended to be exhaustive, nor is it intended to convey the impression that every possible application has found expression in a commercially available program. Some of the more common applications, in fact, are available in a number of programs, while others have not yet been developed in a form suitable for microcomputer work.

Some applications are relatively easy to program. These may be programmed by the designer himself, with reasonable programming skills. Still others are probably too time-consuming to be programmed in-house, and the availability of programs must wait for further expansions in the design software market.

## Hydrostatic Calculations

The area of hydrostatic calculations, and related applications, has been relatively well developed commercially. This is fortunate, since hydrostatic calculations are among the most essential (and tedious) elements in the design spiral. Commercially available packages usually cover the following general types of computations:

(1) Displacements, moments and centers, curves of form, and hull-form coefficients, for various input waterlines and trims.

(2) Calculation of vessel draft and trim for a given vessel weight and longitudinal center of gravity, and trim due to shift of weight.

(3) Intact stability, including curves of statical stability and cross-curves of righting arm, either with a fixed displacement and center of gravity (free to trim), or with a fixed trim. (A feature which is not contained in most general hydrostatic packages, but which could be added with relatively straightforward program modifications, is the calculation of

required stability criteria from intact stability results.)

4. Floodable length calculations.

5. Launching calculations. (Both these and floodable length calculations follow directly from the calculation of Bonjean's curves, which are an integral part of nearly all hydrostatics packages.)

6. Tankage and compartment volume calculations. Since the data entry and numerical integration routines used to calculate total hull volume can just as easily be used to calculate the volume of a part of the hull, most commercial hydrostatics packages can also be used to calculate compartment volumes for the purposes of tankage and ullage schedules and cubic capacities. The principal difficulty arises in connection with compartments or tanks that are bounded inboard by longitudinal bulkheads not on the vessel's centerline. In such cases, additional input information and some modifications to the computational scheme are required, and these must be included in the package.

7. Damaged stability calculations. In general, any hydrostatics package that can handle compartment volume calculations with the more general compartment description mentioned above can also be used to generate damaged stability information. A number of packages that do so are now available.

## Line and Hull Fairing Programs

Fairing of plane curves, such as stations, waterlines, and butts, can be accomplished by a number of well-known algorithms, such as cubic splines. The creation of a fair three-dimensional surface, such as a hull, is a more generalized and complicated process. However, three alternative methods are now widely used in commercially available packages:

1. B-Splines.
2. Direct mathematical definition of hull forms.
3. Systematic alteration of previously faired parent hull forms.

The first method uses a relatively small number of "master stations," usually on the order of five or seven. For each of these stations, a relatively small number of "control points," generally placed at keel, lower turn of bilge (or lower chine), upper turn of bilge (or upper chine), topside knuckle, and sheer, define the shape of the station. Between master stations, the plan and elevation views of the curves formed by similar control points are faired using some form of plane-curve smoothing procedure, generally cubic splines. The result is a method by which corresponding control points can be located on stations not included in the original set of master stations. From these derived control points, the shapes of intermediate stations can be determined, and the lines of the vessel established. This method has already proven successful, and has even been implemented on hand-held programmable calculators.

Direct mathematical generation of hull forms begins with the premise that the hull offsets will be described in the form of a mathematical function. Several alternative forms of mathematical description have been proposed for

use, and a number of these have been implemented in programs. However, most of these applications programs are only now in the developmental stage, and are supported only by main-frame installations. The use of mathematical description for hulls using either developable surfaces or straight-frame construction is a relatively simple programming application. With enough demand, commercial programs for this kind of hull definition will probably be developed for microcomputer-based systems in small yards.

The alteration of parent forms involves taking offsets from an existing, faired hull, and mathematically scaling the station spacings, heights, and transverse offsets, thus resulting in an altered, but still fair, "offspring" hull form. With a suitable library of parent forms, an extremely wide variety of offspring hulls can be generated very rapidly. Microcomputer programs to alter parent forms are fairly simple to implement, although the use of this method has not yet yielded any commercial programs.

Calculations related to hull surface definition include the preparation of hull plating expansions. A number of main-frame based design systems currently include algorithms for mathematical hull definition and plating expansions. However, these programs are in general rather large and complex, and versions suitable for microcomputers are not yet commercially available.

## Weight Schedule Programs

Calculations of weights and centers of gravity are simple but repetitious applications of plain arithmetic. The computer can easily be used to keep track of vessel weights and centers, using weight information supplied directly by the user, but this is an almost trivial application. A more advanced weight program would include estimation equations for various fairly

detailed items of weight.

Ideally, a weight program should enable the user to proceed from a fairly gross division of weight items (in the early stages of design), to more detailed estimates at a later stage, and finally to perform exact weight calculations at the level of final design. Presently, only a relatively simple weight and center editor is commercially available; however, this could be extended to include weight estimation routines with modest programming effort, provided reliable weight data and some analytical skills are available at the yard.

## Performance Prediction Programs

The usual methods of preliminary resistance prediction for displacement vessels are based on systematic series results, and are usually calculated using regression methods. Such results are widely available for large commercial vessel hull forms, such as Taylor Standard Series and Series 60, and for smaller vessels such as the BSRA trawler series and the Ridgely-Nevitt high displacement-length series. For chined displacement hull forms, typical of many offshore work and supply vessels, results are not presently widely available, but they could be, if a suitable body of information could be gathered from cooperating designers and model basins.

For planing craft, a number of similar empirical data presentations also exist: Series 62, 63, and 65. In addition, the Savitsky semi-analytical methods (and related corrections) have been implemented on a number of small computer systems, and some of these programs are now commercially available.

## Propeller Calculations

In many cases, propeller selection consists largely of inspection of design charts, together with the application of rough empirical cavitation data. The process is generally one of repeated calculation and graphical solution on the charts themselves. Usually, the original design charts are derived from test data, and are drawn by cross-fairing experimental results, with the aid of some regression techniques. In such cases, the regression equations themselves can be computerized, and the tedious use of the charts avoided, although the iteration process would still be left under the designer's direct control. Simple cavitation criteria are also easily programmable.

As a refinement, computerized optimization schemes for the selection of propellers from design charts have already been implemented, although these are not yet commercially available to microcomputer users. With increasing use of the micro, such software will undoubtedly become available in the relatively near future.

Relatively straightforward rational design methods, such as lifting line calculations, may be the next step for microcomputer-assisted propeller design. More sophisticated propeller design methods, such as lifting surface programs, have been used with success on mainframe installations. However, the computational requirements for these programs are extremely large, both in storage capacity and run time. Thus, the chances of the microcomputer coming into its own for such calculations, on a competitive footing with larger systems, are slight, at least for the present.

## Ship Dynamics Calculations

The use of the computer for seakeeping and maneuvering problems is well established, although the commercially available programs are presently implemented on larger computers. For certain applications, however, empirical regression methods using either full-scale or model test data may become available if a demand for microcomputer-based analytical tools develops. These possible areas may include seakeeping (i.e., motions and accelerations) for small craft and certain standard commercial vessel hull forms.

For maneuvering and position-keeping problems, a possible future application for the microcomputer might involve direct time-step simulation. Much model development work will have to be done before this application is realized commercially, however.

## Structural Calculations

Simple structural analysis programs have already been written for microcomputers, and a few are commercially available. For the most part these involve the analysis of relatively simple members, such as beams, plating, and trusses. In addition, the calculation of hull section modulus is a straightforward matter of programming.

More sophisticated structural analysis methods, such as finite-element analysis, are currently available for mainframe and minicomputer installations. The practical use of these methods on microcomputers is now in a fairly early stage. As the capacity of microcomputer systems increases, however, the use of finite-element analysis for strength, deflection, and vibration problems may become a part of the microcomputer repertoire.

More immediately, the computerization of empirical structural design methods, such as the ABS rules for steel vessels, should become available within the near future. In addition, the use of the microcomputer in empirical estimating methods for hull girder and local loads should prove a valuable design aid.

## Other Design and Production Applications

The potential uses of the microcomputer for general ship design are many, although only a few applications are currently available in commercial form. Software specifically intended for other branches of engineering, such as electrical and mechanical engineering, may also prove useful in the shipyard. Applications programs related more directly to ship production are also of great potential usefulness, although these are described elsewhere in this series. Some micro programs already exist in the following general areas:

- Electrical system design calculations.
- Piping and pump head and flow calculations.
- Heat balance and steam table calculations.
- Control systems.
- Generalized optimization algorithms.

Program areas related to ship production include:

- Statistical packages (useful for developing new regression models, and for quality control and productivity analysis).
- Database management systems (useful for maintaining part descriptions, pallet lists, specifications, and standards).
- Cost estimation programs for various kinds of work, including structural steel and some items of outfit equipment.
- PERT and critical path programs.

In many cases the use of the computer as a design aid, even in its role as a calculating and data storage device, is supported and extended by graphic capabilities. For example, in many applications, graphic presentation of both input and results may facilitate the design process by allowing the designer to see and interpret information more rapidly: body plans, curves of form, resistance and propeller curves, etc.

Thus, although computer graphics may be thought of as a distinct application, an extension of the conventional drafting methods used in conveying design information, it may also be seen as an integral part of many computational design applications. In any case, computer graphic applications are a vital factor in the application of microcomputers to the design process. In the remainder of this paper we will consider some of the hardware and software concepts that are most useful in microcomputer-based design graphics.

## MICROCOMPUTER GRAPHICS HARDWARE

As mentioned previously, the development of computer-aided design has been greatly enhanced by the expanded ability of the computer to deal with input and output in graphic form. In the field of microcomputer-based design systems, this development has therefore depended to a great extent on the capabilities of specialized peripheral hardware, as well as on advances in the technology of the microprocessor itself. In this section, we will take a more detailed look at some of these peripherals.

### Digitizers and Tablets

A drawing can always be described as a set of points, that is, x and y coordinates, together with the lines and curves connecting them. Prior to the development of the digitizer, these coordinates, and the computer instructions to draw the required lines, had to be laboriously entered in numerical form using a conventional keyboard.

The digitizer is a table or board through which the x and y coordinates of a point can be entered electronically by the use of a hand-held cursor. This can be moved to any desired location and then caused to give a signal which transmits the coordinates directly to the computer without manual measurement and entry. In its most common form, the digitizer board contains an accurately constructed wire grid which can recieve the signal from the cursor. As the cursor is moved from one part of the board to another, only a small subsection of the grid around the cursor is activated. This switching, which is handled entirely automatically, allows even a large digitizer to maintain an absolute accuracy of better than one-hundredth of an inch, regardless of the position of the cursor. Since the resolution is therefore

independent of digitizer size, there is in principle no restriction on the size of the drawing that can be handled. Digitizers of 4 ft by 6 ft and larger are common.

The cursor, or "mouse" is fitted with a cross-hair for accurate positioning of the entered points, and usually with a small magnifier as well. One (or usually more) buttons are installed on the body of the cursor to control the generation of the signal that is sensed by the wire grid in the digitizer board. (The provision of multiple entry buttons, in effect a small keypad, on the mouse, allows the operator to conveniently enter different types of points, or to "flag" certain special data points without interrupting the process to turn to a separate keyboard. The signal generated then carries this additional information to the computer when a particular button is pressed.)

Because of the automatic switching of the active region, and the resolution of points within the active region, each location on the entire digitizer surface is uniquely defined, and the process of entering data does not depend on the computer being able to "track" the mouse continuously. Thus, the cursor can be moved at will, or even removed from the surface completely and replaced, and the computer will not lose track of the location. For graphic input of detailed information, the digitizer allows the designer to enter information both to and from a drawing.

A tablet is a small, relatively low-resolution digitizer, usually about the same size as the standard visual display screen. The user enters data from the tablet in much the same way as on a full-size digitizer. Usually, however, the position of the cursor is directly "echoed" on the screen.

The digitizer or tablet can be used not only for the direct entry of graphic information (point coordinates), but also to control various computer functions. This is usually accomplished by defining an area of the digitizer surface (outside the main drawing area) as a "menu" area. Physically, the menu consists of a paper or plastic overlay taped on the digitizer. Various labelled rectangles drawn on the menu overlay are then used to define the available set of commands. By moving the cursor to a particular labelled area and pressing one of the cursor buttons, "hits" made on a defined menu area are interpreted by the computer as system or program commands rather than as graphic coordinate input. The use of a digitizer-controlled menu is an aid to user efficiency, since it frees the operator from the necessity of constantly moving between the digitizer and the keyboard.

## Visual Display Hardware

All popular microcomputer systems incorporate a visual display screen, either directly coupled in the same cabinet as the processor or in the form of a separate monitor. In most systems, this display is quite similar to a conventional TV picture tube: the image consists of a series of parallel lines, or rasters, scanned by a modulated electron beam to vary the brightness. Because of the discontinuous nature of this image, resolution may be limited by the display screen itself. In some systems, each line is further subdivided into small, discrete image units called pixels, each of which is either set on or off. Due to the finite number of pixels available, the resolution is usually not suitable for very accurate display of line drawings on the screen. However, in some more advanced graphic systems based on this type of screen, resolution can approach one one-hundredth of an inch. In either case, the entire screen surface must be scanned by the moving

electron beam.

In most design applications, however, the user is not concerned with the visual texture of the entire screen surface, as in a TV picture. Usually, the relevant information is confined to actual discrete lines, as on a drawing. In this case it is not necessary to scan the entire screen on each pass of the electron beam; in fact, it would be undesirably slow to do so. Instead, the beam merely traces out each line of the drawing on the screen. As in the raster system, it is possible to vary the brightness of the image by modulating the intensity of the beam as it traces out the drawing. However, in most such systems only two voltages are used, full and zero, so that actual brightness modulation is not available.

More importantly, in both raster and line-tracing systems the image on the screen must be maintained either by scanning the entire surface of the screen line by line, or by retracing the drawn lines, many times per second. The apparent steadiness of the image is then due to the brief persistence of screen phosphorescence after the electron beam has passed. In such systems, the image is "refreshed" continuously, and therefore this type of display is commonly referred to as a "refresh screen." Obviously, the retracing must occur before the image fades, or the display will appear to flicker. If the picture is complicated, or involves lengthy additional calculations, the retracing frequency will fall, and the image will begin to flicker. To avoid this problem, some systems incorporate an auxiliary processor, in addition to the main processing unit. The so-called "display processor" has as its sole function the maintenance of the image on the screen. This is an expensive solution, of course, and it is not particularly useful in microcomputer-based systems.

An alternative method of graphic display which avoids these problems of refresh screens is the direct visual storage tube, or DVST. In this type of display the conventional electron gun which generates the drawing beam is supplemented by a second "flood gun" and a grid electrode in the screen itself. The main drawing beam causes the grid electrode to pick up a local charge in the immediate area of the image only. Then, the flood gun causes this charged area to glow continuously, without any further retracing. Thus, the image is not subject to flicker, and there is no need for an auxiliary display processor in the computer.

However, there are some disadvantages of the DVST system as opposed to the refresh-screen system. Since the entire image is maintained by the constant irradiation of the flood gun, erasing of the image can only be accomplished by clearing the entire screen, rather than by simply modifying or updating the display information used by the refresh system. By the same token, satisfactory "moving" graphics are difficult to achieve because the picture can only be changed by rapid replacement of the entire display. However, for most design graphics applications the DVST yields acceptable capabilities, while providing a steady image and placing modest demands on the microprocessor.

In some refresh systems graphic information can be input directly on the display screen, using a cursor on the screen. The location of the cursor is usually displayed by a small cross or circle, which can be moved either by means of a separate joystick or ball control (or equivalent cursor control buttons), or by a light pen, used directly on the screen.

## Pen Plotters

The plotter is a graphic output device that produces actual ink lines on paper or mylar, replacing the conventional drawing board and instruments. The plotter consists of a drawing surface (which may or may not be the usual flat surface) plus an electromechanical linkage to support and move the pen from point to point, and to raise and lower the pen onto the drawing surface. Thus, the pen can be moved either to produce a line (with the pen "down") or simply to shift to a new position without making a line (with the pen "up").

One of the most common forms of plotter uses a rotating drum, rather than a flat table, as the drawing surface. The paper is fed over the drum, with sprockets at the ends of the drum engaging corresponding perforations near the edges of the paper to maintain alignment. The pen is carried on a holder which moves along a fixed gantry parallel to the axis of the drum. Thus, rotation of the drum, with the paper attached, produces apparent pen motion along one axis, while the actual linear motion of the pen along the gantry produces the other coordinate. Two separate servo-motors control these two motions, responding to impulses generated by the computer, to produce the drawing.

The principal advantages of the drum plotter are compactness, relatively low cost, and the fact that the <u>length</u> of the drawing produced is limited only by the length of the continuous paper roll that is fed over the drum. (Since paper rolls are typically supplied in lengths of 100 ft or more, this is really no limit at all.) On the other hand, the <u>width</u> of the drawing is limited by the width of the drum itself, minus an allowance of a few inches for the sprocket holes at the edges. (The usual drum width is 36 inches, with a useful drawing width of 33 inches.) Furthermore, the paper used must be

specifically manufactured for drum plotter applications, with the required perforations: this may add to the expense of producing drawings. Normally, however, these limitations are not critical for practical design work.

An alternative arrangement is the flatbed plotter. In this type of plotter, the gantry carrying the pen travels on a second set of bearings so that two-dimensional motion of the pen itself is produced. The paper is laid out on a flat surface, and is held in place either by a slight vacuum applied to small openings in the surface, or electrostatically. Again, the two motions of the pen are controlled by separate servo-motors.

The chief advantage of the flatbed plotter is its ability to draw on any kind of paper, cloth, or mylar, without the need for special perforations. Furthermore, with a flatbed plotter, the user is able to view the entire drawing surface at intermediate stages during the drafting process, as in conventional drawing-board work. Combination digitizer-flatbed plotters are now under development, yielding a convenient and compact installation for both graphic input and output.

In most commercial plotters, the pen holder actually incorporates three or four separate pens, so that a selection of different line widths or different ink colors can be made available without manual replacement of the pens while the drawing is produced. The selection of the pen is made automatically according to instructions from the computer.

In many cases, the plotter can be driven either directly by the computer or from instructions previously stored on magnetic media, such as tape or disk. Some plotter systems are also capable of recieving instructions over a telephone line for direct graphic communication with remote computers.

Several of the more sophisticated plotters actually contain their own microprocessors. This permits elementary figure components (such as circles, arcs, ellipses, and other curves) to be pre-programmed into the plotter's repertoire, allowing relatively simple instructions to the plotter to be translated into more complicated shapes.

All pen plotters, regardless of their arrangement, share a number of obvious drawbacks. First, the electromechanical system driving the pen is subject to purely mechanical limitations on drawing speed, arising not only from maximum motor speeds, but also from the maximum accelerations achievable by the drum or gantry and pen holder. Thus, the production of a drawing by a plotter often requires many minutes. While this is obviously incomparably faster than conventional drafting, it must be remembered that this operation ties up the microcomputer, too, unless the plotter is driven by off-line magnetic media rather than by the computer itself.

Second, the quality of the drawing produced depends on the general cleanliness of the drawing surface and, especially, on the careful maintenance of the pens. Although plotters can use a wide variety of inking methods, such as rapidograph, nylon point, or even ball point pens, problems with line quality will result if the pens are left exposed to air for lengthy periods without use.

## Electrostatic Plotters or Hard-Copy Units

An alternative to the pen plotter is the electrostatic hard-copy unit. This device consists of an electrostatic matrix which prints small dots on charge-sensitive paper. The drawing is produced in a raster-type format, as described in the previous section on visual displays. Since matrix densities

of up to 200 dots per inch are available, resolution is high enough to produce acceptable line quality for many applications, although not all.

Electrostatic units, since they require no mechanical movement of a pen holder, drum, or gantry, yield extremely high plotting speeds, and plotting widths up to 72 inches are currently available. However, the electrostatic plotter requires special paper and chemicals. Furthermore, the drawing information must be preprocessed into raster format for plotting, if it is initially in another form for visual display on the screen, such as DVST.

As a secondary application, the electrostatic hard-copy unit can also be used as a printer, usually to reproduce the contents of a raster-format visual display screen. Its advantage over a conventional dot-matrix or daisy-wheel printer is speed, pure and simple, while the character quality is usually inferior to a good quality daisy-wheel.

## Graphic Display Using a Conventional Printer

The conventional printer is, of course, a basic hardware requirement for almost every serious microcomputer installation, regardless of the area of application. Printers have therefore been described in adequate detail in another paper in this series. The application of the conventional printer to graphic output is obviously extremely limited.

Nevertheless, for very simple graphics, such as bar charts, histograms, and even rough schematic drawings, the conventional printer can be manipulated to give an approximation of graphic output. Thus, for these simple graphic applications, a printer may be used as an alternative output device, reducing the load on the pen plotter or electrostatic hard-copy unit. However, the

conventional printer can in no way be regarded as an inexpensive substitute for a pen plotter or hard-copy unit. For the production of real drawings, and therefore for nearly all graphic design applications, the resolution available from a conventional printer is obviously inadequate.

## BASIC CONCEPTS IN COMPUTER GRAPHICS

Computer graphics is at the heart of computer-aided design, and the field of graphic applications is extensive. Indeed, entire college-level courses have been rightly devoted to graphics alone. At the same time, the operational details of graphic systems for various microcomputers are diverse and highly machine-dependent. Thus, for the purposes of this brief survey, a detailed description of the different structures, commands, and options of various graphic systems must be left to the applicable system documentation and user manuals.

However, there are several key concepts that are relevant to nearly all systems for graphics applications, regardless of the different ways in which these concepts are realized in various machines and software packages. In the following sections we will introduce some of these fundamentals, with an eye toward defining some of the capabilities that make computer graphics such a powerful tool for the designer.

### Coordinate Systems and Pen or Cursor Control

As mentioned previously, the graphic information needed to define a drawing consists of a set of points, specified by x and y coordinates, the computer instructions to draw lines or curves between these points and to describe the various lines (pen thickness and type of line), and the information to be incorporated into the drawing in character form (annotation and labelling). The first requirement, however, is the ability to specify the location of points on the drawing, defining a coordinate system.

For most drawing applications, the usual rectangular coordinate system is

adopted, with the x direction horizontal and positive to the right, and the y direction vertical and positive up, as seen on the display screen, and on the drawing. (The variable name actually associated with the two dimensions is arbitrary, of course, but x and y are used in most mnemonics associated with graphic systems.)

To begin with, every graphics device has a built-in coordinate system, corresponding to the surface of the display screen, digitizer, or plotter. This coordinate system is, in effect, determined by the electronic and mechanical details incorporated in the device. It cannot be altered by the user, either with respect to the location of the origin, the units, or the scale. This intrinsic coordinate system for a particular device is referred to as the absolute coordinate system.

For example, a visual display screen on a particular machine may be defined in absolute coordinates as 130 units wide and 100 units high, with the absolute origin at the lower left corner of the screen. Thus, the absolute coordinates of the center of the screen would be (65, 50). Note that the physical length of a "unit" is defined only in terms of the size of the display screen: it has no other significance. Thus, we can produce a drawing on the screen in absolute screen coordinates, even though we have not yet defined the scale. In this particular system, for example, the command sequence

```
MOVE (0,0)
DRAW (130,0)
DRAW (130,100)
DRAW((0,100)
DRAW (0,0)
```

will produce a solid borderline around the useful screen area. (The command MOVE causes the "pen" to move to a designated point, without producing a line,

while the command DRAW causes the "pen" to move from its current position to the designated point while producing a line.) Note that the above sequence of commands leaves the pen at the point (0,0), namely, the lower left corner of the screen. Thus, if the next sequence of commands is

```
DRAW (130,100)
MOVE (0,100)
DRAW (130,0)
```

the result will be to draw the two diagonals of the rectangular screen, and leave the pen at the lower right corner.

Similarly, a digitizer (or flatbed plotter) may be set up with absolute coordinates 72 by 48 units, again with the absolute origin at the lower left corner (the lower left corner of the useful drawing area is the conventional absolute origin for most devices and systems). However, for the digitizer or plotter, the physical drawing surface may also happen to be 72 inches wide and 48 inches high. Thus, the length of the absolute coordinate unit would, in this case, be exactly 1 inch, a convenience.

In fact, any drawing, regardless of how complicated it is, can be generated in absolute coordinates alone. It does not follow, however, that absolute coordinates are most convenient for the designer: they are certainly not.

For example, in drawing a profile of a vessel, the designer will want to position the baseline at some distance above the bottom of the drawing area, and he may want to refer to the height of points (y-coordinates) from this baseline. Similarly, he may also want to define the zero point for length coordinates in the profile (x-coordinates) at, say, the fore perpendicular, which he will not want to be at the very edge of the drawing area. Finally,

the designer will want to enter both heights and longitudinal locations in the profile not in terms of screen width, or even absolute inches, but rather in terms of the actual dimensions of the vessel, expressed in full-scale feet or meters. In fact, he could scale his desired drawing to the absolute coordinate system of the screen or plotter, and enter all his points in absolute coordinates. But the computer can perform the required calculations far more easily and rapidly, and the user should be more than happy to let it.

Therefore, instead of scaling manually and entering drawing information in absolute coordinates, the user must be able to define a new set of coordinates that are convenient for him. On any worthwhile graphic system, this transformation can be set up with a few elementary commands. Thereafter, the user will enter points not in absolute coordinates, but in his own defined coordinate system, in which he can:

(1) Place the origin at any convenient location on the drawing. For example, he may want to instruct the computer that the origin in the profile is to be at the intersection of the baseline and fore perpendicular, located, say, 24 inches up from the bottom of the paper, and 10 inches from the left hand edge.

(2) Define the scale so that coordinates can be entered directly in terms of full-scale units and dimensions. For example, he may want to instruct the computer that his defined input units (full-scale feet) are to be interpreted as one-eighth of an inch on the digitizer or the plotter, or 1/400-th of the screen width on the visual display. At the same time, he may want to define the positive directions as either left or right, up or down, regardless of the machine's internal conventions.

This defined coordinate system is often referred to as the user coordinate system, the scaled coordinate system, or the relative coordinate system, to distinguish it from the absolute coordinates of the device, as shown in Fig. 2.

Various systems have different forms of command for establishing the user

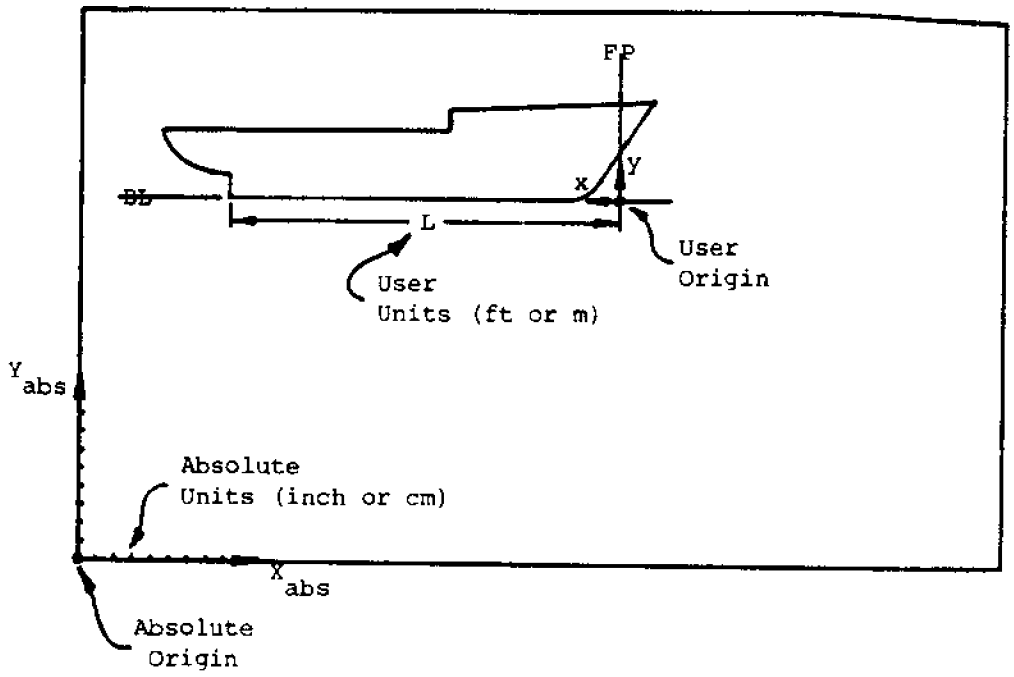



Figure 2. Absolute and user-defined coordinate systems. In this case, the user origin for the profile is defined as the intersection of the baseline and fore perpendicular. Note also that the positive x direction in the user-coordinate system is directed to the left, since the bow is drawn to the right. In many systems, this reversal of direction is achieved by entering a negative scale factor.

coordinate system. Most, however, simply allow the user to enter the desired absolute coordinates of the origin of the new system, and the scale factor, for the purposes of digitizing or plotting, and to enter an over-all image size and range of user coordinates for the purposes of setting up the screen display.

In summary, the provision of a user-defined coordinate system allows the designer to enter graphic input in a form that is most convenient for him, rather than relying on an arbitrary built-in coordinate system. Any good graphic system should have this feature.

Once the designer has set up a scaled coordinate system, some graphic packages also permit some additional flexibility in the commands used to control the movement of the pen or cursor. The usual method of pen movement control is referred to as fixed-origin control, in which the location of any specific point is given with respect to a unique, fixed origin. In the above example, a profile, this origin was specified in user coordinates as the intersection of the baseline and the fore perpendicular. Thus, any point is entered in terms of its x coordinate (from the perpendicular), and its y coordinate (from the baseline), as shown in Fig. 3.

An alternative is the so-called trailing-origin control method, shown in Fig. 4. In this approach, the origin of the system moves from point to point, following the position of the cursor. Thus, for example, in trailing-origin mode, the command to MOVE (10,15) would be interpreted as an instruction to move the cursor 10 units in the positive x direction and 15 units in the positive y direction, from its current position (the trailing origin being the current position of the cursor). In some applications, trailing-origin mode is more convenient than fixed-origin, since it permits the direct entry of

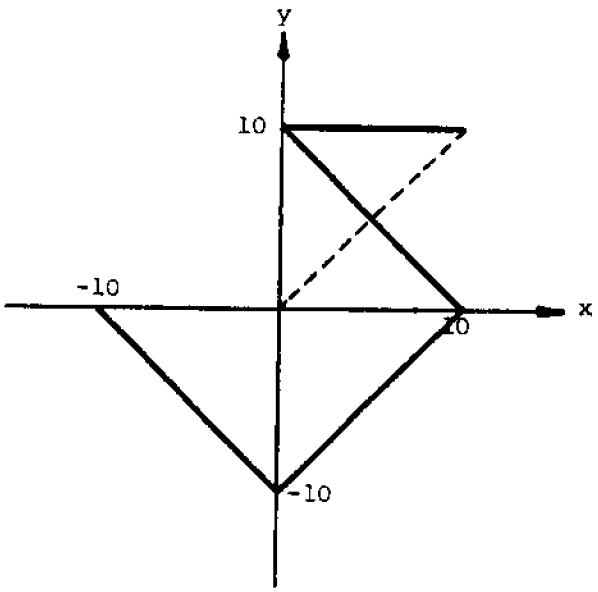



Figure 3. Fixed-origin cursor movement. The cursor is assumed to begin at the origin, (0,0). The command sequence is:

```
MOVE (10,10)
DRAW (0,10)
DRAW (10,0)
DRAW (0,-10)
DRAW (-10,0)
```

Note that each cursor movement is specified by the coordinates of the end-point of the line segment.

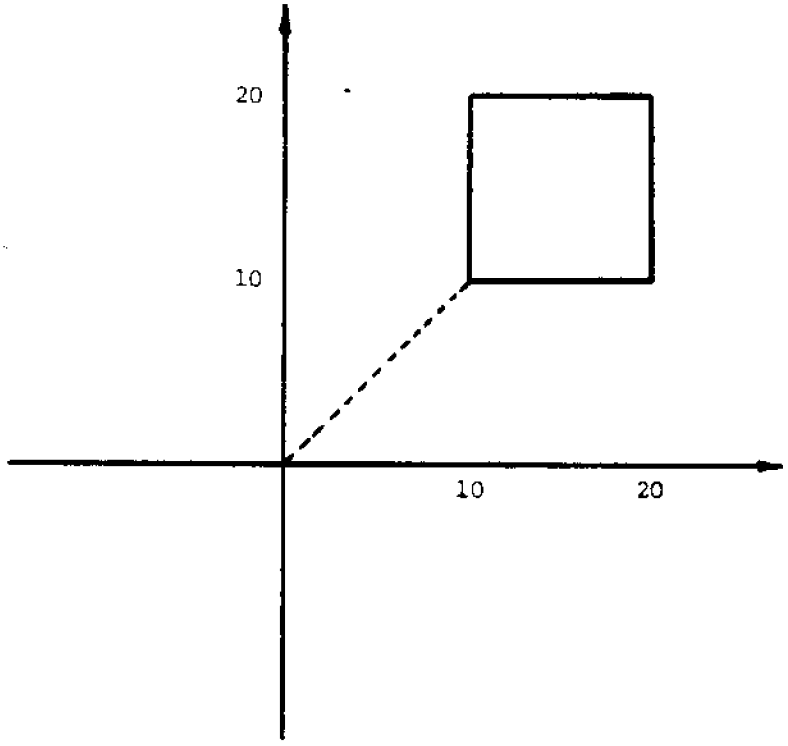



Figure 4. Trailing-origin cursor movement. The cursor begins at the origin, (0,0). The command sequence is identical to that of Fig. 3. In this case, however, each cursor movement corresponds to a vector beginning at the current position of the cursor.

individual line lengths and directions, rather than end points in the fixed coordinate system. Note that the scaling of data from full-scale to drawing units is exactly the same in trailing-origin mode as in fixed-origin: that is, the user may still enter the cursor movement commands in terms of full-scale units, such as feet, while the computer automatically transforms this motion (as seen on an output device) according to a previously entered scale factor.

## Work Space and Menu Space

As noted previously, the entire available area of a graphic input device, whether it is a digitizer or the display screen (used with a cursor control), need not be filled by the drawing. Often, in fact, it is more convenient to define a subsection of the device's available area as the working space. Thus, the drawing will be confined to a particular section of the display or digitizer, known as the work space, or the drawing area. The remaining portion of the display or the digitizer is then free for use as an auxiliary area, for entering system commands, calculating and displaying results, and other functions that are not intended to be shown directly on the drawing.

Commonly, this auxiliary area is devoted to the display and manipulation of a menu, consisting of a physical representation of the available system commands. The principal reason for defining a menu space along with the work space (Fig. 5) is to save the user from the inconvenience of dividing his attention between two separate input devices, such as the digitizer and the keyboard. Where the interaction is confined to the visual display screen, the provision of an auxiliary display area also eliminates the inconvenience of "cluttering" the drawing display with system prompts and commands. This means

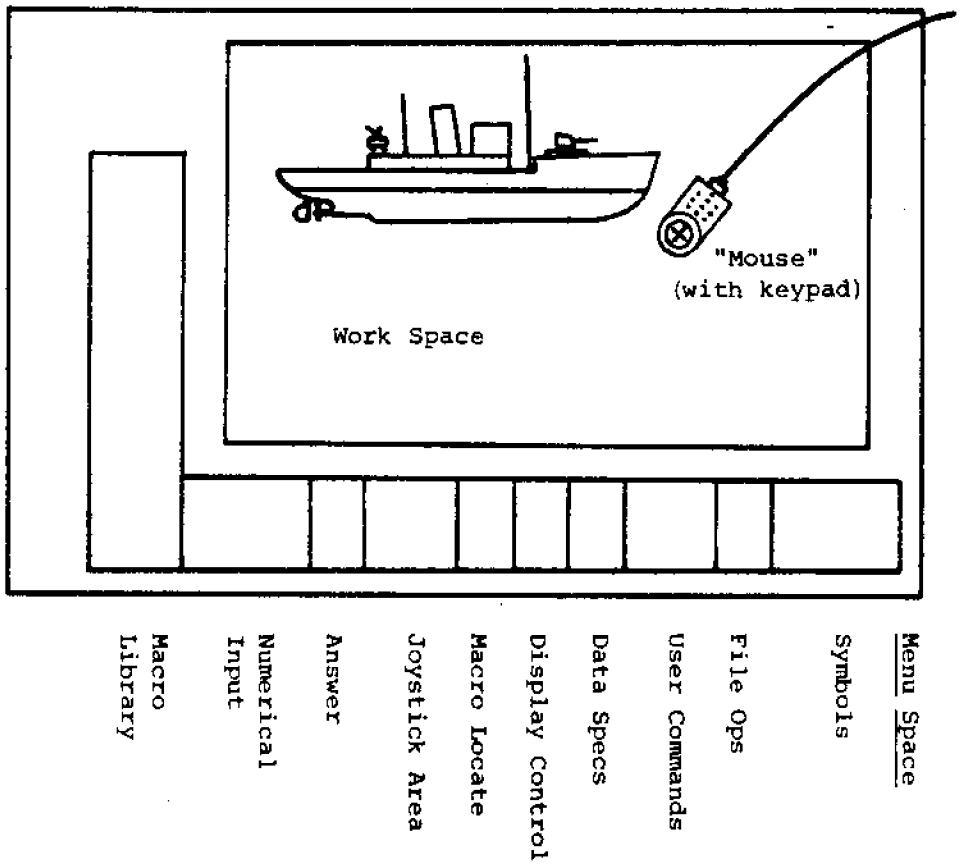



Figure 5. Work space and menu space, shown on a digitizer board. The menu regions shown are similar to those of the GCADS graphic design system, currently implemented on PDP-11 minicomputers. (Adapted from C.R. Besant, Computer-Aided Design and Manufacture, 2nd Ed., Ellis Horwood, Chichester, England, 1983.)

that the user will not have to clear the entire display and instruct the computer to redraw the main display area periodically in order to eliminate the clutter.

The ability to define auxiliary areas independent of the main drawing display, on the same device, is an important feature of good graphic systems.

## Three-Dimensional Graphic Input and Display Features

Within the drawing area itself, the user may wish to define several regions, usually corresponding to several views of the same part or assembly. As a more specific example, the regions so defined may correspond to the profile, half-breadth, and body plan of a hull. Within each region, separate two-dimensional coordinate systems may be defined. If these coordinate systems are chosen properly, the result is the ability to enter a point in three dimensions. For example, a point on the sheer may be entered in the profile, and then in the half-breadth plan, giving its position in three-dimensional space, and on the body plan, as shown in Fig. 6.

The details of three-dimensional data entry from the digitizer vary widely from sytem to system. However, the key concept is the ability to divide the work space into separate regions corresponding to orthogonal views of the object. For most normal design applications, a particular reference plane (such as a station, waterline, or buttock) is selected first, by entering its location in an appropriate view. (For example, the location of a station may be entered in the profile or half-breadth.) Then, by shifting the cursor into the proper viewing region (for stations, the body plan), the points on the specified station will be entered in two-dimensional form, with the computer automatically supplying the required third dimension.

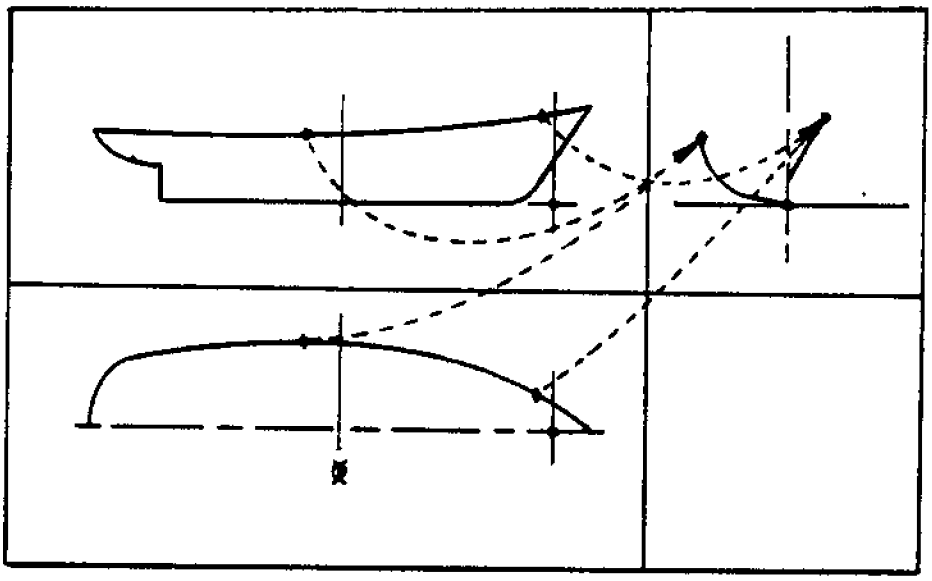


Figure 6. Direct three-dimensional definition of an object on the digitizer. A point entered at a specified station in both profile and half-breadth views is located in three dimensions, and shown in the body plan. For the usual graphic format of the body plan, special additional programming is required to cause the display of the forebody on one side of the centerline, the afterbody on the other.

Obviously, the ability to obtain a three-dimensional representation of an object is most convenient if the system permits this division of regions within the drawing space. The three-dimensional nature of the object can then be directly entered from the digitizer as a single, continuous input process.

However, even in simpler systems that do not permit this simultaneous viewing, an object can be defined in three-dimensions by sequentially entering two-dimensional data at various cross planes, although the entire three-view drawing may never appear on the display at the same time. Needless to say, with very complicated three-dimensional objects, such as piping assemblies, the ability to see and work with the three views at the same time is a major convenience to the designer.

The definition of an object in three dimensions permits the designer to perform numerous operations, both computational and graphic. For example, in hydrostatic programs, the required input is a three-dimensional description of the hull form, although the details of the input process are not essential to the operation of the program provided the points are entered in the correct order.

For more complex graphic operations, such as rotation, perspective drawings, and overlaying of one drawing onto another, the three-dimensional nature of the object must often be described in greater detail. Without a 3-D input process, the labor required to produce this finer description might be excessive. Thus, for advanced graphic design, the ability to digitize on a three-view drawing is a very important factor in system selection.

## Windowing and Blow-Up

When viewing a drawing on a visual display screen, it is important to remember that the display is usually only about 8 by 11 inches, while the drawing may represent an area (on the digitizer or plotter) far larger in size. In fact, with an auxiliary display area occupying part of the screen, it is not unusual to have a main drawing display area only one-twelfth the size of the actual digitized or plotted drawing. Thus, the details of the drawing may become far too small for inspection on the screen.

For this reason, most graphic systems incorporate the ability to define a window, usually by entering opposite corners of a rectangular portion of the displayed drawing, and then blowing up the defined portion to the full size of the display. The command sequence varies from system to system, of course, but it is an essential concept.

## Finding a Previously Entered Point

Often, the user may want to position the pen or cursor accurately at a previously entered point, in order to add (or even to remove) a point or line in a previously worked drawing. Of course, this may be done by calculating the precise coordinates of the desired point and then moving the cursor to this exact point with the usual commands. Sometimes, though, this is not easy. Alternatively, point and line revisions can also be accomplished by editing the disk files corresponding to the drawing. However, to facilitate this operation by direct graphical means, some systems have a feature that may be called (generically) the FIND command.

To FIND a previously entered point in a drawing, the user simply moves

the cursor manually (using a joystick, for example) into the vicinity of the desired point. Using the FIND option, the computer searches for the entered point nearest to the cursor, displays the coordinates of the point for checking, and then moves the cursor to this "found" point. Failure to find any previously entered point within some small radius of the cursor usually results in a message to that effect.

## Data Levels in Graphic Input and Output

In many design applications, and quite often in ship design, the geometric elements (e.g., points and lines) that might be shown in one particular drawing are of computational or graphic significance in many systems. Nevertheless, all of these systems would not normally be shown on the same drawing, for reasons of visual clutter.

To cite one example, the points describing the shape of hull plating at a particular station must be related to the locations of longitudinal hull stiffeners at that station, and must also define the outer edges of transverse frames or bulkheads at that station. However, these structural details would not normally be shown on the lines drawing. Similarly, the structural steel drawings are obviously of importance to the layout of piping, penetrations, and other internal arrangements, although these details would normally be shown in another drawing.

In conventional drawing practice, it would be usual to prepare these related system drawings by overlaying or tracing the relevant parts of previous drawings. In fact, however, with a computer-based graphic system, much of this labor becomes unnecessary. Significant points may be entered once, and subsequently used by various system designers, working within

the same graphic system. Thus, an entire set of system drawings may be supported by a single graphic database, in which the various system designers can have access to the graphic data supplied by the others.

Within the files describing a particular drawing, or rather a particular geometric area of the vessel, the user may define an arbitrary division of the data into several levels, each level corresponding to an individual system. By calling on one particular level, the operator can then display, edit, and process just those items of information corresponding to the system of interest. While this single level is active on the displayed drawing, all other levels remain available for rapid checking of interferences, and for modification as necessary. Available systems provide up to 16 data levels.

By storing and using graphic data in this stratified way, the designer can reduce the required display and processing time (particularly with regard to graphic input), reduce the requirements for manual editing of the graphic support files, and still obtain clear drawings of the various categories of data corresponding to particular systems.

## Filing and Display Functions

Data entered graphically is stored within the computer system in a portion of random access memory, just as any other form of data would be stored internally. The region of internal RAM storage used for this graphic information is often referred to as the "work space," corresponding to the term used for the drawing itself. Long-term storage and retrieval of information is usually managed through disk files. Therefore, the graphic system must be able to handle storage and recall functions during the course of a graphic session.

The following forms of filing and recall operations should be provided within the graphic system.

(1) Transfer from the work space (RAM) to a designated disk file. The specified file must not be locked. The contents of the work space, which may consist of only a single level of data, will be transferred into the file, replacing the corresponding level of data in the file.

(2) Transfer from a disk file to the work space. The contents of the selected file will be added to the work space in the currently active level of data, and displayed.

(3) Direct display from a disk file. The contents of the selected file will be displayed on the screen, but without modifying the work space. In this mode, the user will also be able to display the disk directory or other filed material without disturbing the data in his graphic work space. This is a very important convenience if the designer is to have access to other forms of computerized information while he is working on graphics.

(4) Creation of new files on the disk. This is another important utility that should be available without interrupting the graphic session.

(5) Destruction or emptying of disk files.

## Editing Functions

Editing describes the user's ability to make amendments to the drawing or the RAM work space in the most convenient way, without having to rework the entire drawing. Usually, the editing function includes the ability to move or delete entered points, and occasionally lines. Often, there is also an editing option to delete entire levels of data, and to edit character information and special symbols (such as macros, described in the next section).

The basic element, however, is the point editor. Usually, this command is similar to the FIND option mentioned previously. The user positions the cursor manually, and then, by invoking the point editor, finds the nearest entered point and is given an opportunity to delete it, or to reposition the cursor. If he does move the cursor, the edited point is moved with it.

A more advanced feature, the line editor, permits the user to position the cursor and find the nearest line, rather than the nearest point. He is then given the opportunity to delete the line.

## Macros

A macro is a graphic symbol for a standard drawing element, or more generally, a sub-drawing of an item that appears frequently in the user's graphic work. Typically, macros may include pipe fittings, valves, structural shape cross-sections, weld symbols, and the like, as shown in Fig. 7.

A macro may be defined by the user by entering the graphic data to describe it in the usual way, as if it were a complete drawing in itself. In some systems, the macro may be defined as a three-dimensional object. The macro must also be given a set of reference points to fix its location when it is used on another drawing. For a three-dimensional macro, three reference points are needed to completely specify the location and orientation of the macro. Usually, a scale is also specified.

Predefined macro data is stored in a specific graphic support file, the macro library. Typically, a set of commonly used macro symbols is also represented in an auxiliary area of the digitizer or the display screen. From this area, which is in effect part of the menu space, the designer can select the desired macro at any time during the graphic session, by merely indicating the item with the cursor. Once a macro has been selected and rescaled to suit the drawing, the user simply specifies the location of the predefined reference points on the main drawing, as shown in Fig. 8.

Obviously, the user must study his graphic work in advance in order to

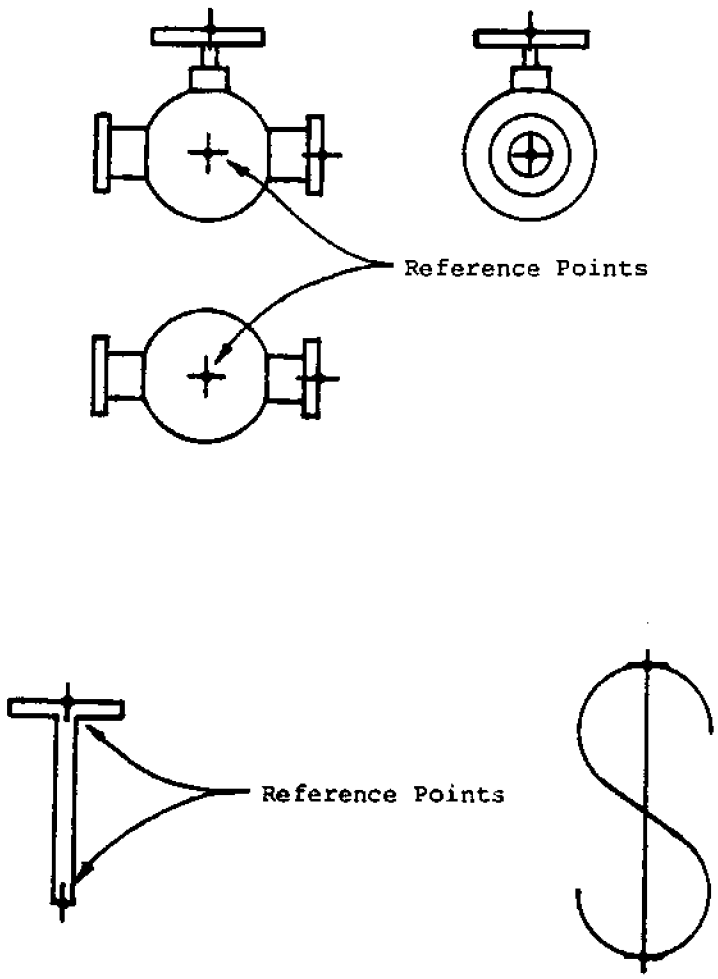



Figure 7. Typical macros. The valve assembly macro is shown as a simplified three-view drawing, permitting the macro to be entered in any orthogonal view. Advanced systems may also permit general rotation of the macro before insertion into the main drawing, although automatic hidden line removal is generally not available. The structural cross-section and weld symbol are simple two-dimensional macros.

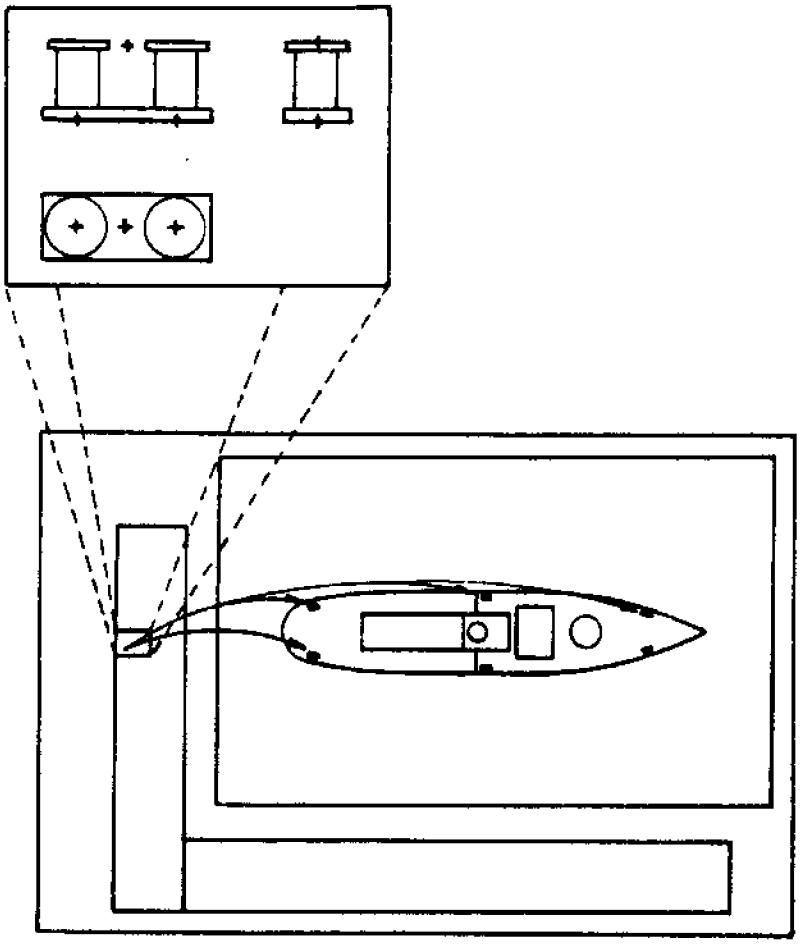


Figure 8. Installation of a macro (a double mooring bit) on a deck layout. Each location of the macro is achieved by fixing one of the macro reference-point locations on the drawing, and specifying a line segment to give the angular orientation of a second reference point.

decide which groups of graphic data can be conveniently put in the form of macros. The ability to define, store, and recall macros is a great advantage to the computer-aided design process, particularly in structural, piping, and other system design tasks.

## Perspective Drawings, Rotation, and Zoom

The perspective drawing, in lieu of the usual orthogonal projection, is occasionally an advantage in certain areas of design. With the present emphasis on modern production techniques in the shipyard, a great deal of engineering information is passed to production in the form of construction sketches, which may be more effective if they are perspective drawings. Graphic systems for producing such drawings now exist, although they have not yet been implemented on very small computers.

In constructing the perspective drawing, the designer first enters a three-dimensional description of the object or assembly in the usual way, as on a digitizer or screen. Subsequently, the system permits the user to specify both the position of the viewing point, and the position of the projection plane, in order to generate a perspective drawing of the object.

By moving the "eye" and the projection plane, the object can be given any apparent rotation and magnification (zoom). The computer performs the required coordinate transformations and displays the resulting drawing on the screen, from which it can be modified or saved for plotting. While the details of the rotation and zoom transformations are too complicated to discuss here, they are sufficiently general that the designer will be able to choose a view that is effective in conveying production information.

Two advanced features are desirable for perspective drawings generated by computer systems. Both, however, are relatively complicated and difficult to obtain. They are:

(1) Automatic hidden line removal.

(2) Automatic shading.

The automatic removal of hidden lines is mathematically fairly complex, while the provision of automatic shading requires substantial computer and plotting time, and will not show up on the usual DVST or pixel display screen, which do not incorporate brightness modulation.

For these reasons, such advanced techniques may not be available on microcomputers for some time. However, even the semi-automatic production of perspective drawings may be of some advantage in the creation of construction sketches, with hidden line removal and shading performed by hand after the computer has drawn out the initial perspective view.

## Computerized Annotation of Drawings

The entry of alphanumeric information into drawings is generally accomplished after all lines have been completely entered. The location of legends, witness lines, and dimension lines is usually first established by hand on a preliminary plot, so that the designer can decide how to lay out the information to best effect.

Finally, the actual lines, arrowheads, and alphanumerics are entered from a special annotation symbol menu, which is usually brought into the menu space just for the purposes of finishing off the drawing. The locations and angles for alphanumeric and symbol data are specified with the cursor, and the

desired "strings" of alphanumerics are constructed using the entries from the menu space. Alternatively, and perhaps more conveniently, the alphanumeric strings can be composed on the keyboard, before the entry locations are specified with the cursor.

Standard character sets for lettering title blocks and general annotation of drawings are furnished with the graphics system. The user may also have the option of changing the size (and sometimes the slant or proportions) of the characters to suit the scale and application, during annotation. For editing alphanumeric material in a drawing, a special graphic editor function, the symbol editor, may be provided.