A SYSTEM FOR THE SYNTHESIS OF MACHINE STRUCTURES

A SYSTEM FOR THE SYNTHESIS OF MACHINE STRUCTURES

by

David James Bonham, M.Eng.

A Thesis

Submitted to the Faculty of Graduate Studies

in Partial Fulfilment of the Requirements

for the Degree

Doctor of Philosophy

McMaster University

August, 1974

Doctor of Philosophy (1974)
(Mechanical Engineering)

McMaster University,
Hamilton, Ontario.

Title: A System for the Synthesis of Machine Structures

Author: David James [illegible], B.[illegible] (Queen's University, Kingston, Ontario.)
M.Eng. (McMaster University, Hamilton, Ontario.)

Supervisor: Professor J. N. Siddall

Number of Pages: xix, 179, Appendices, 112.

ABSTRACT

Throughout this thesis ~~the design~~ of machine structures has been approached as a system comprised of three inseparable but distinct sub-systems; (1) computer hardware, (2) software, and (3) mathematical model. This work has evaluated the role of each sub-section and has developed a strategy to incorporate the capabilities of each sub-system into a cohesive approach to automated structural design.

A governing criterion of this work has been the desirability of retaining the engineer in the design process and devising a system that effectively utilizes his decision making abilities while relieving him of the tedious tasks of data assemblage and analysis.

Part One, the initial stage of this work, has been the development of a remote intelligent terminal system that

has been designed to incorporate a selection of computer peripherals into a total system. This system can be fully utilized in a flexible manner by a user knowledgeable in FORTRAN with a working knowledge of the timesharing system being employed. Included in Part One is a description of the communication strategy developed. As well, a description is included of the generalized software applicable to the peripherals associated with the terminal and resident in the large scale computer.

In addition, Part One involves a description of the development of an operating system for the in-terminal computer as well as a library of programs that can be used with the system while it is operating in a stand-alone mode.

In Part Two of this work a software system has been developed that utilizes the terminal system and a large scale computer to demonstrate the application of this technology to a system for interactive design of machine structures. This software system is modular in approach, allowing the user to enter the system at any point or terminate the program at any point - all files are automatically appended and stored by the system. Thus, no information is lost for partial runs.

Part Three is a description of the sub-optimal algorithm used in the automated design section described in Part Two. Results of trial designs of a standardized milling machine structure (CIRP structure) are presented.

In Part Four the terminal system and the associated software developed has been applied to an industrial design problem in the glass bottle industry. This application is presented to demonstrate how this type of intelligent terminal can be used as the central element in a hierachical computer system for Computer-Aided-Manufacturing (CAM).

ACKNOWLEDGEMENTS

The author wishes to thank his supervisor, Professor J. N. Siddall, who gave his guidance and encouragement at each stage of this work, and who taught the author a great deal more than is described in this thesis.

The author also wishes to thank the other members of his supervisory committee, Dr. M. A. Dokainish, Dr. B. Latto and Dr. A. Smith for their helpful suggestions during the course of this work.

A special thanks is extended to Mr. R. Gillan who translated incomplete ideas into working hardware.

TABLE OF CONTENTS

GLOSSARY OF TERMS

Accumulator - A computer register where the results of an arithmetic operation are left at the end of the operation.

Acoustic Coupler - An inexpensive form of modem that employs an ordinary telephone head set.

Alphanumeric - In reference to a display device, it defines a display capable of writing alphabetic characters and numbers.

Applications Programme - This is the level of programming at which a system is developed that applies a general software system to a specific application.

APT - Automatically Programmed Tools, APT is a large computer program that reads a "part program" written in the language of APT and generates a file of coordinate information for the machine tool controller.

Assembler Language - This is a language of mnemonic representation of the octal (or binary) machine language. It is powerful as the actual machine language for the control of all processor functions. One mnemonic instruction generates one machine language instruction. The assembler language is manufacturer dependent.

Baud - Bits of information per second. A standard communications term to depict the frequency of character transmission between computer and terminal.

BCD Code - Binary-Coded-Decimal. Standard 6 bit code for character representation. Usually employed for magnetic tape or disc storage for data files.

Bit - Smallest segment of a computer word. A bit is a binary digit, 1 or 0.

Buffer Storage - An area of computer memory that is set aside for the temporary storage of information. Usually implies service when the recipient device demands more data.

Character Blocks - A group of binary bits required to code a character for communication.

Clipping - In graphics this is the display of a portion of the display file normally at full scale on the display device and the removal of that portion of the display that lies outside the display device boundaries.

CNC - Computer Numerical Control. The use of a computer to perform the functions of interpolation through software.

Core - Ferromagnetic material that can be polarized to produce two binary states.

It is the most common form of computer memory employed at the present time.

CRT - Cathode Ray Tube. This is a display device that uses an electron beam to energize phosphors on a translucent screen.

D/A Converters - Digital to analogue converters translate the digital representation of a value into a scaled analogue voltage of equivalent value.

Data Base - This is a data file that contains the information defining a system. (coordinate data for structural elements, etc.).

Data Concentrator - This is a computer that is used to "front end" a timesharing system. This concentrator communicates with the main-frame at high data rates and to a series of terminals at lower rates. It does the necessary message formatting. Results in a more efficient employment of the main-frame.

Data File - An identifiable block of data stored in the computer memory, disc file or magnetic tape.

Data Management - Those software functions of data retrieval, modification, creation and storage.

Data Retrieval - Those software functions involved in the "selective" recall of data from a data base.

Device Handler - A software program that formats a data file for a particular device or peripheral.

Disk - A mass storage device used in association with a computer. Resembles a phonograph record - except that the data is recorded magnetically. Allows random access to stored data and is an extremely efficient storage medium with short access time.

Display File - This is a file of information for a display device. It contains the coordinate data and defines the sequence and mode of pen or beam movement for the generation of a graphics display.

Fortran - This is a high level computer programming language that has most closely achieved the status of being the "universal" scientific programming language. It is well defined by standards and is highly transportable between computers of different manufacture.

Function Key - At a terminal a function key is a button that when energized produces a terminal action. This action can be controlled by either software or hardware.

Hard Copy - A soft copy of a display is the creation of the display on a CRT. A hard copy by contrast is the creation of the display on a medium such as paper or film.

High Level Language - These are computer programming languages that allow a programmer to converse with the computer in a more natural manner than machine language coding. These programs are compiled into machine language instructions. One high level instruction may produce a large number of machine language instructions. These languages have the least capability in the control of the processors functions but are usually computer independent.

Host Computer - This is the on-line computer that contains the logic to support terminals in a timeshared mode.

Host Processor - Synonymous with host computer.

Intelligent Terminal - This is a computer terminal that is itself programmable (i.e., the central element in the terminal is itself a computer).

Interactive Program - This is a program that operates in a timeshared environment and solicits data in real time from an on-line user.

Interface - This is the necessary hardware to allow a computer peripheral to be controlled by the computer.

Library Program - A program that is general in nature and is accessible to the user at the time of program loading.

Macros - A general block of coding used repeatedly with different argument values. Referenced by the macro name.

Mainframe - A large scale computer.

Micro-computer - Synonymous with computer on a chip.

Mini-computer - There is no common agreement as to what constitutes a mini-computer generally: 16 bit word, 4-16 k memory with optional disc, tape drive and Teletype.

Mnemonic - The representation to the assembler of a machine language instruction by alphabetic characters:
example for a PDP8-PAL III instruction:
CLA - clear the accumulator, in machine language would be: 7200_8.

Modem - Modulates and Demodulates input serial pulse trains on a communication line into binary representations of characters.

Overlay - The efficient use of a limited computer core can be achieved by "overlaying" a portion of core no longer in use. In FORTRAN this is specified by an OVERLAY declarative.

Positive Logic - The electronic logic is controlled by positive voltages.

Post Processor - In APT the cutter location file (which is machine tool independent) is post processed to add in the necessary codes to control a specific machine tool.

Process Control Computer - This is a (mini) computer that monitors the characteristics of a production process and adjusts the input conditions according to process requirements.

Processor - Synonymous with computer.

Real Time - All activities are being conducted on-line as opposed to prepared input and deferred output.

Scope - Operating system for a CONTROL DATA CORP computer. At McMaster University the CDC 6400 uses the SCOPE 3.4 operating system. Supervisory Control of Program Execution, also oscilloscope.

Shaft Encoder - Produces digital output pulses as the input shaft is rotated. Pulses when counted give a digital representation of the analogue rotational input.

Smart Terminal - Synonymous with intelligent terminal.

Software - The computer programs used to control the functions of the computer (hardware).

Softwired - Softwired control is the use of a computer to perform functions such as interpolation in place of hardware (hardwired) interpolation.

Sync Pulse - A voltage pulse that can be used to time or initiate some additional hardware (software) activity.

System Abort - An abnormal termination of a job processing sequence.

Systems Programmer - A person familiar with the programming of the computer operating system.

Terminal - A device that serves as the interface between the user and the time-shared computer.

Windowing - The "framing" of a portion of a display. This window can then be scaled up or down and translated about the display area.

Word - A fixed number of memory bits define a word. Usually a block of bits that can be transferred into and out of memory intact as one unit.

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1. General

Since the introduction of large scale computing systems to engineering applications, there has been a considerable interest in the development of algorithms for the optimal design of structures. In general, the approach has been to establish a criterion for optimality, and develop an objective function that expresses the influence of a set of structural design variables upon that criterion. All or part of these design variables are generally subject to a set of constraints. There has been a wealth of methods developed to deal with some aspect of this problem.

In spite of the activity in this area, there are remarkably few recorded cases in the literature which chronicle a successful application of these automated procedures to real structural problems. The difficulties encountered by the engineer in the application of automated techniques center about the problem of adequately defining all aspects of optimality in terms of the restricted number of design variables computationally permissible in even the largest computer systems.

This, coupled with the range of techniques each of which has a domain of applicability, often produces a condition of sufficient complexity to warrant a system that allows the engineer to exercise his judgement in the development of an initial configuration by the analysis of several trial designs. At the completion of this stage, the utilization of applicable automated procedures becomes more feasible as a result of the reduction in the number of design variables to a computationally manageable level.

The broadest definition of the requirement of an engineering structure can be stated as those elements required for the efficient transmission of a load vector to a foundation. This definition in the context of machine structures has to be broadened to incorporate the additional considerations of dimensional constraints, aesthetic constraints, material constraints, frequency response constraints and thermal deformation constraints. The imposition of the first class of constraints within the limitations of batch processing is an arduous task. There are several examples quoted in the literature that illustrate the computational complexity that occurs when geometrical variables are introduced into the optimization procedure. Reference (1)[1] cites an example where the batch processing analysis of a simple flange of a pressure vessel was a task of several weeks' duration. Figures 1.1 and 1.2 are photographs of milling machines. It is clear that the inclusion of geometrical variables and the establishment of constraints on these variables, the design of a basic structure for a machine to perform a general milling operation is a difficult task. It is obvious that a system must be

1 Numbers enclosed in parentheses indicate the number of a reference listed at the end of this thesis.

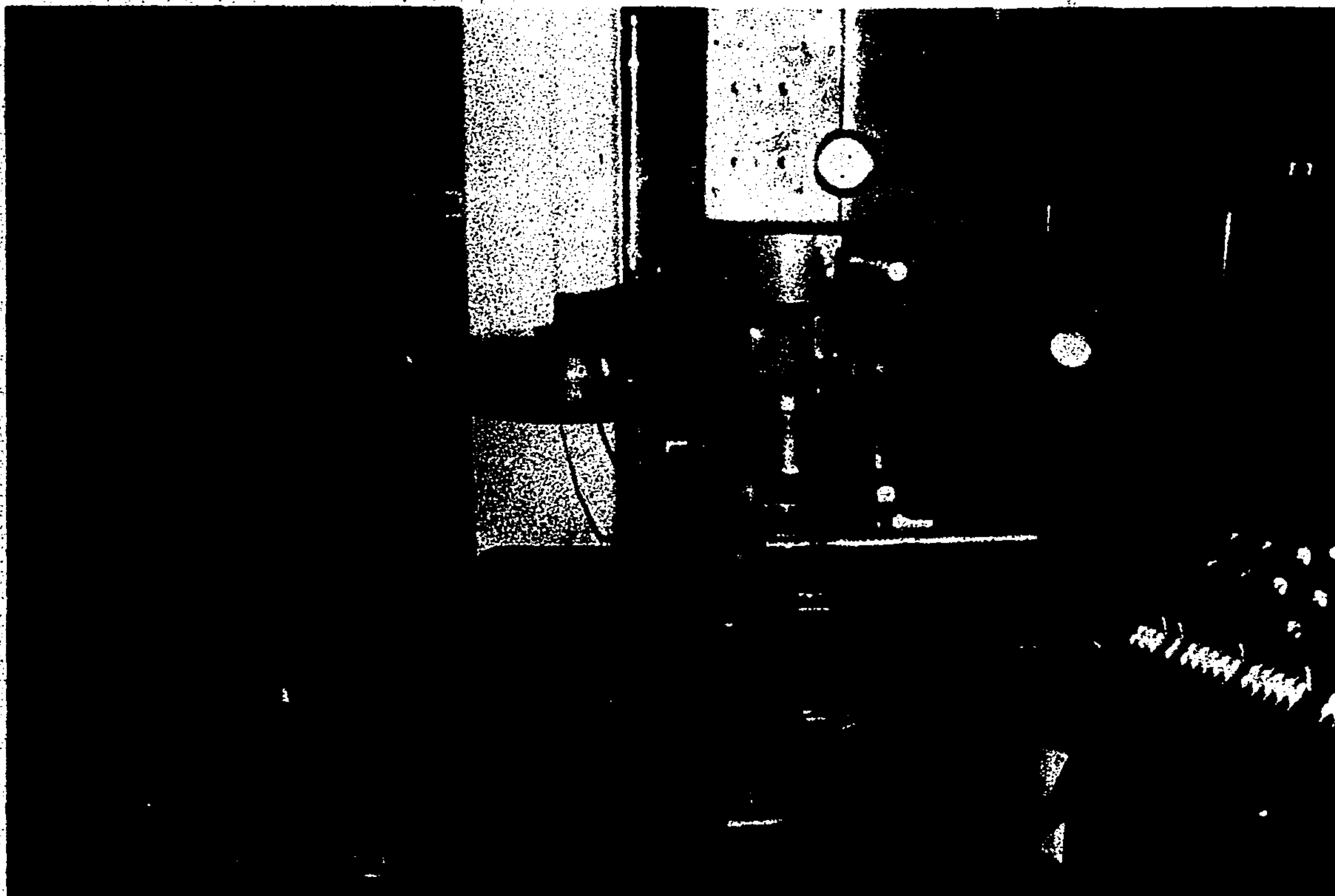


Figure 1.1 NC Controlled MOOG HYDRAPOINT Milling Machine.

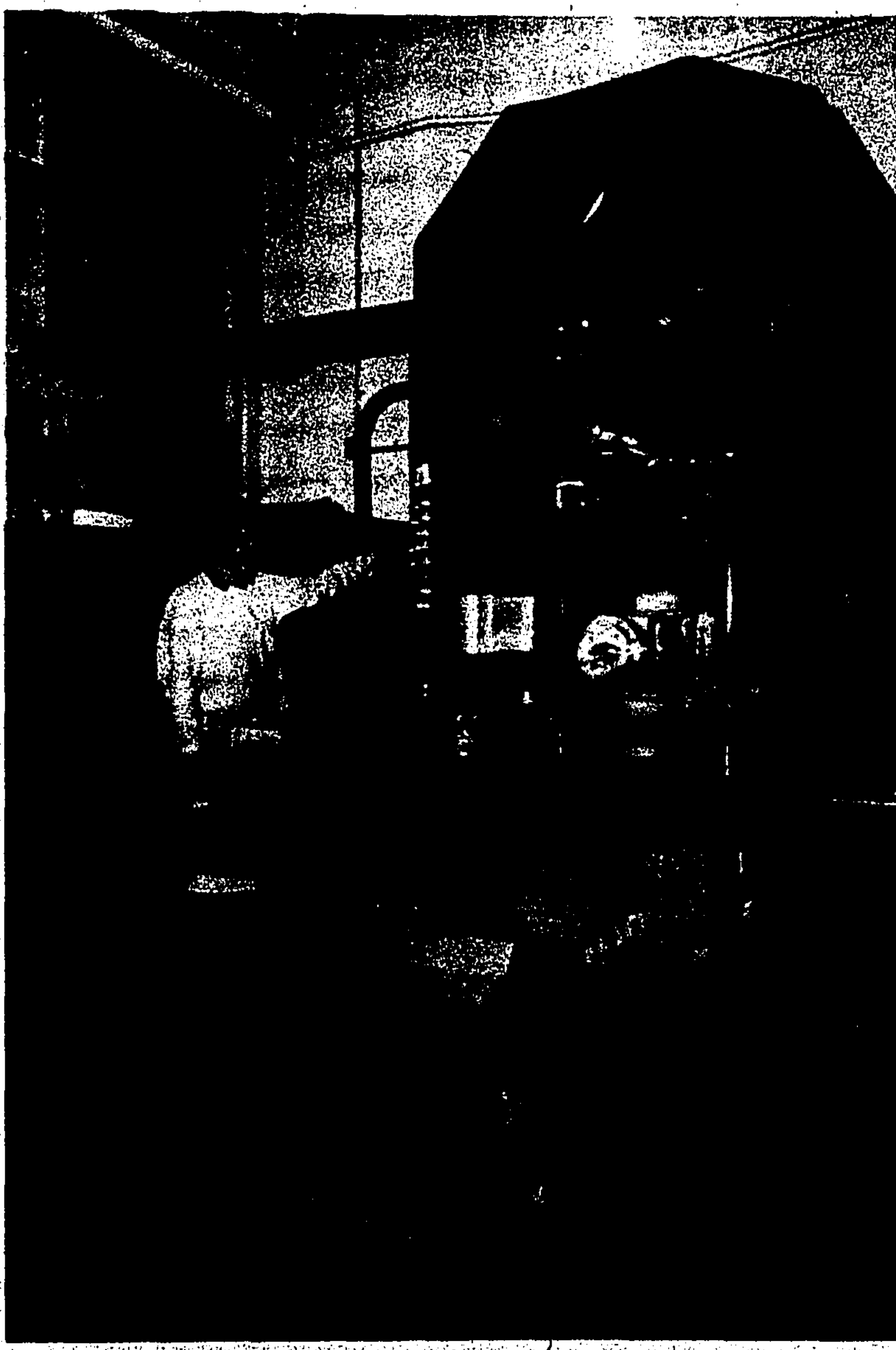


Figure 1.2 SUNDSTRAND OMNIMILL Milling Machine.

capable of efficiently handling the data assemblage of information to describe the structure, display the results in an easily interpretable form and allow rapid modification of design parameters. A system encompassing these capabilities allows the designer to develop a coarse approximation to the requirements of his problem and through several iterations direct the refinement of his design through increments that are not computationally prohibitive.

It is the premise of this work that, in part, the application of a balanced approach between software and design oriented hardware in an interactive computer environment offers a realistic solution to these problems. This thesis presents a new approach, which rather than grafting conventional analysis onto the computer, fully integrates the user and the computer based on the requirements of the design process.

The development of a system to achieve these goals has been divided into three separate sections.

1.) A suitable hardware system was developed that fulfills the stated problem requirements and meets the additional requirements of low cost and general applicability to a broad range of engineering design problems.

2.) An interactive software system was developed that incorporates the generalized software developed for the terminal system. This software structure has been designed to utilize fully both the input-output capabilities of the terminal system as well as the file management capabilities of the host timesharing system.

3.) A sub-optimal algorithm has been employed and tested on a standardized milling machine structure to demonstrate the feasibility of the approach developed in Parts One and Two.

In addition to the system for automated structural synthesis, the general applicability of both the software and hardware systems has been demonstrated in the development of a system to automate bottle mold designs for a real problem that exists in the glass manufacturing industry. In this application, the feasibility has been investigated of using this type of terminal system as the intermadiate link in a hierarchical computer system for the dual roles of computer-aided-design and computer-aided-manufacturing (CAD/CAM).

1.2. EDIT System

Although timesharing systems supported by large scale computers have been available for at least a decade, there has been a significant lag in the application of time sharing to large scale engineering design systems. Fenves (2), after describing the advent of timesharing in the middle 1960's as the second computer revolution, attributed its lack of adoption in civil engineering to a natural lag inherent to the profession and to a lack of standardization amongst systems. Although this lack of standardization is a valid criticism of the state of the industry, it fails to explain completely the lack of applications in design engineering.

More fundamental to the problem of timesharing has been the input/output devices available to the user as the interface between the man and the computer. Until recently these terminals have typically been low speed teletypewriters or alphanumeric CRT terminals. Inherent to the nature of structural design problems is the need to input large amounts of data to describe a structure and to output equally large amounts of data for analysis of the performance of a structure. The output problem is further compounded by the need to analyze the structural performance over a range of values of each design variable.

The introduction of low cost graphics terminals has, in part, relieved this situation. However it has yet to deal adequately with the total requirements of the engineer designer. Chapter 2, Section 2.1 of this thesis describes the activity in the literature that is being conducted in the development of systems to bridge the gap between the designer's requirements of a computer system as described in the previous section and the processing power of a large scale computer.

Chapter 3 describes, in detail, a terminal system developed in the course of this project. The design criteria adhered to in this development can be stated as follows:

1.) Low cost - the total terminal cost must be within the budgeting limitations of an engineering design office.

2.) Ease of operation - the terminals operation must be able to be handled by an analyst with a knowledge of FORTRAN and a minimal familiarity with the time-sharing facility being used.

3.) Local collection of data - the terminal must have the capability to assemble data while operating in a stand-alone mode.

4.) Data editing - the user must be able to edit data locally at the terminal without the need of the large scale computer system.

5.) Verification of input data - the terminal must be able to employ a graphics display to verify the assembled data file.

6.) Data storage and remote access - the facility to store off-line data files and subsequently direct access to these files via directives generated by the host computer's software system.

7.) Off-line review of graphically presented aspects of a design.

8.) Local regeneration of repeated display functions.

9.) Analogue to digital conversion for the reduction of graphical data or non-dimensional drawings to a format suitable for processing by the host computer.

The complete design of this system, the EDIT (Engineering Design Intelligent Terminal) system, included the appropriation of tasks between the system processors; the development of the in-terminal processor operating system; the development of a communication strategy between processors; and finally the generation of a software system that resides in the host computer. This software is required to allow a remote user to employ the full capabilities of the terminal's peripherals. In addition, the development of this system included the design of a number of hardware interfaces.

The terminal system developed in this work has been designed around the needs of engineering design as it relates to the subject areas of machine structure design and computer-aided-manufacturing. It has shown that the non-intelligent terminal is inadequate to serve completely these needs while at the same time the computer "over-kill" of the satellite systems marketed commercially and developed elsewhere do not enhance the process sufficiently to warrant their additional cost. In fact, some very expensive systems have been carefully developed to serve a limited application and are much less flexible than the system developed here.

1.3. CONSTRUCT - A Software System for the Synthesis of Machine Structures [9]

Figure 1.3 is a diagramatic presentation of the relationship that exists between the separate software packages that are employed in a total system for the synthesis of machine structures. The section labelled CONSTRUCT incorporates the logic of the design process as it pertains directly to the specific task of structural design. The section labelled Terminal Support Software (TSS) includes the graphics package, the data encoder/decoder routines, and the device handler programs for selecting the appropriate terminal peripheral. The third system is housed in the terminal processor.

Chapter 2, Section 2.3 reviews the literature as it pertains to the development of systems for the design of generalized structures. Chapter 4 is a detailed description of the system that has been developed in the course of this work.

In general, the approach followed in this work has been to segment the total process into eight related sections each of which is dependent upon the data generated by lower order sections but which can function independently within a single computer run. Information is exchanged between sections via a remote storage device. The appendage of the appropriate files as well as the automatic retention of all generated files is handled by a system file control macro. This provision allows the user to enter the design process at any stage and terminate at any stage without loss of information.

1.4. BOTTLE - A System to Automate Bottle Mold Design

Although this work has principally been directed toward the creation of a system for the synthesis of machine structures, large sections of the work are general in nature and are thus applicable to a range of applications. This section, described in detail in Chapter 5 of this thesis, demonstrates the application of the EDIT system as the central element in a hierarchical distributed computing system for computer-aided-manufacturing.

There is considerable similarity between the needs of a system to automate numerical control part programming as it relates to computer-aided-manufacturing and a system for the computer-aided-design of engineering systems. Both areas, in general, require the input of large data bases, benefit from graphics presentation of the results, and require a large scale computer for processing the data.

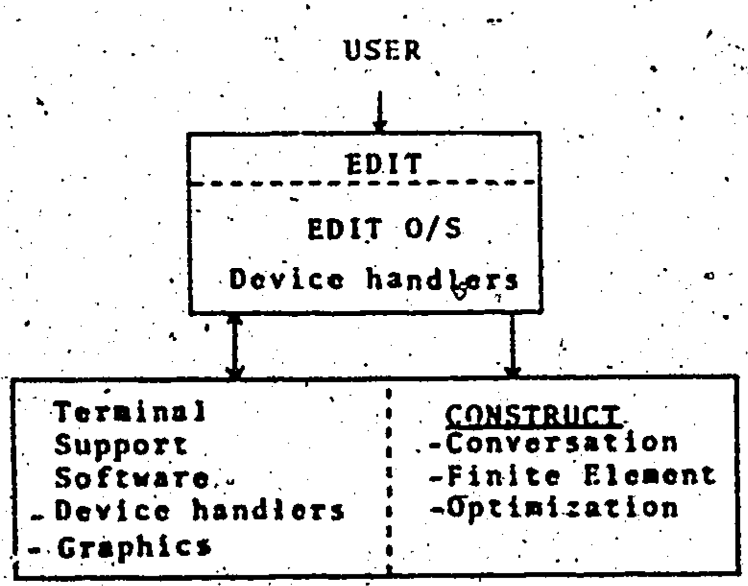


Figure 1.3

CHAPTER 2

STATE OF THE ART

2.0 General

This chapter has been subdivided into four sections in accordance with the separate sections of the thesis. The first section relates the recorded activity in the development of various computer-aided-design hardware systems. The second section then deals with the topic of graphics software development. The third section surveys the existing systems approaches to the computer-aided-design of structures. The fourth and final section of this chapter is concerned with the application of this technology to the area of computer-aided-manufacturing.

2.1 Computer-Aided-Design Hardware Systems

In a survey paper on the developing role of computer communications and its effect on society, Fano (3) concludes that during the last decade one of the most difficult lessons that computer specialists have discovered is that "it is inappropriate to design or evaluate computer equipment out of the context of the software that provides the interface to its user's". He further suggests that the reason is simple: "the equipment characteristics limit in a major way the interface characteristics that can be obtained".

With the rapid rise in the development of time-sharing systems, with their potential to offer the full processing power of large scale computers to relatively remote users, it would seem congruent that, over the past few years, there has been a considerable effort extended to provide a variety of equipment characteristics within the terminal system itself. This would seem to be the most suitable location to translate processing power into a user's individual requirements. As a logical corollary of the above observation it would be expected that a proliferation of specialized terminals dedicated to specific assignments would develop. This corollary has not, for the most part, been validated. There are a variety of reasons why this has not occurred, amongst which the principal one has been the prohibitive cost involved. Although the introduction of MSI and LSI semiconductor logic has produced a dramatic reduction in the capital investment required for computer hardware, the cost of custom built systems, which require development of both hardware and software, tends to be beyond the reach of all but the largest corporations. The introduction and development of intelligent terminals - with the flexibility to be directed via software to perform a multiplicity of tasks - has to some degree provided a feasible solution for the smaller user. This approach is not however, without pitfalls - some of which will be elaborated in more detail in subsequent sections of this thesis.

Hobbs (4) in a review of terminals, divided the terminals essentially into six categories based on both hardware capabilities and application requirements. These categories are:

1.) keyboard/printer terminals;

2.) CRT terminals;

3.) remote-batch terminals;

4.) real-time data-acquisition and control terminals;

5.) transaction and point-of-sale terminals;

6) smart terminals.

It is not the intention of this section to present an elaborate comparison amongst each of these terminal types. However, a brief description of the major differences follows.

Terminals of the first type, the keyboard/printer, although slow, have the decided advantage of producing a hard copy of the information both inputted to and received from the timesharing system. As Hobbs indicates, there are hundreds of thousands of these terminals in use. This class of terminal ranges in price from less than 1,000 dollars to approximately 6,000 dollars.

The CRT terminals are further subdivided into two classes: (a) alphanumeric terminals with sufficient hardware logic to present characters on the CRT and, (b) graphic terminals which can produce characters and draw lines. The alphanumeric terminal has the advantage, when compared to the electromechanical teletypewriters, of operating over a broader speed range, and the advantage of operating quietly.

A disadvantage involves the lack of hard copy without an additional device. These terminals range in price from 2,000 dollars to 15,000 dollars.

In addition, there are multistation systems that function through data concentrators. This approach has the advantage that the logic required by individual terminals is reduced while communication with the large scale system is enhanced.

Graphics CRT's can be further subdivided into two major divisions: These are: (a) refresh terminals and (b) storage terminals. The first type of display has a local buffer (semiconductor memory) that is used to continuously regenerate the display on the CRT. This format has the advantage that it is possible to produce some dynamic display features by modifying the buffer store between or during refresh cycles. This necessarily requires high speed communication systems. The disadvantage of these displays is the limited buffer store which can restrict the display density and the flicker that is produced when the refresh cycle rate is not adequate to support the number of displayed vectors.

The storage terminal enjoys the advantages of having almost unlimited display density capacity. It has the disadvantage that it is necessary to erase the entire display to modify a portion of it. In the past few months Tektronix (5) has released a new large screen terminal that has both storage and refresh operating modes. The refresh mode is supported by decreasing the writing beam intensity below the level required to illuminate the phosphors on the face of the CRT.

These graphics terminals range in price from 5,000 dollars up to 25,000 dollars.

In addition to the conventional CRT terminal there has been development work done on plasma panel displays (6), magnetooptic displays and light-emitting diode displays. As Hobbs notes, these displays have not had a significant impact on the market.

Remote batch terminals are essentially an assemblage of standard computer input/output peripherals such as card readers, line printers, card punches and magnetic tape units. These terminals provide full batch processing capabilities at an alternate input station to the central computing facility. The cost of these terminals ranges between 7,500 dollars and 170,000 dollars.

Real-time data acquisition and control terminals are installations that collect data from instrumentation and control systems and generate the necessary signals for controlling the process (7), (8), (9), (10). This type of terminal is one in which custom terminal design is unavoidable. Further discussion about these systems is included in a subsequent section dealing with a distributed computing/computer-aided-manufacturing system.

Point-of-Sale and Transaction terminals are those systems which have been developed for marketing systems to record sales, debit inventory, book hotels or airline reservations, etc.

Smart terminals, or intelligent terminals (figure 2.1) incorporate a small computer into the terminal itself.

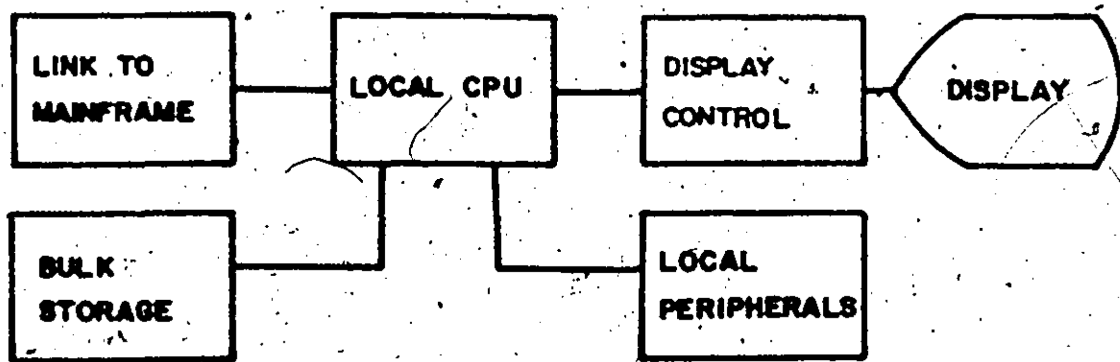


Figure 2.1 General configuration of an intelligent terminal.

In addition, to qualify as an intelligent terminal the computing capability of the mini-computer must be available to the user in a way that permits him to program it to perform part of his unique application.

The characteristics that pertain to this class of terminal are:

1.) self-contained storage;

2.) user interaction - with either the terminal or the central computer;

3.) stored program;

4.) part of processing accomplished in the terminal;

5.) on-line via communications line with large central computer and data base;

6.) human (problem) oriented input;

7.) human (problem) oriented output.

Associated with the terminal, the central computing facility must have;

1.) mass storage;

2.) peripheral equipment suitable for handling those tasks not readily handled by the terminal;

d.) computing power.

Almost without exception, any discussion in the literature of intelligent terminals involves the question of how much intelligence is sufficient intelligence [(4), (11) (12)]. There is no common agreement on the answer to this. The failure to address the problem in the light of the expected requirements of an individual application, can result in what could best be termed computer "over-kill". This phenomenon is most likely to occur when the solution is developed independently of the problem or independently of those most knowledgeable in the area of the proposed application (13).

Hobbs, in his concluding remarks on the subject of smart terminals, subdivides the expected development of such systems into two categories based on expected cost.

"With the current trends in technology, it is reasonable to expect a smart terminal selling for 7,500 dollars to 35,000 dollars in the future to include:

1.) a 4,000-to-16,000 word micro- (or mini-mini) computer;

2.) a 300 to 1,200 baud modem;

3.) a keyboard;

4.) a character serial printer;

5.) a magnetic tape cassette;

6.) an optional CRT;

7.) an optional light pen."

He predicts a second further area of development in the creation of smart terminals in the 50,000 to 200,000 dollar price range. "The typical installation of this type would include:

1.) a 32,000 to 64,000 word mini-computer;

2.) a 9,600 baud modem;

3.) a keyboard;

4.) a character serial printer or other hard copy device;

5.) magnetic tape cassettes;

6.) a graphic CRT display;

7.) a light pen;

8.) optional function keys;

9.) optional discs;

10.) optional compatible magnetic tape units;

11.) an optional line printer."

This last division is directly comparable to a stand-alone computer system. For the purposes of this thesis, the terminals in the first division are termed intelligent terminals, and those in the second division are termed intelligent satellites.

Van Dam et al. (11) define this distinction as follows: "that if the terminal contains a mini, micro or microprogrammable computer which runs a standard program to service the terminal, and not arbitrary user-loaded programs, the terminal has a fixed function and is still just an intelligent terminal. Only when the device contains a general-purpose computer which is easily accessible to the ordinary user for any purpose and program of his choice do we promote the terminal to an intelligent satellite".

This definition is less precise than the definition presented on the basis of Hobbs' separation of classes.

A popular concept, which has emerged over the last few years, is the application of distributed computing systems. These are systems of computers which are linked in a hierarchical fashion such that each link contains sufficient logic (hardware and software) to perform those functions that are most efficiently allocated to it while at the same time, it directs all other processing functions to the most appropriate device. The problem of efficient division of labour is a difficulty which has yet to be resolved. The proposed strategies range from stand-alone systems, that utilize a large scale computer as a peripheral, to systems that are largely dependent upon processing power of the full scale computer, and utilize the in-terminal processor to merge the local peripherals (11), (14), (15), (16). As well, the terminal's processor is used to edit, store and display data files (17), (18), (19). One area of common agreement is the number of advantages offered by using the basic concepts of local processing power. These advantages have been enumerated succinctly by Van Dam.

1.) Local data assemblage and editing.

2.) A reduction in the scope of the areas of conflict that invariably arise between local users of the range of services a central facility should provide.

3.) Conservation of effort that results from the verification of assembled data prior to transmission to the central site for processing.

4.) Conservation of resources that result from the reduction in the number of user interactions with the mainframe.

5.) The ability to perform a wide range of tasks locally in the areas of data conversion and file manipulation.

6.) The ability to emulate more primitive systems.

7.) Flexibility in custom programming the terminal.

Equally, amongst all authors, there is a common acceptance of the areas of difficulty that are experienced in the development of a distributed computing system.

1.) Non-standard terminal operating systems. The concept applied to the terminal logic are transportable to the development of systems with similar capability. However, the complete adoption of one system's software system and application to another is complicated by a myriad of interwoven problems.

2.) The problem of interfacing different manufacturers hardware into a cohesive system. As Van Dam notes, this difficulty leads to the 'our system works, it must be their system' syndrome. This problem is compounded by manufacturers who refuse to service their product if it has been 'tampered with' in the installation.

3.) Changing technology. The development of a useable system necessarily proceeds first with the development of the hardware and then secondly, with the development of the software support. Without the infusion of large amounts of capital and manpower the inherent time lag created can produce a working system that is employing antiquated hardware that has cost as much as ten times its current replacement value.

4.) Communication systems. There is a substantial cost increment that occurs for communication rates in excess of 300 baud (acoustic coupler) and again for rates above 1,200 baud due to leased lines and expensive modems. These cost considerations, for the small facility, have a direct effect on the operating system design.

5.) Software support. Within a single system there are three required levels of support.

a) Mainframe service routines - these constitute the programs that provide the interface between the user's application program and the terminal device handling routines that direct the communications channel for data encoding. These are most likely to be written in one of the higher level languages and can be interpreted by a systems programmer.

b) Intra-device routines - these embody the mainframe's operating systems communication handlers that structure and format the output data blocks for transmission between the central computer and the terminal system. These programs are most often written in the individual computer systems' assembler language. The modification (minor or otherwise) of these programs, to avoid what is often a complicated task at the terminal, requires the understanding of a sympathetic mainframe manager and a skilled systems programmer. This combination is rare even within the university environment.

c) Terminal operating system - The support required for the maintenance and modification of the terminal's operating system is a function of the terminal's size. This consideration is dealt with in greater detail below in a discussion of the concept of critical size. For the satellite system, which can support higher level languages (FORTRAN, PL/1), the modification of the operating system is less difficult than the maintenance of the operating system for the smaller terminal system. With the smaller terminal processor, it is usually necessary to maximize the efficiency of the operating system by writing it in an appended form of the manufacturer's assembler language. Unless this system is well documented, and unless it has a modular construction, the appendage of additional capabilities can be a difficult task.

Thus, it becomes clear that the general maintenance of one system requires familiarity with three levels of programming for its full support and development. This is a problem that can partly be relieved by the 'black box' approach to the design of the terminal's operating system.

Van Dam introduces the concept of critical intelligence, which is a somewhat imprecise definition of the level of processing power that the terminal must possess. Below this critical level of processing power and, in addition, "if the satellite does not have reasonably fast secondary storage, sufficient local memory, and a powerful instruction set, it simply may not be able to do enough processing fast enough to make the division of labour worthwhile".

In contrast with the concept of critical intelligence is the concept of critical size (or perhaps critical cost) which can be stated simply as: There is a level of intelligence in a terminal which, if exceeded, contributes insufficient returns to warrant the substantial increase in cost, when judged by the needs of the intended range of applications. This cost constraint imposed by the decreasing returns experienced in the enhancement of the application of these systems to the engineering design process weakens Van Dam's conclusion that the satellite system (as he developed) is the optimal approach when considering a range of systems between simple display terminals and full stand-alone graphics processors.

There are several strategies related to the division of processing power in a multiprocessor graphics system. One such strategy is the 'black box' approach. Under this mode of operation the system is designed such that all hardware details of the terminal are hidden to the user. The terminal's operating system performs the tasks of display file management, peripheral servicing, and data retrieval and management. These services are used in conjunction with software routines resident in a library of subroutines in the large scale computer. This approach allows the user the full flexibility of a terminal that can be configured to his applications by the simple directive of a higher level language call to a supplied program. Van Dam is critical of this approach on the basis that it leads to under-utilization of the terminal's capabilities. The approach proposed by Van Dam requires the user to be cognitive of both the terminal's operating system as well as that of the host processor. This approach has the advantage that it provides a dynamic appropriation of processing power between the system processors. It has the very serious limitation of requiring a degree of systems familiarity that possibly will exceed the application oriented user's threshold level of acceptance. Experience with various forms of computing systems at McMaster University has indicated that there is a level of complexity above which the potential user will retreat to the safe harbours of batch processing. This situation leaves as the terminal's prinicpal users, those dedicated few who have been instrumental in the development of the system.

The EDIT system developed in this project utilizes a balanced approach between sophisticated and unsophisticated user (in a computer sense) by reserving a block of core that is coincident with and adjacent to the buffer storage area. The operating system's pointer can be directed to the first address of this block by a call to a higher level routine in the host system. This block of terminal core can be overlaid by accessing the terminal library program. The user then has the ability to program the terminal to perform tasks unique to his particular application while still functioning within and independent of both the terminal's and the host's operating system. The final instructions in the user's program is a simple return jump to a specified address in another program block in the system. This feature is described in more detail in Chapter 3 of this thesis.

There is a definite need for the development of generalized hardware systems which are flexible enough to perform a variety of tasks related to computer-aided-design and computer-aided-manufacturing. These systems must be cognitive of the design process and, as such, complement that process in a manner that is easily employed by the engineer who is reluctant to become a computer specialist. This task must be accomplished within a cost that is realistic for a small to medium sized engineering design office.

The complete design of an intelligent terminal consists of the design of two software systems, the selection of the peripherals attached to the terminal and the communication strategy between computers. There exists an infinite number of possible configurations. To a large degree the literature deals with specific devices in a specific environment.

Thus, rather than present a general review of the literature on possible peripheral configurations, this thesis will describe a particular configuration, with a descriptive comparison of possible alternative peripheral configurations in Chapter 3.

2.2 Graphics Software Development

The development of a successful system for computer-aided-design is comprised of two separate functions; the creation of a suitable hardware system that has the physical capabilities required to aid the design process, and the development of a software system to interface the user with the hardware system. At the time that this project was initiated there was in existence no generalized approach to the structure of a graphics software system that was applicable to a wide range of display devices and yet concurrently was cognitive of the capabilities of a specific device. Throughout the course of this work, there have been a number of systems that have been developed and described in the literature, (22), (23), (24), (25). The nature of all programming languages (graphics and otherwise) is such that a truly comparative study of the relative merits of each unique approach is not realistic. This results from an unlimited range of application possibilities on one side coupled with a restricted operating system compatibility on the other. For this reason, this section of the literature review will be restricted to a general description of the graphics systems.

developed as well as a discussion on the minimum capability that a system must possess.

Newman and Sproull (20), (21) have produced a summary of the requirements of a generalized graphics system as well as a description of a strategy they propose as being general enough in principle to be applicable to a wide range of display devices and to fit within most computer operating systems. They identify ten distinct stages in a graphics system. An interpretation of these stages follows:

1.) Input devices. The hardware devices capable of generating information to be transmitted to the operating software. These devices are keyboards, light pens, digitizers, tablets, etc.

2.) Input handlers. These consist of the functional routines that process the system interrupts, control the interrupt priority, service the interrupting device and provide some form of received information to the input routines.

3.) Input routines. The programs that retrieve information from the input handlers and convert that data into the application data base.

4.) Application data base. A data file, generated in part by the input devices and in part by the applications program.

5.) Applications program. This is the program that contains the logic of the synthesis or analysis system to which the graphics system is applied.

6.) Output routines. These programs describe the graphics display from data generated by the applications program, and format non-graphical information for the appropriate output device handler.

7.) Transformation and clipping routines. Programs that are employed to manipulate the display file prior to ordering the presentation of the graphics display at the terminal.

8.) Display file. A generated file of display information coded in a format suitable for a particular display device.

9.) Display controller. A hardware (and software) system that interprets the display file and performs a corresponding hardware action.

10.) Display or output device. This is the hardware device on which the graphics function is employed. This could be a plotter or any form of CRT.

The terminal hardware directly affects the structure of the graphics system in each of the divisions listed above. The choice of display format, storage tube or refresh CRT, affects the form of sections (1), (2), (6), (7) and (8). The inclusion of local intelligence and the degree of intelligence affects the physical location of sections (3), (4), (5), (6), (7) and (8).

The failure to standardize any aspect of the above, has resulted in the generation of a large quantity of redundant effort. The piecemeal acquisition of display devices within an organization has often led to the development of an equal number of software systems. This bewildering array of capabilities has given graphics a bad reputation. One unsatisfactory and primitive solution to this problem has been the creation of a "front end" for any subset of graphics systems such that there is the appearance of device-independence achieved.

The GRAPPLE system developed by Bell Northern Research (22) is a system that has attacked this problem directly. The documented applications for this system attest to its success.

The first decision point reached in the development of a system is the choice of language. This decision has been made in the past in two distinct directions; the creation of an entirely new language for graphics with its own compiler, and the development of a package of graphics functions that are supported within the structure of an existing language. The first choice has the advantage of efficiency for the generation of displays, while the second has the advantage of familiarity for the applications programmer.

The second major decision point is the selection of a set of graphics functions for inclusion in the system. These functions are essentially of two classes, primitives and transformations. The primitive functions deal with the drawing of lines, absolutely or incrementally (vector), and the display of strings of text. The transformation functions include the functions of scaling, rotation (with and without perspective), clipping, and windowing.

In addition to the primitives, there is a higher level of routines that are graphics functions. These include such programs as contouring, graph plotting, arcs, circles, grids, etc. As a subset of these graphic functions there should be a display coded file of often repeated display forms.

The general relationships between the elements of a graphics system are shown in figure 2.2.

In addition to display device independence there is the problem of machine and operating system independence. The normal process for circumventing this problem is the encoding of the entire system in FORTRAN or a similar higher level language. This has the advantage of intra-machine transportability; but, in addition, has the disadvantage of being inefficient. There are some commercial display systems that require operational features of the terminal driving software that can only be implemented in the system's assembler level language.

The detail of the solution to each of the above problem areas is to some degree affected by the hardware system. An integrated hardware/software system that is an inexpensive and functional solution for the application of computer graphics to mechanical engineering design has been developed in the course of this work. This system is described in detail in Chapter 3 of this thesis.

There are two additional dimensions to a graphics

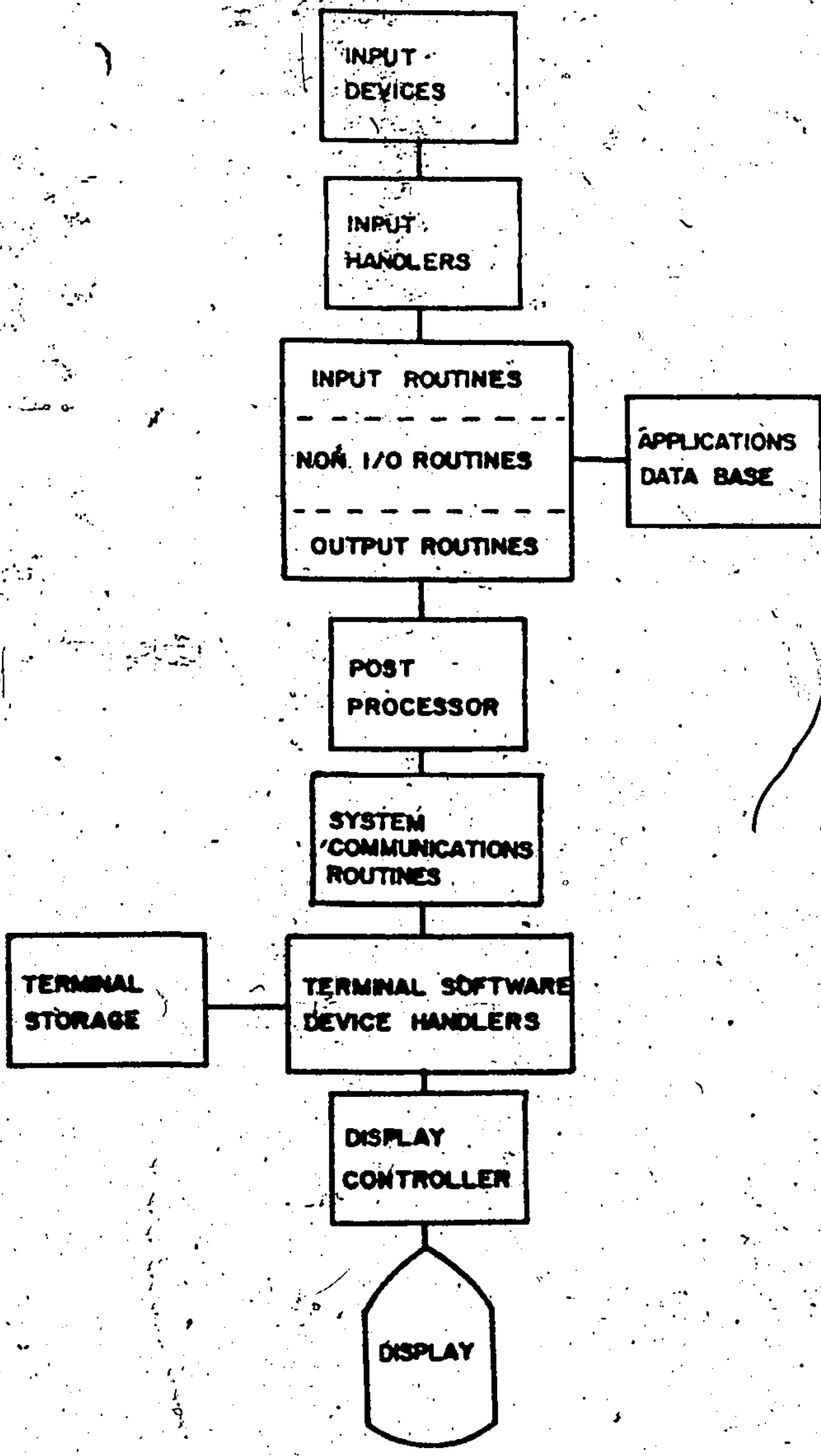


Figure 2.2 Relationship between Graphics Elements.

system that must be dealt with in the development of a complete facility. These are as follows.

1.) The response time for the system in each of its operating modes. Reference (26) describes an experimental study of the effect of user's response in a sequential information processing task. More frustrating examples are the delays, inherent to the system, for the central processor's "roll-out" response lag experienced by users of timesharing systems being supported on mixed (batch and timeshared) mode processors. Foley (27) states that a clever structuring of the order of processing tasks can minimize the psychological effect of this delay on the user. This is related to the second consideration below.

2.) The structure of the conversational portion of the man-machine dialogue. These conversations, both verbal and graphical, must be conducted without ambiguity to either the machine's interpreter or to the user (28).

This second consideration is the bridge between the graphics system and the applications program. The utility functions included in the graphics software must permit the applications programmer to communicate with a potential user in an unambiguous and psychologically satisfying manner. This can only be determined by designers working with applications to real engineering systems.

These questions are dealt with in Chapter 4 of this thesis which describes CONSTRUCT, a software system for structural systems .

2.3 Structural Analysis and Synthesis Systems

The past two decades have seen the generation of several very large and comprehensive systems for the computerized analysis of all forms of structures (29), (30), (31), (32), (33). These have, for the most part, been well received by the user community for application to real structural problems. These systems have all approached the subject of structural analysis with the premise that the engineer should not be forced into the role of a computer programmer. The ICES system and the GENESYS system both employ a problem-oriented language that is command structured (ICETRAN and GENTRAN respectively). These are languages that allow the user to employ the language of his own discipline and not the language of the computing system. This argument has been repeated in several forms by many authors, (27), (28), (29). Alternatively the STARDYNE system gives record by record instructions to guide the user through the creation of a standard FORTRAN-like data deck. Common to each of these systems is the off-line preparation of the input data base. One feature of NASTRAN and GENESYS that should be adopted for all systems, and is essential to the continued growth of automated methods in engineering, has been the establishment of formalized procedures (3) for updating and distributing the program revisions amongst the user groups.

In contrast to the acceptance of computer systems for analysis, optimum-structural-design systems have not yet gained intensive usage. There are many possible explanations for this lack of acceptance by practitioners in the field. Gallagher (34), attributes it to the following reasons; unfamiliarity of the practitioner with mathematical-programming concepts, the array of alternative methods available in mathematical programming, and the large costs of optimum-design analysis. In addition to these considerations, most formulations in the literature are generated independently of the mode of application, and this has left the user to implement the model in one of the conventional computer environments. In batch processing, the user is excluded from the iterative design process and, subsequently, his judgement is not brought to bear on the direction of solution. The cost to converge to a highly accurate solution that violates a neglected or overlooked constraint (aesthetic or logical) can be high. Alternatively, the user that implements the model in an interactive environment is confronted with the high data transfer problems described in the introduction to this work.

Bayer et al. (30) developed an interactive 'front end' for control of the STRESS (36) system. This was accomplished using a problem-oriented language CESLSTRESS. This system was based on manual analysis and did not incorporate any mathematical programming strategy. In essence, the system interactively assembled and reviewed the application data base for the STRESS system. Even at this primitive level, using only a teletype, Bayer was able to demonstrate substantial savings over conventional batch processing methods.

When formal optimization techniques are employed in the solution algorithm, the computational effort is expended to determine the optimum solution in a region bounded by constraint functions which define feasibility or no failure of the system. The imposition of these constraint functions is the portion of the solution that still requires the experience and intuition of the design engineer. Douty and Shore (37), in addressing this aspect of design, state: "The ability to move a design subjectively in an economic direction while maintaining a precise balance between all cost factors in the entire procedure is not acquired easily, and the art of doing so has been endowed with certain mystical traits largely attributed to conditioned intuition reinforced by years of experience".

The development of mathematical programming algorithms has been responsible for the transfer of some of the "mystical" aspects of design into a rational process. However, in spite of this development, there has been a limited acceptance of these tools by the practicing professional. This is understandable in the absence of a completely general procedure for all classes of structures. Machine structures in particular abound in degrees of difficulty for the development of a universal algorithm. As work continues in the development of numerical methods, that proportion of engineering design that lies outside the bounds of realistic quantification will diminish while that proportion that can be appropriated to automated techniques will grow. With these limitations on the present practice of structural design, a system, to be successful, must incorporate both the engineer and mathematical methods into a harmonious balance.

The development of a comprehensive system for the automated synthesis/analysis of machine structures, in addition to the graphics system described previously, macroscopically encompasses the following areas.

1.) Design of a high level language strategy for man-machine dialogue.

2.) Incorporation of a section to develop the application program data base through the graphics system.

3.) A data management system for retrieval of developed data for a particular configuration or catalogued data for general application.

4.) Service programs for data file creation.

5.) A strategy to implement one or more of the set of mathematical programming algorithms whose domain of application intersects the requirements of the problem. This section would benefit from an automated selection technique.

6.) An analysis program.

7.) A strategy for presentation of whole or partial solutions for review by the engineer.

8.) Dynamic response - a system to synthesize a structure under dynamic response constraints.

9.) Thermal response - a system to design a structure subjected to thermal response constraints.

10.) Modularly structured program with documentation - the program must be able to be easily modifiable to incorporate any technical improvement in the state of the art.

The factors that must be considered in the selection of an appropriate algorithm for the optimization portion of the synthesis system is the form of the objective function, the form of the constraint relations and the nature of the design variables. If the objective function and the constraint relations are linear, then the problem is easily resolved by the well known methods of linear programming. If the functions contain any non-linearities the problem of selecting an appropriate algorithm is a great deal more difficult. In addition, if all the design variables are continuous over their range in the design space the solution is more readily obtained than if these variables are discontinuous or discrete. There is unfortunately no satisfactory algorithm that can be employed for the design of machine structures which are non-linear and which possess both continuous variables (wall thickness for cast sections) and discrete variables (plate thickness for welded sections).

There is considerable effort being extended in the search for new methods to resolve the difficulties of non-linear systems, discrete and mixed variable systems.

The work on the non-linear problem described above is being conducted in the following areas:

1.) the transformation of the non-linear problem into a series of linear problems; (38), (39), (40).

2.) the application of gradient methods; (41), (42), (43).

3.) penalty function techniques; (44), (45), (46).

In addition, the deconposition of large problems into a series of smaller problems has been investigated by several authors, (47), (48).

The discrete variable approach has been investigated by Aquilar (49) and Palmer (50) in the application of dynamic programming. Palmer discretized the assigned nodal moments of framed structures and used dynamic programming to select members based on a solution restricted to lower bound limit load analysis. Twisdale and Klachaturian (51) used a similar approach under more restrictive assumptions. Dynamic programming as developed by each of these authors is applicable only to structures that are serially decomposable. This method is hampered by the difficulty of producing a general-purpose computer code (52).

Reinschmidt (53) applied the discrete member approach to a solution of framed structures using upper bound limit analysis and zero-one programming techniques. The large number of collapse mechanisms in practical structures creates a massive problem formulation. Cella and Logcher (54) used a branch and bound technique to solve a framed structural problem drawing from discrete candidate tables. This was accomplished using the conventional elastic analysis constraints solved in the ICES (29) system.

Gisvold and Moe (55) used a penalty function approach for the mixed variable problem where a subset of the design variables were constrained to take on only integer values.

Each of the above methods achieved a degree of efficiency when applied to a particular class of problems or when applied to a restricted subset of more general problems. No method has achieved universal application status. Thus, to design in the real world it is necessary to be able to draw upon these methods within the framework of a larger system. The development of libraries (52), (56), (57) of routines with standardized application procedures has made this incorporation more easily achieved.

Pierson (58), in a survey paper on the development of automated synthesis of structures under dynamic response constraints, indicates that the technique with the broadest application to realistic problems is the finite-element method of analysis directed by a mathematical programming algorithm. This problem, as with the static analysis, can be computationally very large. However, Pierson argues that the method is feasible if the number of design variables is kept small.

Sata et al. (59) demonstrate that the thermal analysis problem, in addition to the static and dynamic analysis, is resolvable by using a finite-element approach.

The common application of the finite-element method to all three areas of structural synthesis, static, dynamic and thermal, dictates that any general system for the automated synthesis of machine structures must include this capability.

A brief description of a proprietary system to automate the design of automobile frames in use by Daimler-Benz is described in a paper by Guedj (66). This system uses a digitizer and an IBM 2250 graphics display in association with an IBM system 360 computer. Interactive modifications can be made at the terminal while operating on-line. The performance of the structure is computed using a finite-element program.

2.4 Computer-Aided-Manufacturing

Chapter 5 of this thesis contains a detailed description of the application of the hardware and software developed in the course of this work to an application in the computer-aided-manufacture of glass bottle molds. This section of the literature review is a general review of the background of computer-aided-manufacturing (CAM) systems and a specific review of the literature pertaining to the automated manufacture of glass bottle molds.

Computer-aided-manufacturing as it is now established is an outgrowth of three previously separate areas of intensive study.

1.) Computerized inventory control and production scheduling. This includes the study of optimal allocation of product to production facility, part classification and data base management systems for information storage and retrieval.

2.) Computer-aided-design.

3.) <u>Direct numerical control of production facilities.</u> De Vries (60) predicts that the manufacturing world is on the threshold of unprecedented advance and that the full impact of this emerging technology on the metalworking industry has just begun to be realized. In this paper, De Vries examined the effect that this technology can have on the small batch durable goods manufacturing industry. He proposes an algorithm to project the comparative costs for CAM versus conventional manufacturing processes.

Kawahata (61) considers the information hierachy as it relates to the management organization. In this system development, the concept of computer-aided-manufacturing is the total process of converting the long range production plan, as developed by senior management, through the information system and management organization until the final stage of generating pulses for the control of the production facility has been reached. A similarly scaled system has been proposed for development by the National Engineering Laboratories (U.K.) in a paper by Fleming and White (62). In this proposed system they describe the first stage of development, an elaborate direct numerical control system. An anomaly exists within the development of such large scale approaches to CAM in that those industries that are sufficiently large to afford their implementation are normally those industries that enjoy the economies of scale associated with large production runs. Conversely, the small batch shop is the operation to gain the greatest advantage from automated production techniques.

There is clearly a need to develop CAM technology for the small job shop. Thus, a computer-aided-manufacturing system can be implemented on a scale that parallels and automates those functions present in the conventional part processing system regardless of the scale of that operation. The initial stage in the manufacturing of any product is engineering design, which is followed by development and finally, the production stage. These functions can all be realized in a simple three stage hierachial computer system. These stages consist of the following:

1.) Large scale computer for processing those tasks that are computational in nature, eg., APT processing, data base extraction, production scheduling.

2.) Design/development computer and data retrieval system.

3.) Control computer.

This work has been concerned with the development of an inexpensive hardware system for the intermediate stage in this proposed system.

The recent growth in the direct application of mini-computer to the control of machine tools has experienced a form of cyclic recreation. This was described by Meyers and Sutherland (63) in a different context; as 'the wheel of reincarnation syndrome'. One of the first applications of direct computer control of machines was done at Lockheed-Georgia in the control of a milling machine. Subsequently; there was considerable interest in the promotion of this technology. The use of general purpose computers in the control function became known as DNC (direct numerical control). The creation of inexpensive mini-computers with an adequate instruction set replaced the larger computer in the control function. These systems have become known as CNC (computer numerical control) systems. This transition brought about a division of labour between two processors in one system. The large scale general purpose computer was employed in the processing of the part program via one of the higher level languages (APT, etc.) and subsequently produced a data file (paper tape) for the controller. The mini-computer controller, working from this data file, performed the processing functions of interpolation and produced the pulses to control the machine tool. Subsequently, mini-computers have grown in both size and capability to the point where users are again considering the total process to be conducted within the control itself. Coincident with this development, work is being done on the production of micro-processor controls. The optimal appropriation of tasks within a multi-computer system is still to be resolved.

The continued development of the higher level languages for part programming is almost exclusively being conducted in the large scale computing system environment. There is no mini-computer based system that possesses the full complement of programming functions of the APT system. To compound this problem, the development of sculptured surface capability is largely being done in association with, and being integrated into, the APT processor. Even for those systems that are developed independently from APT, a large computer is required for processing. Ishimatsu et al. (64), in a paper describing the sculptured surface system used by the Toyota Motor Company, indicate that this system is approximately three times the size of the APT III system - 100,000 FORTRAN statements.

The above developments in programming languages, as well as the improvements in timesharing, are reasons to believe that the large scale computer still has and will continue to have a role in numerical control part programming. The areas that need to be developed are related to the utilization of this capability within an efficient and inexpensive computer-aided-manufacturing system.

The development of stand alone mini-computer processing capability for all aspects of the numerical control part programming process has been described by Gott (65) and Guedj (66). In the first system the functions required for numerical control have been augmented to a graphics system GINO-F developed by the Computer-Aided-Design Center in Cambridge (United Kingdom). This system is proprietary and costs in the order of 100,000 dollars to implement. The argument Gott proposed in support of this approach is that the appropriate level for integration of computer-aided design and computer-aided manufacturing is at the FORTRAN level. An argument that can be used against this level of integration is that it ignores the present state of numerical control programming languages and the degree with which these languages are entrenched in practice. Gott indicates that their system is in an early stage of development and that verification of their approach still requires considerable work.

Gott's paper describes two additional systems that are being developed, POLYSURF and GREYSCALES. The POLYSURF is a system for perspective presentation of three-dimensional parts. The GREYSCALE package is used in conjunction with POLYSURF to produce half-tone images of the perspective part. This form of graphics, although still of a primitive hardware state, offers a tremendous potential for the visualization of manufactured goods for which aesthetic values are an important feature (prior to the production of a prototype).

Guedj's paper describes several applications in Europe in which CAD techniques have been employed in association with the numerical control parts programming process. Guedj claims that interactive graphics, when used in conjunction with a large scale computing system, was sufficiently expensive to generate a need for a stand-alone capability to perform the functions of interactive graphics and NC programming. In this paper he describes a system configured around an IBM 1130 with a plotter and an IBM 2250 graphics console. The NC processor in the system is MINIFAPT. This processor is a restricted subset of the APT processor. The premise for the justification of the stand-alone system is not supported in the paper.

Two papers in the literature deal directly with the application of CAM technology to different aspects of bottle mold design. Gardner (67) describes a commercial system that was developed as a stand-alone graphics facility for the automated design of bottle molds. This system utilizes a storage tube, a tablet and a plotter to translate an aesthetic design into a working drawing for the manual production of a "clay prototype".

Folwell (68) describes the numerically controlled manufacture of glass molds by the family of parts programming concept. In this paper he describes the development of APT macros for the generalized production of TV picture tube molds.

There is no agreement in the literature on a common defintion of computer-aided manufacturing. To some authors, it means the full automation of a production facility including the control of all machines and all service functions. To other authors, it is the application of computers to the control of a particular machine performing a particular function. In this thesis, computer-aided manufacturing is taken to mean the automation of the functions involved in the design, development and manufacture of a product regardless of the number of products produced or the scale of the production operation. The design of a computer-aided manufacturing system is the appropriation of computing tasks amongst the elements comprising the system. Within the context of this definition, a minimum system would comprise the following elements and capabilities.

1.) Sufficient processing power to support a numerical control processor that meets the needs (both actual and anticipated) required in the manufacture of the product; similarly, sufficient capacity to handle all the engineering analysis/synthesis required in the product design.

2.) A design system that has sufficient software and hardware support to allow the designer to exercise his judgement free of any laborious tasks.

3.) A system to develop interactively the product in the pre-prototype stage.

4.) A control system to supervise the production of the product.

5.) A communications link between elements for direct information transfer between system components.

These functions have been integrated into a prototype system for the full automation of the production of bottle molds from a design sketch through to numerically controlled manufacture of a prototype bottle/mold. This work is described in detail in Chapter 5 of this thesis.

CHAPTER 3

EDIT SYSTEM

3.0 General

In the introduction to this thesis, the requirements of an engineering design computer terminal were specified as follows.

1. Low cost. The total terminal cost must be within the budgeting limitations of an engineering design office.

2. Ease of operation. The terminal's operation must be sufficiently simple to be handled by an analyst with a knowledge of FORTRAN and a minimal familiarity with the time-sharing facility being used.

3. Local collection of data. The terminal must have the capability to assemble data while operating in a stand-alone mode.

4. Data editing. The user must be able to edit data locally at the terminal without the need of a large scale computer.

5. Verification of input data. The terminal must be able to employ a graphics display to verify the assembled data file.

6. Data storage and remote access. The facility to off-line store data files and subsequently direct access to these files via directives generated by the host computer's software system.

7. Off-line review of graphically presented aspects of a design.

8. Local regeneration of repeated display functions.

9. Analogue to digital conversion. This is required for the reduction of graphical data or non-dimensional drawings to a format suitable for processing by the host computer.

This chapter of the thesis contains a detailed description of the development of a terminal system that fulfills the above specifications. A block diagram that indicates the physical association of the components that comprise this system is outlined in Figure 3.1. This terminal has been given the mnemonic EDIT for Engineering Design Intelligent Terminal.

Section 3.1 comprises a brief description of the hardware elements employed in the system. Under a subsection for each device, there is a description of the interface characteristics. Also included is a discussion of possible alternate devices that were considered for inclusion in the system. Often financial considerations dictated the choice of hardware employed. Where this has occurred, a preferred choice has been indicated. A minimum system has been described.

Section 3.2 contains a description of the terminal operating system. This section details the development of the operating criteria, and the adopted philosophy employed in the development of the strategies for interaction between system components when the terminal is operating on-line to

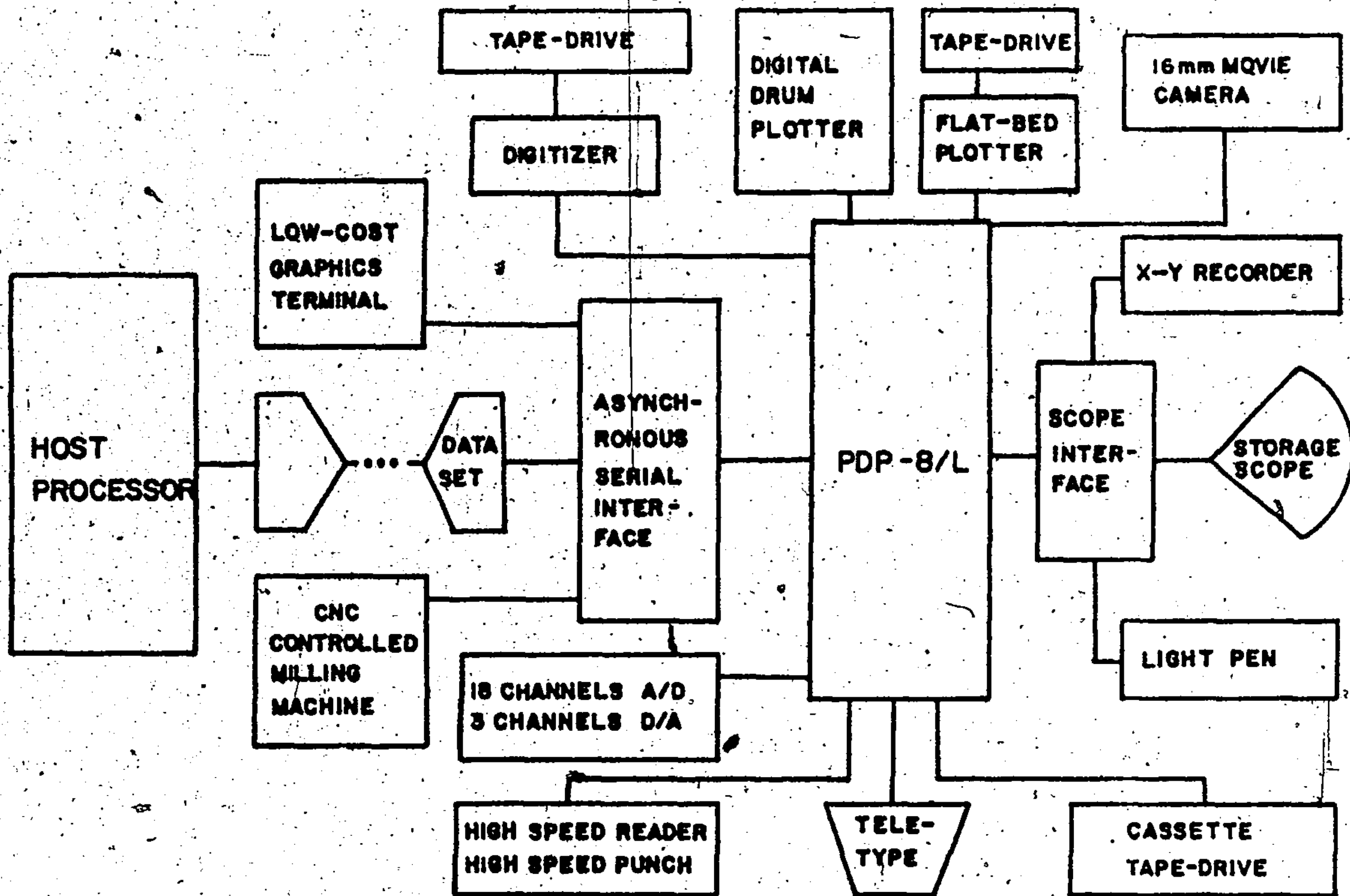


Figure 3.1 EDIT System.

the host timeshared computer.

Section 3.2 also chronicles the development of a graphics/terminal support software system that is located in the large scale computer. This software package is the interface between the user's application program and the terminal's operating system.

Section 3.3 embodies a description of the stand-alone functions that the terminal is capable of supporting.

3.1. Hardware Description

This section of the thesis contains a brief description of the individual hardware elements that comprise the terminal developed in the course of this work.

(a) In-terminal computer

The EDIT system has been developed about a minimal miniprocessor, a PDP8/L[1] with 4K of core. One of the criteria imposed on the development of this terminal was that the cost of the terminal system must be kept within a price range that would produce a cost-effective system suitable for a small to medium sized design office. Since the terminal's total cost is greatly influenced by the cost of the system processor, one of the tasks of this work was to evaluate the capabilities of a system structured about a minimal processor. Functioning within this constraint, it has been possible to

1 PDP8/L is a registered trade mark of the Digital Equipment Corporation, Maynard, Mass.

devise an operating strategy that has resulted in an efficient employment of this processing power to produce a flexible engineering design terminal. Over the course of this work there has been, and continues to be, a considerable development effort in the advancement of minicomputers with this order of processing power. The development of the INTEL 8008[1] chip and the release by the Digital Equipment Corporation of a PDP8 compatible processor based on this technology reinforces the logic of developing terminal systems about this class of computer. The selection of a suitable processor is the development stage during which the system designer is most likely to succumb to the "for only a few dollars more we could" (11) syndrome. A number of advantages were realized by the employment of this particular computer.

1. A wealth of software had been developed which was freely available through a well established independent user's group (DECUS)[2]. This is a "hidden" feature of a computer that is often overlooked at the time of selection.

2. Good documentation is available. The literature on this product line available through the manufacturer and the user's society was sufficiently comprehensive to simplify the task of interface design to the level of the non-specialist.

1 INTEL is a registered trade mark of the INTEL Corporation.

2 DECUS - Digital Equipment User's Society, Maynard, Mass.

Several disadvantages of this particular computer should also be noted.

1. All data transfer between the processor and peripheral devices is conducted through the computer's single accumulator. In the design application this did not prove to be a limitation. It may cause the computer to become input/output bound as the central information distributor in a multicontroller computer-aided manufacturing system.

2. The computer is not equipped with a hardware interrupt priority system. This limitation, necessitating a software priority system in a multiperipheral environment, results in a decrease in processing efficiency when operating with the interrupt facility enabled.

3. The twelve bit word of the PDP-8 is not a compatible word size. The majority of minicomputers currently produced have adopted a sixteen bit word size. This did not detract from this system's performance, but may have limited the transportability of the developed technology.

4. The PDP-8/L is the only positive logic miniprocessor to have been widely marketed. This required the purchase of an expensive logic level converter to employ current logic technology in the system's interfaces. This disadvantage does not apply to the current PDP-8 models on the market.

The minicomputer is equipped with a model 33 teletype as well as a type PR8/L paper tape reader and PP8/L paper tape punch (Figure 3.2).

Figure 3.2 PDP8/L Computer, Paper Tape Reader/Punch and SYKES Cassette Unit.

(b) Display CRT

The CRT used as the graphics display device is the Tektronix 611 Storage Scope. (Figure 3.3). In Chapter 2 of this thesis, a brief comparison was presented of the various display formats available for the generation of graphics. The storage display has the advantages of being able to support a greater display density than an equivalently priced refresh CRT. This distinction between the systems is important in the display of complex cutter paths for the verification of numerical control data files. Although not exploited in this work, this scope, with a non-storage operating mode, offers the possibility of supporting both stored and refreshed displays simultaneously.

The principal objection to the storage scope is the inability to dynamically modify an active display. It is not possible to selectively erase portions of the display without erasing the entire display. This disadvantage has been minimized by the development of a segmented display strategy for the terminal operating system. This is described in detail in Section 3.2 below.

In the scope controller, the X and Y, D/A converters have 14-bit reversible counters operating a transistor switched ladder network. This provides an addressable display of $\pm$ 8192 x $\pm$ 8192 programmable locations. The smallest deflection thus possible within the display area corresponds to a 0.0005 in(.0125 mm) movement. Thick lines are produced by generating a low-level cyclic signal and applying it to the

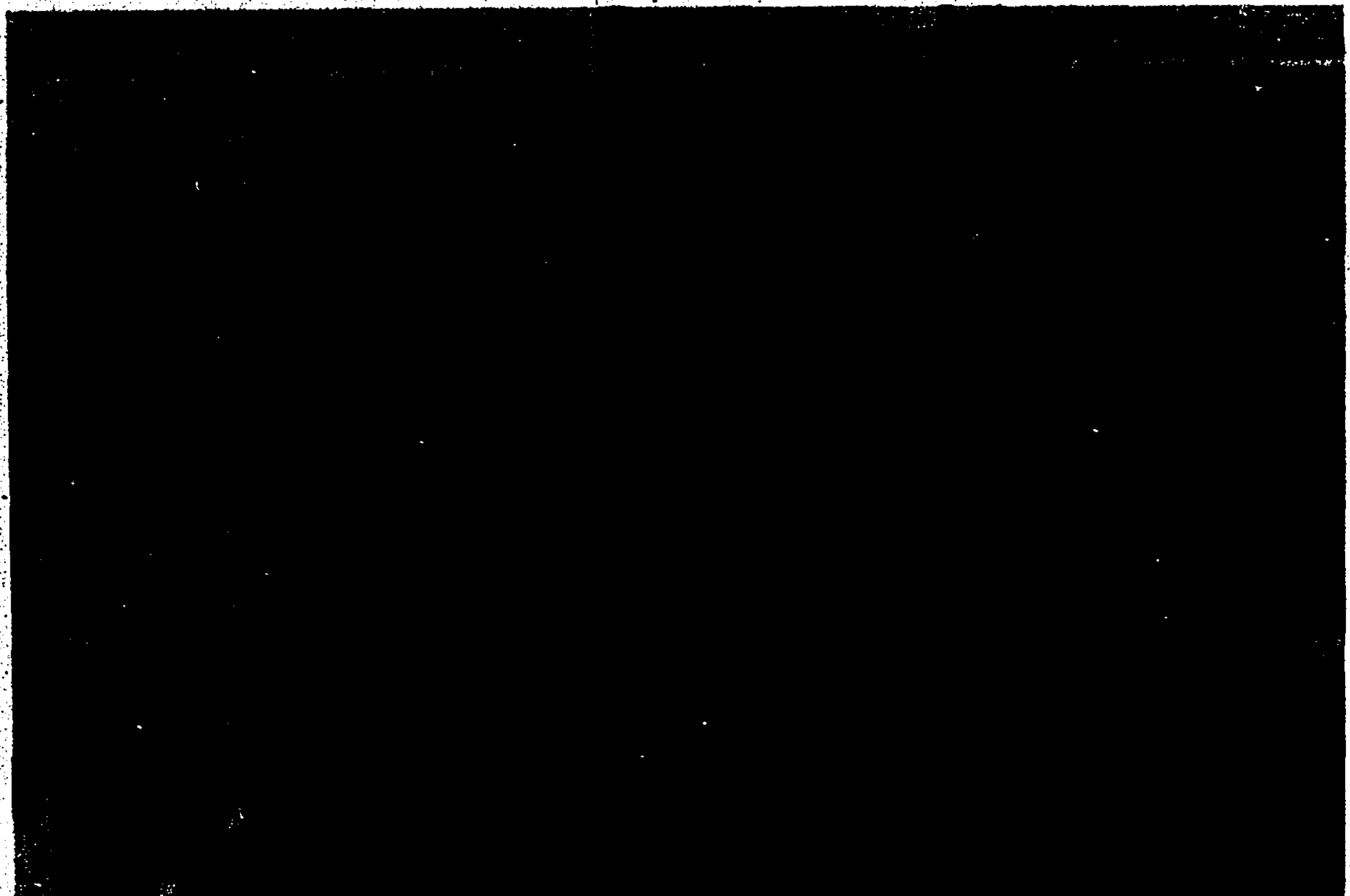


Figure 3.3 Tektronix 611 Storage Display Scope and Light Pen.

analog signal of the deflection amplifiers. The video control also amplifies the CRT beam current when a thick line is being drawn. Since there is a lag in the deflection system it was necessary to apply an 18 millisecond delay to the bright signal to avoid fade at the end of a long line.

Also incorporated into the scope controller is a hardware interpolator which allows the user to specify straight lines, complete circles, or circular arcs. A hardware character generator allows two character sizes in one orientation to be produced.

Since the terminal's operating system has been constructed to distinguish the functions of graphical and conversational communication, the total number of characters generated within the graphics portion of a session is not, generally, large enough to affect the processor's efficiency when these characters are generated through software. The communication strategy adopted has the capacity to distinguish as many discrete character sizes as there are number of characters available in the host system's character set. This is usually a minimum of 64 sizes. Thus, a redesign of this facility would delete the expensive and inflexible hardware character generator. The signal to the CRT beam current generator is modulated to produce hardware generated segmented lines. This scope controller has been manufactured by Ferranti Packard Ltd. A complete description of the controller is contained in Reference (69).

In addition to the above scope controller, the terminal has a two axis scope control which is part of the DEC[1] standard

1 DEC - Digital Equipment Corporation.

AX08 Laboratory peripheral. This controller has a third control channel that has three discrete output voltages that are under program control. This third channel can be used to blank the CRT beam or control the raising or lowering of the pen on an X-Y recorder. The AX08 peripheral is equipped with eight multiplexed analogue/digital converters and is designed to be used for real-time data acquisition and control of laboratory experiments. This peripheral, as with any other peripheral, can be employed in association with the EDIT system. However, there has been no provision made for the on-line (to the host computer) data acquisition capabilities of this peripheral. The low communication rates available, coupled with the small size of the core buffer area available when the operating system is loaded, made the on-line operating mode for this peripheral infeasible.

Associated with the display is a Lake Electronic Model 271 light-pen. This is a light-pen that has been specifically designed to function with the slow phosphor characteristics of the Tektronix 611 storage scope. The write-through mode of this scope has been employed to produce a non-storing cursor with discrete intensification pulses of 25 microseconds. The output pulse, generated by the photo-transistor detection of the scope light beam, is used to activate the interrupt line of the AX08 peripheral. The light-pen has been interfaced to the PDP8 by tieing one of the AX08 A/D input channels to the light-pen output line. The threshold level for triggering the input channel was controlled by a Zener diode.

Even though the light-pen's voltage amplifier for the detected intensification was attenuated 40 dB at 60 Hz, the noise on the output signal was sufficient to produce an interrupt when the pen was activated in the presence of fluorescent lighting. In addition, the capacitive switch, used to activate the pen, reacted to the amount of perspiration on the user's hand. The above two difficulties were not satisfactorily resolved, and further development work is required to implement this device. The details of the hardware interface for the light-pen are included in Appendix H. Alternate approaches to the cursor control function are the use of joysticks, function buttons and tablets.

(c) Digitizer

The digitizer, interfaced to the PDP8 and utilized as the input station for encoding nondigital data, is a Ruscom Logic Model 21. This digitizer is equipped with a control panel, keyboard, and two drawing stations. The drawing stations are a back-lighted table with a working area of 12 x 30 inches (Figure 3A) and a standard tilt-table with a working area of 60 x 36 inches. The smallest digitizable increment is 0.010 inches with a resolution of 0.010 inches and an overall accuracy of ± 0.010 inches in 10.0 inches. The current cursor position is continuously displayed on a resetable digital readout. The digitizer also is equipped with an independent magnetic tape drive for record purposes or off-line data processing. This magnetic tape unit is a write only drive and is available only to the digitizer.

Figure 3.4 Ruscom Logic Model 21 Digitizer.

Reduction of the analogue data to a digital form is achieved with an optical incremental shaft encoder mechanically linked via wire to a cursor. The shaft encoder outputs are connected to an up/down counter. These counters are cleared (to reset the data origin) with a button on the control panel. The output of the counters is used to drive a display for each coordinate axis and are also fed to a memory buffer. This buffer is used to allow the cursor to be moved while data is being processed. A thumb wheel switch is used to select between data transmission rates of 10 characters/second and 300 characters/second. In addition data can be logged via a footswitch, or automatically when a selectable displacement increment has been exceeded in the horizontal axis.

The interface between the PDP8 and the digitizer is used to receive 6 bit parallel BCD coded characters transmitted in serial blocks of up to 10 characters. The digitizer clock is used to clock the characters out of the digitizer and set the interface flag. The IOP4 pulse generated by the computer is used to strobe the character into the accummulator. The interface flag in the zero state inhibits the digitizer from transmitting characters. The interface is designed to allow servicing the data lines under the interrupt facility. The complete design of this interface was done by the author and is included in Appendix E, with a detailed operating description.

There are two essential requirements for a device to translate graphical analogue information into digital information. First, there must be a display available for the digitizing process. Observation of the use of the existing digitizer indicates that a high proportion of input data errors occur from duplicate digitizing of information. Most programs to interpret this data are not flexible enough to handle this type of error.

Secondly, there must be a local coordinate display for visual interaction in the digitizing process. This is required when data logging occurs at a specified interval in one of the axes on the drawing. This is the most common mode of digitizing non-analytic curves for regression analysis or integration. In the design case for instance, a curve may terminate at a known horizontal coordinate but unspecified vertical coordinate, the local readout allows the user to exercise judgement in the acquisition of the digital information for this point. These simple designer decisions are an integral part of any design system. The coding to circumvent this form of logic can be computationally complex.

In addition to the digitizer described above, there is a graphics tablet. This is a device for translating a hand sketch into digital data. This appears to have a great deal of potential as an aid during the problem formulation stage of the design process. The limitations of the tablet are the tremendous amounts of data that can be generated for a relatively simple sketch. The software reduction of this data as well as data smoothing assumptions appears complex. To date there has only been some preliminary testing of this device, and it is not integrated into the system.

The principal disadvantage of the implementation of these graphical/data input translators, when utilized in conjunction with a non-intelligent terminal system, is the requirement that the work be executed on-line. The intelligent system can be used to store, edit and display digitized data while operating as a stand-alone system. This data, when in an acceptable format, can then be transmitted to the host computer for processing in the most efficient mode.

(d) Hard Copy of the Graphics

For the generation of a hard copy of the display there are three techniques that can be employed:

1) simultaneous display copying on an electrostatic printer;

2) photographing the display scope;

3) transmission of the display data to a conventional drum or flat-bed plotter.

The EDIT system has the latter two capabilities. The obvious problem with hard copy devices is the relatively slow data acceptance rates possible with electro-mechanical devices. These plotters require data rates in the order of one hundred and ten bits per second for movements that are in the order of 3.0% of a full scale movement. (These figures are based on tests conducted on an EAI 3500 Dataplotter and ~~on a~~ Hewlett Packard 703513 X-Y recorder). Thus to employ a non-electrostatic device for the production of a hard copy of a CRT display, it is necessary to clock the data rate down and to perform data interpolation for movements greater than 3.0% of full-scale CRT deflections. This restriction obviously eliminates the non-intelligent display terminal-electro-mechanical plotter combinations, since it would be necessary to distinguish the nature (code and data rate) of the plotter data transmitted to each device. The intelligent controller can take the CRT display data and display it at the full display rate allowed by the deflection amplifiers for the CRT, while it buffers or stores the interpolated display information for the plotter. This has the advantage of reducing the data display file data transmission to a single pass. The electrostatic printer is not limited by the slow data acceptance rate restriction; however, to date these devices are not capable of producing the quality copy necessary for engineering analysis.

Photographing of the CRT display is excellent for record purposes but suffers the following disadvantages: The cost for instant processing of the film image (i.e., the 3M 2000 series processor camera) is extremely high. The delay involved in more conventional processing systems is likely unacceptable for most engineering applications. The utilization of negative film images for analysis requires a display device which restricts the engineer's natural predilection to do display modification with a pencil.

This work has shown that acceptable quality drawings can be produced on very inexpensive X-Y recorders by using the CRT digital to analogue converters with the local processor, interpolating the CRT display data file and clocking it out at a compatible rate. Drawings produced in this manner are limited in size as well as methods of character generation.

The flat-bed plotter (Figure 3.5) used in the system is an EAI 3500 Dataplotter. This plotter, although it is rather old and somewhat obsolete, incorporates several features that facilitate its utilization in a timesharing environment. One of these features is a typewriter head with forty eight characters and symbols. This results in the plotter producing a character on a one to one basis with each character transmitted. There is no need for software generation of script characters. This produces either a saving to the user resulting from a reduction in the number of characters transmitted between the host computer and the terminal or from an increase in local processor efficiency. A second feature is the use of servo motors to drive the pen in each axis. The digital input is decoded into analogue signals which are applied across two orthogonal slide wires. The variance from a null condition on these slide wires is used as the input signal to the servo amplifiers and subsequently to the servo motors. This allows the system to absolutely address any point in the plot area. For relocation moves with the pen up, this results in savings again for the same reasons as given above.

Some additional features of this plotter are as follows:

1) It is equipped with an eight pen turret allowing

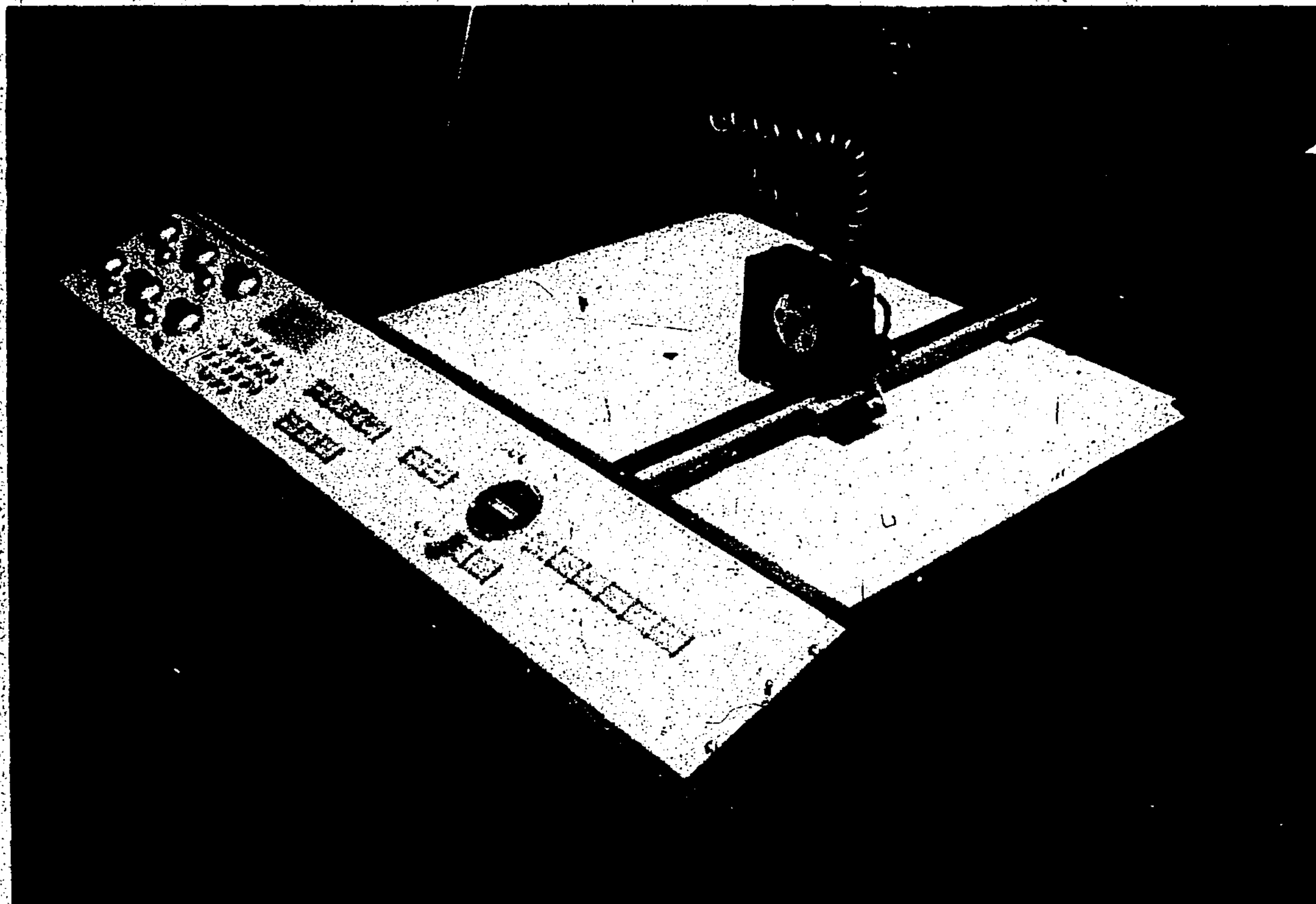


Figure 3.5 EAI 3500 Dataplotter with 8 Pen Turret and 48 Character Printer Head.

software selection of various line widths and/or colours.

2) The plotter's accuracy is ± 0.05% of full scale ± 1 bit of full scale count.

3) The plotting speed is a function of the input data rate and the distance between plotted parts. A typical plotting rate is in the order of one hundred and twenty inches per minute.

There are several design features related to this type of plotting format that produce an adverse effect on its performance in a terminal environment. One of these is the time interval between plotted points. This irregular input rate requires buffering the data and subsequent servicing of the plotter input requirements via the interrupt facility. Another feature is the open loop control for the servo motors. There is no feedback for correction of errors resulting from amplifier drift. This requires an extensive warm up time to stabilize these amplifiers. The design of the servo drive amplifiers includes two final stage reed relays that open circuit the input to the servo amplifiers. This was included to allow the transient signals induced when the plotter executes a mode change to be passed through the system without affecting the arm position. These reed relays can be held open for as long as two seconds. A number of mode changes in rapid succession can result in a buffer overflow and subsequently lost plotter data while operating on-line. These last two features added to the complexity (and the expense) for the development of a suitable interface for the plotter.

A complete description of this interface, which was designed by the author, is included in Appendix F.

A second hard copy device (Figure 3.6), is a Calcomp 565 twelve inch drum plotter. The implementation of this plotter is still in progress at the time of writing, thus no assessment of its actual performance is available. This plotter is a type that has a fixed plot increment, which dictates the inclusion of the interpolation software to be resident in the local processor in the absence of medium to high data communication rates. Characters plotted on this plotter will be generated with software. Complete details of the interface, designed and developed in this work, between the Calcomp plotter and the PDP8 are given in Appendix G.

The third hard copy device in the system is a micro-film plotter. This plotter employs a high resolution non-storage CRT mounted in a 3M 2000 series processor camera. This produces 35 millimeter negative images of the CRT display, mounted on a standard computer card (an aperture card). This plotter is slaved to the same scope controller as the Tektronix 611 storage tube display described above. This plotter was manufactured by Ferranti Packard Ltd. The obvious advantage of this type of system is the operating speed which is only limited by the display rate of the CRT. A disadvantage of this medium is the need for an additional display device to view the plot. In order to compare this plotter's performance with a conventional plot the accuracy is referenced to a 15 times magnification of the 35 millimeter film negative. This corresponds to a drawing size of 23.4x16.5

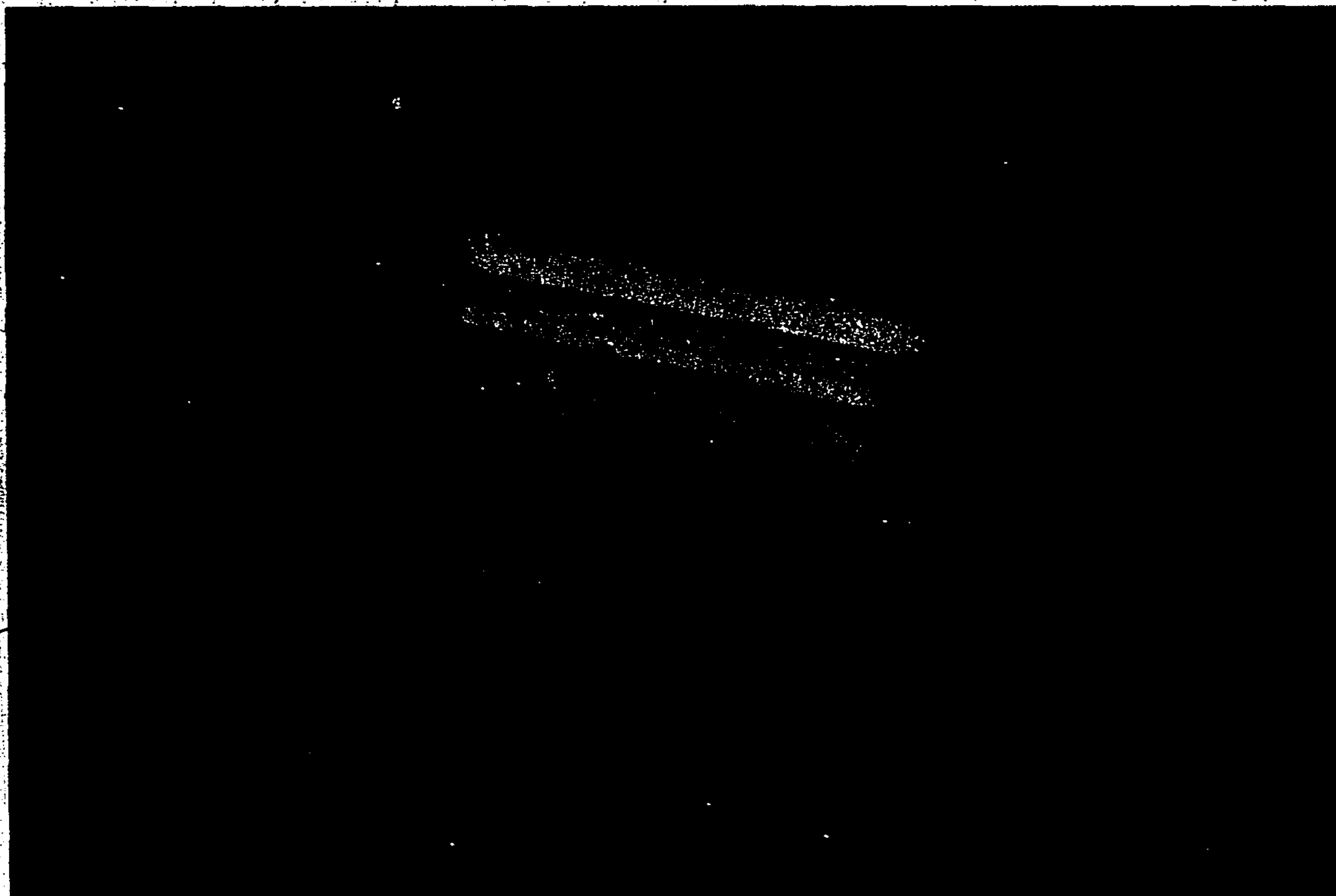


Figure 3.6 Calcomp 565 Incremental Drum Plotter.

inches. At this size, over a 12 inch square in the center of the print, the accuracy is ± 0.5% of the distance from the center of the print along each axis. Over the whole drawing area the accuracy is ± 0.85% of distance from the center of the print along each axis. The addressable resolution is 0.002 inches and the line widths are 0.012 inches and 0.024 inches. Complete operating details of the interface for the microplotter to the PDP8 are given in Appendix P. This interface was constructed by Ferranti Packard Ltd.

(c) Communications

The operating system for the EDIT terminal has been designed to allow communication between the terminal and three external devices. These are: (1) the host timesharing system, (2) another terminal (Computek 400) and (3) a mini-computer used to control a milling machine. The hardware to accomplish this is an extensively modified DEC PT08BFX asynchronous interface. This interface was modified to allow communication rates of 110, 300, 600 and 4800 baud, and to accommodate character blocks of 9.5, 10 and 11.5 bits. The nonstandard 9.5 bit character format was required to achieve compatibility with the Control Data Corporation communications handler. At the present time the terminal-to-host computer communication is conducted via the telephone lines at communication rates of 110 and 300 baud. These low speeds are the only two rates supported by the Control Data Corporation 6400 INTERCOM 4.1 timesharing system at McMaster University.

The modem selected was an Edmund Newhall Model 112 frequency shift-keyed acoustic coupler for both full and half duplex transmission, EIA RS-232 compatible. The other two devices are hardwired to the EDIT terminal, operating over a range of speeds depending on the application. Since the PT08BFX is a single port interface the alternate devices are switched into the port as required. Complete details of the modifications required for the PT08BFX interface are given in Appendix I. The modifications were designed by the author.

The low communication rates of 110 and 300 baud available on the in-house timesharing system had the beneficial effect of forcing a considerable effort to develop an efficient communications strategy that minimized the information flow between processors. This is consistent with the design criteria of low cost and remote operational capability. However, a more desirable system would likely result from communication rates in the order of 1200 baud. This rate is still below the point where it becomes necessary to employ expensive modems and leased lines.

(f) Mass Storage Device

There are four different devices commercially available for the in-terminal retention of data files. These are: (1) additional core memory beyond the system's processing requirements; (2) discs (including fixed head, cartridge and "floppy" discs); (3) magnetic and paper tape; (4) cassette tape. There are advantages to each system as well as disadvantages. The core memory has the fastest retrieval time but has the highest cost per unit of storage capacity. Discs are the next fastest system but have the highest capital cost of the systems described. The use of industry compatible magnetic tape has the advantage of (limited) transportability between computer systems, but also has the disadvantages of being expensive, slow and strictly a linear medium. Paper tape is inexpensive but with a very low recording density; a small amount of stored data can produce a great deal of paper tape. This is also a linear storage medium. Cassette tape units are inexpensive, the storage efficiency is high but they are machine dependent. These are both linear and random access depending upon the recording format of the manufacturer.

In addition to core storage in the PDP8, the mass storage device selected for this terminal is a SYKES COMPU/CORDER 100 magnetic tape cassette transport. This particular system employs a recording strategy that makes it suitable for use in a timesharing environment. A cassette tape consists of a 0.150 inch wide magnetic tape on which two parallel tracks of information are recorded. The majority of manufacturers record the data on both tracks with parallel redundancy. One manufacturer offsets the second track. The justification for redundancy is improved reliability. The SYKES system records data on one track only. The other track contains prerecorded tape addresses in the form of sequential binary numbers. Using the supplied software, the address track may be used to directly access these recorded addresses and, thereby, the relative positions on the adjacent data track.

The data track is recorded at 1000 bits per inch using phase modulation. There are two data access modes for data retrieval. The first is a shaft encoder which generates pulses that correspond to the recorded address locations on the tape. With the head retracted the tape is coarsely positioned at 100 inches per second by monitoring the number of pulses generated by the shaft encoder. In the second access mode the head is then lowered and the address track is read and verified. Thus it is possible to employ this unit in a disc-like fashion. Data is read at 5 inches per second, producing a data transfer rate of 5000 bits per second.

The interface for this unit was supplied by the manufacturer.

The cassette is an inexpensive compact medium by which each terminal user can record and retrieve data files relevant to his particular application. In recent months there have been a number of "floppy" disc systems released. These systems have most of the advantages of the cassette system and, in addition, have true random addressability with slightly faster data access times. The cost of these units are in the same range as the cassette transport.

(h) Additional Hardware

The terminal, in addition to the devices described above, is equipped with eight channels of multiplexed analogue to digital converters, three channels of digital to analogue converters plus a sync pulse output line. This sync pulse has been employed to drive a Bolex 16 millimeter movie camera for the production of computer generated animated movies (Figure 3.7).

This section has dealt with the hardware elements that have been integrated into a computer terminal for the purpose of computer-aided engineering design. The choice of those elements has been subjectively compared with possible alternate components that could be employed to achieve similar capabilities. The subsequent section will describe the software required for the terminal design as well as compare the capabilities of this system with other system designs.

3.2 Software Systems

(a) Terminal System for On-Line Operation

The complete design of an intelligent terminal includes the design and development of two interrelated software systems. One of those systems constitutes an operating system for the terminal's minicomputer. The other system is the terminal support software and is resident in the host (large scale) computer.

The terminal's software system consists of two separate sub-sections; the on-line operating system and the

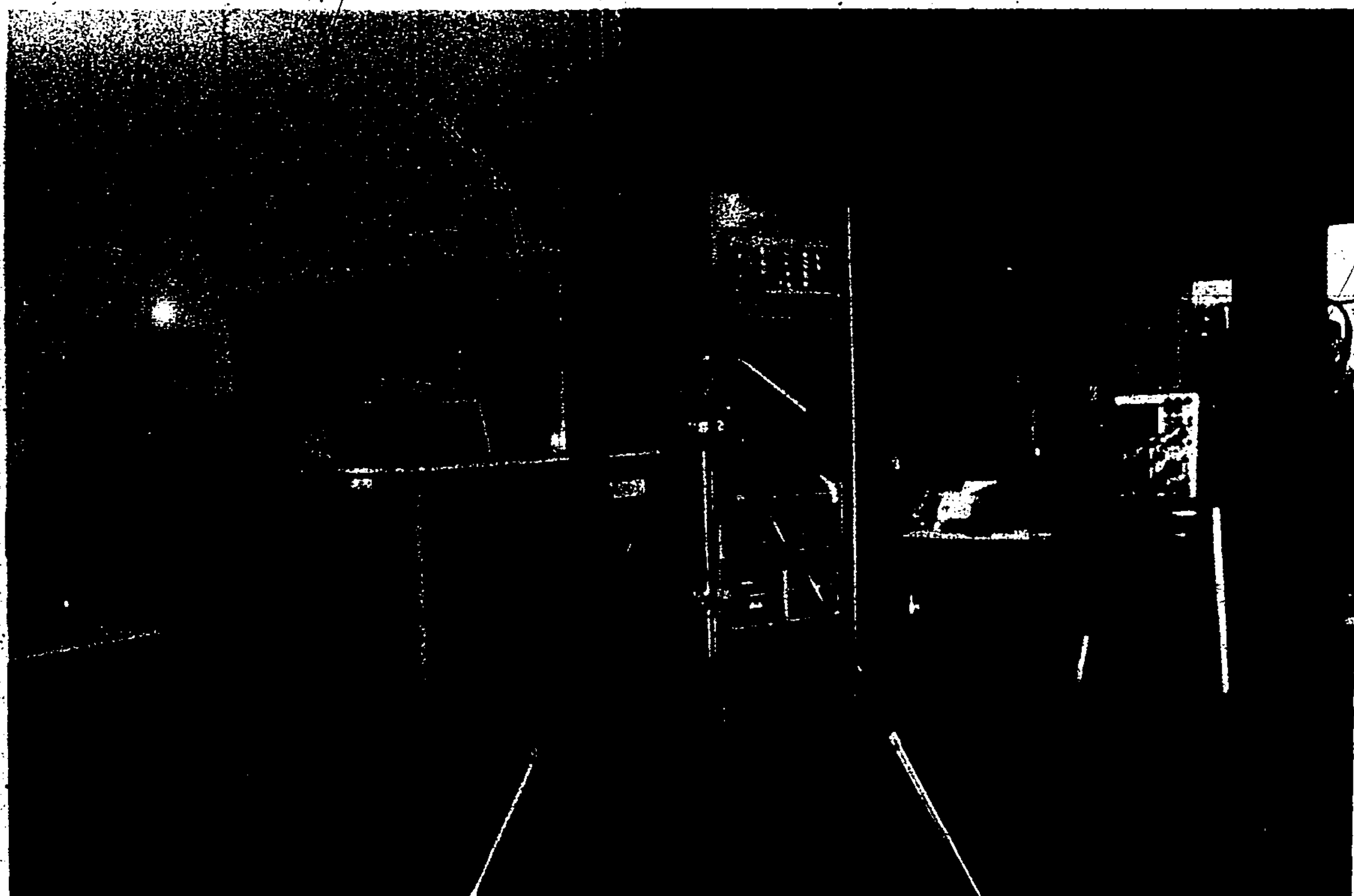


Figure 3.7 Terminal System with Bolex Movie Camera Installed.

stand-alone operating system. The on-line operating system manages the data flow and controls the peripheral functions while the terminal is working in conjunction with the large scale computer. Similarly, the stand-alone system is used to control the interaction of the peripheral devices, but while the terminal is functioning independently of another computer.

The nature of the information communicated between computers when on-line naturally divides into two major divisions. First, conversational information bound for the user to elicit some form of user response and nonconversational information bound for a peripheral device (example: graphical data for the CRT display). The intelligent terminal is thus able to decode character strings, interpret and direct that information to the appropriate device. Conversational text for the user is transmitted to the teletype while (for example) graphics is displayed on the CRT. On the other hand a non-intelligent graphics terminal mixes both forms of communication on the same device. For a storage-tube type display terminal, this requires that the entire graphics display be erased in order to erase and overwrite the conversational portion, requiring expensive software regeneration of the graphics up to the point of interruption. In the EDIT package, system diagnostics generated independently of the user's program can thus be recognized (as not being of the correct form) and transmitted to the user for response. The non-intelligent system, on the other hand while operating in a graphics mode, normally interprets these character strings as valid graphics and attempts to display them as such. The intent of the message is usually lost in the resulting graphical display. In the conversational mode, the local processor appears transparent to the user at the teletype, so that it appears to the user that he is communicating directly with the central computer.

The non-conversational information exchanged between computers is distributed to the appropriate device via a data switch. The ASCII[1] FORM character "/" (code 134_8, even parity) is used as the data switch to trigger a mode change for the terminal, the next character in the block then directs the software pointer to the appropriate address that corresponds to the starting address for the selected device handler. The second character following the form character is used to select peripheral options. Thus, a timesharing system supporting 64 characters can support an equal number of peripherals each with up to 64 operating modes. These character assignments with a brief description of the operating system response are described below.

"/"(134_8)-Starts the CRT display and conducts all incoming data to the scope controller via a buffer area. The ASCII characters CARRIAGE RETURN (2[illegible]$_8$) and LINE FEED (012_8) are stripped from the input data stream, since these characters would be interpreted as valid graphics characters

1. American Standard for Communication Information Interchange.

by the scope controller.

$\underline{S}$(244_8)- Starts the graphics display and the microplotter. The only difference between this mode of operation and that described above is that a copy of the display is recorded on an aperture card.

$\underline{?}$(277_8)- Starts the high speed paper tape punch. The received data stream is punched on the high speed punch.

$\underline{!}$(241_8)- Starts the digitizer. Coordinate data generated by the digitizer is read from the interface in BCD (binary coded decimal) code. These characters are converted to the corresponding ASCII code for transmission via the PT08 output channel. Since there is a keyboard on the control panel of the digitizer, the input modes for data generated by the digitizer consist of two types -- coordinate data (blocks of ten characters) and header data (blocks of unspecified length). These characters are read into a buffer area in the minicomputer using the interrupt facility. After each character has been processed, the computer enters a time loop; if no interrupt occurs during this loop the buffer is emptied and transmitted to the host computer. To the user, this means that coordinate data is processed intact as a complete block of 10 characters comprised of 4 integer numbers for the X-coordinate, a space, 4 integer numbers for the Y-coordinate and a space, while header data is processed in blocks of varying length. Since the character generation rate of the digitizer is substantially faster than the low communication rates between computers, the user must be cautious of overflowing the buffer and losing digitized data.

&(246$_8$)- Starts the cassette transport in the read mode. The device handler posts a query to the user on the teletype requesting a 4 digit tape address for the data record to be retrieved and transmitted. Completion of the record is signified by an ASCII # (243$_8$) character as the last character in the string. This character is transmitted to the host computer to re-initialize the conversational mode.

"(242$_8$)- Starts the cassette transport in the write mode. As with the read operation, the user specifies a 4 digit tape address at the teletype. All host computer activity is suspended until the tape position has been located and the unit is ready to begin to write. The local processor then activates data transmission by the host computer. The input data is buffered into 400$_8$ word blocks and is transferred to the cassette tape when the buffer is full. Transmission of an ASCII # character will cause the recording of a partial buffer, cause the tape to be rewound and returns control to the minicomputer's operating system.

'(247$_8$)- Starts the flat-bed plotter. The input data is decoded and buffered into a circular buffer. This buffering format is required as a consequence of the widely variable data acceptance rate of the flat-bed plotter. While executing a drawing with the pen down moving through small increments the data acceptance rate is such that the processor can virtually throughput the input character stream. However, when the plot increment becomes large - or the plotter mode is changed (i.e., from printer to pen or short line to long line) the acceptance rate is dramatically reduced. This forces all data servicing for the plotter to be conducted under the interrupt service facility. The circular buffer is a linear array of memory locations with a filling pointer and an emptying pointer. These pointers are incremented when the buffer is serviced and are tested against each other to prevent buffer overflow and to flag an empty buffer. The user must be cautioned against structuring a graphics program that schedules a number of plotter mode changes in rapid succession as the buffer could easily overflow. This shortcoming is resolved by employing the plotter in a deferred mode. This operating format is explained below. The input ASCII characters are converted to BCD (binary coded decimal) before transmission to the plotter.

% (245_8)- This character transfers the control of the minicomputer operating system to an unused block of core. This feature allows the user to program a specialized requirement for a terminal application. The user has two requirements to fulfill in exercising this option. First, his program must transmit a character to the host system to reinitialize data transmission, and second, the last instruction in his program must be a return jump to the starting address of the operating system (200_8). The host processor data generation activities are suspended for the period of time required for the local processor to perform all required functions. This is accomplished by placing the support software into a READ operation for data on the PTO8 output channel. If the time duration of the local processing is known precisely, then a much more efficient (in a computer sense) approach would be for the user's application program to schedule a roll-out from the central processor for the required time period. The standard option described above requires the host's communications monitor to periodically pole the user's input channel. With this option the full capability of the terminal is available to the user under control of his own program. A skillful programmer can structure his programs into overlay segments resident on the cassette system which are retrievable via the system's Library facility. Page zero constants, containing buffer storage memory locations, are also addressable by the user's program.

Experience with the timesharing system at McMaster University has indicated that the assumption of unchanging communication strategies on the part of the host system is not justified. The development of a graphics system which is locked into an operating format (such as Computek 400, Tektronix 4010, etc.) by hardware character decoding can cause the user a great deal of inconvenience when there is a change in the host computer's operating system. These changes often require extensive programming to circumvent at the level of the terminal support software system, whereas in the EDIT system they are easily implemented with a change in the terminal's graphics generator software. This problem is due to the lack of standardization in timesharing systems. Some systems are capable of full eight channel transparency on both input and output while others provide only six. Some systems support 128 character sets while others support only 63. These disparities are difficult if not impossible to handle with hardware systems.

Thus the central thesis of the operating system can be summarized as follows.

1. The terminal's operating system remains in one mode of operation until an ASCII FORM character is detected. Control is then returned to the input decoder routine. An error in an expected data string will return control to the user.

2. The first character to follow the data switch is used by the input decoder to direct the minicomputer's software pointer to the starting address of the appropriate device handler.

3. Subsequent characters received serve to direct the peripheral device according to the logic of each individual device handler.

4. Data bound for each device that can potentially operate with a slower acceptance rate than the interprocessor communication rate is buffered.

5. Buffered input devices are serviced using the interrupt facility.

In order to provide the user with the capacity to control the processing of information locally, a software interrupt priority system was established. In the conversational mode the devices which can interrupt the operating system execution in the order of priority are:

(1) PT08 receive channel

(2) teletype keyboard

(3) teletype printer

In the non-conversational mode, the devices capable of interrupting are:

(1) PT08 receive channel

(2) keyboard

(3) high speed punch

(4) graphics CRT

(5) digitizer

(6) flat-bed plotter

(7) drum plotter (under development)

The most important aspect of using this facility is servicing the PT08 receive channel. The response time must be within the transfer time for the stop bit plus one half bit at the completion of receipt of an 8 bit character. If the program fails to respond within this time a character is lost. This priority interrupt structure allows the user to terminate the current activity by pressing a key on the teletype. A subsequent entry of the CARRIAGE RETURN character will place the operating system into the conversational mode (i.e., teletype to timesharing system and vice versa). Unscheduled local system halts (crashes) can be recovered by restarting the system and entering a CARRIAGE RETURN; the activity in progress prior to the failure should then be resumed.

A flow chart depicting the information flow for the terminal's operating system and a program source code listing are given in Appendix C. A users' manual describing the operation of the terminal is provided in Appendix A.

(b) Host Computer Software System

The software system resident in the host timesharing system is an indexed library of relocatable subroutines that has been structured to allow the user to configure the terminal hardware systems into a format suitable for application to his engineering design problem. This software consists of device handlers, a graphics package, input and output decoders and display device encoder routines.

The software has been structured to allow the user to engage and disengage the terminal's peripherals by executing calls to higher level (FORTRAN IV) language subroutines. The peripheral devices are each assumed to have two states; active and deferred. In the active state, the peripheral is being employed in real time with direct access by the host processor. In the deferred state, the peripheral device is employed indirectly via a mass storage device. In this state the data retrieved from, or written to, the storage medium is in the format of the peripheral device. The preparation of these files or subsequent usage of them is accomplished using the stand-alone terminal functions. This operational mode is described in detail below.

There are two common approaches to the use of graphics in an interactive computer environment. One is the use of a command decoder, or user oriented language, which allows the user to enter (or designate via light pen or cursor arrangement) a graphics command that evokes a specific terminal response. This mode of operation is extremely effective for the construction of displays that comprise a relatively few components and where the decision variables are largely geometrical and simple (i.e., easily interpreted visually). This form of graphics is suited to the terminal system that uses the display for both conversation and graphics. An application that has been well developed for this type of graphics is the production of printed circuit boards.

The second approach is the interactive application of a graphics package. Here, the user's application program poses the queries to the user and generates the graphics as a result of the response. This is accomplished by the user's program logic which utilizes the graphics routines in a completely analogous manner to the application of other computationally oriented package programs.

The advantage of the first approach is the efficiency of conversation that can be achieved. The designation of a one word command followed by several parameters is sufficient to produce a graphical display. An example of this is the GRAPPLE (22) command:

$CIRCLE(150);

This produces a circle with a radius of 150 units. These plot units can be defined to have any actual value. The disadvantage lies in the need to become conversant with the language and the need to structure the display in real time with the user responsible for all levels of decision making.

The query-response approach to graphics has the disadvantage of generating a greater volume of conversation between user and host processor. It has the advantage of generating a more descriptive directive to the user in a language more closely related to normal conversation. The intelligent terminal's capability to distinguish the graphics and conversational modes and separate them out to different devices, makes the query-response approach more feasible to employ in an interactive environment. The adverse effect of a high volume of interprocessor communication can be minimized by the adoption of a command responsive/query-response program structure. This form of programming is described in Chapter 4 and Chapter 5 of this thesis.

A graphics system, to provide a flexible and efficient service for a range of applications, should have the capability of functioning under both of the above modes. The graphics system developed in this work has largely focussed on the package approach. However, a command decoder for the management of a few of the primitive operations has been tested to evaluate the terminal's performance while operating under the command decoder mode. The package, as well as the decoder, are coded in ANSI Standard FORTRAN IV to achieve some measure of inter-computer transportability.

The structure of this package is as shown in Figure 3.7. A generalized graphics system constructs a display file which is then post-processed for conversion to a particular display device. The basis of the coordinate system is a hexadecimal representation for each X and Y axis. This plot consists of an addressable array of 65536 (2^{16}) discrete locations in each axis. This is 1 bit greater in accuracy than the most accurate device in the system (the flat-bed plotter). This programmable matrix gives an axial plot length capacity of 27.3 feet and is 2 bits greater in accuracy than the CRT/microplotter. The function of the device post processors is to map this display into the display area available for the device. This is accomplished by dropping a sufficient number of low order bits until capability is achieved.

All transformation and scaling functions are applied at the level of display file generation. The plotter analogy is adhered to in the system application; thus the display file origin is centered in the display matrix. Negative coordinates are indicated by using two's complement arithmetic with a 1-bit in the most significant digit.

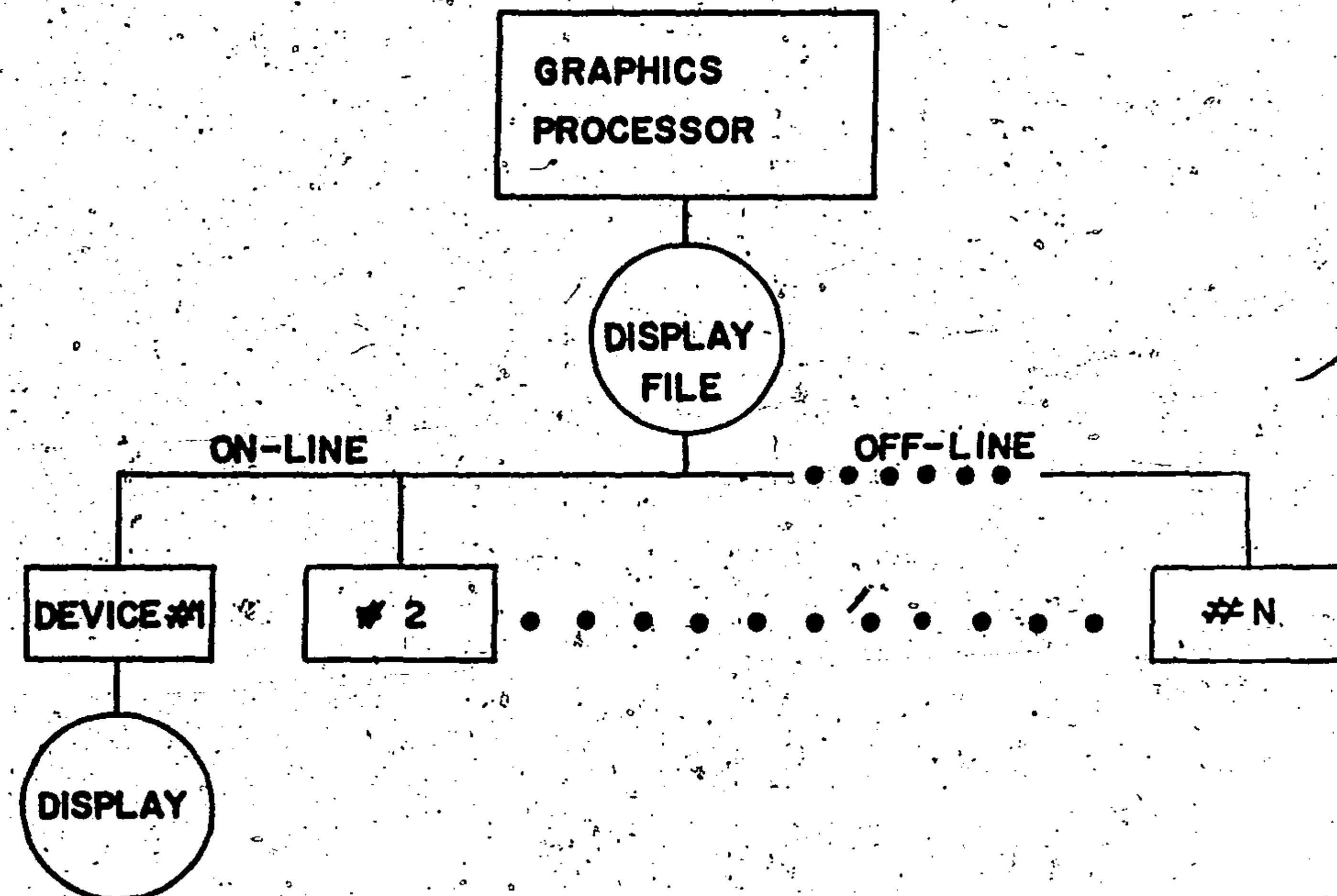


Figure 3.7 Graphics System Structure.

The hexadecimal basis was selected for the display file to be compatible with the word size of most current minicomputers. Each coordinate is represented by a 16 bit binary number. Each such number is represented by four characters from the hexadecimal set, determined by the following relations:

$$(X \text{ or } Y) = N_1 + 16N_2 + 256N_3 + 4096N_4 \text{ where } N_4 = 0,,,,7 \quad (3.1)$$

$$(X \text{ or } Y) = N_1 + 16N_2 + 256N_3 + 4096\,(N_4-16) \text{ where } N_4 = 8,,,,15 \quad (3.2)$$

The hexadecimal number set is the decimal number set (0-9) appended by the characters that follow directly in the ASCII code (P;<=>?). Thus the coordinates of a point that lies at (10 263, 12476) would be represented by: (7182,< ; 03) where:

for the X-coordinate

$$N_4 = 2,\ N_3 = 8,\ N_2 = 1,\ N_1 = 7$$

and

for the Y-coordinate

$$N_4 = 3,\ N_3 = 0,\ N_2 = 11,\ N_1 = 12$$

The hexadecimal base is common to a number of graphics systems. This particular coding format is an adaptation of a system developed by Ferranti Packard Ltd.

In the application of a graphics system to an intelligent terminal, the post processor, for the conversion of the display file, is most suitably located within the terminal processor. With this strategy the display file is transmitted to the terminal only once and is then post-processed for each terminal display device. In the EDIT terminal the CRT/microplotter system functions from the display file while the flat-bed plotter's post processor is located at the host processor.

In contrast to the EDIT system, with a conventional storage tube graphics terminal the display file is generated and transmitted by the software system functioning in the host computer system. To accomplish a modification or restructuring of a portion of the display, it is necessary to erase the entire display and regenerate and retransmit both the modified and unmodified portions of the display. This is a very inefficient consumption of communication channel time. Using an intelligent terminal, it is possible to segment the display file into a number of indexed records. These records are transmitted to the terminal for the initial display generation and storage on the terminal's mass storage device. Subsequent display modification is required only to update those display file records that are affected. The new display is thus regenerated from the local storage medium which contains both modified and unmodified records. Thus the information transmitted between processors is only the information relating to the modifications of the display and is usually only a small proportion of the total information contained within the display. At low communication rates, this can represent a significant reduction of processor time as well as terminal "connect" time.

In the EDIT terminal, display file segmentation is accomplished by writing the display file to the cassette tape in records of 400_8 characters. The 4 integer digit starting address for the location of the segment on the tape is used as the reference record identifier.

In addition to the above savings, it should be noted that the regeneration of the display from the local storage medium can be done repeatedly and in the absence of the host processor. This is another significant advantage of the intelligent terminal system.

The local display regeneration is normally processed at the maximum possible display rate set by the physical limitation of the display device. However, if dynamic effects are important in the sequential construction of the display, the data output rate of the local processor can be "clocked" out at a rate that is slow enough to produce an "animated" effect in the display construction. Both the repetitive display regeneration and the display rate modification features are beneficially employed in the verification of a parts program of complex sequential motions for the cutter of a numerically controlled machine tool.

To the terminal user, the ability to regenerate the display locally, at his request, relieves him of the psychological stress induced by the desire to minimize the cost of supporting the host processor in a relatively inactive state while the graphics display is reviewed.

The segmentation feature, of course, can also be applied to non-graphics files which were generated by the large computer and which are to be stored at the terminal.

The minimum capabilities that should be incorporated into a graphics system have been well established in the literature. The functions supported by this system are stated briefly here and are stated in detail in Appendix A.

1. Absolute lines. Display coordinates can be addressed absolutely with the pen up or down.

2. Incremental lines (vectors). Display coordinates can be addressed via long or short vector moves with the pen up or down.

3. Circles of any radius centered at present pen position.

4. Arcs.

5. Auto-scaling. The user can specify the hexadecimal coordinate system directly or can use his own data units and request auto-scaling.

6. Perspective. A three-dimensional representation of an object can be mapped into the two-dimensional display. The perspective technique used in this work is based on what Newman and Sproul (20) termed the eye coordinate system. A viewing transformation is employed to transform a point in object space (X, Y, Z) to a point in the eye coordinate system (X_e, Y_e, Z_e) (Figure 3.8). This transformation can be a concatenation of rotation and transformation. A perspective

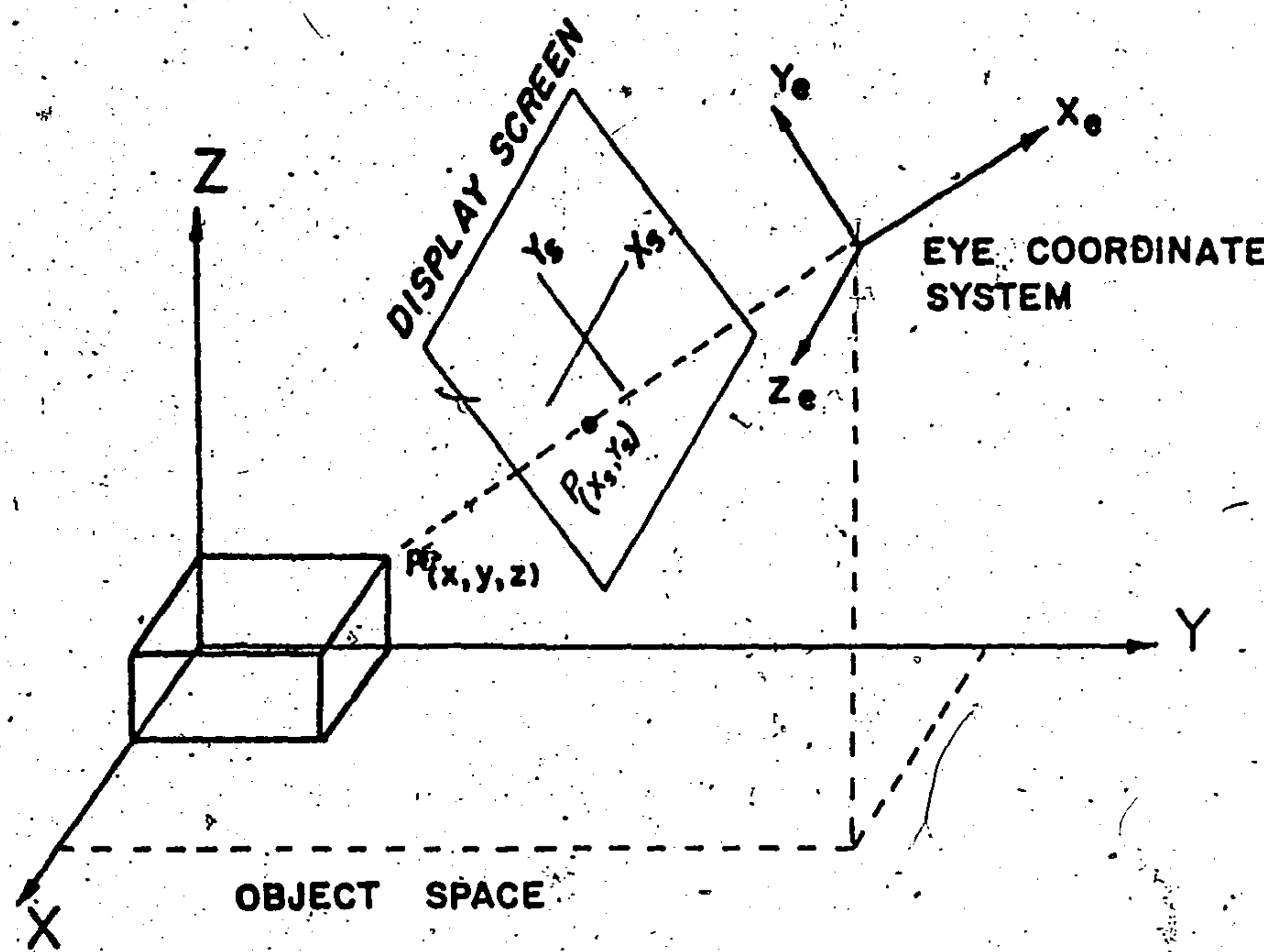


Figure 3.8 Perspective Transformation of Coordinates.

display is generated by projecting each point of an object onto the plane of the display screen. The display coordinate (X_s, Y_s) of the projected image of a point P in the eye coordinate system (X_e, Y_e, Z_e) are easily evaluated.

7. Text. A string of characters can be generated starting at a specified coordinate location.

8. Graph. A series of data points can be connected by straight lines. X and Y axes are drawn with calibration marks.

In summary, the graphics package consists of three parts, input device handlers, graphics functions and output handlers. The graphics system generates a device independent display file that is based on a hexadecimal coordinate structure. This display file is "post-processed" for each individual display device. The display file can be segmented into records of 400_8 characters, each with a 4 digit integer record identifier.

3.3. Stand-Alone Functions

An intelligent terminal, in addition to being able to share the processing load with the host computer, has the advantage of being able to perform a range of tasks as a stand-alone system. Most of these tasks in the context of engineering design center about data management, preparation, and display. A number of engineering design processes involve a substantial proportion of "think" time and are best performed while the designer is not under any implied pressure to keep a large scale computer system "active". An example of this process is the reduction of a detailed drawing of a proposed machine tool structure to sufficient information to perform a finite element analysis to determine the structural response to a number of loading conditions. The designer is faced with the determination of nodal locations that will provide the maximum information on the structure's performance, while at the same time will not computationally overburden the central computer. The assemblage of this information is greatly facilitated by the employment of a digitizer to record the modal coordinate data. However, it can still be a lengthy process. Using the terminal's stand-alone functions it is possible to record the coordinate data on a local storage medium (cassette tape, paper tape), edit this information and reformat it prior to connecting the terminal into the host computer. This file can then be transferred to the host system at relatively high speeds, reducing the terminal connect time appreciably. On the output, plot files for either of the mechanical plotters associated with the terminal can be transmitted to the terminal and stored on a local storage medium. The host computer can then be disengaged and the plot created from local storage. A frustrating experience that is avoided by this technique, as opposed to on-line plot generation, is the failure of the plotter (pen out of ink) in the midst of a lengthy plot.

For the EDIT terminal each peripheral has been assigned a two character mnemonic. These are:

(1) CA - cassette transport;

(2) PT - paper tape reader/punch (high speed);

(3) DI - digitizer;

(4) FP - flat-bed plotter;

(5) LP - light-pen;

(6) MP - microplotter;

(7) DP - drum plotter;

(8) TY - teletype (low speed paper tape reader/punch).

The stand-alone programs are each characterized by a 4 character mnemonic that forms a logical combination of peripheral interfacing. These mnemonics are concatenations of the device mnemonics which are structured as follows -

(information generating peripheral: information receiving peripheral)

Example:

PTPT - paper tape reader to paper tape punch.

Note that there is no ambiguity to this formulation as the inverse peripheral interaction is not logical (i.e., it is not possible to generate information with the paper tape punch or receive information with the paper tape reader). Thus, the program is used to read information from a paper tape and punch that same information on paper tape.

Other programs mnemonics are given in Table 3.1 with a brief indication of their function. A detailed description of each of these programs, with operating instructions, is given in Appendix A of this thesis.

TABLE 3.1 STAND-ALONE PROGRAMS

NAME	FUNCTION
PTFP	Paper Tape to Flat-bed Plotter
PTCA	Paper Tape to Cassette Tape
CAMP	Cassette Tape to Microplotter
PTMP	Paper Tape to Microplotter
PTCR	Paper Tape to CRT
DIPT	Digitizer to Paper Tape
DICA	Digitizer to Cassette
CAFP	Cassette to Flat-bed Plotter
PTPT	Paper Tape Reader to Paper Tape Punch
TYMP	Teletype to Microplotter
TYCR	Teletype to CRT
CACR	Cassette to CRT
PTMC	Paper Tape to Microplotter

These programs are all stored as absolute binary programs stored on an indexed library cassette and are retrieved and executed with a simple load command from the teletype.* A sample of this command is:

L, DICA

which tells the system to load the stand-alone program that records information from the digitizer and writes it to the cassette transport. The library program that is used for retrieval is equally easily employed to punch a copy of the program, list an index of programs available, delete programs, add new programs and change program names. This library program is part of the supplied software from SYKES DATATRONICS INC.

3.4 Summary

This chapter has described the development of a computer terminal system called EDIT. This terminal has been structured about a minimal minicomputer, and as such has demonstrated the capability of providing a greater degree of flexibility when applied to the engineering design process than can be achieved with the non-intelligent graphics terminals currently available commercially. Within the context of the definitions of terminal types described in Chapter 2, the EDIT system is classed as an intelligent terminal with some stand-alone features. Although this system is configured out of specific peripheral devices, the integration of these peripherals and the design of the software operating system have been developed in a general sense, and are therefore applicable to the development of any terminal system utilizing the fundamental concept of a programmable terminal controller. For the purposes of this work the terminal has been developed as a prototype facility that is completely modular in structure and is designed to be expanded easily to accommodate both additional hardware and software systems. A minimal system that has most of the capabilities of the system described above would consist of the hardware shown in Table 3.2. Representative costs have also been included.

A summary of the capabilities and features of this terminal is as follows.

1. The terminal is not locked into a communications strategy. The ability to program the terminal controller greatly facilitates the handling of changes within a time-sharing service, as well as differences between timesharing systems.

2. The terminal can perform the stand-alone functions of data assembly, data editing, data storage and repetitive display of previously generated graphics (via CRT or plotters). These features can be advantageously employed to reduce the harrassment of the engineer/designer by reducing the stress of minimizing incurred costs of supporting a large scale computer on-line while he performs an analysis of displayed results, or prepares an input data file.

3. The terminal can be employed in conjunction with a minimal timesharing service over ordinary telephone lines at

TABLE 3.2 MINIMUM HARDWARE CONFIGURATION

Peripheral	Description	Approximate Cost
Computer	4K Minimal Processor Intel or Similar Technology	$3,000.00
Serial Printer	Teletype ASR33	$1,100.00
Display Unit	Tektronix 611	$3,000.00
Display Controller	Hardware Interpolator Software Character Generator	$ 500.00
Asynchronous Interface	110 baud to 1200 baud	$ 500.00
Digitizer	Similar Technology to Ruscom Logic Digitizer using Shaft Encoders	$6,000.00
Plotter	Drum Type Incremental Plotter	$6,000.00
Mass Storage Device	Cassette Tape Drive or "Floppy" Disc Unit	$3,000.00

relatively low communication rates.

4. The terminal is modular in structure and can easily be upgraded to support a broad spectrum of applications. Peripheral devices can be changed as they become obsolete without incurring the expense of a total system replacement.

5. The terminal is cost effective for the engineering design application when compared with non-intelligent systems and with stand-alone graphics systems.

6. The support software to drive the terminal is display device independent which allows the user to easily add new graphics devices without a major alteration of the software system. The software is written in ANSI FORTRAN IV and is relatively machine independent.

CHAPTER 4

CONSTRUCT - A SOFTWARE SYSTEM FOR STRUCTURAL SYNTHESIS

4.0 General

This chapter describes the development of a software system for the analysis/synthesis of machine tool structures. This system has been designed to integrate the engineer/designer into a larger system comprising the above high level application software system, the terminal support software system and the EDIT terminal hardware. This work has employed the query/response approach to the interactive program structure. Flexibility and efficiency are achieved by using a command responsive input decoding system in conjunction with the query/response approach. This input scheme is explained more fully below.

Throughout the course of the development of this program, a standardized model milling machine has been used as the test structure. For part of the work, this structure was the idealized mathematical model developed by the CIRP (International Institute for Production Engineering Research). This is a milling machine model constructed to promote a program of cooperative development of computer-aided design for machine tool structures. This model is completely defined in reference (69). The synthesis portion of the system was tested for a structure similar to the CIRP structure, except for two minor modifications. The CIRP model is a milling machine that is geometrically similar to the structure of the machine in Figure 4.1.

The primary consideration in the design or analysis of the structure of a metal cutting machine tool is the determination of the relative displacement that occurs between the cutter and the part that is being cut due to structural deformations. These deformations are caused by the direct forces that are exerted between the tool and the part (static); thermal deformations due to the conduction of the heat generated during the cutting process by the machine structure; and vibrational distortion due to the presence of harmonic exciting forces (dynamic). All three of these deformation modes can be determined by an analysis technique known as the finite element method. The design/analysis of structures for machine tools is complicated by the operational mode of the tool itself. The function of a metal cutting tool is "to support the cutter and the work and to move them relatively to produce machined contours" (70). Thus, the geometry of the structures is not fixed within the tool and there are a large number of possible loading conditions that can arise within any subset of the possible geometrical configurations.

The design of machine structures was originally conducted by the construction of a model of the machine tool prior to the fabrication of the first actual structure. The investigation of a range of possible configurations was prohibitively expensive. The development of computer based

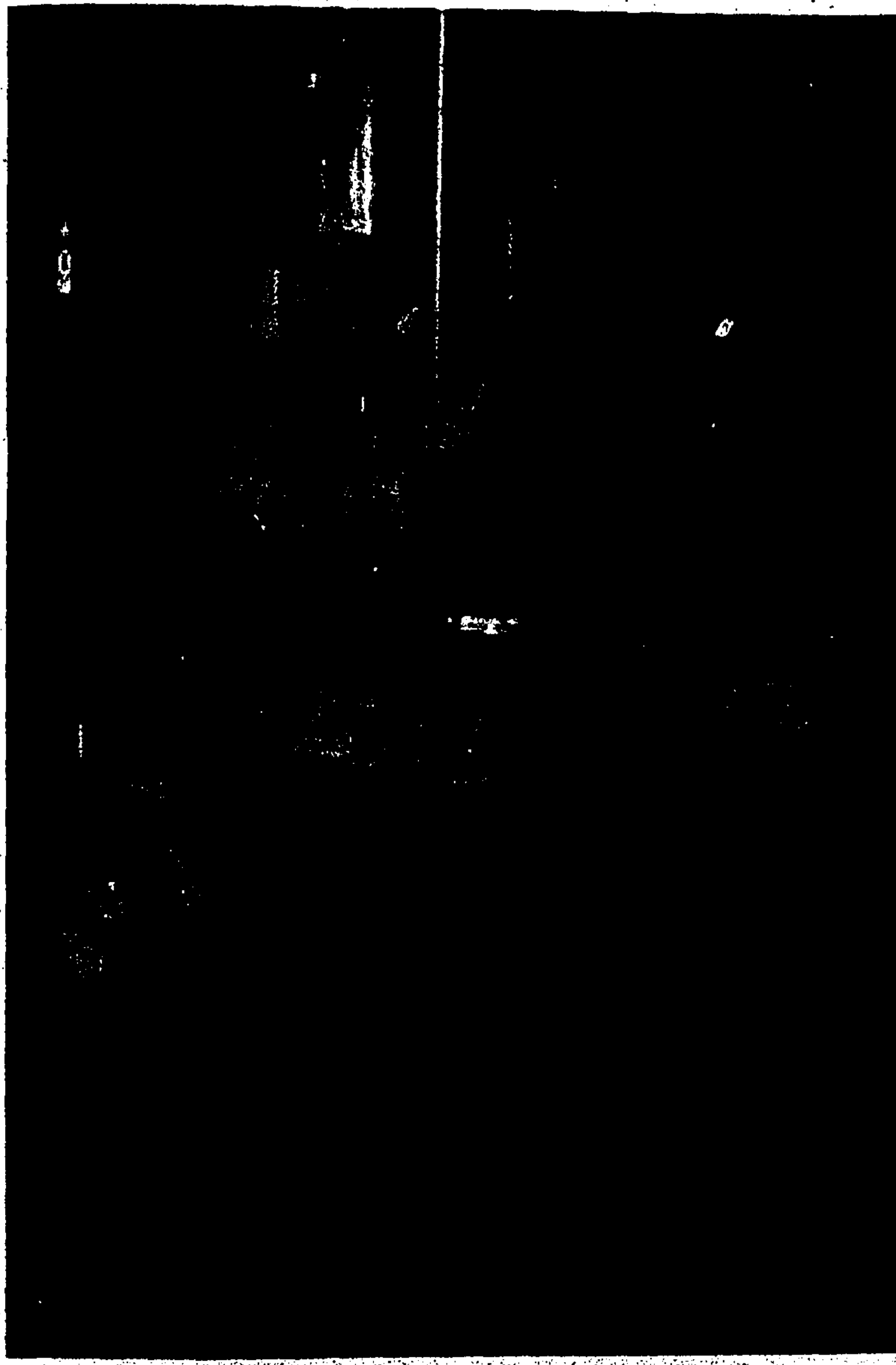


Figure 4.1 A Milling Machine with a Structure Similar to CIRP Model.

large scale analysis methods, such as the finite element method, has allowed the machine tool designer to transfer some of the preliminary design work to the digital computer. The transfer of all aspects of machine structural design/analysis has not yet been fully achieved. One of the primary obstacles to the complete automation of the design of this class of structures has been the need to meet a large number of complex geometrical constraints. Thus, machine tool design is still best conducted by the development of a preliminary design, followed by successive iterations until all constraints (geometrical or failure) are satisfied.

The basic concept of the finite element method is that every structure may be considered as a mathematical assemblage of individual structural components or elements. A finite number of these elements are connected at nodal points. The behaviour of this model will closely approximate the behavioural characteristics of the real structure if a sufficient number of elements are employed. Some basic definitions follow .

Nodes - Associated with each structural node are six degrees of freedom (three translational X, Y, Z, and three rotational, θ_X, θ_Y, θ_Z). These displacements are expressed in a right-hand global cartesian coordinate system.

Elements - Associated with each element interconnecting with two or more nodes is a stiffness matrix that represents the displacement at the structural nodes due to the applied forces. These element stiffness matrices can be combined to produce a displacement relationship for the structure of the form:

$$[k]\{\delta\} = \{F\} \qquad (4.1)$$

where $\{k\}$ = stiffness matrix

$\{\delta\}$ = nodal displacement vector

$\{F\}$ = applied force vector

The solution of this equation is the nodal displacement for the static analysis.

For the dynamic response, considering only free vibrations (no forcing function), the eigenvalues (natural frequencies) and eigenvectors (normal modes) of a structural system are determined by solving the equation

$$[k]\{\delta\} + [M]\{\ddot{\delta}\} = 0 \qquad (4.2)$$

where $[M]$ = mass matrix for the structure

$$\{\ddot{\delta}\} = \frac{d^2\{\delta\}}{dt^2}$$

Assuming a harmonic solution of the form

$$\{\delta\} = \{\bar{\delta}\}e^{i\omega t} \qquad (4.3)$$

Equation (4.2) becomes

$$\left([k^{-1}][M] - \frac{1}{\omega^2}[I]\right)\{\bar{\delta}\} = \{0\} \qquad (4.4)$$

The eigenvalues of $[k^{-1}][M]$, therefore, give the values of $\frac{1}{\omega^2}$ and hence the natural frequencies. The eigenvectors give the mode shapes.

The problem with the finite element approach is that the mathematical model is a reduced form or idealization of the preliminary design for the actual structure. Although this reduction is generally a straightforward task for the experienced machine tool designer, the logic employed in this modelling is largely intuitive and is thus difficult to automate. The designer must choose the location of the structural nodes such that the information derived from the mathematical model will be representative of the actual machine structure, while at the same time he must keep the number of nodes (degrees of freedom) within computationally feasible limits.

The inverse approach of formulating the mathematical model as the primary step and extrapolating the resulting design to a preliminary design is a complex problem involving the interconnection of elements and offers little advantage to the design process.

In addition, the transfer of the design process for machine structures to the digital computer has necessitated the generation of a great deal of information to describe both the geometry of the preliminary design and the properties of its component elements.

All of the above considerations affect the nature of the technique by which automated methods are adapted to the design process. Thus, the requirements of a system for the synthesis of machine structures can be outlined as follows.

1. There must be a system to aid the designer in the intuitive process of translating the preliminary model to the mathematical model.

2. There must be a rapid review procedure to verify that all of the input data has been assembled correctly. The computational aspects of a finite element analysis are sufficiently complex to warrant the visual review of all input parameters by the system designer. The batch processing approach can be costly if the structure has a large number of degrees of freedom and the data has been assembled incorrectly.

3. The load vector must be easily redefined. The system must have the capability of examining the response of the structure for varying load conditions without incurring the expense of inversion of the system stiffness matrix for each load case.

4. There must be provision made to retrieve information generated during previous runs as well as to store all information generated during the current run.

5. The design system must be modular in structure as well as modular in application. The structure of the system must be such that advantage can be taken of developing technology within any subsection of the system by simply inserting an appropriately coded block into the program without disturbing the operating sequence of the system. The application of the program should be modular to allow the user to

complete only partial runs within one terminal session. Information generated during these partial runs must be stored and accessible to retrieval for the execution of subsequent stages of the system.

6. There must be an easily employed mathematical programming or optimization technique to direct the variation of the design variables in the presence of performance constraints. The user must be able to designate the variables in a structural subset that will be allowed to enter the automated optimization process.

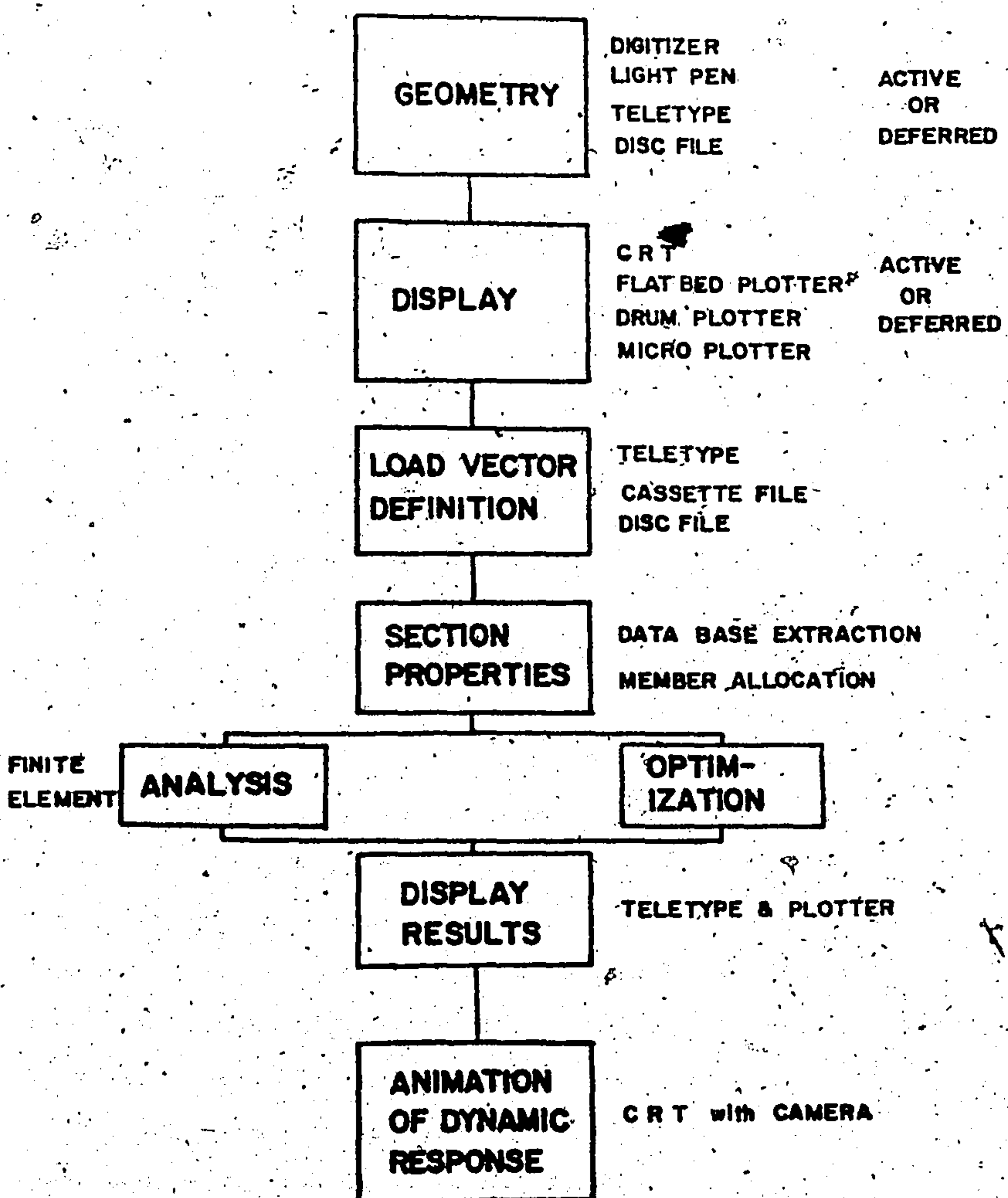
7. The user must also be able to manually direct the evolutionary process via an iterative application of the analysis technique. This must be possible by entering or reentering the system at any input stage.

8. The results must be presented both graphically as well as in tabular form. The graphical presentation must be unambiguous in the designation of structural deformations for both static and dynamic responses.

9. The system should be easily employed by the inexperienced user in a language that is natural and precise. At the same time, the experienced user must be able to employ the system efficiently without the annoying delay while lengthy explanations are transmitted.

4.1 Program Structure

A system for the synthesis of machine structures has been developed in the course of this work to fulfill the requirements stated in the previous section. The general structure of the software is outlined in Figure 4.2. The total program has been divided into eight separate sections that have been distinguished on the basis of full utilization of the features of the EDIT terminal system in order to attain the optimal division of tasks between the system elements. The program has been developed employing an OVERLAY structure where each of the eight sections comprise one primary level OVERLAY. Each OVERLAY has been constructed to function as a stand-alone program retrieving input data generated by a lower level section from disc files as well as recording all generated data on a disc file. Thus, all sections exchange data through an external mass storage device. All relevant files are attached to the user's terminal (control point) via a system macro. In order to prevent a system abort the initial application of the system will retrieve null files. The created files are automatically assigned a retention period of two days. Since it is possible to store up to five files under one file name under the CDC scope operating system, the program can be employed to update all information files a maximum of four times in the two day period before the user is required to purge files independently of the system. This restriction applies only to central site data files, as opposed to terminal data files. Those files stored at the terminal (cassette tape), of course, have a potentially infinite retention period. Each program section exercises the option of retrieving information via central site files or from the



CONSTRUCT

A System for the Synthesis of Machine Structures

Figure 4.2

terminal storage medium. Any files created at the central site can be transmitted to the terminal for storage with the aid of an independent program. This program structure enables the user to access the system and direct the system pointer to load any of the eight sections for execution. The user then has the option of terminating the session or proceeding with the execution of any other program section. Naturally, the user must be cognizant of the need to have previously executed (at least once) all lower order sections of the system within the file retention period.

(a) Command Structure

All input to the synthesis system that pertains to execution directives are passed through a command decoder. The purpose of this approach is to allow the user the flexibility to direct the order of the program execution as well as solicit assistance. This approach has been described in a general sense by Martin (28) and applied to a specific application in a computer-aided instruction package by James and Zachar (71). The system used in this work is an independent subset of the command structure described in the latter reference. The system commands available are given in Table 4.1 with a brief explanation of their function. Figure 4.3 is an example of the usage of one of the directives to obtain a more complete explanation of the computer generated query. The experienced user would be able to enter a response directly, thus requiring a smaller amount of information to be transmitted.

TABLE 4.1 SYSTEM COMMANDS

COMMAND	ACTION
BEGIN	Restarts program from beginning. This command allows the user to direct the software pointer to any program section since the program begins with a directive requesting section number to begin processing.
BACK	Steps the program back one step within a major program division.
HELP	This solicits a more detailed description of the program requirements for a particular program query or input request.
HALT	Rewinds all active program files. Stores all data on permanent files and returns control to the timesharing system.
AHEAD	Steps the program forward one step within a major program division.

This system has been tested for ambiguity by several inexperienced users and has been found to be easily comprehended and effectively employed. A complete conversation for structural synthesis is included in Appendix M.

```
CONSTRUCT - A SYSTEM FOR THE SYNTHESIS OF MACHINE STRUCTURES
---------

ENTER SECTION NUMBER                                      HELP

PROGRAM SECTIONS ARE:
     (1) GEOMETRY
     (2) DISPLAY (GEOMETRY)
     (3) LOAD VECTOR
     (4) SECTION PROPERTIES
     (5) ANALYSIS
     (6) OPTIMIZATION
     (7) DISPLAY (RESULTS)
     (8) ANIMATION OF DYNAMIC RESPONSE

ENTER SECTION NUMBER                                      1

PART ONE - GEOMETRY DEFINITION STAGE

THE NUMBER OF STRUCTURAL NODES =                          16

DEGREES OF FREEDOM/NODE=                                  3

ENTER (1) FOR PLANE OR (2) FOR SPATIAL FRAME              1

COORDINATE DATA INPUT SECTION

ENTER (1) FOR FPS OR (2) FOR CGS UNITS                    1

SELECT INPUT DEVICE:
ENTER (1) FOR DIGITIZER
      (2) FOR LIGHT PEN
      (3) FOR TELETYPE
      (4) FOR DISC FILE                                   1

NODAL COORDINATE INPUT DATA VIA DIGITIZER

ENTER SCALE VALUE (FT/IN),(F10.4)                         0.634

ENTER (1) FOR ACTIVE STATE OR (2) FOR DEFERRED STATE          HELP

ACTIVE - ON LINE FROM DIGITIZER
DEFERRED - FROM PREVIOUSLY DIGITIZED DATA ON CASSETTE TAPE

ENTER (1) FOR ACTIVE STATE OR (2) FOR DEFERRED STATE          1
```

Figure 4.3 Sample Conversation Indicating the Use of the HELP Command.

(b) Section One - Geometry Definition

Figure 4.4 is a schematic of the CIRP model milling machine and Figure 4.5 is a finite element model of this same structure. The model shown consists of 20 nodes and 20 members with a total of 120 degrees of freedom. Node member 0, which corresponds to the base of the structure, is assumed to be fixed in each of the six possible degrees of freedom. The machine is modelled with beam type elements. The first step in the performance of either an analysis or synthesis is the preparation of a digitized base of coordinate data and nodal connectively. Sutherland (73), in an elaborate expository on the reduction of multiple views of an object to a three-dimensional coordinate representation, defines a "magic" transformation matrix [M] such that the redundant coordinate is "automatically" discarded by the system processing the digitized data. He defines [M] as

$$[X_a \; Y_a \; X_b \; Y_b] \quad [M] = [X \; Y \; Z \; W]$$

where W is the redundant coordinate. This is a somewhat formal presentation of a very simple concept. The judicious selection of two views, when there are more than two views available, is the choice of those views that have the greatest number of points to be digitized colinear and parallel to the axis that is orthogonal to the plane of the digitizer. The reduction of complex structures can be greatly simplified in this manner. The redundant coordinate is then simply ignored or averaged with the value obtained from the previous

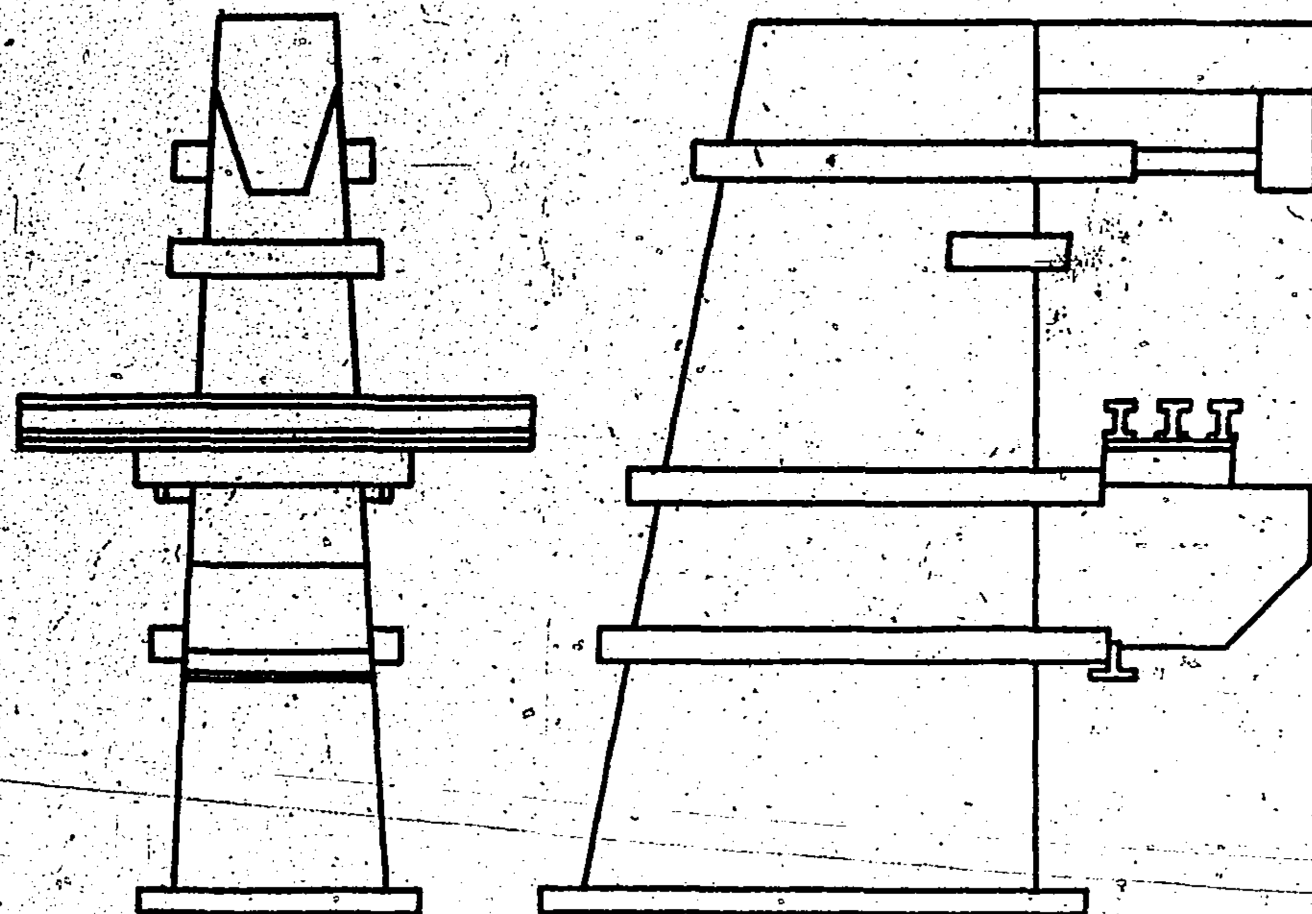


Figure 4.4 Schematic of CIRP Model Milling Machine.

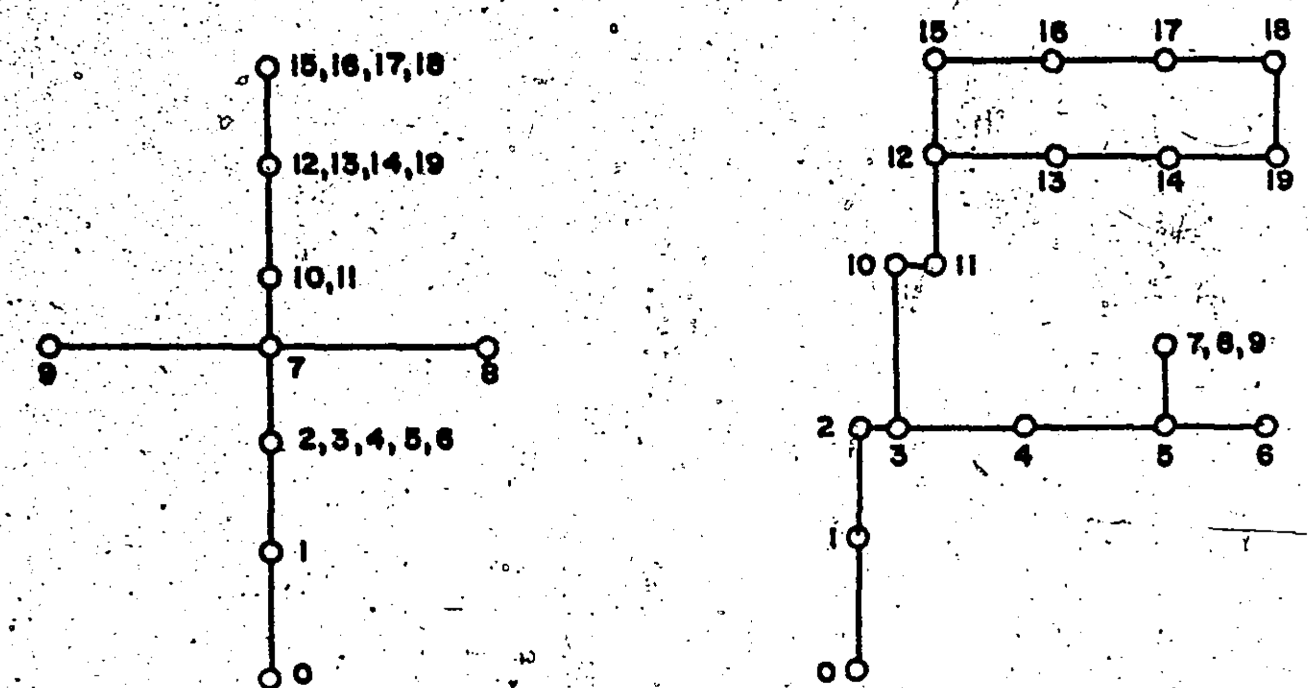


Figure 4.5 Finite Element Model of CIRP Milling Machine.

view. Sutherland's paper described a two cursor (pen) system for the simultaneous digitizing of two views of a three-dimensional object.

Using the EDIT system the user must overlay two views of his preliminary design with a nodal representation for a finite element model. These views are then digitized in sequence with the nodal coordinates being recorded in the same order in each of the views. The relative location of the views to each other is immaterial, as the user is free to record a coordinate origin for each view.

Since the location and the number of structural nodes are arbitrary decisions on the part of the designer, the digitizing process may involve "think" time and best be conducted off-line using one of the stand-alone terminal programs. Each data request made by the program can be satisfied by real time entry via the most appropriate peripheral functioning in an "active" state; or via the most appropriate peripheral functioning in the deferred state through the terminal's cassette system; or from previously transmitted files stored at the central site on a disc.

Section One is completed by the generation of a connectivity matrix that designates which nodes are interconnected by elements. This is achieved in the normal manner by assigning each element a number and associated with the element number are the numbers of the nodes attached to that element. In an interactive environment this can be a time consuming and tedious data entry task. This file is most suitable for local storage and subsequent transmission upon request. Modifications to this file, as well as any other terminal file, can be implemented using the terminal's editing facility.

At the completion of section one the nodal coordinate data and the nodal connectivity data are stored on a central site disc file.

(c) Section Two - Geometry Display

Section two of the synthesis program generates a graphical verification of the input geometry. This is an important aspect of any structural synthesis or analysis system. The cost to generate a solution for a structure that has been coded incorrectly can be quite high. This section of the program employs the terminal support software and the graphics package to enable the designer to specify the point of observation in three dimensions relative to the structural envelope. Figures 4.6 and 4.7 are photographs from the CRT of a perspective presentation of the finite element model for the milling machine of Figure 4.8. The user can dispose the display file to any of the terminal's graphics devices.

(d) Section Three - Load Vector Definition

The definition of the load vector is another potentially tedious task for the structural designer. For the same reasons given in reference to the nodal connectivity file, the file of nodal loads is suited to remote storage and editing for the development of a particular machine

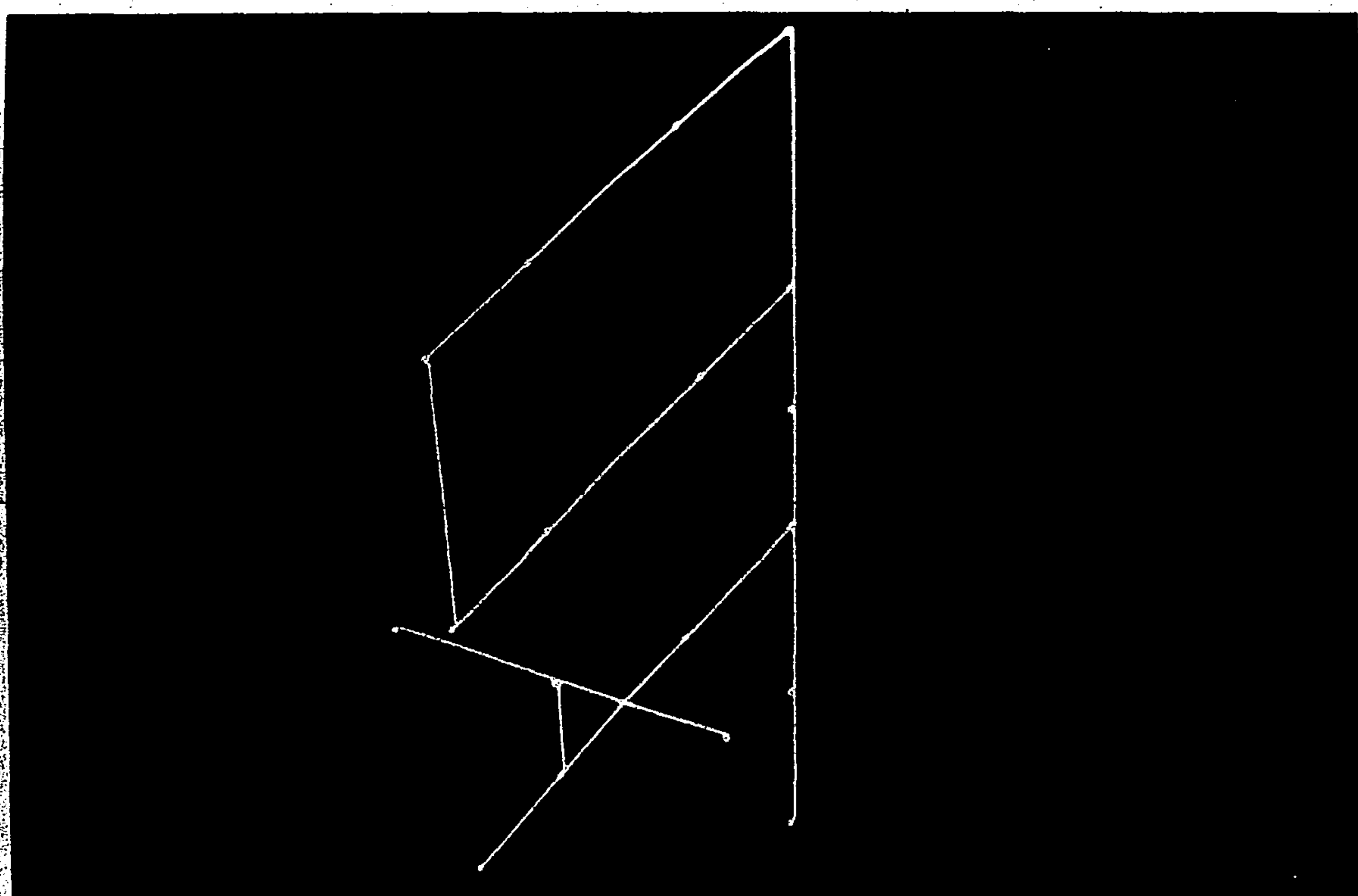


Figure 4.6 Perspective View of a Finite Element Model of a Milling Machine Photographed from CRT Display.

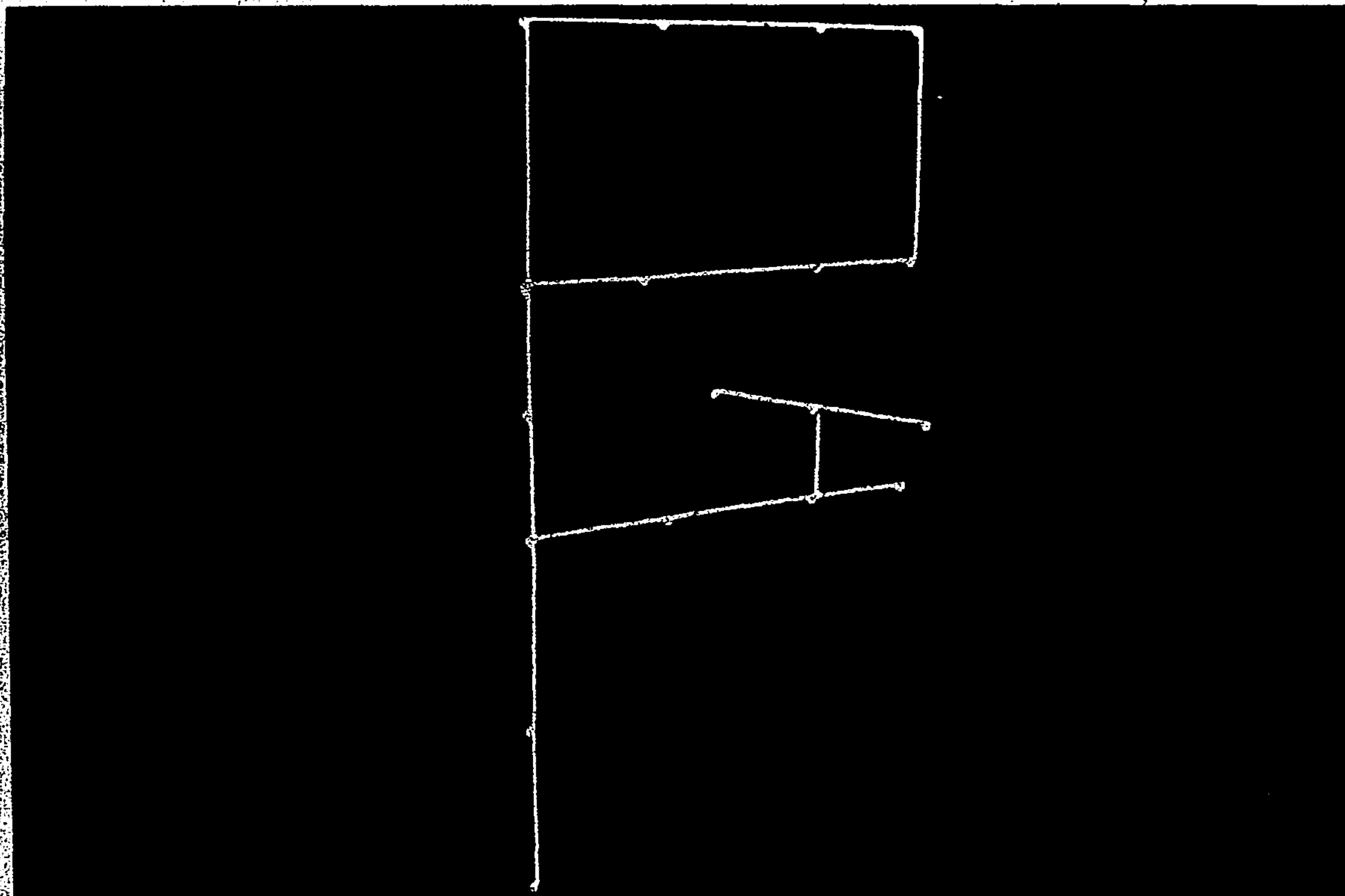


Figure 4.7 Perspective View of a Finite Element Model of a Milling Machine Photographed from CRT Display.

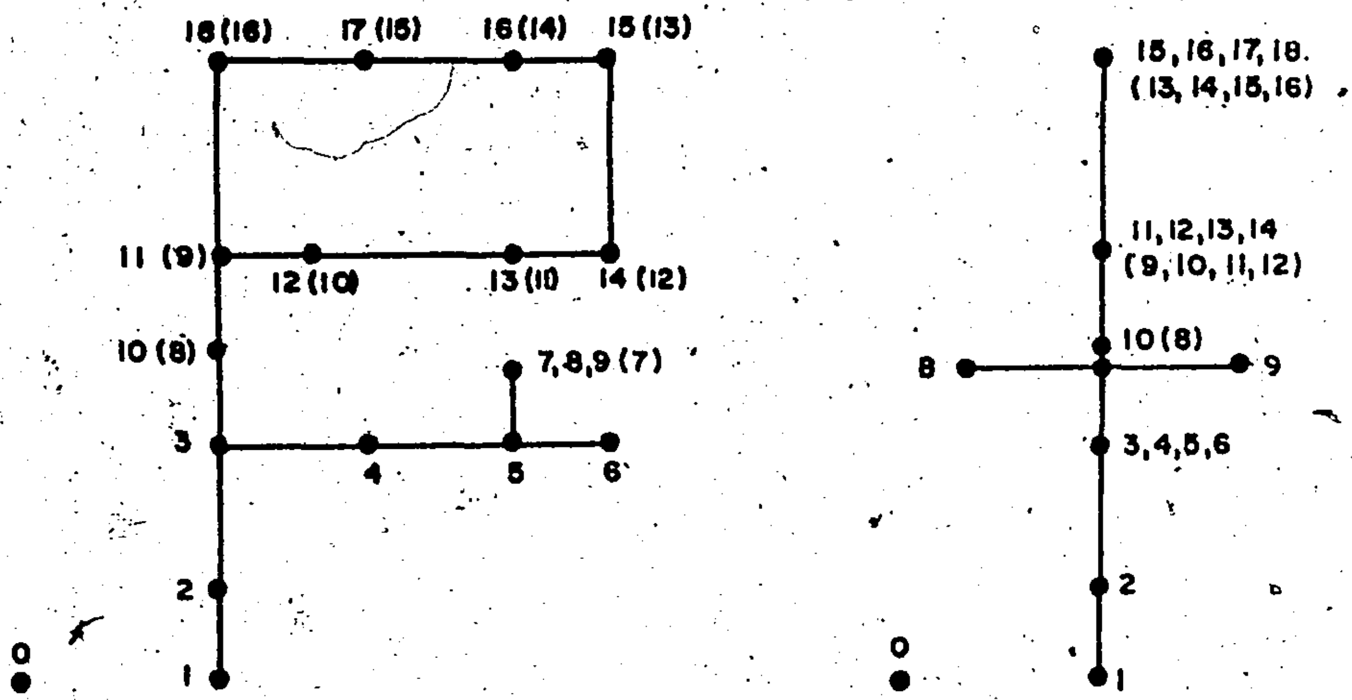
structure. A beneficial addition to the system would be a graphical presentation of the load vector superimposed on the structural geometry. This feature has not been implemented in the current version of the software system.

(c) <u>Section Four - Element Properties</u>

It is necessary for the user to define the modulus of elasticity for the structural material, the material density and the area moment for each beam type structural element. This input requirement must be satisfied for both the analysis system as well as the synthesis system. In the latter case the element properties are used as starting points in the automated search procedure. Again this can be a tedious task for a complex structure being designed in an interactive computer environment. A technique to relieve the designer of this task and allow him to draw from a heuristically constructed data base of standard element forms was investigated. This approach was hindered by the lack of a useable data-base management system. The Query-Update system developed by CONTROL DATA Corporation available on the McMaster CDC 6400 proved to be computationally overwhelming. This reduced its application to the generation of a file of candidate element properties which had to be created independently of the central synthesis system. This complication greatly reduced the desirability of the approach. The implementation of automated data retrieval was further complicated by the severe restrictions placed on the use of the

Query-Update system by the McMaster Computer Center. Section four of this program has been written to accommodate this type of data retrieval system in the event that the computational problems are resolved in the future. The element property file can be stored and accessed from a local terminal file, input on-line via the teletype or retrieved from a central site disc file. The common use of welded sections in the structure of machine tools complicates the definition of the area moments of inertia for these irregular sections. The digitizer can be used in association with a software supported program to generate approximate values of the moments of inertia for these sections. A future addition to the program should include the ability to display the resulting section and associated properties at the terminal using the graphics capability.

The finite element model employed to test the effectiveness of this software system is shown in Figure 4.8. This model corresponds exactly to the previously given CIRP structure except that the two infinitely stiff members that comprise the "knee" sections of the vertical column were removed. This reduced the number of structural nodes to 18 with a total of 18 members. The test load case was a load of 0.2 kips acting vertically on node 13 (axially on the cutter) and in the opposing force acting down on node 7 (table). Since the loads were in the plane of the structure the analysis was not affected by the elements that comprise the table portion of the machine. This had the effect of decreasing the structure to 16 members and 48 degrees of freedom.



MILLING MACHINE — MAC I

Figure 4.8 Finite Element Model of the Test Structure. Node Numbers in Brackets Indicate Plane Frame Model for In-Plane Load Case.

The element properties used for the test structures are given in Table 4.2. The element lengths were computed from the nodal coordinate date

TABLE 4.2 STRUCTURAL PROPERTIES

MEMBER	AREA (in^2)	I_z (in^4)
1 (1,2)	32.62	485.31
2 (2,3)	32.62	485.31
3 (3,4)	18.4	182.35
4 (4,5)	18.4	182.35
5 (5,6)	18.4	182.35
6 (5,7)	27.7	252.26
7 (3,8)	23.26	144.63
8 (8,9)	23.26	144.63
9 (9,10)	27.7	252.26
10 (10,11)	1.1	0.09
11 (11,12)	1.1	0.09
12 (12,13)	5.3	24.99
13 (13,14)	5.67	11.34
14 (14,15)	5.67	11.34
15 (15,16)	27.7	252.26
16 (16,9)	23.26	144.63

(f) Section Five - Analysis

It has not been the intention of this work to develop a sophisticated finite element program, but rather to integrate this capability into a larger system that encompasses all aspects of the design process. There are a number of finite element programs available both commercially and in the public domain that represent many man-years of programming effort. The software system developed in this work employs a relatively simple finite element program that possessed a sufficient number of features to make it feasible to apply it to a real structural problem and to test the merit of an interactive systems approach. Some of the features included in the program are as follows.

1. Since the stiffness matrix assembled by the structure is symmetrical and banded, the program only generates and stores as a linear array those elements contained in the lower half of the matrix band width.

2. The stiffness matrix is inverted (producing a flexibility matrix) using the Choleski decomposition method (73). The resulting flexibility matrix is stored on a disc and can be employed to evaluate the structural response for a number of load cases without incurring the computational expense of matrix inversion for each program iteration.

3. The program is presently written to handle only beam type elements. However, since the program is contained wholly within one FORTRAN OVERLAY it is a simple task to append the capacity to handle additional element types.

4. The program has the capacity to handle springs at the foundation supports as well as members bearing distributed loads.

5. The relatively small core size available to an interactive user on McMaster's CDC 6400 made it prohibitive to expand the program to handle more than 1000 degrees of freedom.

6. A further limitation is that at the present time the program does not compute the thermal stiffness matrix.

A future addition to the program should implement an automated procedure to renumber the nodes to minimize the stiffness matrix band width. In addition, when functioning with more core and shorter "roll-out" times, the capability of the analysis portion of the program should be expanded to handle different elements and more degrees of freedom.

The results for the test structure are given in Table 4.3 (computed member loads) and Table 4.4 (nodal deflections). This analysis consumed 3.5 seconds of central processor time on the CDC 6400.

It should be noted that in terms of machine structures the CIRP model is a relatively simple structure. The system has the capability to handle more complex systems in terms of core capacity. However, when the host computer is used to support a low priority timesharing system in addition to a high priority batch system, the roll out times for testing a system can become excessive. For this reason, the CIRP structure proved to be an ideal test model.

TABLE 4.3 COMPUTED MEMBER LOADING FOR A .2 kip LOAD ACTING AT NODES 7 AND 11

MEMBER	AXIAL LOAD (kips)	SHEAR (kips)	M_1 (kip ft)	M_2 (kip ft)
1				
2				
3		.20	-.3195	-.1699
4		.20	-.1699	-.0013
5				
6	-.20	.0033	-.0013	
7	.20		.3195	.3195
8	.20		.3195	.3195
9	.0485	-0.06	.0666	.0282
10	.0485	-0.063	.0282	.0340
11	.0985	.1370	.0340	.0441
12	-.1370	.0485	.0441	.0924
13	-.0485	-.1370	.0924	.0151
14	-.0485	-.1370	.0151	.1065
15	-.0485	-.1370	-.1065	.2047
16	.1370	-.0485	-.2047	.2530

TABLE 4.4 NODAL DEFORMATIONS

NODE	X-defl. (ft.) x 10^{-5}	Y-defl (ft.) x 10^{-5}	ROTATION (radians x 10^{-5}
1	0.0	0.0	0.0
2	0.0	0.0	0.0
3	0.0	0.0	0.0
4	0.0	-0.2	-0.5
5	0.0	-0.7	-0.7
6	0.0	-1.1	-0.7
7	0.3	-0.7	-0.7
8	-0.1	0.0	0.5
9	-0.5	0.0	1.1
10	-0.5	0.7	1.1
11	-0.4	21.3	14.2
12	-0.3	6.3	1.2
13	-2.0	6.2	2.5
14	-2.0	4.3	3.8
15	-2.0	1.4	2.0
16	-2.0	0.0	1.8

(g) Section Seven - Display of Results

In this section of the program, the user can selectively print or suppress the tables of computed results at the Teletype as well as plot out the structure geometry with the structural deformation superimposed with an enlarged scale. Figure 4.9 is a reproduction of the plot produced for the

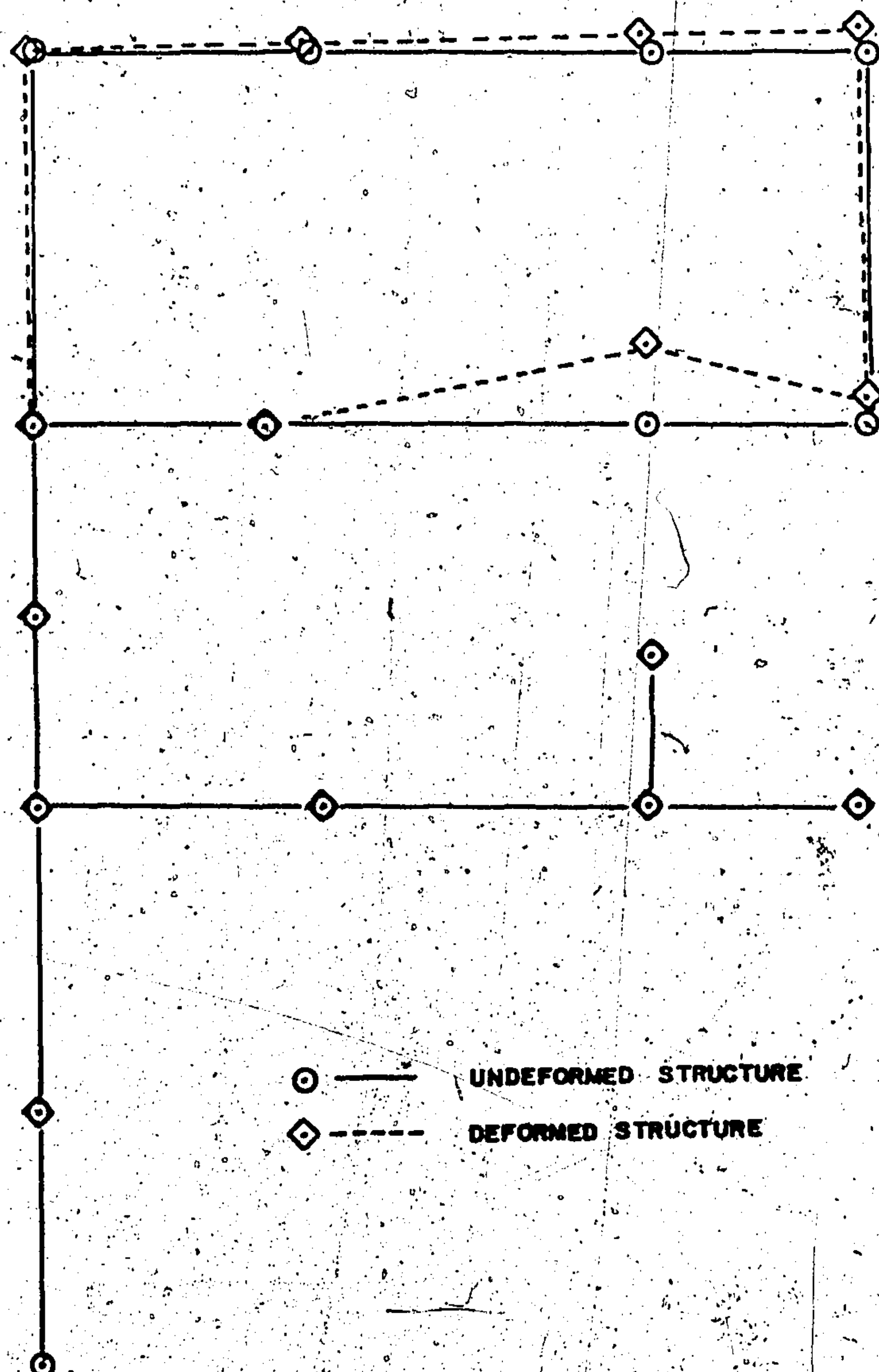


Figure 4.9 Reproduction of Display of Deformed Structure Plotted on Flat-Bed Plotter.

test structure loaded with the test load of 0.2 kips acting at nodes 7 and 11 as produced by the EDIT terminal's flat-bed plotter. The original structure was plotted in black ink while the deformed structure was plotted in red. The deformations in this figure are enlarged by a factor of 1000. This graphic display of the computed results for the structure could have been assigned equally well to any of the terminal display devices. This form of presentation gives an immediate "feel" for the problem to the designer, enabling him to modify those members that are most likely to contribute to improving the structural response, using a manual iterative approach. Alternately, the designer may iterate with the assistance of a mathematical programming algorithm, the logic for which is contained in section six of the program and is described below.

(h) Section Six -- Optimization

This program section allows the structural designer to allow a mathematical optimization technique to vary a specified subset of element cross-sectional areas to arrive at a feasible solution that required the addition of the least structural material.

The objective function has the form

$$U = \rho \sum_{i=1}^{n} A_i L_i \rightarrow \text{minimum} \qquad (4.6)$$

where: A_i = Cross-sectional area of member i

L_i = Computed length of member i

ρ = Material density

N_e = a subset of the total number of structural elements N

In the current version the designer can describe the required structural performance by applying constraints to the nodal deformations for absolute or relative deformations. Those two constraint types have the following form.

For absolute nodal deflections

$$\phi_j = b_{jk} - |\delta_{jk}| \geq 0 \qquad j = 1,m \quad k = 1,6 \qquad (4.7)$$

where: b_{jk} = specified upper limit or the permissible deflection of node j for degree of freedom k.

δ_{jk} = computed value of the deflections of node j in degree of freedom k. (This value is computed via the finite element package of section five of the synthesis program for each iteration of the design variables).

m = defines a subset of the total number of nodes M.

Note: For the analysis for the in-plane loading of the test case the nodal degrees of freedom were reduced to a total of three. The constraint functions are defined by simple FORTRAN statements in a separate subroutine. Thus the user can easily append additional constraints as required.

The optimization strategy employed has been adopted from the SEEK1 program in reference (56). This is the direct search method of Hooke and Jeeves followed by a random search check. The strategy is an unconstrained search for an optimum of an artificial objective function of the form:

$$U_{art} = U_{act} + 10^{20} \sum_{j=1}^{m} |\phi_j| \tag{4.8}$$

for all $\phi_j < 0$.

A complete description of the program and the relevant parameters is contained in the reference. For the test structure a single absolute constraint was placed on the deflection of node 11 (cutter support). The maximum permissible value was specified as 0.00006 ft. The members allowed to vary in cross-section were members 10 and 11 which are bounded by nodes (10, 11) and (11, 12) respectively. The convergence criteria of minimum step-length was arbitrarily established for the test structure as:

$$(0.0675)\ A_{s_i}$$

where: A_{s_i} = input starting value for the cross-sectional area of member for each member allowed to enter the search procedure.

For machine tool structure design the influence of the deflection constraints on the response surface generated by the equation (4.8) normally would preclude the possibility of a violation of a failure constraint. In the event that a failure constraint is violated, there is an additional penalty function applied to the objective function.

The failure constraints are only applied when a member fails that would lead to a structural failure. The inclusion of a load free redundant member in the subset of members being allowed to vary would result in the deletion of that member from the structure.

In this automated search strategy, which employs a finite element analysis at each design point, there is a problem related to the explicit expression of the area moment of inertia for the cross-section in terms of the design variables that govern the members' cross-sectional area. Provision has been made in the program to employ a file of member types that are in common usage. The test structure was allowed only circular cross sectional members.

Under the above criteria, convergence for the test structure was achieved in fewer than five iterations producing the results given in Table 4.5 and Table 4.6. This synthesis required 7.5 seconds of central processor time on the CDC 6400.

At the present time there is only the logic for one search strategy in the code for section six. A future version of the structural synthesis program should incorporate the capability of allowing the designer to override the default option by requesting a different strategy. The structure of

TABLE 4.5 COMPUTED NODAL DEFLECTIONS

Node	Y-Deflection in (ft.) X 10^{-5}
1	0.0
2	0.0
3	0.0
4	-0.2
5	-0.7
6	-1.1
7	- .7
8	0.0
9	0.0
10	0.7
11	5.8
12	5.8
13	5.7
14	4.0
15	1.4
16	0.0

TABLE 4.6 COMPUTED MEMBER PROPERTIES DETERMINED BY SECTION SIX

Member (Node)	Area in^2	I_z (in)^4
1 (1,2)	32.67	485.31
2 (2,3)	32.62	485.31
3 (3,4)	18.40	182.35
4 (4,5)	18.40	182.35
5 (5,6)	18.40	187.35
6 (5,7)	27.70	252.26
7 (3,8)	23.26	144.63
8 (8,9)	23.26	144.63
9 (9,10)	27.70	252.26
10 (10,11)	3.54	0.99
11 (11,12)	3.15	0.76
12 (12,13)	5.3	24.99
13 (13,14)	5.67	11.34
14 (14,15)	5.67	11.34
15 (15,16)	27.70	252.26
16 (16,9)	23.26	149.63

section six of the structural synthesis system has been written to allow this capability to be easily installed.

(i) Section Eight - Animation of the Dynamic Response

Section eight of the structural synthesis system is used to animate the dynamic response of the structure for each of the natural modes of vibration. The eigenvalues and eigenvectors obtained in the solution of Equation (4.4) can be employed in conjunction with Equation (4.3) to simulate the dynamic response of the structure. Our current system does not include a dynamic analysis program. The eigenvalues and eigenvectors may be calculated by an external package (in this case STARDYNE), and these are user input to the system. The normal output procedure is to use a conventional plotter to produce a series of rotated views showing (at an enlarged scale) the structure in its most extreme position. Figure 4.10 shows this procedure in a series of views of the CIRP model milling machine as analyzed and plotted on a CDC 6600 computer with the STARDYNE finite element package. This static representation of a dynamic process fails to completely convey a true indication of the actual deformation process. These extreme plots do not indicate the interaction between elements of the structure as it oscillates from one extreme position to another. In contrast to this unsatisfactory static output, our system provides an animated dynamic display and serves an analogous function for the visualization of the dynamic aspects of structural design as the graphic display of section

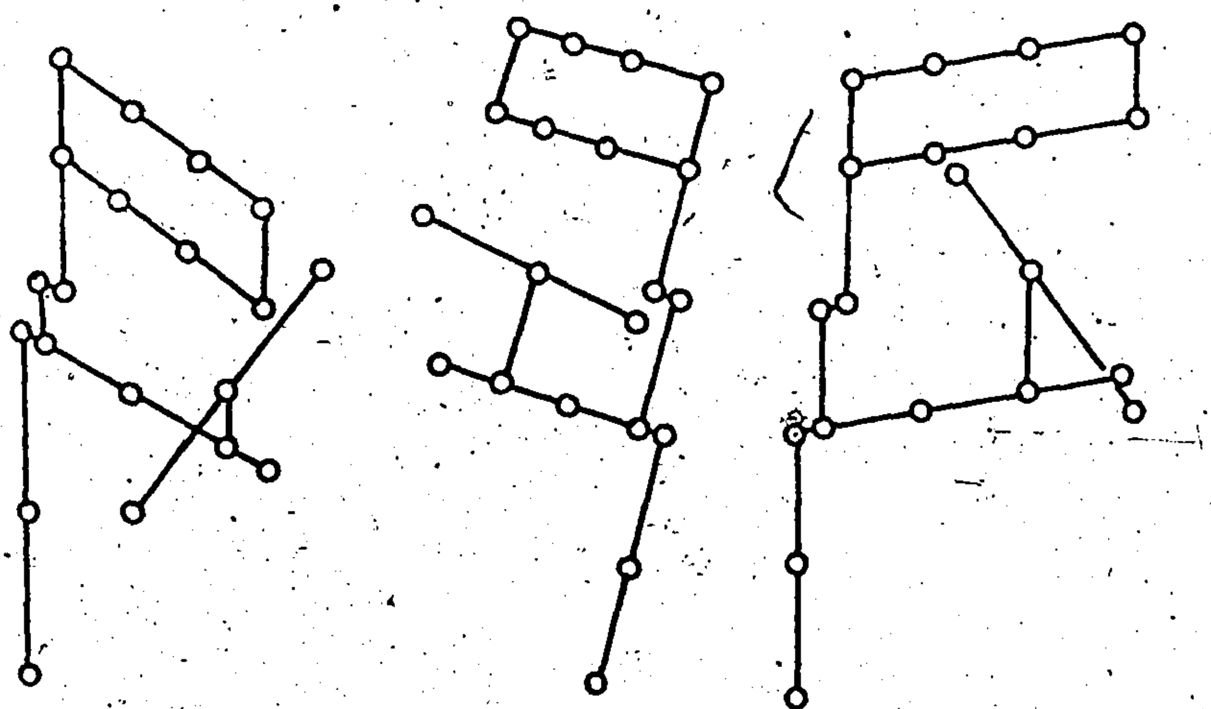


Figure 4. 10 Rotated extreme views of CIRP structure as plotted by STARDYNE.

seven of the program did for the static aspects of design.

It is very difficult to quantify the information available to the engineer/designer resulting from an animated display of a time dependent process. It is equally difficult to demonstrate the qualitative advantage of animation in the static medium of a paper.

The animation of the structure is not conducted at the terminal in real time. To achieve this would require a refresh type display CRT and a fast data channel between the data interpolation processor and the display controller. The animation is accomplished by producing a 16 mm movie by generating each frame of the movie on the EDIT terminal's storage display CRT and photographing it with a Bolex movie camera. The camera shutter is driven by the terminal and the exposure times can be controlled through a parameter in the local terminal's operating system. The interface was designed by the author and is described in Appendix J. Similar to the display portions of sections two and seven of the program, the user can control the perspective view of the structure by inputting the coordinate locations of the point of observation. After viewing a representative frame from the movie, the user can transmit the balance of the graphical data to the terminal.

The amount of information that must be transmitted for the complete production of a movie is enormous for even relatively simple figures. For example, a one minute movie, with 1000 transmitted data characters required per frame, would require the total transmission of 1.44×10^6 characters. At 300 baud this would require 13 hours to transmit the data. This combined with a 1.5 second exposure time for each frame is a prohibitively long time to expect a host system to remain stable. For the structural system the problem is alleviated by the cyclic nature of the system response. Even for this reduced problem the most advantageous mode of employment of the system is to transmit the required display data to the local processor storage facility at relatively high data rates and then employ the terminal in a stand-alone mode to generate the movie. This has proved to produce a reasonably good quality

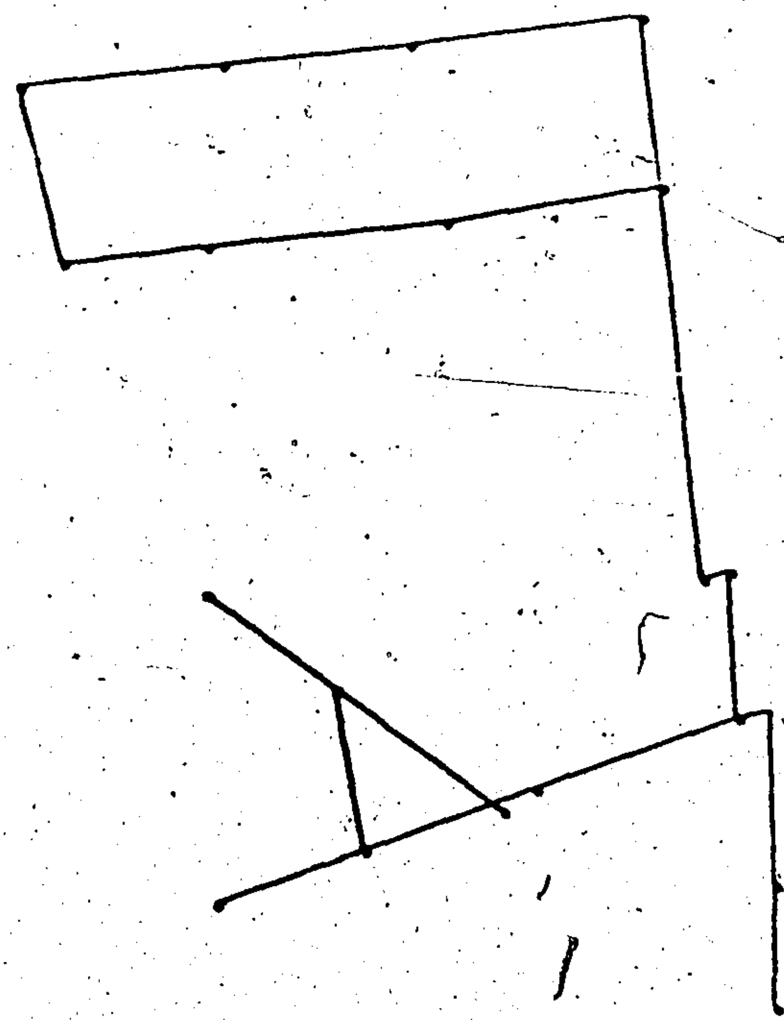


Figure 4.11   Frame from animated movie.

movie that is technically accurate at a very low hardware and software cost. Several cost saving methods of interpolation should be included in a future version of this system. One is repetitive filming of frames. This produces an aesthetic film but no longer retains the precise relationship to the mathematical model. Another method is local linear interpolation of intermediate film frames.

Figure 4.11 is a print of a frame from a movie made that depicted the test structure vibrating in the first natural mode (184 Hz). The cycle time used in the movie was 12 seconds producing a reduction factor of $\frac{1}{2208}$ of the actual period of oscillation. This required the generation of 288 independent frames for the complete cycle.

Although a real-time animation display would be more appealing in the design process, the system described above provided a very inexpensive technique by which the contribution to the design process of displaying time dependent functions in animation could be evaluated.

4.2 Summary

The program described in this chapter, although skeletal in some aspects, has served to demonstrate the enhancement of the design process that occurs when the three elements, designer, software and hardware are integrated into a cohesive system. The software has been developed as a system comprised of independently executable subsections that interchange data files via an external device. This has provided the user with the flexibility to execute any subsection of the program during one terminal session, after which the information that has been created is automatically stored by the system. The terminal features have been exploited to reduce the normal hindrances that occur when attempting to implement a large scale program for structural synthesis in an interactive computer environment. A dialogue has been structured to provide an efficient system that can be utilized effectively by both experienced and inexperienced users.

CHAPTER 5

BOTTLE - A COMPUTER-AIDED MANUFACTURING SYSTEM TO AUTOMATE BOTTLE MOLD PRODUCTION

5.0 General

In Chapter 2 of this thesis, computer-aided manufacturing has been defined as the automation of the functions in the design, development, and manufacture of a product regardless of the number of products produced or the scale of the production operation. In this chapter a simple and minimal system is described which has been developed to fulfill a real industrial need in the manufacture of glass bottle molds. It serves as an illustration of how the EDIT system may also be used in computer-aided manufacturing. In this application the EDIT terminal is employed as the central element in a three stage system that allocates data processing and data management tasks to the three system processors. Figure 5.1 is a block diagram showing the relationship between the three levels of processing power and the functions allocated to each level. The link between the EDIT terminal and the process control computer (a Hewlett Packard 2100A, 16k mini-computer) has just recently been completed. The logic of using this system, for computer-aided manufacture was developed in this work by simulating this connection using an off-line paper tape link to a hardwired GE Mark Century Control on an American Hustler Turning Center.

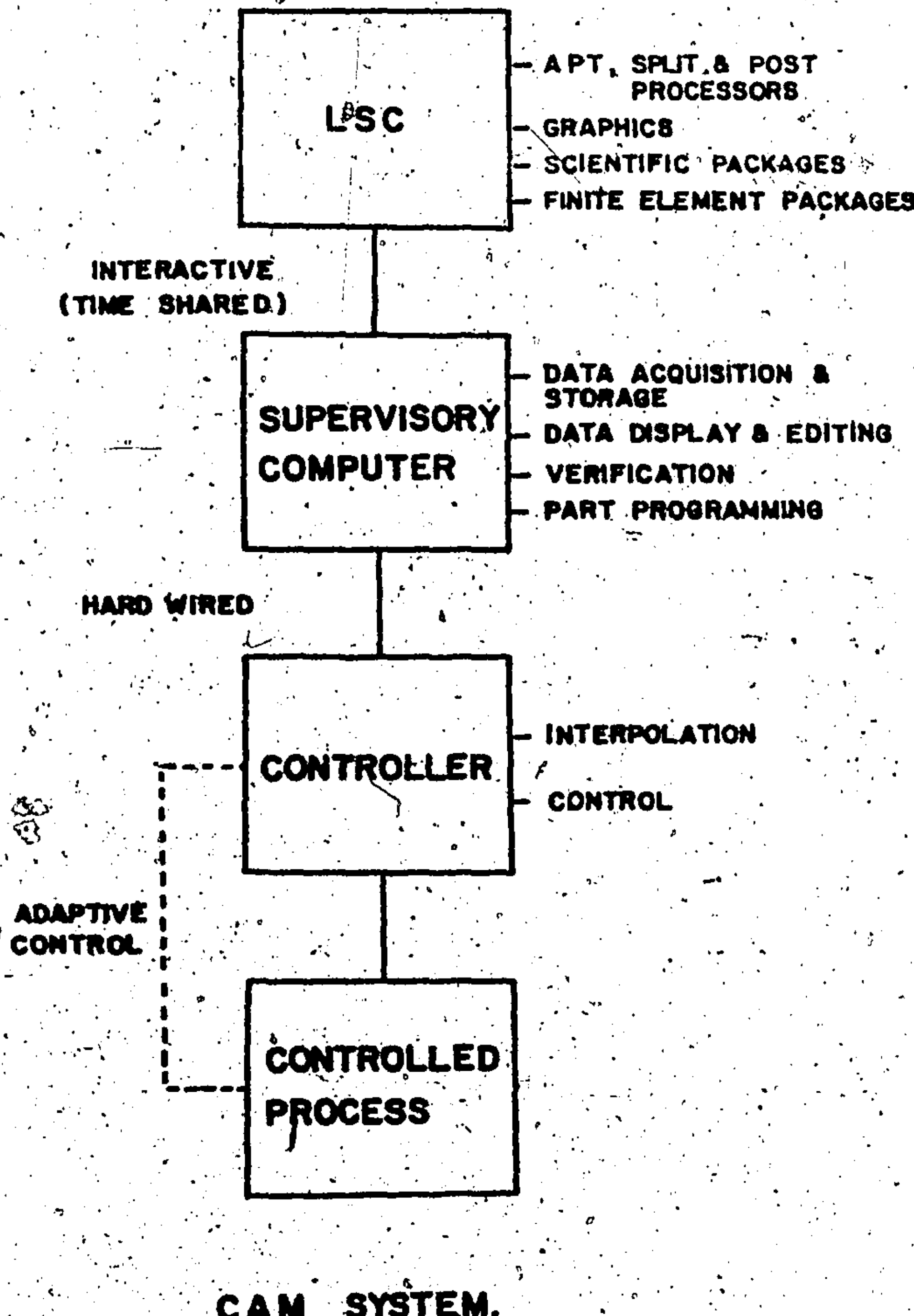

LSC
APT, SPLIT & POST PROCESSORS
GRAPHICS
SCIENTIFIC PACKAGES
FINITE ELEMENT PACKAGES
INTERACTIVE (TIME SHARED)
SUPERVISORY COMPUTER
DATA ACQUISITION & STORAGE
DATA DISPLAY & EDITING
VERIFICATION
PART PROGRAMMING
HARD WIRED
CONTROLLER
INTERPOLATION
CONTROL
ADAPTIVE CONTROL
CONTROLLED PROCESS
CAM SYSTEM.

Figure 5.1

5.1 Present Practice

The glass manufacturing industry has provided a interesting case study for the examination of the effects on all aspects of the manufacturing process that occur as a result of the introduction of a new technology into one stage of the production system. In this case, the mold design process and the present production practices have been extensively influenced by the introduction of numerically controlled turning centers into the mold manufacturing proces.

In the present system, the design of a new bottle follows generally the following steps.

1. An industrial artist in communication with the customer arrives at an aesthetic sketch of a proposed bottle design. The artist then prepares a profile drawing from this sketch.

2. The profile sketch is translated into a detailed drawing similar to the drawing shown in Figure 5.2. This is a difficult task as the draftsman, in the preparation of the detailed drawing, must attempt to compute the contained volume and resize the drawing to meet the specified values. This is a task that is often at variance with the retention of the original aesthetic requirements of the design. There is a tendency to introduce straight lines into the profile where possible. In addition the draftsman must meet any dimensional constraints imposed on the bottle by the customer's transport mechanisms.

3. A wooden prototype is produced by a pattern maker from the detailed drawing. The volume of the prototype is

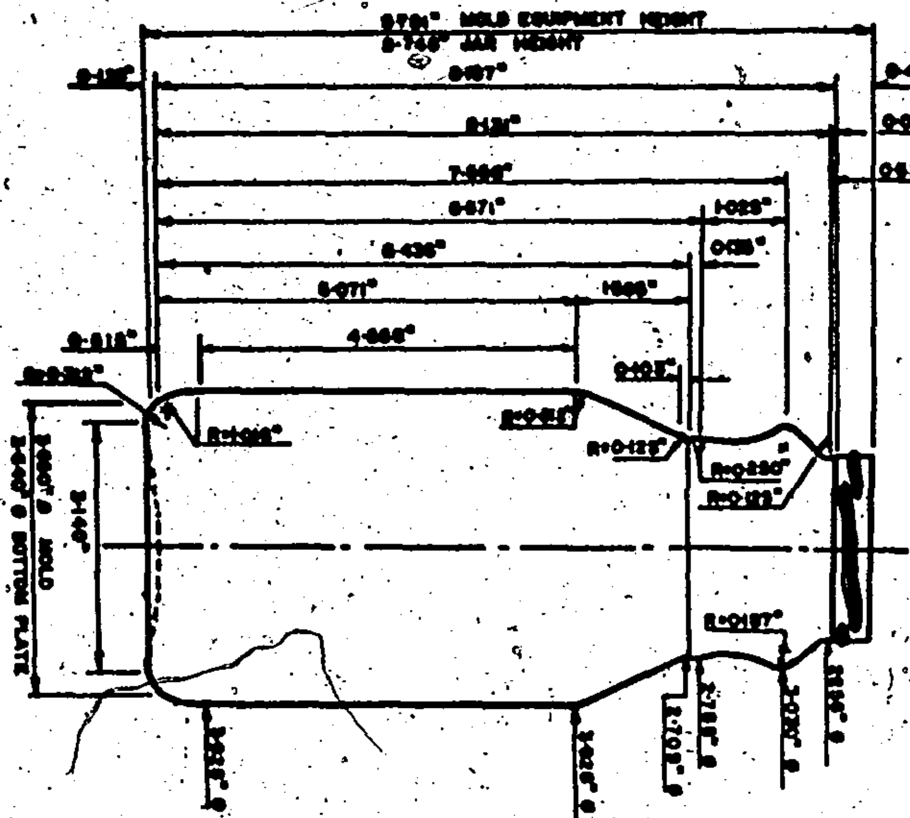


Figure 5.2 Detailed Mold Drawing.

then found by immersion in water. Following this the pattern is modified until the correct volume is attained by trial. At this stage the expensive detailed drawing becomes redundant. The incompatibility between drawing and prototype introduces a manufacturing problem between competing glass companies. When a bottle contract is awarded that exceeds the productive capacity of one supplier, the order is sub-contracted to the competition. However, mold development expenses are only recovered by the original contract recipient. The sub-contractor is provided with the above detailed drawing. One firm confided that they resolved the problem by purchasing the competitor's product in the market place and used this as a prototype model.

4. The wooden pattern is then used with a tracer-controlled cutter to produce a steel pattern.

5. The steel pattern is used with a tracer controlled lathe to cut the mold cavity.

After completion of step three, the wooden pattern is checked by the original designer against the original design concept. A rejection at this stage precipitates an extensive delay in the production of the mold. The move to metric sizes (requiring the conversion of a great number of existing bottle molds to new sizes, while again maintaining the original design concept and meeting dimensional constraints) has added an additional incentive to adopt automated methods for mold manufacturing.

5.2 Proposed System

The introduction of numerical control technology into the mold production process has made it possible to employ automated techniques to the manufacture of these molds. The EDIT terminal hardware, the terminal support software and the graphics package have provided an inexpensive system to integrate the design process directly into the production process.

The program to implement this process has been divided into two major divisions. The first stage involves the preparation of an APT compatible input file. The second stage embodies the verification of the cutter information generated by APT and the preparation of the data file for the machine tool controller.

The first step in the first stage of the process is the determination of the bottle specification. Here the user must input the required volume, the fill space volume, the bottle height, the design weight (amount of glass to be used in each bottle), as well as the specified neck diameter. The user has the option of working in either British or metric units. These specifications are supplied to the terminal using a program conversational mode that has been structured about the same strategy as the structural synthesis system. Thus the user can control the order of program execution, solicit an explanation of a system query or terminate execution, by entering one of the COMMANDS defined in Table 4.1.

The second step is the reduction of the profile of the bottle produced by the industrial artist to digital information. This is accomplished by tracing out the curve with the EDIT terminal digitizer with the digitizer in either the active or deferred state (i.e., on-line to the host computer or off-line via the terminal cassette tape system). Once the profile has been defined the coordinate data (incremented by 0.010 inches in the axial direction of the bottle) is numerically integrated to calculate the bottle volume for the profile as digitized. This value is returned to the designer to provide him with a "feel" for the direction that the volume modification must follow (i.e., the value has to be increased 10% or decreased 30%, etc.).

Under section three of the program the user can select one of four volume modification strategies that distribute the radial dimensional variations of the profile along the axis of the bottle to accommodate the required volume modification. This is a unimodal function of the form

$$\text{Min } U = \left| V_r - \frac{\pi}{4} \sum_{i=1}^{N-1} (x_i + x_{i+1} + 2.F.\Delta X)^2 \, \Delta z \right| \qquad (5.1)$$

where: V_r = required volume

$F = 1.0$ for modification strategy 1

$F = (1-e^{-b(i-1)})$ for modification strategy 2

$F = (i-1).\frac{\Delta Z}{L}$ for modification strategy 3

$F = \text{erf}(X)$ for modification strategy 4

The effect of these strategies on the translation or modification of the profile is shown in Figure 5.3.

Convergence was achieved through the application of a simple algorithm for a Fibonnacci Search. The logic of this algorithm is given in Appendix N. The convergence criterion used for the test problems was .02 fluid ounces. Convergence with this simple functional form is very rapid, usually consuming fewer than 10 iterations.

Section four of the bottle design system is the presentation of the computed bottle shape. Similar to the display of the geometry in the structural synthesis program; the user can specify the coordinate position of the point of observation for a perspective view of the bottle. Figure 5.4 and 5.5 are photographs from the EDIT graphics CRT of two different bottle shapes designed with this system. A simple hidden line algorithm has been employed to remove the portions of the base (or top depending on point of observation) that are not visible. Several forms of presentation were tried to create a realistic appearance for the bottle, the simple form shown in these figures was found to have the greatest appeal. This perspective presentation gives a rapid review of the effects that the volume modification strategies will have on the finished bottle's appearance. If the user is not satisfied he can simply re-enter the program at any point and reset one of the design parameters (bottle length or shape strategy, etc.). The bottle display could have been created on any of the EDIT graphics hardware systems for the

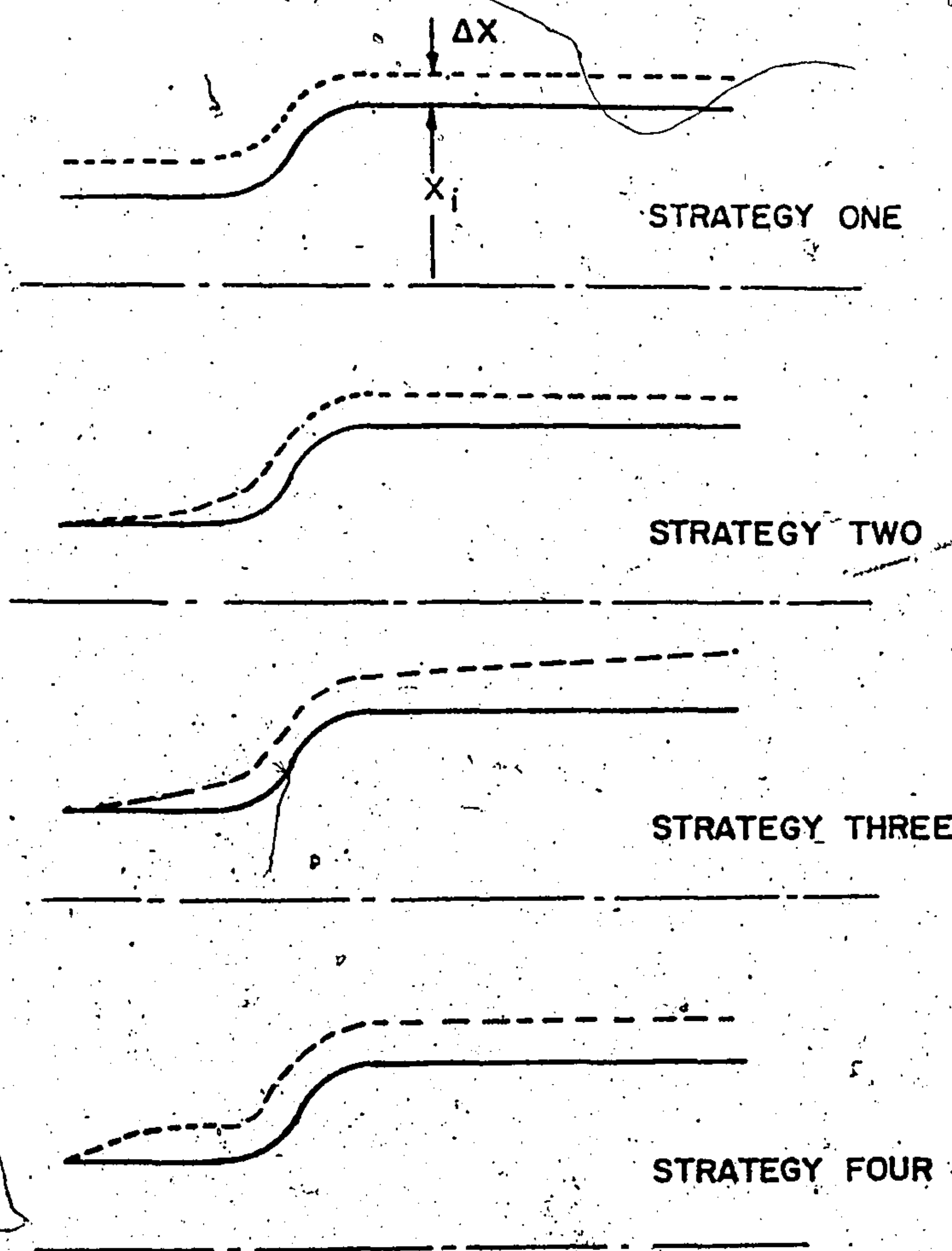
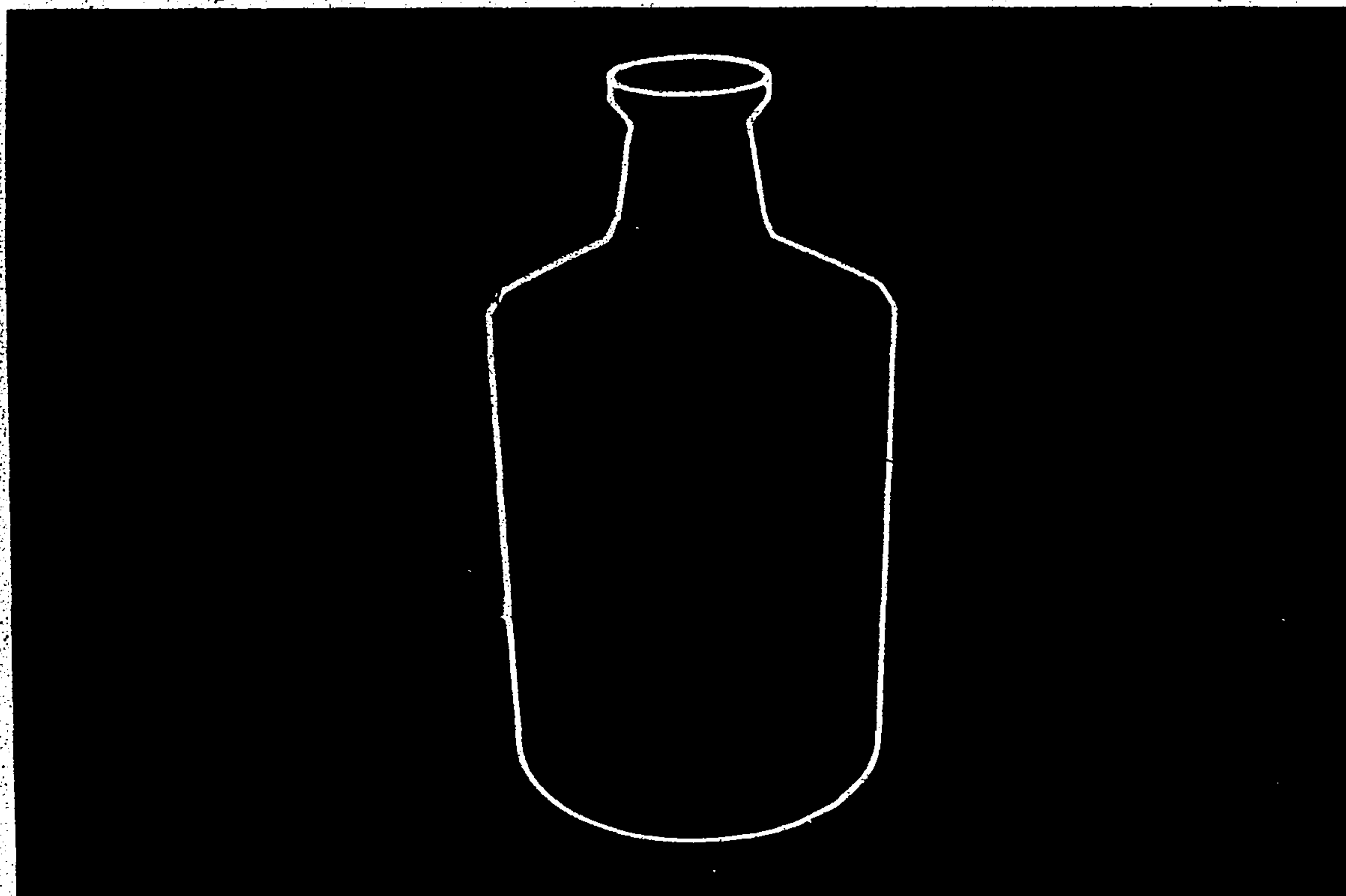


Figure 5.3 Effect of Modification Strategies on Bottle Profile.



Figue 5.4 Computed Bottle Shape Displayed on the CRT.

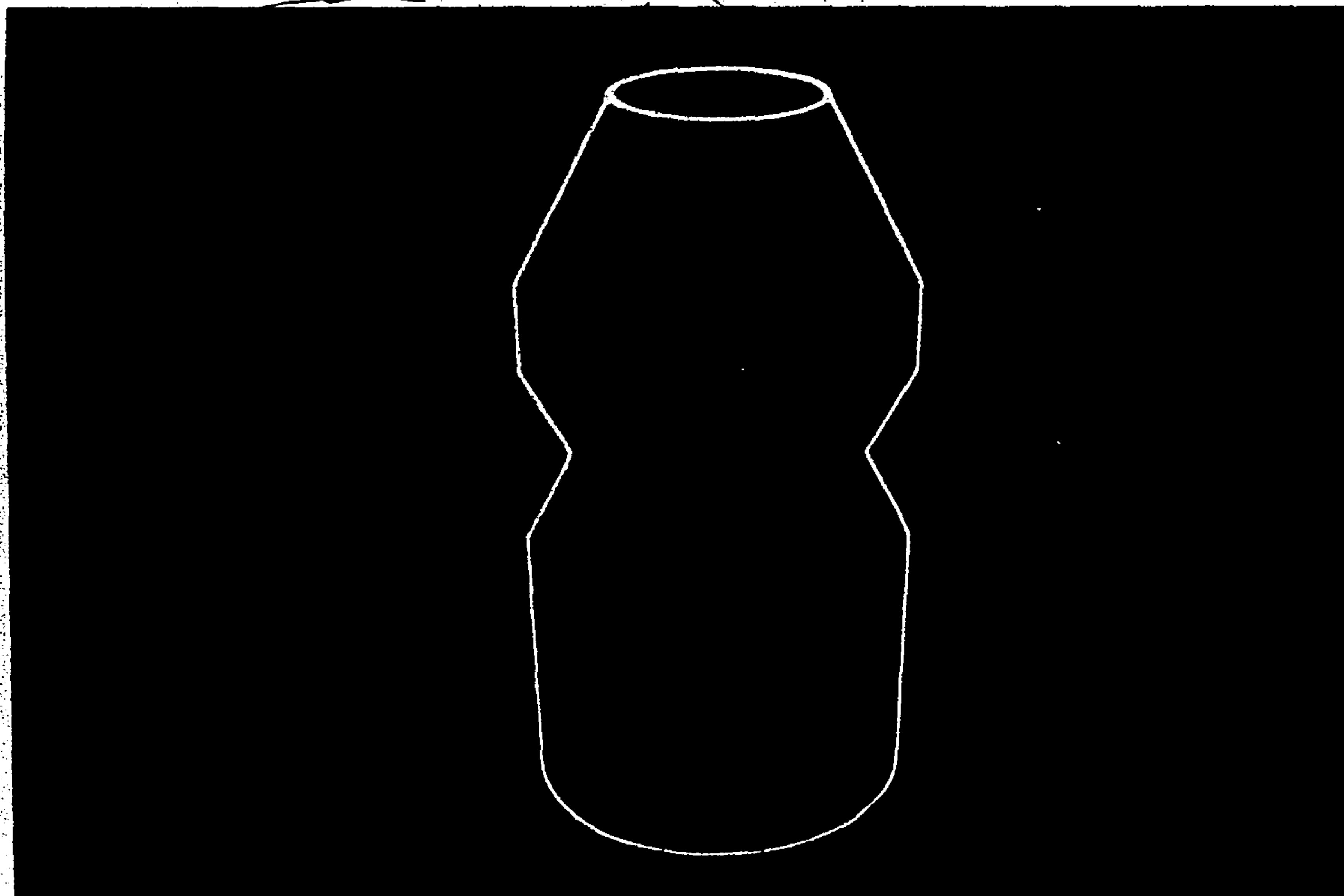


Figure 5.5 Computed Bottle Shape Displayed on the CRT.

creation of a hard copy.

The development of an adequate and inexpensive display device for the real time display of greyscale graphics would contribute to the appreciation of the aesthetic aspects of the design. The perfection of this process may eventually make the production of a prototype bottle redundant.

Part five of the bottle is the creation of an APT input file which can call the APT package to generate a cutter path. This file must be in the APT language. This is a complicated system programming problem to circumvent an inadequacy in the present version of APT (version 3.0). This limitation is APT's inability to read data from external mass storage devices. The approach used in this work was to create the APT roughing and finishing macro in three segments. The first segment contains the file loading and execution directives, the second section contains the finished profile data and the third section contains the logic of the APT macro. The execution of the design program creates the second section of the above APT input file. This file is then merged with the other two segments and the whole file is then loaded as a remote entry batch processing job. The soon to be released APT version 4.0 will be able to read from external data files eliminating this file manipulation problem. Figure 5.6 is a listing of the APT macro used for the production of a prototype bottle using the EDIT terminal system. The macro segments are indicated. The profile data has been interpolated with the APT TABCYL routine to produce a smooth surface.

Section One

```
      BOTTLE MOLD NO1
   /CIRAPI,SL ,CIRCUL,1,4,6,4
  ,ATANGL,CA,CIRCLE,CA,CANON,?,CENTER,CT,CUTTER,D,NOCUT,F,FEDRAT,$
  ,FROM,FR,GO,GODLTA,GF,GOFWD,GL,GOLFT,G?,GORGT,GT,GOTO,IP,INDIRP,$
  IT,INTOF,IV,INDIRV,L,LINE,?Y,MATRIX,MX,NOMORE,O?,OBTAIN,P,POINT,$
  ??P,PARLEL,PL,PLANE,PT,???? ?,R,RADIUS,R?,RAPID,T,TRACUT,$
  TL,TLLFT,TN,TLON,TR,TLRGT,T?,TRANFL,TT,TANTO,V,VECTOR,XL,XLARGE,$
  X?,XSMALL,YL,YLARGE,YS,YSMALL,ZL,ZLARGE,ZS,ZSMALL
  ????
  ????/X,133,U,134
```

Section Two

```
** ......INSERT CONTOUR DATA HERE.........................................
** ?? ??      WITH DIMENSIONS
** ??.
** LST=           NO. OF Y( ) WIDTH DIMENSIONS
** LGTH=          TOTAL LENGTH OF MOLD
** CHK = A LINE DEFINED FOR ROUGHING THE NECK,
**        PARALLEL TO Z AXIS, GREATER THAN X(1) BY
**        2 X THE MAX. CONCAVE DEVIATION
$$ STKDIA =        STOCK DIA OF THE BAR
$$ PASSES =          NO. OF TOOL PASSES EXCLUDING A FINISH PASS

Y(1)=.35
Y(2)=.35
.
.
.
Y(N)=1.
LST=N.
```

Section Three

```
                                    $$ LAST X DIST (END OF MOLD)
LGTH=4.95
CHK=L/?,-1.5,?,-1.5
STKDIA=2.5
PASSES=5

ZINC=LGTH/(LST-1)
TOP=L/?,?,?,1
BOT=L/-LGTH,?,-LGTH,1
A?=?(/ ?,-1, ,?,$
      ,?,-?,?,$
      1,?,?,?

LOOPST
I=1
Z=?
1?) U(I)=P/Z,-X(I),?
I=I+1
Z=Z-ZINC
IF(I-LST)1?,1?,2?
2?) LOOPND
CONTR=TABCYL/NOZ,SPLINE, $
U(1),$
U(2),$
U(3),$
U(4),$
U(5),$
U(6),$
U(7),$
U(8),$
U(9),$
U(1?),$
U(11),$

U(12),$
U(13),$
U(14),$
U(15),$
U(16),$
U(17)
PPRINT.....SET MINUTE MAN 4.? FROM FINISH FACE
PPRINT.....TRACING TOOL, FACE 1, OFFSET 1 REQUIRED
PPRINT.....PART PROTRUDES 7.? FROM CHUCK  (+.?6,-.?)
```

Figure 5.6 APT Roughing and Finishing Macro.

```
[illegible]TN[illegible]G90[illegible]
[illegible]TN [illegible]2G35[illegible]
[illegible]TN [illegible]G92Y[illegible]9257[illegible]
[illegible]TN [illegible]G.4X[illegible]25T.1S35[illegible]
[illegible]TN [illegible]G 1Y Z.1 F[illegible]

[illegible]/[illegible]
[illegible]L/[illegible]
[illegible]F/(PL/[illegible],1,[illegible])
[illegible]T/5
[illegible]/.[illegible]
[illegible]

[illegible]1=MACRO/THK
THI[illegible]/[illegible],THK
[illegible]
[illegible]
[illegible]/TOP
[illegible]
TL,GL/TOP,PAST,CONTR
TL,G[illegible]/CONTR,ON,BOT,I2,ON,CHK,I1
I1)TRANTO/I2
I2)GO/-.2,[illegible]
GO/[illegible],-1,[illegible]
GT/2,-2,0
T[illegible]AD

T/AX

LOOPST
J=1
PAS1=((STKDIA/2)-X(1))/PASSES
PASS= (STKDIA/2)-X(1)-PAS1+.[illegible]4 . [illegible]3 . .4=TOOL RAD + .01 FOR FINISH CUT
I1 )CALL/TURN1,THK=PASS
J=J+1
PASS=PASS-PAS1
IF(J-PASSES)I1[illegible],I1[illegible],I1[illegible]
I1[illegible])OUTTOL/.[illegible]5
CALL/TURN1,THK=.031
LOOPND

INSERTN99771[illegible]
INSERTN998G[illegible]4T[illegible]
INSERTN999M[illegible]

PRINT/3,ALL.
END
FINI
```

Figure 5.6 Continued.

Section six of the program is the verification of the cutter location file or parts program generated by APT or APT with a post processor. The user at the terminal can retrieve the data file produced by APT; where the system extracts the information defining cutter motions, and, with the aid of the terminal support software and graphics package, simulates the cutter actions of the numerically controlled machine tool. The user can examine the whole cutter path or selectively designated portions of the path. This display can be conducted on any of the EDIT terminal's graphics devices and the displayed portion of the cutter path can be displayed at any appropriate scale. If an error is detected, the data file can be edited, corrected and redisplayed at the terminal. Figure 5.7 is a block diagram representing the relationship that exists between the software tasks of the systems processors for both the parts programming and verification processes.

The EDIT terminal's flat-bed plotter is a device particularly suited to the verification of numerical control cutter location files. For parts that require several cuts that are located in close physical proximity to each other, the plotter's multiple pen turret can be used to distinguish each cut with a different colour of ink. An additional important feature is that the operating format of this plotter closely resembles the operating format of a machine tool. Since the plotter's pen (machine tool's cutter) is moved about a table by analogue drives, it produces

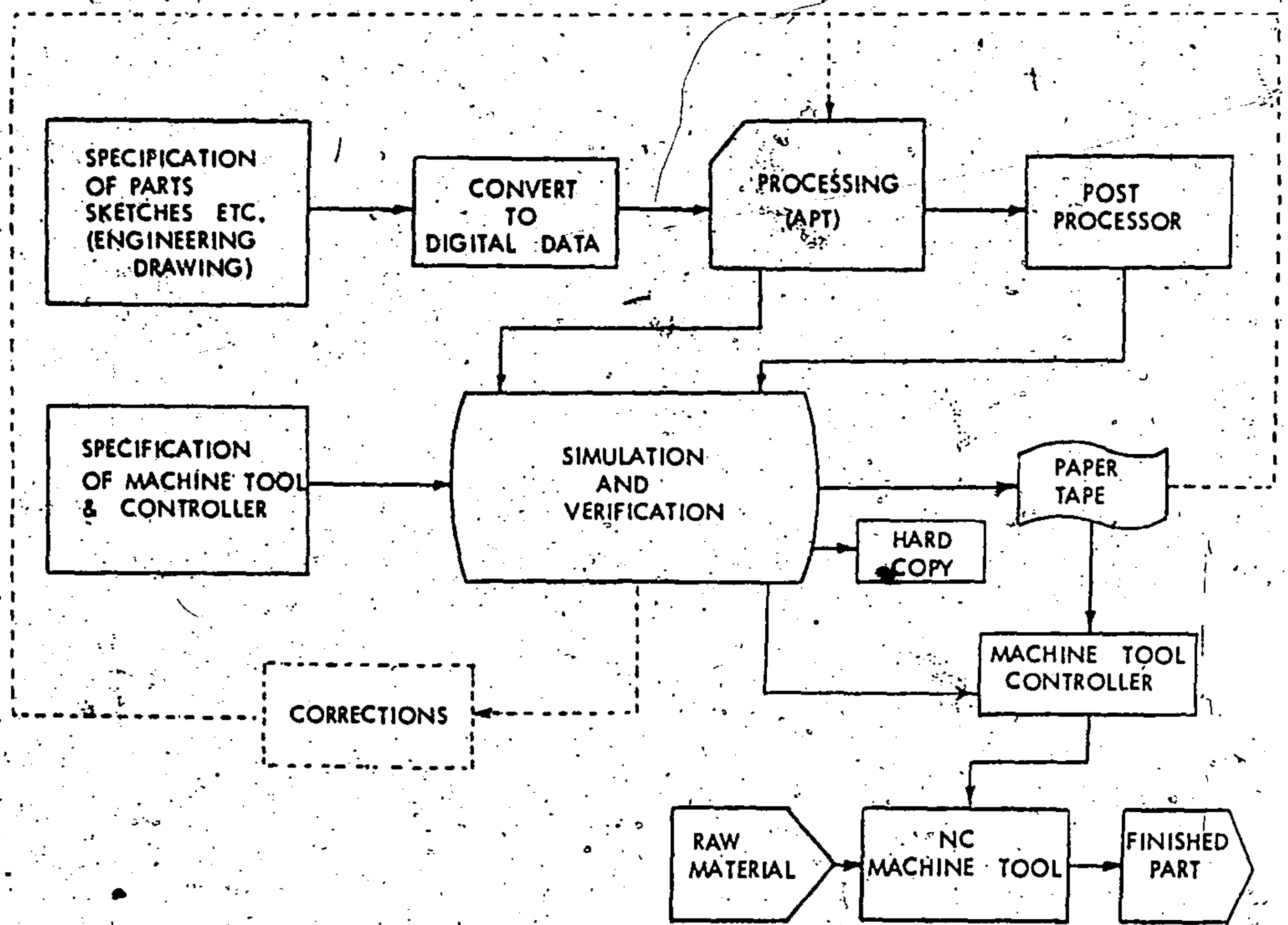


Figure 5.7 Software Hardware Systems for Part Programming and Verification Processes.

the best visual simulation of the actual cutting process.

Once the cutter path file is in a suitable format it can be transmitted to the terminal for storage on the cassette system. When the file is required by the on-line softwired machine tool controller, it can be retrieved directly from the storage medium and transmitted into the core of the controller itself. This retrieval is initiated by a request generated from the controller, when it is set up to begin machining. The terminal converts the cutter file into the required code for the particular machine tool controller.

A complete listing of a conversation for the production of a bottle mold is given in Appendix O.

Figure 5.8 is a photograph of a prototype bottle pattern, that has been designed using this system, being cut on an American Hustler lathe. Figure 5.9 is a photograph of a finished bottle pattern produced by this system. This also serves as a model of the production bottle. This would replace the wooden pattern in the method in current use.

The system described above has only been developed to a sufficient level to test the feasibility of producing bottle molds via a small scale computer-aided manufacturing system. To the present time, the system has been developed up to and including the automated production of a prototype pattern. The extension of this system to the complete production of a mold only requires a modification in the APT macro developed for the bottle pattern to produce an internal cut on the mold casting rather than an external cut on the Perspex stock.

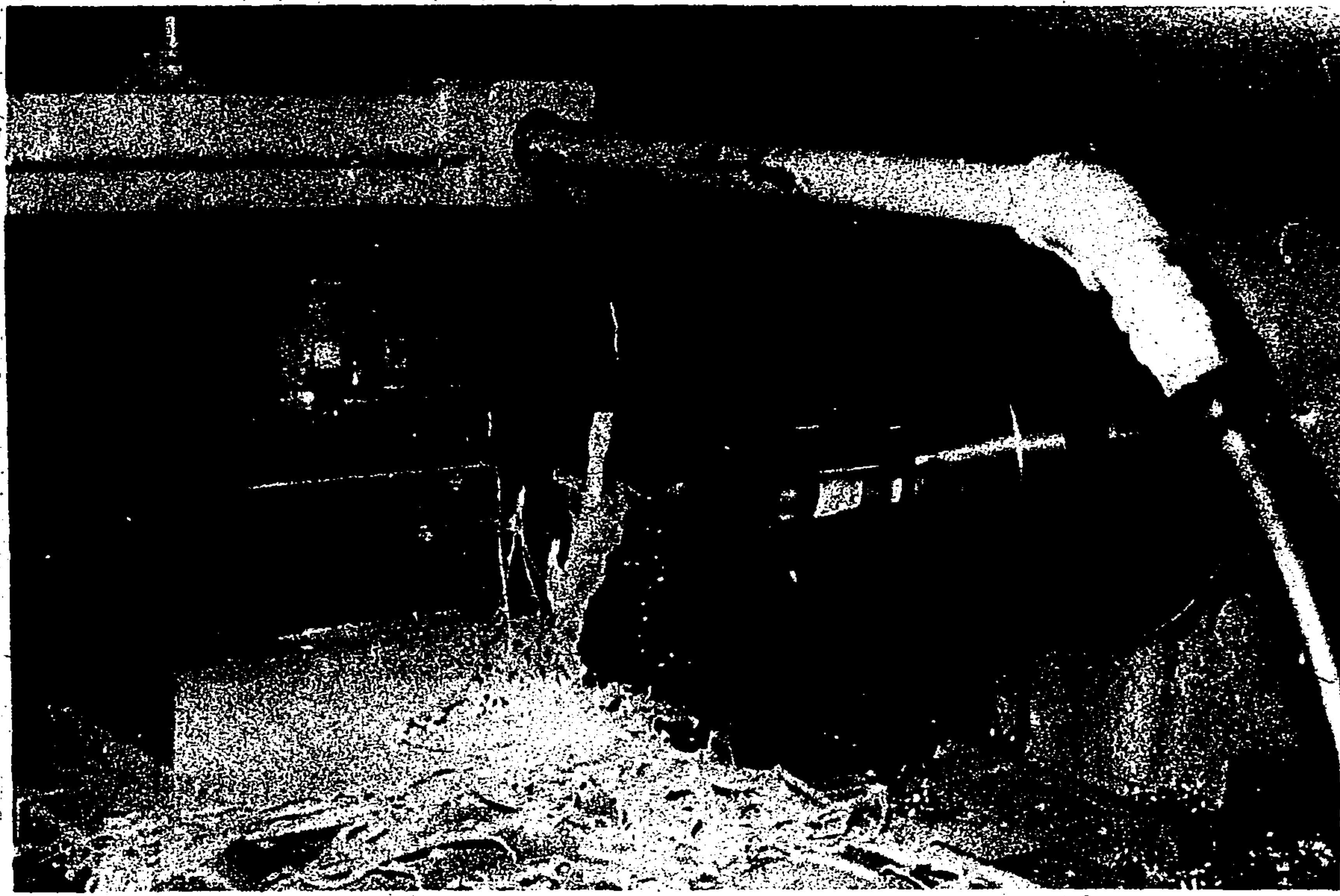


Figure 5.8 Prototype Bottle Pattern Being Turned on an American Hustler Lathe with a GE Mark Century Controller.

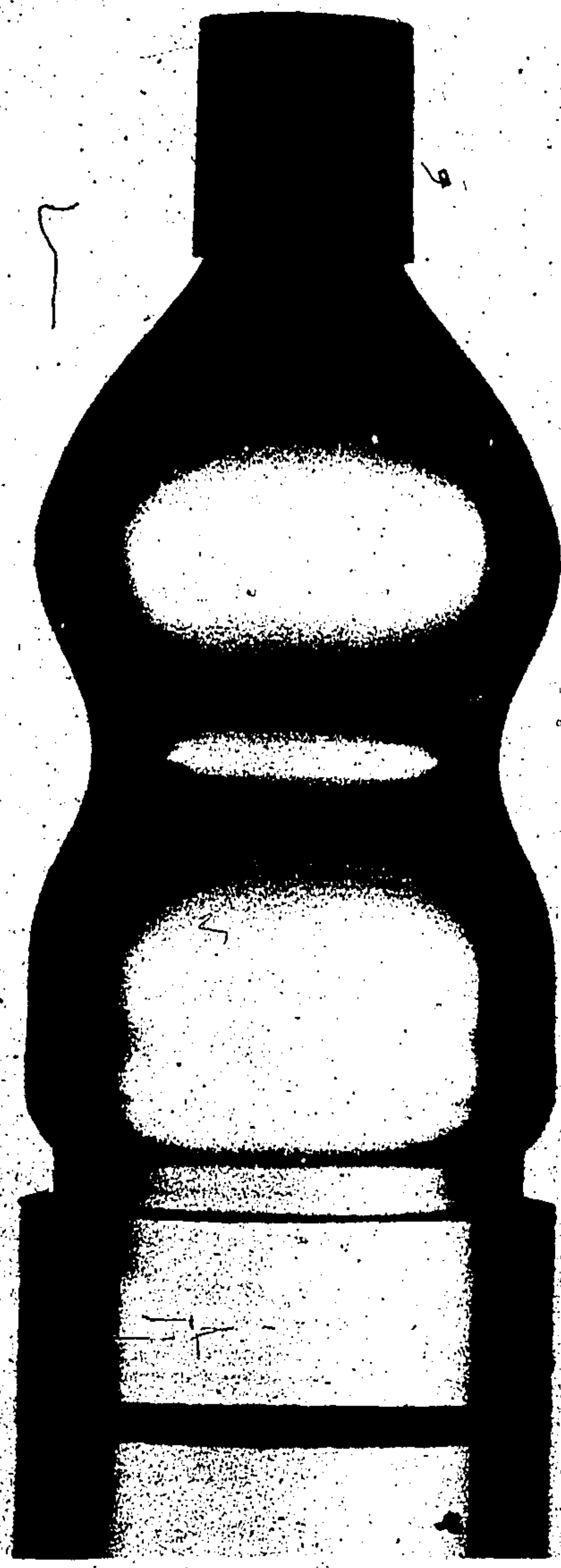


Figure 5.9 Finished Prototype Pattern - Displayed as a Model of the Finished Bottle.

The finished system would ideally include a library of macros that could be automatically selected by the system. This could be accomplished by a search for the macro for which the final roughing cut most closely approximated the finished part profile.

In a paper to be presented at the CIRP meeting in Japan in August 1974, Kakazu et al. (75) have proposed an algorithm to automatically generate cutter location data independently from APT. The adoption of this type of approach, after it has been fully proven, would greatly enhance the performance of systems such as the system described above, by eliminating APT and the need for a library of macros.

Not completed in this preliminary system, but easily appended, would be the production of an engineering drawing of the finished bottle. This drawing, for example, could be produced on the EDIT system's microfilm plotter and stored for record purposes.

The full system should be developed to include the following additional functions.

1. A sculptured surface routine, similar to those being developed in the automotive industry, for the definition and design of non-round bottles.

2. An automated data base selection of an appropriate mold casting from a file of existing patterns.

3. An automated selection of an appropriate neck ring design again from a data file of standard parts. The perspective display of the finished bottle should include the

capability to exchange possible neck and cap configurations.

The system to automate bottle mold design has been developed to demonstrate the feasibility of employing a terminal that has a programmable controller. The problems encountered in this application are common to a number of engineering design systems.

The expansion to include a direct connection between the terminal and the controllers associated with the range of machine tools available at McMaster University would create a system that could be employed to conduct research activities into all aspects of computer-aided manufacturing.

As mentioned in Chapter 2 of this thesis, one of the anomalies that exists in the current development of CAM systems research being recorded in the literature, is the scale of the hardware systems involved. The total systems cost involved is well beyond the reach of all but the largest manufacturing corporations. These same corporations, because of the economies of scale, are the least likely to benefit from the concepts of computer-aided manufacturing. The system developed in this work has been an attempt to demonstrate that a judicious selection of peripheral devices and a balanced application of processing tasks between the three levels of processing power can produce an effective and inexpensive system for computer-aided manufacturing in the small job shop and for the small manufacturer.

CHAPTER 6

CONCLUSIONS

The development of a system for the synthesis of machine structures has been completed by the effective integration of the engineer/designer into an interactive computer environment. This integration has been achieved through the design and development of both a design oriented computer terminal and a comprehensive modular software system. The general aspects of this approach have been demonstrated through the extension of the hardware and software to an additional application to an industrial problem in the manufacture of glass containers.

The thesis has been divided into chapters, each of which deals with one aspect of the system design. These chapters have been concluded with a summary of the pertinent aspects of the work as it related to the topic of the chapter. In this chapter, the system is reviewed in a general sense, and the salient features are briefly summarized.

The initial portion of the work has been the development of a comprehensive engineering design intelligent terminal (EDIT) that is a cohesive assemblage of design oriented computer peripherals capable of being employed in association with a large scale timeshared computing system. This terminal has been designed to provide a cost effective approach to computer-aided design through the inclusion of features that are designed to specifically deal with the traditionally cumbersome aspects of engineering design/analysis. Almost all engineering design consists of the formulation of large data bases for the problem definition and equally large data bases are generated for review of the solution. Low cost computer graphics have provided some relief for the review of the geometrical aspects of both the problem formulation and solution stages. However, the graphics systems available to date, are devices that are only active in the design process while in association with the large scale system. The large costs involved in even timeshared use of such systems has produced an imbalance between the designer and the computer. There is a psychological stress imposed on the designer to minimize the cost of computer employment. The development of stand-alone dedicated systems has failed to provide a solution to the problem due to the high costs involved and the inability of these systems to handle the large scale software packages available for engineering design/analysis.

The terminal developed in this work employs sufficient intelligence to allow the engineer to formulate and review his problem in a local relaxed environment while retaining the ease of programming and the computational capabilities of the large scale computing system. This has been realized through the design of an operating philosophy for the terminal that has the following features.

(1) Conversation and graphics are distinguished and are distributed to the appropriate device.

(2) Terminal peripherals can be used in a local mode through an easily comprehended strategy for program retrieval and execution. Under this mode the engineer can assemble, modify and edit data; retrieve information files, and review display data with the aid of any of the terminal's peripherals.

(3) On-line to a host computer the engineer, through a comprehensive software library can configure the terminal hardware to fit the requirements of his particular application. These peripherals can be employed in either an active state - the information is generated and received in real time, or a deferred state - information is generated from or transmitted to the terminal's storage medium. This latter operating mode serves as the link between the on-line and stand-alone operating systems for the terminal.

The terminal developed in this work has served the purpose of being a prototype facility for the development of a low priced, flexible engineering design station. Although the hardware itself is a one-of-a-kind system, the operating philosophy, the communications strategy developed, as well as the support software programs and concepts are fully employable and transferable to the development of a class of intelligent terminals for application to design engineering. This class of terminal has proved to be cost-effective in relation to systems that are available commercially and have been developed elsewhere. The overall concept of the terminal, and many of the detailed strategies, are original with this work.

The second stage of this work has been concerned with the development of a software system for the design of machine structures. This system, in conjunction with the terminal hardware and support software, has demonstrated the enhancement that occurs to the design process when the elements, hardware, software and designer are integrated into a cohesive system.

The software system has been divided into eight segments that can be executed independently or in succession within one terminal run. All information generated is automatically stored by the system for subsequent usage. This information can be transmitted to the terminal for storage.

The terminal's ability to distinguish graphics from conversation and physically separate these modes has made the query/response approach to interactive graphics feasible. The more conventional command oriented interactive systems which are suitable for highly repetitive design applications are not generally flexible enough to accommodate the casual user or handle the data requirements of engineering design. The adoption of a command responsive language structure has further enhanced the query/response strategy.

The software package has incorporated the terminal's features into a system to generate very inexpensive animated movies via a timeshared communication link. This feature now allows the engineer the opportunity to visually simulate a dynamic system response to gain a better "feel" for the design problem.

This type of designer/software/hardware integration has produced a system that utilizes the abilities and capabilities of each component in a very efficient form. Through the use of the terminal the data assemblage and graphics for the structure can be reviewed on-line or off-line; the software system structure allows complete or partial runs to be executed and thus has made possible a heuristic construction of a total design; and the query/response form of interactive programming has made it possible to develop a system that the designer can direct and review the solution algorithm at each stage of execution. The computer has been integrated with the designer of structures to a degree not previously achieved.

The final stage of the work has developed a new system to automate the design and production of molds for glass bottles. This system has been included in this work to demonstrate the flexibility of both the hardware and software described above. The engineering design terminal has been used as the central element of a three stage computer structure. In this application the terminal has been employed to interactively design bottles and prepare a data base for automated production of the bottle mold with very great potential for cost saving and better aesthetics.

CHAPTER 7

RECOMMENDATIONS

This chapter includes a brief discussion of recommended changes and additions to be made to the following five separate divisions: (1) the terminal Hardware; (2) the terminal software; (3) the terminal support software and graphics package; (4) CONSTRUCT software system; and, (5) BOTTLE software system.

The terminal hardware was described in Chapter 3 of this thesis. This chapter also included some recommendations for the acquisition of alternate peripheral devices that would be employed in the development of a new terminal. The terminal as it is presently constituted is capable of handling a wide range of engineering design applications. However, the following additional capabilities would enhance the terminal's performance in the area of computer-aided manufacturing. These recommendations are listed in order of deemed importance.

(1) More asynchronous ports. Eventually this type of data preparation and verification facility should be expanded to service a number of numerical control machines. There is ample software capacity to accommodate all the machine tools installed at the present time in McMaster. The hardwired machine tool controllers on the existing machine tools would have to be equipped with BTR (Behind the Tape Reader) converters.

(2) There should be a modem installed to handle at least 1200 baud. A simple modification to the PTOS would be required to accommodate this speed. This change is described in Appendix I.

(3) An alphanumeric display in the area of the digitizer would enable the user to view operating instructions, generated by his applications program, while seated at the digitizer.

(4) An additional cassette drive would completely eliminate the need for paper tape.

(5) The light pen should be replaced by a Joystick control.

(6) The COMPUTEK 400 terminal should be hardwired into the PDP8.

(7) The flat-bed plotter should be equipped with a modified version of the printer head. This portion of this unit has been responsible for a disproportionate number of failures.

(8) A low cost alphanumeric display should be added to allow conversational messages to be displayed to the user at the control station at the communication rate of the interprocessor link.

(9) A color filter adapter should be added to the movie camera to allow filming foreground and background displays in different colors.

The terminal support software will have to be appended to provide character generation on the drum plotter. This should not prove difficult as there are several programs available through DECUS to perform this function.

The terminal support software is the type of system that will benefit from continual expansion of its capabilities. The present system could be enhanced by the addition of a library of predefined (in display file format) objects that can be scaled and translated about the display area. The digitizer can easily be employed, within the present software structure, to select such options. This logic, however, should be more formally developed at the graphics function level.

The software system for the synthesis of machine structures in its present form is only a skeletal system that has almost unlimited room for expansion. This type of system, by definition, lends itself to continuous enhancement. The system's finite element analysis section should be expanded to handle different element types, more degrees of freedom, dynamic analysis, and thermal analysis. Data base management systems should be incorporated to provide the ability to draw upon files of element types and material properties.

The glass bottle system in its present state is at a very primitive level. This system should be expanded to incorporate sculptured surface techniques for non-round bottles. The graphics display of prototype bottles should include the capability to present shaded representations for more aesthetic appeal. The data base management system referenced above should be employed to select standard mold blanks and pre-defined neck ring configurations. The capability to produce a finished drawing for record purposes should be incorporated.

REFERENCES

1. Leckie, F. A., G. A. Butlin, and M. J. Platts, "The Use of Interactive Graphics in Engineering Design", International Symposium on Optimization of Structural Designs", University of Wales, Jan. 1972.

2. Fenves, S. J., "Scenario for a Third Computer Revolution in Structural Engineering", Journal of the Structural Division, Proceedings of the American Society of Civil Engineers, Jan. 1972.

3. Fano, R. M., "On the Social Role of Computer Communications", Proceedings of the IEEE, Vol. 60, No. 11, Nov. 1972.

4. Hobbs, L. C., "Terminals", Proceedings of IEEE, Vol. 60, No. 11, Nov. 1972, 1273-1284.

5. Tektronix Corp., Advertising Bulletin for the 4014 and 4015 Terminals.

6. McLean, R. S., and W. P. Oliver, "Initial Use of a Plasma Display Panel", Third Man-Computer Communications Seminar, N.R.C., Ottawa, 1973, 13.1-13.15.

7. Tlusty, J., Y. Koren and P. MacNeil, "Numerical and Adaptive Control for Die Sinking", Preprint of paper for presentation at the CIRP, 1974.

8. Vladov, V., A. Vuzeldv, G. Nachev, and N. Todorov, "Turning Operations Machining Systems Controlled by Minicomputer ISOT", International Future of NC/CAM, NCS Proceedings Eleventh Annual Meeting and Technical Conference, 1974, 302-308.

9. Hass, P. R., "Flexible Manufacturing Systems", International Future of NC/CAM, NCS Proceedings Eleventh Annual Meeting and Technical Conference, 1974, 288-301.

10. Miller, J. B., "SWING - Sundstrand's CNC", International Future of NC/CAM, NCS Proceedings Eleventh Annual Meeting and Technical Conference, 1974, 277-279.

11. Van Dam, A., G. M. Stabler and R. J. Harrington, "Intelligent Satellites for Interactive Graphics", Proceedings of IEEE, Vol. 62, No. 4, April 1974, 483-492.

12. Barstow, J. N., "The Terminal that Thinks for Itself:, Computer Decisions, Jan. 1973, 10-13.

13. Boardman, Jr., T. L., "On the Exploitation of Computing Systems and Computer Graphics in the Development of Effective, Economical Engineering Design Processes", Purdue University Ph.D Thesis, Mechanical Engineering, 1973.

14. Foley, J. L., "An Approach to the Optimum Design of Computer Architecture", Communications Association of Computer Machinery, Vol. 11, 1971, 380-390.

15. Computer Vision Corp, Advertising Bulletin.

16. Gerber Scientific Inc., Advertising Bulletin.

17. Hiltz, F. L., "How to Make Your Laboratory Computer Into a Smart Timesharing Terminal", DECUS Proceedings, Spring, 1973, 129-134.

18. Bonham, D. and J. E. Crozier, "Comprehensive Computer-Aided-Design Laboratory in an NC Center", International Future of NC/CAM, NCS Proceedings Eleventh Annual Meeting and Technical Conference, 53-59.

19. Dudley, T. K., "Xerox Computer Graphics", Proceedings of the Third Man-Computer Communications Seminar, NRC, Ottawa, 1973, 13.1-13.15.

20. Newman, W. M. and R. F. Sproull, "Principles of Interactive Computer Graphics", McGraw-Hill Book Company.

21. Newman, W. M. and R. F. Sproull, "An Approach to Graphics System Design", Proceedings IEEE, Vol. 62, No. 4, April 1974.

22. Williams, D. L., "GRAPPLE - Graphics Application Programming Language", Proceedings of the Third Man-Computer Communications Seminar, NRC, Ottawa, 1973, 5.1-5.10.

23. Brown, H. G., W. A. Hartman and R. E. Warburton, "Applications of the Interactive Computer Language, ICPL", Third Man-Computer Communications Seminar, NRC, Ottawa, 1973.

24. Forran, G., "Le Language Graphique GASTON", Third Man-Computer Communications Seminar, NRC, 10.1-10.5.

25. Pieke, B. and G. Schrack, "Implementation of an Interactive Graphics Language", Third Man-Computer Communications Seminar, Ottawa, 1973, 7.1-7.9.

26. Kyle, R. G., "Speed in Information Processing with a Computer-Driven Visual Display in a Real-Time Digital Simulation", M.Eng Thesis, Virginia Polytechnic Institute and State University, 1972.

27. Foley, J. D., "The Art of Natural Graphic Man-Machine Conversation", Proceedings of IEEE, Vol. 62, No. 4, April 1974, 462-471.

28. Martin, J., "Design of Man-Computer Dialogues", Prentice Hall, 1973.

29. Roos, D., ICES System Design", The M.I.T. Press, 1966.

30. MRI/STARDYNE, "CDC Cybernet Publication, 1973.

31. Atkins Research, "ASAS User's Information Note No. 4", Nov. 1971.

32. McCormick, C. W., "The NASTRAN User's Manual (Level 15.5)", The MacNeal-Schwendler Corp., Los Angeles, California, March 1974.

33. Alcock, D. G. and B. H. Shearing, "GENESYS - An Attempt to Rationalise the Use of Computers in Structural Engineering", The Structural Engineer, Vol. 48, No. 4, April 1970, 143-152.

34. Gallagher, R. H. and O. C. Zienkiewicz, "Optimum Structural Design, Theory and Applications", John Wiley and Sons, 1973.

35. Bayer, L. T. and J. C. Casson, "Experiences with Interactive Computing System", Journal of the Structural Division, Proceedings of the American Society of Civil Engineers, Jan. 1971, 289-299.

36. STRESS - User's Manual CDC Corporation.

37. Douty, R. and S. Shore, "Technique for Interactive Computer Graphics in Design", Journal of the Structural Divison, Proceedings of the American Society of Civil Engineers, Jan. 1971, 273-287.

38. Cornell, C. A., K. Renschmidt, and J. F. Brotchie, "A Method for the Optimum Design of Structures", Proceedings International Symposium on the Use of Computers in Structural Engineering", Newcastle, England, 1966.

39. Moses, F., "Optimum Structural Design using Linear Programming", Journal of the Structural Division, Proceedings of the American Society of Civil Engineers, Vol. 90, Dec. 1964, 89-104.

40. Ramstad, K. M. and C. K. Wang, "Optimum Design of Framed Structures", Journal of the Structural Division, Proceedings of the American Society of Civil Engineers, Vol. 94, Dec. 1968, 2817-2845.

41. Fletcher, R., "Mathematical-Programming Methods - A Critical Review", John Wiley and Sons, 1973, Edited by R. H. Gallagher and O. C. Zienkiewicz, 51-77.

42. Moses, F., R. L. Fox and G. Gubbe, "Mathematical Programming Applications in Structural Design", in Computer Engineering, S.M. Study No. 5, University of Waterloo Press, 1971.

43. Brown, D. M., and A. H. S. Ang, "Structural Optimization by Nonlinear Programming", Journal of the Structural Division, American Society of Civil Engineers, Vol. 92, 1966, 319-340.

44. Moe, J., "Penalty Function Methods", Optimum Structural Design, John Wiley and Sons, 1973, Edited by R. H. Gallagher and O. C. Zienkiewicz, 143-177.

45. Kaulie, D. and J. Moe, "Automated Design of Frame Structures", Journal Structural-Division, American Society of Civil Engineers, Vol. 97, 1971, 33-62.

46. Fox, R. L., "Mathematical Methods in Optimization", An Introduction to Structural Optimization, Study No. 1, Solid Mechanics Division, University of Waterloo, 1968, 47-80.

47. Lasdon, L. S., "Optimization Theory for Large Systems", The MacMillan Co., London, 1970.

48. Kirsch, U., M. Reiss and U. Shamir, "Optimum Design by Partitioning into Substructures", Journal of the Structural Division, Proceedings of the American Society of Civil Engineers, Jan. 1972, 249-267.

49. Aguilar, R. J., "An Application of Dynamic Programming to Structural Optimization", Engineering Research Bulletin No. 91, The Division of Engineering Research, Louisiana State University, 1967.

50. Palmer, A. C., "Optimal Structural Design by Dynamic Programming", Journal of the Structural Division, Proceedings of the American Society of Civil Engineering, Aug. 1968, 1887-1906.

51. Twisdale, L. A. and Khachaturian, "Absolute Minimum Weight Structures by Dynamic Programming", Journal of the Structural Division, Proceedings of the American Society of Civil Engineers, Vol. 99, Nov. 1973.

52. Kaester, J. L. and J. H. Mize, "Optimization Techniques with Fortran", McGraw Hill Inc., 1973.

53. Reinschmidt, K. F., "Discrete Structural Optimization", Journal of Structural Division, Proceedings of the American Society of Civil Engineers, Vol. 97, Jan. 1971, 133-155.

54. Cella, A. and R. D. Logcher, "Automated Optimum Design from Discrete Components", Journal of the Structural Division, Proceedings of the American Society of Civil Engineers, Vol. 97, Jan. 1971, 175-189.

55. Gisvold, K. M. and J. Moe, "A Method for Nonlinear Mixed-Integer Programming and its Applications to Design Problems", Journal of Engineering for Industry, Transactions of the American Society of Mechanical Engineers, May 1972, 353-364.

56. Siddall, J. N. and D. J. Bonham, "Optisep - Designers' Optimization Subroutines", Department of Mechanical Engineering, McMaster University, Hamilton, Ontario, 1974.

57. Birta, L. G., "Optpak - A Program Package for Unconstrained Function Minimization", Department of Computer Science, University of Ottawa, Ottawa, Ontario, 1974.

58. Pierson, B. L., "A Survey of Optimal Structural Design Under Dynamic Constraints", International Journal for Numerical Methods in Engineering, Vol. 4, 1972, 491-499.

59. Suta, T., N. Sato, Y. Takeuchi, and N. Takashiva, "Analysis of Thermal Deformation of Machine Tools by the Finite Element Method", Annals of the CIRP, Vol. 21-1, 1972, 123-124.

60. DeVries, M. F., "Analyzing Costs in Conventional and CAM Systems", The International Future of NC/CAM, NCS Proceedings Eleventh Annual Meeting and Technical Conference, 1974, 22-31.

61. Kawahaba, M., "CAM Challenge to Productivity" NC/CAM Profits for the 70's, NCS Proceedings Tenth Annual Meeting and Technical Conference, 1973, 11-26.

62. Fleming, I. and R. White, "Computer-Aided Management of Production", The International Future of NC/CAM, NCS Proceedings Eleventh Annual Meeting and Technical Conference, 1974, 77-87.

63. Meyer, T. H. and I. E. Sutherland", On the Design of Display Processes", Communications of the Association of Computer Machinery, Vol. 11, No. 6, June 1968.

64. Ishimatsu, Y., J. Nagaska, H. Araki and T. Uchiyamada, "Computer-Aided Generation of Sculptured Surfaces", Journal of Numerical Control, April 1974, 33-39.

65. Gott, B., "NC: Interface with Design", The International Future of NC/CAM, NCS Proceedings the Eleventh Annual Meeting and Technical Conference, 1974, 35-52.

66. Guedj, R. A., "The Challenge of Computer Graphics in Continental Western Europe", Proceedings of the IEEE, Vol. 62, No. 4, April 1974.

67. Gardner, N., "Container Design by Computer" SME Technical Paper No. MS73-961, 1973.

68. Folwell, J. D., "NC Applications in Glass Mold Manufacturing", NC/CAM Profit for the 70's, NCS Proceedings, The Tenth Annual Meeting and Technical Conference, 1973, 333-340.

69. Cowley, A., "Cooperative Work in Computer-Aided Design in the CIRP", Internal Report U.M.I.S.T., Jan. 1972.

70. Van den Noortgate, L., "Calculation of Static and Dynamic Characteristics of Axial Structures by Computer Techniques", Internal Report Center De Recherches Scientifique et Techniques De L'Industrie Der Fabrications Metalliques, 1970.

71. James, W. and P. Zachar, "Development and Application of Large Interactive CAI (Simulation) Program Packages in Environmental Science", Proceedings of Fourth Ontario University Computing Conference, Feb., 1973, 7-13.

72. Sutherland, I. E., "Three-Dimensional Data Input by Tablet", Proceedings of the IEEE, Vol. 62, No. 4, April 1974, 453-461.

73. Rubenstein, M. F., "Structural Systems - Statics, Dynamics and Stability", Prentice Hall, Englewood Cliffs, N.J., 1970.

74. Kakazu, Y. N. Okino, K. Hoshi, "Penalty Method Determining NC Cutter Path Automatically", To be presented at the General Assembly of CIRP, 1974.

75. EAI Dataplotter Reference Manuals, Vols. 1 to 3.

76. Digital Equipment Corp., "Introduction to Programming", Vols. 1 to 3, 1972.

CREDITS

Mr. G. Gouldson of CARED-CIM, McMaster University assisted with the APT parts program of the Bottle mold project.

Mr. J. Forster, Mechanical Engineering Department, McMaster University assisted with the diagrams.

APPENDIX A

EDIT SYSTEM USER'S MANUAL

ABSTRACT

The EDIT (Engineering Design Intelligent Terminal) is a collection of design oriented computer peripherals, controlled by a 4k PDP8 minicomputer, that is employed in association with a timeshared large scale computing system. This terminal has been structured to enable a remote user to configure a hardware system through software to meet the needs of an engineering design application. In addition, the terminal has been designed to incorporate a number of time and cost saving stand-alone funotions.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Hardware Overview

The EDIT system is controlled by a PDP8/L mini-computer that has 4096 words of memory. In association with this computer are the following peripheral devices:

(1) Digitizer - Ruscom Logics Model 21. This digitizer is equipped with a control panel, two input stations, and a write-only 7 track tape recorder. It is interfaced with a Teletype and the PDP8.

(2) Cassette Tape Transport - Sykes Compu/corder 100.

(3) Flat-bed Plotter - EAI 3500 Dataplotter. This plotter is equipped with an 8 pen turret, a 48 character printing head, and a read-only 7 track tape drive.

(4) High Speed Paper Tape Reader/Punch.

(5) Asynchronous I/O Interface - This interface has communication rates of 110, 300, 600, and 4800 baud.

(6) Tektronix 611 Storage CRT - This CRT is controlled by a scope controller with a character generator, interpolator, and a vector generator.

(7) Microfilm Plotter - Ferranti Packard Model 1.

(8) ASR 33 Teletype

(9) Light Pen - Lake Electronics (not in service)

(10) Incremental Drum Plotter - Calcomp Model 565 with a 11? in drum and a .005 in step size. (being developed)

(11) 16 mm Movie Camera - Bolex camera with a computer controlled solenoid operated shutter.

In addition, the following are related peripherals available to the terminal for use in an off-line mode.

(12) AX08 Laboratory Peripheral - This lab peripheral has 8 multiplexed channels of A/D and 3 channels of D/A.

(13) Computek 400 Graphics CRT.

1.2 Operating Modes

There are two principle operating modes for the terminal.

(1) On-line - In this mode the terminal's peripherals can be used in conjunction with the terminal support software resident in a large scale timeshared computer. (At McMaster the system employed is the INTERCOM 4.1 timesharing system on a CDC 6400 computer.)

(2) Off-line - In this mode the terminal is used as a stand-alone system to collect, store, and edit data. It is also used to regenerate display information on any graphics peripheral from the local storage devices.

1.3 Terminal System Software

All the programs related to the operation of the terminal are stored on a cassette labelled EDIT SYSTEM Version 1.0. This cassette is written in a format by which these programs are easily retrieved and executed. The following is a general description of a convention for naming programs, a method for the retrieval, and a method for printing an index of available programs.

1.3.1 Convention for Naming Programs

All associated peripherals on the system are identified by a two letter mnemonic as follows:

- CR - Cathode Ray Tube
- PT - Paper Tape Reader/Punch (high speed)
- FP - Flat-bed Plotter
- CA - Cassette Tape Unit
- DI - Digitizer
- MP - Microplotter
- TY - Teletype
- LP - Light Pen
- DP - Drum Plotter
- MC - Movie Camera

The terminal programs associated with operating these peripherals in a stand-alone operating mode are named with 4 character mnemonics (2+2) composed as follows:

(Information generating peripheral) (Information receiving peripheral)

example:

DICA

this program reads data from the digitizer and records it on a cassette tape.

1.3.2 How to Retrieve and Execute Programs

The programs on the Library Cassette (EDIT SYSTEM Version 1.0) can be retrieved and executed as follows.

(i) Set PDP8 Switch Register to 7777_8; press LOAD ADDRESS.

(ii) Turn on cassette drive and place EDIT SYSTEM cassette into the tape transport with label facing the user.

(iii) Press START.

(iv) When the terminal responds with READY at the Teletype the user responds at the Teletype with:

L,XXXX,H followed by CARRIAGE RETURN

where: XXXX is a 4 character program name.

(v) When the MOTION light on the cassette flashes, the casette drive can be turned off, the cassette removed and the program executed.

(vi) Set the Switch Register to the starting address of the program, press LOAD ADDRESS and then START.

1.3.3 How to Print an Index

To print an Index of available programs follow steps (i) to (iii) above then:

(iv) When the terminal responds with READY, the user responds with

T followed by CARRIAGE RETURN

and index of available programs will be printed showing their starting addresses and their addresses on the cassette tape.

1.4 Host System Software

The terminal support software and graphics package, which are used in association with the large scale time-shared computer to operate the terminal peripherals, are stored at the host computer as a library of SUBROUTINES.

At McMaster University these SUBROUTINES can be used with the following INTERCOM commands.

```
COMMAND -  ATTACH, EDITPLT, ID = ABCD.
COMMAND -  FTN (I = XXXX)
COMMAND -  XEQ
OPTION  =  LDSET, LIB = EDITPLT
OPTION  =  LGO
```

where ABCD is the account number under which the library is stored

XXXX is the file name under which the user's application program resides in source code.

CHAPTER 2

DEADSTART PROCEDURES

2.1 General

The PDP8 will normally retain the basic programs required to load system programs in the manner described in the previous chapter. However, on occasion it may be necessary to load the system from scratch. (This may result from a system failure or following a different application for the PDP8.) The programs that must be entered into core are the RIM LOADER and the CASSETTE LOADER. The RIM loader must be entered via the Switch Register on the computer's front panel. The CASSETTE LOADER is available as a binary program on paper tape.

2.2 System Loading

(1) Check RIM loader in core. If this loader has been overwritten it must be entered into core using the Switch Register on the PDP8 front panel. This loader is described in detail in the DEC programming manual. If is repeated here for convenience.

ADDRESS	INSTRUCTION
7756	6014
7757	6011
7760	5357
7761	6016

7762	7106
7763	7006
7764	7510
7765	5374
7766	7006
7767	6011
7770	5367
7771	6016
7772	7420
7773	3776
7774	3376
7775	5357

(2). Load the CASSETTE LOADER by the following.

(i) Place paper tape marked CASSETTE LOADER in high speed paper tape reader.

(ii) Set Switch Register to 7756; press LOAD ADDRESS and START. The paper tape will be read into the computer. The basic loaders are now in core and the user may proceed with the loading of specific programs.

CHAPTER 3

HARDWARE OPERATING INSTRUCTIONS

3.0 General

This chapter describes the procedures required to initialize the terminal's peripherals. A general rule to follow when using the terminal in the on-line operating mode is to turn on all the required peripherals in advance of completing the communications link to the host computer. There are two reasons for this (1) some peripherals have a "warm up" time to stabilize their response and (2) other peripherals energize the interrupt facility on the computer when they are started. This latter consideration will cause a system crash of the local terminal software.

3.2 PDP8

The PDP8 is started with a key on the lower left side of the PDP8 front panel. This key has three positions (1) POWER, (2) PANEL LOCK and (3) OFF. The normal operating condition is in the POWER position. The PANEL LOCK inhibits computer control from the front panel.

The MEM PROT switch when down (0 state) allows the computer to write into the last memory page (where the system loaders are resident). When this switch is up (1 state) this page of core can not be written over.

The following peripherals are energized with the computer -

AX08, PT08, DW08, and High Speed Paper Tape Reader.

3.3 Digitizer

The POWER ON button is located on the lower right side of the small input table. All the operating options for the digitizer are implemented through switch settings on the control panel.

(i) Large or small input station. Set center thumbwheel switch to

0 - Small back-lighted table
1 - Large tilt table.

Note: The appropriate shaft encoders must be connected.

(ii) Select data acquisition rate. Set right-hand thumbwheel switch:

0 - 10 characters/second
1 - 300 characters/second

(iii) Select output option.

For magnetic tape: Mount tape as indicated by diagram inside cover of tape drive. Turn switch on drive to RECORD position. Press EOF and BOT buttons and press IRG button 9 times on control panel. Set output rate to 300 characters/second.

For Teletype List/Paper Tape: Turn on digitizer prior to energizing the Teletype. Set Teletype switch to LINE position. Set switches on front panel below the keyboard to DIGITIZER. For paper tape copy press paper tape punch ON button. Set output to 10 characters/second.

For PDP8. No special options required.

3.4 Flat-bed Plotter

(i) Press POWER button

(ii) Press STANDBY

(iii) Press RESET and SYM together.

(iv) Select input device.

Magnetic Tape. Press POWER on tape drive. Mount tape by pressing START/BRAKE switch to BRAKE. Press START/BRAKE switch to START. Press LOAD POINT. Press AUTO.

PDP8. Disconnect tape drive at rear of plotter.

(v) Press STOP on plotter control.

(vi) Press START.

3.5 Microplotter/CRT

(i) Press MAINS ON on plotter.

(ii) Set display options (STORAGE UNIT), as follows.

HOLD - display can not be erased under program control

NON-STORE - display will not be stored on screen (i.e., disappears as created)

AUTO-ERASE - display is stored until erase character is received from computer (normal operating mode).

(iii) Set SINGLE/AUTO switch to SINGLE.

(iv) Set PROJECTOR switch as follows:

ON - a border is overlayed on the aperture card produced (if any).

OFF - aperture card printed as displayed on CRT.

(v) Press RESET and FAULT RESET.

(vi) Press ERASE on CRT - this should be done as soon as the screen has been fully illuminated to prevent phosphor damage.

(vii) When LOGIC READY and FILMSORT READY lamps are lit, the plotter is ready.

3.6 Teletype

(i) Set front panel switch as follows.

LINE - the characters typed on the keyboard are transmitted to the PDP8. A character (verification) transmitted by the PDP8 is printed on the paper.

LOCAL - characters typed on the keyboard are printed but not transmitted to the computer.

(ii) Set paper tape punch ON to punch a copy of all information transmitted to the Teletype.

3.7 Acoustic Coupler

(i) Set switches on rear of unit as:

ACST/DAA to ASCT

Full/Half to Full

(ii) Turn power on.

3.8 Cassette Tape Unit

The cassette tape drive is activated by a POWER button on the unit's front panel. Cassettes must be loaded with the label on the cassette facing the user.

3.9 Computek 400

This is a graphics terminal that is designed to be used directly, in association with the timesharing system. It can be used as with the EDIT system indirectly by first transmitting the display file to the EDIT system's storage device, then regenerating the file for local display on the Computek terminal. This is done by connecting the RS232 pin connector on the PT08 output channel to the input connector for the Computek terminal via the cable strip labelled TERMINAL to TERMINAL.

3.10 Asynchronous Interface

(i) Select communication rate at front panel as required. Note: Some changes are required to change from 110 baud to a higher rate (see Appendix I).

3.11 Movie Camera

(i) Connect coaxial cables between the BNC connector on the front panel of the computer and the corresponding BNC connector on the camera driver.

(ii) The SYNC signal available on the S_0 BNC connector on the AX08 front panel is used to generate the pulse to trigger the camera.

CHAPTER 4

SOFTWARE

4.1 General

The software systems associated with the terminal were discussed in a general sense in the Introduction of this manual. This section describes these programs in detail and indicates the manner in which they are employed. The systems involved are as follows: (1) Operating System for the Terminal for on-line use with a host computer; (2) Package of programs for employing the terminal in its stand-alone functions; (3) host computer terminal support software and graphics package.

4.2 Stand-Alone Terminal System

The strategy employed to name the programs required for the stand-alone functions was described in Chapter 1, Section 3. The programs that are presently in the library are listed below with a description of their operating procedures.

The paper tape or cassette programs to drive the terminal peripherals in the stand-alone mode can be created locally at the terminal by manual programming or by employing the terminal support software and subsequently disposing the files created to the appropriate medium.

NAME	S.A.	FUNCTION AND OPERATING PROCEDURE
PTFP	(0200_8)	Paper Tape to Flat-bed Plotter (i) Start flat-bed plotter in accordance with instructions under EDIT OPERATING PROCEDURE, Section 3.4. (ii) Load the paper tape to be plotted in the high speed tape reader. Caution: start tape on blank leader - if the leader contains a valid code the plotter will attempt to execute it. (iii) Set Switch Register to (0200_8); press LOAD ADDRESS and START. Plotter will plot information recorded on the paper tape. Paper tapes can be created using any Teletype manually or in association with the terminal support software and the timesharing system.
PTCA	(0200_8)	Paper Tape to Cassette (i) Place paper tape to be transferred to cassette tape in high speed reader. Caution: all characters are written to tape including leader. (ii) Set Switch Register to (0200_8). characters. Press ~~LOAD~~ ADDRESS.

(iii) Place cassette in tape transport with write-enable plug inserted for track B. Turn on power to cassette unit.

(iv) Press START.

(v) In response to T.A. = at the Teletype the user must supply a 4 digit tape address at which the data will be recorded. Information is recorded in records of 400_8 8 bit characters until an ASCII # (243_8) is read from the paper tape.

(vi) The next available tape address is returned to the user at the termination of recording. Cassette is removed.

CAMP (0200$_i$) Cassette to Microplotter

(i) Start microplotter in accordance with the instructions given in Section 3.5.

(ii) Load cassette into the transport and turn power on.

(iii) Set Switch Register to (0200_8); press LOAD ADDRESS; press START.

(iv) In response to T.A. = the user must supply a 4 digit tape address corresponding to the starting address of the display data.

(vi) Plot is generated, aperture card is produced and the cassette is rewound.

PTMP (0200_8) **Paper Tape to Microplotter**

(i) Start microplotter in accordance with the instructions given in Section 3.5.

(ii) Place paper tape to be plotted in high speed paper tape reader.

(iii) Set Switch Register to (0200_8); press LOAD ADDRESS and START.

(iv) Set bit 10 of Switch Register to "1" and press CONTINUE. An aperture card will be produced when an ASCII EOM (203_8) is read from input tape.

PTCR (0200_8) **Paper Tape to CRT**

(i) Follow first steps (i) to (iii) under PTMP.

(ii) Press CONTINUE.

DIPT (0200_i) **Digitizer to Paper Tape**

(i) Start digitizer in accordance with the instructions in Section 3.5.

(ii) Turn on high speed paper tape punch.

(iii) Set Switch Register to (0200_8); press LOAD ADDRESS; press START.

(iv) The transmission of characters to the PDP8 is initiated by either the foot-switch or the LOG button at the digitizer. All characters are buffered until a 1.0 second delay on input is encountered. The buffer is then punched on paper tape followed by CARRIAGE RETURN (215_8). This feature provides the user with control over the punch record length.

DICA (0200_8) <u>Digitizer to Cassette</u>

(i) Start digitizer in accordance with the instructions in Section 3.3.

(ii) Load a cassette with a write-enable plug in track B.

(iii) Set Switch Register to (0200_8); press LOAD ADDRESS; press START.

(iv) In response to T.A. = the user must supply a 4 digit tape address at which the data will commence to be recorded.

(v) Data is recorded in records of (400_8) characters.

(vi) Press "R" on digitizer control panel to terminate data recording and dump final buffer.

(vii) Next available tape address is returned to the user.

CAFP (0200_8) Cassette to Flat-bed Plotter

(i) Start flat-bed plotter in accordance with the instructions in Section 3.4.

(ii) Load cassette into transport. Turn transport power on.

(iii) Set Switch Register to (0200_8); press LOAD ADDRESS; press START.

(iv) In response to T.A. = the user must supply the 4 digit tape address at which the plotter data is recorded.

PTPT (0200_8) Paper Tape to Paper Tape

(i) Turn on high speed punch.

(ii) Place paper tape to be copied in high speed reader.

(iii) Set Switch Register to (0200_8); press LOAD ADDRESS; press START.

TYMP (0200_8) Teletype to Microplotter

(i) Start Microplotter in accordance with the instructions in Section 3.5.

(ii) Set Switch Register to (0200_8); press LOAD ADDRESS; press START.

(iii) Set bit 10 of Switch Register to a 1 and press CONTINUE.

(iv) Plots can be generated from the Teletype keyboard (note: input characters are not echoed). An aperture card will be produced upon receipt of an ASCII EOM (203_8).

TYCR (0200_8) <u>Teletype to CRT</u>

(i) Follow steps (i) and (ii) of TYMP.

(ii) Press CONTINUE.

CACR (0200_8) <u>Cassette to CRT</u>

(i) Follow steps (i) to (v) of CAMP.

(ii) Cassette is rewound.

PTMC (0200_8) <u>Paper Tape to Movie Camera</u>

(i) Load paper tape of display to be filmed into high speed reader.

(ii) Set up camera.

(iii) Set Switch Register to (0200_8); press LOAD ADDRESS; press START.

MAC3 (0200_8) <u>Assembler</u>

This a 3 pass program used to assemble source coded programs into binary. This assembler includes the mnemonics for the EDIT peripherals and is employed as indicated in Appendix L.

EDTR (0200_8)

(i) See DEC reference (78) under chapter titled LOADING, EDITING and DEBUGGING, section SYMBOLIC EDITOR.

DUPO (0400_8)

(i) Load paper tape to be copied in high speed reader.

(ii) Turn on high speed punch.

(iii) Set Switch Register to (0400_8); press LOAD ADDRESS.

(iv) Place the octal compliment of the characters to be deleted into memory locations 0436 to 0441. Octal compliment is obtained as follows (eg., for deletion of CARRIAGE RETURN).

$$\begin{array}{r} 1000_8 \\ 0215_8 \\ \hline 7563_8 \end{array} \quad \text{(Complement of } 215_8\text{)}$$

(v) Press START.

TEIA (4400)

(i) Place ASCII coded NC tape in high speed reader.

(ii) Turn on high speed punch.

(iii) Set Switch Register to (4400_8); press LOAD ADDRESS; press START. (Note: This program has been contributed by A. Srivastava).

4.3 On-line Terminal Operating System

The logic of the on-line terminal operating system is contained in a single program called EDIT resident on the library cassette. This program can be loaded with the instructions given for general program loading in the introduction. The general operating procedure with helpful suggestions follow.

(i) Load the EDIT program as per the instructions in the Introduction, Section 1.3.2.

(ii) Start all required peripherals according to the operating procedures described in Chapter 3.

(iii) Set Switch Register to (0200_8); press LOAD ADDRESS; press START.

(iv) Dial the telephone number of the timesharing system being employed. (4756 for 110 baud and 4721 for 300 baud at McMaster).

(v) Place the headset of the telephone on the acoustic coupler with the cord towards the indentation on the coupler.

(vi) Return to the Teletype to complete the "logging in" operation.

Suggestion 1. If the computer appears to be an extraordinary length of time responding - press the RETURN key on the Teletype.

2. If the local operating system should crash, complete step (iii) above. The system can now be reactivated by entering a command to the timesharing system from the Teletype keyboard.

3. If the user wishes to suspend host activities for the performance of a local stand-alone operation but does not wish to lose the connection to the host, simply halt the computer (press STOP), load the required program following the instructions above, execute the program, reload EDIT and repeat step (iii) above. The host system may be held inactive in the middle of execution by using a dummy (READ) statement in the host program. This option will require the entry of a character at the Teletype followed by a CARRIAGE RETURN to reinitiate activity.

4. Simulation of the host system to test program options can be done with the Computek Terminal connected as per the instructions in Chapter 3. The user is cautioned that the following program location contents must be altered to account for different codes used by the Computek from those of the standard ASCII codes.

Location	New Contents
0054	7566
0057	7536
0060	7534
0062	7531
0063	7537

4.4 On-line Host Software

The host computer software is resident, as a library of SUBROUTINES, in the memory of the timeshared computer. The functions performed by this software package are to control the terminal peripherals and generate the graphics.

A description of these SUBROUTINES follow with an explanation of their application.

NAME	DESCRIPTION
DIGIN	To read coordinate data produced by the digitizer Call: DIGIn (X, Y, Z IV, ICNT, IU) where: Input, IV = 1 Coordinates are returned under X and Y IV = 2 Coordinates are returned under X and Z IV = 3 Coordinates are returned under Y and Z IU = Logical unit number of device on which coordinate data file is to be written. IU = 0 - no external file is written

IU < 0 - digitizer data is read from cassette unit. (Deferred state)

Output, X, Y, Z are coordinate data in inches (actual values of digitized coordinates).

These arrays must be dimensioned in calling program with Values ≥ ICNT

ICNT - Number of digitized points.

PROJECT To create a 2 dimensional perspective plot of a 3-dimensional object.

Call:

PROJECT (XYZ, XY, OBS)

where:

XYZ is a 3 dimensional array of the points coordinates in space where:

XYZ (1) = X
XYZ (2) = Y
XYZ (3) = Z

cartesion right hand coordinate system

XY is a 2 dimensional array of transformed coordinates where:

XY (1) = X
XY (2) = Y

display coordinate system

OBS is a 3 dimensional array defining the point of observation in space where:

OBS (1) = X

OBS (2) = Y

OBS (3) = Z

(Original coding contributed by R. K. Shepard)

PLTLN

To draw a line from present beam (pen) position to location (X, Y)

Call:

PLTLN (X, Y, LNTYPE, LINCR, IUNITS)

Where:

LNTYPE defines line type to be drawn

LNTYPE = 1 dark move (pen up)
= 2 bright (pen down), thin and continuous
= 3 bright, thick and continuous
= 4 bright, thin and dotted
= 5 bright, thick and dotted
= 6 bright, thin and dashed
= 7 bright, thick and dashed
= 8 bright, thin and chain dotted
= 9 bright, thick and chain dotted

LINCR = 0 coordinates are specified absolutely
= 1 coordinates are specified incrementally

UNITS = 0 coordinates are in user's units
= 1 coordinates are in display units (See note below).

Note: Display units are the numerical values of the hexadecimal display file coordinates. In this package the origin (display file origin) is translated to the lower left position in the display area. This gives an addressable area of 65,536 units in each axis. The visible display area is 40,960 units in X and 32,768 units in Y.

LETTER

To write a string of characters starting at position (X, Y)

Call:

LETTER (X, Y, N, ISIZE, BCD, IUNITS, LINCR)

Where:

BCD	-	an array of characters to be printed, dimensioned with the value of N. Display code; one character/word.
N	-	number of characters to be printed
ISIZE	= 1	character size one (small)
	= 2	character size two (large)
IUNITS and LINCR	-	as defined above.

SYMBOL

To draw a symbol centered about (X, Y)

Call:

SYMBOL (X, Y, ISIZE, NSYMBL, IUNITS, LINCR) where:

NSYMBL = display coded symbol to be printed.

ISIZE, IUNITS, MINCR as defined above.

DATATO

To convert user's units to display units.

Call:

DATATO (XD, YD, XP, YP)

Where:

(XD, YD) = X and Y coordinates in user's units

(XP, YP) = X and Y coordinates in display units.

PLOTTO

To convert display units into user's units.

Call:

PLOTTO (XP, YP, XD, YD)

Where:

(XP, YP) = X and Y coordinates in display units.

(XD, YD) = X and Y coordinates in user's units.

PLTIN

To initialize scales and boundary values.

Call:

PLTIN (XSCALE, YSCALE, XMIN, XMAX, YMIN, YMAX)

Where:

XSCALE = user specified X-axis scale, user's units per display unit

YSCALE = user specified Y-axis scale;

XMAX, XMIN - maximum and minimum values of X

YMAX, YMIN - maximum and minimum values of Y

FACTOR

To automatically compute scale values.

Call:

FACTOR (M, X, Y, XMARG, YMARG)

Where:

M = Number of points in the coordinate arrays X and Y

XMARG, YMARG - Fraction of total plot length along X and Y axis specified as margin (divided equally on each side of display).

CIRCLE

To draw a circle of a specified radius centered at X, Y.

Call:

CIRCLE (X, Y, RAD; IUNITS, LINCR)

Where:

(X, Y) = coordinates of circle center.

RAD = circle radius.

IUNITS, LINCR as defined above.

ARCP

To draw an arc from present position to point (X, Y).

Call:

ARCP (X, Y, XC, YC, IARC, IUNITS)

Where:

(XC, YC) - coordinates of center of arc

IARC = 1 drawn clockwise

= 1 drawn anti-clockwise

Iunits as defined above.

ARC

To draw an arc from (X1, Y1) to (X2, Y2).

Call:

ARC (X1, Y1, X2, Y2, XC, YC, IARC, IUNITS, LINCR).

PLTLIN

To draw a line from (X1, Y1) to (X2, Y2).

Call:

PLTLIN (X1, Y1, X2, Y2, LNTYPE, LINCR, IUNITS).

Where:

LNTYPE, LINCR, IUNITS are defined above.

MPLOT

To make a dark move to (X(1), Y(1)) and then successive bright moves to (X(2), Y(2))---(X(M), Y(M)).

Call:

MPLOT (M, X, Y, LNTYPE, LINCR, IUNITS)

Where:

LNTYPE, LINCR, IUNITS are defined above.

MSYMBL

To plot a symbol at a series of M coordinates (X, Y).

Call:

MSYMBL (M, X, Y, ISIZE, NSYMBL, IUNITS, LINCR)

Where:

M = number of coordinates

ISIZE, NSYMBL, IUNITS, LINCR are defined above.

GRAPH

To draw a graph with specified X and Y values joined by straight lines. Calibrated axis are drawn in the display area.

Call:

GRAPH (NN, X, Y, XINCR, YINCR, LNTYPE)

Where:

NN = number of data points

(X, Y) array of data points (must be specified as absolute values)

XINC - distance between calibration marks on X-axis in data units.

YINC - distance between calibration marks on Y-axis in data units.

DASH

To draw a dashed line from (X1, Y1), where lines are segmented as (2L, L, 4L, L, 2L).

Call:

DASH (X1, Y1, X2, Y2, LNTYPE, IUNITS)

INPT

To decode input commands in an interactive program. Used to decode directive options:

Call:

INPT (IO, FO, IX, IC)

Where:

IO = Integer number output

FO = Floating point output

IX = 1 integer output required

= 0 floating point number required

IC = 1 the command BEGIN has been decoded

= 2 the command BACK has been decoded

= 3 the command AHEAD has been decoded

= 4 the command HALT has been decoded

= 5 the command HELP has been decoded

RDCAS — To initialize cassette unit to transmit data to host computer.

Call:

RDCAS

This program must be followed by user specified READ statements to read from INPUT file according to tape format.

WCAS — To write cassette files from host computer.

Call:

WCAS

The user must follow this with formatted WRITE statements to the OUTPUT file.

MPPCT — To read display file and format for microplotter.

Call:

MPFLT (K)

Where:

K = 0 microplotter is in active state

K = 1 microplotter is in deferred state.

CRPLT

To read display file and format for CRT.

Call:

CRPLT (K)

Where:

K is as defined above.

FPPLT

To read display file and format for Flat-bed plotter.

Call:

FPPLT (K)

Where:

K is defined above.

DPPLT

To read display file and format for flat-bed plotter.

(To be developed.)

Note: A portion of the above programs have been developed by A. Srivastava.

These SUBROUTINES are employed in association with the user's application program in a manner completely analogous to the use of any other computationally oriented SUBROUTINE. A representative block of code follows that demonstrates a few salient points.

```
      CONTINUE
         :
      WRITE (6, 100) ENTER (1) TO START DIGITIZER/1H,
100   FORMAT (1HO, 28H 1 5 X, (19 H(2) TO HALT PROGRAM)
      CALL INPT (10, 80, 1, IC)
      IC = IC+1
      GO TO (10, 20, 30, 30, 40, 50), IC
10    CALL DIGIN (X, Y, Z, 1, ICNT, 2)
      GO TO 30
50    WRITE (6, 200)
200   FORMAT (22 HO YOU NEED HELP ALRIGHT)
30    CONTINUE GO TO 20
40    STOP
```

APPENDIX B

GRAPHICS SOFTWARE

APPENDIX C

EDIT OPERATING SYSTEM

```
                /EDIT OPERATING SYSTEM
                *0
0000  0002      INTRPT,  0
0001  5402              JMP I 2
0002  0250              SERVIN
                /AUTOINDEX FOR BUFFER STORE
                *10
0010  0000      RBFPTR,  0
0011  2177      ELEV,    2177
0012  2177      TOLVI,     2177
                *20
0020  0000      TA,        0000
0021  2200      K2200,     2200
0022  2377      K2377,     2377
0023  2177      RTWLE,     2177
0024  2577      RTWLV,     2577
0025  2177      RELEV,   2177
0026  3000      KDIG,      DIGIT
0027  3200      FBPC,       FLBP
0030  4000      RCAS,      RDCAS
0031  3400      VCAS,      VRCAS
0032  0000      MODET,   0
0033  1777      RB2000,  1777
0034  0000      RCVPTR,  0
0035  0000      SNDPTR,  0
0036  0000      ACPTR,   0
0037  0000      PCPTR,   0
0040  0700      ITPTR,   ICNTTY
0041  0263      TIPTR,   TTYICH
0042  1200      IPLPTR,  ICHPLT
0043  7652      KV,      -126
0044  7454      KT,      -324
0045  7540      KSPACE,  -240
0046  7574      KEDT,  -204
0047  7575      KEOM,    -203
0050  7644      KESCP,   -134
0051  0001      K0001,   1
0052  7777      K7777,   7777
0053  7563      KCR,     -215
0054  7766      KLF,     -12
0055  0377      KCHLRB,  377
0056  7532      KAMP,    -246
0057  7736      KINVCH,  -42
0060  7734      KDOLLR,  -44
0061  7535      KHASH,   -243
0062  7731      KQUOTE,  -047
0063  7737      KEOMK,   -041
0064  7533      KOFFLN,  -245
0065  0002      K0002,   2
0066  0000      LOOPR1,   0
0067  0000      LOOPR2,   0

0070  0000      MDETV,   0
0071  0000      SVRUB,   0
0072  0000      PLTLIN,  0
0073  0000      RBPLIN,  0
0074  1614      PLTPTR,  PLOT
0075  1204      MPLPTR,  PLTTYP
0076  0000      MDMDPL,  0
0077  1334      RUBPTR,  RUBOUT
0100  0000      MDCARD,  0
0101  0000      TEMP1,   0
0102  0000      CHAR,    0
0103  0000      SVESC,   0
0104  1400      TCHPTR,  TSTCHR
0105  1640      FPLPTR,  FPLOT
0106  1633      FPSPTR,  FPLTST
0107  0600      KBSPTR,    KBDSND

0110  0202      RSTPTR,  RCVST
0111  0707      KBRPTR,  KBDRCV
0112  0740      PTRPTR,  PTORCV
0113  1000      TPRPTR,  TPRCV
0114  0713      SSTPTR,  SENDST
0115  1203      PSTPTR,  PLTST
0116  1600      PLSPTR,  PLOTST
0117  6512      CRT,     6512
0120  6514      CARDS,   6514
0121  7401      KRUB,    -377

0122  1066      WTRPTR,  WAITR
0123  0203      CDEOM,   203
0124  0204      CDEOT,   204
0125  7550      KCTRLX,  -230
0126  7701      KOMRK,   -77
0127  0000      SVPNCH,  0
                /
0130  7760      KRCVCY,   -20
0131  0000      LOOPI1,  0
0132  0000      KBCVWT,  0
0133  2000      PLFPTR,  2000
0134  0000      SVLFCR,  0
0135  0000      SVCTLX,  0
0136  0000      COUNTX,  0
0137  0000      SVXIA,   0
0140  0000      TEMP2,   0
0141  0000      OFLPTR,  0
                PAUSE
                *200
0200  7200      START,   CLA
0201  3032               DCA MODET
0202  6002      RCVST,   IOF
0203  7300               CLA CLL
0204  1033               TAD RB2000
0205  3010               DCA RBFPTR
0206  1033               TAD RB2000
0207  7001               IAC
0210  3034               DCA RCVPTR
0211  3071               DCA SVRUB
0212  3076               DCA MDMDPL
0213  3100               DCA MDCARD
0214  3127               DCA SVPNCH
                /
0215  6402      FSTCHR,  6402
0216  6001               ION
0217  5513               JMP I TPRPTR
                *250
0250  3036      SERVIN,  DCA ACPTR
0251  1000               TAD 0
0252  3037               DCA PCPTR
                /
0253  7200               CLA
0254  1032               TAD MODET
0255  7650               SNA CLA
0256  5440               JMP I ITPTR
0257  1032               TAD MODET
0260  1052               TAD K7777
0261  7640               SZA CLA
0262  5442               JMP I IPLPTR
                /
0263  6031      TTYICH,  KSF
0264  7410               SKP
0265  5507               JMP I KBSPTR
                *600
0600  6002      KBDSND,  IOF
0601  6416               6416
0602  7200               CLA
                /
```

6

```
0603  6036  SNDPTG,  KRB
0604  6046           TLS
0605  3035           DCA SNDPTR
0606  1035           TAD SNDPTR
0607  1053           TAD KCR
0610  7650           SNA CLA
0611  4054           JMS OUTLF
0612  6411           6411
0613  5212           JMP .-1
0614  1035           TAD SNDPTR
0615  6415           6415

0616  1053  CHKRTN,  TAD KCR
0617  7650           SNA CLA
0620  5241           JMP RTNRCV
0621  1035           TAD SNDPTR
0622  1054           TAD KLF
0623  7650           SNA CLA
0624  5241           JMP RTNRCV
0625  1035           TAD SNDPTR
0626  1125           TAD KCTRLX
0627  7650           SNA CLA
0630  5234           JMP CHCTX
0631  6031           KSF
0632  5231           JMP .-1
0633  5203           JMP SNDPTG
                /
0634  7001  CHCTX,   IAC
0635  3135           DCA SVCTLX
0636  7001           IAC
0637  7040           CMA
0640  3136           DCA COUNTX
                /
0641  7200  RTNRCV,  CLA
0642  3134           DCA SVLFCR
0643  3032           DCA MODET
0644  6411           6411
0645  5244           JMP .-1
0646  6412           6412
0647  6041           TSF
0650  5247           JMP .-1
0651  6042           TCF
0652  6032           KCC
0653  5510           JMP I RSTPTR
0654  0000  OUTLF,   0000
0655  1063           TAD OLFC
0656  6041           6041
0657  5256           JMP .-1
0660  6046           6046
0661  7200           CLA
0662  5454           JMP I OUTLF
0663  0012  OLFC,    0012
            *700
0700  6031  ICNTTY,  KSF
0701  7410           SKP
0702  5511           JMP I KBRPTR
0703  6401           6401
0704  7410           SKP
0705  5512           JMP I PTRPTR
0706  7402           HLT
0707  6002  KBDRCV,  IOF
0710  7201           CLA IAC
0711  3032           DCA MODET
0712  5514           JMP I SSTPTR
                /
0713  7200  SENDST,  CLA
0714  5507           JMP I KBSPTR
                     *740
0740  6402  PTORCV,  6402
0741  6042           TCF
0742  6002           IOF
0743  6406           6406
0744  3410           DCA I RBFPTR
0745  1032           TAD MODET
0746  7450           SNA
0747  5513           JMP I TPNPTR
0750  7001           IAC
0751  7650           SNA CLA
0752  1076  HDPL,    TAD MDHDPL
0753  7640           SZA CLA
0754  5474           JMP I FLTPTR
0755  5475           JMP I HPLPTR
            PAUSE
            *1000
1000  6002  TPRCV,   IOF
1001  7200  CHKESC,  CLA
1002  1010           TAD RBFPTR
1003  3140           DCA TEMP2
1004  1540           TAD I TEMP2
1005  1050           TAD KESCP
1006  7640           SZA CLA
1007  5213           JMP TYPUT
1010  1052           TAD K7777
1011  3032           DCA MODET
1012  5515           JMP I PSTPTR
                /
1013  1135  TYPUT,   TAD SVCTLX
1014  7640           SZA CLA
1015  5217           JMP CTLX
1016  5236           JMP NOLFCR
                /
1017  1434  CTLX,    TAD I RCVPTR
1020  6046           TLS
1021  2136           ISZ COUNTX
1022  5225           JMP CTX2
1023  7200           CLA
1024  3135           DCA SVCTLX
1025  7200  CTX2,    CLA
1026  6041           TSF
1027  5226           JMP .-1
1030  6042           TCF
1031  1135           TAD SVCTLX
1032  7650           SNA CLA
1033  5265           JMP PRSTP
1034  2034           ISZ RCVPTR
1035  5217           JMP CTLX
                //
1036  6001  NOLFCR,  ION
1037  1130           TAD KRCVCT
1040  3132           DCA KRCVWT
1041  3131           DCA LOOPI1
1042  2131           ISZ LOOPI1
1043  5242           JMP .-1
1044  2132           ISZ KRCVWT
1045  5242           JMP .-3
                /
1046  6002  RCPRNT,  IOF
1047  1034  PRCYL,   TAD RCVPTR
1050  7041           CIA
1051  7001           IAC
1052  1010           TAD RBFPTR
1053  7650           SNA CLA
1054  5265           JMP PRSTP
1055  1434           TAD I RCVPTR
1056  6046           TLS
1057  2034           ISZ RCVPTR
1060  6041           TSF
1061  5260           JMP .-1
1062  6042           TCF
1063  7200           CLA
1064  5247           JMP PRCYL
1065  6001  PRSTP,   ION
                /
1066  2066  WAITR,   ISZ LOOPR1
1067  5266           JMP .-1
1070  2067           ISZ LOOPR2
1071  5270           JMP .-1
```

```
1072  5266              JMP .-4
                 PAUSE
                 *1200
                 /INTERRUPT SERVICING, RECEIVE
1200  6031  ICMPLT,  KSF
1201  7410           SKP
1202  5511           JMP I KBRPTR
1203  6401           6401
1204  7410           SKP
1205  5517           JMP I PTAPTR
1206  6501           6501
1207  7410           SKP
1210  5220           JMP MPLINT
1211  6651           6651
1212  7410           SKP
1213  5426           JMP I KDIG
1214  6601           6601
1215  7410           SKP
1216  5427           JMP I FBPC
1217  7402           HLT
                 /
1220  6502  MPLINT,  6502
1221  6001           ION
1222  5522           JMP I WTRPTR
                 //
                 /START OF NON-TTY DATA
1223  6002  PLTST,   IOF
1224  6401  SCHDCK,  6401  /KSFPTI
1225  5224           JMP .-1
1226  6406           6406  /KRBPTI
1227  3410           DCA I RBFPTR
1230  3103           DCA SVESC
1231  7040           CMA
1232  3032           DCA MODST
1233  1010           TAD RBFPTR
1234  3101           DCA TEMP1
1235  1501           TAD I TEMP1
1236  1063           TAD KCOMK
1237  7650           SNA CLA
1240  5426           JMP I KDIG
1241  1501           TAD I TEMP1
1242  1056           TAD KAMP
1243  7650           SNA CLA
1244  5430           JMP I RCAS
1245  1501           TAD I TEMP1
1246  1057           TAD KINVCH
1247  7650           SNA CLA
1250  5431           JMP I VCAS
1251  1501           TAD I TEMP1
1252  1062           TAD KQUOTE
1253  7650           SNA CLA
1254  5427           JMP I FBPC
1255  1501           TAD I TEMP1
1256  1060           TAD KDOLLR
1257  7650           SNA CLA
1260  5270           JMP NCD
1261  1501           TAD I TEMP1
1262  1106           TAD KCMMK
1263  7640           SZA CLA
1264  5272           JMP JUMP1
1265  7201           CLA IAC
1266  3127           DCA SVPNCH
1267  5272           JMP JUMP1
1270  7201  NCD,     CLA IAC
1271  3100           DCA NDCARD
1272  7040  JUMP1,   CMA
1273  1101           TAD TEMP1
1274  3034           DCA RCVPTR
1275  5475           JMP I MPLPTR
                 /
                 /
                 *1304
1304  2034  PLTTYP,  ISZ RCVPTR
1305  1434           TAD I RCVPTR
1306  1050           TAD KESCP
1307  7650           SNA CLA
1310  5516           JMP I PLSPTR
1311  1434           TAD I RCVPTR
1312  1126           TAD KCMMK
1313  7640           SZA CLA
1314  5320           JMP NDR
1315  7201           CLA IAC
1316  3127           DCA SVPNCH
1317  5334           JMP NDI
                 /
1320  1434  NDR,     TAD I RCVPTR
1321  1121           TAD KRUB
1322  7650           SNA CLA
1323  5477           JMP I RUBPTR
1324  1127           TAD SVPNCH
1325  7640           SZA CLA
1326  5334           JMP NDI
1327  1434           TAD I RCVPTR
1330  6026           PLS
1331  6021           PSF
1332  5331           JMP .-1
1333  6022           PCF
1334  6001  NDI,     ION
1335  5522           JMP I WTRPTR
                 *1336
1336  2071  RUBOUT,  ISZ SVRUB
1337  1071           TAD SVRUB
1340  1052           TAD K7777
1341  7640           SZA CLA
1342  5363           JMP WAITRB
1343  1034           TAD RCVPTR
1344  1065           TAD K0002
1345  3073           DCA RBFLIM
1346  7100           CLL
1347  1032           TAD MODST
1350  7012           RTR
1351  7650           SNA CLA
1352  5363           JMP WAITRB
1353  1127           TAD SVPNCH
1354  7640           SZA CLA
1355  5363           JMP WAITRB
1356  1434           TAD I RCVPTR
1357  6026           PLS
1360  6021           PSF
1361  5360           JMP .-1
1362  6022           PCF
                 /
1363  7200  WAITRB,  CLA
1364  1073           TAD RBFLIM
1365  7041           CIA
1366  1010           TAD RBFPTR
1367  7640           SZA CLA
1370  5334           JMP NDI
                 /
1371  1033  REINLZ,  TAD KB2000
1372  3010           DCA RBFPTR
1373  1033           TAD KB2000
1374  3034           DCA RCVPTR
1375  3071           DCA SVRUB
1376  5334           JMP NDI
                 PAUSE
                 *1400
1400  0000  TSTCHR,  0
1401  1102           TAD CHAR
1402  1050           TAD KESCP
1403  7640           SZA CLA
1404  5210           JMP ION
1405  7201           CLA IAC
1406  3103           DCA SVESC
1407  5600           JMP I TSTCHR
1410  1102  ION,     TAD CHAR
1411  1056           TAD KAMP
1412  7640           SZA CLA
```

```
1413  5220            JMP EOT
1414  1123            TAD CDEOM
1415  3102            DCA CHAR
1416  3103            DCA SVESC
1417  5600            JMP I TSTCHR
1420  1102  EOT,      TAD CHAN
1421  1057            TAD KINVCM
1422  7640            SZA CLA
1423  5230            JMP NXTCHR
1424  1124            TAD CDEOT
1425  3102            DCA CHAR
1426  3103            DCA SVESC
1427  5600            JMP I TSTCHR
1430  7200  NXTCHR,   CLA
1431  1103            TAD SVESC
1432  1052            TAD K7777
1433  7650            SNA CLA
1434  5236            JMP MDTXVR
1435  5270            JMP INTXVR

1436  1102  MDTXVR,   TAD CHAR
1437  1045            TAD KSPACE
1440  7640            SZA CLA
1441  5245            JMP TXT
1442  1055            TAD KCRLRB
1443  3102            DCA CHAR
1444  5600            JMP I TSTCHR
1445  1102  TXT,      TAD CHAR
1446  1044            TAD KT
1447  7640            SZA CLA
1450  5254            JMP VCTR
1451  7001            IAC
1452  3070            DCA MDITV
1453  5270            JMP INTXVR
1454  7200  VCTR,     CLA
1455  1102            TAD CHAR
1456  1043            TAD KV
1457  7640            SZA CLA
1460  5264            JMP INPLN
1461  7040            CMA
1462  3070            DCA MDITV
1463  5270            JMP INTXVR
1464  7200  INPLN,    CLA
1465  3070            DCA MDITV
1466  3103            DCA SVESC
1467  5600            JMP I TSTCHR
1470  1070  INTXVR,   TAD MDITV
1471  7650            SNA CLA
1472  5310            JMP RETRN
1473  1102            TAD CHAR
1474  1053            TAD KCR
1475  7640            SZA CLA
1476  5302            JMP LF
1477  1055            TAD KCRLRB
1500  3102            DCA  CHAR
1501  5310            JMP RETRN
1502  1102  LF,       TAD CHAR
1503  1054             TAD KLF
1504  7640            SZA CLA
1505  5310            JMP RETRN
1506  1055            TAD KCRLRB
1507  3102            DCA CHAR
1510  7200  RETRN,    CLA
1511  3103            DCA SVESC
1512  5600            JMP I TSTCHR
            PAUSE
            *1600
1600  7201  FLOTST,   CLA IAC
1601  3076            DCA MDMDFL
1602  1127            TAD SVPNCH
1603  7640            SZA CLA
1604  5207            JMP DCH
1605  1434            TAD I RCVPTR
1606  6026            PLS
1607  7200  DCH,      CLA
1610  1434            TAD I RCVPTH
1611  3102            DCA CHAR
1612  2103            ISZ SVESC
1613  5506            JMP I FPSPTR

1614  2034  PLOT,     ISZ RCVPTR
1615  1434            TAD I RCVPTR
1616  1121            TAD KRUB
1617  7650            SNA CLA
1620  5477            JMP I RUBPTR
1621  1434            TAD I RCVPTR
1622  3102            DCA CHAR
1623  4504            JMS I TCHPTR
1624  1127            TAD SVPNCH
1625  7640            SZA CLA
1626  5231            JMP .+3
1627  1102            TAD CHAR
1630  6026            PLS
1631  7200            CLA
1632  5505            JMP I FPLPTR

1633  7200  FFLTST,   CLA
1634  6511            6511
1635  5234            JMP .-1
1636  4311            JMS MDSLCT
1637  0000  CODE,     0

1640  6002  FFLOT,    IOF
1641  1102            TAD CHAR
1642  1046            TAD KEOT
1643  7650            SNA CLA
1644  5264            JMP PLTRCV
1645  1102            TAD CHAR
1646  1047            TAD KEOM
1647  7650            SNA CLA
1650  5266            JMP PLTRCF
1651  1102            TAD CHAR
1652  6506            6506
1653  7200            CLA
1654  1127            TAD SVPNCH
1655  7640            SZA CLA
1656  5262            JMP FPI
1657  6021            PSF
1660  5257            JMP .-1
1661  6022            PCF
1662  6001  FPI,      ION
1663  5522            JMP I WTRPTR

1664  1102  PLTRCV,   TAD CHAR
1665  6506            6506
1666  7200  PLTRCF,   CLA
1667  1127            TAD SVPNCH
1670  7640            SZA CLA
1671  5275            JMP .+4
1672  6021            PSF
1673  5272            JMP .-1
1674  6022            PCF
1675  7200            CLA
1676  3030            DCA MODET
1677  3071            DCA SVRUB
1700  3076            DCA MDMDFL
1701  3100            DCA MDCARD
1702  1033            TAD RB2000
1703  3010            DCA RBFPTR
1704  1033            TAD RB2000
1705  7001            IAC
1706  3034            DCA RCVPTR
1707  6001            ION
1710  5522            JMP I WTRPTR

1711  0000  MDSLCT,   0
1712  1100            TAD MDCARD
1713  7640            SZA CLA
```

```
1714  5320            JMP .+4
1715  1117            TAD CRT
1716  3737            DCA CODE
1717  5711            JMP I MDSLCT
            PAUSE
            *3000
            /DIGITIZER HANDLING ROUTINE
3000  7200  DIGIT,    CLA
3001  6031            KSF
3002  7410            SKP
3003  5511            JMP I KBRPTR
3004  6401            6401
3005  7410            SKP
3006  5300            JMP INPTO
3007  6651            6651
3010  5276            JMP GOAG
3011  6656            6656
3012  3411            DCA I ELEV
3013  6001  AGA2,     ION
3014  1130            TAD KRCVCT
3015  3132            DCA KRCVWT
3016  3134            DCA LOOP11
3017  2131            ISZ LOOP11
3020  5217            JMP .-1
3021  2132            ISZ KRCVWT
3022  5217            JMP .-3
3023  6002            IOF
3024  1011            TAD ELEV
3025  7041            CIA
3026  3140            DCA TEMP2
3027  1025            TAD RELEV
3030  3011            DCA ELEV
3031  6416            6416
3032  1411  AGAIN,    TAD I ELEV
3033  3377            DCA TTDI
3034  1327            TAD KBCD7
3035  1377            TAD TTDI
3036  7650            SNA CLA
3037  5300            JMP INPTO
3040  1377            TAD TTDI
3041  1326            TAD TS0
3042  6411            6411
3043  5242            JMP .-1
3044  6416            6416
3045  7200            CLA
3046  1011            TAD ELEV
3047  1140            TAD TEMP2
3050  7640            SZA CLA
3051  5232            JMP AGAIN
3052  1325            TAD CR
3053  6411            6411
3054  5253            JMP .-1
3055  6416            6416
3056  7200            CLA
3057  1331            TAD DLY12
3060  3332            DCA DLY13
3061  2332            ISZ DLY13
3062  5261            JMP .-1
3063  6401            6401
3064  5263            JMP .-1
3065  6406            6406
3066  7200            CLA
3067  6411            6411
3070  5267            JMP .-1
3071  6412            6412
3072  6402            6402
3073  7200            CLA
3074  1025            TAD RELEV
3075  3011            DCA ELEV
3076  6001  GOAG,     ION
3077  5522            JMP I VTRPTR
3100  7200  INPTO,    CLA
3101  6652            6652
3102  7200            CLA
3103  1330            TAD ASCR
3104  6411            6411
3105  5304            JMP .-1
3106  6416            6416
3107  7200            CLA
3110  1333            TAD CRDI
3111  6411            6411
3112  5311            JMP .-1
3113  6416            6416
3114  7200            CLA
3115  6411            6411
3116  5315            JMP .-1
3117  6412            6412
3120  7200            CLA
3121  1025            TAD RELEV
3122  3011            DCA ELEV
3123  3032            DCA MODET
3124  5510            JMP I RSTPTR
3125  0215  CR,       0215
3126  0260  TS0,      0260
3127  7727  KBCD7,    7727
3130  0322  ASCR,     0322
3131  7771  DLY12,    -7
3132  0000  DLY13,    0000
3133  0215  CRDI,     0215
            /
            /
            PAUSE
            *3200
3200  7200  FLBP,     CLA
3201  6606            6606
3202  7200            CLA
3203  6414            6414
3204  7200            CLA
3205  6404            6404
3206  7200            CLA
3207  6031            KSF
3210  7410            SKP
3211  5511            JMP I KBRPTR
3212  6401  GETCH1,   6401
3213  5212            JMP .-1
3214  6406            6406
3215  3325            DCA CHR1
3216  1325            TAD CHR1
3217  3412            DCA I TWLVE
3220  1325            TAD CHR1
3221  1061            TAD KHASH
3222  7650            SNA CLA
3223  5230            JMP OUTFL
3224  1012            TAD TWLVE
3225  1024            TAD KTWLV
3226  7640            SZA CLA
3227  5212            JMP GETCH1
3230  1012  OUTFL,    TAD TWLVE
3231  7041            CIA
3232  3326            DCA TMP1
3233  1023            TAD RTWLE
3234  3012            DCA TWLVE
3235  1412  GETCH2,   TAD I TWLVE
3236  3325            DCA CHR1
3237  1061            TAD KHASH
3240  1325            TAD CHR1
3241  7650            SNA CLA
3242  5276            JMP RET0
3243  1325            TAD CHR1
3244  1056            TAD KAMP
3245  7650            SNA CLA
3246  5305            JMP RET1
3247  1325            TAD CHR1
3250  6601            6601
3251  5250            JMP .-1
3252  6606            6606
3253  7200            CLA
3254  1325            TAD CHR1
```

```
3255  1331              TAD KPLT1
3256  7650              SNA CLA
3257  5265              JMP SLWDN
3260  1012                TAD TWLVE
3261  1326                TAD TMP1
3262  7650                SNA CLA
3263  5305                JMP RET1
3264  5235                JMP GETCH2
3265  7200     SLWDN,   CLA
3266  1430              TAD KRCVCY
3267  3132              DCA KRCVWT
3270  3131              DCA LOOP11
3271  2131              ISZ LOOP11
3272  5271              JMP .-1
3273  2132              ISZ KRCVWT
3274  5271              JMP .-3
3275  5212              JMP GETCH1
3276  1327     RET2,      TAD STNBY
3277  6601                6601
3300  5277                JMP .-1
3301  6606                6606
3302  7200                CLA
3303  6652              6652
3304  5314              JMP RET4
3305  1330     RET1,      TAD SPC
3306  6411              6411
3307  5306              JMP .-1
3310  6416              6416
3311  7200              CLA
3312  6601     RET3,    6601
3313  5312              JMP .-1
3314  6602     RET4,    6602
3315  1023              TAD RTWLE
3316  3012                DCA TWLVE
3317  6411              6411
3320  5317              JMP .-1
3321  6412                6412
3322  6402                6402
3323  3032                DCA MODET
3324  5510              JMP I RSTPTR
3325  0000     CHR1,      0000
3326  0000     TMP1,      0000
3327  0017     STNBY,     0017
3330  0240     SPC,       0240
3331  7731     KPLT1,   7731
               PAUSE
               *3400
3400  6404     VRCAS,   6404
3401  7200              CLA
3402  6414              6414
3403  7200              CLA
3404  6044              6044
3405  7200              CLA
3406  4756              JMS I RESU
3407  4755              JMS I INCT
3410  4757              JMS I RTIC
3411  7240              CLA CMA
3412  3533              DCA I MLFPTR
3413  7300              CLA CLL
3414  1021              TAD K2200
3415  3763              DCA I BUSY
3416  3422              DCA I K2377
3417  4760              JMS I OPEN
3420  2200              2200
3421  3561              HLTO
3422  4762              JMS I CKBD
3423  4334              JMS SPCO
3424  7200     STARTO,    CLA
3425  3367              DCA PRIN
3426  4334     STARTW,  JMS SPCO
3427  1133     GETCH1,  TAD MLFPTR
3430  3010              DCA RBFPTR
3431  1365     GETI,    TAD MUNCH
3432  3366              DCA CNTIT
3433  6401     GETIT,   6401
3434  5233              JMP .-1
3435  6406              6406
3436  3374              DCA TCHAN
3437  1374              TAD TCHAN
3440  3410              DCA I RBFPTR
3441  1374              TAD TCHAN
3442  1061              TAD KHASH
3443  7650              SNA CLA
3444  5253              JMP END
3445  2366              ISZ CNTIT
3446  5233              JMP GETIT
3447  1367              TAD PRIN
3450  7650              SNA CLA
3451  5275              JMP ENP1
3452  5255              JMP AEND
3453  7240     END,     CLA CMA
3454  3367              DCA PRIN
3455  1021     AEND,    TAD K2200
3456  3423              DCA I RTWLE
3457  7240              CLA CMA
3460  3421              DCA I K2200
3461  3422              DCA I K2377
3462  1020              TAD TA
3463  3533              DCA I MLFPTR
3464  4770              JMS I WRITE
3465  2000              2000
3466  3561              HLTO
3467  4762              JMS I CKBD
3470  7240              CLA CMA
3471  3020              DCA TA
3472  2367              ISZ PRIN
3473  5224              JMP STARTO
3474  5301              JMP STPP
3475  2367     ENP1,    ISZ PRIN
3476  2010              ISZ RBFPTR
3477  2010              ISZ RBFPTR
3500  5231              JMP GETI
3501  1133     STPP,    TAD MLFPTR
3502  3763              DCA I BUSY
3503  7240              CLA CMA
3504  3533              DCA I MLFPTR
3505  3423              DCA I RTWLE
3506  4771              JMS I POST
3507  2000              2000
3510  3561              HLTO
3511  4762              JMS I CKBD
3512  4773              JMS I POSTIT
3513  7240              CLA CMA
3514  3533              DCA I MLFPTR
3515  3423              DCA I RTWLE
3516  1133              TAD MLFPTR
3517  3763              DCA I BUSY
3520  4772              JMS I CLOSE
3521  2000              2000
3522  3561              HLTO
3523  4762              JMS I CKBD
3524  4334              JMS SPCO
3525  6042              6042
3526  6402              6402
3527  6411              6411
3530  5327              JMP .-1
3531  6412              6412
3532  3032              DCA MODET
3533  5510              JMP I RSTPTR
3534  0000     SPCO,    0000
3535  7200              CLA
3536  1364              TAD SPCC
3537  6411              6411
3540  5337              JMP .-1
3541  6416              6416
3542  7200              CLA
3543  1375              TAD CRO
3544  6411              6411
```

```
3545 5344        JMP .-1
3546 6416        6416
3547 7200        CLA
3550 6401        6401
3551 5350        JMP .-1
3552 6406        6406
3553 7200        CLA
3554 5734        JMP I SPCO
3555 3600 INCI,  INCH
3556 3622 MESA,  MSG
3557 3661 RTIC,  RTIL
3560 4210 OPEN,  CCOPEN
3561 7402 HLTO,  HLT
3562 3711 CXBD,  CXB1
3563 3710 BUSY,  BUSY1
3564 0240 SPCC,  0240
3565 7402 HUNCH, 7402
3566 0000 CNTIT, 0000
3567 0000 PRIN,  0000
3570 4400 WRITE, CCPUT
3571 5000 POST,  CCPOST
3572 4200 CLOSE, CCLOSE
3573 3724 POSTIT, POSTR
3574 0000 TCHAN, 0000
3575 0215 CRO,   0215
          *3600
3600 0000 INCH,  0000
3601 7200        CLA
3602 6031        6031
3603 5202        JMP .-1
3604 6036        6036
3605 6046        6046
3606 1210        TAD CASC
3607 5600        JMP I INCH
3610 7520 CASC,  7520
3611 0000 CHLF,  0000
3612 7200        CLA
3613 1220        TAD CR1
3614 4253        JMS OUT
3615 1221        TAD LF1
3616 4253        JMS OUT
3617 5611        JMP I CHLF
3620 0215 CR1,   0215
3621 0212 LF1,   0212
3622 0000 MSG,   0000
3623 7200        CLA
3624 1240        TAD LOC1
3625 3241        DCA CNT1
3626 1242        TAD NUM1
3627 3243        DCA NUM21
3630 4211        JMS CHLF
3631 7200 A011,  CLA
3632 1641        TAD I CNT1
3633 2241        ISZ CNT1
3634 4253        JMS OUT
3635 2243        ISZ NUM21
3636 5231        JMP A011
3637 5622        JMP I MSG
3640 3644 LOC1,  CHAR1
3641 0000 CNT1,  0000
3642 7771 NUM1,  7771
3643 0000 NUM21, 0000
3644 0324 CHAR1, 0324
3645 0256        0256
3646 0301        0301
3647 0256        0256
3650 0240        0240
3651 0275        0275
3652 0240        0240
3653 0000 OUT,   0000
3654 6041        6041
3655 5254        JMP .-1
3656 6046        6046
3657 7200        CLA
3660 5653        JMP I OUT
3661 0000 RTIL,  0000
3662 7106        CLL RTL
3663 7006        RTL
3664 7006        RTL
3665 7006        RTL
3666 7004        RAL
3667 3020        DCA TA
3670 4200        JMS INCH
3671 7106        CLL RTL
3672 7006        RTL
3673 7006        RTL
3674 1020        TAD TA
3675 3020        DCA TA
3676 4200        JMS INCH
3677 7106        CLL RTL
3700 7004        RAL
3701 1020        TAD TA
3702 3020        DCA TA
3703 4200        JMS INCH
3704 1020        TAD TA
3705 3020        DCA TA
3706 4211        JMS CHLF
3707 5661        JMP I RTIL
3710 0000 BUSY1, 0000
3711 0000 CXB1,  0000
3712 7300        CLA CLL
3713 1710        TAD I BUSY1
3714 7650        SNA CLA
3715 5711        JMP I CXB1
3716 6302        6302
3717 5316        JMP .-1
3720 7200        CLA
3721 4723        JMS I SERVE
3722 5312        JMP CXB1 +1
3723 4315 SERVE, CCSERV
3724 0000 POSTR, 0000
3725 1310        TAD BUSY1
3726 7001        IAC
3727 3310        DCA BUSY1
3730 1710        TAD I BUSY1
3731 3020        DCA TA
3732 4222        JMS MSG
3733 7200        CLA
3734 1020        TAD TA
3735 0370        AND MSK1
3736 7012        RTR
3737 7012        RTR
3740 7012        RTR
3741 7012        RTR
3742 7010        RAR
3743 1374        TAD ASCII
3744 4253        JMS OUT
3745 1020        TAD TA
3746 0371        AND MSK2
3747 7012        RTR
3750 7012        RTR
3751 7012        RTR
3752 1374        TAD ASCII
3753 4253        JMS OUT
3754 1020        TAD TA
3755 0372        AND MSK3
3756 7012        RTR
3757 7010        RAR
3760 1374        TAD ASCII
3761 4253        JMS OUT
3762 1020        TAD TA
3763 0373        AND MSK4
3764 1374        TAD ASCII
3765 4253        JMS OUT
3766 4211        JMS CHLF
3767 5724        JMP I POSTR
3770 7000 MSK1,  7000
3771 0700 MSK2,  0700
```

```
3772   0070  MSK3,   0070
3773   0007  MSK4,   0007
3774   0260  ASCII,  0260
             PAUSE
             *4000
4000   6404  RDCAS,  6404
4001   7200          CLA
4002   6044          6044
4003   7200          CLA
4004   6414          6414
4005   7200          CLA
4006   4737          JMS I MESG2
4007   4740          JMS I INCR
4010   4741          JMS I RTIR
4011   7240          CLA CMA
4012   3533          DCA I RLFPTR
4013   3423          DCA I RTBLE
4014   1133          TAD RLFPTR
4015   3752          DCA I BUSYR
4016   4742          JMS I OPN
4017   2000          2000
4020   4143          HLTC
4021   4744          JMS I CKB2
4022   7200  READA,  CLA
4023   3351          DCA PRI22
4024   1020          TAD TA
4025   3533          DCA I RLFPTR
4026   1021          TAD K2200
4027   3423          DCA I RTBLE
4030   7240          CLA CMA
4031   3421          DCA I K2200
4032   3422          DCA I K2377
4033   1021          TAD K2200
4034   3752          DCA I BUSYR
4035   4745          JMS I READ
4036   2000          2000
4037   4143          HLTC
4040   4744          JMS I CKB2
4041   7240          CLA CMA
4042   3020          DCA TA
4043   1133          TAD RLFPTR
4044   3010          DCA RBFPTR
4045   1346  LOA23,  TAD NUMCB
4046   3347          DCA CNT22
4047   1410  LOA22,  TAD I RBFPTR
4050   3350          DCA CHAR22
4051   1350          TAD CHAR22
4052   1054          TAD KLF
4053   7650          SNA CLA
4054   5247          JMP LOA22
4055   1350          TAD CHAR22
4056   6411          6411
4057   5256          JMP .-1
4060   6416          6416
4061   7200          CLA
4062   1350          TAD CHAR22
4063   1053          TAD KCR
4064   7650          SNA CLA
4065   4331          JMS RCVL
4066   7200          CLA
4067   1350          TAD CHAR22
4070   1061          TAD KHASH
4071   7650          SNA CLA
4072   5304          JMP ENDR2
4073   2347          ISZ CNT22
4074   5247          JMP LOA22
4075   1351          TAD PRI22
4076   7640          SZA CLA
4077   5222          JMP READA
4100   2010          ISZ RBFPTR
4101   2010          ISZ RBFPTR
4102   2351          ISZ PRI22
4103   5245          JMP LOA23
4104   1133  ENDR2,  TAD RLFPTR
4105   3752          DCA I BUSY2
4106   7240          CLA CMA
4107   3533          DCA I RLFPTR
4110   3423          DCA I RTBLE
4111   4753          JMS I CLSE
4112   2000          2000
4113   4143          HLTC
4114   4744          JMS I CKB2
4115   7200          CLA
4116   1354          TAD CHRD
4117   6411          6411
4120   5317          JMP .-1
4121   6416          6416
4122   7200          CLA
4123   4331          JMS RCVL
4124   6402          6402
4125   6412          6412
4126   6042          6042
4127   3032          DCA MODE1
4130   5510          JMP I RSTPTR
4131   0000  RCVL,   0000
4132   6401          6401
4133   5332          JMP .-1
4134   6406          6406
4135   7200          CLA
4136   5731          JMP I RCVL
4137   3622  MESG2,  MSG
4140   3600  INCR,   INCH
4141   3661  RTIR,   RTIL
4142   4210  OPN,    CCOPEN
4143   7402  HLTC,   HLT
4144   3711  CKB2,   CKB1
4145   4463  READ,   CCGET
4146   7602  NUMCB,  7602
4147   0000  CNT22,  0000
4150   0000  CHAR22, 0000
4151   0000  PRI22,  0000
4152   3710  BUSY2,  BUSY1
4153   4200  CLSE,   CCLOSE
4154   0215  CHRD,   0215
             PAUSE
             *4200
4200   0000  CCLOSE,0
4201   4234  JMS CCSET
4202   1754  TAD I CCAIN
4203   0373  AND CCEOT
4204   7640  SZA CLA
4205   5230  JMP CCMOT+11
4206   1366  TAD CCREV
4207   5217  JMP CCMOT
4210   0000  CCOPEN,0
4211   4234  JMS CCSET
4212   1754  TAD I CCAIN
4213   0373  AND CCEOT
4214   7650  SNA CLA
4215   5230  JMP .+13
4216   1365  TAD CCFLP
4217   4262  CCMOT,JMS CCEXI
4220   4332  JMS CCEXIT
4221   4753  JMS I CCAST
4222   0226  AND .+4
4223   7640  SZA CLA
4224   5220  JMP .-4
4225   7410  SKP
4226   7357  7357
4227   7300  CLA CLL
4230   1751  TAD I CCAPIN
4231   3361  DCA CCSPI
4232   3761  DCA I CCSPI
4233   5341  JMP CCEXIT+7
4234   0000  CCSET,0
4235   6002  IOF
4236   3375  DCA CCUNIT
4237   1234  TAD CCSET
```

```
4240  1374  TAD CCM2
4241  3315  DCA CCSERV
4242  1715  TAD I CCSERV
4243  3315  DCA CCSERV
4244  1715  TAD I CCSERV
4245  3751  DCA I CCAPIN
4246  2315  ISZ CCSERV
4247  1715  TAD I CCSERV
4250  3756  DCA I CCAEN
4251  2315  ISZ CCSERV
4252  4753  JMS I CCAST
4253  3754  DCA I CCAIN
4254  5434  JMP I CCSRT
4255  4324  CCEOUT,JMS CCHALT
4256  1756  TAD I CCAEN
4257  3361  DCA CCSP1
4260  1752  TAD I CCASPD
4261  5761  JMP I CCSP1
4262  0000  CCEX1,0
4263  3364  DCA CCSPIC
4264  1375  TAD CCUNIT
4265  1367  TAD CCMD1
4266  3270  DCA .+2
4267  1364  TAD CCSPIC
4270  0000  0
4271  7300  CLA CLL
4272  5662  JMP I CCEX1
4273  0000  CCUIT,0
4274  1375  TAD CCUNIT
4275  1357  TAD CCATAB
4276  3362  DCA CCSPIA
4277  1360  TAD CCDATA
4300  3361  DCA CCSP1
4301  4303  JMS CCPASS
4302  5673  JMP I CCUIT
4303  0000  CCPASS,0
4304  1372  TAD CCM8
4305  3363  DCA CCSPIB
4306  1762  TAD I CCSPIA
4307  3761  DCA I CCSP1
4310  2362  ISZ CCSPIA
4311  2361  ISZ CCSP1
4312  2363  ISZ CCSPIB
4313  5306  JMP .-5
4314  5703  JMP I CCPASS
4315  0000  CCSERV,0
4316  6002  IOF
4317  3375  DCA CCUNIT
4320  4273  JMS CCUIT
4321  1754  TAD I CCAIN
4322  3361  DCA CCSP1
4323  5761  JMP I CCSP1
4324  0000  CCHALT,0
4325  7300  CLA CLL
4326  1370  TAD CCSTOP
4327  4262  JMS CCEX1
4330  7300  CLA CLL
4331  5724  JMP I CCHALT
4332  0000  CCEXIT,0
4333  4753  JMS I CCAST
4334  0371  AND CCMOV
4335  7650  SNA CLA
4336  5255  JMP CCEOUT
4337  1332  TAD CCEXIT
4340  3754  DCA I CCAIN
4341  1375  TAD CCUNIT
4342  1357  TAD CCATAB
4343  3361  DCA CCSP1
4344  1360  TAD CCDATA
4345  3362  DCA CCSPIA
4346  4303  JMS CCPASS
4347  7300  CLA CLL
4350  5715  JMP I CCSERV
4351  4576  CCAPIN,CCPIN
4352  4537  CCASPD,CCSPDA
4353  4531  CCAST,CCSTAT
4354  4575  CCAIN,CCINTO
4355  4600  CCANUM,CCNUMB
4356  4577  CCAEN,CCERET
4357  5060  CCATAB,CCTAB
4360  4575  CCDATA,CCINTO
4361  0000  CCSP1,0
4362  0000  CCSPIA,0
4363  0000  CCSPIB,0
4364  0000  CCSPIC,0
4365  0414  CCFLP,414
4366  0440  CCHEV,440
4367  6301  CCMD1,6301
4370  0410  CCSTOP,410
4371  0004  CCMOV,4
4372  7770  CCM8,-10
4373  0020  CCEOT,20
4374  7776  CCM2,-2
4375  0000  CCUNIT,0
            *CCLOSE+200
4400  0000  CCPUT,0
4401  4766  JMS I CCBSET
4402  1776  TAD I CCPIN
4403  3763  DCA I CCBNUM
4404  2763  ISZ I CCBNUM
4405  4767  JMS I CCBSEX
4406  1354  TAD CCWRYT
4407  4770  JMS I CCBEXI
4410  1355  TAD CCLDR
4411  3763  DCA I CCBNUM
4412  4771  JMS I CCBEXT
4413  1356  TAD CCLEAD
4414  4772  JMS I CCBOUT
4415  2763  ISZ I CCBNUM
4416  5213  JMP .-3
4417  1357  TAD CCSYNC
4420  4772  JMS I CCBOUT
4421  1776  TAD I CCPIN
4422  4661  JMS I CCATUF
4423  4772  JMS I CCBOUT
4424  1776  TAD I CCPIN
4425  4772  JMS I CCBOUT
4426  1360  CCUTOP,TAD CCREC
4427  4661  JMS I CCATUF
4430  1262  TAD CCMID
4431  4772  JMS I CCBOUT
4432  1360  TAD CCREC
4433  4772  JMS I CCBOUT
4434  1360  TAD CCREC
4435  3763  DCA I CCBNUM
4436  2376  ISZ CCPIN
4437  1776  TAD I CCPIN
4440  4772  JMS I CCBOUT
4441  2763  ISZ I CCBNUM
4442  5236  JMP .-4
4443  2376  ISZ CCPIN
4444  1776  TAD I CCPIN
4445  7650  SNA CLA
4446  5251  JMP .+3
4447  4320  JMS CCHAN
4450  5226  JMP CCUTOP
4451  4772  JMS I CCBOUT
4452  4772  JMS I CCBOUT
4453  4772  JMS I CCBOUT
4454  4773  CCEND,JMS I CCBHLT
4455  4771  JMS I CCBEXT
4456  4320  JMS CCHAN
4457  4331  JMS CCSTAT
4460  5765  JMP I CCBSEE
4461  5035  CCATUF,CCTUF
4462  0360  CCMID,360
4463  0000  CCUET,0
```

```
4464  4766  JMS I CCBSET
4465  7201  CLA IAC
4466  1776  TAD I CCPIN
4467  7640  SZA CLA
4470  5343  JMP CCDELY
4471  4331  JMS CCSTAT
4472  7300  CLA CLL
4473  1362  TAD CCREED
4474  4770  JMS I CCBEXI
4475  4774  JMS I CCBIN
4476  4774  JMS I CCBIN
4477  4774  JMS I CCBIN
4500  4774  CCSTOP, JMS I CCBIN
4501  4774  JMS I CCBIN
4502  7300  CLA CLL
4503  1360  TAD CCREC
4504  3763  DCA I CCBNUM
4505  2376  ISZ CCPIN
4506  4774  JMS I CCBIN
4507  3776  DCA I CCPIN
4510  2763  ISZ I CCBNUM
4511  5305  JMP .-4
4512  2376  ISZ CCPIN
4513  1776  TAD I CCPIN
4514  7650  SNA CLA
4515  5254  JMP CCEND
4516  4320  JMS CCHAN
4517  5300  JMP CCSTOP
4520  4520  CCHAN, 0
4521  7240  CLA CMA
4522  1360  TAD CCREC
4523  1376  TAD CCPIN
4524  3337  DCA CCSP2A
4525  3737  DCA I CCSP2A
4526  1776  TAD I CCPIN
4527  3376  DCA CCPIN
4530  5720  JMP I CCHAN
4531  0000  CCSTAT, 0
4532  7300  CLA CLL
4533  1764  TAD I CCBUN
4534  1352  TAD CCCMD4
4535  3337  DCA .+2
4536  1353  TAD CCAC3
4537  0000  CCSP2A, 0
4540  3337  DCA CCSP2A
4541  1337  TAD CCSP2A
4542  5731  JMP I CCSTAT
4543  4767  CCDELY, JMS I CCBSEX
4544  1361  TAD CCM12
4545  3763  DCA I CCBNUM
4546  4774  JMS I CCBIN
4547  2763  ISZ I CCBNUM
4550  5346  JMP .-2
4551  5271  JMP CCBET+6
4552  6304  CCCMD4, 6304
4553  0400  CCAC3, 400
4554  0471  CCUNTT, 471
4555  7735  CCLDR, -43
4556  0125  CCLEAD, 125
4557  0167  CCSYNC, 167
4560  7602  CCREC, -176
4561  7764  CCM12, -14
4562  0461  CCREED, 461
4563  4600  CCBNUM, CCNUMB
4564  4375  CCBUN, CCUNIT
4565  4347  CCBSEE, CCEXIT+15
4566  4034  CCBSET, CCSET
4567  4605  CCBSEX, CCSEEX
4570  4262  CCBEX1, CCEX1
4571  4332  CCBEXT, CCEXIT
4572  4726  CCBOUT, CCOUT
4573  4324  CCBHLT, CCHALT
4574  4740  CCBIN, CCIN
            * CCLOSE+375

4575  0000  CCINTO, 0
4576  0000  CCPIN, 0
4577  0000  CCEXET, 0
4600  0000  CCNUMB, 0
4601  0000  CCLINK, 0
4602  0000  CCWK1, 0
4603  0000  CCWK2, 0
4604  0000  CCWK3, 0

4605  0000  CCSEEX, 0
4606  1205  TAD CCSEEX
4607  3200  DCA CCNUMB
4610  4273  CCSFWD, JMS CCADD
4611  1770  TAD I CCCPIN
4612  3204  DCA CCWK3
4613  1203  TAD CCWK2
4614  7041  CIA
4615  7100  CLL
4616  1604  TAD I CCWK3
4617  7450  SNA
4620  5600  JMP I CCNUMB
4621  3202  DCA CCWK1
4622  7420  SNL
4623  5255  JMP CCRVSE
4624  1202  TAD CCWK1
4625  1353  TAD CC1
4626  7470  SNA SZL
4627  5210  JMP CCSFWD
4630  3202  DCA CCWK1
4631  4771  JMS I CCCHLT
4632  4772  JMS I CCCEXT
4633  1360  TAD CCSTFV
4634  3204  DCA CCWK3
4635  1357  TAD CCFFWD
4636  4773  JMS I CCCEX1
4637  1202  CCFAST, TAD CCWK1
4640  7041  CIA
4641  3202  DCA CCWK1
4642  4772  JMS I CCCEXT
4643  4774  JMS I CCCSAT
4644  7012  RTR
4645  7620  SNL CLA
4646  5775  JMP I CCCEUT
4647  2202  ISZ CCWK1
4650  5242  JMP .-6
4651  1204  TAD CCWK3
4652  4773  JMS I CCCEX1
4653  4772  JMS I CCCEXT
4654  5210  JMP CCSFWD
4655  1202  CCRVSE, TAD CCWK1
4656  1355  TAD CC3
4657  7620  SNL CLA
4660  1356  TAD CC4
4661  1202  TAD CCWK1
4662  1354  TAD CC2
4663  3202  DCA CCWK1
4664  4771  JMS I CCCHLT
4665  4772  JMS I CCCEXT
4666  1366  TAD CCSTRV
4667  3204  DCA CCWK3
4670  1367  TAD CCREVS
4671  4773  JMS I CCCEX1
4672  5242  JMP CCFAST+3
4673  0000  CCADD, 0
4674  7300  CLA CLL
4675  1273  TAD CCADD
4676  3204  DCA CCWK3
4677  1361  CCTRY, TAD CCRD0
4700  4773  JMS I CCCEX1
4701  4340  JMS CCIN
4702  4340  JMS CCIN
4703  7110  CLL RTR
4704  7012  RTR
4705  7010  RAR
```

```
4706  3203  DCA CCWK2
4707  4340  JMS CCIN
4710  1203  TAD CCWK2
4711  3203  DCA CCWK2
4712  4340  JMS CCIN
4713  7112  CLL RTR
4714  7012  RTR
4715  7010  RAR
4716  0365  AND CCHIGM
4717  3202  DCA CCWK1
4720  4340  JMS CCIN
4721  1202  TAD CCWK1
4722  1203  TAD CCWK2
4723  7640  SZA CLA
4724  5277  JMP CCTRY
4725  5604  JMP I CCWK3
4726  0000  CCOUT,0
4727  3204  DCA CCWK3
4730  1326  TAD CCOUT
4731  3201  DCA CCLINK
4732  4777  JMS I CCCLIN
4733  1204  TAD CCWK3
4734  0363  AND CCLOW
4735  4773  JMS I CCCEX1
4736  4772  JMS I CCCEXT
4737  5601  JMP I CCLINK
4740  0000  CCIN,0
4741  7300  CLA CLL
4742  1340  TAD CCIN
4743  3201  DCA CCLINK
4744  4772  JMS I CCCEXT
4745  4777  JMS I CCCLIN
4746  1776  TAD I CCCUN
4747  1362  TAD CCMDA
4750  3351  DCA .+1
4751  0000  0
4752  5601  JMP I CCLINK
4753  7750  CC1,-30
4754  7775  CC2,-3
4755  0100  CC3,100
4756  0023  CC4,23
4757  0450  CCFFWD,450
4760  0410  CCSTPW,410
4761  0460  CCRDB,460
4762  6304  CCMDA,6304
4763  0377  CCLOW,377
4764  0000  CCSP3,0
4765  7400  CCHIGM,7400
4766  0400  CCSTRW,400
4767  0440  CCREVS,440
4770  4576  CCCPIN,CCPIN
4771  4324  CCCHLT,CCHALT
4772  4332  CCCEXT,CCEXIT
4773  4262  CCCEX1,CCEX1
4774  4531  CCCSAT,CCSTAT
4775  4255  CCCEOUT,CCEOUT
4776  4375  CCCUN,CCUNIT
4777  5014  CCCLIN,CCLINE
              *CCLOSE+600
5000  0000  CCPOST,0
5001  4652  JMS I CCDSET
5002  4653  JMS I CCDADD
5003  4655  JMS I CCDHLT
5004  4656  JMS I CCDEXT
5005  1657  TAD I CCDPIN
5006  3200  DCA CCERR
5007  3600  DCA I CCERR
5010  1654  TAD I CCDWK2
5011  2200  ISZ CCERR
5012  3600  DCA I CCERR
5013  5645  JMP I CCDSEV
5014  0000  CCLINE,0
5015  4646  JMS I CCDSAT
5016  1043  TAD CCNORM
5017  7650  SNA CLA
5020  5614  JMP I CCLINE
5021  1647  TAD I CCDLNK
5022  7141  CLL CIA
5023  1246  TAD CCDSAT
5024  7420  SNL CLA
5025  5614  JMP I CCLINE
5026  1650  TAD I CCDERR
5027  3200  DCA CCERR
5030  1600  TAD I CCERR
5031  7650  SNA CLA
5032  5614  JMP I CCLINE
5033  1651  TAD I CCDTAT
5034  5600  JMP I CCERR
5035  0000  CCTUF,0
5036  7004  RAL
5037  7006  RTL
5040  7006  RTL
5041  0244  AND CCMSK
5042  5635  JMP I CCTUF
5043  7373  CCNORM,-405
5044  0017  CCMSK,17
              CCERR=CCPOST
5045  4457  CCDSEV,CCEND+3
5046  4531  CCDSAT,CCSTAT
5047  4601  CCDLNK,CCLINK
5050  4577  CCDERR,CCERET
5051  4537  CCDTAT,CCSPRA
5052  4234  CCDSET,CCSET
5053  4673  CCDADD,CCADD
5054  4603  CCDWK2,CCWK2
5055  4324  CCDHLT,CCHALT
5056  4332  CCDEXT,CCEXIT
5057  4576  CCDPIN,CCPIN
              CCTAD=.
```

APPENDIX D

CONSTRUCT - A SYSTEM FOR THE SYNTHESIS OF MACHINE STRUCTURES SOURCE PROGRAM LISTING

[illegible]

D6

[illegible]

[illegible]

APPENDIX E

THE DIGITIZER TO PDP-8/L INTERFACE

Introduction

This interface allows the PDP-8/L to process data that has been generated by the RUSCOM LOGICS Digitizer model 11. The logic for this interface is mounted on 2 boards, both of which are mounted in the PDP-8/L rack. There were several modifications to the logic of the digitizer control to accommodate this interface. The interconnection between the PDP-8/L and the control module on the digitizer is by a cable with 11 twisted pairs of shielded conductors of about 30 feet in length.

The connector terminology employed in this interface is consistent with the standard connector terminology used by the Digital Equipment Corporation (DEC) as shown in the reference drawings for the DEC DW08 logic converter.

The logic symbology used in this work conforms to the symbology defined in the DEC reference

Computer Instructions

Device code 65 was selected for the digitizer. The I/O specifications for this interface conform to those of all DEC standard peripherals. A complete list of device code assignments within the EDIT terminal is given in Appendix (K). The mnemonic representation for the group 65 instructions have been incorporated into an assembler called MAC 3. This allows the programmer to encode his routines using the appended list of mnemonics for the DEC assembly language PAL 3. Details of the MAC 3 assembler are included in Appendix 4.

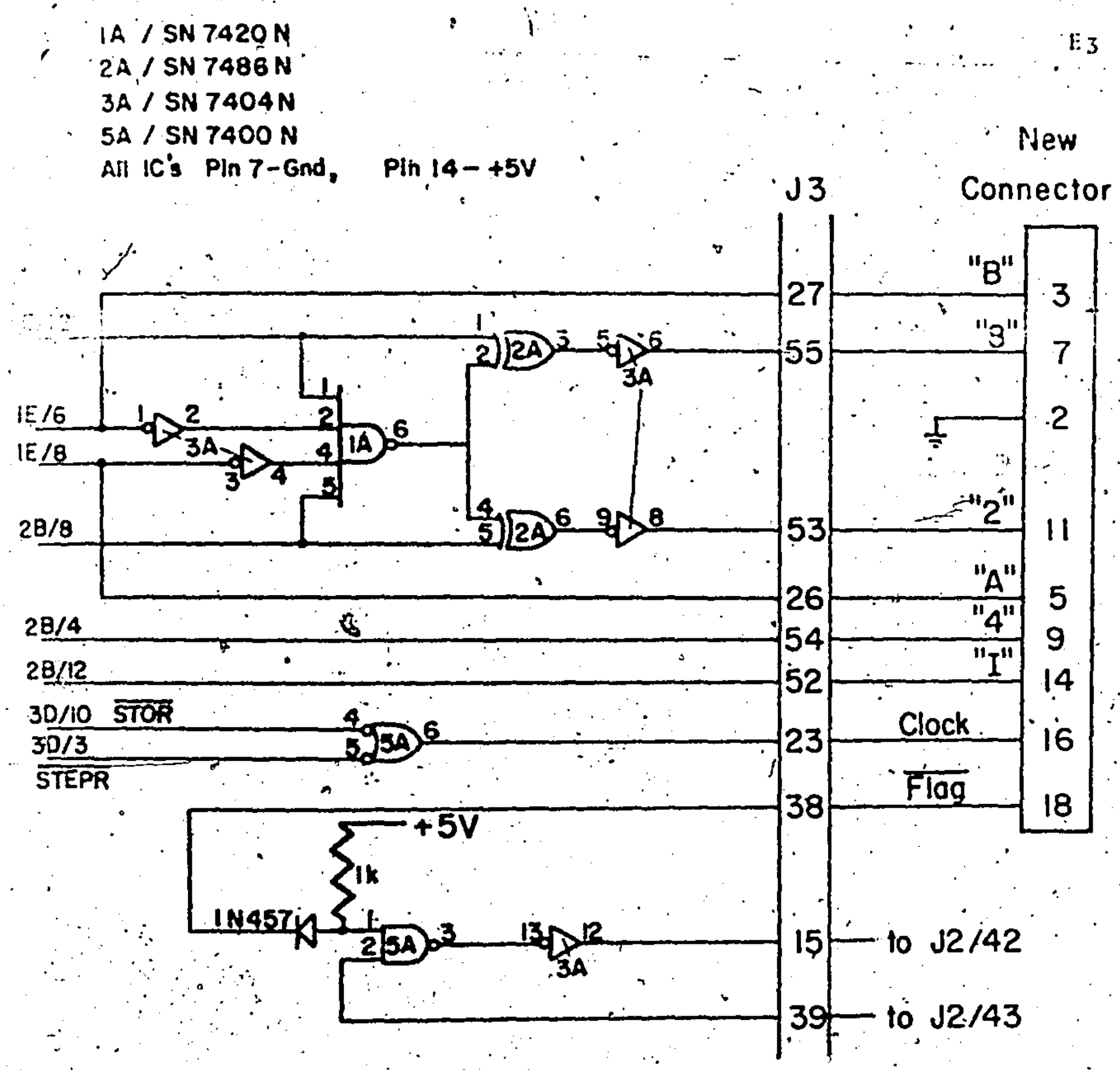
Mnemonic	Octal Code	Description
DSF	6651	Skip if the digitizer flag is a "one". The flag is set to the "one" state when the digitizer is ready to transmit more data.
DCC	6652	Clear the digitizer flag, (set flag to zero state) and clear the accumulator.
DRS	6654	Read from data lines into bits 6-11 of accumulator. Data is coded as 6 bit BCD characters.
DRB	6656	Clear flag and read character into bits 6-11 of the accumulator. The digitizer will set the flag when the next character is available for transmission.

Interrupts

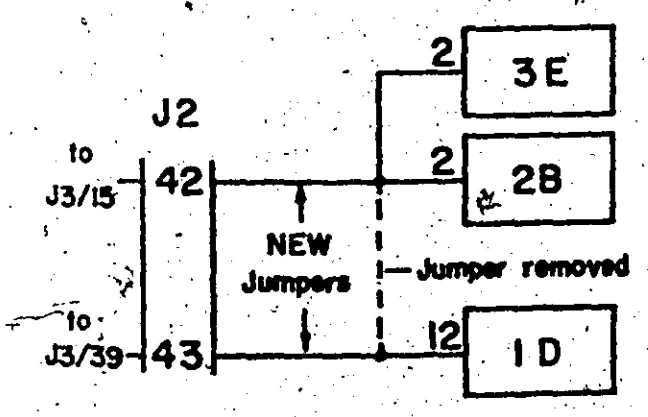
The setting of the digitizer flag will enable the interrupt bus on the PDP-8. Thus characters may be read using the program interrupt facilty.

Details of the Interface Logic and Circuitry

Figure E.1 shows the changes necessary to board 3 of the digitizer logic to make the data lines plus a FLAG signal from the PDP-8/L and a CLOCK signal available to the



Additions to Board 3, RUSCOM Digitizer



Modification to Board 2, RUSCOM Digitizer

Digitizer—PDP 8 Interface

Figure E.1

PDP-8/L. The characters are strobed out of the digitizer by its own I/O pulse. The data output rate can be selected between 300 and 10 characters/second by means of thumbwheel switches on the front control panel of the digitizer. All characters transmitted (0-9, + , -R,. ,?) are coded in the BCD character code.

Board 1 (1A22) Figure E.2

The two boards containing the interface logic are located in positions 1A22 and 1A23 within the DEC DW08/A logic converter. The cable connector a W023A is in location 1A24. These locations have been disconnected from their design function of logic level conversion for the data break option on the PDP-8/L.

The INIT pulse generated by the PDP-8/L is received on pin J2. This pulse is used to clear the flag (pin 13 on E4.). The purpose of inverting the INIT pulse (pins 5 to 6 on E1) is to provide a low for compatibility with the $\overline{IOP2}$ pulse at the OR gate between pins 5 and 6 on E2.

Since BMB(3) through BMB(8) are available as high or low on the DW08/A input bus, the device code selector gate E3 has input lines BMB(3), BMB(4), $\overline{BMB}(5)$, BMB(6), $\overline{BMB}(7)$, and BMB(8) connected to pins 3, 4, 5, 6 11, and 12 respectively. Thus a low is produced on pin 8 when device code 65_8 is active. This low is inverted between pins 9 and 8 on E4 and is used to AND the IOP4 pulses into the interface.

The IOP1 pulse AND the output pulse from the device selector becomes the check flag pulse CHFL at pin H2.

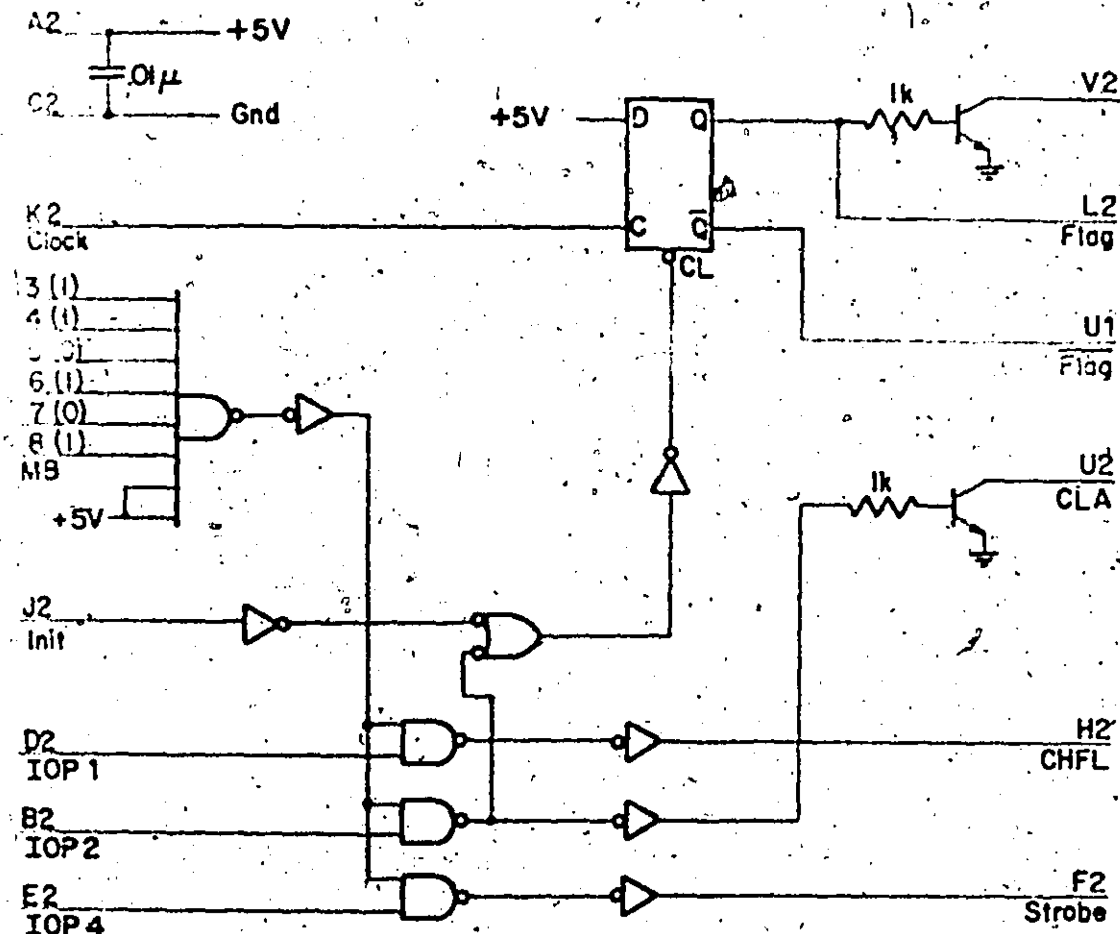
The IOP2 pulse AND the output pulse from the device selector is used after inversion between pins 2 and 8 on E1 to turn on transistor Q2 which clears the PDP-8's accummulator. This pulse is also ORed with the $\overline{\text{INIT}}$ pulse, inverted between pins 3 and 4 on E1, to clear the interface flag at pin 13 on E4. This produces the $\overline{\text{FLAG}}$ pulse on pin U1.

The IOP4 pulse AND the output pulse from the device selector after inversion between pins 5 and 12 on E1 becomes the STROBE pulse on pin F2.

The CLOCK pulse generated by the digitizer is received on pin K2 and is used to set the interface flag to the "one" state at pin 11 on E4. This pulse produces a high on pin 9 of E4 which turns on the transistor Q1. This enables the interrupt bus at pin V2. The output of the flag (FLAG) is available on pin L2.

<u>Board 2 (1A23) Figure E.3</u>

The STROBE pulse is received on pin V2 where it is ANDed with the data lines from pines B2, E2, H2, P2, S2 and U2 to strobe the data into the accummulator on pins D2, F2, J2, N2, R2 and T2.



BOARD 1, DIGITIZER – PDP-8 Interface

Figure E.2

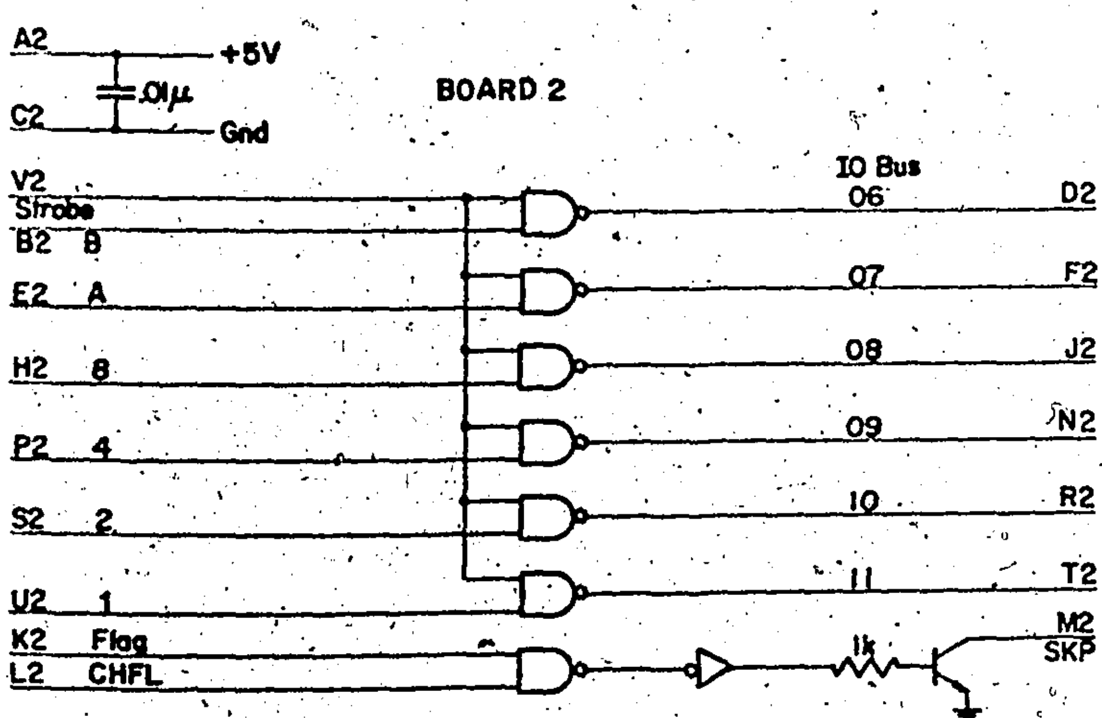
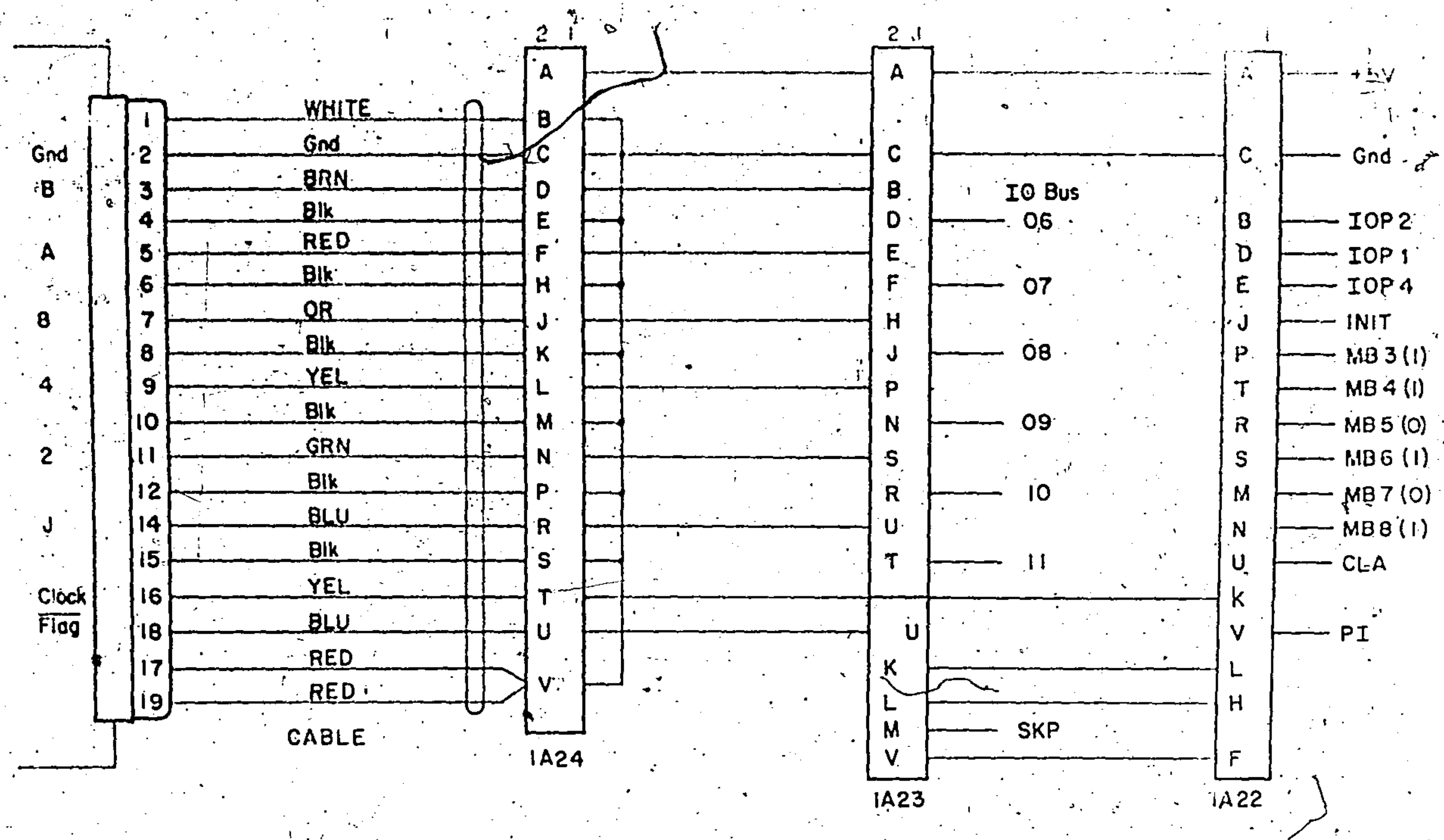


Figure E.3

The FLAG level is ANDed with the check flag pulse CHFL from pins K2 and L2 respectively. The low produced on pin 8 of E1 is inverted between pins 5 and 6 of E1 to turn on transistor Q1. This enables the skip bus on pin M2.

The interconnections between modules are shown in Figure E.4.

The board layouts are shown in Figures E.5 (board 1) and E.6 (board 2) respectively.



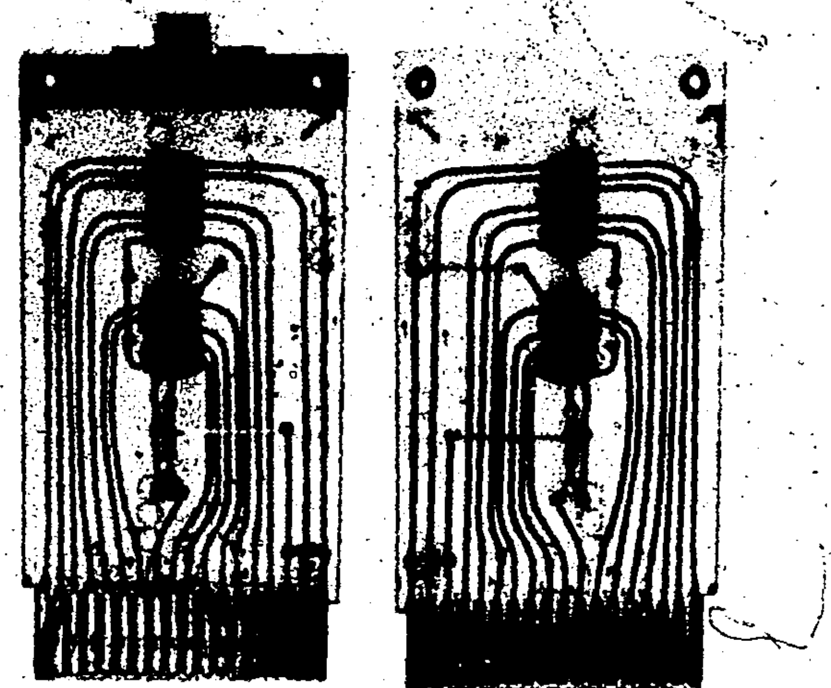
RUSCOM DIGITIZER — PDP8 INTERFACE Interconnection

Figure E.4

Side 1 Side 2

Board 1

Figure E.5



Side 1 Side 2

Board 2

Figure E.6

APPENDIX F

THE EAI 3500 DATAPLOTTER TO PDP 8/L INTERFACE

Introduction

This interface allows the PDP-8/L to control the EAI 3500 Dataplotter. The logic for the interface is mounted on four cards - three are located in the PDP-8/L rack and one is in the plotter D/A drawer. The interconnection is via an 11 twisted pair of individually shielded cables of 50 feet in length.

The connection terminology is consistent with the definitions established in the plotter reference manual for connections within the plotter (76). Connections within the PDP-8/L rack conform to the standard connector terminology used by the Digital Equipment Corporation (DEC) as shown in the reference drawings for the DEC DW08 logic converter.

The logic symbology used in this work conforms to the symbology defined in the DEC reference

Computer Instructions

Device code 60_8 was selected for the plotter. The I/O specifications for this interface conform to those of all DEC standard peripherals. A complete list of device code assignments within the EDIT terminal is given in Appendix K The mnemonic representation for the group 60 instructions have been incorporated into an assembler called MAC 3 to allow utilization of mnemonic code representation when constructing a program in the DEC assembly language PAL 3. Details of the MAC 3 assembler are included in Appendix L.

Mnemonic Code	Octal Code	Description
ESF	6601	Skip if the plotter flag = 1. (Flag is set to the "one" state when the plotter is ready to receive more data.)
ECF	6602	Clear the plotter flag. (Set the flag to the "zero" state.)
EPC	6604	Output bits 6-11 of the accumulator in parallel to the plotter. These bit patterns correspond to the BCD coding specified as the EIA input requirements. The IOP4 pulse generated by this instruction is sent to the plotter as a clock pulse to strobe the data into the plotter.
ELS	6606	Clear the plotter flag and output bits 6-11 of the accummulator in parallel to the plotter. The flag is reset after a delay of approximately 240μ seconds. (Produces an IOP2 and an IOP4 pulse.)

Interrupts

The plotter, after processing the data in the input buffer, will activate the interrupt bus on the interface card. Thus the programmed interrupt facility can be employed to output data to the plotter.

Details of the Interface Logic and Circuitry

The three cards for the interface, located in the

PDP-8/L rack, are in locations 1A19, 1A20 and 1B24 of the DW08/A bus converter. These locations have been dewired from the original application as logic level converters for the data-break option on the PDP-8/L.

Card FB1 (1A19) Figure (F.1)

The initialize pulse (INIT) is received on pin P2 inverted by gate E3 by typing pins 1 and 2 to the input high to produce an output low on pin 3. This $\overline{INIT}$ or the $\overline{IOP2}$ pulse is inverted by the gate between input pins 9 and 10 and output pins 8 on E3. The CLEAR FLAG pulse is then produced at pin R2.

Since BMB(3) through BMB(8) are available as high or low on the DW08/A input bus, the device code selector gate E3 has inputs of BMB(3) and BMB(4) in pins 1 and 2 and $\overline{BMB}(5)$ thu $\overline{BMB}(8)$ on pins 5, 11, 12 and 4 producing a logical low on pin 8 when device code 60_8 is active. The low produced by the device selector is inverted at gate E2 (input pins 1 and 2) producing a high on pin 3.

This high is ANDed with the IOP1, IOP2 and IOP4 pulse producing lows at pins 6, 8 and 11 of the gates in E2 respectively.

The low on pin 6 of E2 is inverted between pins 5 and 6 of E4 to become the CHECK FLAG pulse on pin U2.

The low in pin 8 of E2 is ORed with the INIT pulse between pins 5 and 6 of gate E3. This, upon being inverted

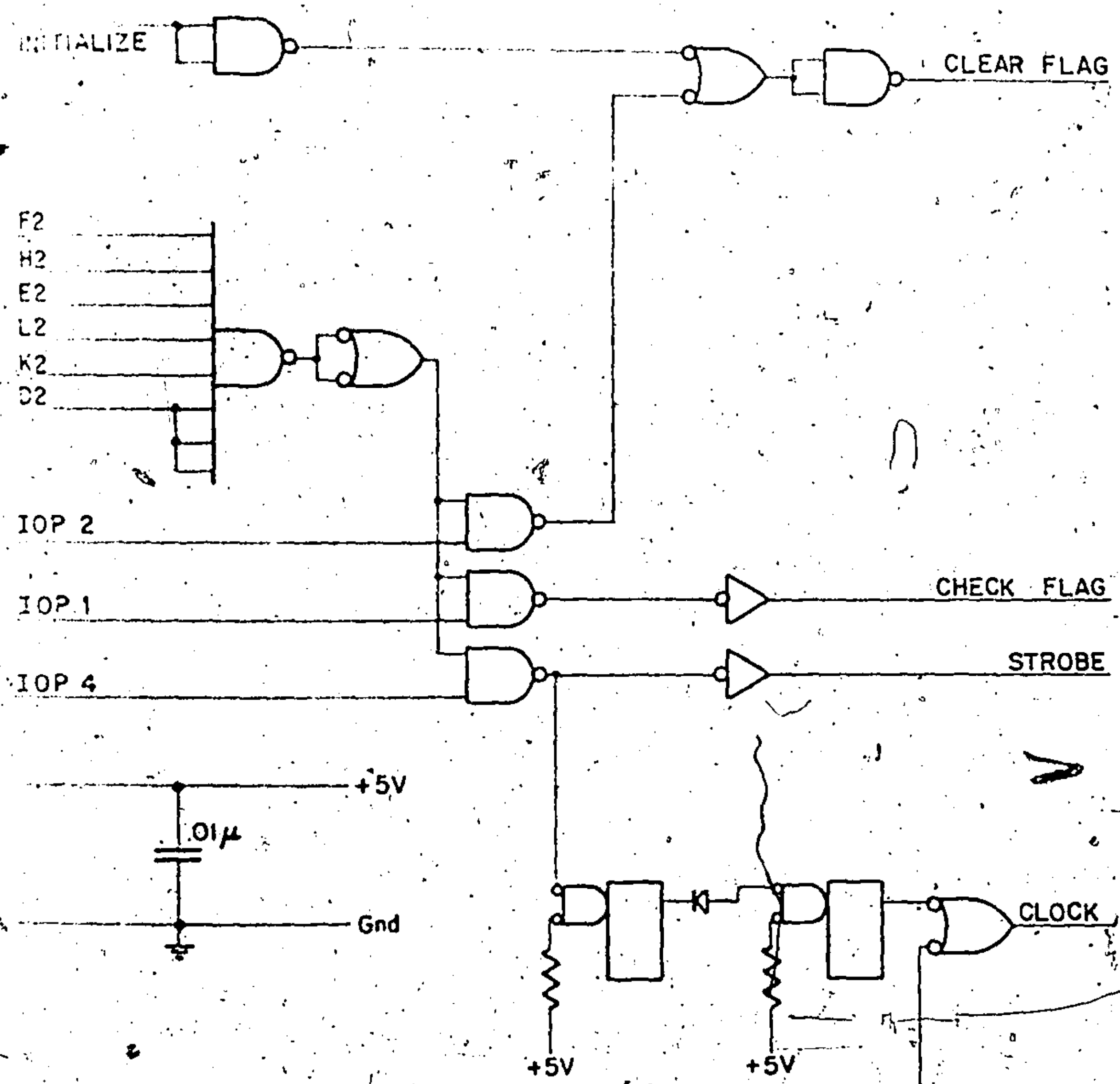


Figure F.I Board FB1

between pins 9, 10 and 8 on E3, becomes the CHECK FLAG on pin E2.

The low produced by the IOP4 pulse at pin 11 of E2 is inverted between pins 1 and 2 of E4 to become the strobe pulse on pin V2. This low also serves as an input to the RC clock between pins 1 and 12 of E4 delaying the pulse by 240μ sec. This delayed pulse, after inversion between pins 12 and 11 of E3 becomes CLOCK pulse on pin S2.

The purpose of the first stage of the RC clock network is to create a 240μ second pulse delay while the second stage reshapes and widens the pulse sufficiently to trigger the flag located on card FBP2.

Card EBP2 (1A20) Figure (F.2)

The CHECK FLAG pulse received a pin E2 is inverted between pins 1 and 3 of E1. The high produced of pin 3 of E1 is ANDed with a high produced at the anode of diodes D1 thru D3. If both conditions are high the resulting low on pin 8 of E1 is inverted at pin 11 of E1. This high then turns on transistor Q1 enabling the skip bus on pin F2.

The CLOCK pulse (IOP4 delayed by 240μ seconds) will set the Flip-Flop E2 producing a high at pin 5 of E2. This high in the presence of a high on pin H2 (data request signal from the plotter) will cause a high to occur at the anodes of diodes D1 thu D3. This high will turn on transistor Q2 which enables the interrupt bus on pin J2.

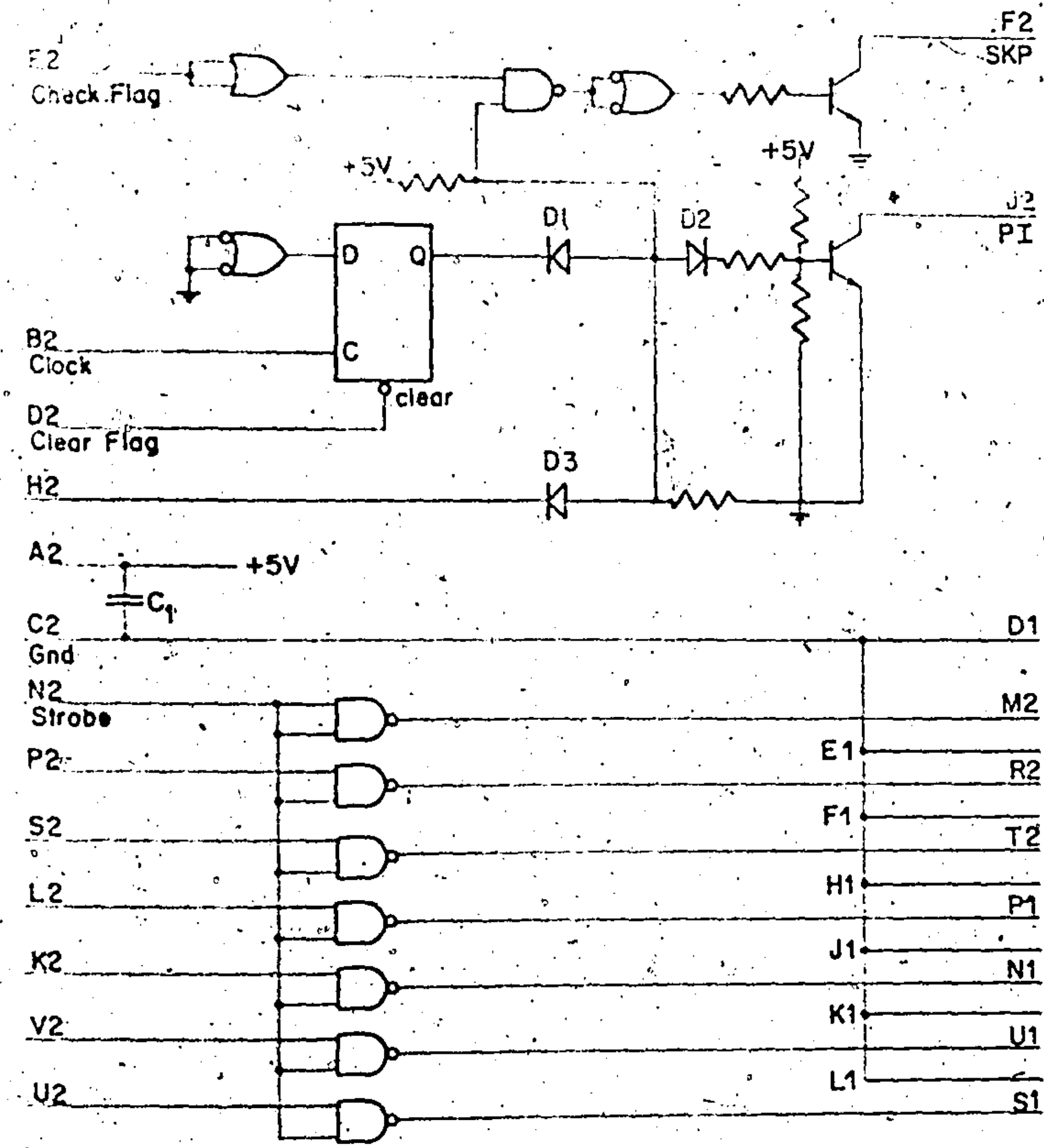


Figure F.2 Board FBP2

The DATA REQUEST signal produced by the plotter is received on pin H2. This signal is held low while the data plotter is processing previous data (upon the receipt of a plot pulse only). Thus the plot data is transmitted to the plotter at the maximum rate (limited by the plotter input circuit) of one character every 240 μ seconds. When a plot command has been received by the plotter the DATA REQUEST signal (H2) is held low until the plotter has completed the previous operation: Note that the input data to the plotter is buffered at the plotter thus allowing two complete instructions to be processed by the plotter at the higher data rate before the DATA REQUEST goes low. The plotting command structure is defined in reference (76).

The STROBE pulse received on pin K2 is ANDed with the logic levels from the accumulator (on pins P2, S2, L2, K2, V2, and U2) causing a character to be transmitted on pins M2, R2, T2, P1, N1, U1 and S1.

CARD FBP3 (J14B) FIGURE (F.3)

This card serves to parallel the input from the magnetic tape reader and the PDP-8/L. Since the six data circuits are identical, only the circuit for input level (32) received on pins 11 and F producing output on pin Y is described.

Data received by the plotter from the PDP-8/L is on pin 11 while data received from the tape drive is on pin F. The signal from the tape drive 0-(+10V) is passed through

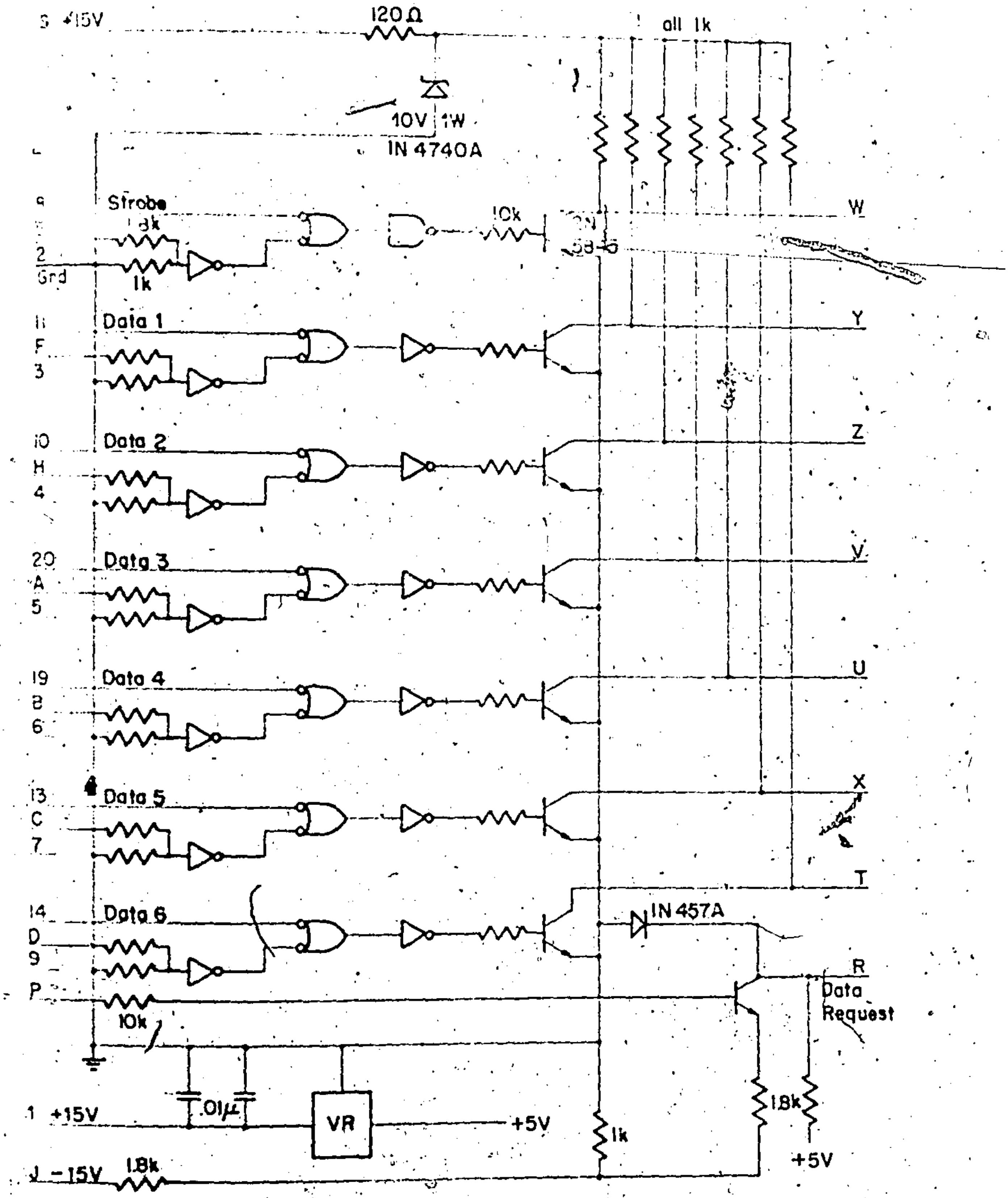


Figure F.3 Board FBP3

potential divider and inverted between pins 3 and 4 of E2. The resulting low or the low produced by the computer is inverted between pins 3 and 4 of E5. The high turns on transistor Q6 producing a logical high of +10V on output pin Y.

The DATA READY signal applied on pin P at [illegible] (-10V to 0 V) level is inverted by transistor Q8 to a (+5V to gnd) level available on pin R.

All internodule connections are as described in Figure F.4.

The component layouts for boards FBP1, FBP2 and FBP3 are shown in Figures (F.5), (F.6) and (F.7) respectively.

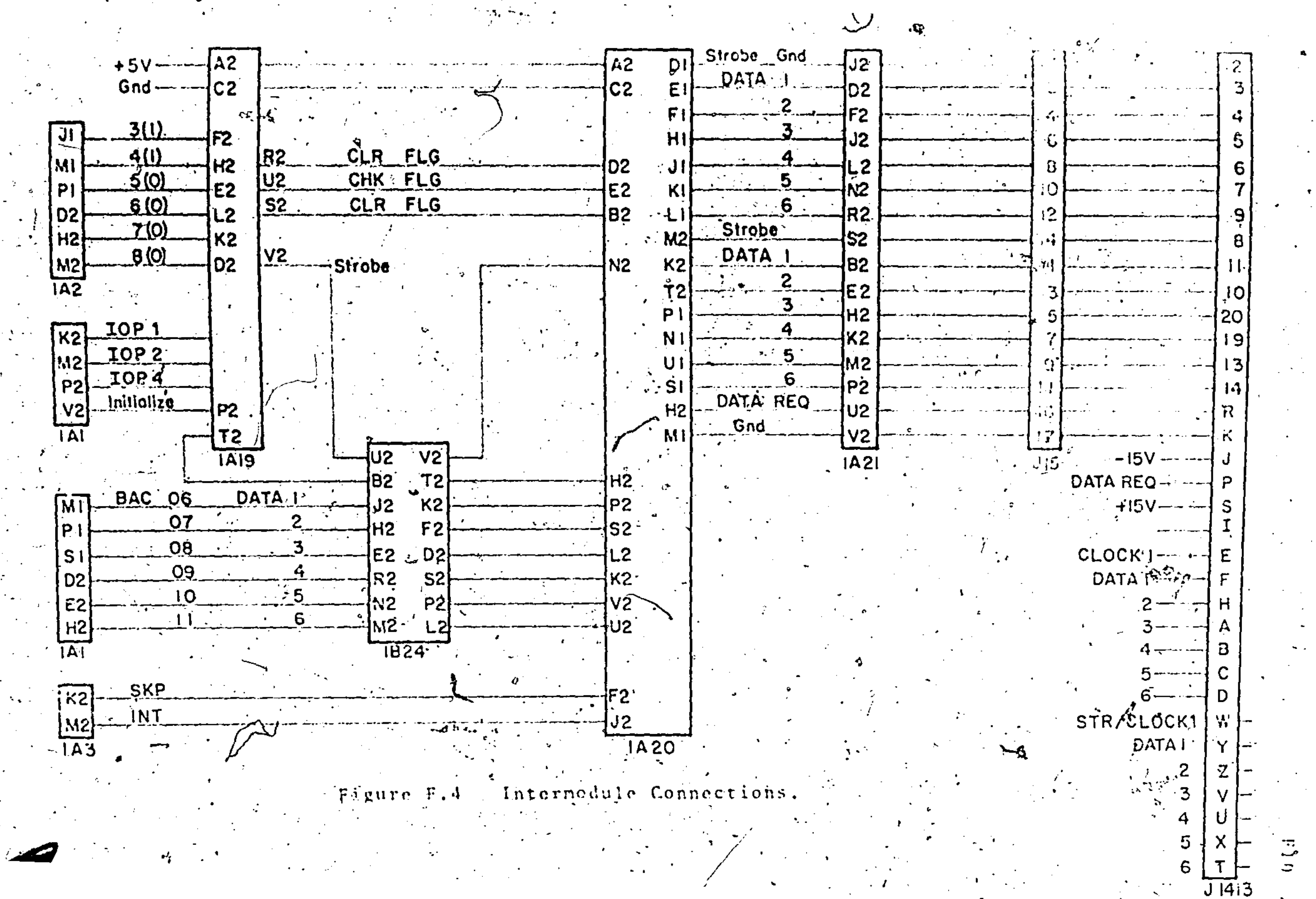
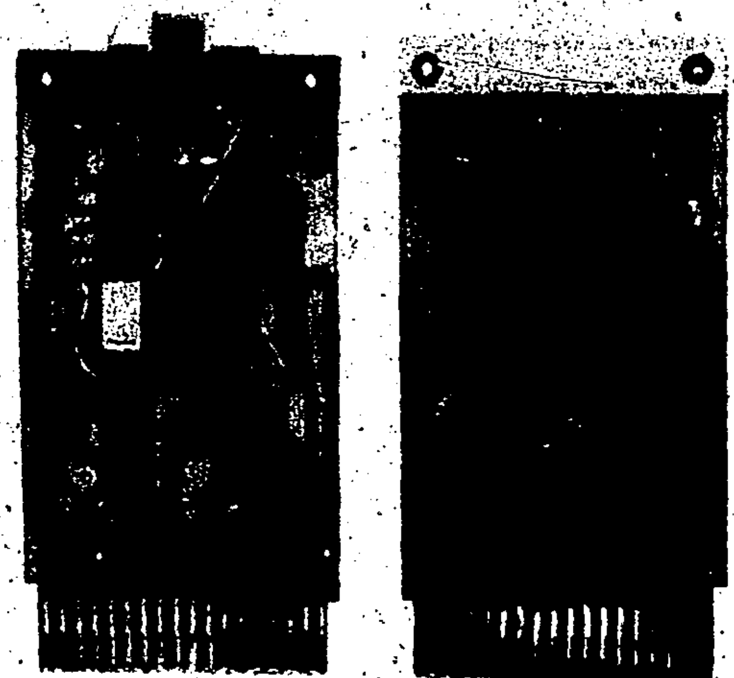


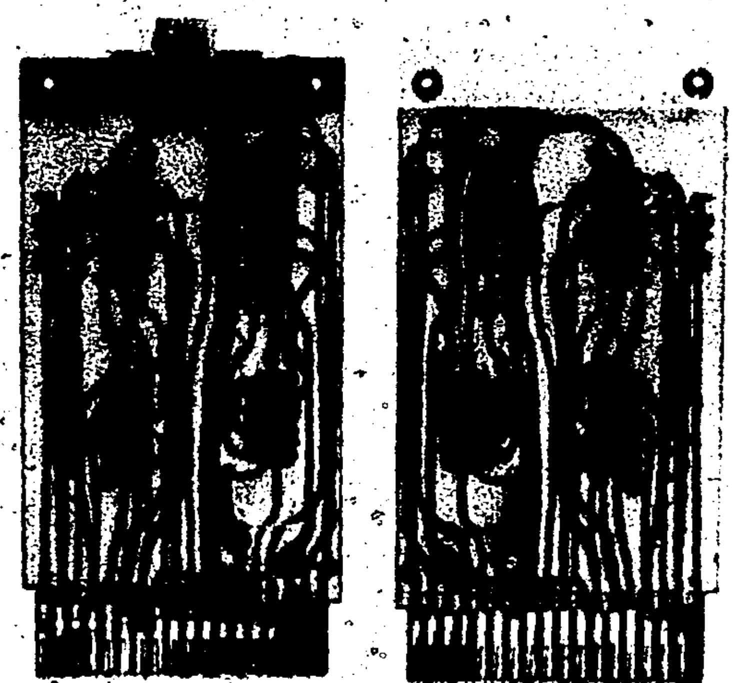
Figure F.4 Intermodule Connections.



Side 1 Side 2

Board FB1

Figure F.5.



Side 1 Side 2

Board FB2

Figure F.6.

Side 1

Side 2

Figure F.7 Board FBP3.

APPENDIX G

PDP8 TO CALCOMP PLOTTER INTERFACE

Introduction

At the time of writing, the interface is still in the early stages of development. This Appendix describes the work to date.

The logic of the interface will be on 3 boards located in the DH08 peripheral slots B22-25. The connection to the plotter will be an 8 twisted pair shielded cable of about 20 feet.

Programming

The device code selected for the plotter is 45. The following instructions have been incorporated in an assembler called MAC3.

DPS	6451	Skip if the drum plotter flag is a one.
DPO	6452	Clear the flag.
DPP	6454	Output bits 8-11 of the accumulator to the drum plotter.
DPA	6456	Clear the flag and output bits 8-11 of the accumulator. Where: bit 7 1 - Pen down; 0 - Pen up

bit 8	1	-	Pen left
	0	-	Pen right or stop
bit 9	1	-	Pen right
	0	-	Pen left or stop
bit 10	1	-	Drum up
	0	-	Drum down or stop
bit 11	1	-	Drum down
	0	-	Drum up or stop

The proposed logic is given in Figure G.1.

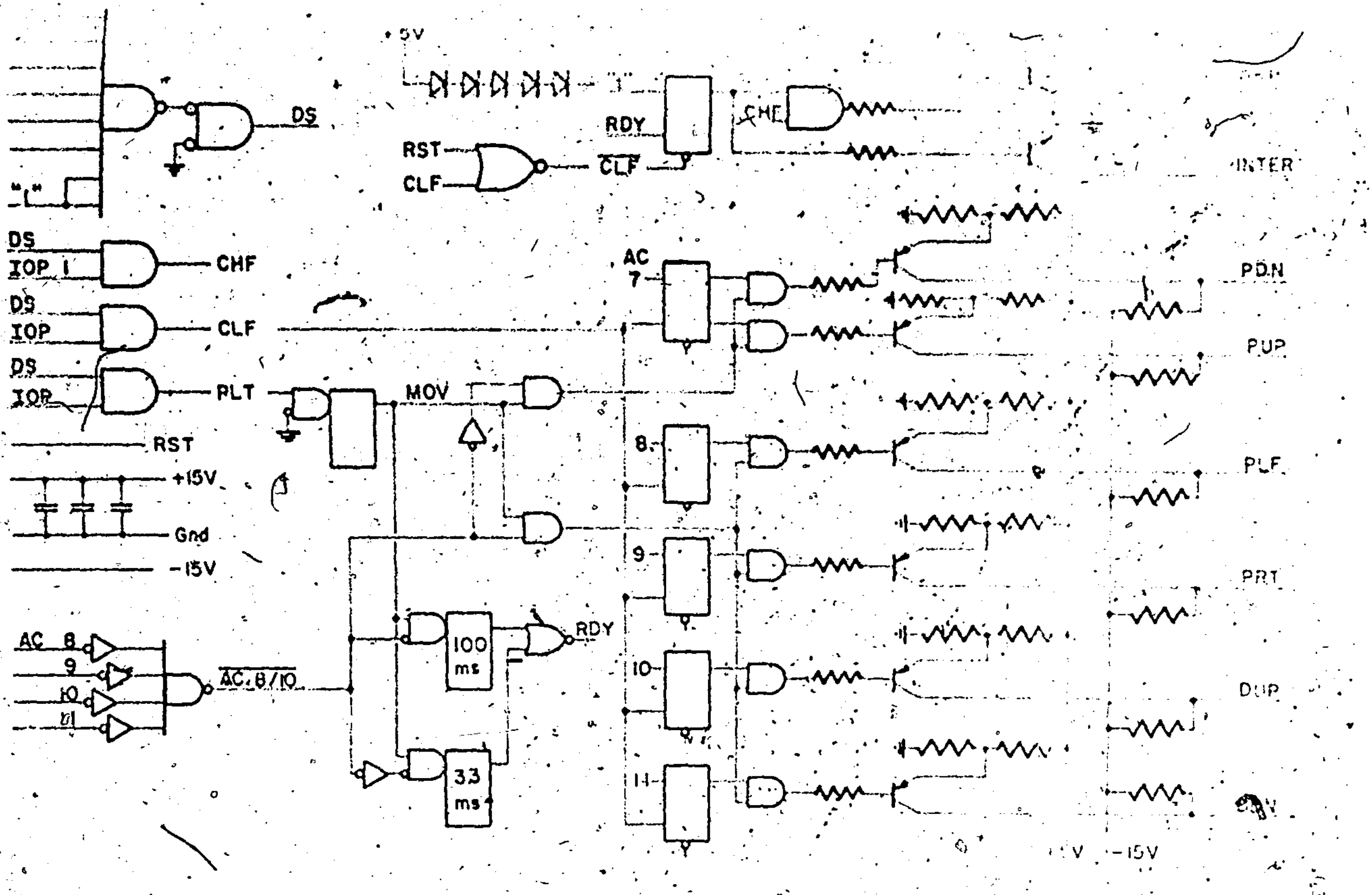


Figure G.1 Drum Plotter Interface.

APPENDIX B

PDP8 TO LIGHT PEN INTERFACE

[illegible]

[illegible] the S_0 connector on the AXO8 peripheral through a BNC connector installed on the rear of the AXO8 rack. The execution of an AXO8 OTEN instruction grounds pins 17 and 18 on the Tektronix 611 storage scope placing the scope in the write-thru mode. The execution of a scope controller IOT will cause the beam to be intensified. The resulting pulse generated by the pen is received on the analogue to digital converter channel EXTERNAL. With the interrupt facility enabled, this resulting pulse will generate a systems interrupt on the AXO8 interrupt line. Thus the beam position is decoded. A filter has been installed on this line to filter high frequency noise. The logic to invert the SYNC pulse is located in the AXO8 rack in location F1.

APPENDIX I

ASYNCHRONOUS INTERFACE
PDP8/L TO CDC6400

This appendix describes the modifications for the PT08 BFX asynchronous interface to provide switch selectable communication rates.

Step One

Three sockets were added to the PT08 in locations A17, A18 and B17. These were glued into place using contact cement.

Step Two

The following wiring changes were completed:

From	To	From	To
A01B	A18C	B17J	B15J
A01A	A18V	B12J	A17A
B04E	A18E	A17B	A15J
B04D	A18H	B12L	A17C
A18K	B17B	A17D	A15F
A18M	B17C	B12P	B17C
B17B	B05D	B17L	B05U
B12E	B17C	B12S	B17H
B04V	B17D	B17R	B15S
B17E	B16D	B12V	B17P

B12D	B17F	B27R	B15V
B12F	B17P	A03J	A17E
A17F	B04J	A04D	A17H
B01D	B17S	A17J	B02D
[illegible]	B02L	[illegible]	B02D

Step Three.

Break the connection B04D to B05D.

Step Four

A Switchcraft DH multiswitch no. 65041K-206 was mounted on the front panel. A "Flip-Chip" W023A connector with pins K, H and E connected to the switch input stations and pin M to the output. Stations 1 and 2 on the switch have been jumpered.

Step Five

Three chips were constructed as follows:

A.	pin A to pin B	
B.	pin C to pin A	
	F	B
	H	J
	K	L
	M	N
	P	R
	S	T

pin A to pin B

C	[illegible]
E	F
H	J

With the R401 chips in positions A16 and B04, the switch set to A18H, board A (above) in position B17, the interface is set at 110 baud; (configuration A). By removing the R401's from A16 and B04 and inserting an R405 in B16, M708 in B12, M709 in B04, board B in B17 and board C in A17 the interface will run at any of 3 higher data rates (configuration B).

When operating in configuration B the data rates are governed by the crystal clock frequency of 38.4 KHz, and the divider networks on the M709. As an example: with a crystal frequency of 38.4 KHz and a divider network that is divided by 8 on pin B04D and divided by 16 on pin B04E, the three switch positions will correspond to 4800, 600, and 300 baud. Of course, any speed is possible by changing either the crystal clock or divider network.

APPENDIX J

THE ANIMATED MOVIE HARDWARE

The system to produce an animated movie on the EDIT terminal consists of a small electro-mechanical camera driver that has been fabricated to mount under a 16 mm Bolex movie camera on a standard tripod (Figure J.1).

The graphical data that comprises one complete frame is terminated by an ASCII*. This character, when received by the software driver, loads 0001_8 into the accumulator and processes the AX08 IOT instruction OTEN (6344). This produces a logical "one" on the external I/O channel S0 on the front panel of the AX08. This condition is reset by the software driver, after the required exposure time, by loading the accumulator with 1000_8 and executing the IOT XRCL. The pulse produced by this sequence of instructions becomes the TRIP pulse at the external connector S0.

The logic for the interface between the PDP-8 and the camera is located on a single board (vector 873 WE) located adjacent to the camera mounting. The connection to the PDP-8/L rack is via three coaxial cables of approximately 10 feet. All connectors, both at the camera and at the PDP-8/L rack are BNC coaxial connectors.

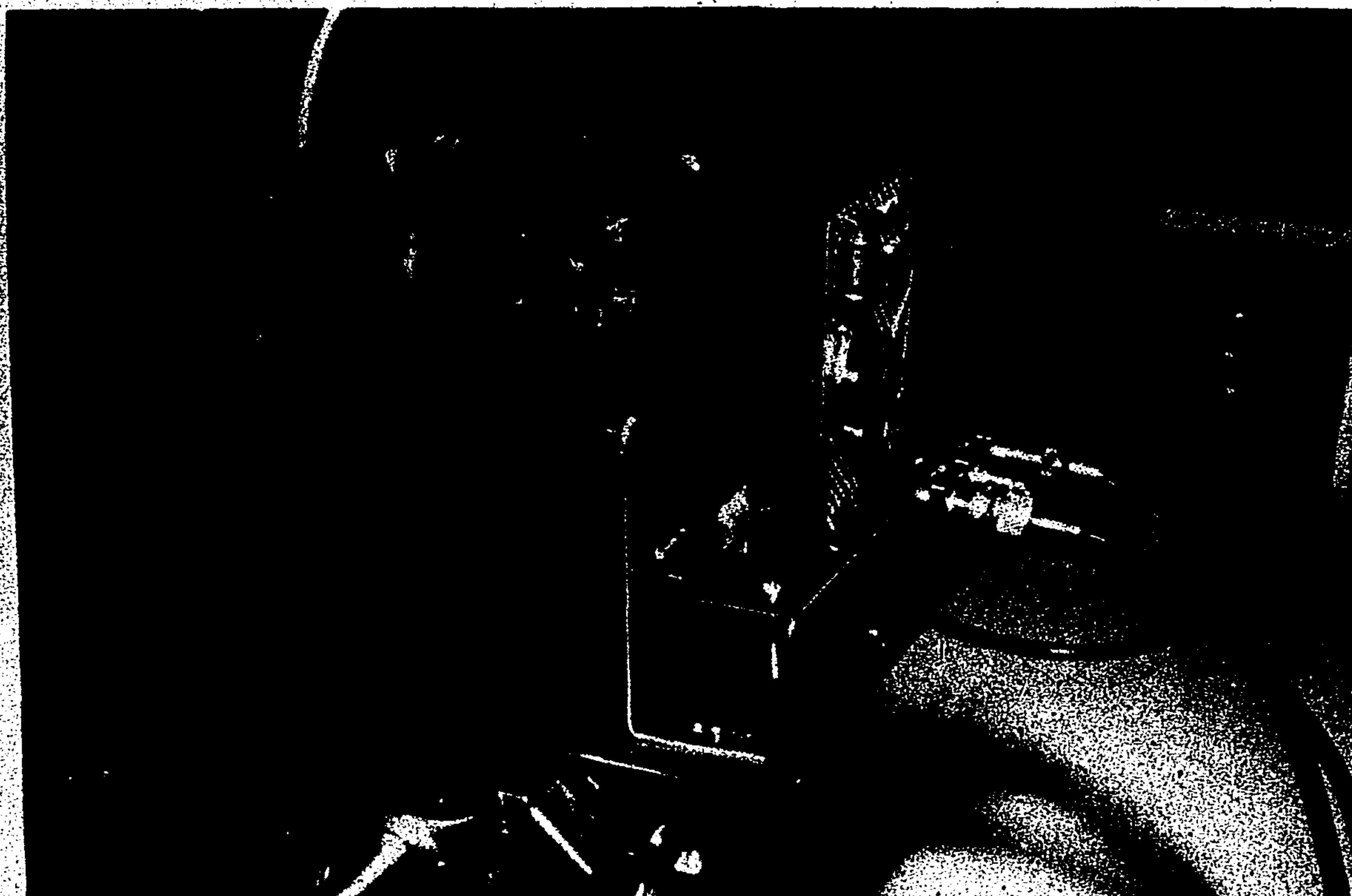


Figure J.1 Camera Driver Showing Interface and Hardware Connections to the Camera.

Figure J.2 shows the camera driver circuit. The TRIP pulse is received on pin 2 which turns on transistor Q_1, cutting off transistor Q_2 and turning on transistor Q_3. The solenoid valve that opens the shutter is connected across pins 6 and 5. The three stage switching circuit was required because of the high power requirements of the solenoid and the low power switching pulse.

The IOT instructions for the generation of the TRIP pulse on the AX08 front sync connector are shown in Figure J.3.

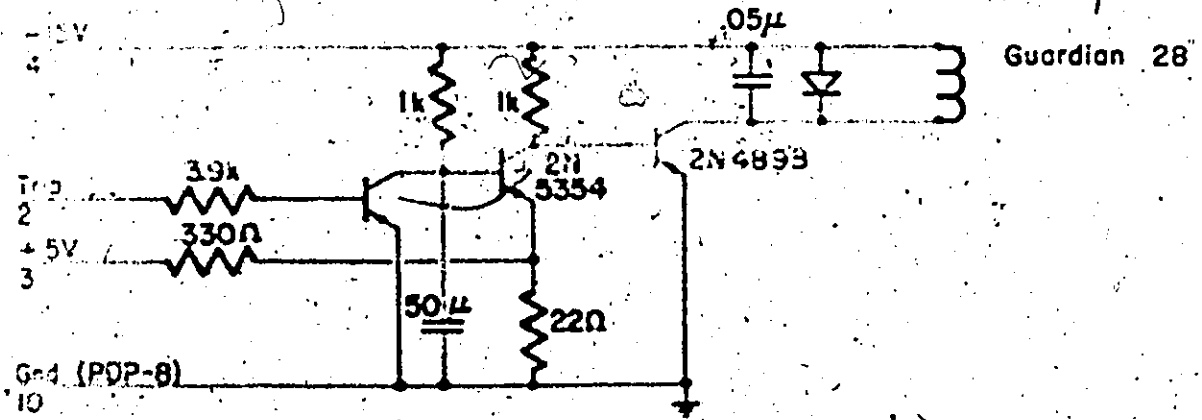


Figure J.2 Camera Driver.

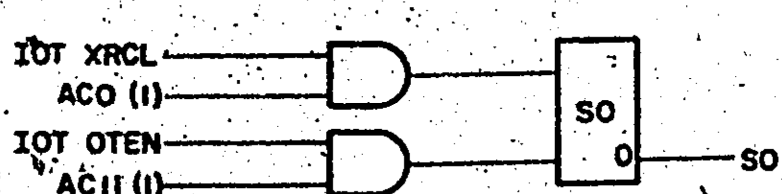


Figure J.3 Sync (TRIP) Pulse Generation.

APPENDIX K

EDIT SYSTEM DEVICE CODE ASSIGNMENTS

The following is a list of the device codes that have been assigned to the peripheral devices associated with the EDIT terminal.

01	High speed paper tape reader
02	High speed paper tape punch
03	Teletype keyboard/low speed paper tape reader
04	Teletype printer/low speed paper tape punch
30	Cassette tape drive
30-35	AX08 Lab peripheral - light pen
70-75	Movie camera
40	PT08 receive channel
41	PT08 transmit channel
45	Calcomp drum plotter
50	CRT/microplotter input output interface
51	Microplotter error flag
60	Flat-bed plotter
65	Digitizer

The AX08 and the cassette tape drive interfaces as produced by their respective manufacturers have conflicting device codes. For the AX08 the device code has been changed from the 30_8 series to the 70_8 series to operate in association with the EDIT system. The device code was made switch selectable in order that the extensive software developed for this peripheral for its design function of data acquisition be negated.

The switch to activate this device code change is located on the DEC module R121 in the AX08 rack location A08. The function of the switch is shown in Figure K.1.

The cassette interface has a pin selectable device code, however, these device codes are all in the 30 series. The SYKES LIBRARY program will only function on a transport with a device code of 30_8.

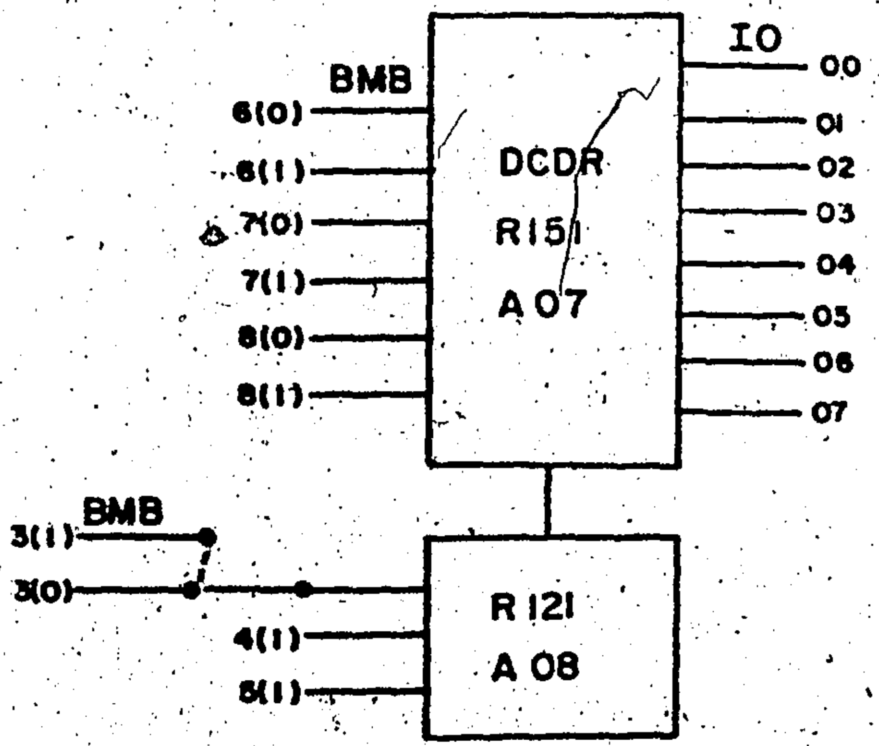


Figure X.1 Modifications to Change AX08 Device Code from 30_8 Series to 70_8.

APPENDIX L

MAC3 ASSEMBLER

Introduction

The MAC3 assembler has been developed to allow the terminal systems programmer to employ mnemonic representations of the IOT instructions for the peripheral interfaces that have been developed in the course of this work. This assembler is an appended version of the DEC PAL III assembler with a modified permanent symbol table to allow mnemonic programming for all the DEC standard peripherals plus the following:

(1) the digitizer;

(2) the flat-bed plotter;

(3) the drum plotter;

(4) PT08 transmit and receive channels;

(5) AX08 instructions related to EDIT peripheral devices.

Operating Instructions

An operating procedure is described below.

(1) Place EDIT system cassette in cassette transport.

(2) Turn on both computer and transport drive.

(3) Set 7777_8 into the Switch Register; press LOAD ADDRESS.

(4) Press START;

(5) When the Teletype responds with READY, Enter the following command at the Teletype keyboard.

L, MAC3, H

followed by CARRIAGE RETURN.

(6) When the MOTION light on the cassette drive flashes, turn off cassette drive and remove EDIT system cassette. From this point the operating procedure is identical to the DEC PAL III assembler.

(7) Place the source language copy of the program to be assembled in the high speed reader.

(8) Set 0200_8 in the Switch Register and press LOAD ADDRESS.

(9) MAC3 is a 3 pass assembler. Set bits 0 and 1 of the Switch Register for the proper pass. These settings are:

Bit 0	Bit 1	
0	1	pass 1
1	0	pass 2
1	1	pass 3

Pass 1 is required so that the assembler can initialize its symbol table and define all user symbols. After pass 1 has been completed either pass 2 or pass 3 can be completed.

(10) Set input and output options with bit 11 as follows:

pass 1	Bit 11 = 1	Punch the symbol table on the high speed punch.
	Bit 11 = 0	Print (punch) the symbol table on the Teletype.
pass 2	Bit 11 = 1	Punch binary tape on high speed punch
	Bit 11 = 0	Punch binary tape on low speed punch
pass 3	Bit 11 = 1	Punch source listing on high speed punch
	Bit 11 = 0	Print (punch) source listing on Teletype.

Bit 2 switch options for passes 1 and 3

Bit 2 = 1 Suppress output of symbol table.

2 = 0 Output symbol table

(11) Turn on high speed punch.

(12) Press START to begin pass 1 only. Press CONTINUE to begin passes 2 or 3. The assembler halts at the end of each pass. Proceed from step 3. If the assembler has halted because of a PAUSE statement, put the next tape into the reader and press CONTINUE.

(13) If the program executes correctly the binary tape can be added to the EDIT system library with the following instructions. Place the write enable plug in the cassette, place cassette in cassette transport.

(14) Set 7777_8 in Switch Register, press LOAD ADDRESS and START.

(15) When Teletype responds with READY, load binary tape in high speed reader and enter the following command at the Teletype:

C, ABCD followed by CARRIAGE RETURN.

where: ABCD is new program mnemonic.

(16) When teletype responds with INPUT, enter H followed by CARRIAGE RETURN.

(17) When Teletype responds with TAPES, enter the number of binary tapes associated with this program followed by CARRIAGE RETURN.

(18) When MOTION light flashes, turn off tape drive, remove cassette and remove write-enable plug from cassette.

APPENDIX H

CONSTRUCT - CONVERSATION LISTING

CONSTRUCT - A SYSTEM FOR THE SYNTHESIS OF MACHINE STRUCTURES

```
ENTER SECTION NUMBER                                      1

PART ONE - GEOMETRY DEFINITION STAGE

THE NUMBER OF STRUCTURAL NODES =                          16

DEGREES OF FREEDOM/NODE=                                  3

ENTER (1) FOR PLANE OR (2) FOR SPATIAL FRAME              1

COORDINATE DATA INPUT SECTION

ENTER (1) FOR FPS OR (2) FOR CGS UNITS                    1

SELECT INPUT DEVICE:
ENTER (1) FOR DIGITIZER
      (2) FOR LIGHT PEN
      (3) FOR TELETYPE
      (4) FOR DISC FILE                                   1

NODAL COORDINATE INPUT DATA VIA DIGITIZER

ENTER SCALE VALUE (FT/IN),(F10.4)                         0.634

ENTER (1) FOR ACTIVE STATE OR (2) FOR DEFERRED STATE      1

(1) DIGITIZE ALL NODES IN SEQUENCE FROM 1 TO 16

(2) RESTRICT NODAL COORDINATES TO FIRST QUADRANT

(3) PRESS THE R KEY ON THE CONTROL PANEL TO TERMINATE DIGITIZING

(4) ENTER S FROM TTY TO START DIGITIZER                   S

NODAL CONNECTIVITY SECTION

SELECT INPUT DEVICE:
ENTER (1) FOR TELETYPE
      (2) FROM CASSETTE TAPE
      (3) DISC FILE                                       1
```

NUMBER OF MEMBERS 16

MEMBER TYPES AVAILABLE:
(1) PIN - PIN
(2) FIX - FIX
(3) PIN - FIX
(4) FIX - PIN

MEMBER NO. LOWER NODE NO UPPER NODE NO MEMBER TYPE

FORMAT(4X, I3, 10X, I3, 10X, I3, 10X, I3)

MEMBER NO.	LOWER NODE NO	UPPER NODE NO	MEMBER TYPE
1	1	2	4
2	2	3	1
3	3	4	1
4	4	5	1
5	5	6	1
6	5	7	1
7	3	8	1
8	8	9	1
9	9	10	1
10	10	11	1
11	11	12	1
12	12	13	1
13	13	14	1
14	14	15	1
15	15	16	1
16	16	9	1

PART TWO - DISPLAY SECTION

SELECT DISPLAY DEVICE:
ENTER (1) FOR FLAT BED PLOTTER
(2) FOR CRT AND MICROPLOTTER
(3) FOR CRT ONLY
(4) FOR DRUM PLOTTER 3

ENTER (1) FOR ACTIVE STATE (2) FOR DEFERRED STATE 1

ENTER (1) FOR SATISFACTORY GEOMETRY
(2) FOR UNSATISFACTORY GEOMETRY 1

PART THREE - LOAD VECTOR STAGE

```
SELECT INPUT DEVICE:
ENTER (1) FOR TELETYPE
      (2) CASSETTE TAPE
      (3) DISC FILE                                      1

ENTER NUMBER OF LOADED NODES                             2

ENTER NODE NUMBER, X-COMPONENT, Y-COMPONENT, MOMENT (I3,3F10.4)
KIPS-FT
  7   0.0       -0.2       0.0
 11   0.0        0.2       0.0


PART FOUR - SECTION PROPERTIES

ENTER (1) TO CREATE FILE OF ELEMENT SECTIONS
      (2) TO USE EXISTING FILE OF ELEMENT SECTIONS
      (3) TO SPECIFY PROPERTIES FOR EACH STRUCTURAL ELEMENT

SELECT INPUT DEVICE:
ENTER (1) FOR TELETYPE
      (2) CASSETTE TAPE
      (3) DISC FILE                                      1

ENTER MODULUS OF ELASTICITY (KIPS/IN**2) (F10.4)

 30000.

ENTER MATERIAL DENSITY (LBS/IN**3) (F10.4)

0.238

ENTER AREA, MOMENT, FOR EACH MEMBER (IN**2, IN**4) (2F10.4)
MEMBER   1          32.62      485.31
MEMBER   2          32.62      485.31
MEMBER   3          18.40      182.35
MEMBER   4          18.40      182.35
MEMBER   5          18.40      182.35
MEMBER   6          27.70      252.26
MEMBER   7          23.26      144.63
MEMBER   8          23.26      144.63
```

MEMBER	9	27.70	252.26
MEMBER	10	1.10	.09
MEMBER	11	1.10	.09
MEMBER	12	5.30	24.99
MEMBER	13	5.67	11.34
MEMBER	14	5.67	11.34
MEMBER	15	27.70	252.26
MEMBER	16	23.26	144.63

SECTION TO REDUCE NUMBER OF ACTIVE DISPLACEMENTS

SELECT INPUT DEVICE:
ENTER (1) FOR TELETYPE
(2) CASSETTE TAPE
(3) DISC FILE 1

ENTER THE NUMBER OF NODES WITH RESTRAINTS 1

ENTER NODE NUMBER, ENTER (1) IF DIRECTION
VALID, (0) IF RESTRAINED - FOR EACH POSSIBLE DIRECTION
FORMAT(4I3)

1 0 0 0

PART FIVE - ANALYSIS

PART SEVEN - DISPLAY OF RESULTS

ENTER (1) FOR CRT AND TELETYPE
(2) FOR FLAT BED PLOTTER 1

ENTER (1) FOR DISPLACEMENTS, FORCES, AND SECTIONS,
(2) FOR DISPLACEMENTS, AND FORCES
(3) FOR DISPLACEMENTS 1

NODE NO	X-DISP(FT)	YDISP(FT),	ROTATION(RAD)
1	0.000000	0.000000	0.000000
2	-0.000000	-0.000000	0.000000
3	-0.000000	-0.000000	0.000000
4	-0.000000	-.000002	-.000005
5	-0.000000	-.000007	-.000007
6	-0.000000	-.000011	-.000007
7	.000003	-.000007	-.000007
8	-.000001	0.000000	.000005
9	-.000005	0.000000	.000011
10	-.000005	.000007	.000012
11	-.000005	.000058	.000026
12	-.000005	.000058	.000011
13	-.000020	.000057	.000022
14	-.000020	.000049	.000034
15	-.000020	.000014	.000019
16	-.000020	0.000000	.000018

ENTER (1) TO CONTINUE 1

MEMBER	AXIAL LOAD	SHEAR LOAD	MOMENT 1	MOMENT 2
1	-0.0000	0.0000	.0013	.0013
2	-0.0000	0.0000	.0013	.0013
3	-0.0000	.2000	-.3183	-.1699
4	-0.0000	.2000	-.1699	0.0000
5	-0.0000	0.0000	0.0000	-0.0000
6	-.2000	0.0000	0.0000	0.0000
7	.2000	0.0000	.3195	.3195
8	.2000	0.0000	.3195	.3195
9	.0520	-.0821	.0933	.0439
10	.0520	-.0821	.0439	-.0378
11	.0520	.1179	-.0378	.0295
12	-.1179	.0520	.0295	.0822
13	-.0520	-.1179	.0822	.0141
14	-.0520	-.1179	.0141	-.0905
15	-.0520	-.1179	-.0905	-.1743
16	.1182	-.0512	-.1743	-.2262

ENTER (1) TO CONTINUE 1

MEMBER	AREA(IN**2)	MOMENT(IN**4)
1	32.6200	485.3100
2	32.6200	485.3100
3	18.4000	182.3500
4	18.4000	182.3500
5	18.4000	182.3500
6	27.7000	252.2600
7	23.2600	144.6300
8	23.2600	144.6300
9	27.7000	252.2600
10	3.5400	.9915
11	3.1500	.7580
12	5.3000	24.9900
13	5.6700	11.3400
14	5.6700	11.3400
15	27.7000	252.2600
16	23.2600	144.6300

ENTER (1) FOR ACTIVE STATE (2) FOR DEFERRED STATE 2

APPENDIX N

FIBON - THE FIBONNACCI SEARCH ALGORITHM USED IN PROGRAM BOTTLE

Introduction

The logic of a single variable search strategy based on the Fibonnacci search strategy has been coded to solve the volume relationship for bottle design described in Chapter 5 of this thesis. This Appendix describes this method and the algorithm employed.

Algorithm

The Fibonnacci search algorithm employs an interval elimination technique that successively narrows the bounds on the function f(X) maximum or minimum value. The search proceeds until those bounds are within a user's defined convergence criterion.

A parameter R is defined as:

$$R = 0.5\ (\sqrt{5} - 1) = 0.618033 \qquad (N.1)$$

The search strategy proceeds in the following pattern to minimize a function f(X) in the range bounded by a_1 and a_2.

1. Let $A_1 = a_1$, $A_2 = a_2$, $H = A_2 - A_1$, $X_{LPT} = A_1 + r^2 H$ and $X_{RT} = A_1 + r\ H$.

2. Compare $f(X_{LPT})$ with $f(X_{RT})$. Go to step 3 or step 4 whichever is appropriate.

3. If $f(X) \leq f(X_{RT})$, let $A_2 = X_{RT}$ and $H = X_{RT} - A_1$.

Stop if H is less than the convergence value. If not, let the new X_{RT} be the previous X_{LFT}, and let the new $X_{LFT} = A_1 + R^2H$. Return to step 2.

4. If $f(X_{LFT}) > F(X_{RT})$, let $A_1 = X_{LFT}$ and $H = A_2 - X_{LFT}$. Stop if H is less than the convergence value. If not, let the new X_{LFT} be the previous X_{RT}, and let the new $X_{RT} = A_1 + R\ H$. Return to step 2.

APPENDIX O

BOTTLE - CONVERSATION LISTING

```
BOTTLE - A SYSTEM TO AUTOMATE BOTTLE DESIGN
------

PART ONE - BOTTLE PARAMETER INPUT SECTION

ENTER (1) FOR BRITISH OR (2) FOR METRIC UNITS                    1

ENTER THE REQUIRED VOLUME IN FL OZS                              48.0

ENTER THE FILL SPACE VOLUME IN FL OZS                            4.0

PART TWO - BOTTLE SHAPE DEFINITION SECTION

ENTER THE OVERALL BOTTLE HEIGHT                                  8.0

ENTER THE DESIGN WEIGHT IN OZS                                   18.0

ENTER THE NECK DIAMETER IN INCHES                                2.0

DIGITIZE BOTTLE SHAPE

ENTER (1) FOR ACTIVE OR (2) FOR DEFERRED STATE                   1

(1) PLACE SKETCH ON DIGITIZER WITH LONG DIMENSION HORIZONTAL

(2) DIGITIZE ONE SIDE ONLY

(3) SET ORIGIN AT CENTER-LINE LEFT SIDE

(4) ENTER R FROM CONTROL PANEL TO TERMINATE DIGITIZING

(5) ENTER S FROM TTY TO START DIGITIZER                          S

PART THREE - VOLUME CALCULATION

SELECT MODIFICATION STRATEGY (1-4)                               2

VOLUME AS DIGITIZED =          100.53088 CU INS.
                    =           57.97963 FL OZS.

THE DETERMINED VOLUME =          52.013 OZS.

PART FOUR - PERSPECTIVE DISPLAY OF COMPUTED BOTTLE

ENTER (1) FOR CRT/HP OR (2) FOR FLAT BED PLOTTER                 1

ENTER THE COORDINATES OF THE OBSERVER (X,Y,Z)
     10.0        10.0        10.0

DO YOU WANT AN APERTURE CARD PRODUCED ?            NO
```

DO YOU WANT THE PLOTTER DATA TO BE PUNCHED ON THE HI SPEED PUNCH ? NO

IS THIS SHAPE SATISFACTORY ? YES

PART FIVE - PREPARATION OF APT INPUT FILE
-COMPLETED-

PART SIX - VERIFICATION OF CL FILE

DO YOU WANT THE CL SEARCHED FOR A SPECIFIC BLOCK ? NO

SPECIFY DESIRED VIEW XZ

IS THE CL FILE CORRECT ? YES

APPENDIX P

MICROPLOTTER TO PDP8 COMPUTER INTERFACE

Introduction

This interface allows the PDP8 to control the microplotter. The logic for the interface is mounted on a circuit card located in the microplotter. The interconnection between the PDP8 and the microplotter is provided by coaxial cables up to 40 feet in length.

Computer Instructions

Three groups of instructions were chosen. Group 50 instructions control the transfer of data from the computer to the microplotter. Group 51 instructions control the mode in which the microplotter is to be used and provide the starting pulses. Group 52 instructions provide interrogation of fault conditions. There is also an interrupt facility provided. A detailed list of the instructions is given below:

Group 50 instructions:

6501 Skip if the microplotter flag is 1. (microplotter flag =1 when the microplotter is ready to receive data). This flag is 0 after a start instruction is given to the microplotter and must be set to a 1 with a 6504 instruction.

6502 Reset the microplotter flag.

6504 Clear the microplotter flag; output bits 4 to 11 of the PDP8 accumulator to the microplotter, and set microplotter flag to 1 when it is next ready to receive more data.

Group 51 instructions:

6511 Skip if the microplotter is ready to receive a start pulse.

6512 Start the microplotter to draw on the storage display unit only.

6514 Start the microplotter to draw on an aperture card and on the storage display unit.

Group 52 instructions:

6521 Skip if a program or parity error has occurred in the microplotter.

6522 Reset the error flag.

6524 Not used.

Interrupts

An interrupt signal will be given to the computer when the microplotter is ready to receive data or when a program or parity error has occurred in the microplotter.