DEVELOPMENT OF AN INTERACTIVE GRAPHICAL SIMULATOR FOR THE IBM 7545 ROBOT

by

Velluva P. Mohandas

Thesis submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

Master of Science

in

Industrial Engineering and Operations Research

APPROVED:

Michael P. Deisenroth, Chairman

Arvid Myklebust

Aseem Chandawarkar

June 1987

Blacksburg, Virginia

DEVELOPMENT OF AN INTERACTIVE GRAPHICAL SIMULATOR FOR THE IBM 7545 ROBOT

by

Velluva P. Mohandas

Michael P. Deisenroth, Chairman

Industrial Engineering and Operations Research

(ABSTRACT)

In this work an enhanced graphical simulator for the IBM 7545 robot running on the AML/E language was developed. The simulator provided two views with the facility to chose either one as the major view, and pan/zoom into that view. It gives the user the facility to define equipment and workcell setups, accepting data in the International Graphics Exchange Specification (IGES) Version 3.0 format. The user can interactively simulate either the complete program, partial program or any subroutine. The system was integrated including the facility to edit, compile, generate cross references, set system configuration, simulate and run the robot either continuously or interactively.

The system was developed on an IBM Personal Computer using two monitors, text and enhanced graphics for maximizing the display surface for the graphics. The programs developed for this work can be broadly classified into the various menu programs, the definition programs to define and display the equipments and workcells, and the graphic programs for driving and displaying the graphics and altering the views.

The limitations and assumptions made in developing this system along with the scope for further work are presented.

Acknowledgements

I would like to thank my advisor Dr. M. P. Deisenroth for his guidance, support and encouragement during the course of this work. My thanks to Dr. A. Myklebust and Dr. A. Chandawarkar for their support and agreeing to be on my committee. I would like to thank my friends and for their suggestions and help throughout this work.

This work is dedicated to my parents, and my loving wife My parents, whose constant encouragement and support in all my endeavours, and my wife who shares in all my aspirations, have made it possible for me to undertake this second graduate program.

My gratitude to Dr. C. E. Nunnally, Assistant Dean for Engineering Computing who not only gave me the assistantship that allowed me to complete this graduate program, but was my guiding force, as only he can be.

Table of Contents

List of Illustrations

1.0 Chapter 1. Introduction

1.1 Background

Robots are being increasingly utilized in industry today to meet the goals of flexibility, productivity and quality. The flexibility of a robot is related to what facilities are available to program and simulate it prior to actual execution. The ability to utilize the robot for different sequence of operations in quick succession determines its flexibility in operation. Robots can be programmed either on-line or off-line. In the on-line method, the robot controller is needed to create the program. The teach pendant mode is an example where the user teaches different positions which are stored in memory and used for executing the sequence of operations. On-line methods requires the robot to stop production resulting in lower yields. It also creates operator safety concerns while teaching the positions.

Disadvantages inherent in on-line programming brought about the introduction of off-line programming concepts and techniques. Here the programming is done elsewhere,

in an environment not requiring the use of the robot controller, and the program is downloaded to the robot controller from the program development system. Off-line programming can be either textual or graphic. In textual offline programming the program is written using a text editor. The programmer manually enters the entire program with all the positions and sequences on a system independent of the robot and subsequently downloads the program to the robot controller. In graphical off-line programming the user actually sees the location of the robot, workpiece and workcell layout on the graphics screen of the development system, while creating the program. Many of the graphical off-line programming systems allow the user to define the equipment in the workcell and to setup the workcell in the desired fashion. The user then creates the program by graphically manipulating the graphical model of the robot and its associated end effector. The program that is created in this manner is usually in the motion command language created by the software system designer and is independent of the robot itself. This program must then be post processed into the language of the robot manufacturer prior to downloading to the robot controller.

1.2 Simulators

In off-line programming, after the program has been developed in textual or graphics mode, it would be desirable to be able to simulate the robot motion graphically and see the sequence of operations prior to downloading to the robot controller. This can result in a substantial saving in time which otherwise would have been lost in debugging the program and modifying the sequence of operations after the program has been downloaded into the production robot. Possible danger to equipment and personnel caused

by motions not anticipated or visualized by the programmer can be avoided by graphical simulation.

Simulators have been developed to simulate programs written either textually or graphically. Most of these simulators are part of off-line graphical programming packages. They allow the user to see the execution of the robot program on the graphics screen and make modifications to the program prior to the actual run. Simulators are also used for computer aided design and research on robots. They are used to study different robot configurations, and select between different available robots and proposed robot designs. Simulators take advantage of computer generated graphics to address specific problem areas in robot actuator design and manipulator design. They are used to study tooling and wrist movements.

1.3 Problem Statement

Graphical simulators capable of modeling different types of robots use their own robot motion language for creating the robot program. Graphical simulators specific to a robot or language use the robot language to create the robot program and graphically simulate the program. In the first case the simulators do not make use of the intrinsic programming capabilities of different languages that have been developed to take full advantage of the specific robot. The robot programmer has to be aware of the capabilities of the simulation language, and is restricted to its capabilites, which are usually limited to path information.

There are simulators that display graphical simulation of programs based on their own simulation language that are capable of accepting a program in the robot language program as input. They then translate the robot language program to their own simulation language and in this process are limited to the capability of this language. Translators required to do this are not available for many languages.

Simulators developed for a specific language and robot make full use of all the capabilities of the robot language. However they have limited viewing and simulation capability, and no provision for defining workcell setup or equipment.

Both of the above types of simulators simulate from the beginning through to the end. The user does not have the facility to interactively do partial simulation of a part of the robot program. The user cannot choose a few lines of program, a particular subroutine in the program, or step through the program one line at a time and and have it graphically simulated. The lack of this facility increases the debugging process and consequently the productivity of the programmer. Another limitation is that most of the currently available simulators need graphics hardware, at least equivalent to that available in independent graphics workstations and do not perform on the popular and economical personal computers.

Hence it can be said that for a person wanting to interactively simulate a robot program on a personal computer in the robot language itself, along with the facility to define workcells and equipment, it would be necessary to develop tailor made systems, unique to the robot and its language.

1.4 Objective

The IBM 7545 robot is a four degree of freedom industrial robot driven by electric motors. The end effector can be vacuum, mechanical or electrical power driven. It is mainly used for pick and place assembly operations. The robot is programmed off-line using the robot language AML/E. The program is created on a personal computer using a text editor, and then compiled using the AML/E compiler before downloading to the robot controller. There is a facility to do limited simulation of the robot program prior to downloading to the robot controller.

The objective of this thesis is to develop a simulator that will be specific to the IBM 7545 robot and AML/E robot language. It will give the user, the facility to define his own equipment, and workcell setups. The user can do partial simulation of the robot program at any stage of program, or for any chosen subroutine. The user will have the option of deciding whether the plan or side view of the robot and its setup will be the major view and which part of the setup must be enlarged during the simulation run. These facilities will make it easier for the user to debug the program, and prevent accidents to the personnel and equipment on the site. It will also aid instructional purposes. The proposed simulator will have the following features as compared to the existing simulator, developed by the International Business Machines (IBM) corporation.

Present Simulator	Proposed Simulator
1. Shows plan view only.	1. Shows both plan and side view, with options to switch between the two as major/minor view.

2. One size view

3. Robot represented as a 2D stick image.

4. Only complete program can be simulated

4. Only the robot and work space is shown

5. Functions as a stand alone program module.

2. Has facility to zoom into any chosen area

3. Robot represented as a 2D wire frame image.

4. Has facility to simulate:

   - Partial
   - Complete
   - Subroutine

4. Equipment and workcell can be defined. Accepts IGES format data files as input.

5. Integrated into the AML/E command menu structure.

2.0 Chapter 2. Literature Review

2.1 Introduction

The field of simulation in robots is closely related to off-line programming, both textual and graphical. This is because most simulators have been developed as part of off-line programming systems, for robots. Computer graphic simulation is a valuable tool for off-line programming(23).

2.2 Commercial Systems

A significant amount of the development work for graphical offline programming and simulation of robots has been done by industry and commercial software organizations. The brief review that follows provides a basic understanding of the currently available systems and their features.

Autosimulations Inc., Utah, have developed an integrated system of software for simulation of cell-factory situations(4). The module that does the robot simulation is called AutoBots. The simulation is done in the simulation language of the system. It requires an independent workstation and is capable of defining multiple robots and equipment.

Dassault Systems, France has developed the general purpose 3D graphics package CATIA, (Computer Aided Three Dimensional Interactive Graphics). It has a module called CATIA Robotics that is capable of simulating any robot. The module consists of two functions, ROBOT function and TASK function.

Robot function is used to design robots and optimizing robot cells. It has nine menus to lead the user through the different items. Some of the menu activity items are Start, Erase, Rename, Transform, Define, Modify, Use, Static, etc. The user can design his own robot in 3D space mode, specifying the arm lengths, number of degrees of freedom, and coordinate system. Facilities are available to find the effect of loads in static condition. A robot that has been previously defined can be modified.

The TASK function has seventeen menus. Depending on whether the program exists or not different menus are chosen. Some of the menu activity items are Start, Erase, Copy, Sensor, Edit, Record, Statement, Motion, Logic, Exit, Task, etc. It is used to create and execute a sequence of actions forming a task. The simulating language is capable of handling I/O sensors, mathematical and logical operations. It has two levels the BASIC output with simple tasks only with motion orders and commands., and the ADVANCED level with logic and sensors. The simulation can be done continuously or step by step for different robot configurations. It has the facility to specify the display in joint interpolation or linear interpolation mode. The user must use his own post processor for language of the particular robot. The software has capabilities for error

correction. It runs on the IBM mainframe computer using the IBM 5080 graphics terminal.

The General Robot Arm Simulation Program GRASP, is both an off-line programming and graphical simulation system. It was developed by Stephen J. Derby at Renesealer Polytechnic Institute(9, 10). It is used for designing new robots, modifying existing ones, off-line programming, and simulation. It is a general purpose simulator and uses a post processor to translate the program to the target robots language.

There is another package with the acronym GRASP developed at Nottingham University, U.K. (35). Here GRASP stands for Graphical Robot Application Simulation. It is a generalized system for different types of robots in varying situations. It has a robot modeller. The simulated position of the robot can be achieved in three ways. (a) Specify the angles particular joints must move through (b) Specify the desired position of the tool and allow GRASP to compute the required angles for the joints. (c) Remember the configuration of the particular point. GRASP can handle several robots and conveyors working at one time, and compute the total cycle time taken.

Deneb Robotics, Michigan introduced IGRIP which stands for Interactive Graphics Robot Instruction program(25, 26). The basics of this package was developed by Mahajan, R., and Dr. Leu, M. C., at Cornell University. It can be described as a fully integrated modular system for workcell layout, off-line device programming and simulation. Its features include real time solid shaded animation of complete workcell environments with multiple concurrently operating devices. The user can define his own devices and robots. They can be displayed in color as shaded objects or wire frame objects. It uses the UNIX operating system and is written in C language. The simulation

is done in a generic simulation language independent of the robot language. The worksation is an independent graphics workstation like IRIS 2000.

Interactive Robot Programming and Simulation System (IRPASS) consists of the Calma CAD system and robot processor of VAL controlled robots(16, 17). Here there is interactive communication between the CAD system and the robot processor. Though it uses the robot processor it can be considered robot independent by using the proper interfaces for each processor. It is capable of sensing I/O operations and collision detection.

KARMA was developed at the University of California at Davis by Professor Dorf, R. C., and doctoral student Krischbrown, R. H.(18, 19). It is an acronym for Knowledge Based Robot Manipulation System. It is based on the Artificial Intelligence technique of knowledge based systems. KARMA draws from a knowledge base of information about the robot and its environment and can makes its own inferences. The user tells only what task to perform. It is user friendly and has a graphic display with icons. The simulation is done in real time.

The U.S. Air Force ICAM (Integrated Computer Aided Manufacturing) contract awarded to McDonnel Aircraft Co., marked the begining of the development of McAuto by McDonnel Douglas Automation Company (13, 15, 21). It consists of four packages, PLACE, BUILD, COMMAND, and ADJUST. PLACE stands for Positioner Layout and Cell Evaluation. It allows the user to compare different robots, different tools, and different objects in the workcell. It can simulate all the movements and analyze cycle times. It allows the user to do computer simulation, user controlled animation, and acess to a library of robots. BUILD allows the user to define robot configurations. The user specifies the degrees of freedom, joint motion types, and limits. COMMAND is for

creating and debugging programs off-line. ADJUST is for compensating differences between the real world and the simulated program.

MnCell was developed at the University of Minnesota. This package allows simulation of several robots and other objects in the workcell(22). It has real time animation and provides collision detection. The user can model his own robot and object. The tasks are described to the robot in a series of procedure calls, and are not intended to control actual cells. Here the accent is on simulation.

SRI International, Ca., developed RCode(27). It is a 3-D multiple robot simulator that is capable of (a) simulating simultaneous motion of multiple robots (b) Present graphics representations of the robot scene (c) perform geometric collisions for the entire volume of each robot arm as it moves, all of the above in real time. It has wire frame graphics simulating capability for moving robot scenes, and shaded polygon capability for static robots. The special feature of this package is collision avoidance. This must be distinguished from collision detection. It is capable of not only detecting collision but also plan a new path once the original path is blocked. It can handle collision between robots with any combination of translational and rotational joints and degrees of freedom. Rcode is written in C language.

Developed by Robotics Research and Development Ca., RoboCam runs on the Apollo Series of computers(8). It can be used to create and edit 3-D wire frame models. Data can be transferred through the IGES interface. RoboCam supports kinematic and dynamic models of the manipulators. It uses a feature called Autoplace to determine reachability. It allows only robots with six degrees of freedom. The debugged and simulated program is translated into Rcode an intermediate language, before translating to the target robots language. General Electrics Calma Co. developed Robot-Sim(32). It

is based on Stephen J. Derbys GRASP developed at Rensselear Polytechnic Institute. The software can be used to design the workcell and define the objects. The user can interactively create and edit the work. The user defines the robot points and motion path. The software is sophisticated in the sense it can compute the velocity, acceleration, torque, and deformation of each joint and end effector. This results in very accurate time cycle predictions and enables proper selection of the robot.

RoboTeach was a software that was under development at General Motors, and made use of the GM Corporate Graphics System (CGS) as the database(34). It consisted of four major subsystems (a) Workcell layout (b) Program Generation (c) Editing System (d) Simulation Subsystem. The simulation subsystem operated on the robots task sequence, and related data in the database. Most of the systems were in the development stages, when the project was discontinued.

2.3 Classification

As discussed there is a wide range of simulators available today. Simulators can be classified in many ways, depending upon the characteristic being compared. They can be classified based on (a) the simulation objective. (b) the graphic display capability, and (c) the simulation capability.

2.3.1 Simulation Objective

Based on the objective for which the simulation is done, simulators can be broadly classified into two types. One class developed for the primary purpose of computer aided design and research of robot characteristics like dynamic simulation and analysis of robot arms, path verification, tooling and manipulator arm movements, and the other class for the purpose of simulating robot motion sequence. In the category of simulators for design purposes fall Robocon, Dram, and HD. Robocon(12) is used to design the actuator drive system of the robot. It is written in C and Lisp, and is menu driven. Dram(27) developed by the University of Michigan is used to study the kinematics and cycle time of robots. The user can see the simulation and plots of tip velocity, acceleration, and force for any time increment. HD(31) is used for the design of robot manipulators, It is Lisp based and has facility for graphics simulation. These simulators are ideal for the researcher or designer, who can define desired arm length, rotational and translational capablities and subsequently compare between available robots or potential designs.

Within the second category we have simulators that run on general simulation languages, like RoboCam and IGRIP and simulators that run on the robot language itself like the IBM simulator. The purpose of these simulators are to graphically display the sequence of motions of the robot program prior to the actual execution in the production facility. This will reduce the downtime of the robot and help prevent accidents.

2.3.2 Display Capability

Simulators can be classified based on the type of workstation and based on the computer system required to simulate them. Based on the workstation they can be classified in terms of resolution and number of colors available. In this class we have the IBM simulator which displays on the IBM Personal Computer's color graphic display monitor with a maximum of 4 colors at one time and a resolution of 320 X 400 . Simulators like McAuto, and IGRIP that can display 256 colors and have a resolution of 1K X 1K fall in the second class. Based on the computer system required to run a simulator we have simulators like McAuto that run on the Vax mini series, IGRIP that runs on independent workstations like the Silicon Graphics IRIS workstation, and the IBM simulator that run on a personal computer. The graphic capability needed for a simulator is determined by the system design, taking the time taken for the simulation, the necessity and economic justification for such a display into consideration.

2.3.3 Simulation Capability

Simulators can be classified based on what is displayed graphically and the nature of the display. Based on what is displayed we have only the robot in one group, and simulators with the robot and entire workcell in another group. The IBM simulator with only the robot on display is a case for the first group. GRASP with capability to define new robots, modify existing ones, and define work place falls in the second group. Other simulators in this group include McAuto and Roboteach. Most simulators fall in the second group. Based on how the robot and its environment is displayed we have two dimensional and three dimensional images as two basic classes. KARMA and the IBM

simulator fall in the first class. KARMA has wire frame images while IBM has a stick representation. In the three dimension image class there is ROBOCAM and GRASP with wireframe displays, and McAuto and IGRIP with solid images. IGRIP has facilities for part of the robot to be in wireframe, solid, and invisible at the same time.

2.4 *Summary*

The literature review presents the state of the art in the field, considering all types of simulators for robots. It is seen that most of the simulators have sophisticated graphical features to display the robot and its workcell environment. In general they require independent workstations or mini-computers to operate. They run on general simulation languages and cannot simulate in the language of the robot, as applied to most simulators. None of the available systems offer the capability to interactively simulate a robot program. Presently there is no system capable of interactively simulating a robot program in the robots own language on a personal computer, at the same time providing facilities to define equipment and setup a workcell.

3.0 Chapter 3. Methodology

3.1 Introduction

The Graphical Simulation and Programming Package (GSAPP) was developed using a batch file to start up and pass control between some of the programs. In general programs were compiled separately and linked together. Control was passed by calls within the programs. The computer programs written in the FORTRAN language for this system are : FIRST, BETWEEN, MENU2, MENU3, ALTER, MAIN, DEPUTY1, DEFEQP, DEFSETUP, DISPEQP, and DIPSSETU. The function of each of this programs is outlined in Figure 2. The flow of control between the different menus is shown in Figure 1.

This system is integrated with the system developed by Jayaraman, R.,(20) for interactive programming of the IBM 7545 robot. The pseudo compiler in that system reads the users AML/E program and generates five files, namely a listing file, a variable file, a constants file, an object file, and a symbol file. These files are then read by the system

control program to interactively run the robot based on the users AML/E program. The existing pseudo compiler was used without any modifications in this research. The system control program at present interacts with the robot communication controller to run the robot. This system control program was modified to become the graphics driver program and drive the graphics display program instead of the robot communications controller. The graphics routines developed for the graphics display program DEPUTY1 include the following major subroutines, MOVE, PLANSIDE, ZOOM, REDRAW, STANDARD, SWITCH, PORTIN, PORTOUT, GRIPPER, and WKSPACE. The function of each of these are explained in Figure 2.

3.2 System Hardware/Software

The GSAPP system is designed to run on the IBM personal computer family PC/XT/AT or the Personal System/2 Model 30/50/60/80. The system needs dual floppy disk drives and 640K of memory. A math coprocessor greatly enhances its performance. This system was developed on the IBM AT with a color graphic adapter supporting a color graphics monitor with a resolution of 320 X 400. The system use two monitors, the other being a monochrome text display. The two monitors are used to be able to display the text and graphics simultaneously, maximizing the graphics display surface.

The graphics display routines used to develop the GSAPP system is FORGRF which stands for Fortran Graphics Routines. It is a library of machine language subroutines.

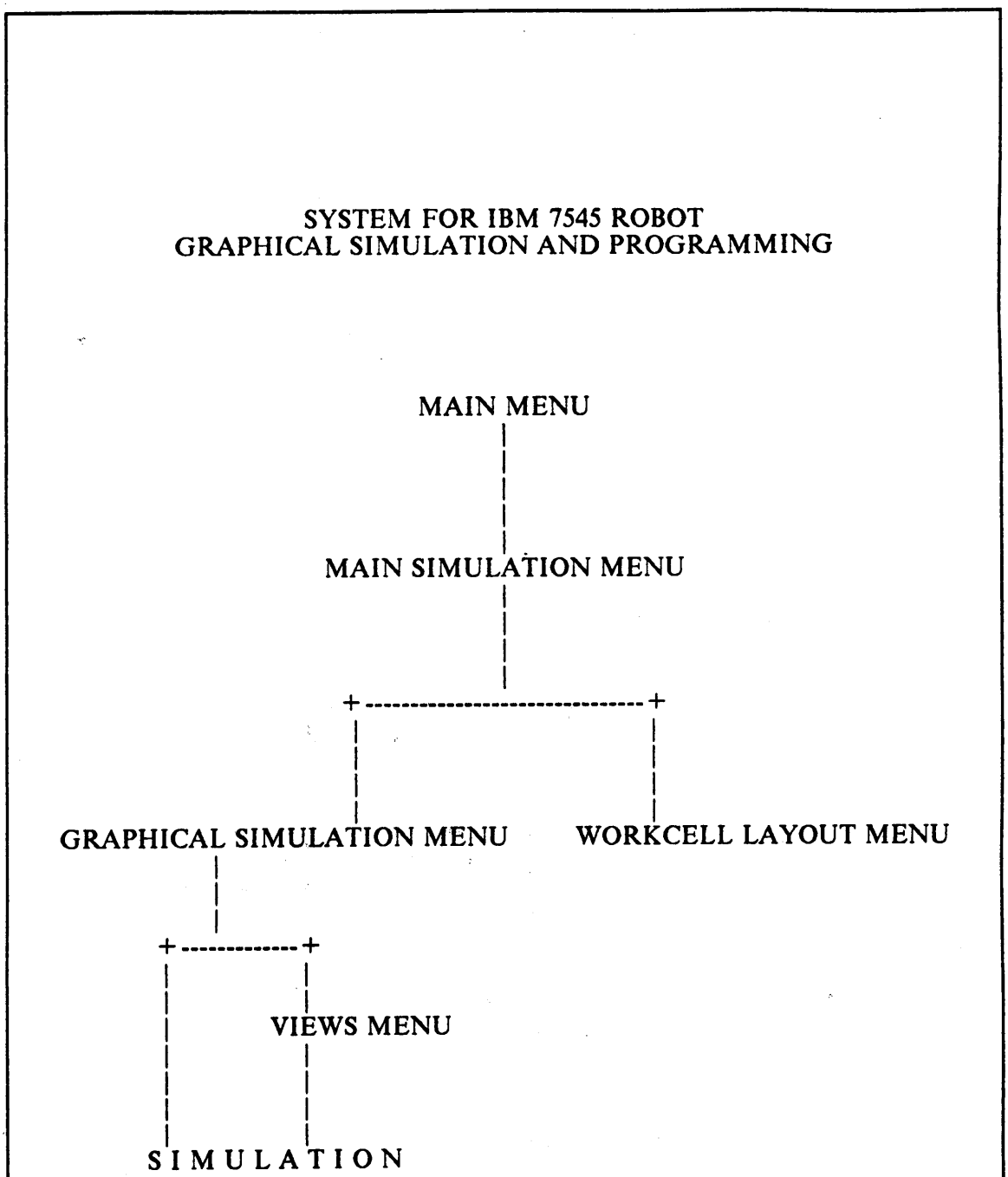

Figure 1. This is the Menus Chart

FILE	FUNCTION
GSAPP.BAT	Initialize system, transfer control between programs, and quit system.
FIRST.EXE	Displays Main Menu, and writes the flags to direct control to appropriate programs.
BETWEEN.EXE	Displays the Main Simulation Menu.
MENU2.EXE	Displays Graphical Simulation Menu and passes control to the program MAIN or the Alter Views menu as chosen.
MENU3.EXE	Displays Workcell Layout Menu and directs to define/display equipment, define/display workcell setup, or display directory as chosen.
ALTER.EXE.	Displays Views Menu, and directs to/Zoom/ /Switch/Normal/Simulation.
DISPEQP.EXE	Define/Display equipment.
DEFSETUP.EXE	Arrange Workcell layout.
DISPSETU.EXE	Display Workcell layout.
DIRECTORY.EXE	Display directory of equipment and workcells.
MAIN.EXE	Graphics driver program that controls the graphics display program.
DEPUTY1.EXE	Graphics display program that does all the computations and displays on the graphics screen.
MOVE	A subroutine in DEPUTY1 that computes the final locations of the arms of the robot in plan and side view.
PLANSIDE	A subroutine in DEPUTY1 that does the actual draw on the graphics screen.
INPORT/OUTPORT	A subroutine in DEPUTY1 that controls the display of the input and output port on the screen

Figure 2. List of Files developed for the Simulator.

It consists of three types of subroutines (1) Tektronix AGII-compatible subroutines, (2) Tektronix TCS-compatible subroutines, and (3) The FORGRF Kernels. Between these three sets of subroutines which are all contained in a single library it is possible to create high resolution bar charts, and line graphs, scale, rotate and have windows. Features to clear screen, set graphics display mode, return the ASCII code of any key, sound a short tone on the speaker, etc are some of the other features available with this set of routines. These routines are called from MicroSoft Fortran version 3.31. The operating system must be MS-DOS Version 2.1 or higher. The FORGRF routines were chosen over the standard Graphics Kernel System, (GKS) to take advantage of its inherent speed of execution. To accommodate the capabilities of FORGRF the drawing must consist of only point and line elements. One of the limitations of the FORGRF package is that color is possible only for points displayed in the screen coordinates. The screen coordinates are defined to be 0 to 1023 in the X direction and 0 to 780 in the Y direction.

The FORGRF set of assembly language subroutines can be called from MS-Fortran. The basic routines are MOVE and DRAW in the user and screen coordinates. The move and draw commands can be absolute or relative. These routines are also capable of a variety of other functions that include the following: (1) CSROFF - Turn the cursor off (2) CSRON - Turn the cursor ON (3) LOCATE - Move the cursor to the desired row/column on the specified page. (4) RDCHART - returns the character and attribute at the current cursor position on the specified page. (5) INKEY - removes the next character form the keyboard buffer and returns its ASCII value (6) CLRBUFF - clears the system's keyboard buffer. All these subroutines are among those used in the development of this system.

3.3 *System Specifications*

The GSAPP system was developed in seven phases. In the first phase the various menus, and batch file was developed. In the second phase the core graphics module that displays the standard view was developed followed by facilities to alter the viewing parameters in the third phase. The fourth phase allows the user to create equipment and define different workcell setups. The fifth phase involved developing the graphics drivers corresponding to the different AML/E commands. The next phase was to integrate these drivers with the existing system control program. The last phase was the development of a test program

3.3.1 Menus and Batch file.

This system can be initialized by typing in GSAPP, the name of the batch file, at the DOS prompt. This displays the MAIN MENU which is displayed in Figure 3. The user has the option of creating an AML/E program, editing and compiling it using the AML/E version 4.1 editor and compiler. The program can be downloaded to the robot controller and run continuously or interactively as desired. The system configuration can be set or a cross reference listing can be generated as required. The option to simulate it graphically causes the main simulation menu to be displayed. The user has the option to set the program name and compiler options in this menu itself.

In the main simulation menu displayed in Figure 4, the user chooses either to simulate the program or to define/display equipment and workcell. The first option displays the

SYSTEM FOR IBM 7545 ROBOT

GRAPHICAL SIMULATION AND PROGRAMMING

MAIN MENU

F1 - Return to DOS

F2 - Edit/Teach a Program

F3 - Compile Program

F4 - Load a Program to Controller

F5 - Set System Configuration

F6 - Communicate with Controller

F7 - Generate Cross Reference Listing

F8 - Set Program Name and Option

F9 - Interactively Run Robot

F10 - Simulate Program Graphically

Select Option = = = = = = >

Figure 3. This is the Main Menu

graphical simulation menu, also shown in Figure 4, while the second option displays the workcell layout menu which is shown in Figure 5. The graphical simulation menu gives the user the option of simulating the program with the existing screen layout or simulating the program with altered views. The standard view is displayed by default. The user can have any workcell setup displayed on the screen by using the set workcell option in the main simulation menu. If the user chooses to alter the viewing parameter, the views menu shown in Figure 5 is displayed.

In both the cases the control is then passed to the pseudo compiler COMPPC which then passes it to the graphics driver program MAIN.FOR to graphically simulate the AML/E program on the graphics screen with the graphics display program DEPUTY1. The user can enter the name of the AML/E program in the main menu, or else will be prompted for it by the pseudo compiler.

3.3.2 Graphic Display Modes

The standard view is invoked when the user decides to simulate the program in the main simulation menu. A wireframe image of the robot in two views, the plan and side view are displayed simultaneously. The workspace of the robot in the two views are also displayed. The position of the gripper in terms of the rotation angle is displayed along with the indication whether it is open or close. The 16 input and output ports are indicated separately. All the above displays are in four different windows with their own local coordinates.

When the user chooses the alter view option in the graphical simulation menu, switch views, zoom/pan, and standard view will be available as options in the views menu that

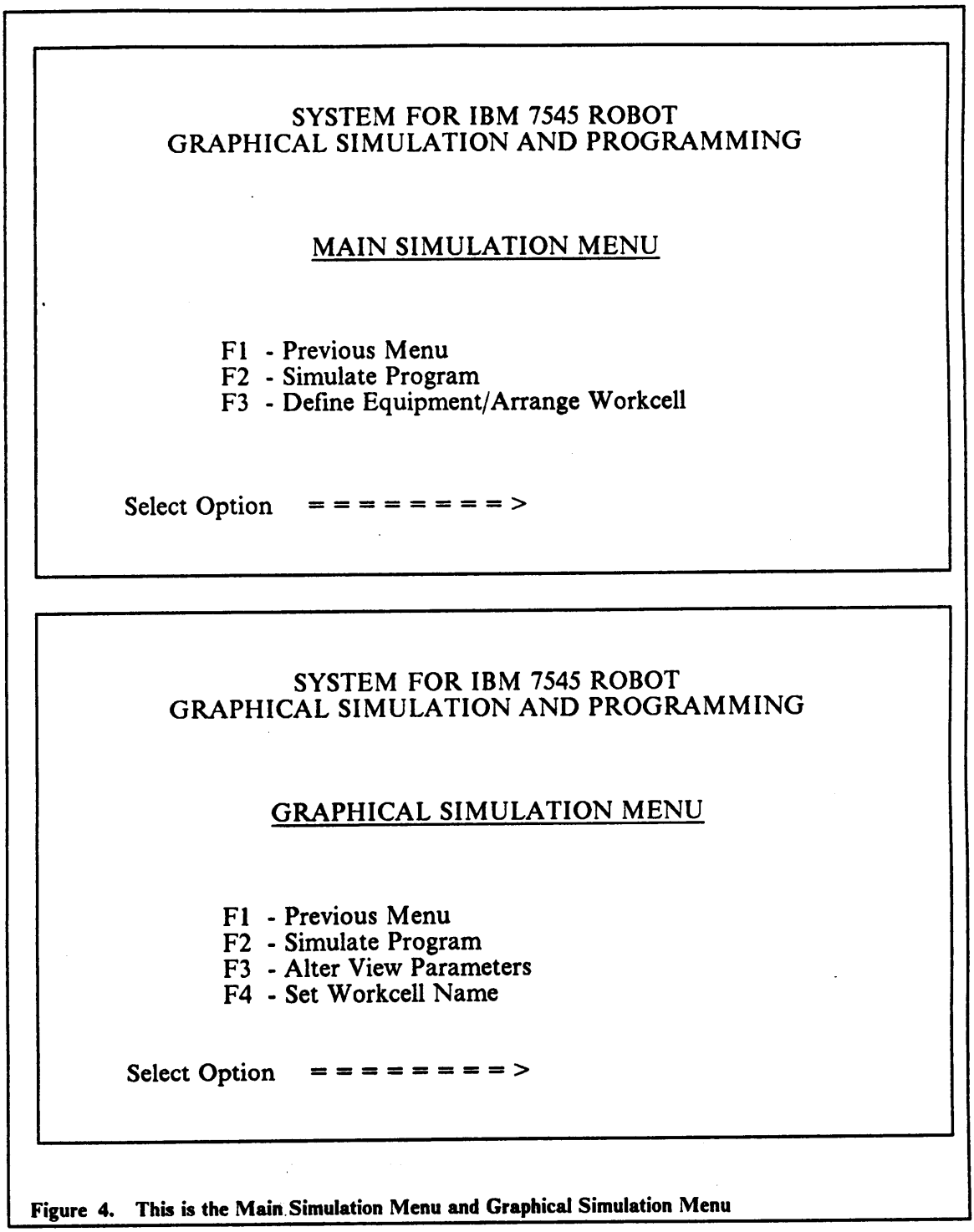

Figure 4. This is the Main Simulation Menu and Graphical Simulation Menu

SYSTEM FOR IBM 7545 ROBOT
GRAPHICAL SIMULATION AND PROGRAMMING

VIEWS MENU

F1 - Previous Menu
F2 - Zoom
F3 - Switch Plan/Side View
F4 - Standard View
F5 - Simulate Program

Select Option = = = = = = = = >

SYSTEM FOR IBM 7545 ROBOT
GRAPHICAL SIMULATION AND PROGRAMMING

WORKCELL LAYOUT MENU

F1 - Previous Menu
F2 - Define/Display Equipment
F3 - Arrange Workcell
F4 - Display Workcell Layout
F5 - Directory of Equipments/Workcells

Select Option = = = = = = = = >

Figure 5. This is the Views Menu and Workcell Layout Menu.

is displayed. Prompts appears in the text screen to guide the user through each of these options. The user is prompted to choose between the two views for a major view. The default view is the plan view. Zooming can be done on whichever view is chosen as the major view. The user is prompted to choose the lower left corner, and upper right corner of the desired area to be enlarged. Facilities to do subsequent enlargements are available, till the user decides to return to the views menu. There is also a provision to pan the image.

3.3.3 Define/Display Equipment and Workcell

These modules are invoked in the workcell layout menu shown in Figure 5. The user has the option to define/display a equipment, arrange or display a workcell or get a directory listing of available equipment and workcells. In the typical sequence the user has to first define an equipment. A prompt appears for the equipment name and the program opens a new file adding the extension .EQP to the name given by the user. The data can be entered using the editor provided. The name of the equipment is written to the DOS directory where all the equipment, workcells, and other files are listed. If the file name already exists a message will be displayed informing the user of the same. To display the equipment the user enters the name, and chooses the major view. The program opens the data file with the same name, reads in the data and displays it in both the plan and side view on the graphics screen. The data should be in the International Graphics Exchange Specification (IGES) Version 3.0 format. If the data file is imported form another graphics system in the IGES format, the user need not edit it. The file should have the extension .EQP to be displayed.

To arrange the workcell the user is led through a series of prompts to define the workcell name, number of equipment, and locations. The program reads in data from each of the equipment files and writes them to a new workcell file with the new X,Y,Z,R coordinates. This information is stored in new data files created as and when the user defines new workcells. The files takes the name of the workcell the user inputs and has the extension .CEL added to it. The name of the workcell is written to the DOS directory. The user can view this directory with the option of displaying only equipment files or workcell files.

3.3.4 Graphics Display

The graphics driver program is the modified system control program(20). It drives the graphics routines. This program reads the five files generated by the pseudo compiler interprets each line of the AML/E program and accordingly calls the appropriate subroutine in the graphics display program DEPUTY1, needed to display the graphical simulation of the robot.

The subroutine to compute the new intermediate locations to finally move the robot arm to a chosen X, Y, Z, R location in work space is MOVE. For all robot motions the arms of the robot goes through a series of rotational and translational motions around different axis. This program computes the angles of the robot arms. After the angles are obtained, it computes the location of the end of arms for graphical display purposes. This program also checks whether the point is within the workspace of the robot. It then displays the final position of the robot in three stages. The user has the option to redraw the entire screen or continue to simulate the next line of the AML/E program.

Prior to the actual simulation the first option that is available is to select between single step or a continuous run mode. In either case the user then chooses either complete program execution, partial program execution or subroutine execution. If partial execution is chosen prompts appear for the starting and ending line in the program. For subroutine execution the user is prompted for the subroutine name, and the values of any formal parameters. For both these options a menu to setup the initial conditions prior to simulation appears. It allows the user to set the pallet and counter conditions, the status of any output port, and the position of the robot just before simulation starts. The user can abort single step at any time and go into continuous mode, or abort the program at any time.

The program while executing interacts with the user only when statements involving input ports and breakpoint are encountered. The user has to decide whether the particular input port is ON/OFF as appropriate. The WAITI and TESTI statements will branch or stop execution depending on the status indicated by the user for that port. Many of the AML/E commands like the counter and most of the pallet commands do not require the graphics display program to be called. The DELAY statement causes a do loop to be executed in the graphics driver program itself. The WRITEO command turns the particular output port ON on the screen. The motion commands include PMOVE, DPMOVE, and ZMOVE. They cause the robot arm/end effector to be moved in both the plan and side view to the specified location. Any error in the robot program causes the program to abort.

Once the simulation is over the graphical simulation menu appears, and the user can repeat the simulation with another view or the standard view. If the simulation was done

in the standard view the control returns to the main simulation menu, or if the view was altered the control returns to the alter views menu.

4.0 Chapter 4. Development of the Graphic Simulator

4.1 Introduction

In this chapter each of the programs and subroutines developed for the Graphical Simulation and Programming Package (GSAPP) are discussed in detail. The programs associated with this system can be broadly grouped into three categories, menu programs, display programs, and definition programs. In the first group the major programs are the batch file GSAPP, and all the Menu display and selection programs,FIRST, BETWEEN, MENU2, and MENU3. In the display group the Graphics Driver Program MAIN, the Graphics Display Program DEPUTY1, and the alter views program ALTER are included. The workcell layout programs DEFEQP, DEFSETUP, DISPSETU, and DIRECTOR, can be merged into a definition programs group. These classifications are done for ease of discussion and identification of programs.

4.2 *Menu Programs*

Five main menus have been developed for this system. They are the Main Menu, Main Simulation Menu, Graphical Simulation Menu, Views Menu, and Workcell Layout Menu. The Main Menu is invoked through a batch file GSAPP.BAT by typing in GSAPP at the DOS prompt.

The program FIRST displays the main menu and provides the basis of an integrated system. It allows the user to carry out all the functions in one complete integrated software without returning to DOS level.

The user has the facility to edit a program using the AML/E editor, compile a program using the AML/E editor, and download it to the robot controller. The robot can be run either directly as provided by IBM or interactively as provided by R. Jayaraman(20). The user can generate cross reference listing or setup system configuration if desired. The option to interactively graphically simulate it is provided by this work and invoked from this menu. The user can set the name of the AML/E program along with the compiler options, in the MAIN menu and these are written to a file IRPS0, for subsequent use by other programs. For each of the above options a batch file is created and command passes to that batch file which invokes that particular program. At the end of execution the control returns to the main menu. The user can quit and return to the DOS prompt at the end of all the functions.

The program BETWEEN displays the main simulation menu. This program is called when the simulation option is chosen in the program FIRST by selecting F10 in the

main menu. It provides a selection between the graphical simulation menu and workcell layout menu.

The program MENU2 displays the graphical simulation menu. It provides the user the option of simulating the program with the standard view or the option of going to the Views Menu. The standard view option calls the subroutine STANDARD in the graphics display program program which causes the standard view with the plan as the major view and side view as the minor view with the original coordinates to be displayed on the graphics screen. This creates a flag SIM.FLG and control then passes to the batch file GSAPP which calls the pseudo compiler program COMPPC followed by the graphics driver program MAIN. The pseudo compiler generates five files which are the input to the graphics driver program. The user can display any workcell setup using the workcell option. The set cell name option causes a file WORK to be opened and the user input of the name of the workcell is written into it for subsequent use by the subroutine REDRAW. Both the side and plan view of the workcell will be displayed along with the robot and its workspace.

In the alter views option the program ALTER is called. This displays the views menu. The user has the option to Zoom/Plan which calls the subroutine ZOOM, or switch between the plan and side views as major views which is done by the subroutine SWITCH, or comeback to the standard view by selecting that option and invoking the subroutine STANDARD. All these subroutines are in the graphical display program DEPUTY1 which will be discussed in detail under the display programs section. The simulation option creates a flag SIM1.FLG and control passes to the batch file as in the previous case. Two different flags are used so that the control returns to the views menu in this case and to the graphical simulation menu in the other.

The program MENU3 displays the workcell layout menu and allows the user to select between the options of defining an equipment, displaying an equipment, defining a workcell setup and displaying it. The programs associated with these options are discussed under definition programs. The user has a directory option and can choose between the directory of equipment which are data files with the extension .EQP and workcells which are data files with the extension .CEL. Here control is passed to the batch file GSAPP and the DOS command DIR is used to display the directory listing along with file size and date. Control returns to the views menu when the user hits return.

4.3 Display Programs

Program COMPPC (20) is the pseudo compiler which reads the users AML/E program and generates five files namely a listing file (.OUT), a variables file (.VAR), a constants file (.CN1), an object file(.OBJ), and a symbol table file (.TAB). The object file and the symbol table are direct access files while the rest are sequential files. This program writes the name of the users AML/E program in a file IRPS00 for the program MAIN to read.

The program MAIN, the graphics driver program is the system control program which has been redesigned to interact with the graphics display program DEPUTY1 and the graphics monitor. This program reads one line of the AML/E program at a time and depending on the type of command the corresponding subroutine is called. The AML/E commands are classified into motion commands, sensor commands, flow of control

commands, counter and pallet commands, and subroutines. The subroutines in the graphics driver program corresponding to these AML/E groups are MOT, SENS, FLOW, COUN, PALL, SUB1 and SUB2. These are the subroutines in the system control program which have been modified to drive the graphics display program instead of the robot communications controller. This program allows the user to choose the mode of execution, which is either step through or continuous. In either case the Execution Options Menu that gives the user the option of complete program simulation, partial program simulation, or subroutine simulation, is invoked. In partial simulation the starting line and ending line have to be entered. In subroutine simulation the name of the subroutine along with the number of parameters and parameter values in either variable form or numeric form is entered. At all this stages the appropriate checks are made. For these two options the partial setup menu is displayed.

The partial setup menu gives the user the facility to setup the graphics screen in any desired way prior to partial program simulation. The options available include SETPART which allows the user to set the current value of the part for the chosen pallet name. If the pallet does not exist a message is displayed and the menu is displayed again. The GETPART option causes the robot arms to be moved on the graphics screen to the indicated value. PMOVE moves the robot to the choosen X,Y,Z, and R location. The option ZMOVE in the menu does essentially the same as PMOVE but only in the Z direction. The user can turn any output port on the graphics screen ON by using the WRITEO command. SETC is used to change the value of any counter in the AML/E program to the desired value. Checks are made to insure the counter has been defined in the program. The graphics display program DEPUTY1 is called to execute those commands that would represent a change in the status of the graphics display. The call to the graphics display program passes the values of the X, Y, Z, and R values. A code

is used to indicate which subroutine within the program DEPUTY1 is to be called to display the feature corresponding to the AML/E command. Figure 6. gives the codes used and what they signify. The value is passed using common blocks. All the AML/E commands this system will accept and the code corresponding to them is shown in Figure 7.

The program DEPUTY1 which is the graphics display program is the program that does the actual computations and display of the graphics. The subroutines within this program are MOVE, ZOOM, SWITCH, STANDARD, REDRAW, PLANSIDE, GRIPPER, PORTS, INPORT, OUTPORT, STROUT, and FRAMES.

The subroutine MOVE computes the location of the end of both the arms of the robot corresponding to value of X, and Y. It then computes the location of all the arm parameters in both the plan and side view. It checks for the validity of the location. This is repeated for each of the stages in every movement. The location of the gripper is indicated in the plan and side view. The orientation is displayed in the gripper window. At the end of this subroutine the final angles of the two arms THONI and THTWI and the Z and R positions are reinitialized for the next movement of the robot arm in the next pass. After every statement in the AML/E program that causes a change on the graphics screen has been simulated, the user has the option of continuing the simulation or redrawing the graphics display. The user can change the mode of execution from step by step to continuous or abort the execution if desired.

The subroutine ZOOM allows the user to pan and zoom in the major window irrespective of whether the plan view or side view is displayed there. The user is prompted to

CODE	FUNCTION
1	Calls the subroutine MOVE which computes the values of the intermediate and final coordinates. It then calls PLANSIDE to draw these positions.
2	Calls the subroutine INPORT.
3	Calls the subroutine OUTPORT. This also causes the gripper status to be displayed in the gripper window.
4	Calls the subroutine HOME and draws the robot and workcell in the home position in the view and coordinates selected by the user.
5	Calls the subroutine ZOOM, to pan and zoom in the major view which can be either the plan or side view. This calls the subroutine REDRAW to refresh the graphics screen.
6	Calls the subroutine SWITCH to switch between the plan and side view in the standard coordinates.
7	Calls the subroutine STANDARD to display the standard view which is the plan in the major view using the standard coordinates.
8	Calls the subroutine REDRAW which redraws the the entire screen in the current location of robot using the current coordinates and view.

Figure 6. Function of codes used in the Graphics Display Program

select the two corners of the area to be zoomed. The X and Y coordinates of each of these are read, the difference between the two is computed and the shorter side is taken as the side of the square to be enlarged. Using this new values of XMIN, XMAX, YMIN, and YMAX for the user coordinates if plan is the major view or SXMIN, SXMAX, SYMIN, SYMAX if side view is the major view, the subroutine REDRAW is invoked. The new values of the user coordinates in both the plan and side view are written to a file INFO for subsequent use by the subroutine REDRAW. In the FORGRF routines used here, the user coordinate system for the current screen window is defined by DWINDO. Prior to defining the user coordinate system the screen window must be invoked by the command TWINDO. After zooming the pan option is displayed on the screen. The user chooses the location of the new center and this value is used to compute the new user coordinates for redraw. As in the case of zoom the coordinates will depend on whether it is the plan or side view.

The subroutine SWITCH prompts the user to choose the plan or side view as the major view. The screen coordinates used for the major view is 0 to 750 for the X axis and 0 to 750 for the Y axis. For the minor view it is 751 to 1023 in the X direction and 400 to 672 in the Y direction. The variable BASC takes the value 80 or 83 depending on the whether the plan or side view is selected and this is written to the file INFO for subsequent use by REDRAW.

The subroutine STANDARD will draw the plan and side view in their original position and without pan or zoom. The file WORK is read to see if there is any workcell to be displayed and if any exists it will be displayed. The program writes the current value of BASC and the user coordinates to the file INFO for subsequent redraws.

AML/E Command	Code
DELAY	1
DPMOVE	1
GRASP	3
PMOVE	1
RELEASE	3
ZMOVE	1
LINEAR	-
PAYLOAD	-
ZONE	-
WAITI	2
WRITEO	2
BRANCH	-
TESTC	-
TESTI	2
TESTP	-
BREAKPOINT	2
COMPC	-
DECR	-
INCR	-
SETC	-
GETPART	1
NEXTPART	-
PREVPART	-
SETPART	-

The dash indicates that for these commands the graphics display program DEPUTY1 is not called. The graphics driver program MAIN does the setting and alterations within the program itself.

Figure 7. Codes corresponding to the AML/E commands.

The subroutine REDRAW first calls the subroutines PORTS and GRIPPER. The file WORK is opened and if any workcell name is to be displayed, it displays it in the view and coordinates the user has selected. The workcell is drawn in both the plan and the side view. The program reads in one line at a time of the .CEL data file and draws a point or line corresponding to the entity type number. It then calls subroutines PLANSIDE which draws the robot and workspace in plan and side view, and FRAMES which draws the frames. All these subroutines are discussed in detail below.

The subroutine PLANSIDE does the actual drawing of the plan and side views using the FORGRF motion commands MOVEA and DRAWA and the values computed from the subroutine MOVE. MOVEA moves the cursor to the user specified coordinates and is an absolute move statement. DRAWA draws a line form the current position of the cursor to the point specified by the user. It is also an absolute move statement. Relative moves in the user coordinates are got by MOVER and DRAWR. The subroutine WKSPACE which displays the workspace in plan view is called first. The display of the base, middle, and end joints of the robot, the gripper head and gripper, and robot stand, in plan view. and side view is done in this subroutine.

The subroutine GRIPPER displays the orientation of the gripper.The screen coordinates are 751 to 859, and 672 to 780, for X and Y coordinates in that order. The user coordinates are -400.0 to 400.0 and -400.0 to 400.0 in the X and Y direction. As the gripper is rotated the radius of the circle displayed will be rotated to indicate the current position of the gripper. The status of the gripper is indicated by OPEN or CLOSE in the gripper window. This display is invoked from the subroutine OUTPORT to be discussed below.

Subroutine PORTS displays the numbers 1 to 16 and the characters PORTS within the screen coordinates 751 to 883 and 0 to 400 in the X and Y direction respectively. The

user coordinates here are 0.0 to 10.0 and 0.0 to 10.0 in the X and Y direction. It calls the subroutine INPORT and OUTPORT. Subroutine INPORT displays the character DI and depending on the value of the variable VAL2 marks 'X' or ' ' to indicate the inport is on or off for each of the sixteen input ports. The screen coordinates are 883 to 958 and 0 to 400 in the X and Y direction respectively. These display are done using the screen coordinates itself, hence the ability to give different colors. The graphics routines does not display in color for the user coordinate locations. Subroutine OUTPORT displays the character DO. It is identical to the inport subroutine except for port two, which corresponds to the gripper status. and causes the display of the characters OPEN and CLOSE in the gripper window. The screen coordinates for OUTPORT are 958 to 1023 and 0 to 400 in the X and Y direction respectively.

The subroutine FRAMES draws the frame around the views using the screen coordinates and absolute move statements. The FORGRF commands in screen coordinates are MOVEABS and DRAWABS for absolute move and draw. It is MOVEREL and DRWREL for relative move and draw. Subroutine WKSPACE computes the point corresponding to the four arcs that make up the plan view of the workspace, and joins them together in the current user coordinates. The workspace for the side view is a rectangle and is completed in the subroutine PLANSIDE itself.

The subroutine STROUT is used to locate a string of characters at any location of the graphics screen, within the screen coordinates 0 to 1023 and 0 to 780 in the X direction and Y direction. It is written in assembly language and can be called in MS Fortran.

4.4 *Definition Programs*

Prior to the actual simulation the user has the option to define equipment and workcell layouts. The Workcell Layout Menu displayed in program MENU3 provides the facility to define and display equipment and workcells, view a directory listing of all the equipment and workcells, or return to the previous menu.

The program DISPEQP allows the user to define an equipment, display it on the graphics screen or return to the workcell layout menu. To define an equipment for the first time the user needs to enter the data for the equipment in the International Graphics Exchange Specification (IGES) version 3.0 format using the editor provided. The data must be entered within the limitation that only points and lines can be drawn. To enter a point data the entity number 116 must be entered in the first 3 columns, followed by the X, Y, Z, coordinates of the point as real numbers separated by a comma in between them and ending with a semi colon. To enter a line the entity number 110 in the first three columns must be followed by the X, Y, Z coordinates of the starting point, and then the X, Y, Z, coordinates of the ending point of the line. Here also commas must separate the numbers and a semi colon must follow the last number. The 73rd column must have a P to indicate it is a parameter. The user can have comments by entering S on the 73rd column. The other options provided by IGES are global and directory entries. These are not considered by these program. If the equipment name already exists the user is warned. All equipment files have extension .EQP for identification purposes and only equipment with this extension can be displayed. The program reads the 73rd column of the IGES format data file for the character P to indicate it is a parameter. It then reads the first 3 columns to see if it is a 110 or 116 to

indicate a line and point respectively. The program draws the equipment in both the plan and side view in the standard coordinates. The user has the option to pan and zoom the equipment. The subroutine FRAME is called to display the frames.

The program DEFSETUP defines the workcell setup menu and gives the user option of arranging a workcell or returning to the previous menu. A workcell can be defined to suit any particular production or experimental setup, and store it for subsequent use. The user is prompted for the name of the setup, number of equipment, and X, Y, Z, R location corresponding to each of the equipment. The user enters the coordinates in free format. The program checks to see if the workcell file name has been duplicated. It reads the equipment data files and writes them to to a new setup data file with the extension .CEL. The new data file omits the character in the 73rd column since all the records are data records only and there is no need to distinguish between them. All new files created will show up in a directory search.

The program DISPSETU is the program that displays the workcell layout menu. The user can either enter the name of the workcell or return to the previous menu. The setup is displayed in both the plan and side view along with the robot and workspace. The workspace is displayed by the subroutine WKSPACE and robot by the subroutine PLANSIDE. The user has the option to zoom/pan the display or switch views if desired. The routines ZOOM and SWITCH discussed earlier are called here.

4.5 Assembly Routines

All the graphics displays are done by the FORGRF routines that are written in assembly language and called by FORTRAN statements. It allows the user to MOVE and DRAW in either screen coordinates or user coordinates. Subroutine DSWAP used to switch between the text monitor and the enhanced color graphics monitor is also written in assembly language. The contents of the graphics screen is retained when the active screen switches from text to graphics, and this helps save the time needed to redraw the screen.

5.0 Chapter 5. Analysis of the Simulator

5.1 Introduction

In this chapter the assumptions made in the development of the interactive graphical simulator and the limitations of the system with respect to the pseudo compiler, the graphics driver and the data files are discussed. The features of the test programs developed to test the GSAPP system are also discussed.

5.2 Assumptions and Limitations

5.2.1 Pseudo Compiler

The input to this work is the five output files created by the program COMPPC (20). The limitations of that program will automatically be the limitations of this system also. Hence some of the major limitations stated for that program is repeated here. The users

AML/E program must be compiled using the AML/E compiler, to overcome the limited error detecting facilities of the pseudo compiler. The pseudo compiler does not handle all the AML/E commands. The commands that are not handled include : CSTATUS, GET, GROUP, GUARDI, ITERATE, MSTATUS, NOGUARD, PUT, REGION, and XMOVE There are two restrictions on formal parameters in a subroutine. Pallet names cannot be passed to a formal parameter and the maximum number of formal parameters for a subroutine is five.

5.2.2 Graphics Driver

The graphics driver program MAIN calls the graphics display program DEPUTY1 to display the arm movements and port status on the screen. However it is not required to call the the graphics display program for many of the AML/E commands. These include BRANCH, TESTC, TESTP, COMPC, DECR, INCR, SETC, NEXTPART, PREVPART, SETPART, PAYLOAD, ZONE, and LINEAR. The program MAIN changes the value of the appropriate counters and variables within itself and proceeds to the next line in the AML/E program. The simulation is not effected by the values of PAYLOAD, ZONE or LINEAR. This is because for the simulator the speed of the robot arm will not effect how the robot is graphically represented and moved on the screen. This simulator is used primarily for debugging purposes, as a visual aid to help avoid collisions, and for instructional purposes. It does not compute actual cycle times depending on the payload, zone and linear commands.

5.2.3 Data Files

The data used to define the equipment can only be point and line data. This is because the FORGRF routines used to display the equipment has only MOVE and DRAW commands. The data must be entered to represent drawing the figure in one stroke from start to end.

The other assumption are that the data file must be in ASCII format as against a binary format and conform to the IGES version 3.0. No text or character can be introduced in the drawing. All dimensions are in millimeter and the cartesian coordinate system is used. The equipment must not be rotated translated or scaled when it is defined or imported into the system, that is the transformation matrix must be unity. Once the data file has been read by this program the equipment can be rotated, translated or scaled. No multiple levels of data with back pointers, group, or associativity are considered.

The equipment defined in this way are placed in the workcells using the workcell setup menu provided. These equipment are defined with the bottom left hand corner as the origin, but most locations required for robot programming are specified in terms of pickup or drop points. Hence the user has to take care to add or subtract these dimensions when specifying the location of the different equipment. Another recommended procedure in the case of equipment that appear too small on the screen, is to double the dimensions provided they do not mislead the users concept of the setup or effect the pickup/drop points used in the AML/E program. It is recommended that unnecessary details like structural details be omitted to save the time required to display them and in subsequent redrawing.

5.3 *Test Programs*

Three test programs were used to test the simulator. Two of the programs, Test Program I and II were the ones used to test the system control program developed for interactive programming. Since they test all the possible AML/E commands, and pass both constants and variables to subroutines there was no need to rewrite them. All commands except pallet commands were used in subroutines and up to five formal parameters were passed into the subroutine. The third program Test Program III is more of a demonstration program. It was written to highlight the effectiveness of the simulator in helping to visualize how a program performs. Two equipment are used in this program, a conveyor, and a pallet. The program tests to see if there is a block on the conveyor and then picks the block from the conveyor and arranges them on a pallet. After six blocks have been placed on the 3 x 2 pallet the program aborts. The user is able to see exactly how the program performs and be sure of avoiding any possible damage to equipment or injury to personnel It also helped in debugging the program due to the interactive feature.

Between the three test programs the GSAPP system was tested to see the performance under all of the following conditions:

I. Continuous Execution

A. Complete Program

B. Partial Program

1. SETPART

2. GETPART

3. PMOVE

4. ZMOVE

5 WRITEO

6 SETC

C. Subroutine

Setup as in partial program

D. Abort

II. Stepthrough

All the cases discussed above.

It was seen the graphics simulator performed as per the system design under all these conditions for all the AML/E commands and under all the graphic views possible.

6.0 Chapter 6. Conclusions and Recommendations

This system was developed specifically for the IBM 7545 robot and the AML/E language. The facility to switch views, zoom into each view, and to define equipment and workcells, helped in visualizing the setup, in the debugging process, and avoiding damage to equipment and injury to personnel. It also serves instructional purposes. The facility to do interactive simulation, stepping through the program one line at a time helped to get a better understanding of the program. The ability to do partial setup of the graphics display gave the user the convenience of starting and stopping the simulation at any chosen line of the AML/E program.

The system uses two monitors to gain more display area. This causes the program to switch between the two monitors. If this can be avoided in the future by having a bigger display like a 16 inch monitor and having the menu on one side, and the display on the other, this will avoid the time taken and the inconvenience of switching monitors. The speed at which the system executes, and the facilities for refresh can be improved by increasing the capabilities of the personal computer through additional hardware and

software. It is desirable to have the entire display in color by using other graphics routine.

The system has the limitations of the pseudo compiler program COMPPC and hence can handle only a subset of the AML/E commands. Future work can include this commands.

Future work could use three dimensional graphics routines as opposed to the current two dimensional routines. This will give a more real life like appearance and will also allow the full scope of the IGES version 3.0 standard to be utilized. Currently the equipment to be used in any workcell must be placed in the desired location by entering the coordinates in the define workcell program and later viewing it in the display setup program. This process can be modified by having facilities to pick, place and rotate equipment in the define setup program itself and then getting the desired coordinates which then can be used to define the setup.

The same approach can be used for other robots running on the AML/E language. In the case of robots running on other languages it would be required to develop a pseudo compiler to provide the facility for interactive simulation if such a facility does not already exist. The graphics display would be identical to the present system.

7.0 Bibliography

1. AML/E Version 4.0/4.1 User's Guide, IBM Manual 8577150, August 1985.

2. Azzam, S. J., and Unuvar, M. U., 'Offline Robot Programming with GRIPPS,' Proceedings 13th International Conference on Industrial Robots & Robots 7 , April 1983, pp. 18-23 - 18-33.

3. Barel, M., Loten, A., and Jrnon, D., 'TDL: A Simulation Language For Automated Workcell,' Proceedings - AUTOFACT'85, June 1985, pp. 12-1 -12-12.

4. Best, M., 'Integrated Cell-Factory Simulation,' Paper presented at SME Simulation and Computer Graphics For Robotic Workshop, Oct. 1986.

5. CATIA Robotics Users Manual, IBM Program No: 5668-836, 2nd Edition July 1986.

6. Charny, J.,'Off-Line Robot Programming,' Proceedings Computers in Engineering Vol. 2, 1983, pp. 51-55.

7. Chawla, S.D., & Gruver, W.A., ' Off-line Robot Programming with an Integrated Graphics Subsystem,' Proceedings Computers in Engineering, 1984, Vol. 1, August 1984, pp. 111-114.

8. Craig J. J., 'Anatomy of an Off-Line Programming System,' Robotics Today, Vol. 7, No. 1, Feb. 1985, pp. 45-47.

9. Derby, S.J., 'GRASP From Computer Aided Robot Design to Off-line Programming,' Robotics Age, Vol. 6, No.2, 1984, pp. 11-13.

10. Derby, S.J.,'General Robot Arm Simulation Program (GRASP) Part 1 and Part 2,' Proceedings Computers in Engineering ,Vol. 2, August 1982, pp. 139-145 and 147-154.

11. Derby, S.J., 'Simulating Motion Elements Of General Purpose Robot Arms,' International Journal of Robotics Research' Vol. 2, No 1, Spring 1983.

12. Durrant-Whyte, H. F., ' ROBCON : An Interactive Robot Dynamics and Control Simulator,' Proceedings Computers in Engineering August, 1985, Vol. 1, pp. 73-77.

13. Hammond, R., Hanse, J., and Hansen, L. 'Elimination of Application Risk Utilizing McAuto PLACE Software And Corporate Robot Facilities,' Proceedings - Robot 9 June 1985, pp. 17-58 -17-66.

14. Heigenbotham, W. B., Dooner, M., and Case, K., 'Assesing Robot Performance With Interactive Computer Graphics,' Robotics Today, Vol. 1, No. 2, pp. 33-35.

15. Howie, P., 'Graphic Simulation For Off-Line Robot Programming,' Robotics Today, Vol. 6, No. 1, 1984, pp. 63-66.

16. Katajamaki, M., and Kinerva, J, 'CAD/CAM - Revolutionizing Robot Applications Design,' Proceedings - 14th International Symposium on Industrial Robots, 1984, pp. 689-700.

17. Katajamaki, M., 'CAD/CAM in Robotics Applications Design and Simulation,' Proceedings - AUTOFACT 6, October 1984.

18. Kirschbrown, R. H. and Dorf, R. C., 'KARMA- A knowledge Based Robot Manipulation System: Determining Problem Characteristics,' Proceedings - Robots 8 June 1984.

19. Kirschbrown, R. H., Dorf, R. C., ' KARMA- A Knowledge Based Robot Manipulation System,' Robotics Vol. 1, No. 1, May 1985, pp. 3-12.

20. Jayaraman, R.,'Development of An Interactive Programming System For The IBM 7545 Robot,' M.S. Thesis Virginia Polytechnic Institute and State University,1986.

21. Kretch, S. J., 'Robotic Animation,' Proceedings - Annual Industrial Engineering Conference 1983, pp. 102-105.

22. Larson, G., and Donath, M., ' Animated Simulation of Intellegent Robot Workcells,' Proceedings - Robots 9, June 1985, pp. 19-54-19-69.

23. Leu M. C., 'Elements of Computer Graphic Robot Simulation,' Proceedings - Computers in Engineering, Vol. 1, August 1985, pp. 65-72.

24. Mahajan R., and Walter S. E., 'Computer Aided Automation Planning: Workcell Design and Simulation,' Robotics Engineering, Vol. 8, No. 8, August 1986, pp. 12-15.

25. Mahajan, R., & Mogal, J. S., ' An Interactive Graphic Robotics Instructional Program - I GRIP , A Study of Robot Motion and Workspace Constraints,' Proceedings - Robots 8, June 1984, pp. 16-41 - 16-56.

26. Mogal, J. S., 'IGRIP - A Graphics Simulation Program For Workcell Cell Layout And Offline Programming,' AUTOFACT V, 1983, pp. 10-65 - 10-77.

27. Myers, J. K., 'A Robotic Simulator With Collision Detection: RCODE,' Robotics Laboratory Technical Note - SRI International, California, 1985.

28. Newcomer, K. L., and Discopink, C. A., 'Computer-Aided Analysis Of A Prab FC Industrial Robot,' Proceedings-Robots 9, June 1985, pp. 21-77 - 21-102.

29. Norton, R. L., 'Graphic Simulation Of Puma Robot Motions On The Apple Computer,' Proceedings - Computers In Engineering August 1983, Vol. 2, pp. 125-130.

30. Price,R. B., 'Off-Line Programming with a Microcomputer,' Proceedings - Robots 8, June 1984, pp. 20-96 - 20-102.

31. Stauffer, R. N.,'Robot System Simulation,' Robotics Today Vol. 6, No. 3, 1984, pp. 81-90.

32. Soroka, B. I., ' A Program for Computer-Aided Robot Design,' Proceedings - Computers in Engineering Vol. 2, August 1982, pp. 93-99.

33. Steiner, M. W.,'Robots: Practical Application and Simulation,' Proceedings Computers in Engineering Vol. 1, August 1982, pp. 101-105.

34.Tilove, R. B., Shapiro, V., and Pickett, M. S., 'Modelling And Anlalysis of Robot Workcells in RoboTeach,' Computer Integrated Manufacturing And Robotics PEF-Vol. 113, 1984, pp. 33-52.

35.Yong, Y. F., Gleave, J. A., Green, J. L., and Bonney, M. C., 'Offline Programming of Robots,' Handbook Of Industrial Robots, Edited by S. Y.Nof, John Wiley & Sons, 1985.

Appendix A.

This section contains the following programs.

(1) GSAPP.BAT

(2) DSWAP.ASM

****

(1) GSAPP.BAT - The batch file that initializes the system and controls the different programs.

```
ECHO OFF
REM : DELETE ALL FLAGS ETC IN THE BEGINING ALSO TO TAKE CARE OF
REM : SITUATIONS WHERE THE USER USES CTRL-BRK AND EXITS PROGRAM.
IF EXIST *.FLG DEL *.FLG
IF EXIST CELNAM DEL CELNAM
IF EXIST WORK DEL WORK
IF EXIST INFO DEL INFO
IF EXIST EQPT.BAT DEL EQPT.BAT
IF EXIST EDTR.BAT DEL EDTR.BAT
IF EXIST COMPILE.BAT DEL COMPILE.BAT
IF EXIST INTER.BAT DEL INTER.BAT
IF EXIST LOAD.BAT DEL LOAD.BAT
IF EXIST CONFIG.BAT DEL CONFIG.BAT
IF EXIST REF.BAT DEL REF.BAT
IF EXIST COMA.BAT DEL COMA.BAT
IF EXIST IRPS0 DEL IRPS0
IF EXIST IRPS00 DEL IRPS00
IF EXIST IRPS01 DEL IRPS01
REM; DISPLAY THE MAIN MENU
:FIRST
FIRST
IF EXIST BYE.FLG GOTO END
IF EXIST BETWEEN.FLG GOTO BETWEEN
IF EXIST EDIT.BAT GOTO EDIT
IF EXIST COMPILE.BAT GOTO COMPILE
IF EXIST INTER.BAT GOTO INTER
IF EXIST LOD.BAT GOTO LOD
IF EXIST CONFIG.BAT GOTO CONFIG
IF EXIST COMA.BAT GOTO COMA
IF EXIST REF.BAT GOTO REF
REM: TO PASS CONTROL TO THE BATCH FILE FOR THE AML/E EDITOR
:EDIT
COMMAND /C EDTR.BAT
DEL EDTR.BAT
GOTO FIRST
REM: TO PASS CONTROL TO THE BATCH FILE FOR THE AML/E COMPILER
:COMPILE
```

```
COMMAND /C COMPILE.BAT
DEL COMPILE.BAT
GOTO FIRST
REM: TO PASS CONTROL TO THE BATCH FILE FOR THE INTERACTIVE PROGRAMMING
:INTER
COMMAND /C INTER.BAT
DEL INTER.BAT
GOTO FIRST
REM: TO PASS CONTROL TO THE BATCH FILE FOR LOADING AML/E PROGRAM
:LOD
COMMAND /C LOD.BAT
DEL LOD.BAT
GOTO FIRST
REM: TO PASS CONTROL TO THE BATCH FILE FOR SETTING CONFIGURATIONS
:CONFIG
COMMAND /C CONFIG.BAT
DEL CONFIG.BAT
GOTO FIRST
REM: TO PASS CONTROL TO THE BATCH FILE FOR COMMUNICATION W/CONTROLLER
:COMA
COMMAND /C COMA.BAT
DEL COMA.BAT
GOTO FIRST
REM: TO PASS CONTROL TO THE BATCH FILE TO GENERATE CROSS LISTING
:REF
COMMAND /C REF.BAT
DEL REF.BAT
GOTO FIRST
REM: TO DISPLAY MAIN SIMULATIOM MENU
:BETWEEN
IF EXIST WORK DEL WORK
IF EXIST BETWEEN.FLG DEL BETWEEN.FLG
BETWEEN
REM : THE DELETE CELNAM IS TO TAKE CARE OF SIMULATION WITHOUT ANY
REM : WORKCELL. ONCE THE USER COMES BACK THIS MAIN SIMULATION MENU
REM : THE CELNAM IS DELETED
IF EXIST CELNAM DEL CELNAM
IF EXIST FIRST.FLG GOTO FIRST
IF EXIST MENUTW.FLG GOTO MENUTW
IF EXIST MENUTH.FLG GOTO MENUTH
REM: TO DISPLAY GRAPHICAL SIMULATION MENU
:MENUTW
IF EXIST MENUTW.FLG DEL MENUTW.FLG
IF EXIST SIM.FLG DEL SIM.FLG
IF EXIST SIM1.FLG DEL SIM1.FLG
MENU2
IF EXIST BETWEEN.FLG GOTO BETWEEN
IF EXIST SIM.FLG GOTO SIM
IF EXIST ALT.FLG GOTO ALT
REM; TO DO INTERACTIVE SIMULATION
:SIM
IF EXIST SIM.FLG DEL INFO
COMPPC
BOSS1
REM :AFTER SIMULATING WITH NORMAL VIEW GO BACK TO MENU 2
IF EXIST SIM.FLG GOTO MENUTW
IF EXIST SIM1.FLG GOTO ALT
REM: TO DISPLAY VIEWS MENU
:ALT
IF EXIST INFO DEL INFO
DEL ALT.FLG
IF EXIST SIM1.FLG DEL SIM1.FLG
ALTER
IF EXIST MENUTW.FLG GOTO MENUTW
IF EXIST SIM1.FLG GOTO SIM
REM: TO DISPLAY WORKCELL LAYOUT MENU
:MENUTH
```

```
IF EXIST MENUTH.FLG DEL MENUTH.FLG
MENU3
IF EXIST BETWEEN.FLG GOTO BETWEEN
IF EXIST DEFSETUP.FLG GOTO DEFSETUP
IF EXIST DISPSETU.FLG GOTO DISPSETU
IF EXIST DISPEQP.FLG GOTO DISPEQP
IF EXIST DIR.FLG GOTO DIRECTOR
REM: TO DISPLAY DEFINE WORKCELL MENU
:DEFSETUP
IF EXIST DEFSETUP.FLG DEL DEFSETUP.FLG
DEFSETUP
GOTO MENUTH
REM: TO DISPLAY DISPLAY WORKCELL MENU
:DISPSETU
CLS
IF EXIST DISPSETU.FLG DEL DISPSETU.FLG
DISPSETU
GOTO MENUTH
REM: TO DISPLAY DEFINE/DISPLAY EQUIPMENT MENU.
:DISPEQP
CLS
IF EXIST DISPEQP.FLG DEL DISPEQP.FLG
DISPEQP
IF EXIST EQPT.BAT GOTO EQPT
GOTO MENUTH
REM: TO PASS CONTROL TO THE BATCH FILE TO INVOKE TEXT EDITOR
:EQPT
COMMAND /C EQPT.BAT
IF EXIST EQPT.BAT DEL EQPT.BAT
GOTO DISPEQP
REM: TO DISPLAY DIRECTORY MENU
:DIRECTOR
IF EXIST DIR.FLG DEL DIR.FLG
DIRECTOR
IF EXIST DIRCT1.FLG GOTO DIRCT1
IF EXIST DIRCT2.FLG GOTO DIRCT2
GOTO MENUTH
REM: TO PASS CONTROL TO THE BATCH FILE TO DISPLAY .EQP DIRECTORY
:DIRCT1
IF EXIST DIRCT1.FLG DEL DIRCT1.FLG
DIR *.EQP/P
PAUSE
GOTO MENUTH
REM: TO PASS CONTROL TO THE BATCH FILE TO DISPLAY .CEL DIRECTORY
:DIRCT2
IF EXIST DIRCT2.FLG DEL DIRCT2.FLG
DIR *.CEL/P
PAUSE
GOTO MENUTH
REM: DELETE ALL FLAGS AND TEMPORARY FILES
:END
IF EXIST *.FLG DEL *.FLG
IF EXIST CELNAM DEL CELNAM
IF EXIST WORK DEL WORK
IF EXIST INFO DEL INFO
IF EXIST EQPT.BAT DEL EQPT.BAT
IF EXIST EDTR.BAT DEL EDTR.BAT
IF EXIST COMPILE.BAT DEL COMPILE.BAT
IF EXIST INTER.BAT DEL INTER.BAT
IF EXIST IRPS0 DEL IRPS0
IF EXIST IRPS00 DEL IRPS00
IF EXIST IRPS01 DEL IRPS01
```

****

```
C     (2) DSWAP.ASM - THE ASSEMBLY LANGUAGE PROGRAM TO SWITCH MONITORS.

C      THE ACTIVE TERMINAL IS SWITCHED FROM TEXT TO COLOR GRAPHICS
C      OR ENHANCED GRAPHICS DEPENDING ON THE CODE 1 OR 2.

        title   dswap.asm
        page    ,132

        comment     {

     usage          call dswap (idsply)

                    idsply      display adapter to address
                                0 - monochrome adapter - text mode
                                1 - CGA adapter - 640X200 2 color mode
                                2 - EGA adapter - 640X350 16 color mode

     language       Microsoft FORTRAN77

     notes          1. The address of the BIOS equipment flag (EQUIP_FLAG)
                       is coded into this subroutine.
                    2. The high bit of the mode value is set high during
                       the BIOS call to prevent buffer clear.  This
                       dosen't seem to work for the MONO adapter.

                    }

VIDEO_IO        equ     10H

DMASK           equ     11001111B
DMONO           equ     00110000B
DEGA            equ     00100000B
DCGA            equ     00100000B

MODE_TEXT       equ     10000010B
MODE_GRPHX      equ     10000110B

EQUIP_FLAG_OFF  equ     0410H

        public  dswap

code    segment para 'CODE'
        assume  cs:code,ds:code,es:nothing

dswap           proc    far

        push    bp
        mov     bp,sp

        push    ds

        les     bx,[bp+06]
        mov     ax,es:[bx]

        cmp     al,01
        jz      ds200
        cmp     al,02
        jz      ds300

;    handle swap to MONO adapter

ds100:
        sub     ax,ax
        mov     es,ax
        mov     dl,es:[EQUIP_FLAG_OFF]
        or      dl,00110000B
```

```
        mov     es:[EQUIP_FLAG_OFF],dl
        mov     ah,0
        mov     al,00H
        int     VIDEO_IO
        jmp     ds900

;   handle swap to CGA adapter

ds200:
        sub     ax,ax
        mov     es,ax
        mov     dl,es:[EQUIP_FLAG_OFF]
        and     dl,11001111B
        or      dl,00010000B
        mov     es:[EQUIP_FLAG_OFF],dl
        mov     ah,0
        mov     al,86H
        int     VIDEO_IO
        jmp     ds900

;   handle swap to EGA adapter

ds300:
        sub     ax,ax
        mov     es,ax
        mov     dl,es:[EQUIP_FLAG_OFF]
        and     dl,11001111B
        or      dl,00100000B
        mov     es:[EQUIP_FLAG_OFF],dl
        mov     al,90H
        mov     ah,0
        int     VIDEO_IO
        jmp     ds900

ds900:
        pop     ds

        pop     bp

        ret     4

dswap           endp

code    ends

        end
```

Appendix B.

```
     THIS SECTION CONTAINS THE FOLLOWING PROGRAMS WHICH ARE
     CATEGORIZED AS MENU PROGRAMS.

     (1) FIRST

     (2) BETWEEN

     (3) MENU2

     (4) MENU3

     (5) ALTER

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

C    (1) FIRST - PROGRAM TO DISPLAY THE MAIN MENU

C    THIS PROGRAM IS CALLED BY THE BATCH FILE GASP.BAT
C    AND DISPLAYS THE MAIN MENU FOR THE SYSTEM.

C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C       AMLED, AMLCOM,AMLLO, AMLCONF, AMLCOM,
C       AMLREF,CLRBUF,CLS,CSROFF,CSRON,INKEY,
C       INTER,MENU,TTOUT,DSWAP.

      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME
      INTEGER * 2 J,L,KYCOD1,KYCOD2
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
      COMMON /B/ L

C     VARIABLES C1,C2,C3,C4 REPRESENT THE COMPILER OPTIONS.
C     C1 AND C3 MAY BE '/' OR A BLANK CHARACTER.
C     C2 AND C4 MAY BE 'L','S' OR A BLANK CHARACTER.

      L = 0
      C1 = ' '
      C2 = ' '
      C3 = ' '
      C4 = ' '

C     READ THE FILE IRPSO FOR PROGRAM OPTIONS IF ANY, ELSE CARRY ON.
      OPEN (1,FILE='IRPSO',STATUS='OLD',ERR=199)
      READ (1,110,END=10) L,NAME,C1,C2,C3,C4
110   FORMAT(I2,1X,A20,4A1)
10    REWIND (1)
      CLOSE(1)
```

```
199   CALL DSWAP(0)
      CALL CSRON
      CALL CLRBUF

C     DISPLAY MAIN MENU IF NO FILENAME EXISTS IN THE FILE
C     "IRPSO",OTHERWISE DISPLAY THE PROGRAM OPTIONS MENU.

      CALL TTOUT(0,1,23,'SYSTEM FOR IBM 7545 ROBOT',25,25,
     *31)

      CALL TTOUT(0,3,12,
     *'GRAPHICAL SIMULATION AND INTERACTIVE PROGRAMMING',
     *48,48,31)

      IF (L .LE. 1) THEN
         CALL TTOUT(0,5,30,'MAIN MENU',9,9,31)
         CALL TTOUT(0,6,30,'---------',9,9,31)
      ELSE
         CALL TTOUT(0,6,14,'Filename (.AML assumed):',24,24,26)
         CALL TTOUT(0,7,21,'Compiler options:',17,17,26)
      J = 0

      DO 1 I = 22-L,20
      J = J + 1
      CALL TTOUT(0,6,38+J,NAME(I:I),L-1,L-1,26)
1     CONTINUE

         IF (C1 .NE. ' ' .AND. C2 .NE. ' ') THEN
         CALL TTOUT(0,7,39,C1,1,1,26)
         CALL TTOUT(0,7,40,C2,1,1,26)
         ENDIF

         IF (C3 .NE. ' ' .AND. C4 .NE. ' ') THEN
         CALL TTOUT(0,7,41,C3,1,1,26)
         CALL TTOUT(0,7,42,C4,1,1,26)
         ENDIF



      ENDIF

      CALL TTOUT(0,9,24,'Select a function:',18,18,27)
      CALL TTOUT(0,12,21,'FN1  - Return to DOS',20,20,31)
      CALL TTOUT(0,13,21,'FN2  - Edit/Teach a Proram,27,27,31)
      CALL TTOUT(0,14,21,'FN3  - Compile a Program',24,24,31)
      CALL TTOUT(0,15,21,'FN4  - Load a Program to Controller',35,35,31)
      CALL TTOUT(0,16,21,'FN5  - Set System Configuration',31,31,31)
      CALL TTOUT(0,17,21,'FN6  - Communicate with Controller',34,34,31)
      CALL TTOUT(0,18,21,'FN7  - Generate Cross Reference Listing',39,
     *39,31)
      CALL TTOUT(0,19,21,'FN8  - Set Program Name and Options',34,34,31)
      CALL TTOUT(0,20,21,'FN9  - Interactively Run Robot',30,30,31)
      CALL TTOUT(0,21,21,'FN10 - Simulate Program Graphically',35,35,31)
      CALL TTOUT(0,24,24,'Select Option===>',17,17,27)
11    CALL LOCATE(0,24,41)
      CALL CLRBUF
      CALL INKEY(KYCOD1,KYCOD2)
      IF (KYCOD1 .NE. 0) THEN
      CALL CSROFF
         CALL TTOUT(0,24,1,'KEY NOT DEFINED -- Hit any key to resume',
     *                  40,40,28)
      CALL CLRBUF
      CALL INKEY(KYCOD1,KYCOD2)
      CALL CLRBUF
      CALL CSRON
      CALL TTOUT(0,24,1,BLANK,80,80,31)
      GOTO 11
```

```
      ENDIF

C     IF F1 IS PRESSED, OPEN A TEMPORARY FILE 'BYE.FLG'.

      IF (KYCOD2 .EQ. 59) THEN
      CALL CLRBUF
         OPEN (30,FILE='BYE.FLG',STATUS='NEW')
         CLOSE (30)
         GOTO 100

C     F2 IS PRESSED - EDIT/TEACH A PROGRAM

      ELSEIF (KYCOD2 .EQ. 60) THEN
      CALL CLRBUF
      CALL AMLED
      GOTO 100

C     F3 IS PRESSED - COMPILE A PROGRAM

      ELSEIF (KYCOD2 .EQ. 61) THEN
      CALL CLRBUF
      CALL AMLCOM

C     F4 IS PRESSED -LOAD A PROGRAM TO CONTROLLER

      ELSEIF (KYCOD2 .EQ. 62) THEN
      CALL CLRBUF
      CALL AMLLO

C     F5 IS PRESSED -SET SYSTEM CONFIGURATION

      ELSEIF (KYCOD2 .EQ. 63) THEN
      CALL CLRBUF
      CALL AMLCONF

C     F6 IS PRESSED -COMMUNICATE WITH CONTROLLER

      ELSEIF (KYCOD2 .EQ. 64) THEN
      CALL CLRBUF
      CALL AMLCOM

C     F7 IS PRESSED -GENERATE CROSS LISTING

      ELSEIF (KYCOD2 .EQ. 65) THEN
      CALL CLRBUF
      CALL AMLREF

C     F8 IS PRESSED -SET PROGRAM NAME AND OPTION

      ELSEIF (KYCOD2 .EQ. 66) THEN
      CALL CLRBUF
      CALL MENU
      GOTO 11

C     F9 IS PRESSED -  INTERACTIVELY RUN ROBOT

      ELSEIF (KYCOD2 .EQ. 67) THEN
      CALL CLRBUF
      CALL INTER
      GO TO 100

C     F10 IS PRESSED - SIMULATE PROGRAM GRAPHICALLY

      ELSEIF (KYCOD2 .EQ. 68) THEN
      CALL CLRBUF
        OPEN (31,FILE='BETWEEN.FLG',STATUS='NEW')
       CLOSE (31)
```

```
       GOTO 100

      ELSE
      CALL CSROFF
      CALL TTOUT(0,24,1,'KEY NOT DEFINED -- Hit any key to resume',
     *40,40,28)
      CALL CLRBUF
      CALL INKEY(KYCOD1,KYCOD2)
      CALL CLRBUF
      CALL CSRON
      CALL TTOUT(0,24,1,BLANK,80,80,31)
      GOTO 11
      ENDIF

100   CALL CLS(31)
      CALL CSRON

      END


C     SUBROUTINE AMLED- (THRU F2)

C     THIS SUBROUTINE CREATES THE BATCH FILE EDTR.BAT, AND WRITES
C     THE CALL TO THE EDITOR ALONG WITH THE AML/E FILE NAME, IF
C     SPECIFIED.CONTROL RETURNS TO THE MAIN MENU WHEN EDITING IS COMPLETE

      SUBROUTINE AMLED

      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME
      INTEGER * 2 L
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
      COMMON /B/ L

      OPEN (2,FILE='EDTR.BAT',STATUS='NEW')
      WRITE (2,10)
      WRITE (2,20) NAME
      CLOSE(2)

10    FORMAT('ECHO OFF')
20    FORMAT('EDIT',1X,A20)
100   FORMAT(I2,1X,A20)

      RETURN
      END


C     SUBROUTINE AMLCOM (THRU-F3)

C     THIS SUBROUTINE CREATES THE BATCH FILE COMPILE.BAT, AND WRITES
C     THE CALL TO THE AML/E COMPILER ALONG WITH THE AML/E FILE NAME AND
C     OPTIONS, IF SPECIFIED. CONTROL RETURNS TO THE MAIN MENU PROGRAM
C     WHEN COMPILING IS COMPLETE.

      SUBROUTINE AMLCOM

      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME
      INTEGER * 2 L
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
      COMMON /B/ L

      OPEN (1,FILE='IRPSO',STATUS='OLD',ERR=201)
      OPEN (2,FILE='COMPILE.BAT',STATUS='NEW')

      WRITE (2,10)
      READ (1,100,END=15) L,NAME,C1,C2,C3,C4
```

```
      IF (L .LE. 1) GOTO 15

      IF (C1 .NE. ' ' .AND. C2 .NE. ' ') THEN

         IF (C3 .NE. ' ' .AND. C4 .NE. ' ') THEN
         WRITE (2,20) NAME,C1,C2,C3,C4
         ELSE
        WRITE (2,20) NAME,C1,C2
        ENDIF

        ELSE
        WRITE (2,20) NAME
        ENDIF

      GOTO 200

15    WRITE (2,20)

10    FORMAT('ECHO OFF')
20    FORMAT('COMPILER',1X,A20,4A1)
100   FORMAT(I2,1X,A20,4A1)

200   CLOSE(1)
      CLOSE(2)


201   RETURN
      END



C     SUBROUTINE AMLLO (THRU-F4)

C     THIS SUBROUTINE CREATES THE BATCH FILE LOD.BAT, AND WRITES
C     THE CALL TO THE LOAD MODULE. ALSO  CONTROL RETURNS
C     TO THE MAIN MENU PROGRAM WHEN THE PROGRAM HAS BEEN
C     LOADED .

      SUBROUTINE AMLLO

      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME
      INTEGER * 2 L
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
      COMMON /B/ L

      OPEN (2,FILE='LOD.BAT',STATUS='NEW')

      WRITE (2,10)
      WRITE (2,20)

10    FORMAT('ECHO OFF')
20    FORMAT('LOAD')

      CLOSE(2)

      RETURN
      END



C     SUBROUTINE AMLCONF (THRU-F5)

C     THIS SUBROUTINE CREATES THE BATCH FILE CONFIG.BAT, AND WRITES
C     THE CALL TO THE CONFIG MODULE. ALSO  CONTROL RETURNS
C     TO THE MAIN MENU PROGRAM WHEN THE SYSTEM CONFIGURATION
C     HAS BEEN SET.
```

```
      SUBROUTINE AMLCONF

      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME
      INTEGER * 2 L
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
      COMMON /B/ L

      OPEN (2,FILE='CONFIG.BAT',STATUS='NEW')

      WRITE (2,10)
      WRITE (2,20)

10    FORMAT('ECHO OFF')
20    FORMAT('CFG')

      CLOSE(2)

      RETURN
      END


C     SUBROUTINE AMLCMA (THRU-F6)

C     THIS SUBROUTINE CREATES THE BATCH FILE COMA.BAT, AND WRITES
C     THE CALL TO THE COMM MODULE. ALSO  CONTROL RETURNS
C     TO THE MAIN MENU PROGRAM WHEN THE COMMUNICATION WITH
C     THE CONTROLLER IS COMPLETE.

      SUBROUTINE AMLCMA

      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME
      INTEGER * 2 L
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
      COMMON /B/ L

      OPEN (2,FILE='COMA.BAT',STATUS='NEW')

      WRITE (2,10)
      WRITE (2,20)

10    FORMAT('ECHO OFF')
20    FORMAT('COMAID')

      CLOSE(2)

      RETURN
      END


C     SUBROUTINE AMLREF (THRU-F7)

C     THIS SUBROUTINE CREATES THE BATCH FILE REF.BAT, AND WRITES
C     THE CALL TO THE REFERENCE MODULE. ALSO  CONTROL RETURNS
C     TO THE MAIN MENU PROGRAM WHEN THE CROSS REFERENCE LISTING
C     HAS BEEN GENERATED.

      SUBROUTINE AMLREF

      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME
      INTEGER * 2 L
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
```

```
      COMMON /B/ L

      OPEN (2,FILE='REF.BAT',STATUS='NEW')

      WRITE (2,10)
      WRITE (2,20)

10    FORMAT('ECHO OFF')
20    FORMAT('XREF')

      CLOSE(2)

      RETURN
      END


C     SUBROUTINE MENU  (THRU -F8)

C     THIS SUBROUTINE DISPLAYS THE PROGRAM OPTIONS MENU AND READS
C     THE USER'S INPUT FOR THE PROGRAM NAME, AND THE COMPILER OPTIONS.

C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C     CLRBUF,CLS,CSROFF,CSROFF,ID,INKEY,TTOUT.

      SUBROUTINE MENU

      CHARACTER * 1 CHAR,ALPHA(26),NUMER(10)
      INTEGER * 2 J,KYCOD1,KYCOD2
      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME,AFILE
      INTEGER * 2 L
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
      COMMON /B/ L

      DATA ALPHA/'A','B','C','D','E','F','G','H','I','J','K','L','M',
     *'N','O','P','Q','R','S','T','U','V','W','X','Y','Z'/
      DATA NUMER/'0','1','2','3','4','5','6','7','8','9'/

      CALL CLS(31)
      CALL CSRON
      CALL TTOUT(0,1,28,'SELECT PROGRAM ',15,15,31)
      CALL TTOUT(0,2,28,'---------------',15,15,31)
      CALL TTOUT(0,4,14,'Filename (.AML assumed):',24,24,26)
      CALL TTOUT(0,5,21,'Compiler options:',17,17,26)
      CALL TTOUT(0,9,24,'Select a function:',18,18,27)
      CALL TTOUT(0,12,21,'FN1  - Return to DOS',20,20,31)
      CALL TTOUT(0,13,21,'FN2  - Edit/Teach a Program',27,27,31)
      CALL TTOUT(0,14,21,'FN3  - Compile a Program',24,24,31)
      CALL TTOUT(0,15,21,'FN4  - Load a Program to Controller',35,35,31)
      CALL TTOUT(0,16,21,'FN5  - Set System Configuration',31,31,31)
      CALL TTOUT(0,17,21,'FN6  - Communicate with Controller',33,33,31)
      CALL TTOUT(0,18,21,'FN7  - Generate Cross Reference Listing',38,
     *38,31)
      CALL TTOUT(0,19,21,'FN8  - Set Program Name and Options',34,34,31)
      CALL TTOUT(0,20,21,'FN9  - Interactively Run Robot',30,30,31)
      CALL TTOUT(0,21,21,'FN10 - Simulate Program Graphically',34,34,31)
      CALL TTOUT(0,24,24,'Select Option===>',16,16,27)
      OPEN (1,FILE='IRPSO',STATUS='NEW')
      WRITE (1,2)
      REWIND (1)

2     FORMAT('                         ')

C     INITIALIZE NAME TO BLANKS.

      DO 45 I = 1,20
```

```
         NAME(I:I) = ' '
45     CONTINUE

       C1 = ' '
       C2 = ' '
       C3 = ' '
       C4 = ' '

       L = 0

C      LOCATE CURSOR, AND DISPLAY VALID INPUTS ON THE SCREEN AND
C      CONCATENATE THE NAME.

       DO 10 I = 1,200
       L = L + 1
       IF (L .GE. 21) GOTO 20
5      J = L
       CALL LOCATE(0,4,38+L)
       CALL CLRBUF
       CALL INKEY(KYCOD1,KYCOD2)

       IF (KYCOD1 .EQ. 13) THEN
       GOTO 20
       ELSE
       CALL ID(KYCOD1,KYCOD2,CHAR,J,ALPHA,NUMER)
       IF (J .EQ. 0) THEN
        CALL CSROFF
            CALL TTOUT(0,24,1,'INVALID CHARACTER - Hit any key to resume'
     *     ,41,41,28)
        CALL CLRBUF
        CALL INKEY(KYCOD1,KYCOD2)
        CALL CLRBUF
        CALL TTOUT(0,24,1,BLANK,80,80,31)
        CALL CSRON
        GOTO 5
        ENDIF

        IF (J .LT. L) THEN
        L = J
        CALL TTOUT(0,4,J+38,BLANK(1),1,1,31)
        GOTO 5
       ELSE
        CALL TTOUT(0,4,38+L,CHAR,1,1,26)

        IF (J .NE. 1) THEN
        AFILE = NAME(1:L-1)
        CALL MODSTG (AFILE,CHAR)
        NAME(1:L) = AFILE

          ELSE
        NAME(1:L) = CHAR
        ENDIF

        ENDIF

        ENDIF

10     CONTINUE

20     CALL CLRBUF

C      POSITION CURSOR TO READ COMPILER OPTIONS.

31     CALL LOCATE(0,5,39)
       CALL INKEY(KYCOD1,KYCOD2)
       IF (KYCOD1 .EQ. 13) GOTO 36
```

```
      IF (KYCOD1 .NE. 47) THEN
      CALL CSROFF
         CALL TTOUT(0,24,1,'INVALID CHARACTER - Hit any key to resume'
     * ,41,41,28)
        CALL CLRBUF
        CALL INKEY(KYCOD1,KYCOD2)
        CALL CLRBUF
        CALL TTOUT(0,24,1,BLANK,80,80,31)
        CALL CSRON
        GOTO 31
        ELSE
          C1 = '/'
      CALL TTOUT(0,5,39,C1,1,1,26)
      ENDIF

32    CALL LOCATE(0,5,40)
      CALL INKEY(KYCOD1,KYCOD2)

        IF (KYCOD1 .EQ. 13) THEN
          C1 = ' '
      GOTO 36
      ENDIF

      CALL ID(KYCOD1,KYCOD2,CHAR,J,ALPHA,NUMER)

      IF (CHAR .NE. 'L' .AND. CHAR .NE. 'S') THEN
      CALL CSROFF
         CALL TTOUT(0,24,1,'INVALID CHARACTER - Hit any key to resume'
     * ,41,41,28)
        CALL CLRBUF
        CALL INKEY(KYCOD1,KYCOD2)
        CALL CLRBUF
        CALL TTOUT(0,24,1,BLANK,80,80,31)
        CALL CSRON
        GOTO 32
        ELSE
       C2 = CHAR
       CALL TTOUT(0,5,40,C2,1,1,26)
      ENDIF

33    CALL LOCATE(0,5,41)
      CALL INKEY(KYCOD1,KYCOD2)
      IF (KYCOD1 .EQ. 13) GOTO 36

      IF (KYCOD1 .NE. 47) THEN
      CALL CSROFF
         CALL TTOUT(0,24,1,'INVALID CHARACTER - Hit any key to resume'
     * ,41,41,28)
        CALL CLRBUF
        CALL INKEY(KYCOD1,KYCOD2)
        CALL CLRBUF
        CALL TTOUT(0,24,1,BLANK,80,80,31)
        CALL CSRON
        GOTO 33
        ELSE
          C3 = '/'
      CALL TTOUT(0,5,41,C3,1,1,26)
      ENDIF

34    CALL LOCATE(0,5,42)
      CALL INKEY(KYCOD1,KYCOD2)

      IF (KYCOD1 .EQ. 13) THEN
         C3 = ' '
      GOTO 36
      ENDIF
```

```
      CALL ID(KYCOD1,KYCOD2,CHAR,J,ALPHA,NUMER)

      IF (CHAR .NE. 'L' .AND. CHAR .NE. 'S') THEN
      CALL CSROFF
         CALL TTOUT(0,24,1,'INVALID CHARACTER - Hit any key to resume'
     *  ,41,41,28)
        CALL CLRBUF
        CALL INKEY(KYCOD1,KYCOD2)
        CALL CLRBUF
        CALL TTOUT(0,24,1,BLANK,80,80,31)
        CALL CSRON
         GOTO 34
        ELSE
         C4 = CHAR
      CALL TTOUT(0,5,42,C4,1,1,26)
      ENDIF

36    IF (C2 .EQ. C4) THEN
         C3 = ' '
         C4 = ' '
      ENDIF

C     WRITE THE PROGRAM NAME AND COMPILER OPTIONS INTO THE FILE IRPS0.

      WRITE (1,30) L,NAME(1:L-1),C1,C2,C3,C4
      REWIND (1)
      CLOSE(1)
30    FORMAT(I2,1X,A20,4A1)

      RETURN
      END



C     SUBROUTINE INTER (THRU-F9)

C     THIS SUBROUTINE CREATES THE BATCH FILE IRPS2.BAT, AND WRITES
C     THE CALL TO THE PSEUDO-COMPILER, ALONG WITH THE AML/E FILENAME,
C     IF SPECIFIED, AND ALSO THE CALL TO THE SYSTEM CONTROL PROGRAM.
C     CONTROL RETURNS TO THE MAIN MENU PROGRAM WHEN THE PROGRAM HAS BEEN
C     EXECUTED INTERACTIVELY.

      SUBROUTINE INTER

      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME
      INTEGER * 2 L
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
      COMMON /B/ L

      OPEN (2,FILE='INTER.BAT',STATUS='NEW')

      WRITE (2,10)
      WRITE (2,20)
      WRITE (2,30)

10    FORMAT('ECHO OFF')
20    FORMAT('COMPPC')
30    FORMAT('SYS1')

      CLOSE(2)

      RETURN
      END
```

```
C     BLOCK DATA

C     THIS BLOCK DEFINES THE ARRAY BLANK, WHICH IS AN ARRAY OF BLANK
C     CHARACTERS.

      BLOCK DATA

      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME
      INTEGER *2 L
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
      COMMON /B/ L

      DATA BLANK/80 * ' '/

      END


C     SUBROUTINE TTOUT

C     THIS SUBROUTINE WRITES A STRING A CHARACTERS AT A SPECIFIED
C     ROW AND COLUMN OF THE DISPLAY.

C     PARAMETER DEFINITION:

C      PAGE       -   PAGE NUMBER
C      ROW        -   ROW NUMBER
C      COLUMN     -   COLUMN NUMBER
C      BUF        -   CHARACTER BUFFER NAME
C      SIZE       -   SIZE OF BUFFER
C      LENGTH     -   LENGTH OF CHARACTER STRING
C      WRATT      -   ATTRIBUTE OF CHARACTER TO BE WRITTEN


C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C      LOCATE, WRCHAT.

      SUBROUTINE TTOUT(PAGE,ROW,COLUMN,BUF,LENGTH,SIZE,WRATT)

      INTEGER * 2 PAGE,ROW,COLUMN,SIZE,WRATT
      CHARACTER * 1 BUF(SIZE)
      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME
      INTEGER * 2 L
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
      COMMON /B/ L

      CALL LOCATE(PAGE,ROW,COLUMN)

      DO 10 I = 1,LENGTH
 CALL WRCHAT(PAGE,BUF(I),WRATT,1)
10    CONTINUE

      RETURN
      END


C     SUBROUTINE ID

C     THIS SUBROUTINE IDENTIFIES THE CHARACTER TYPED IN BY THE USER.
C     VALID CHARACTERS ARE ALL ALPHABETS, NUMBERS, THE BACKSPACE KEY,
C     THE ENTER KEY AND SOME SPECIAL CHARACTERS.

C     PARAMETER DEFINITION:

C     KYCOD1    -   ASCII CODE OF KEY
C     KYCOD2    -   EXTENDED CHARACTER CODE OF KEY
```

```
C     CHAR      -   CHARACTER IDENTIFIED
C     J         -   COLUMN POSITION
C     ALPHA     -   ARRAY OF ALPHABETS
C     NUMMER    -   ARRAY OF NUMBERS

      SUBROUTINE ID(KYCOD1,KYCOD2,CHAR,J,ALPHA,NUMER)

      INTEGER * 2 KYCOD1,KYCOD2,J
      CHARACTER * 1 CHAR,ALPHA(26),NUMER(10)
      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME
      INTEGER * 2 L
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
      COMMON /B/ L

      IF (KYCOD1 .GE. 48 .AND. KYCOD1 .LE. 57) THEN
      CHAR = NUMER(KYCOD1-47)
      ELSEIF (KYCOD1 .GE. 65 .AND. KYCOD1 .LE. 90) THEN
      CHAR = ALPHA(KYCOD1-64)
      ELSEIF (KYCOD1 .GE. 97 .AND. KYCOD1 .LE. 122) THEN
      CHAR = ALPHA(KYCOD1-96)
      ELSEIF (KYCOD1 .EQ. 29) THEN
        J = J - 1
      ELSEIF (KYCOD1 .EQ. 8 .AND. KYCOD2 .EQ. 14) THEN
        J = J - 1
      ELSEIF (KYCOD1 .EQ. 127 .AND. KYCOD2 .EQ. 14) THEN
        J = J - 1
      ELSEIF (KYCOD1 .EQ. 0 .AND. KYCOD2 .EQ. 75) THEN
        J = J - 1
      ELSEIF (KYCOD1 .EQ. 33) THEN
         CHAR = '!'
      ELSEIF (KYCOD1 .EQ. 64) THEN
         CHAR = '@'
      ELSEIF (KYCOD1 .EQ. 35) THEN
         CHAR = '#'
      ELSEIF (KYCOD1 .EQ. 36) THEN
         CHAR = '$'
      ELSEIF (KYCOD1 .EQ. 37) THEN
         CHAR = '%'
      ELSEIF (KYCOD1 .EQ. 94) THEN
         CHAR = '      '
      ELSEIF (KYCOD1 .EQ. 38) THEN
         CHAR = '&'
      ELSEIF (KYCOD1 .EQ. 40) THEN
         CHAR = '('
      ELSEIF (KYCOD1 .EQ. 41) THEN
         CHAR = ')'
      ELSEIF (KYCOD1 .EQ. 95) THEN
         CHAR = '_'
      ELSEIF (KYCOD1 .EQ. 45) THEN
         CHAR = '-'
      ELSEIF (KYCOD1 .EQ. 123) THEN
         CHAR = '{'
      ELSEIF (KYCOD1 .EQ. 125) THEN
         CHAR = '}'
      ELSEIF (KYCOD1 .EQ. 39) THEN
         CHAR = ''''
      ELSEIF (KYCOD1 .EQ. 58) THEN
         CHAR = ':'
      ELSEIF (KYCOD1 .EQ. 47) THEN
         CHAR = '/'
      ELSEIF (KYCOD1 .EQ. 92) THEN
         CHAR = '\'
      ELSE
         J = 0
      ENDIF
```

```
      RETURN
      END


C     SUBROUTINE MODSTG

C      THIS SUBROUTINE CONCATENATES TWO STRINGS
C
C       AFILE     FIRST STRING
C       AFIX      SECOND STRING
C
C       AFILE     COMBINED STRING AND UPPERCASE
C

      SUBROUTINE  MODSTG (AFILE,AEXT)

      CHARACTER *20 AFILE
      CHARACTER*1 AEXT

100   DO 130 I = 1,20
       IF(AFILE(I:I) .EQ. ' ') GOTO 140
130   CONTINUE


 140  I = I - 1
      DO 150 J =1,1
      AFILE(I+J:I+J) = AEXT(J:J)
 150  CONTINUE


C     CAPITALIZE

      DO 200 K = I,I+1
      IF(AFILE(K:K) .GE. 'a' .AND. AFILE(K:K) .LE. 'z' ) then
      AFILE(K:K) = CHAR (ICHAR (AFILE(K:K)) -32 )
      ENDIF
 200  CONTINUE

      RETURN
      END

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

C     (2) BETWEEN - PROGRAM TO DISPLAY THE MAIN SIMULATION MENU.

C     THIS PROGRAM, BETWEEN.FOR, IS THE PROGRAM FOR THE MENU WHERE THE
C     USER CHOOSES BETWEEN SIMULATING AND DEFINING EQUIPMENTS/WRKCELL.
C     THAT IS CHOOSING BETWEEN THE GRAPHICAL SIMULATION MENU AND THE
C     WORK CELL LAYOUT MENU.

C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C     CLRBUF,CLS,CSROFF,CSRON,DSWAP,INKEY,TTOUT

      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME,WKNAME
      INTEGER * 2 J,L,KYCOD1,KYCOD2,BASC
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
      COMMON /B/ L
      COMMON /H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON /J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON /O/ XARM,YARM,ZARM,THETA,PI,GRIPX,GRIPY
      COMMON /G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC

      XMIN  = -800.0
      XMAX  =  800.0
      YMIN  = -650.0
      YMAX  =  950.0
```

```
      SXMIN = -420.0
      SXMAX =  684.0
      SYMIN = -420.0
      SYMAX =  684.0
      KOUNT =  1
      GRIPX =  175.0
      GRIPY =  0.0
      L     =  0
      XVAL  =  650.0
      YVAL  =  0.0
      ZVAL  =  0.0
      RVAL  =  0.0
      BASC  =  80

      CALL DSWAP(0)
      CALL CLS(31)
      CALL CSRON
      CALL CLRBUF

      CALL TTOUT(0,12,24,'All selections using alphabets',30,30,28)
      CALL TTOUT(0,14,24,'must be in CAPITAL LETTERS',26,26,28)
      CALL TTOUT(0,16,24,'Press any key to Continue....',29,29,28)
      CALL INKEY(KYCOD1,KYCOD2)
      CALL CLS(31)
      CALL CSRON
      CALL CLRBUF


      CALL TTOUT(0,1,23,'SYSTEM FOR IBM 7545 ROBOT',25,25,
     *31)
      CALL TTOUT(0,3,12,
     *'GRAPHICAL SIMULATION AND INTERACTIVE PROGRAMMING',
     *48,48,31)
      CALL TTOUT(0,5,25,'MAIN SIMULATION MENU',20,18,27)
      CALL TTOUT(0,6,25,'--------------------',20,18,31)
      CALL TTOUT(0,8,24,' ',1,1,31)
      CALL TTOUT(0,9,24,'Select a function:',19,19,27)
      CALL TTOUT(0,12,21,'FN1 - Previous Menu',19,19,31)
      CALL TTOUT(0,13,21,'FN2 - Simulate Program',22,22,31)
      CALL TTOUT(0,14,21,
     *'FN3 - Define Equipment/Arrange Workcell',39,39,31)
      CALL TTOUT(0,20,24,'Select Option===>',17,17,27)
11    CALL LOCATE(0,20,41)
      CALL CLRBUF
      CALL INKEY(KYCOD1,KYCOD2)
      IF (KYCOD1 .NE. 0) THEN
         CALL CSROFF
         CALL TTOUT(0,24,1,'KEY NOT DEFINED -- Hit any key to resume',
     *   40,40,28)
         CALL CLRBUF
         CALL INKEY(KYCOD1,KYCOD2)
         CALL CLRBUF
         CALL CSRON
         CALL TTOUT(0,24,1,BLANK,80,80,31)
         GOTO 11
      ENDIF

C     F1 IS SELECTED - QUIT PROGRAM

      IF (KYCOD2 .EQ. 59) THEN
         CALL CLRBUF
         OPEN(30,FILE='FIRST.FLG',STATUS='NEW')
         CLOSE(30)
         GO TO 100
```

```
C     F2 IS SLELCTED - CALL MAIN SIMULATION MENU. THE FLAG MENUTH.FLG
C     IS CREATED FOR THE BATCH FILE GASP TO TRANSFER CONTROL TO THAT
C     MENU.

      ELSEIF (KYCOD2 .EQ. 60) THEN
         CALL CLRBUF
         OPEN(23,FILE='MENUTH.FLG',STATUS='NEW')
         CLOSE(23)
         CODE = 4.0
         CALL DEPUTY1(XVAL,YVAL,ZVAL,RVAL)
         CALL DSWAP(0)
         GO TO 100


C     F3 IS SELECTED - CALL WORK CELL LAYOUT MENU. THE FLAG MENUTH.FLG
C     IS CREATED FOR THE BATCH FILE GASP TO TRANSFER CONTROL TO THAT
C     MENU.

      ELSEIF (KYCOD2 .EQ. 61) THEN
         CALL CLRBUF
         OPEN(24,FILE='MENUTH.FLG',STATUS='NEW')
         CLOSE(24)
         GO TO 100

      ELSE
         CALL CSROFF
         CALL TTOUT(0,24,1,'KEY NOT DEFINED -- Hit any key to resume',
     *   40,40,28)
         CALL CLRBUF
         CALL INKEY(KYCOD1,KYCOD2)
         CALL CLRBUF
         CALL CSRON
         CALL TTOUT(0,24,1,BLANK,80,80,31)
         GOTO 11
      ENDIF

100   CALL CLS(0)
      CALL CSRON

      STOP
      END

C     BLOCK DATA

C     THIS BLOCK DEFINES THE ARRAY BLANK, WHICH IS AN ARRAY OF BLANK
C     CHARACTERS.

      BLOCK DATA

      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME, WKNAME
      INTEGER *2 L
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
      COMMON /B/ L

      DATA BLANK/80 * ' '/

      END

C     SUBROUTINE TTOUT

C     THIS SUBROUTINE WRITES A STRING A CHARACTERS AT A SPECIFIED
C     ROW AND COLUMN OF THE DISPLAY.


C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C             LOCATE, WRCHAT.
```

```
      SUBROUTINE TTOUT(PAGE,ROW,COLUMN,BUF,LENGTH,SIZE,WRATT)

      INTEGER * 2 PAGE,ROW,COLUMN,SIZE,WRATT
      CHARACTER * 1 BUF(SIZE)
      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME,WKNAME
      INTEGER * 2 L
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
      COMMON /B/ L

      CALL LOCATE(PAGE,ROW,COLUMN)

      DO 10 I = 1,LENGTH
         CALL WRCHAT(PAGE,BUF(I),WRATT,1)
10    CONTINUE

      RETURN
      END


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

C     (3) MENU2 - PROGRAM TO DISPLAY THE GRAPHICAL SIMULATION MENU

C     THIS PROGRAM, MENU2.FOR, IS THE  PROGRAM TO DISPLAY THE GRAPHICAL
C     SIMULATION MENU. THIS PROGRAM IS CALLED BY THE "MENU2.BAT" FILE.
C     IT ALLOWS THE USER TO GO TO THE VIEWS MENU, SETUP THE WORKCELL
C     NAME AND DISPLAY IT, GO BACK TO THE MAIN SIMULATION MENU, OR
C     SIMULATE THE PROGRAM IN THE STANDARD VIEW.


C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C      CLRBUF,CLS,CSROFF,CSRON,INKEY,SETMEN,CONCA,MODSTG

      CHARACTER * 1 BLANK(80)
      CHARACTER * 20 NAME,CELNAM*10
      INTEGER * 2 J,L,KYCOD1,KYCOD2,BASC
      COMMON /A/ NAME,BLANK
      COMMON /B/ L
      COMMON /H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON /J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON /O/ XARM,YARM,ZARM,THETA,PI,GRIPX,GRIPY
      COMMON /G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC

      XMIN  = -800.0
      XMAX  =  800.0
      YMIN  = -650.0
      YMAX  =  950.0
      SXMIN = -420.0
      SXMAX =  684.0
      SYMIN = -420.0
      SYMAX =  684.0
      KOUNT =  1
      GRIPX =  175.0
      GRIPY =  0.0
      L     = 0
      XVAL  = 650.0
      YVAL  = 0.0
      ZVAL  = 0.0
      RVAL  = 0.0
      BASC  = 80


      OPEN (31,FILE='WORK',STATUS='OLD',ERR=66)

C     READ THE FILE WORK FOR WORKCELL NAME, IF ANY.
```

```
      READ (31,110,END=10) L,CELNAM

 110  FORMAT(I2,1X,A10)
 10   CLOSE  (31)

 66   CALL DSWAP(0)
      CALL CSRON
      CALL CLRBUF

C     DISPLAY MAIN MENU IF NO FILENAME  EXISTS IN THE FILE "WORK",
C     OTHERWISE DISPLAY THE FACILITY TO ENTER FILENAME.

      CALL TTOUT(0,1,23,'SYSTEM FOR IBM 7545 ROBOT',25,25,
     *31)
      CALL TTOUT(0,3,12,
     *'GRAPHICAL SIMULATION AND INTERACTIVE PROGRAMMING',
     *48,48,31)
         CALL TTOUT(0,5,25,'GRAPHICAL SIMULATION MENU',25,25,31)
         CALL TTOUT(0,6,25,'------------------------',25,25,31)

C     CHECK FILE CELNAM FOR WORKCELL NAME . IF NO NAME THEN DISPLAY
C     THE OPTIONS ONLY. IF NAME EXISTS THEN SHOW NAME OF WORKCELL NAME
C     THAT IS IN CELNAM TO THE USER.

      IF (L .LE. 1) THEN
      GOTO 67
      ELSE

 800  OPEN (21,FILE=CELNAM,STATUS='OLD',ERR=81)
      CLOSE(21)
      GOTO 802
 81   CALL CLS(31)
      CALL TTOUT(0,17,24,'This workcell does not exist',28,28,27)
      GOTO  65
802   CODE = 4.0
      CALL DEPUTY1(XVAL,YVAL,ZVAL,RVAL)
      CALL DSWAP(0)
      ENDIF


65    CALL TTOUT(0,1,23,'SYSTEM FOR IBM 7545 ROBOT',25,25,
     *31)
      CALL TTOUT(0,3,12,
     *'GRAPHICAL SIMULATION AND INTERACTIVE PROGRAMMING',
     *48,48,31)
         CALL TTOUT(0,5,25,'GRAPHICAL SIMULATION MENU',25,25,31)
         CALL TTOUT(0,6,25,'------------------------',25,25,31)
      CALL TTOUT(0,7,14,'Workcell Name',13,13,26)
      CALL TTOUT(0,7,29,CELNAM,10,10,26)

67    CALL TTOUT(0,9,24,'Select a function:',18,18,27)
      CALL TTOUT(0,12,21,'FN1 - Previous Menu',19,19,31)
      CALL TTOUT(0,13,21,'FN2 - Simulate Program',22,22,31)
      CALL TTOUT(0,14,21,'FN3 - Alter Viewing Parameters',30,30,31)
      CALL TTOUT(0,15,21,'FN4 - Set Workcell Name',23,23,31)
      CALL TTOUT(0,20,24,'Select Option===>',17,17,27)
11    CALL LOCATE(0,20,41)
      CALL CLRBUF
      CALL INKEY(KYCOD1,KYCOD2)
      IF (KYCOD1 .NE. 0) THEN
         CALL CSROFF
         CALL TTOUT(0,24,1,'KEY NOT DEFINED -- Hit any key to resume',
     *   40,40,28)
         CALL CLRBUF
         CALL INKEY(KYCOD1,KYCOD2)
         CALL CLRBUF
```

```
         CALL CSRON
         CALL TTOUT(0,24,1,BLANK,80,80,31)
         GOTO 11
      ENDIF

C     F1 IS PRESSED - PREVIOUS MENU

      IF (KYCOD2 .EQ. 59) THEN
         CALL CLRBUF
         OPEN(30,FILE='BETWEEN.FLG',STATUS='NEW')
         CLOSE(30)
         GO TO 100

C     F2 IS PRESSED- SIMULATE PROGRAM

      ELSEIF (KYCOD2 .EQ. 60) THEN
         CALL CLRBUF
         OPEN(32,FILE='SIM.FLG',STATUS='NEW')
         CLOSE(32)
         GOTO 100

C     F3 IS PRESSED - ALTER VIEWING PARAMETERS

      ELSEIF (KYCOD2 .EQ. 61) THEN
         CALL CLRBUF
         OPEN(33,FILE='ALT.FLG',STATUS='NEW')
         CLOSE(33)
         GOTO 100

C     F4 IS PRESSED - SET WORKCELL NAME

      ELSEIF (KYCOD2 .EQ. 62) THEN
         CALL SETMEN
         GOTO 66
      ELSE
         CALL CSROFF
         CALL TTOUT(0,24,1,'KEY NOT DEFINED -- Hit any key to resume',
     *   40,40,28)
         CALL CLRBUF
         CALL INKEY(KYCOD1,KYCOD2)
         CALL CLRBUF
         CALL CSRON
         CALL TTOUT(0,24,1,BLANK,80,80,31)
         GOTO 11
      ENDIF

100   CALL CLS
      CALL CSRON

      END


C     SUBROUTINE SETMEN (THRU -F4)
C     THIS SUBROUTINE DISPLAYS THE MENU2 OPTIONS AND READS
C     THE USER'S INPUT FOR THE WORKCELL NAME.


C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C           CLRBUF,CLS,CSROFF,CSROFF,ID,INKEY,TTOUT.

      SUBROUTINE SETMEN

      CHARACTER * 1 CHAR,ALPHA(26),NUMER(10)
      INTEGER * 2 J,KYCOD1,KYCOD2,L,BASC
      CHARACTER * 1 BLANK(80),AEXT*3
      CHARACTER * 20 NAME, CELNAM*10,AFILE*10
      COMMON /A/ NAME,BLANK
```

```
      COMMON /B/ L
      COMMON /H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON /J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON /O/ XARM,YARM,ZARM,THETA,PI,GRIPX,GRIPY
      COMMON /G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC


      DATA ALPHA/'A','B','C','D','E','F','G','H','I','J','K','L','M',
     *'N','O','P','Q','R','S','T','U','V','W','X','Y','Z'/
      DATA NUMER/'0','1','2','3','4','5','6','7','8','9'/

800   CALL CLS(31)
      CALL CSRON
      CALL TTOUT(0,1,23,'SYSTEM FOR IBM 7545 ROBOT',25,25,
     *31)
      CALL TTOUT(0,3,12,
     *'GRAPHICAL SIMULATION AND INTERACTIVE PROGRAMMING',
     *48,48,31)
          CALL TTOUT(0,5,30,'MENU TWO',8,8,31)
          CALL TTOUT(0,6,30,'--------',8,8,31)
      CALL TTOUT(0,8,14,'Workcell Name (.CEL assumed)',28,28,26)
      CALL TTOUT(0,10,24,'Select a function:',18,18,27)
      CALL TTOUT(0,12,21,'FN1 - Previous Menu',19,19,31)
      CALL TTOUT(0,13,21,'FN2 - Simulate Program',22,22,31)
      CALL TTOUT(0,14,21,'FN3 - Alter Viewing Parameters',30,30,31)
      CALL TTOUT(0,15,21,'FN4 - Set Workcell Name',23,23,31)
      CALL TTOUT(0,20,24,'Select Option===>',17,17,27)

      OPEN (31,FILE='WORK',STATUS='NEW')
      WRITE (31,2)
      REWIND (31)


2     FORMAT('          ')

C     INITIALIZE CELNAM TO BLANKS.

      DO 45 I = 1,10
       CELNAM(I:I) = ' '
45    CONTINUE


      L = 0

C     LOCATE CURSOR, AND DISPLAY VALID INPUTS ON THE SCREEN AND
C     CONCATENATE THE NAME.

      DO 10 I = 1,200
         L = L + 1
         IF (L .GE. 21) GOTO 20
  5      J = L
         CALL LOCATE(0,8,43+L)
         CALL CLRBUF
         CALL INKEY(KYCOD1,KYCOD2)

         IF (KYCOD1 .EQ. 13) THEN
            GOTO 20
         ELSE
            CALL ID(KYCOD1,KYCOD2,CHAR,J,ALPHA,NUMER)
            IF (J .EQ. 0) THEN
               CALL CSROFF
           CALL TTOUT(0,24,1,'INVALID CHARACTER - Hit any key to resume'
     *      ,41,41,28)
               CALL CLRBUF
               CALL INKEY(KYCOD1,KYCOD2)
               CALL CLRBUF
               CALL TTOUT(0,24,1,BLANK,80,80,31)
```

```
            CALL CSRON
            GOTO 5
         ENDIF

         IF (J .LT. L) THEN
            L = J
            CALL TTOUT(0,8,J+43,BLANK(1),1,1,31)
            GOTO 5
         ELSE
            CALL TTOUT(0,8,43+L,CHAR,1,1,26)

            IF (J .NE. 1) THEN
               AFILE = CELNAM(1:L-1)
               CALL MODSTG (AFILE,CHAR)
               CELNAM(1:L) = AFILE
            ELSE
             CELNAM(1:L) = CHAR
            ENDIF

         ENDIF

      ENDIF

10    CONTINUE

C     WRITE THE WORKCELL NAME INTO THE FILE WORK.
C     ADD THE EXTENSION .CEL BEFORE THAT

20    CALL CLRBUF

         AFILE = CELNAM
         CALL CONCA(AFILE,AEXT)
         CELNAM = AFILE

      WRITE (31,30) L,CELNAM
30    FORMAT(I2,1X,A10)
      CLOSE(31)

      RETURN
      END


C      SUBROUTINE CONCA

      SUBROUTINE  CONCA (AFILE,AEXT)

      CHARACTER *20 AFILE
      CHARACTER*3 AEXT

      AEXT = 'CEL'

100   DO 130 I = 1,20
       IF(AFILE(I:I) .EQ. ' ') GO TO 140
130   CONTINUE

140   AFILE(I:I) = '.'
      DO 150 J =1,3
      AFILE(I+J:I+J) = AEXT(J:J)
150   CONTINUE


C     CAPITALIZE

      DO 200 K = 1,I+J
      IF(AFILE(K:K) .GE. 'a' .AND.  AFILE(K:K) .LE. 'z' ) then
          AFILE(K:K) = CHAR (ICHAR (AFILE(K:K)) -32 )
```

```
      ENDIF
200   CONTINUE

      RETURN
      END

C     BLOCK DATA

C     THIS BLOCK DEFINES THE ARRAY BLANK, WHICH IS AN ARRAY OF BLANK
C     CHARACTERS.

      BLOCK DATA

      CHARACTER * 1 BLANK(80)
      CHARACTER * 20 NAME, CELNAM*10
      INTEGER *2 L
      COMMON /A/ NAME,BLANK
      COMMON /B/ L

      DATA BLANK/80 * ' '/

      END


C     SUBROUTINE TTOUT

C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C             LOCATE, WRCHAT.

      SUBROUTINE TTOUT(PAGE,ROW,COLUMN,BUF,LENGTH,SIZE,WRATT)

      INTEGER * 2 PAGE,ROW,COLUMN,SIZE,WRATT,L
      CHARACTER * 1 BUF(SIZE),BLANK(80)
      CHARACTER * 20 NAME,CELNAM*10
      COMMON /A/ NAME,BLANK
      COMMON /B/ L

      CALL LOCATE(PAGE,ROW,COLUMN)

      DO 10 I = 1,LENGTH
         CALL WRCHAT(PAGE,BUF(I),WRATT,1)
10    CONTINUE

      RETURN
      END


C      SUBROUTINE ID


      SUBROUTINE ID(KYCOD1,KYCOD2,CHAR,J,ALPHA,NUMER)

      INTEGER * 2 KYCOD1,KYCOD2,J,L
      CHARACTER * 1 CHAR,ALPHA(26),NUMER(10),BLANK(80)
      CHARACTER * 20 NAME,CELNAM*10
      COMMON /A/ NAME,BLANK
      COMMON /B/ L

      IF (KYCOD1 .GE. 48 .AND. KYCOD1 .LE. 57) THEN
         CHAR = NUMER(KYCOD1-47)
      ELSEIF (KYCOD1 .GE. 65 .AND. KYCOD1 .LE. 90) THEN
         CHAR = ALPHA(KYCOD1-64)
      ELSEIF (KYCOD1 .GE. 97 .AND. KYCOD1 .LE. 122) THEN
         CHAR = ALPHA(KYCOD1-96)
      ELSEIF (KYCOD1 .EQ. 29) THEN
         J = J - 1
```

```
      ELSEIF (KYCOD1 .EQ. 8 .AND. KYCOD2 .EQ. 14) THEN
         J = J - 1
      ELSEIF (KYCOD1 .EQ. 127 .AND. KYCOD2 .EQ. 14) THEN
         J = J - 1
      ELSEIF (KYCOD1 .EQ. 0 .AND. KYCOD2 .EQ. 75) THEN
         J = J - 1
      ELSEIF (KYCOD1 .EQ. 33) THEN
         CHAR = '!'
      ELSEIF (KYCOD1 .EQ. 64) THEN
         CHAR = '@'
      ELSEIF (KYCOD1 .EQ. 35) THEN
         CHAR = '#'
      ELSEIF (KYCOD1 .EQ. 36) THEN
         CHAR = '$'
      ELSEIF (KYCOD1 .EQ. 37) THEN
         CHAR = '%'
      ELSEIF (KYCOD1 .EQ. 94) THEN
         CHAR = '      '
      ELSEIF (KYCOD1 .EQ. 38) THEN
         CHAR = '&'
      ELSEIF (KYCOD1 .EQ. 40) THEN
         CHAR = '('
      ELSEIF (KYCOD1 .EQ. 41) THEN
         CHAR = ')'
      ELSEIF (KYCOD1 .EQ. 95) THEN
         CHAR = '_'
      ELSEIF (KYCOD1 .EQ. 45) THEN
         CHAR = '-'
      ELSEIF (KYCOD1 .EQ. 123) THEN
         CHAR = '{'
      ELSEIF (KYCOD1 .EQ. 125) THEN
         CHAR = '}'
      ELSEIF (KYCOD1 .EQ. 39) THEN
         CHAR = ''''
      ELSEIF (KYCOD1 .EQ. 58) THEN
         CHAR = ':'
      ELSEIF (KYCOD1 .EQ. 47) THEN
         CHAR = '/'
      ELSEIF (KYCOD1 .EQ. 92) THEN
         CHAR = '\'
      ELSE
         J = 0
      ENDIF

      RETURN
      END

C      SUBROUTINE MODSTG

      SUBROUTINE  MODSTG (AFILE,AEXT)

C      THIS SUBROUTINE CONCATENATES TWO STRINGS
C
C          AFILE    FIRST STRING
C          AFIX     SECOND STRING
C
C          AFILE    COMBINED STRING AND UPPERCASE

      CHARACTER *20 AFILE
      CHARACTER*1 AEXT

100   DO 130 I = 1,20
       IF(AFILE(I:I) .EQ. ' ') GOTO 140
130   CONTINUE

 140  I = I - 1
      DO 150 J =1,1
```

```
      AFILE(I+J:I+J) = AEXT(J:J)
 150  CONTINUE

C     CAPITALIZE

      DO 200 K = I,I+1
      IF(AFILE(K:K) .GE. 'a' .AND. AFILE(K:K) .LE. 'z' ) then
          AFILE(K:K) = CHAR (ICHAR (AFILE(K:K)) -32 )
      ENDIF
 200  CONTINUE

      RETURN
      END

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

C     (4) MENU3 - PROGRAM TO DISPLAY THE WORKCELL LAYOUT MENU

C     THIS PROGRAM, MENU3.FOR, IS THE  PROGRAM FOR MENU3.
C     THIS PROGRAM IS CALLED BY THE "MENU3.BAT" FILE.
C     THIS PROGRAM PUTS UP THE MENU3 FOR DEFINING THE WORKCELL AND
C     AND CREATING THE OBJECTS.


C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C      CLRBUF,CLS,CSROFF,CSRON,INKEY,TTOUT

      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME
      INTEGER * 2 J,L,KYCOD1,KYCOD2
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
      COMMON /B/ L


C     DISPLAY MAIN MENU

      CALL CLS(31)

      CALL TTOUT(0,1,23,'SYSTEM FOR IBM 7545 ROBOT',25,25,
     *31)
      CALL TTOUT(0,3,12,
     *'GRAPHICAL SIMULATION AND INTERACTIVE PROGRAMMING',
     *48,48,31)
          CALL TTOUT(0,5,25,'WORKCELL LAYOUT MENU',20,20,31)
          CALL TTOUT(0,6,25,'--------------------',20,20,31)
      CALL TTOUT(0,9,24,'Select a function:',18,18,27)
       CALL TTOUT(0,12,21,'FN1 - Previous Menu',19,19,31)
      CALL TTOUT(0,13,21,
     *'FN2 - Define/Display Individual Equipment',41,41,31)
      CALL TTOUT(0,14,21,'FN3 - Arrange Workcell',22,22,31)
      CALL TTOUT(0,15,21,'FN4 - Display Workcell Layout',29,29,31)
      CALL TTOUT(0,16,21,'FN5 - Directory of Equipments/Workcells',39,
     *39,31)
       CALL TTOUT(0,20,24,'Select Option===>',17,17,27)
11    CALL LOCATE(0,20,41)
      CALL CLRBUF
      CALL INKEY(KYCOD1,KYCOD2)
      IF (KYCOD1 .NE. 0) THEN
          CALL CSROFF
          CALL TTOUT(0,24,1,'KEY NOT DEFINED -- Hit any key to resume',
     *    40,40,28)
          CALL CLRBUF
          CALL INKEY(KYCOD1,KYCOD2)
          CALL CLRBUF
          CALL CSRON
          CALL TTOUT(0,24,1,BLANK,80,80,31)
```

```
         GOTO 11
      ENDIF

C     F1 IS PRESSED - QUIT PROGRAM

      IF (KYCOD2 .EQ. 59) THEN
         CALL CLRBUF
         OPEN(24,FILE='BETWEEN.FLG',STATUS='NEW')
         CLOSE(24)
         GO TO 100


C     F2 IS PRESSED - DISPLAY INDIVIDUAL EQUIPMENTS

      ELSEIF (KYCOD2 .EQ. 60) THEN
         CALL CLRBUF
         OPEN(25,FILE='DISPEQP.FLG',STATUS='NEW')
         CLOSE(25)


C     F3 IS PRESSED- ARRANGE WORKCELL

      ELSEIF (KYCOD2 .EQ. 61) THEN
         CALL CLRBUF
         OPEN(26,FILE='DEFSETUP.FLG',STATUS='NEW')
         CLOSE(26)

C     F4 IS PRESSED-DISPLAY WORKCELL

      ELSEIF (KYCOD2 .EQ. 62) THEN
         CALL CLRBUF
         OPEN(27,FILE='DISPSETU.FLG',STATUS='NEW')
         CLOSE(27)

C     F5 IS PRESSED - DIRECTORY OF AVAILABLE EQUIPMENTS

      ELSEIF (KYCOD2 .EQ. 63) THEN
         CALL CLRBUF
         OPEN(28,FILE='DIR.FLG',STATUS='NEW')
         CLOSE(28)
         GO TO 100


      ELSE
         CALL CSROFF
         CALL TTOUT(0,24,1,'KEY NOT DEFINED -- Hit any key to resume',
     *   40,40,28)
         CALL CLRBUF
         CALL INKEY(KYCOD1,KYCOD2)
         CALL CLRBUF
         CALL CSRON
         CALL TTOUT(0,24,1,BLANK,80,80,31)
         GOTO 11
      ENDIF

100   CLOSE(1)
      CLOSE(2)
      CALL CLS(0)
      CALL CSRON

      STOP
      END


C     BLOCK DATA
```

```
C     THIS BLOCK DEFINES THE ARRAY BLANK, WHICH IS AN ARRAY OF BLANK
C     CHARACTERS.

      BLOCK DATA

      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME
      INTEGER *2 L
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
      COMMON /B/ L

      DATA BLANK/80 * ' '/

      END

C     SUBROUTINE TTOUT

C     THIS SUBROUTINE WRITES A STRING A CHARACTERS AT A SPECIFIED
C     ROW AND COLUMN OF THE DISPLAY.


C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C             LOCATE, WRCHAT.

      SUBROUTINE TTOUT(PAGE,ROW,COLUMN,BUF,LENGTH,SIZE,WRATT)

      INTEGER * 2 PAGE,ROW,COLUMN,SIZE,WRATT
      CHARACTER * 1 BUF(SIZE)
      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME
      INTEGER * 2 L
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
      COMMON /B/ L

      CALL LOCATE(PAGE,ROW,COLUMN)

      DO 10 I = 1,LENGTH
         CALL WRCHAT(PAGE,BUF(I),WRATT,1)
10    CONTINUE

      RETURN
      END

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

C    (5) ALTER - PROGRAM TO DISPLAY THE ALTER VIEWS MENU.

C    THIS PROGRAM, ALTER.FOR, IS THE  PROGRAM FOR ALTERING VIEWS
C    SELECTION. THIS PROGRAM IS CALLED BY THE BATCH FILE AFTER THE
C    ALT.FLG HAS BEEN FLAGED. THIS PROGRAM PUTS UP THE VIEWS MENU
C    FOR THE USER TO CHOOSE THE DESIRED VIEW AND SIZE, AND SIMULATE.
C    THIS PROGRAM IS LINKED WITH THE PROGRAM DEPUTY1 FOR STANDARD VIEW
C    TO BE REDRAWN, FOR SWITCH VIEW, AND ZOOM.

C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C      CLRBUF,CLS,CSROFF,CSRON,INKEY,TTOUT,DEPUTY1,DSWAP

      CHARACTER * 1 BLANK(80)
      CHARACTER * 20 NAME,CELNAM*10
      INTEGER * 2 J,L,KYCOD1,KYCOD2
      INTEGER BASC
      REAL XVAL,YVAL,ZVAL,RVAL,CODE,THONI,THTWI,ZIN,RIN
      REAL XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX

      COMMON /H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON /J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON /P/ NAME,BLANK,L
```

```
      DATA XVAL/650.0/
      DATA YVAL/0.0/
      DATA ZVAL/-250.0/
      DATA RVAL/0.0/

      XMIN  = -800.0
      XMAX  =  800.0
      YMIN  = -650.0
      YMAX  =  950.0
      SXMIN = -420.0
      SXMAX =  684.0
      SYMIN = -420.0
      SYMAX =  684.0
      BASC  =  80

      OPEN(25,FILE='WORK',STATUS='OLD',ERR=104)
      READ(25,170)CELNAM
 170  FORMAT(3X,A10)
      CLOSE(24)

104   OPEN (22,FILE='INFO',STATUS='NEW')
      WRITE(22,250)BASC,XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
 250  FORMAT(1X,I2,1X,F8.2,1X,F8.2,1X,F8.2,1X,F8.2,1X,F8.2,
     *1X,F8.2,1X,F8.2,1X,F8.2)
      CLOSE(22)

      CODE = 4.0
      CALL DEPUTY1(XVAL,YVAL,ZVAL,RVAL)
      CALL DSWAP(0)

 103  CALL DSWAP(0)
      CALL CSRON
      CALL CLRBUF

      CALL TTOUT(0,1,23,'SYSTEM FOR IBM 7545 ROBOT',25,25,
     *31)
      CALL TTOUT(0,3,12,
     *'GRAPHICAL SIMULATION AND INTERACTIVE PROGRAMMING',
     *48,48,31)
         CALL TTOUT(0,5,30,'VIEWS MENU',10,10,31)
         CALL TTOUT(0,6,30,'----------',10,10,31)
      CALL TTOUT(0,9,24,'Select a function:',18,18,27)
      CALL TTOUT(0,12,21,'FN1 - Previous Menu',19,19,31)
      CALL TTOUT(0,13,21,'FN2 - Zoom/Pan',14,14,31)
      CALL TTOUT(0,14,21,'FN3 - Switch Plan/Side View',27,27,31)
      CALL TTOUT(0,15,21,'FN4 - Standard View',19,19,31)
      CALL TTOUT(0,16,21,'FN5 - Simulate Program',22,22,31)
      CALL TTOUT(0,20,24,'Select Option===>',17,17,27)
11    CALL LOCATE(0,20,41)
      CALL CLRBUF
      CALL INKEY(KYCOD1,KYCOD2)
      IF (KYCOD1 .NE. 0) THEN
         CALL CSROFF
         CALL TTOUT(0,24,1,'KEY NOT DEFINED -- Hit any key to resume',
     *   40,40,28)
         CALL CLRBUF
         CALL INKEY(KYCOD1,KYCOD2)
         CALL CLRBUF
         CALL CSRON
         CALL TTOUT(0,24,1,BLANK,80,80,31)
         GOTO 11
      ENDIF

C     F1 IS PRESSED - PREVIOUS MENU(MENU TWO)

      IF (KYCOD2 .EQ. 59) THEN
```

```
         CALL CLRBUF
        OPEN(23,FILE='MENUTH.FLG',STATUS = 'NEW ')
         CLOSE(23)
         GO TO 100

C     F2 IS PRESSED- ZOOM/PAN

      ELSEIF (KYCOD2 .EQ. 60) THEN
        CALL CLRBUF
         CODE = 5.0
      CALL DEPUTY1(XVAL,YVAL,ZVAL,RVAL)
         CALL DSWAP(0)
         GO TO 103


C     F3 IS PRESSED-SWITCH VIEWS

      ELSEIF (KYCOD2 .EQ. 61) THEN
         CALL CLRBUF
         CODE = 6.0
      CALL DEPUTY1(XVAL,YVAL,ZVAL,RVAL)
         CALL DSWAP(0)
         GO TO 103

C     F4 IS PRESSED-STANDARD VIEW

      ELSEIF (KYCOD2 .EQ. 62) THEN
         CALL CLRBUF
         CODE = 7.0
      CALL DEPUTY1(XVAL,YVAL,ZVAL,RVAL)
         CALL DSWAP(0)
        GO TO 103

C     F5 IS PRESSED-SIMULATE PROGRAM

      ELSEIF (KYCOD2 .EQ. 63) THEN
         CALL CLRBUF
         OPEN(15,FILE='SIM1.FLG',STATUS='NEW')
         CLOSE(15)
         GO TO 100

      ELSE
         CALL CSROFF
         CALL TTOUT(0,24,1,'KEY NOT DEFINED -- Hit any key to resume',
     *   40,40,28)
         CALL CLRBUF
         CALL INKEY(KYCOD1,KYCOD2)
         CALL CLRBUF
         CALL CSRON
         CALL TTOUT(0,24,1,BLANK,80,80,31)
         GOTO 11
      ENDIF

100   CALL CLS
      CALL CSRON

      END


C     BLOCK DATA

C     THIS BLOCK DEFINES THE ARRAY BLANK, WHICH IS AN ARRAY OF BLANK
C     CHARACTERS.

      BLOCK DATA

      CHARACTER * 1 BLANK(80)
```

```
      CHARACTER * 20 NAME,CELNAM*10
      INTEGER *2 L

      COMMON /H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON /J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON /P/ NAME,BLANK,L

      DATA BLANK/80 * ' '/

      END

C     SUBROUTINE TTOUT

C     THIS SUBROUTINE WRITES A STRING A CHARACTERS AT A SPECIFIED
C     ROW AND COLUMN OF THE DISPLAY.


C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C             LOCATE, WRCHAT.

      SUBROUTINE TTOUT(PAGE,ROW,COLUMN,BUF,LENGTH,SIZE,WRATT)

      INTEGER * 2 PAGE,ROW,COLUMN,SIZE,WRATT
      CHARACTER * 1 BUF(SIZE)
      CHARACTER * 1 BLANK(80)
      CHARACTER * 20 NAME,CELNAM*10
      INTEGER * 2 L

      COMMON /H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON /J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON /P/ NAME,BLANK,L

      CALL LOCATE(PAGE,ROW,COLUMN)

      DO 10 I = 1,LENGTH
         CALL WRCHAT(PAGE,BUF(I),WRATT,1)
10    CONTINUE

      RETURN
      END
```

Appendix C.

```
      THIS SECTIONS CONTAINS THE PROGRAMS IN THE DISPLAY PROGRAMS
      CATEGORY WHICH INCLUDE THE FOLLOWING.

      (1) MAIN

      (2) DEPUTY1

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

C     (1) MAIN  -  PROGRAM  WHICH READS THE OBJECT FILE

C     GENERATED BY THE COMPILER AND DRIVES THE GRAPHICS ROUTINES TO
C     DISPLAY THE SIMULATION ON THE GRAPHICS MONITOR.


C  VARIABLES USED IN THIS PROGRAM

C  AGG(I)       =  Array containing aggregate number
C  CNTVAL(I)    =  Array containing counter value
C  CONS(I)      =  Array containing constant number
C  CONVAL(I)    =  Array containing constant value
C  COUNT(I)     =  Array containing counter number
C  IONUM(I)     =  Array containing DI/DO value
C  LAST         =  Last statement number to be executed
C  LASUB        =  Last statement number of subroutine
C  NEXT         =  Next statement number to be executed
C  NREC         =  Number of records in constants file
C  NUM1         =  Index for constants
C  NUM2         =  Index for points
C  NUM3         =  Index for counters
C  NUM4         =  Index for aggregates
C  NUM5         =  Index for parameters
C  NUM6         =  Index for pallets
C  PALLET(I)    =  Array containing pallet number
C  PALPTS(I)    =  Array containing number of points in pallet
C  PAR(I)       =  Array containing parameter number
C  PART1        =  First statement number to be executed (partial execution)
C  PART2        =  Last statement number to be executed (partial execution)
C  PARTNO(I)    =  Array containing current part number of pallet
C  PNTVAL(I)    =  Array containing point number
C  RAGG         =  R coordinate of aggregate
C  RPAL         =  R coordinate of pallet point
C  RPAR         =  R coordinate of parameter
C  RPNT         =  R coordinate of point
C  RVAL         =  Current R coordinate of robot arm
C  SS           =  Index specifying whether single-stepping or not
C  STAT         =  Array containing line of AML/E program
C  XAGG         =  X coordinate of aggregate
```

```
C  XPAL          =  X coordinate of pallet point
C  XPAR          =  X coordinate of parameter
C  XPNT          =  X coordinate of point
C  XVAL          =  Current X coordinate of robot arm
C  YAGG          =  Y coordinate of aggregate
C  YPAL          =  Y coordinate of pallet point
C  YPAR          =  Y coordinate of parameter
C  YPNT          =  Y coordinate of point
C  YVAL          =  Current Y coordinate of robot arm
C  ZAGG          =  Z coordinate of aggregate
C  ZPAL          =  Z coordinate of pallet point
C  ZPAR          =  Z coordinate of parameter
C  ZPNT          =  Z coordinate of point
C  ZVAL          =  Current Z coordinate of robot arm

C      THE MAIN PROGRAM OPEN ALL FILES AND CALLS SUBROUTINE SYS
C      TO START INTERACTIVE SIMULATION.

C      SUBROUTINES CALLED BY THIS PROGRAM ARE :
C              CLS,CSROFF,FINI,INKEY,SYS,TTOUT.

       INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
       INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100),VAL5
       INTEGER SS,SLPAY,PART1,PART2,VAL1,VAL2,RETVAL,BASC
       INTEGER * 2 LL,KYCOD1,KYCOD2
       INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
       INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
       INTEGER P29,P30,P31,P32
       REAL THONI,THTWI,ZIN,RIN,CODE
       CHARACTER * 3 AEXT,ANS*1
       CHARACTER * 20 NAME,AFILE
       CHARACTER * 24 NAMCON,NAMOUT,NAMVAR,NAMOBJ,NAMTAB
       CHARACTER * 24 NA2CON,NA2OUT,NA2VAR,NA2OBJ,NA2TAB
       CHARACTER * 24 NAMCN2,NA3CON,NA2AML,NAMAML,CELNAM*10

       DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
       DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
       DIMENSION XPNT(100),YPNT(100),ZPNT(100),RPNT(100)
       DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
       DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)

       COMMON/A/COUNT,CNTVAL,PALLET,PALPTS,PARTNO
       COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
       COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
       COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
       COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
       COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
       COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
       COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
       COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
       COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
       COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
       COMMON/R/ P29,P30,P31,P32

C      THE USER'S AML/E PROGRAM NAME IS READ FROM THE FILE IRPS00.
C      THIS FILE IS CREATED BY THE PSEUDO-COMPILER.
C      THE PROGRAM THEN CONCATENATES THE EXTENSIONS FOR THE OTHER
C      FILES CREATED BY THE COMPILER, AND OPENS THEM UP.

       OPEN (9,FILE='IRPS00')
       READ (9,100) LL,NAME
100    FORMAT (I2,1X,A20)
       CLOSE (9,STATUS='DELETE')

       OPEN (24,FILE='WORK',STATUS='OLD',ERR=101)
       READ (24,160) CELNAM
160    FORMAT (3X,A10)
```

```
      CLOSE (24)
101   CONTINUE

      XMIN  =  -800.0
      XMAX  =   800.0
      YMIN  =  -650.0
      YMAX  =   950.0
      SXMIN =  -420.0
      SXMAX =   684.0
      SYMIN =  -420.0
      SYMAX =   684.0
      BASC  =   80
      P1    =   0
      P2    =   0
      P3    =   0
      P4    =   0
      P5    =   0
      P6    =   0
      P7    =   0
      P8    =   0
      P9    =   0
      P10   =   0
      P11   =   0
      P12   =   0
      P13   =   0
      P14   =   0
      P15   =   0
      P16   =   0
      P17   =   0
      P18   =   0
      P19   =   0
      P20   =   0
      P21   =   0
      P22   =   0
      P23   =   0
      P24   =   0
      P25   =   0
      P26   =   0
      P27   =   0
      P28   =   0
      P29   =   0
      P30   =   0
      P31   =   0
      P32   =   0


      OPEN (22,FILE='INFO',STATUS='OLD',ERR=103)
      READ(22,250)BASC,XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
 250  FORMAT(1X,I2,1X,F8.2,1X,F8.2,1X,F8.2,1X,F8.2,1X,F8.2,
     *1X,F8.2,1X,F8.2,1X,F8.2)
      CLOSE(22)

C     CONCATENATE THE FILE NAME WITH THE EXTENSION .AML

103   AFILE = NAME(22-LL:20)
      AEXT = 'AML'
      CALL CONCA(AFILE,AEXT)
      NAMAML(22-LL:24) = AFILE

C     CONCATENATE THE FILE NAME WITH THE EXTENSION .CN1

      AFILE = NAME(22-LL:20)
      AEXT = 'CN1'
      CALL CONCA(AFILE,AEXT)
      NAMCN2(22-LL:24) = AFILE

C     CONCATENATE THE FILE NAME WITH THE EXTENSION .OUT
```

```
      AFILE = NAME(22-LL:20)
      AEXT = 'OUT'
      CALL CONCA(AFILE,AEXT)
      NAMOUT(22-LL:24) = AFILE

C     CONCATENATE THE FILE NAME WITH THE EXTENSION .VAR

      AFILE = NAME(22-LL:20)
      AEXT = 'VAR'
      CALL CONCA(AFILE,AEXT)
      NAMVAR(22-LL:24) = AFILE

C     CONCATENATE THE FILE NAME WITH THE EXTENSION .CON

      AFILE = NAME(22-LL:20)
      AEXT = 'CON'
      CALL CONCA(AFILE,AEXT)
      NAMCON(22-LL:24) = AFILE

C     CONCATENATE THE FILE NAME WITH THE EXTENSION .OBJ

      AFILE = NAME(22-LL:20)
      AEXT = 'OBJ'
      CALL CONCA(AFILE,AEXT)
      NAMOBJ(22-LL:24) = AFILE

C     CONCATENATE THE FILE NAME WITH THE EXTENSION .TAB

      AFILE = NAME(22-LL:20)
      AEXT = 'TAB'
      CALL CONCA(AFILE,AEXT)
      NAMTAB(22-LL:24) = AFILE

      J = 0

      DO 400 JJ = 22-LL,24
         J = J + 1
         NA2AML(J:J) = NAMAML(JJ:JJ)
         NA3CON(J:J) = NAMCN2(JJ:JJ)
         NA2OUT(J:J) = NAMOUT(JJ:JJ)
         NA2VAR(J:J) = NAMVAR(JJ:JJ)
         NA2CON(J:J) = NAMCON(JJ:JJ)
         NA2OBJ(J:J) = NAMOBJ(JJ:JJ)
         NA2TAB(J:J) = NAMTAB(JJ:JJ)
400   CONTINUE

C     THE CONSTANTS FILE .CN1, IS READ AND CONVERTED INTO A DIRECT ACCESS
C     FILE .CON.

      OPEN (1,FILE=NA3CON(1:LL+3))
      OPEN (2,FILE=NA2CON(1:LL+3),STATUS='NEW')
      NREC = 0

10    READ (1,*,END=52) I1,I2,I3,I4,T
      WRITE(2,51) I1,I2,I3,I4,T
      NREC = NREC + 1
      GOTO 10

51    FORMAT(1X,I3,2X,I3,2X,I4,2X,I3,2X,F8.2,50(' '))
52    CLOSE (1,STATUS='DELETE')
      CLOSE (2)

      OPEN (2,FILE=NA2OUT(1:LL+3),ACCESS='DIRECT',FORM='FORMATTED',
     *RECL=80)
      OPEN (3,FILE=NA2VAR(1:LL+3))
      OPEN (4,FILE=NA2CON(1:LL+3),ACCESS='DIRECT',FORM='FORMATTED',
```

```
     *RECL=80)
      OPEN (7,FILE=NA2OBJ(1:LL+3),ACCESS='DIRECT',FORM='FORMATTED',
     *RECL=80)
      OPEN (8,FILE=NA2TAB(1:LL+3),ACCESS='DIRECT',FORM='FORMATTED',
     *RECL=80)

      CALL SYS(NA2AML,LL)

C     AFTER COMPLETION CONTROL IS BACK TO THE TEXT SCREEN.
      CALL DSWAP(0)
      CALL CSROFF
      CALL TTOUT(0,24,1,'Simulation Completed...',22,22,28)
      CALL TTOUT(0,24,24,'Hit any key to return',21,21,31)
      CALL INKEY(KYCOD1,KYCOD2)

      CALL FINI

      CALL CLS(0)
      CALL CSRON
      STOP
      END

C     SUBROUTINE FINI

C     THIS SUBROUTINE DOES ALL THE HOUSEKEEPING BY CLOSING ALL FILES
C     CREATED BY THIS PROGRAM BEFORE EXECUTION TERMINATES.

C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C             CLRBUF,CLS,CSRON.

      SUBROUTINE FINI

      INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
      INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100),VAL5
      INTEGER SS,SLPAY,PART1,PART2,VAL1,VAL2,RETVAL,BASC
      INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      INTEGER P29,P30,P31,P32
      REAL THONI,THTWI,ZIN,RIN,CODE
      CHARACTER *10 CELNAM

      DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
      DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
      DIMENSION XPNT(100),YPNT(100),ZPNT(100),RPNT(100)
      DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
      DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)

      COMMON/A/COUNT,CNTVAL,PALLET,PALPTS,PARTNO
      COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
      COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
      COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
      COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
      COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
      COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

      CALL CLS(0)
      CALL CLRBUF
      CALL CSRON

      CLOSE (2)
      CLOSE (3,STATUS='DELETE')
      CLOSE (4,STATUS='DELETE')
```

```
      CLOSE (7,STATUS='DELETE')
      CLOSE (8,STATUS='DELETE')

      STOP
      END

C     SUBROUTINE SYS

C     THIS SUBROUTINE READS THE FILES GENERATED BY THE COMPILER
C     AND INTERACTS WITH THE USER FOR SIMULATION OPTIONS.

C     PARAMETER DEFINITION:

C          NA2AML    -       USER'S AML/E PROGRAM NAME
C          LL        -       LENGTH OF PROGRAM NAME

C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C             CHECK,CLRBUF,CLS,CNTRL,CSRON,CSROFF,COUN,EXEC,FLOW,IFKEY,INKEY,
C             LOCATE,MOT,PAL,PALL,PART,SENS,SLOW,SUB1,SUB2,TTOUT,WHERE,WRITER.

      SUBROUTINE SYS(NA2AML,LL)

      INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
      INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100),VAL5
      INTEGER SS,SLPAY,PART1,PART2,VAL1,VAL2,RETVAL,BASC
      INTEGER * 2 KYCOD1,KYCOD2,LL
      INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      INTEGER P29,P30,P31,P32
      REAL THONI,THTWI,ZIN,RIN,CODE

      CHARACTER * 80 STAT
      CHARACTER * 24 NA2AML,CELNAM*10
      CHARACTER * 1 BLANK(80)

      DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
      DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
      DIMENSION XPNT(100),YPNT(100),ZPNT(100),RPNT(100)
      DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
      DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)

      COMMON/A/COUNT,CNTVAL,PALLET,PALPTS,PARTNO
      COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
      COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
      COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
      COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
      COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
      COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

      DATA BLANK/80 * ' '/

C     INITIALIZE ALL PROGRAM VARIABLES.

      DO 1001 JJJJ = 1,80
         STAT(JJJJ:JJJJ) = BLANK(1)
1001  CONTINUE

      SS     = 0
      PART1  = 1
      PART2  = 1000
      IC     = 1
      XVAL   = 650.0
```

```
      YVAL   = 0.0
      ZVAL   = 0.0
      RVAL   = 0.0
      NUM    = 1
      NUM1   = 1
      NUM2   = 1
      NUM3   = 1
      NUM4   = 1
      NUM5   = 1
      NUM6   = 1
      VAL1   = 1
      VAL2   = 0
      VAL5   = 0
      RETVAL = 0

      DO 1 I = 1,32
         IONUM(I) = 0
1     CONTINUE

C     READ EACH LINE OF THE VARIABLES FILE, AND ASSIGN VALUES
C     READ FROM THE SYMBOL TABLE FILE, TO VARIABLE NAMES.

10    READ (3,100,END=1000) IC1,ICODE

C     THE NATURE OF THE VARIABLE IS GOT FROM THE ICODE  VALUE.
C     IF THE VARIABLE NAME IS A LABEL, ASSIGN THE STATEMENT NUMBER AS THE
C     VALUE FOR THE LABEL, AND WRITE INTO THE SYMBOL TABLE FILE.

11    IF (ICODE .EQ. 1) THEN
         CC = IC1
         WRITE (8,400,REC=NUM) NUM,CC

C     IF THE VARIABLE IS A CONSTANT, ASSIGN VALUE TO ARRAY CONVAL.

      ELSEIF (ICODE .EQ. 2) THEN

         READ (8,200,REC=NUM) CODE
         CONS(NUM1) = NUM
         CONVAL(NUM1) = CODE
         NUM1 = NUM1 + 1

C     IF THE VARIABLE IS A POINT, ASSIGN VALUES TO ARRAYS XPNT,YPNT,RPNT,
C     AND ZPNT.

      ELSEIF (ICODE .EQ. 3) THEN
         READ (8,300,REC=NUM) X,Y,Z,R
         PNTVAL(NUM2) = NUM
         XPNT(NUM2) = X
         YPNT(NUM2) = Y
         ZPNT(NUM2) = Z
         RPNT(NUM2) = R
         NUM2 = NUM2 + 1

C     IF THE VARIABLE IS A COUNTER, INITIALIZE COUNTER VALUE TO ZERO, AND
C     WRITE TO THE SYMBOL TABLE FILE.

      ELSEIF (ICODE .EQ. 4) THEN
         COUNT(NUM3) = NUM
         CNTVAL(NUM3) = 0
         CC = CNTVAL(NUM3)
         WRITE (8,400,REC=NUM) NUM,CC
         NUM3 = NUM3 + 1

C     IF THE VARIABLE IS A PALLET, CALL SUBROUTINE PAL.

      ELSEIF (ICODE .EQ. 5) THEN
         CALL PAL(NUM)
```

```
C     IF THE VARIABLE IS A SUBROUTINE NAME, FIRST CHECK FOR FORMAL PARAMETERS.
C     IF THERE ARE ANY FORMAL PARAMETERS, INITIALIZE VALUES TO ZEROES, AND
C     WRITE THE VARIABLE NUMBER OF THE FORMAL PARAMETER TO THE OBJECT FILE.

      ELSEIF (ICODE .EQ. 6) THEN
         NN = 0
12       NUM = NUM + 1
         NN = NN + 1
         READ (3,100,END=1000) IC1,ICODE

         IF (ICODE .EQ. 7) THEN
            PAR(NUM5) = NUM
            XPAR(NUM5) = 0.0
            YPAR(NUM5) = 0.0
            ZPAR(NUM5) = 0.0
            RPAR(NUM5) = 0.0
            NUM5 = NUM5 + 1

            READ (7,700,REC=IC1) L1,L2,L3,L4,L5,L6,(L(I),I=1,5)
            L(NN) = NUM
            WRITE (7,700,REC=IC1) L1,L2,L3,L4,L5,L6,(L(I),I=1,5)
            GOTO 12
         ENDIF

         GOTO 11

C     IF THE VARIABLE IS AN AGGREGATE, READ IN THE FOUR VALUES, AND
C     ASSIGN THEM TO THE ARRAYS XAGG,YAGG,ZAGG,RAGG.

      ELSEIF (ICODE .EQ. 8) THEN
         READ (8,300,REC=NUM) X,Y,Z,R
         AGG(NUM4) = NUM
         XAGG(NUM4) = X
         YAGG(NUM4) = Y
         ZAGG(NUM4) = Z
         RAGG(NUM4) = R
         NUM4 = NUM4 + 1

      ENDIF

C     INCREMENT THE VARIABLE NUMBER AND READ THE NEXT ONE.

      NUM = NUM + 1
      GOTO 10

1000  REWIND (3)

C     ONCE ALL VARIABLES HAVE BEEN READ, DISPLAY MESSAGE TO USER.

      CALL CLS(31)
      CALL TTOUT(0,10,15,'The Robot will now be redrawn',29,29,28)
      CALL TTOUT(0,12,19,'in the HOME position.',21,21,28)
      CALL TTOUT(0,24,1,'Hit any key to proceed....',26,26,28)
      CALL CSROFF
      CALL INKEY(KYCOD1,KYCOD2)
      CALL CLS(31)

C     CODE TO RETURN TO HOME POSITION
      CODE = 4.0

C     CALL THE GRAPHICS DISPLAY PROGRAM

      CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)

C     PROMPT THE USER FOR SINGLE-STEPPING OR CONTINUOUS OPERATION.
C     IF THE USER OPTS FOR SINGLE-STEPPING SET SS TO 1.
```

```
      CALL DSMAP(0)
      CALL CLS(31)
      CALL CSRON
      CALL TTOUT(0,10,1,'Do you want to single step',26,26,31)
      CALL TTOUT(0,11,1,'through the program (Y/N) ?',27,27,31)
      CALL LOCATE(0,11,29)
      CALL INKEY(KYCOD1,KYCOD2)
      IF (KYCOD1 .EQ. 89 .OR. KYCOD1 .EQ. 121) SS = 1

C     READ THE FIRST RECORD IN THE OBJECT FILE FOR THE FIRST
C     EXECUTABLE STATEMENT NUMBER AND THE STATEMENT NUMBER OF THE
C     LAST "END"STATEMENT.

      READ (7,500,REC=1) K1,K2

C     CALL SUBROUTINE PART WHICH ASKS THE USER IF THEY WOULD LIKE
C     TO EXECUTE THE COMPLETE PROGRAM, PART OF THE PROGRAM OR
C     EXECUTE A SUBROUTINE.

      CALL PART

C     BASED ON THE USER INPUTS, THE VARIABLES LAST AND NEXT ARE SET.
C     NEXT CONTAINS THE NEXT STATEMENT NUMBER TO BE EXECUTED, AND LAST
C     CONTAINS THE LAST STATEMENT NUMBER TO BE EXECUTED.

      IF (PART1 .EQ. 1 .AND. PART2 .EQ. 1000) THEN
         LAST = K1
         NEXT = K2
      ELSE
         LAST = PART2
         NEXT = PART1
      ENDIF

C     DISPLAY NAME OF PROGRAM BEING EXECUTED, AND THE USE OF FUNCTION
C     KEYS WHICH COULD BE PRESSED TO EITHER ABORT PROGRAM EXECUTION,

      CALL CLS(31)
      CALL TTOUT(0,24,1,'Simulating Program....',22,22,28)
      CALL TTOUT(0,24,27,NA2AML(1:LL+3),3+LL,3+LL,26)

C     THIS MESSAGE IS DISPLAYED ONLY IF CONTINUOUS IS IN EFFECT.

      IF (SS .EQ. 0) THEN
         CALL TTOUT(0,21,1,'Press F1 key to abort execution.',32,
     *   32,26)
      ENDIF


C     IF THE NEXT STATEMENT TO BE EXECUTED IS THE LAST ONE, STOP.
C     OTHERWISE, READ THE NEXT STATEMENT FROM THE OBJECT FILE, AND
C     THE LINE FROM THE LISTING FILE, AND DISPLAY ON THE SCREEN.

20    IF (NEXT .EQ. LAST) GOTO 2000
      READ (7,600,REC=NEXT) K54,K1,K2

      CALL TTOUT(0,11,1,STAT,80,80,26)

      READ (2,22,REC=K54) STAT
22    FORMAT(A80)

      CALL TTOUT(0,12,1,STAT,80,80,31)

C     DISPLAY FUNCTION KEY USAGE IF SINGLE-STEPPING IS IN EFFECT.

      IF (SS .EQ. 1) THEN
```

```
         CALL CSRON
         CALL TTOUT(0,1,5,'Press <Enter> key to continue single step.',
     *   42,42,26)
         CALL TTOUT(0,3,5,'Press F1 key to abort execution.',32,
     *   32,26)
         CALL TTOUT(0,5,5,'Press F2 key for continuous execution.',38,
     *   38,26)


C     WAIT FOR USER INPUT. IF USER STRIKES "ENTER" KEY, CONTINUE
C     WITH SINGLE-STEPPING. IF USER HITS F1, STOP PROGRAM EXECUTION,
C     AND IF USER STRIKES F2, EXECUTE PROGRAM CONTINUOUSLY.

         CALL LOCATE(0,5,45)
21       CALL INKEY(KYCOD1,KYCOD2)

         IF (KYCOD1 .EQ. 13) THEN
            SS = 1
         ELSEIF (KYCOD1 .EQ. 0 .AND. KYCOD2 .EQ. 59) THEN
            CALL FINI
         ELSEIF (KYCOD1 .EQ. 0 .AND. KYCOD2 .EQ. 60) THEN
            SS = 0
            CALL TTOUT(0,21,1,'Press F1 key to abort execution.',32,
     *      32,26)
            ENDIF
      ENDIF


C     READ USER INPUT AFTER EVERY STATEMENT IS EXECUTED. IF THE
C     USER STRIKES F1, STOP PROGRAM EXECUTION.

      CALL IFKEY(KYCOD1,KYCOD2)
      IF (KYCOD1 .EQ. 0 .AND. KYCOD2 .EQ. 59 ) CALL FINI
      CALL CLRBUF

C     CHECK THE COMMAND TYPE. K1 IS 0 FOR LABELS, 1 FOR MOTION
C     COMMANDS, 2 FOR SENSOR COMMANDS, 3 FOR FLOW OF CONTROL
C     COMMANDS, 4 FOR COUNTER COMMANDS, 5 FOR PALLET COMMANDS,
C     6 FOR SUBROUTINE COMMANDS, 7 FOR CALLS TO SUBROUTINES,
C     AND 8 FOR THE COMMAND "WHERE". BASED ON THE COMMAND TYPE
C     CALL THE APPROPRIATE SUBROUTINE.

      IF (K1 .EQ. 0) THEN
         NEXT = NEXT + 1
      ELSEIF (K1. EQ. 1) THEN
         CALL MOT(K2)
      ELSEIF (K1. EQ. 2) THEN
         CALL SENS(K2)
      ELSEIF (K1. EQ. 3) THEN
         CALL FLOW(K2)
      ELSEIF (K1. EQ. 4) THEN
         CALL COUN(K2)
      ELSEIF (K1 .EQ. 5) THEN
         CALL PALL(K2)
      ELSEIF (K1. EQ. 6) THEN
         CALL SUB1(K2)
      ELSEIF (K1. EQ. 7) THEN
         CALL SUB2(K2)
      ELSEIF (K1. EQ. 8) THEN
         CALL WHERE
      ENDIF

C     IF THE LAST EXECUTED COMMAND WAS ONE THAT CHANGED THE GRAPHICS
C     DISPLAY GIVE THE USER THE OPTION OF REDRAWING THE SCREEN.

      CALL DSWAP(0)
        IF (K1 .EQ. 1) THEN
```

```
          IF ((K2. EQ. 2) .OR. (K2.EQ. 3) .OR. (K2 .EQ. 4)
     *.OR. (K2 .EQ. 5)) THEN
          GOTO 124
          ENDIF
        ELSEIF (K1 .EQ. 5)   THEN
              IF (K2. EQ. 1) THEN
              GOTO 124
              ENDIF
        ENDIF
       GOTO 125

 124   CALL CLS(31)
       CALL TTOUT(0,11,1,'DO YOU WANT TO REDRAW THE SCREEN(Y/N)',
      *37,37,31)
       CALL TCSKEY (ISCAN,IASC,NCHAR)
       IF (IASC .EQ. ICHAR('Y')) THEN
       CODE  = 8.0
       CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)
       CALL DSWAP(0)
       ENDIF

C      GO READ THE NEXT STATEMENT TO BE EXECUTED.

125    GOTO 20

123    FORMAT(A1)
100    FORMAT(11X,I4,7X,I1)
200    FORMAT(6X,F8.2)
300    FORMAT(4X,4(2X,F8.2))
400    FORMAT(1X,I3,2X,F8.2)
500    FORMAT(18X,I4,1X,I4)
600    FORMAT(6X,I4,1X,I1,1X,I4)
700    FORMAT(1X,I4,1X,I4,1X,I1,8(1X,I4))

2000   RETURN

       END

C      SUBROUTINE PAL

C      THIS SUBROUTINE COMPUTES THE POINTS OF EVERY PALLET POSITION.

C      PARAMETER DEFINITION:

C           NUM       -       VARIABLE NUMBER OF PALLET

       SUBROUTINE PAL(NUM)

       INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
       INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100),VAL5
       INTEGER SS,SLPAY,PART1,PART2,VAL1,VAL2,RETVAL,BASC
       INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
       INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
       INTEGER P29,P30,P31,P32
       REAL THONI,THTNI,ZIN,RIN,CODE
       CHARACTER *10 CELNAM

       DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
       DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
       DIMENSION XPNT(100),YPNT(100),ZPNT(100),RPNT(100)
       DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
       DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)

       COMMON/A/COUNT,CNTVAL,PALLET,PALPTS,PARTNO
       COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
       COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
       COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
```

```
      COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
      COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
      COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

C     READ THE VARIABLE / CONSTANT NUMBERS OF PARAMETERS

      READ (8,100,REC=NUM) L1,L2,L3,L4,L5,L6,L7,L8,L9,L10

C     READ THE VALUES OF THE THREE CORNER POINTS FROM THE SYMBOL
C     TABLE FILE.

      READ (8,200,REC=L2) X1,Y1,Z1,R1
      READ (8,200,REC=L4) X2,Y2,Z2,R2
      READ (8,200,REC=L6) X3,Y3,Z3,R3
      PALLET(NUM6) = NUM

C     READ IN THE VALUE OF PART PER ROW, IPPR, FROM THE SYMBOL TABLE
C     FILE OR CONSTANTS FILE.

      IF (L7 .EQ. 1) THEN
         READ (8,200,REC=L8) CC
         IPPR = CC
      ELSEIF (L7 .EQ. 2) THEN
         READ (4,300,REC=L8) CC
         IPPR = CC
      ENDIF

C     READ IN THE VALUE OF NUMBER OF PALLET POINTS, PALPTS, FROM THE SYMBOL
C     TABLE FILE OR CONSTANTS FILE.

      IF (L9 .EQ. 1) THEN
         READ (8,200,REC=L10) CC
         PALPTS(NUM6) = CC
      ELSEIF (L9 .EQ. 2) THEN
         READ (4,300,REC=L10) CC
         PALPTS(NUM6) = CC
      ENDIF

C     THIS SECTION COMPUTES THE COORDINATES OF EACH PALLET POINT, BASED
C     ON THE COORDINATES OF THE CORNER POINTS, NUMBER OF PALLET POINTS,
C     AND THE PARTS PER ROW.

      XTOTX = X2 - X1
      XTOTY = X3 - X2
      YTOTX = Y2 - Y1
      YTOTY = Y3 - Y2
      IPPC = PALPTS(NUM6) / IPPR

      IF (IPPR .NE. 1) THEN
         XINCX = XTOTX / (IPPR-1)
         YINCX = YTOTX / (IPPR-1)
      ELSE
         XINCX = XTOTX
         YINCX = YTOTX
      ENDIF

      IF (IPPC .NE. 1) THEN
         XINCY = XTOTY / (IPPC-1)
         YINCY = YTOTY / (IPPC-1)
      ELSE
         XINCY = XTOTY
         YINCY = YTOTY
```

```
      ENDIF
      XPAL(NUM6,1) = X1
      YPAL(NUM6,1) = Y1
      K = 1

      DO 10 I = 1,PALPTS(NUM6),IPPR
         XPAL(NUM6,I) = XPAL(NUM6,1) + (XINCY * (K - 1))
         YPAL(NUM6,I) = YPAL(NUM6,1) + (YINCY * (K - 1))

         DO 20 J = 1,IPPR
            XPAL(NUM6,I+J-1) = XPAL(NUM6,I) + ((J - 1) * XINCX)
            YPAL(NUM6,I+J-1) = YPAL(NUM6,I) + ((J - 1) * YINCX)
20       CONTINUE

         K = K + 1
10    CONTINUE

C     THE Z VALUE FOR ALL PALLET POINTS ARE SET TO ZERO.
C     THE R VALUE FOR ALL PALLET POINTS IS SET TO THE R VALUE
C     OF THE LOWER LEFT CORNER POINT.

      DO 30 I = 1,PALPTS(NUM6)
         ZPAL(NUM6,I) = 0.0
         RPAL(NUM6,I) = R1
30    CONTINUE

C     THE INITIAL PART NUMBER FOR THE PALLET IS SET TO ONE.

      PARTNO(NUM6) = 1
      NUM6 = NUM6 + 1

100   FORMAT(14X,10(1X,I4))
200   FORMAT(4X,4(2X,F8.2))
300   FORMAT(22X,F8.2)

      RETURN
      END

C     SUBROUTINE MOT

C     THIS SUBROUTINE HANDLES ALL THE MOTION COMMANDS.

C     PARAMETER DEFINITION:

C          K2        -       SUBTYPE OF COMMAND
C                              1 - DELAY
C                              2 - DPMOVE
C                              3 - GRASP
C                              4 - PMOVE
C                              5 - RELEASE
C                              6 - ZMOVE
C                              7 - LINEAR
C                              8 - PAYLOAD
C                              9 - ZONE

C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C             CHECK,ERROR,USER,WRITEP,WRITER.

      SUBROUTINE MOT(K2)

      INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
      INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100),VAL5
      INTEGER SS,SLPAY,PART1,PART2,STATE,VAL1,VAL2,RETVAL,BASC
      INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      INTEGER P29,P30,P31,P32
      REAL THONI,THTWI,ZIN,RIN,CODE
```

```
      CHARACTER *10 CELNAM

      DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
      DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
      DIMENSION XPNT(100),YPNT(100),ZPNT(100),RPNT(100)
      DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
      DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)

      COMMON/A/COUNT,CNTVAL,PALLET,PALPTS,PARTNO
      COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
      COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
      COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
      COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
      COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
      COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

C     DELAY

C     READ THE OBJECT FILE FOR THE COMMAND PARAMETER. L1 IS 1 IF THE
C     PARAMETER IS A VARIABLE NAME, AND 2 IF IT IS A NUMBER. L2 IS THE
C     VARIABLE / CONSTANT NUMBER.
C     L1 AND L2 REPRESENT THE TIME VALUE.

      IF (K2 .EQ. 1) THEN
         READ (7,100,REC=NEXT) L1,L2

C     THE VARIABLE NAME IS COMPARED AGAINST THE NAMES OF CONSTANTS, COUNTERS
C     AND PARAMETERS IN THE PROGRAM, AND ONCE THE MATCH HAS BEEN FOUND, ITS
C     VALUE IS ASSIGNED TO TIME.

         IF (L1 .EQ. 1) THEN

            DO 10 I = 1,NUM1-1
               IF (L2 .EQ. CONS(I)) THEN
                  TIME = CONVAL(I)
                  GOTO 11
               ENDIF

10          CONTINUE

            DO 12 I = 1,NUM3-1

               IF (L2 .EQ. COUNT(I)) THEN
                  TIME = CNTVAL(I)
                  GOTO 11
               ENDIF

12          CONTINUE

            DO 19 I = 1,NUM5-1

               IF (L2 .EQ. PAR(I)) THEN
                  READ (8,500,REC=L2) TIME
                  GOTO 11
               ENDIF

19          CONTINUE

C     IF THE VALUE IS A NUMBER, ITS VALUE IS READ FROM THE CONSTANTS
C     FILE, AND ASSIGNED TO TIME.

         ELSEIF (L1 .EQ. 2) THEN
```

```
            READ (4,200,REC=L2) TIME
         ENDIF

C     THE VALUE IS CHECKED TO BE WITHIN VALID LIMITS, AND IF NOT
C     AN ERROR MESSAGE IS DISPLAYED.

11       IF (TIME .LT. 0.0 .OR. TIME .GT. 25.5) CALL USER(1)

C       DELAY

        Z=0.0
        DO 20 I = 1,10000
        Z= (I+1)*2
20      CONTINUE


C     DPMOVE

C     READ THE OBJECT FILE FOR THE COMMAND PARAMETERS. L1 IS 1 IF THE
C     PARAMETER IS A VARIABLE NAME, AND 2 IF IT IS A NUMBER. L2 IS THE
C     VARIABLE / CONSTANT NUMBER.
C     L1 AND L2 REPRESENT THE AGGREGATE VALUE.

      ELSEIF (K2 .EQ. 2) THEN
         READ (7,100,REC=NEXT) L1,L2

C     THE VARIABLE NAME IS COMPARED AGAINST THE NAMES OF AGGREGATES
C     OR PARAMETERS, AND ONCE THE MATCH HAS BEEN FOUND, ITS
C     VALUES ARE ADDED TO XVAL,YVAL,ZVAL, AND RVAL.

         IF (L1 .EQ. 1) THEN

            DO 79 I = 1,NUM4-1

               IF (L2 .EQ. AGG(I)) THEN
                  XVAL = XVAL + XAGG(I)
                  YVAL = YVAL + YAGG(I)
                  ZVAL = ZVAL + ZAGG(I)
                  RVAL = RVAL + RAGG(I)
                  GOTO 21
               ENDIF

79          CONTINUE

               DO 29 I = 1,NUM5-1

                  IF (L2 .EQ. PAR(I)) THEN
                     READ (8,500,REC=L2) XAGG(I),YAGG(I),ZAGG(I),RAGG(I)
                     XVAL = XVAL + XAGG(I)
                     YVAL = YVAL + YAGG(I)
                     ZVAL = ZVAL + ZAGG(I)
                     RVAL = RVAL + RAGG(I)
                     GOTO 21
                  ENDIF

29             CONTINUE

C     IF THE VALUE IS A NUMBER, THAT VALUE AND THE NEXT THREE VALUES
C     ARE READ FROM THE OBJECT FILE, AND ADDED TO XVAL,YVAL,ZVAL, AND
C     RVAL.

         ELSEIF (L1 .EQ. 2) THEN
            READ (4,200,REC=L2) X
            READ (4,200,REC=L2+1) Y
            READ (4,200,REC=L2+2) Z
            READ (4,200,REC=L2+3) R
            XVAL = XVAL + X
```

```
            YVAL = YVAL + Y
            ZVAL = ZVAL + Z
            RVAL = RVAL + R
            GOTO 21

C     IF THE VALUES ARE SPECIFIED AS A SET OF COUNTERS, THE FOUR COUNTER
C     VALUES ARE READ FROM THE SYMBOL TABLE FILE AND ADDED TO THE
C     VARIABLES XVAL,YVAL,ZVAL, AND RVAL.

         ELSEIF (L1 .EQ. 3) THEN
            READ (7,600,REC=NEXT) L3,L4,L5
            READ (8,500,REC=L2) X
            READ (8,500,REC=L3) Y
            READ (8,500,REC=L4) Z
            READ (8,500,REC=L5) R
            XVAL = XVAL + X
            YVAL = YVAL + Y
            ZVAL = ZVAL + Z
            RVAL = RVAL + R
         ENDIF

C     THE R AND Z VALUES ARE CHECKED TO BE WITHIN VALID LIMITS,
C     AND IF NOT, AN ERROR MESSAGE IS DISPLAYED.

21       CODE = 1.0

         IF (RVAL .LT. -180.0 .OR. RVAL .GT. 180.0) CALL USER(2)
         IF (ZVAL .LT. -250.0 .OR. ZVAL .GT. 0.0) CALL USER(3)


C     THE COORDINATES OF THE POINT ARE THEN PASSED
C     TO THE GRAPHICS DRIVERS.


C     TO MOVE

      CODE = 1.0

      ZVAL = ZVAL + 250.0

C     CALL THE GRAPHICS DISPLAY PROGRAM

      CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)

C     GRASP

C     THE DO PORT IS SET TO 2, AND ITS VALUE TO 1.

      ELSEIF (K2 .EQ. 3) THEN

C     SETTING DO PORT 2 TO VALUE 1 TO INDICATE GRASP

      P18 = 1
      CODE = 3.0
      VAL5 = 1

C     CALL THE GRAPHICS DISPLAY PROGRAM

      CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)


C     PMOVE

C     READ THE OBJECT FILE FOR THE COMMAND PARAMETERS. L1 IS 1 IF THE
C     PARAMETER IS A VARIABLE NAME, AND 2 IF IT IS A NUMBER. L2 IS THE
C     VARIABLE / CONSTANT NUMBER.
C     L1 AND L2 REPRESENT THE POINT VALUE.
```

```
      ELSEIF (K2 .EQ. 4) THEN
         READ (7,100,REC=NEXT) L1,L2

C     THE VARIABLE NAME IS COMPARED AGAINST THE NAMES OF POINTS
C     AND PARAMETERS IN THE PROGRAM, AND ONCE THE MATCH HAS BEEN FOUND, ITS
C     VALUES     ASSIGNED TO XVAL,YVAL,ZVAL, AND RVAL.

         IF (L1 .EQ. 1) THEN

            DO 40 I = 1,NUM2-1

              IF (L2 .EQ. PNTVAL(I)) THEN


                 XVAL = XPNT(I)
                 YVAL = YPNT(I)
                 ZVAL = ZPNT(I)
                 RVAL = RPNT(I)
                 GOTO 41
               ENDIF

40          CONTINUE

               DO 49 I = 1,NUM5-1

                  IF (L2 .EQ. PAR(I)) THEN
                     READ (8,500,REC=L2) XVAL,YVAL,ZVAL,RVAL
                     GOTO 41
                  ENDIF

49             CONTINUE

C     IF THE VALUE IS A SET OF NUMBERS, THEIR VALUES ARE READ FROM THE
C     CONSTANTS FILE, AND ASSIGNED TO XVAL,YVAL,ZVAL, AND RVAL.

         ELSEIF (L1 .EQ. 2) THEN
            READ (4,200,REC=L2) XVAL
            READ (4,200,REC=L2+1) YVAL
            READ (4,200,REC=L2+2) ZVAL
            READ (4,200,REC=L2+3) RVAL
            GOTO 41

C     IF THE VALUES ARE SPECIFIED AS A SET OF COUNTERS, THE FOUR COUNTER
C     VALUES ARE READ FROM THE SYMBOL TABLE FILE AND ASSIGNED TO THE
C     VARIABLES XVAL,YVAL,ZVAL, AND RVAL.

         ELSEIF (L1 .EQ. 3) THEN
            READ (7,600,REC=NEXT) L3,L4,L5
            READ (8,500,REC=L2) XVAL
            READ (8,500,REC=L3) YVAL
            READ (8,500,REC=L4) ZVAL
            READ (8,500,REC=L5) RVAL
         ENDIF

41       CODE = 1.0


C     THE R AND Z VALUES ARE CHECKED TO BE WITHIN VALID LIMITS,
C     AND IF NOT, AN ERROR MESSAGE IS DISPLAYED.

         IF (RVAL .LT. -180.0 .OR. RVAL .GT. 180.0) CALL USER(2)
         IF (ZVAL .LT. -250.0 .OR. ZVAL .GT. 0.0) CALL USER(3)

C     THE COORDINATES OF THE POINT ARE THEN PASSED
C     TO THE GRAPHICS DRIVERS.
```

```
      CODE = 1.0
      ZVAL = ZVAL + 250.0

C     CALL THE GRAPHICS DISPLAY PROGRAM

      CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)


C     RELEASE

C     THE DO PORT IS SET TO 2, AND ITS VALUE TO 0.

      ELSEIF (K2 .EQ. 5) THEN

      P18 = 0
      CODE = 3.0
      VAL5 = 0

C     CALL THE GRAPHICS DISPLAY PROGRAM

      CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)

C
C     ZMOVE
C

C     READ THE OBJECT FILE FOR THE COMMAND PARAMETERS. L1 IS 1 IF THE
C     PARAMETER IS A VARIABLE NAME, AND 2 IF IT IS A NUMBER. L2 IS THE
C     VARIABLE / CONSTANT NUMBER.
C     L1 AND L2 REPRESENT THE Z VALUE.

      ELSEIF (K2 .EQ. 6) THEN
         READ (7,100,REC=NEXT) L1,L2

C     THE VARIABLE NAME IS COMPARED AGAINST THE NAMES OF CONSTANTS, COUNTERS
C     AND PARAMETERS IN THE PROGRAM, AND ONCE THE MATCH HAS BEEN FOUND, ITS
C     VALUE IS ASSIGNED TO ZVAL.

         IF (L1 .EQ. 1) THEN

            DO 60 I = 1,NUM1-1

               IF (L2 .EQ. CONS(I)) THEN
                  ZVAL = CONVAL(I)
                  GOTO 61
               ENDIF

60          CONTINUE

            DO 62 I = 1,NUM3-1

               IF (L2 .EQ. COUNT(I)) THEN
                  ZVAL = CNTVAL(I)
                  GOTO 61
               ENDIF

62          CONTINUE

            DO 69 I = 1,NUM5-1

               IF (L2 .EQ. PAR(I)) THEN
                  READ (8,500,REC=L2) ZVAL
                  GOTO 61
               ENDIF

69          CONTINUE
```

```
C     IF THE VALUE IS A NUMBER, ITS VALUE IS READ FROM THE CONSTANTS
C     FILE, AND ASSIGNED TO ZVAL.

         ELSEIF (L1 .EQ. 2) THEN
            READ (4,200,REC=L2) ZVAL
         ENDIF

61       CODE = 1.0


C     THE R AND Z VALUES ARE CHECKED TO BE WITHIN VALID LIMITS,
C     AND IF NOT, AN ERROR MESSAGE IS DISPLAYED.

         IF (RVAL .LT. -180.0 .OR. RVAL .GT. 180.0) CALL USER(2)
         IF (ZVAL .LT. -250.0 .OR. ZVAL .GT. 0.0) CALL USER(3)

C     THE COORDINATES OF THE POINT ARE THEN PASSED
C     TO THE GRAPHICS DRIVERS.


         CODE = 1.0
         ZVAL = ZVAL + 250.0

C     CALL THE GRAPHICS DISPLAY PROGRAM

      CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)

      ELSE
      GO TO  700

700   ENDIF

      NEXT = NEXT + 1

100    FORMAT(17X,2(1X,I4))
200    FORMAT(22X,F8.2)
300    FORMAT(2X,I1,4(2X,F8.2))
400    FORMAT(2X,I1,2X,I8)
500    FORMAT(4X,4(2X,F8.2))
600    FORMAT(27X,3(1X,I4))


      RETURN
      END



C     SUBROUTINE SENS

C     THIS SUBROUTINE HANDLES ALL THE SENSOR COMMANDS.

C     PARAMETER DEFINITION:

C          K2        -      SUBTYPE OF COMMAND
C                             1 - WAITI
C                             2 - WRITEO


C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C             CHECK,ERROR,READR,USER,WRITER.

      SUBROUTINE SENS(K2)

      INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
      INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100),VAL5
      INTEGER SS,SLPAY,PART1,PART2,VAL1,VAL2,RETVAL,BASC
```

```
      INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      INTEGER P29,P30,P31,P32,ANS
      REAL THONI,THTWI,ZIN,RIN,CODE
      CHARACTER *10 CELNAM

      DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
      DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
      DIMENSION XPNT(100),YPNT(100),ZPNT(100),RPNT(100)
      DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
      DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)

      COMMON/A/COUNT,CNTVAL,PALLET,PALPTS,PARTNO
      COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
      COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
      COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
      COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
      COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
      COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

C     WAITI

C     READ THE OBJECT FILE FOR THE COMMAND PARAMETERS. L1 IS 1 IF THE
C     PARAMETER IS A VARIABLE NAME, AND 2 IF IT IS A NUMBER. L2 IS THE
C     VARIABLE / CONSTANT NUMBER. L3 IS 1 IF THE PARAMETER IS A VARIABLE
C     NAME AND 2 IF IT IS A NUMBER. L4 IS THE VARIABLE / CONSTANT NUMBER.
C     L5 IS 1 IF THE PARAMETER IS A VARIABLE NAME, AND 2 IF IT IS A
C     NUMBER. L6 IS THE VARIABLE / CONSTANT NUMBER.
C     L7 IS THE VARIABLE NUMBER OF THE LABEL, WHICH IS OPTIONAL.
C     L1 AND L2 REPRESENT THE DI PORT, L3 AND L4 REPRESENT THE
C     DI VALUE, L5 AND L6 REPRESENT THE TIME LIMIT.

      IF (K2 .EQ. 1) THEN
         READ (7,200,REC=NEXT) L1,L2,L3,L4,L5,L6,L7
         L8 = 0

C     THE SYMBOL TABLE FILE IS READ FOR THE STATEMENT NUMBER OF THE LABEL,
C     AND THIS VALUE IS ASSIGNED TO L8. THIS HAPPENS ONLY IF THERE IS A
C     LABEL IN THE STATEMENT., OTHERWISE L8 IS SET TO ZERO.

         IF (L7 .GT. 0) THEN
            READ (8,210,REC=L7) CC
            L8 = CC
         ENDIF

C     THE VARIABLE NAME IS COMPARED AGAINST THE NAMES OF CONSTANTS, COUNTERS
C     AND PARAMETERS IN THE PROGRAM, AND ONCE THE MATCH HAS BEEN FOUND, ITS
C     VALUE IS ASSIGNED TO L2.

         IF (L1 .EQ. 1) THEN

            DO 10 I = 1,NUM1-1

               IF (L2 .EQ. CONS(I)) THEN
                  L2 = CONVAL(I)
                  GOTO 11
               ENDIF

10          CONTINUE

            DO 12 I = 1,NUM3-1
```

```
               IF (L2 .EQ. COUNT(I)) THEN
                  L2 = CNTVAL(I)
                  GOTO 11
               ENDIF

12          CONTINUE

            DO 19 I = 1,NUM5-1

               IF (L2 .EQ. PAR(I)) THEN
                  READ (8,500,REC=L2) T
                  L2 = T
                  GOTO 11
               ENDIF

19          CONTINUE

C     IF THE VALUE IS A NUMBER, ITS VALUE IS READ FROM THE CONSTANTS
C     FILE, AND ASSIGNED TO L2.

         ELSEIF (L1 .EQ. 2) THEN
           READ (4,300,REC=L2) T
           L2 = T
           GOTO 11
         ENDIF

C     THE VARIABLE NAME IS COMPARED AGAINST THE NAMES OF CONSTANTS, COUNTERS
C     AND PARAMETERS IN THE PROGRAM, AND ONCE THE MATCH HAS BEEN FOUND, ITS
C     VALUE IS ASSIGNED TO L4.

11       IF (L3 .EQ. 1) THEN

            DO 13 I = 1,NUM1-1

               IF (L4 .EQ. CONS(I)) THEN
                  L4 = CONVAL(I)
                  GOTO 14
               ENDIF

13          CONTINUE

            DO 15 I = 1,NUM3-1

               IF (L4 .EQ. COUNT(I)) THEN
                  L4 = CNTVAL(I)
                  GOTO 14
               ENDIF

15          CONTINUE

            DO 29 I = 1,NUM5-1

               IF (L2 .EQ. PAR(I)) THEN
                  READ (8,500,REC=L4) T
                  L4 = T
                  GOTO 14
               ENDIF

29          CONTINUE

C     IF THE VALUE IS A NUMBER, ITS VALUE IS READ FROM THE CONSTANTS
C     FILE, AND ASSIGNED TO L4.

         ELSEIF (L1 .EQ. 2) THEN
           READ (4,300,REC=L4) T
           L4 = T
           GOTO 14
```

```
         ENDIF

C      THE VARIABLE NAME IS COMPARED AGAINST THE NAMES OF CONSTANTS, COUNTERS
C      AND PARAMETERS IN THE PROGRAM, AND ONCE THE MATCH HAS BEEN FOUND, ITS
C      VALUE IS ASSIGNED TO THE VARIABLE TIME.

14       IF (L5 .EQ. 1) THEN

            DO 16 I = 1,NUM1-1

               IF (L6 .EQ. CONS(I)) THEN
                  TIME = CONVAL(I)
                  GOTO 17
               ENDIF

16          CONTINUE

            DO 18 I = 1,NUM3-1

               IF (L6 .EQ. COUNT(I)) THEN
                  TIME = CNTVAL(I)
                  GOTO 17
               ENDIF

18          CONTINUE

            DO 39 I = 1,NUM5-1

               IF (L6 .EQ. PAR(I)) THEN
                  READ (8,500,REC=L6) T
                  TIME = T
                  GOTO 17
               ENDIF

39          CONTINUE

C      IF THE VALUE IS A NUMBER, ITS VALUE IS READ FROM THE CONSTANTS
C      FILE, AND ASSIGNED TO TIME.

         ELSEIF (L5 .EQ. 2) THEN
           READ (4,300,REC=L6) TIME
           GOTO 17
         ENDIF

C      IN THIS SIMULATION THERE IS NO DIFFERENCE BETWEEN INDEFINITE WAIT
C      AND WAIT FOR 10 SECONDS. ONLY DISTINCTION IS BETWEEN WHETHER THERE IS
C      IS A LABEL OR NOT. THE USER ENTERS THE STATE OF THE PORT.


C      THE DI NUMBER IS CHECKED TO BE WITHIN VALID LIMITS, AND ITS VALUE
C      TO BE ZERO OR ONE, AND THE TIME LIMIT TO BE WITHIN THE VALID RANGE,
C      OTHERWISE ERROR MESSAGES ARE DISPLAYED.
C      THE VALUES OF THE DI NUMBER, DI VALUE AND THE TIME VALUE ARE
C      PASSED TO THE GRAPHICS DRIVERS

17     IF (L2 .LT. 1 .OR. L2 .GT. 16) CALL USER(7)
       VAL1 =L2
       IF (L4 .NE. 0 .AND. L4 .NE. 1) CALL USER(8)
       VAL2=L4
       IF (TIME .LT. 0.0 .OR. TIME .GT. 25.5) CALL USER(1)

C     VAL1 IS THE PORT NUMBER AND  VAL2 IS THE STATE OF PORT
C     THE USER ENTERS THE STATE OF THE PORT.

       CALL TTOUT(0,16,1,'IS THE PORT ON/OFF (1/0) ?',26,26,31)
       CALL TTOUT(0,17,1,'ENTER 0 OR 1 AND HIT RETURN?',28,28,31)
       READ(*,*)ANS
```

```
      IF (ANS .EQ. 1) THEN
      VAL2 = 1
      ELSE
      VAL2 = 0
      ENDIF

C     DEPENDING ON THE PORT NUMBVER AND THE STATE OF THE PORT THE
C     CORESSPONDING PORT IS GIVEN THE VALUE 0 OR 1

         IF (VAL1 .EQ. 1) THEN
            IF (VAL2 .EQ. 1)  THEN
              P1 = 1
            ELSE
              P1 = 0
            ENDIF
         ELSEIF (VAL1 .EQ. 2) THEN
            IF (VAL2 .EQ. 1) THEN
              P2 = 1
            ELSE
              P2 = 0
            ENDIF
         ELSEIF (VAL1 .EQ. 3) THEN
            IF (VAL2 .EQ. 1) THEN
              P3 = 1
            ELSE
              P3 = 0
            ENDIF
         ELSEIF (VAL1 .EQ. 4) THEN
            IF (VAL2 .EQ. 1)  THEN
              P4= 1
            ELSE
              P4 = 0
            ENDIF
         ELSEIF (VAL1 .EQ. 5) THEN
            IF (VAL2 .EQ. 1)  THEN
              P5 = 1
            ELSE
              P5 = 0
            ENDIF
         ELSEIF (VAL1 .EQ. 6) THEN
            IF (VAL2 .EQ. 1)  THEN
              P6 = 1
            ELSE
              P6 = 0
            ENDIF
         ELSEIF (VAL1 .EQ. 7) THEN
            IF (VAL2 .EQ. 1)  THEN
              P7 = 1
            ELSE
              P7 = 0
            ENDIF
         ELSEIF (VAL1 .EQ. 8) THEN
            IF (VAL2 .EQ. 1)  THEN
              P8 = 1
            ELSE
              P8 = 0
            ENDIF
         ELSEIF (VAL1 .EQ. 9) THEN
            IF (VAL2 .EQ. 1)  THEN
              P9 = 1
            ELSE
              P9 = 0
            ENDIF
         ELSEIF (VAL1 .EQ. 10) THEN
            IF (VAL2 .EQ. 1)   THEN
              P10 = 1
            ELSE
```

```
            P10 = 0
         ENDIF
      ELSEIF (VAL1 .EQ. 11) THEN
         IF (VAL2 .EQ. 1)   THEN
            P11 = 1
         ELSE
            P11 = 0
         ENDIF
      ELSEIF (VAL1 .EQ. 12) THEN
         IF (VAL2 .EQ. 1)   THEN
            P12 = 1
         ELSE
            P12 = 0
         ENDIF
      ELSEIF (VAL1 .EQ. 13) THEN
         IF (VAL2 .EQ. 1)   THEN
            P13 = 1
         ELSE
            P13 = 0
         ENDIF
      ELSEIF (VAL1 .EQ. 14) THEN
         IF (VAL2 .EQ. 1)   THEN
            P14 = 1
         ELSE
            P14 = 0
         ENDIF
      ELSEIF (VAL1 .EQ. 15) THEN
         IF (VAL2 .EQ. 1)   THEN
            P15 = 1
         ELSE
            P15 = 0
         ENDIF
      ELSEIF (VAL1 .EQ. 16) THEN
         IF (VAL2 .EQ. 1)   THEN
            P16 = 1
         ELSE
            P16 = 0
         ENDIF
      ENDIF


       CODE = 2.0

C     CALL THE GRAPHICS DISPLAY PROGRAM

      CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)

C     DEPENDING ON THE VALUE OF VAL2 BEING EQUAL TO L4 THE NEXT LINE
C     OR IF THERE IS A LABEL TO BRANCH TO, IS DECIDED. CONTROL TRANSFERS
C     TO THE STATEMENT NUMBER OF THE LABEL. OTHERWISE, AN ERROR MESSAGE IS
C     DISPLAYED.

         IF (VAL2 .EQ. L4) THEN
            NEXT = NEXT + 1
         ELSE
            IF (L8 .NE. 0) THEN
               NEXT = L8
            ELSE
               CALL USER(9)
            ENDIF

         ENDIF

C     WRITE0

C     READ THE OBJECT FILE FOR THE COMMAND PARAMETERS. L1 IS 1 IF THE
C     PARAMETER IS A VARIABLE NAME, AND 2 IF IT IS A NUMBER. L2 IS THE
```

```
C     VARIABLE / CONSTANT NUMBER. L3 IS 1 IF THE PARAMETER IS A VARIABLE
C     NAME AND 2 IF IT IS A NUMBER. L4 IS THE VARIABLE / CONSTANT NUMBER.
C     L1 AND L2 REPRESENT THE DO PORT, L3 AND L4 REPRESENT THE
C     DO VALUE.

      ELSEIF (K2 .EQ. 2) THEN
         READ (7,200,REC=NEXT) L1,L2,L3,L4

C     THE VARIABLE NAME IS COMPARED AGAINST THE NAMES OF CONSTANTS, COUNTERS
C     AND PARAMETERS IN THE PROGRAM, AND ONCE THE MATCH HAS BEEN FOUND, ITS
C     VALUE IS ASSIGNED TO L2.

         IF (L1 .EQ. 1) THEN

            DO 20 I = 1,NUM1-1

               IF (L2 .EQ. CONS(I)) THEN
                  L2 = CONVAL(I)
                  GOTO 21
               ENDIF

20          CONTINUE

            DO 22 I = 1,NUM3-1

               IF (L2 .EQ. COUNT(I)) THEN
                  L2 = CNTVAL(I)
                  GOTO 21
               ENDIF

22          CONTINUE

            DO 49 I = 1,NUM5-1

               IF (L2 .EQ. PAR(I)) THEN
                  READ (8,500,REC=L2) T
                  L2 = T
                  GOTO 21
               ENDIF

49          CONTINUE

C     IF THE VALUE IS A NUMBER, ITS VALUE IS READ FROM THE CONSTANTS
C     FILE, AND ASSIGNED TO L2.

         ELSEIF (L1 .EQ. 2) THEN
            READ (4,300,REC=L2) T
            L2 = T
         ENDIF

C     THE VARIABLE NAME IS COMPARED AGAINST THE NAMES OF CONSTANTS, COUNTERS
C     AND PARAMETERS IN THE PROGRAM, AND ONCE THE MATCH HAS BEEN FOUND, ITS
C     VALUE IS ASSIGNED TO L4.

21       IF (L3 .EQ. 1) THEN

            DO 23 I = 1,NUM1-1

               IF (L4 .EQ. CONS(I)) THEN
                  L4 = CONVAL(I)
                  GOTO 25
               ENDIF

23          CONTINUE

            DO 24 I = 1,NUM3-1
```

```
               IF (L4 .EQ. COUNT(I)) THEN
                  L4 = CNTVAL(I)
                  GOTO 25
               ENDIF

24          CONTINUE

            DO 59 I = 1,NUM5-1

               IF (L4 .EQ. PAR(I)) THEN
                  READ (8,500,REC=L4) T
                  L4 = T
                  GOTO 25
               ENDIF

59          CONTINUE

C     IF THE VALUE IS A NUMBER, ITS VALUE IS READ FROM THE CONSTANTS
C     FILE, AND ASSIGNED TO L4.

         ELSEIF (L3 .EQ. 2) THEN
            READ (4,300,REC=L4) T
            L4 = T
         ENDIF

25       L2 = L2 + 16
         IONUM(L2) = L4
         IF (L2 .LT. 16 .OR. L2 .GT. 32) CALL USER(10)
          VAL1= L2

         IF (L4 .NE. 0 .AND. L4 .NE. 1) CALL USER(8)
         VAL2 = L4

C     DEPENDING ON THE PORT NUMBER AND THE STATE OF THE PORT THE
C     CORESSPONDING PORT IS GIVEN THE VALUE 0 OR 1

         IF (VAL1 .EQ. 17) THEN
            IF (VAL2 .EQ. 1) THEN
              P17 = 1
            ELSE
              P17 = 0
            ENDIF
         ELSEIF (VAL1 .EQ. 18) THEN
            IF (VAL2 .EQ. 1)   THEN
              P18 = 1
            ELSE
              P18 = 0
            ENDIF
         ELSEIF (VAL1 .EQ. 19) THEN
            IF (VAL2 .EQ. 1)   THEN
              P19 = 1
            ELSE
              P19 = 0
            ENDIF
         ELSEIF (VAL1 .EQ. 20) THEN
            IF (VAL2 .EQ. 1)   THEN
              P20 = 1
            ELSE
              P20 = 0
            ENDIF
         ELSEIF (VAL1 .EQ. 21) THEN
            IF (VAL2 .EQ. 1)   THEN
              P21 = 1
            ELSE
              P21 = 0
            ENDIF
         ELSEIF (VAL1 .EQ. 22) THEN
```

```
   IF (VAL2 .EQ. 1)   THEN
     P22 = 1
   ELSE
     P22 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 23) THEN
   IF (VAL2 .EQ. 1)   THEN
     P23 = 1
   ELSE
     P23 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 24) THEN
   IF (VAL2 .EQ. 1)   THEN
     P24 = 1
   ELSE
     P24 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 25) THEN
   IF (VAL2 .EQ. 1)   THEN
     P25 = 1
   ELSE
     P25 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 26) THEN
   IF (VAL2 .EQ. 1)   THEN
     P26 = 1
   ELSE
     P26 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 27) THEN
   IF (VAL2 .EQ. 1)   THEN
     P27 = 1
   ELSE
     P27 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 28) THEN
   IF (VAL2 .EQ. 1)   THEN
     P28 = 1
   ELSE
     P28 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 29) THEN
   IF (VAL2 .EQ. 1)   THEN
     P29 = 1
   ELSE
     P29 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 30) THEN
   IF (VAL2 .EQ. 1)   THEN
     P30 = 1
   ELSE
     P30 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 31) THEN
   IF (VAL2 .EQ. 1)   THEN
     P31 = 1
   ELSE
     P31 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 32) THEN
   IF (VAL2 .EQ. 1)   THEN
     P32 = 1
   ELSE
     P32 = 0
   ENDIF
ENDIF
```

```
C     VAL1 IS THE OUTPUT PORT NUMBER AND  VAL2 IS THE STATE OF THE PORT

          CODE = 3.0

C     CALL THE GRAPHICS DISPLAY PROGRAM

      CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)

      NEXT = NEXT + 1

      ENDIF

200   FORMAT(17X,7(1X,I4))
210   FORMAT(6X,F8.2)
300   FORMAT(22X,F8.2)
400   FORMAT(2X,I1,2(2X,I8))
500   FORMAT(4X,4(2X,F8.2))
900   FORMAT(2X,I1,2X,I8,2X,I8,2X,F8.2,2X,I8)

      RETURN
      END

C     SUBROUTINE FLOW

C     THIS SUBROUTINE HANDLES ALL THE FLOW OF CONTROL COMMANDS.

C     PARAMETER DEFINITION:

C          K2        -       SUBTYPE OF COMMAND
C                              1 - BRANCH
C                              2 - TESTC
C                              3 - TESTI
C                              4 - TESTP
C                              5 - BREAKPOINT

C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C             CHECK,ERROR,RDIDO,USER,WRITER.

      SUBROUTINE FLOW(K2)

      INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
      INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100),VAL5
      INTEGER SS,SLPAY,PART1,PART2,VAL1,VAL2,RETVAL,BASC
      INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      INTEGER P29,P30,P31,P32,ANS
      REAL THONI,THTWI,ZIN,RIN,CODE
      CHARACTER *10 CELNAM

      DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
      DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
      DIMENSION XPNT(100),YPNT(100),ZPNT(100),RPNT(100)
      DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
      DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)

      COMMON/A/COUNT,CNTVAL,PALLET,PALPTS,PARTNO
      COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
      COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
      COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
      COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
      COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
      COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32
```

```
C     BRANCH

C     READ THE OBJECT FILE FOR THE VARIABLE NUMBER OF THE LABEL. READ THE
C     SYMBOL TABLE FILE FOR THE STATEMENT NUMBER OF THE LABEL, AND
C     ASSIGN IT TO THE VARIABLES L3 AND NEXT, WHICH IS THE NEXT
C     STATEMENT NUMBER TO BE EXECUTED.
C     L1 AND L2 REPRESENT THE LABEL.

      IF (K2 .EQ. 1) THEN
         READ (7,100,REC=NEXT) L1,L2
         READ (8,110,REC=L2) CC
         L3 = CC
         NEXT = L3

C     TESTC

C     READ THE OBJECT FILE FOR THE COMMAND PARAMETERS. L1 IS 1 IF THE
C     PARAMETER IS A VARIABLE NAME, AND 2 IF IT IS A NUMBER. L2 IS THE
C     VARIABLE / CONSTANT NUMBER. L3 IS 1 IF THE PARAMETER IS A VARIABLE
C     NAME AND 2 IF IT IS A NUMBER. L4 IS THE VARIABLE / CONSTANT NUMBER.
C     L5 IS THE VARIABLE NUMBER OF THE LABEL.
C     THE SYMBOL TABLE FILE IS READ FOR THE STATEMENT NUMBER OF THE LABEL,
C     AND THIS VALUE IS ASSIGNED TO L6.
C     L1 AND L2 REPRESENT THE COUNTER NAME, L3 AND L4 REPRESENT THE
C     COUNTER VALUE, L5 REPRESENTS THE LABEL.


      ELSEIF (K2 .EQ. 2) THEN
         READ (7,100,REC=NEXT) L1,L2,L3,L4,L5
         READ (8,110,REC=L5) CC
         L6 = CC

C     THE COUNTER NAME IS COMPARED AGAINST THE NAMES OF COUNTERS AND
C     PARAMETERS IN THE PROGRAM, AND ONCE THE MATCH HAS BEEN FOUND, ITS
C     VALUE IS ASSIGNED TO L2.

         DO 20 I = 1,NUM3-1

            IF (L2 .EQ. COUNT(I)) THEN
               L2 = CNTVAL(I)
               GOTO 21
            ENDIF

20       CONTINUE

         DO 27 I = 1,NUM5-1

            IF (L2 .EQ. PAR(I)) THEN
               READ (8,500,REC=L2) T
               L2 = T
               GOTO 21
            ENDIF

27       CONTINUE

C     IF THE VALUE TO BE COMPARED TO IS A VARIABLE NAME, IT IS CHECKED
C     CHECKED AGAINST NAMES OF CONSTANTS, COUNTERS, AND PARAMETERS TILL
C     A MATCH IS FOUND. THEN ITS VALUE IS ASSIGNED TO L4.

21       IF (L3 .EQ. 1) THEN

            DO 22 I = 1,NUM1-1

               IF (L4 .EQ. CONS(I)) THEN
                  L4 = CONVAL(I)
                  GOTO 23
```

```
               ENDIF

22          CONTINUE

            DO 24 I = 1,NUM3-1

               IF (L4 .EQ. COUNT(I)) THEN
                  L4 = CNTVAL(I)
                  GOTO 23
               ENDIF

24          CONTINUE

            DO 28 I = 1,NUM5-1

               IF (L4 .EQ. PAR(I)) THEN
                  READ (8,500,REC=L4) T
                  L4 = T
                  GOTO 23
               ENDIF

28          CONTINUE

C     IF THE VALUE IS A NUMBER, ITS VALUE IS READ FROM THE CONSTANTS
C     FILE, AND ASSIGNED TO L4.

         ELSEIF (L3 .EQ. 2) THEN
            READ (4,200,REC=L4) T
            L4 = T
         ENDIF

C     THE VALUE IS CHECKED TO BE WITHIN VALID LIMITS, AND IF NOT, AN
C     ERROR MESSAGE IS DISPLAYED. OTHERWISE THE TEST IS DONE, AND
C     DEPENDING ON THE OUTCOME, THE NEXT STATEMENT TO BE EXECUTED
C     IS DETERMINED.

23       IF (L4 .GT. 32767 .OR. L4 .LT. -32767) CALL USER(11)

         IF (L2 .EQ. L4) THEN
            NEXT = L6
         ELSE
            NEXT = NEXT + 1
         ENDIF

C     TESTI

C     READ THE OBJECT FILE FOR THE COMMAND PARAMETERS. L1 IS 1 IF THE
C     PARAMETER IS A VARIABLE NAME, AND 2 IF IT IS A NUMBER. L2 IS THE
C     VARIABLE / CONSTANT NUMBER. L3 IS 1 IF THE PARAMETER IS A VARIABLE
C     NAME AND 2 IF IT IS A NUMBER. L4 IS THE VARIABLE / CONSTANT NUMBER.
C     L5 IS THE VARIABLE NUMBER OF THE LABEL.
C     THE SYMBOL TABLE FILE IS READ FOR THE STATEMENT NUMBER OF THE LABEL,
C     AND THIS VALUE IS ASSIGNED TO L6.
C     L1 AND L2 REPRESENT THE DI PORT, L3 AND L4 REPRESENT THE
C     DI VALUE, L5 REPRSENTS THE LABEL.


      ELSEIF (K2 .EQ. 3) THEN
         READ (7,100,REC=NEXT) L1,L2,L3,L4,L5
         READ (8,110,REC=L5) CC
         L6 = CC

C     THE VARIABLE NAME IS COMPARED AGAINST THE NAMES OF CONSTANTS, COUNTERS
C     AND PARAMETERS IN THE PROGRAM, AND ONCE THE MATCH HAS BEEN FOUND, ITS
C     VALUE IS ASSIGNED TO L2.

         IF (L1 .EQ. 1) THEN
```

```
            DO 30 I = 1,NUM1-1

               IF (L2 .EQ. CONS(I)) THEN
                  L2 = CONVAL(I)
                  GOTO 31
               ENDIF

30          CONTINUE

            DO 37 I = 1,NUM3-1

               IF (L2 .EQ. COUNT(I)) THEN
                  L2 = CNTVAL(I)
                  GOTO 31
               ENDIF

37          CONTINUE

            DO 38 I = 1,NUM5-1

               IF (L2 .EQ. PAR(I)) THEN
                  READ (8,500,REC=L2) T
                  L2 = T
                  GOTO 31
               ENDIF

38          CONTINUE

C     IF THE VALUE IS A NUMBER, ITS VALUE IS READ FROM THE CONSTANTS
C     FILE, AND ASSIGNED TO L2.

            ELSEIF (L1 .EQ. 2) THEN
               READ (4,200,REC=L2) T
               L2 = T
            ENDIF

C     THE VARIABLE NAME IS COMPARED AGAINST THE NAMES OF CONSTANTS, COUNTERS
C     AND PARAMETERS IN THE PROGRAM, AND ONCE THE MATCH HAS BEEN FOUND, ITS
C     VALUE IS ASSIGNED TO L4.

31          IF (L3 .EQ. 1) THEN

               DO 32 I = 1,NUM1-1

                  IF (L4 .EQ. CONS(I)) THEN
                     L4 = CONVAL(I)
                     GOTO 33
                  ENDIF

32             CONTINUE

               DO 34 I = 1,NUM3-1

                  IF (L4 .EQ. COUNT(I)) THEN
                     L4 = CNTVAL(I)
                     GOTO 33
                  ENDIF

34             CONTINUE

               DO 39 I = 1,NUM5-1

                  IF (L4 .EQ. PAR(I)) THEN
                     READ (8,500,REC=L4) T
                     L4 = T
                     GOTO 33
```

```
               ENDIF

39          CONTINUE

C      IF THE VALUE IS A NUMBER, ITS VALUE IS READ FROM THE CONSTANTS
C      FILE, AND ASSIGNED TO L4.

           ELSEIF (L3 .EQ. 2) THEN
              READ (4,200,REC=L4) T
              L4 = T
           ENDIF

C      THE DI PORT IS CHECKED TO BE WITHIN LIMITS, OTHERWISE AN ERROR
C      MESSAGE IS DISPLAYED.
C      THE VALUE OF THE DI PORT IS GOT FROM THE USER AND ITS VALUE IS


33         IF (L2 .LT. 1 .OR. L2 .GT. 16) CALL USER(7)

           VAL1=L2
           VAL2=L4

         CALL TTOUT(0,16,1,'IS THE PORT ON/OFF (1/0) ?',26,26,31)
         CALL TTOUT(0,17,1,'ENTER 0 OR 1 AND HIT RETURN?',28,28,31)
         READ(*,*)ANS
         IF (ANS .EQ. '1') THEN
         VAL2 = 1
         ELSE
         VAL2 = 0
         ENDIF

C      VAL1 IS THE PORT NUMBER AND  VAL2 IS THE STATE OF PORT
C      THE USER ENTERS THE STATE OF THE PORT.

           IF (VAL1 .EQ. 1) THEN
              IF (VAL2 .EQ. 1) THEN
                P1 = 1
              ELSE
                P1 = 0
              ENDIF
           ELSEIF (VAL1 .EQ. 2) THEN
              IF (VAL2 .EQ. 1)  THEN
                P2 = 1
              ELSE
                P2 = 0
              ENDIF
           ELSEIF (VAL1 .EQ. 3) THEN
              IF (VAL2 .EQ. 1)  THEN
                P3 = 1
              ELSE
                P3 = 0
              ENDIF
           ELSEIF (VAL1 .EQ. 4) THEN
              IF (VAL2 .EQ. 1)  THEN
                P4= 1
              ELSE
                P4 = 0
              ENDIF
           ELSEIF (VAL1 .EQ. 5) THEN
              IF (VAL2 .EQ. 1)  THEN
                P5 = 1
              ELSE
                P5 = 0
              ENDIF
           ELSEIF (VAL1 .EQ. 6) THEN
              IF (VAL2 .EQ. 1)  THEN
                P6 = 1
```

```
      ELSE
        P6 = 0
      ENDIF
    ELSEIF (VAL1 .EQ. 7) THEN
      IF (VAL2 .EQ. 1)  THEN
        P7 = 1
      ELSE
        P7 = 0
      ENDIF
    ELSEIF (VAL1 .EQ. 8) THEN
      IF (VAL2 .EQ. 1)  THEN
        P8 = 1
      ELSE
        P8 = 0
      ENDIF
    ELSEIF (VAL1 .EQ. 9) THEN
      IF (VAL2 .EQ. 1)  THEN
        P9 = 1
      ELSE
        P9 = 0
      ENDIF
    ELSEIF (VAL1 .EQ. 10) THEN
      IF (VAL2 .EQ. 1)   THEN
        P10 = 1
      ELSE
        P10 = 0
      ENDIF
    ELSEIF (VAL1 .EQ. 11) THEN
      IF (VAL2 .EQ. 1)   THEN
        P11 = 1
      ELSE
        P11 = 0
      ENDIF
    ELSEIF (VAL1 .EQ. 12) THEN
      IF (VAL2 .EQ. 1)   THEN
        P12 = 1
      ELSE
        P12 = 0
      ENDIF
    ELSEIF (VAL1 .EQ. 13) THEN
      IF (VAL2 .EQ. 1)   THEN
        P13 = 1
      ELSE
        P13 = 0
      ENDIF
    ELSEIF (VAL1 .EQ. 14) THEN
      IF (VAL2 .EQ. 1)   THEN
        P14 = 1
      ELSE
        P14 = 0
      ENDIF
    ELSEIF (VAL1 .EQ. 15) THEN
      IF (VAL2 .EQ. 1)   THEN
        P15 = 1
      ELSE
        P15 = 0
      ENDIF
    ELSEIF (VAL1 .EQ. 16) THEN
      IF (VAL2 .EQ. 1)   THEN
        P16 = 1
      ELSE
        P16 = 0
      ENDIF
    ENDIF

C     VAL1 IS THE PORT AND VAL2 IS THE STATE OF THE PORT (L6=LABEL)
```

```
       CODE = 2.0

C      CALL THE GRAPHICS DISPLAY PROGRAM

       CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)



C         L4 = VAL2
C         IF (L4 .NE. 0 .AND. L4 .NE. 1) CALL USER(8)

C      THE DI VALUE IS THEN CHECKED AGAINST THE USER'S VALUE, AND BASED
C      ON THE OUTCOME, THE NEXT STATEMENT TO BE EXECUTED IS DETERMINED.

          IF (VAL2 .EQ. L4) THEN
             NEXT = L6
          ELSE
             NEXT = NEXT + 1
          ENDIF

C      TESTP

C      READ THE OBJECT FILE FOR THE COMMAND PARAMETERS. L1 IS 1 IF THE
C      PARAMETER IS A VARIABLE NAME, AND 2 IF IT IS A NUMBER. L2 IS THE
C      VARIABLE / CONSTANT NUMBER. L3 IS 1 IF THE PARAMETER IS A VARIABLE
C      NAME AND 2 IF IT IS A NUMBER. L4 IS THE VARIABLE / CONSTANT NUMBER.
C      L5 IS THE VARIABLE NUMBER OF THE LABEL.
C      THE SYMBOL TABLE FILE IS READ FOR THE STATEMENT NUMBER OF THE LABEL,
C      AND THIS VALUE IS ASSIGNED TO L6.
C      L1 AND L2 REPRESENT THE PALLET NAME, L3 AND L4 REPRESENT THE
C      PALLET VALUE, L5 REPRESENTS THE LABEL.


       ELSEIF (K2 .EQ. 4) THEN
          READ (7,100,REC=NEXT) L1,L2,L3,L4,L5
          READ (8,110,REC=L5) CC
          L6 = CC

C      THE VARIABLE NAME IS COMPARED AGAINST THE NAMES OF PALLETS
C      IN THE PROGRAM, AND ONCE THE MATCH HAS BEEN FOUND, ITS
C      VALUE IS ASSIGNED TO L2.

          IF (L1 .EQ. 1) THEN

             DO 40 I = 1,NUM6-1

                IF (L2 .EQ. PALLET(I)) THEN
                   L2 = PARTNO(I)
                   GOTO 41
                ENDIF

40           CONTINUE

          ENDIF

C      THE VARIABLE NAME IS COMPARED AGAINST THE NAMES OF CONSTANTS, COUNTERS
C      AND PARAMETERS IN THE PROGRAM, AND ONCE THE MATCH HAS BEEN FOUND, ITS
C      VALUE IS ASSIGNED TO L4.

41        IF (L3 .EQ. 1) THEN

             DO 42 I = 1,NUM1-1

                IF (L4 .EQ. CONS(I)) THEN
                   L4 = CONVAL(I)
                   GOTO 43
```

```
               ENDIF

42          CONTINUE

            DO 44 I = 1,NUM3-1

               IF (L4 .EQ. COUNT(I)) THEN
                  L4 = CNTVAL(I)
                  GOTO 43
               ENDIF

44          CONTINUE

            DO 49 I = 1,NUM5-1

               IF (L4 .EQ. PAR(I)) THEN
                  READ (8,500,REC=L4) T
                  L4 = T
                  GOTO 43
               ENDIF

49          CONTINUE

C     IF THE VALUE IS A NUMBER, ITS VALUE IS READ FROM THE CONSTANTS
C     FILE, AND ASSIGNED TO L4.

         ELSEIF (L3 .EQ. 2) THEN
            READ (4,200,REC=L4) T
            L4 = T
         ENDIF

C     THE COMPARISON IS THEN MADE, AND BASED ON THE OUTCOME, THE NEXT
C     STATEMENT TO BE EXECUTED IS DETERMINED.

43       IF (L2 .EQ. L4) THEN
            NEXT = L6
         ELSE
            NEXT = NEXT + 1
         ENDIF

C     BREAKPOINT

C     WHEN THIS COMMAND IS ENCOUNTERD, A MESSAGE IS SEND TO THE USER
C     INFORMING HIM OF THE OCCURENCE OF A BREAKPOINT AND TO HIT ANY KEY
C     TO CONTINUE WITH THE PROGRAM SIMULATION.

      ELSEIF (K2 .EQ. 5) THEN


         CALL USER(12)
         NEXT = NEXT + 1

      ENDIF

99    FORMAT(2X,I1,3(2X,I8))
100   FORMAT(17X,5(1X,I4))
110   FORMAT(6X,F8.2)
200   FORMAT(22X,F8.2)
500   FORMAT(4X,4(2X,F8.2))

      RETURN
      END

C     SUBROUTINE COUN

C     THIS SUBROUTINE HANDLES ALL THE COUNTER COMMANDS.
```

```
C     PARAMETER DEFINITION:

C          K2        -       SUBTYPE OF COMMAND
C                              1 - COMPC
C                              2 - DECR
C                              3 - INCR
C                              4 - SETC

C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C             USER.

      SUBROUTINE COUN(K2)

      INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
      INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100),VAL5
      INTEGER SS,SLPAY,PART1,PART2,VAL1,VAL2,RETVAL,BASC
      INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      INTEGER P29,P30,P31,P32
      REAL THONI,THTWI,ZIN,RIN,CODE
      CHARACTER *10 CELNAM

      DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
      DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
      DIMENSION XPNT(100),YPNT(100),ZPNT(100),RPNT(100)
      DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
      DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)

      COMMON/A/COUNT,CNTVAL,PALLET,PALPTS,PARTNO
      COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
      COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
      COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
      COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
      COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
      COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32


C     COMPC

C     READ THE COMMAND PARAMETER NUMBERS. L1 IS THE VARIABLE NUMBER OF
C     OF THE COUNTER. L2 IS THE CONDITION IN THE STATEMENT. L3 IS 2
C     IF THE VALUE IS A NUMBER AND 1 IF THE VALUE IS SPECIFIED AS A
C     COUNTER OR A CONSTANT. L5 IS THE LABEL NUMBER. THE SYMBOL TABLE
C     FILE IS READ AR RECORD L5, TO READ THE STATEMENT NUMBER OF THE
C     LABEL, AND L6 IS ASSIGNED THE STATEMENT NUMBER.
C     L1 REPRESENTS COUNTER NAME, L2 REPRESENTS THE CONDITION,,
C     L3 AND L4 REPRESENT THE COUNTER VALUE, L5 REPRESENTS THE LABEL.


      IF (K2 .EQ. 1) THEN
         READ (7,300,REC=NEXT) L1,L2,L3,L4,L5
         READ (8,310,REC=L5) CC
         L6 = CC

C     THE COUNTER TO BE COMPARED IS CHECKED AGAINST THE COUNTER NAMES
C     AND PARAMETER NAMES, AND ITS VALUE IS ASSIGNED TO VARIABLE J1.

           DO 10 I = 1,NUM3-1

              IF (L1 .EQ. COUNT(I)) THEN
                 J1 = CNTVAL(I)
```

```
                  GOTO 11
               ENDIF

10          CONTINUE

            DO 17 I = 1,NUM5-1

               IF (L1 .EQ. PAR(I)) THEN
                  READ (8,600,REC=L1) T
                  J1 = T
                  GOTO 11
               ENDIF

17          CONTINUE

C     THE VARIABLE NAME IS COMPARED AGAINST CONSTANTS, COUNTERS, AND
C     PARAMETER NAMES, AND ITS VALUE IS ASSIGNED TO J2.

11          IF (L3 .EQ. 1) THEN

               DO 14 I = 1,NUM1-1

                  IF (L4 .EQ. CONS(I)) THEN
                     J2 = CONVAL(I)
                     GOTO 13
                  ENDIF

14             CONTINUE

               DO 12 I = 1,NUM3-1

                  IF (L4 .EQ. COUNT(I)) THEN
                     J2 = CNTVAL(I)
                     GOTO 13
                  ENDIF

12             CONTINUE

            DO 18 I = 1,NUM5-1

               IF (L4 .EQ. PAR(I)) THEN
                  READ (8,600,REC=L4) T
                  J2 = T
                  GOTO 13
               ENDIF

18          CONTINUE

C     IF THE VALUE IS A NUMBER, IT IS READ FROM THE CONSTANTS FILE, AND
C     ASSIGNED TO J2.

            ELSEIF (L3 .EQ. 2) THEN
               READ (4,400,REC=L4) T
                  J2 = T
                  GOTO 13
            ENDIF

C     THE CONDITIONS ARE COMPARED HERE. L2 IS 1 IF THE CONDITION IS
C     "<", 2 FOR "<=", 3 FOR ">", 4 FOR "=>", 5 FOR "=", AND 6 FOR "<>".
C     THE NEXT STATEMENT TO BE EXECUTED IS ASSIGNED TO THE VARIABLE NEXT,
C     DEPENDING ON THE OUTCOME OF THE COMPARISON.

13          IF (L2 .EQ. 1 .AND. J1 .LT. J2) THEN
               NEXT = L6
            ELSEIF (L2 .EQ. 2 .AND. J1 .LE. J2) THEN
               NEXT = L6
            ELSEIF (L2 .EQ. 3 .AND. J1 .GT. J2) THEN
```

```
            NEXT = L6
         ELSEIF (L2 .EQ. 4 .AND. J1 .GE. J2) THEN
            NEXT = L6
         ELSEIF (L2 .EQ. 5 .AND. J1 .EQ. J2) THEN
            NEXT = L6
         ELSEIF (L2 .EQ. 6 .AND. J1 .NE. J2) THEN
            NEXT = L6
         ELSE
            NEXT = NEXT + 1
         ENDIF

         GOTO 910

C     DECR

C     THE OBJECT FILE IS READ FOR THE COUNTER VARIABLE NUMBER.
C     L1 AND L2 REPRESENT THE COUNTER NAME.

      ELSEIF (K2 .EQ. 2) THEN
         READ (7,300,REC=NEXT) L1,L2

C     THE COUNTER IS COMPARED TO THE COUNTER NAMES IN THE PROGRAM, AND
C     ONCE THE MATCH HAS BEEN FOUND, THE VALUE IS DECREMENTED, AND ITS
C     VALUE IS WRITTEN TO THE SYMBOL TABLE FILE.

         DO 20 I = 1,NUM3-1

            IF (L2 .EQ. COUNT(I)) THEN
               CNTVAL(I) = CNTVAL(I) - 1
               CC = CNTVAL(I)
               WRITE (8,200,REC=L2) L2,CC
               NEXT = NEXT + 1
               GOTO 910
            ENDIF

20       CONTINUE

C     INCR

C     THE OBJECT FILE IS READ FOR THE COUNTER VARIABLE NUMBER.
C     L1 AND L2 REPRESENT THE COUNTER NAME.

      ELSEIF (K2 .EQ. 3) THEN
         READ (7,300,REC=NEXT) L1,L2

C     THE COUNTER IS COMPARED TO THE COUNTER NAMES IN THE PROGRAM, AND
C     ONCE THE MATCH HAS BEEN FOUND, THE VALUE IS INCREMENTED, AND ITS
C     VALUE IS WRITTEN TO THE SYMBOL TABLE FILE.

         DO 30 I = 1,NUM3-1

            IF (L2 .EQ. COUNT(I)) THEN
               CNTVAL(I) = CNTVAL(I) + 1
               CC = CNTVAL(I)
               WRITE (8,200,REC=L2) L2,CC
               NEXT = NEXT + 1
               GOTO 910
            ENDIF

30       CONTINUE

C     SETC

C     THE OBJECT FILE IS READ FOR THE COUNTER VARIABLE NUMBER AND THE
C     CONSTANT / VARIABLE NUMBER TO BE ASSIGNED TO IT.
C     L1 AND L2 REPRESENT THE COUNTER NAME, L3 AND L4 REPRESENT THE
C     COUNTER VALUE.
```

```
      ELSEIF (K2 .EQ. 4) THEN
         READ (7,300,REC=NEXT) L1,L2,L3,L4

C     THE COUNTER NAME IS COMPARED TO THE NAMES IN THE PROGRAM. ONCE THE
C     MATCH AHS BEEN FOUND, THE INDEX FOR THE COUNTER IS ASSIGNED TO L5.

         DO 40 I = 1,NUM3-1

            IF (L2 .EQ. COUNT(I)) THEN
               L5 = I
               GOTO 999
            ENDIF

40       CONTINUE

C     THE VARIABLE NAME FOR THE COUNTER VALUE IS CHECKED AGAINST
C     CONSTANTS, COUNTERS, AND PARAMETERS NAMES, AND THE VALUE IS
C     ASSIGNED TO THE ARRAY CNTVAL.

999      IF (L3 .EQ. 1) THEN

            DO 41 I = 1,NUM1-1

               IF (L4 .EQ. CONS(I)) THEN
                  CNTVAL(L5) = CONVAL(I)
                  NEXT = NEXT + 1
                  GOTO 990
               ENDIF

41          CONTINUE

            DO 42 I = 1,NUM3-1

               IF (L4 .EQ. COUNT(I)) THEN
                  CNTVAL(L5) = CNTVAL(I)
                  NEXT = NEXT + 1
                  GOTO 990
               ENDIF

42          CONTINUE

            DO 43 I = 1,NUM5-1

               IF (L4 .EQ. PAR(I)) THEN
                  READ(8,600,REC=L4) T
                  CNTVAL(L5) = T
                  NEXT = NEXT + 1
                  GOTO 990
               ENDIF

43          CONTINUE

C     IF THE VALUE IS SPECIFIED AS A NUMBER, THE CONSTANTS FILE IS
C     READ AND THE VALUE IS ASSIGNED TO THE ARRAY CNTVAL.

         ELSEIF (L3 .EQ. 2) THEN
            READ (4,400,REC=L4) T
            CNTVAL(L5) = T
            NEXT = NEXT + 1
         ENDIF

C     THE COUNTER VALUE IS CHECKED TO BE WITHIN LIMITS, AND IF NOT, AN
C     ERROR MESSAGE IS DISPLAYED. OTHERWISE, THE VALUE IS WRITTEN
C     TO THE SYMBOL TABLE FILE.
```

```
990       CC = CNTVAL(L5)
          IF (CC .GT. 32767 .OR. CC .LT. -32767) CALL USER(11)
          WRITE (8,200,REC=L2) L2,CC

      ENDIF

200   FORMAT(1X,I3,2X,F8.2)
300   FORMAT(17X,5(1X,I4))
310   FORMAT(6X,F8.2)
400   FORMAT(22X,F8.2)
500   FORMAT(2X,I1,5(2X,I8))
600   FORMAT(4X,4(2X,F8.2))

910   RETURN
      END

C     SUBROUTINE PALL

C     THIS SUBROUTINE HANDLES ALL THE PALLET COMMANDS.

C     PARAMETER DEFINITION:

C          K2        -       SUBTYPE OF COMMAND
C                              1 - GETPART
C                              2 - NEXTPART
C                              3 - PREVPART
C                              4 - SETPART

C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C             CHECK,ERROR,USER,WRITEP,WRITER.

      SUBROUTINE PALL(K2)

      INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
      INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100),VAL5
      INTEGER SS,SLPAY,PART1,PART2,VAL1,VAL2,RETVAL,BASC
      INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      INTEGER P29,P30,P31,P32
      REAL THONI,THTWI,ZIN,RIN,CODE
      CHARACTER *10 CELNAM

      DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
      DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
      DIMENSION XPNT(100),YPNT(100),ZPNT(100),RPNT(100)
      DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
      DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)

      COMMON/A/COUNT,CNTVAL,PALLET,PALPTS,PARTNO
      COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
      COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
      COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
      COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
      COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
      COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

C     GETPART

C     READ THE COMMAND PARAMETER VARIABLE NUMBER.
C     L1 AND L2 REPRESENT THE PALLET NAME.

      IF (K2 .EQ. 1) THEN
```

```
         READ (7,100,REC=NEXT) L1,L2

C     CHECK THE VARIABLE NUMBER AGAINST PALLET NUMBERS. TRANSMIT THE
C     COORDINATES OF THE PALLET POINT OT THE GRAPHICS DRIVERS. WITH A CODE

         IF (L1 .EQ. 1) THEN

            DO 10 I = 1,NUM6-1

               IF (L2 .EQ. PALLET(I)) THEN
                  XVAL = XPAL(I,PARTNO(I))
                  YVAL = YPAL(I,PARTNO(I))
                  RVAL = RPAL(I,PARTNO(I))

         CODE = 1.0

C     CALL THE GRAPHIC DISPLAY PROGRAM

      CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)

              GO TO 900

               ENDIF

10          CONTINUE

         ENDIF

C     NEXTPART

C     READ THE COMMAND PARAMETER.
C     L1 AND L2 REPRESENT THE PALLET NAME.

      ELSEIF (K2 .EQ. 2) THEN
         READ (7,100,REC=NEXT) L1,L2

C     CHECK THE VARIABLE NUMBER AGAINST PALLET NUMBERS. ONCE THE PALLET
C     NAME HAS BEEN FOUND, INCREMENT THE ARRAY PARTNO, IF ITS VALUE
C     IS LESS THAN THE TOTAL NUMBER OF POINTS. OTHERWISE SET PARTNO TO 1.
C     THE Z VALUE IS SET TO THE CURRENT Z VALUE OF THE ARM.

         IF (L1 .EQ. 1) THEN

            DO 20 I = 1,NUM6-1

               IF (L2 .EQ. PALLET(I)) THEN

                  IF (PARTNO(I) .LT. PALPTS(I)) THEN
                     PARTNO(I) = PARTNO(I) + 1
                  ELSE
                     PARTNO(I) = 1
                  ENDIF

                  ZPAL(I,PARTNO(I)) = ZVAL
                  GOTO 900

               ENDIF

20          CONTINUE

         ENDIF

C     PREVPART

C     READ THE COMMAND PARAMETER.
C     L1 AND L2 REPRESENT THE PALLET NAME.
```

```
      ELSEIF (K2 .EQ. 3) THEN
         READ (7,100,REC=NEXT) L1,L2

C     CHECK THE VARIABLE NUMBER AGAINST PALLET NUMBERS. ONCE THE PALLET
C     NAME HAS BEEN FOUND, DECREMENT THE ARRAY PARTNO, IF ITS VALUE
C     IS GREATER THAN 1. OTHERWISE SET PARTNO TO THE MAXIMUM VALUE OF PARTS.
C     THE Z VALUE IS SET TO THE CURRENT Z VALUE OF THE ARM.

         IF (L1 .EQ. 1) THEN

            DO 30 I = 1,NUM6-1

               IF (L2 .EQ. PALLET(I)) THEN

                  IF (PARTNO(I) .GT. 1) THEN
                     PARTNO(I) = PARTNO(I) - 1
                  ELSE
                     PARTNO(I) = PALPTS(I)
                  ENDIF

                  ZPAL(I,PARTNO(I)) = ZVAL
                  GOTO 900
               ENDIF

30          CONTINUE

         ENDIF

C     SETPART

C     READ THE COMMAND PARAMETERS.
C     L1 AND L2 REPRESENT THE PALLETNAME, L3 AND L4 REPRESENT THE
C     PALLET VALUE.

      ELSEIF (K2 .EQ. 4) THEN
         READ (7,100,REC=NEXT) L1,L2,L3,L4

C     SEE IF THE PALLET VALUE IS DEFINED AS A CONSTANT, COUNTER OR AS A
C     PARAMETER AND DETERMINE ITS VALUE AND ASSIGN IT TO L4.

         IF (L3 .EQ. 1) THEN

            DO 40 I = 1,NUM1-1

               IF (L4 .EQ. CONS(I)) THEN
                  L4 = CONVAL(I)
                  GOTO 41
               ENDIF

40          CONTINUE

            DO 42 I = NUM3-1

               IF (L4 .EQ. COUNT(I)) THEN
                  L4 = CNTVAL(I)
                  GOTO 41
               ENDIF

42          CONTINUE

            DO 43 I = 1,NUM5-1

               IF (L4 .EQ. PAR(I)) THEN
                  READ (8,200,REC=L4) T
                  L4 = T
                  GOTO 41
               ENDIF
```

```
43          CONTINUE

C     IF THE PALLET VALUE IS A NUMBER, READ FROM THE CONSTANTS FILE AND
C     ASSIGN IT TO THE VARIABLE L4.

         ELSEIF (L3 .EQ. 2) THEN
            READ (4,300,REC=L4) T
            L4 = T
         ENDIF

C     CHECK THE VARIABLE NUMBER OF THE PALLET AGAINST THE PALLET NUMBERS.
C     ONCE THE MATCH HAS BEEN FOUND, SET THE ARRAY PARTNO TO L4, IF L4
C     IS IN THE VALID RANGE, OTHERWISE, DISPLAY AN ERROR MESSAGE.
C     ALSO SET THE Z VALUE OF THE POINT EQUAL TO THE CURRENT Z VALUE.

41       IF (L1 .EQ. 1) THEN

            DO 44 I = 1,NUM6-1

               IF (L2 .EQ. PALLET(I)) THEN

                  IF (L4 .LE. PALPTS(I)) THEN
                     PARTNO(I) = L4
                  ELSE
                     CALL USER(13)
                  ENDIF

                  ZPAL(I,PARTNO(I)) = ZVAL
                  GOTO 900
               ENDIF

44          CONTINUE

         ENDIF

      ENDIF

99    FORMAT(2X,I1,4(2X,F8.2))
100   FORMAT(17X,7(1X,I4))
200   FORMAT(6X,F8.2)
300   FORMAT(22X,F8.2)

C     SET THE NEXT STATEMENT NUMBER TO BE EXECUTED, AND RETURN.

900   NEXT = NEXT + 1

      RETURN
      END

C     SUBROUTINE SUB1

C     THIS SUBROUTINE HANDLES SUBROUTINE COMMANDS

C     PARAMETER DEFINITION:

C          K2        -       SUBTYPE OF COMMAND
C                               1 - SUBR
C                               2 - END

      SUBROUTINE SUB1(K2)

      INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
      INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100),VAL5
      INTEGER SS,SLPAY,PART1,PART2,VAL1,VAL2,RETVAL,BASC
      INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
```

```
      INTEGER P29,P30,P31,P32
      REAL THONI,THTWI,ZIN,RIN,CODE
      CHARACTER *10 CELNAM

      DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
      DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
      DIMENSION XPNT(100),YPNT(100),ZPNT(100),RPNT(100)
      DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
      DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)

      COMMON/A/COUNT,CNTVAL,PALLET,PALPTS,PARTNO
      COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
      COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
      COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
      COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
      COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
      COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

C     IF THE COMMAND IS "END", READ THE OBJECT FILE TO FIGURE OUT THE
C     THE NEXT STATEMENT TO BE EXECUTED, AND ASSIGN IT TO THE VARIABLE, NEXT.

      IF (K2 .EQ. 2) THEN
         READ (7,100,REC=NEXT). L2
         NEXT = L2
      ENDIF

99    FORMAT(2X,I1,2X,I8)
100   FORMAT(23X,I4)

      RETURN
      END

C     SUBROUTINE SUB2

C     THIS SUBROUTINE HANDLES CALLS TO SUBROUTINES.
C     THIS HANDLES A MAXIMUM OF 5 FORMAL PARAMETERS.

C     PARAMETER DEFINITION:

C          K2        -       STATEMENT NUMBER WHERE SUBROUTINE IS DEFINED

      SUBROUTINE SUB2(K2)

      INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
      INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100),VAL5
      INTEGER SS,SLPAY,PART1,PART2,VAL1,VAL2,RETVAL,BASC
      REAL THONI,THTWI,ZIN,RIN,CODE
      CHARACTER *10 CELNAM

      DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
      DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
      DIMENSION XPNT(100),YPNT(100),ZPNT(100),RPNT(100)
      DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
      DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)

      COMMON/A/COUNT,CNTVAL,PALLET,PALPTS,PARTNO
      COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
      COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
      COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
      COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
      COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
      COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
```

```
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VALS
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32


      NN = 1

C     READ THE STATEMENT WHERE SUBROUTINE IS DEFINED FOR THE STATEMENT
C     NUMBER OF THE "END" STATEMENT OF SUBROUTINE, L1, FIRST EXECUTABLE
C     STATEMENT OF SUBROUTINE, L2, AND THE FIVE FORMAL PARAMETER NUMBERS,
C     L(1) THROUGH L(5).

      READ (7,100,REC=K2) L1,L2,(L(I),I=1,5)

C     READ THE "END" STATEMENT OF SUBROUTINE, AND WRITE THE SAME
C     INFORMATION ALONG WITH THE NEXT STATEMENT NUMBER TO BE EXECUTED
C     ONCE THE EXECUTION OF THE SUBROUTINE IS COMPLTE.

      READ (7,110,REC=L1) N1,N2,N3,N4,N5
      WRITE (7,110,REC=L1) N1,N2,N3,N4,N5,NEXT+1

C     READ THE STATEMENT WHERE THE CALL TO THE SUBROUTINE IS MADE, FOR
C     THE VARIABLE / CONSTANT NUMBERS OF THE FIVE FORMAL PARAMETERS.

      READ (7,120,REC=NEXT) (M(I),I=1,10)

C     CHECK IF THERE ARE ANY FORMAL PARAMETERS. IF THERE ARE ANY, SEE
C     IF THEY ARE PASSED AS NUMBERS OR AS VARIABLE NAMES.

      DO 10 I = 1,9,2
         IF (M(I) .EQ. 0) GOTO 999

C     IF THE PARAMETER IS A VARIABLE NAME, SEE IF THE VARIABLE IS A
C     CONSTANT, OR A POINT, OR A COUNTER, OR AN AGGREGATE.
C     DEPENDING ON THE TYPE OF VARIABLE, EITHER ONE OR FOUR VALUES
C     ARE ASSIGNED TO THE ARRAY P, AND VARIABLE NN IS SET TO 1 OR 4.

         IF (M(I) .EQ. 1) THEN
            DO 11 J = 1,NUM1-1

               IF (M(I+1) .EQ. CONS(J)) THEN
                  P(1) = CONVAL(J)
                  NN = 1
                 GOTO 9
               ENDIF

11          CONTINUE

            DO 12 J = 1,NUM2-1

               IF (M(I+1) .EQ. PNTVAL(J)) THEN
                  P(1) = XPNT(J)
                  P(2) = YPNT(J)
                  P(3) = ZPNT(J)
                  P(4) = RPNT(J)
                  NN = 4
                 GOTO 9
               ENDIF

12          CONTINUE

            DO 13 J = 1,NUM3-1

               IF (M(I+1) .EQ. COUNT(J)) THEN
                  P(1) = CNTVAL(J)
```

```
                  NN = 1
                  GOTO 9
               ENDIF

13          CONTINUE

            DO 14 J = 1,NUM4-1

               IF (M(I+1) .EQ. AGG(J)) THEN
                  P(1) = XAGG(J)
                  P(2) = YAGG(J)
                  P(3) = ZAGG(J)
                  P(4) = RAGG(J)
                  NN = 4
                  GOTO 9
               ENDIF

14          CONTINUE

C     IF THE PARAMETER IS A CONSTANT, READ THE CONSTANTS FILE FOR THE
C     VALUE, AND SET NN EQUAL TO 1.

         ELSEIF (M(I) .EQ. 2) THEN
            READ (4,300,REC=M(I+1)) IC1,P(1)
            NN = 1
         ENDIF

9        II = (I+1) / 2

C     WRITE THE VALUE PASSED INTO THE LOCATION OF THE FORMAL PARAMETER
C     IN THE SYMBOL TABLE FILE.

         IF (L(II) .GT. 0) THEN
            WRITE (8,400,REC=L(II)) L(II),(P(K),K=1,NN)
         ENDIF

10    CONTINUE

C     ONCE ALL THE FORMAL PARAMETERS HAVE BEEN IDENTIFIED, SET THE
C     NEXT STATEMENT TO BE EXECUTED TO BE THE FIRST EXECUTABLE
C     STATEMENT OF THE SUBROUTINE.

999   NEXT = L2

99    FORMAT(2X,I1,2X,I8)
100   FORMAT(17X,7(1X,I4))
110   FORMAT(1X,I4,1X,I4,1X,I1,3(1X,I4))
120   FORMAT(17X,10(1X,I4))
300   FORMAT(11X,I4,7X,F8.2)
400   FORMAT(1X,I3,4(2X,F8.2))

      RETURN
      END

C     SUBROUTINE WHERE

C     THIS SUBROUTINE HANDLES THE "WHERE" COMMAND.

      SUBROUTINE WHERE

      INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
      INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100),VAL5
      INTEGER SS,SLPAY,PART1,PART2,VAL1,VAL2,RETVAL,BASC
      REAL THONI,THTWI,ZIN,RIN,CODE
      CHARACTER *10 CELNAM

      DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
```

```
      DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
      DIMENSION XPNT(100),YPNT(100),ZPNT(100),RPNT(100)
      DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
      DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)

      COMMON/A/COUNT,CNTVAL,PALLET,PALPTS,PARTNO
      COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
      COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
      COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
      COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
      COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
      COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

C     THE PROGRAM READS THE VARIABLE NUMBER OF THE COMMAND PARAMETER.
C     L1 AND L2 REPRESENT THE VARIABLE NAME.

      READ (7,100,REC=NEXT) L1,L2

      IF (L1 .EQ. 1) THEN

C     CHECK IF THE VARIABLE IS A POINT. IF SO, SET THE CURRENT VALUE
C     OF THE COORDINATES OF THE ROBOT ARM TO THE ARRAYS, XPNT,YPNT,
C     ZPNT AND RPNT.

         DO 10 I = 1,NUM2-1

            IF (L2 .EQ. PNTVAL(I)) THEN
               XPNT(I) = XVAL
               YPNT(I) = YVAL
               ZPNT(I) = ZVAL
               RPNT(I) = RVAL
               GOTO 11
            ENDIF

10       CONTINUE

C     IF THE VARIABLE IS A FORMAL PARAMETER, ASSIGN THE COORDINATES TO
C     THE ARRAYS XPAR,YPAR,ZPAR, AND RPAR.

         DO 12 I = 1,NUM5-1

            IF (L2 .EQ. PAR(I)) THEN
               XPAR(I) = XVAL
               YPAR(I) = YVAL
               ZPAR(I) = ZVAL
               RPAR(I) = RVAL
               GOTO 11
            ENDIF

12       CONTINUE
      ENDIF

C     INCREMENT THE STATEMENT NUMBER TO BE EXECUTED, AND WRITE THE
C     THE CURRENT VALUE OF THE ROBOT ARM COORDINATES INTO THE LOCATION
C     OF THE VARIABLE IN THE SYMBOL TABLE FILE.

11    NEXT = NEXT + 1
      WRITE (8,200,REC=L2) L2,XVAL,YVAL,ZVAL,RVAL

100   FORMAT(17X,2(1X,I4))
200   FORMAT(1X,I3,4(2X,F8.2))
300   FORMAT(2X,I1,4(2X,F8.2))
```

```
      RETURN
      END


C     SUBROUTINE USER

C     THIS SUBROUTINE REPORTS USER ERRORS IN THE AML/E PROGRAM AND
C     ERRORS DURING USER INPUT FOR INTERACTIVE SIMULATION.

C     PARAMETER DEFINITION:

C          NNNN     -      ERROR NUMBER

C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C             CSROFF,CSRON,FINI,INKEY,TTOUT.

      SUBROUTINE USER(NNNN)

      INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
      INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100),VAL5
      INTEGER SS,SLPAY,PART1,PART2,VAL1,VAL2,RETVAL,BASC
      INTEGER * 2 KYCOD1,KYCOD2
      INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      INTEGER P29,P30,P31,P32
      CHARACTER * 1 BLANK(80),CELNAM*10
      REAL THONI,THTWI,ZIN,RIN,CODE

      DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
      DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
      DIMENSION XPNT(100),YPNT(100),ZPNT(100),RPNT(100)
      DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
      DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)

      COMMON/A/COUNT,CNTVAL,PALLET,PALPTS,PARTNO
      COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
      COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
      COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
      COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
      COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
      COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

      DATA BLANK/80 * ' '/

C     THE ERROR NUMBERS ARE LESS THAN 20 FOR ERRORS IN THE USER'S
C     PROGRAM. THE SAME ERRORS DURING USER INPUT FOR INTERACTIVE
C     SIMULATION HAVE A VALUE OF 20 ADDED TO THEM. IF THE ERROR
C     IS GREATER THAN 20, ISET IS SET TO 1, AND PROGRAM SIMULATION
C     DOES NOT TERMINATE. HOWEVER, IF THE ERROR IS IN THE USER'S
C     PROGRAM, PROGRAM SIMULATION TERMINATES.

      CALL CSROFF
      ISET = 0

      IF (NNNN .GT. 20) THEN
         ISET = 1
         NN = NNNN - 20
      ELSE
         NN = NNNN
      ENDIF
```

```
C     MESSAGE IS DISPLAYED, AND PROGRAM SIMULATION DOES NOT TERMINATE WHEN
C     THE COMMAND "BREAKPOINT" IS FOUND IN THE USER'S PROGRAM.

      IF (NN .EQ. 12) THEN
      CALL TTOUT(0,20,1,'BREAKPOINT found. Hit any key to resume...',
     *42,42,28)
          CALL INKEY(KYCOD1,KYCOD2)
          CALL TTOUT(0,20,1,BLANK,80,80,31)
          RETURN
      ENDIF

      IF (NN .EQ. 1) THEN
      CALL TTOUT(0,20,1,'Time value out of range. Simulation aborted.',
     *44,44,28)

      ELSEIF (NN .EQ. 2) THEN
      CALL TTOUT(0,20,1,'R value out of range. Simulation aborted.',
     *41,41,28)

      ELSEIF (NN .EQ. 3) THEN
      CALL TTOUT(0,20,1,'Z value out of range. Simulation aborted.',
     *41,41,28)

      ELSEIF (NN .EQ. 4) THEN
      CALL TTOUT(0,20,1,'X value out of range. Simulation aborted.',
     *41,41,28)

      ELSEIF (NN .EQ. 5) THEN
      CALL TTOUT(0,20,1,'Y value out of range. Simulation aborted.',
     *41,41,28)

      ELSEIF (NN .EQ. 7) THEN
      CALL TTOUT(0,20,1,'DI port out of range. Simulation aborted.',
     *41,41,28)

      ELSEIF (NN .EQ. 8) THEN
      CALL TTOUT(0,20,1,'DI/DO value out of range. Simulation aborted.',
     *45,45,28)

      ELSEIF (NN .EQ. 9) THEN
      CALL TTOUT(0,20,1,'Timeout error in WAITI. Simulation aborted.',
     *43,43,28)

      ELSEIF (NN .EQ. 10) THEN
      CALL TTOUT(0,20,1,'DO port out of range. Simulation aborted.',
     *41,41,28)

      ELSEIF (NN .EQ. 11) THEN
      CALL TTOUT(0,20,1,
     *'Counter value out of range. Simulation aborted.',47,47,28)

      ELSEIF (NN .EQ. 13) THEN
      CALL TTOUT(0,20,1,'Pallet value out of range. Simulation aborted.',
     *46,46,28)

      ENDIF

      IF (ISET .EQ. 1) THEN
         CALL TTOUT(0,23,1,'Error in value.............',29,29,28)
         CALL TTOUT(0,24,1,'Press any key to continue....',29,29,28)
         CALL INKEY(KYCOD1,KYCOD2)
         CALL TTOUT(0,20,1,BLANK,80,80,31)
         CALL TTOUT(0,24,1,BLANK,80,80,31)
         CALL CSRON
         RETURN

      ELSE
```

```
      CALL TTOUT(0,24,1,'Hit any to key to terminate Simulation...',
     *41,41,28)
         CALL INKEY(KYCOD1,KYCOD2)
         CALL CSRON
         CALL FINI

      ENDIF

      END


C     SUBROUTINE PART

C     THIS SUBROUTINE ASKS THE USERS IF THEY WANT TO EXECUTE THE
C     COMPLETE PROGRAM OR ONLY A PARTIAL PROGRAM.

C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C             CLS,CSROFF,CSRON,FINI,GOSUB,INKEY,LOCATE,PARTN,TTOUT.

      SUBROUTINE PART

      INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
      INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100),VAL5
      INTEGER SS,SLPAY,PART1,PART2,VAL1,VAL2,RETVAL,BASC
      INTEGER * 2 KYCOD1,KYCOD2,JJ1(4),JJ2(4)
      INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      INTEGER P29,P30,P31,P32
      CHARACTER * 1 BLANK(80),C(4),CELNAM*10
      REAL THONI,THTWI,ZIN,RIN,CODE

      DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
      DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
      DIMENSION XPNT(100),YPNT(100),ZPNT(100),RPNT(100)
      DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
      DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)

      COMMON/A/COUNT,CNTVAL,PALLET,PALPTS,PARTNO
      COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
      COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
      COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
      COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
      COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
      COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

      DATA BLANK/80 * ' '/

*     DISPLAY THE EXECUTION OPTIONS MENU AND READ USER'S INPUT.

33    CALL CLS(31)
      CALL CSRON
      CALL TTOUT(0,1,29,'SIMULATION OPTIONS MENU',23,23,31)
      CALL TTOUT(0,2,29,'-----------------------',23,23,31)
      CALL TTOUT(0,5,31,'Select a function:',18,18,27)
      CALL TTOUT(0,9,26,'FN1  -  Complete program Simulation',35,35,31)
      CALL TTOUT(0,10,26,'FN2  -  Partial program Simulation',34,34,31)
      CALL TTOUT(0,11,26,'FN3  -  Subroutine Simulation',29,29,31)
      CALL TTOUT(0,12,26,'FN4  -  Abort Simulation',24,24,31)
      CALL TTOUT(0,20,31,'Enter option===>',16,16,27)
36    CALL LOCATE(0,20,48)
      CALL INKEY(KYCOD1,KYCOD2)
```

```
C     IF THE USER'S INPUT IS INVALID, DISPLAY ERROR MESSAGE.

      IF (KYCOD1 .NE. 0) THEN
         CALL CSROFF
         CALL TTOUT(0,24,1,'KEY NOT DEFINED -- Hit any key to resume',
     *   40,40,28)
         CALL INKEY(KYCOD1,KYCOD2)
         CALL CSRON
         CALL TTOUT(0,24,1,BLANK,80,80,31)
         GOTO 36
      ENDIF

C     IF THE USER STRIKES THE F1 KEY, RETURN AND SIMULATE THE
C     COMPLETE PROGRAM.

      IF (KYCOD2 .EQ. 59) THEN
         RETURN

C     IF THE USER STRIKES THE F2 KEY, PROMPT THE USER FOR THE LINE
C     NUMBERS AT WHICH SIMULATION SHOULD START AND END.

      ELSEIF (KYCOD2 .EQ. 60) THEN
         GOTO 37

C     IF THE USER STRIKES THE F3 KEY, CALL THE SUBROUTINE GOSUB.

      ELSEIF (KYCOD2 .EQ. 61) THEN
         IER=0
         CALL GOSUB(IER)

C     IF THERE ARE ANY ERRORS FROM THE SUBROUTINE GOSUB, THEN
C     PROMPT USER FOR SETUP OPTIONS, OTHERWISE
C     RETURN TO SIMULATION OPTIONS MENU.

         IF (IER .EQ. 0) THEN
            GOTO 1110
         ELSE
            GOTO 33
         ENDIF

C     IF THE USER STRIKES THE F4 KEY, CALL THE SUBROUTINE FINI.

      ELSEIF (KYCOD2 .EQ. 62) THEN
         CALL FINI

C     DISPLAY ERROR MESSAGE IF INVALID KEY IS STRUCK.

      ELSE
         CALL CSROFF
         CALL TTOUT(0,24,1,'KEY NOT DEFINED -- Hit any key to resume',
     *   40,40,28)
         CALL INKEY(KYCOD1,KYCOD2)
         CALL CSRON
         CALL TTOUT(0,24,1,BLANK,80,80,31)
         GOTO 36
      ENDIF

C     IF USER SPECIFIES PARTIAL PROGRAM SIMULATION, PROMPT THE USER FOR
C     STARTING LINE NUMBER. READ THE USER'S INPUT. VARIABLE PART1 IS
C     SET TO THE STARTING LINE NUMBER.

37    CALL CLS(31)
      CALL TTOUT(0,10,1,'Please specify line number at which',35,35,31)
      CALL TTOUT(0,10,37,'you want to start the Simulation.',33,33,31)
      JJJ = 1
34    CALL LOCATE(0,10,71+JJJ)
```

```
      CALL INKEY(KYCOD1,KYCOD2)

C     IF THE USER TYPES THE RETURN KEY, AND THE CURSOR IS AT THE FIRST
C     CHARACTER POSITION, PROGRAM SIMULATION STARTS AT THE FIRST LINE.
C     THEN THE USER IS PROMPTED FOR THE ENDING LINE NUMBER.
C     THE FOUR CHARACTERS TYPED IN ARE STORED IN THE VARIABLES, JJ1(1),
C     JJ1(2),JJ1(3),AND JJ1(4).
C     IF A NUMBER IS STRUCK, IT IS DECODED, AND DISPLAYED ON THE SCREEN
C     AND ASSIGNED TO ONE OF THESE VARIABLES. THE INDEX, JJJ, IS THEN
C     INCREMENTED.

      IF (KYCOD1 .EQ. 13 .AND. JJJ .EQ. 1) THEN
         PART1 = 1
         GOTO 1000
      ELSEIF (KYCOD1 .EQ. 13 .AND. JJJ .EQ. 2) THEN
         PART1 = JJ1(1)
         GOTO 1000
      ELSEIF (KYCOD1 .EQ. 13 .AND. JJJ .EQ. 3) THEN
         PART1 = JJ1(1) * 10 + JJ1(2)
         GOTO 1000
      ELSEIF (KYCOD1 .EQ. 13 .AND. JJJ .EQ. 4) THEN
         PART1 = JJ1(1) * 100 + JJ1(2) * 10 + JJ1(3)
         GOTO 1000
      ELSEIF (KYCOD1 .EQ. 48) THEN
         C(JJJ) = '0'
         JJ1(JJJ) = 0
         CALL TTOUT(0,10,71+JJJ,C(JJJ),1,1,31)
      ELSEIF (KYCOD1 .EQ. 49) THEN
         C(JJJ) = '1'
         JJ1(JJJ) = 1
         CALL TTOUT(0,10,71+JJJ,C(JJJ),1,1,31)
      ELSEIF (KYCOD1 .EQ. 50) THEN
         C(JJJ) = '2'
         JJ1(JJJ) = 2
         CALL TTOUT(0,10,71+JJJ,C(JJJ),1,1,31)
      ELSEIF (KYCOD1 .EQ. 51) THEN
         C(JJJ) = '3'
         JJ1(JJJ) = 3
         CALL TTOUT(0,10,71+JJJ,C(JJJ),1,1,31)
      ELSEIF (KYCOD1 .EQ. 52) THEN
         C(JJJ) = '4'
         JJ1(JJJ) = 4
         CALL TTOUT(0,10,71+JJJ,C(JJJ),1,1,31)
      ELSEIF (KYCOD1 .EQ. 53) THEN
         C(JJJ) = '5'
         JJ1(JJJ) = 5
         CALL TTOUT(0,10,71+JJJ,C(JJJ),1,1,31)
      ELSEIF (KYCOD1 .EQ. 54) THEN
         C(JJJ) = '6'
         JJ1(JJJ) = 6
         CALL TTOUT(0,10,71+JJJ,C(JJJ),1,1,31)
      ELSEIF (KYCOD1 .EQ. 55) THEN
         C(JJJ) = '7'
         JJ1(JJJ) = 7
         CALL TTOUT(0,10,71+JJJ,C(JJJ),1,1,31)
      ELSEIF (KYCOD1 .EQ. 56) THEN
         C(JJJ) = '8'
         JJ1(JJJ) = 8
         CALL TTOUT(0,10,71+JJJ,C(JJJ),1,1,31)
      ELSEIF (KYCOD1 .EQ. 57) THEN
         C(JJJ) = '9'
         JJ1(JJJ) = 9
         CALL TTOUT(0,10,71+JJJ,C(JJJ),1,1,31)
      ELSE
         CALL CSROFF
         CALL TTOUT(0,24,1,'INVALID CHARACTER - Hit any key to resume'
     *   ,41,41,28)
```

```
         CALL INKEY(KYCOD1,KYCOD2)
         CALL TTOUT(0,24,1,BLANK,80,80,31)
         CALL CSRON
         GOTO 34
      ENDIF

      JJJ = JJJ + 1
      IF (JJJ .LE. 4) GOTO 34
      PART1 = JJ1(1) * 1000 + JJ1(2) * 100 + JJ1(3) * 10 + JJ1(4)

C     THE USER IS NOW PROMPTED FOR THE END LINE NUMBER. THE END
C     LINE NUMBER IS STORED IN THE VARIABLE PART2.

1000  CALL CLS(31)
      CALL TTOUT(0,10,1,'Please specify line number at which',35,35,31)
      CALL TTOUT(0,10,37,'you want to stop the Simulation.',32,32,31)
      JJJ = 1
35    CALL LOCATE(0,10,73+JJJ)
      CALL INKEY(KYCOD1,KYCOD2)

C     IF THE USER TYPES THE RETURN KEY, AND THE CURSOR IS AT THE FIRST
C     CHARACTER POSITION, PROGRAM SIMULATION ENDS AT THE LAST LINE.
C     THE FOUR CHARACTERS TYPED IN ARE STORED IN THE VARIABLES, JJ2(1),
C     JJ2(2),JJ2(3),AND JJ2(4).
C     IF A NUMBER IS STRUCK, IT IS DECODED, AND DISPLAYED ON THE SCREEN
C     AND ASSIGNED TO ONE OF THESE VARIABLES. THE INDEX, JJJ, IS THEN
C     INCREMENTED.

      IF (KYCOD1 .EQ. 13 .AND. JJJ .EQ. 1) THEN
         PART2 = 1000
         GOTO 1001
      ELSEIF (KYCOD1 .EQ. 13 .AND. JJJ .EQ. 2) THEN
         PART2 = JJ2(1)
         GOTO 1001
      ELSEIF (KYCOD1 .EQ. 13 .AND. JJJ .EQ. 3) THEN
         PART2 = JJ2(1) * 10 + JJ2(2)
         GOTO 1001
      ELSEIF (KYCOD1 .EQ. 13 .AND. JJJ .EQ. 4) THEN
         PART2 = JJ2(1) * 100 + JJ2(2) * 10 + JJ2(3)
         GOTO 1001
      ELSEIF (KYCOD1 .EQ. 48) THEN
         C(JJJ) = '0'
         JJ2(JJJ) = 0
         CALL TTOUT(0,10,73+JJJ,C(JJJ),1,1,31)
      ELSEIF (KYCOD1 .EQ. 49) THEN
         C(JJJ) = '1'
         JJ2(JJJ) = 1
         CALL TTOUT(0,10,73+JJJ,C(JJJ),1,1,31)
      ELSEIF (KYCOD1 .EQ. 50) THEN
         C(JJJ) = '2'
         JJ2(JJJ) = 2
         CALL TTOUT(0,10,73+JJJ,C(JJJ),1,1,31)
      ELSEIF (KYCOD1 .EQ. 51) THEN
         C(JJJ) = '3'
         JJ2(JJJ) = 3
         CALL TTOUT(0,10,73+JJJ,C(JJJ),1,1,31)
      ELSEIF (KYCOD1 .EQ. 52) THEN
         C(JJJ) = '4'
         JJ2(JJJ) = 4
         CALL TTOUT(0,10,73+JJJ,C(JJJ),1,1,31)
      ELSEIF (KYCOD1 .EQ. 53) THEN
         C(JJJ) = '5'
         JJ2(JJJ) = 5
         CALL TTOUT(0,10,73+JJJ,C(JJJ),1,1,31)
      ELSEIF (KYCOD1 .EQ. 54) THEN
         C(JJJ) = '6'
         JJ2(JJJ) = 6
```

```
         CALL TTOUT(0,10,73+JJJ,C(JJJ),1,1,31)
      ELSEIF (KYCOD1 .EQ. 55) THEN
         C(JJJ) = '7'
         JJ2(JJJ) = 7
         CALL TTOUT(0,10,73+JJJ,C(JJJ),1,1,31)
      ELSEIF (KYCOD1 .EQ. 56) THEN
         C(JJJ) = '8'
         JJ2(JJJ) = 8
         CALL TTOUT(0,10,73+JJJ,C(JJJ),1,1,31)
      ELSEIF (KYCOD1 .EQ. 57) THEN
         C(JJJ) = '9'
         JJ2(JJJ) = 9
         CALL TTOUT(0,10,73+JJJ,C(JJJ),1,1,31)
      ELSE
         CALL CSROFF
         CALL TTOUT(0,24,1,'INVALID CHARACTER - Hit any key to resume'
     *   ,41,41,28)
         CALL INKEY(KYCOD1,KYCOD2)
         CALL TTOUT(0,24,1,BLANK,80,80,31)
         CALL CSRON
         GOTO 35
      ENDIF

      JJJ = JJJ + 1
      IF (JJJ .LE. 4) GOTO 35
      PART2 = JJ2(1) * 1000 + JJ2(2) * 100 + JJ2(3) * 10 + JJ2(4)

C     IF THE END LINE NUMBER IS SMALLER THAN THE START LINE NUMBER,
C     AN ERROR MESSAGE IS DISPLAYED.

1001  IF (PART2 .LT. PART1) THEN
         CALL CLS(31)
         CALL CSROFF
         CALL TTOUT(0,24,1,'End line number smaller than',28,28,28)
         CALL TTOUT(0,24,30,'starting line number.',21,21,28)
         CALL TTOUT(0,24,52,'Press any key to continue..',27,27,28)
         CALL INKEY(KYCOD1,KYCOD2)
         GOTO 33
      ENDIF

C     THE FIRST RECORD OF THE OBJECT FILE IS READ TO DETERMINE THE
C     THE FIRST EXECUTABLE STATEMENT NUMBER AND THE LAST STATEMENT
C     NUMBER IN THE PROGRAM.

      READ (7,100,REC=1) K1,K2

C     IF THE USER WANTS PROGRAM SIMULATION TO START AT THE BEGINNING,
C     THE VARIABLE PART1 IS SET TO THE FIRST EXECUTABLE STATEMENT NUMBER.

      IF (PART1 .EQ. 1) THEN
         PART1 = K2
         GOTO 1004
      ENDIF

C     THE OBJECT FILE IS READ FOR THE LINE NUMBERS OF EACH STATEMENT.

      JJJ = 1
1002  READ (7,101,REC=JJJ) JJJJ
100   FORMAT(18X,I4,1X,I4)
101   FORMAT(6X,I4,8X,I4,1X,I4)

C     IF THE LINE NUMBER IS LESS THAN PART1, READ THE NEXT STATEMENT
C     IN THE OBJECT FILE.

      IF (JJJJ .LT. PART1) THEN
         JJJ = JJJ + 1
```

```
C     IF THE STATEMENT NUMBER IS GREATER THAN THE LAST
C     STATEMENT NUMBER, DISPLAY AN ERROR MESSAGE.

         IF (JJJ .GE. K1) THEN
            CALL CLS(31)
            CALL CSROFF
            CALL TTOUT(0,24,1,'Invalid start line number specified.',36,
     *      36,28)
            CALL TTOUT(0,24,38,'Press any key to continue....',29,29,28)
            CALL INKEY(KYCOD1,KYCOD2)
            GOTO 33
         ENDIF

         GOTO 1002

      ENDIF

C     THE PROGRAM SIMULATION STARTS AT THE STATEMENT NUMBER DENOTED BY JJJ.

      PART1 = JJJ

C     THIS SECTION CHECKS FOR THE LAST STATEMENT NUMBER.

1004  JJJ = PART1

C     IF THE USER DOES NOT SPECIFY ANY NUMBER FOR THE END LINE, PART2 IS
C     EQUAL TO 1000, AND PROGRAM SIMULATION ENDS AT THE LAST STATEMENT
C     NUMBER, K2.

      IF (PART2 .EQ. 1000) THEN
         PART2 = K1
         GOTO 1110
      ENDIF

C     THE OBJECT FILE IS READ FOR THE LINE NUMBERS, AND COMPARED TO THE
C     USER'S INPUT. IF THE LINE NUMBER IS LESS THAN PART2, THE NEXT LINE
C     IN THE OBJECT FILE IS READ.

1003  READ (7,101,REC=JJJ) JJJJ

      IF (JJJJ .LT. PART2) THEN
         JJJ = JJJ + 1

C     IF THE LAST STATEMENT NUMBER IS REACHED, AN ERROR MESSAGE IS
C     DISPLAYED.

         IF (JJJ .GE. K1) THEN
            CALL CLS(31)
            CALL CSROFF
            CALL TTOUT(0,24,1,'Invalid end line number specified.',34,
     *      34,28)
            CALL TTOUT(0,24,36,'Press any key to continue....',29,29,28)
            CALL INKEY(KYCOD1,KYCOD2)
            GOTO 33
         ENDIF

         GOTO 1003

      ENDIF

C     IF THE END STATEMENT NUMBER IS THE LAST STATEMENT OF THE PROGRAM,
C     PART2 IS SET TO THE LAST STATEMENT NUMBER, OTHERWISE, IT IS SET
C     TO THE END STATEMENT NUMBER PLUS 1.

      IF (JJJ .EQ. K1) THEN
         PART2 = JJJ
      ELSE
```

```
         PART2 = JJJ + 1
      ENDIF

C     DISPLAY MESSAGE TO THE USER.

1110  CALL CLS(31)
      CALL CSROFF
      CALL TTOUT(0,7,1,'Please move the robot arm to desired position',
     *45,45,31)
      CALL TTOUT(0,7,47,'before partial simulation begins.',33,33,31)
      CALL TTOUT(0,9,1,'The next menu will show some of the',35,35,31)
      CALL TTOUT(0,9,37,'interactive commands you are allowed',36,
     *36,31)
      CALL TTOUT(0,11,1,'to use to move the robot arm.',29,
     *29,31)
      CALL TTOUT(0,11,31,'You are allowed to use names of variables',41,
     *41,31)
      CALL TTOUT(0,13,1,'from your program to define points,',35,35,31)
      CALL TTOUT(0,13,37,'values or counters.',19,19,31)
      CALL TTOUT(0,24,1,'Press any key to continue....',29,29,28)
      CALL INKEY(KYCOD1,KYCOD2)

C     DISPLAY SETUP MENU, AND READ USER'S INPUT.

1112  CALL CLS(31)
      CALL CSRON
      CALL TTOUT(0,1,23,'SETUP FOR PARTIAL PROGRAM SIMULATION',36,36,31)
      CALL TTOUT(0,2,23,'----------------------------------',36,36,31)
      CALL TTOUT(0,5,30,'Select a function:',18,18,27)
      CALL TTOUT(0,8,33,'FN1  -  SETPART',15,15,31)
      CALL TTOUT(0,9,33,'FN2  -  GETPART',15,15,31)
      CALL TTOUT(0,10,33,'FN3  -  PMOVE',13,13,31)
      CALL TTOUT(0,11,33,'FN4  -  ZMOVE',13,13,31)
      CALL TTOUT(0,12,33,'FN5  -  WRITEO',14,14,31)
      CALL TTOUT(0,13,33,'FN6  -  SETC',12,12,31)
      CALL TTOUT(0,14,33,'FN7 - Exit from setup menu',26,26,31)
      CALL TTOUT(0,20,30,'Enter option===>',16,16,27)
1111  CALL LOCATE(0,20,47)
      CALL INKEY(KYCOD1,KYCOD2)

C     DISPLAY ERROR MESSAGE IF USER STRIKES AN INVALID KEY.

      IF (KYCOD1 .NE. 0 .OR. KYCOD2 .LT. 59 .OR. KYCOD2 .GT. 65) THEN
         CALL CSROFF
         CALL TTOUT(0,24,1,'Key not defined -- Hit any key to resume',
     *   40,40,28)
         CALL INKEY(KYCOD1,KYCOD2)
         CALL CSRON
         CALL TTOUT(0,24,1,BLANK,80,80,31)
         GOTO 1111
      ENDIF

C     IF USER STRIKES F7 KEY, EXIT FROM SETUP MENU, OTHERWISE CALL
C     SUBROUTINE PARTN.

      IF (KYCOD2 .EQ. 65) GOTO 2000

      CALL PARTN(KYCOD2)

      GOTO 1112

2000  RETURN
      END
C     SUBROUTINE PARTN

C     THIS SUBROUTINE EXECUTES COMMANDS INTERACTIVELY, SO THAT THE
C     ROBOT ARM IS SETUP PRIOR TO PARTIAL PROGRAM SIMULATION.
```

```
C     PARAMETER DEFINITION:

C          KCOD2     -       FUNCTION KEY SELECTED BY USER


C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C             CHECK,CLS,CSROFF,CSRON,INKEY,LOCATE,PARTM,TTOUT,
C             USER,WRITEP,WRITER.

      SUBROUTINE PARTN(KCOD2)

      INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
      INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100)
      INTEGER SS,SLPAY,PART1,PART2,VAL1,VAL2,RETVAL,BASC
      INTEGER * 2 KYCOD1,KYCOD2,KCOD2,MM,LL
      INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      INTEGER P29,P30,P31,P32
      REAL THONI,THTWI,ZIN,RIN,CODE,VALU1,VALU2,VAL3,VAL4
      REAL XVAL,YVAL,ZVAL,RVAL
      CHARACTER * 1 BLANK(80),CELNAM*10
      CHARACTER * 72 CHAR,CNT

      DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
      DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
      DIMENSION XPNT(100),YPNT(100),ZPNT(100),RPNT(100)
      DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
      DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)

      COMMON/A/COUNT,CNTVAL,PALLET,PALPTS,PARTNO
      COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
      COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
      COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
      COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
      COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
      COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

      DATA BLANK/80 * ' '/

      VALU1 = 0.0
      VALU2 = 0.0
      VAL3 = 0.0
      VAL4 = 0.0

C     OPEN A TEMPORARY FILE TO WRITE AND READ THE USER'S INPUT.

      OPEN (9,FILE='IRPS01',STATUS='NEW')
      CALL CLS(31)
      CALL CSRON

C     SETPART

C     IF THE USER HAD STRUCK F1 KEY, PROMPT THE USER FOR THE PALLET
C     NAME. READ THE INPUT, AND WRITE THE NAME INTO THE FILE IRPS01.
C     COMPARE THIS NAME TO THE VARIABLE NAMES IN THE VARIABLES FILE.
C     IF NAME DOES NOT EXIST, DISPLAY ERROR MESSAGE. IF PALLET NAME
C     HAS BEEN FOUND, FIND THE INDEX NUMBER FOR THE ARRAY "PALLET".

      IF (KCOD2 .EQ. 59) THEN
198      CALL TTOUT(0,10,37,'SETPART',7,7,31)
         CALL TTOUT(0,11,37,'-------',7,7,31)
```

```
         CALL TTOUT(0,14,25,'Pallet name  = ',15,15,31)
         CALL PARTM(1,14,40,CHAR,LL,2)
         WRITE (9,100) LL,(CHAR(MM:MM),MM=1,LL)
         REWIND (9)
199      READ (3,300,END=201) IC2,NNN,ICODE,CNT
         IF (ICODE .NE. 5) GOTO 199
         IF (CNT(1:NNN) .NE. CHAR(1:LL)) GOTO 199
         DO 203 I = 1,NUM6 - 1
            IF (IC2 .EQ. PALLET(I)) THEN
               L5 = I
               GOTO 202
            ENDIF
203      CONTINUE

201      CALL CSROFF
         CALL TTOUT(0,24,1,'Pallet undefined -- Hit any key to resume',
     *   41,41,28)
         CALL INKEY(KYCOD1,KYCOD2)
         REWIND (3)
         GOTO 1000

C     PROMPT THE USER FOR THE PALLET VALUE. READ THE VALUE, AND CHECK
C     THAT THE VALUE LIES BETWEEN 1 AND THE MAXIMUM NUMBER OF POINTS
C     FOR THE PALLET. IF NOT, DISPLAY ERROR MESSAGE. IF THE NUMBER IS
C     IS VALID, SET THE VALUE OF THE ARRAY PARTNO EQUAL TO THE VALUE,
C     AND THE Z VALUE OF THE PALLET POINT TO BE EQUAL TO THE CURRENT
C     Z VALUE OF THE ROBOT ARM.

202      CALL TTOUT(0,15,25,'Pallet value = ',15,15,31)
         CALL PARTM(1,15,40,CHAR,LL,1)
         WRITE (9,100) LL,(CHAR(MM:MM),MM=1,LL)
         REWIND (9)
         READ (9,200) VALU1
         REWIND (9)
         IF (VALU1 .GT. PALPTS(L5) .OR. VALU1 .LT. 1.0) THEN
            PARTNO(L5) = 1
            CALL USER(33)
            REWIND (3)
            GOTO 1000
         ELSE
            PARTNO(L5) = VALU1
            ZPAL(L5,PARTNO(L5)) = ZVAL
         ENDIF

         REWIND (3)

C     GETPART

C     IF THE USER HAD STRUCK THE F2 KEY, PROMPT THE USER FOR THE
C     PALLET NAME. CHECK THAT PALLET NAME IS VALID, AS DESCRIBED
C     ABOVE.

      ELSEIF (KCOD2 .EQ. 60) THEN
498      CALL TTOUT(0,10,37,'GETPART',7,7,31)
         CALL TTOUT(0,11,37,'-------',7,7,31)
         CALL TTOUT(0,14,25,'Pallet name  = ',15,15,31)
         CALL PARTM(2,14,40,CHAR,LL,2)
         WRITE (9,100) LL,(CHAR(MM:MM),MM=1,LL)
         REWIND (9)
499      READ (3,300,END=501) IC2,NNN,ICODE,CNT
         IF (ICODE .NE. 5) GOTO 499
         IF (CNT(1:NNN) .NE. CHAR(1:LL)) GOTO 499
         DO 503 I = 1,NUM6 - 1
            IF (IC2 .EQ. PALLET(I)) THEN
               L5 = I
               GOTO 502
```

```
            ENDIF
503         CONTINUE

501         CALL CSROFF
            CALL TTOUT(0,24,1,'Pallet undefined -- Hit any key to resume',
     *      41,41,28)
            CALL INKEY(KYCOD1,KYCOD2)
            REWIND (3)
            GOTO 1000

C       SET THE CURRENT VALUES OF THE ROBOT ARM POSITION TO THE COORDINATES
C       OF THE POINT AS DEFINED BY THE PALLET POSITION.

502         XVAL = XPAL(L5,PARTNO(L5))
            YVAL = YPAL(L5,PARTNO(L5))
            RVAL = RPAL(L5,PARTNO(L5))

C       THE COORDINATES ARE PASSED TO THE GRAPHICS DRIVERS


            CODE = 1.0
         CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)

         REWIND (3)
         CALL DSWAP(0)

         CALL TTOUT(0,11,1,'Do You Want to Redraw the Screen (Y/N)',
        *38,38,31)
         READ(*,123) ANS
         IF (ANS .EQ. 'Y') THEN
         CODE = 8.0
         CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)
         CALL DSWAP(0)
         ENDIF
 123     FORMAT(A1)



C       PMOVE

C       IF THE USER STRUCK THE F3 KEY, ASK THE USER WHETHER THE COMMAND
C       PARAMETER IS A VARIABLE NAME OR A SET OF NUMBERS. IF THE PARAMETER
C       IS A VARIABLE NAME, PROMPT THE USER FOR THE NAME.

         ELSEIF (KCOD2 .EQ. 61) THEN
598         CALL TTOUT(0,10,38,'PMOVE',5,5,31)
            CALL TTOUT(0,11,38,'-----',5,5,31)
            CALL TTOUT(0,14,20,'Is the point a variable name (Y/N) ?',36,
     *      36,31)
            CALL LOCATE(0,14,57)
            CALL INKEY(KYCOD1,KYCOD2)
            IF (KYCOD1 .NE. 89 .AND. KYCOD1 .NE. 121) THEN
               CALL TTOUT(0,14,1,BLANK,80,80,31)
               GOTO 507
            ENDIF

C       IF THE USER WANTS TO PASS A VARIABLE NAME, READ THE INPUT
C       AND CHECK FOR THE VALIDITY OF THE NAME AS BEFORE.

            CALL TTOUT(0,14,1,BLANK,80,80,31)
            CALL TTOUT(0,14,25,'Point name = ',13,13,31)
            CALL PARTM(3,14,38,CHAR,LL,2)
            WRITE (9,100) LL,(CHAR(MM:MM),MM=1,LL)
            REWIND (9)
599         READ (3,300,END=601) IC2,NNN,ICODE,CNT
            IF (ICODE .NE. 3) GOTO 599
            IF (CNT(1:NNN) .NE. CHAR(1:LL)) GOTO 599
```

```
            DO 603 I = 1,NUM2 - 1
               IF (IC2 .EQ. PNTVAL(I)) THEN
                  L5 = I
                  GOTO 602
               ENDIF
603         CONTINUE

601         CALL CSROFF
            CALL TTOUT(0,24,1,'Point undefined -- Hit any key to resume',
          *  40,40,28)
            CALL INKEY(KYCOD1,KYCOD2)
            REWIND (3)
            GOTO 1000

C       ASSIGN THE VALUES OF THE POINT TO THE FOUR VARIABLES, VALU1,
C       VALU2, VAL3, AND VAL4.

602         VALU1 = XPNT(L5)
            VALU2 = YPNT(L5)
            VAL3 = ZPNT(L5)
            VAL4 = RPNT(L5)

            GOTO 607

C       IF THE USER WANTS TO PASS NUMBERS AS THE PARAMETER, PROMPT THE
C       USER FOR THE X,Y,Z, AND R VALUES, ONE AT A TIME. READ THE USER'S
C       INPUT FOR EACH VALUE.

507         CALL TTOUT(0,14,25,'X value = ',10,10,31)
            CALL PARTM(3,14,35,CHAR,LL,1)
            WRITE (9,100) LL,(CHAR(MM:MM),MM=1,LL)
            REWIND (9)
            IF (LL .LE. 1) GOTO 1
            READ (9,200) VALU1
            REWIND (9)
1           CALL TTOUT(0,15,25,'Y value = ',10,10,31)
            CALL PARTM(3,15,35,CHAR,LL,1)
            WRITE (9,100) LL,(CHAR(MM:MM),MM=1,LL)
            REWIND (9)
            IF (LL .LE. 1) GOTO 2
            READ (9,200) VALU2
            REWIND (9)
2           CALL TTOUT(0,16,25,'Z value = ',10,10,31)
            CALL PARTM(3,16,35,CHAR,LL,1)
            WRITE (9,100) LL,(CHAR(MM:MM),MM=1,LL)
            REWIND (9)
            IF (LL .LE. 1) GOTO 3
            READ (9,200) VAL3
            REWIND (9)
3           CALL TTOUT(0,17,25,'R value = ',10,10,31)
            CALL PARTM(3,17,35,CHAR,LL,1)
            WRITE (9,100) LL,(CHAR(MM:MM),MM=1,LL)
            REWIND (9)
            IF (LL .LE. 1) GOTO 607
            READ (9,200) VAL4
            REWIND (9)

C       CHECK FOR THE VALIDITY OF THE R AND Z VALUES. IF THEY ARE
C       INVALID, DISPLAY AN ERROR MESSAGE.

607         IF (VAL4 .LT. -180.0 .OR. VAL4 .GT. 180.0) THEN
               CALL USER(22)
               REWIND (3)
               GOTO 1000
            ENDIF

            IF (VAL3 .LT. -250.0 .OR. VAL3 .GT. 0.0) THEN
```

```
            CALL USER(23)
            REWIND (3)
            GOTO 1000
         ENDIF

         IF (VALU1 .LT. -650.0 .OR. VALU1 .GT.650.0) THEN
            CALL USER(24)
            REWIND (3)
            GOTO 1000
         ENDIF

         IF (VALU2 .LT. -387.0 .OR. VAL3 .GT. 650.0) THEN
            CALL USER(25)
            REWIND (3)
            GOTO 1000
         ENDIF


C     ASSIGN THE CURRENT POSITION OF THE ROBOT ARM TO THE VALUES
C     VALU1, VALU2, VAL3 AND VAL4. THESE VALUES ARE PASSED TO THE
C     GRAPHICS DRIVERS

         XVAL = VALU1
         YVAL = VALU2
         ZVAL = VAL3
         RVAL = VAL4

          CODE = 1.0
          ZVAL = ZVAL + 250.0
      CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)

         REWIND (3)

      CALL DSWAP(0)

      CALL TTOUT(0,11,1,'Do You Want to Redraw the Screen (Y/N)',
     *38,38,31)
      READ(*,123) ANS
      IF (ANS .EQ. 'Y') THEN
      CODE = 8.0
      CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)
      CALL DSWAP(0)
      ENDIF


C     ZMOVE

C     IF THE USER STRUCK THE F4 KEY, PROMT THE USE FOR THE Z VALUE.

      ELSEIF (KCOD2 .EQ. 62) THEN
         CALL TTOUT(0,10,38,'ZMOVE',5,5,31)
         CALL TTOUT(0,11,38,'-----',5,5,31)
         CALL TTOUT(0,14,25,'Z value = ',10,10,31)
         CALL PARTM(4,14,35,CHAR,LL,1)
         WRITE (9,100) LL,(CHAR(MM:MM),MM=1,LL)
         REWIND (9)
         READ (9,200) VALU1
         REWIND (9)

C     CHECK THAT THE Z VALUE IS VALID, AND IF SO, SET THE CURRENT
C     Z VALUE, ZVAL, TO THE USER'S INPUT. THESE ARE PASSED TO THE
C     GRAPHICS DRIVERS.

         IF (VALU1 .LT. -250.0 .OR. VALU1 .GT. 0.0) THEN
            CALL USER(23)
            GOTO 1000
         ENDIF
```

```
         ZVAL = VALU1

           CODE = 1.0
          ZVAL = ZVAL + 250.0
       CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)
       CALL DSWAP(0)


       CALL TTOUT(0,11,1,'Do You Want to Redraw the Screen (Y/N)',
      *38,38,31)
       READ(*,123) ANS
       IF (ANS .EQ. 'Y') THEN
       CODE = 8.0
       CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)
       CALL DSWAP(0)
       ENDIF


C      WRITE0

C      IF THE USER STRUCK THE F5 KEY, PROMPT THE USER FOR THE DO
C      NUMBER. THE INPUT MUST BE FROM 1 THROUGH 16, OTHERWISE AN
C      ERROR MESSAGE IS DISPLAYED.

       ELSEIF (KCOD2 .EQ. 63) THEN
          CALL TTOUT(0,10,37,'WRITE0',6,6,31)
          CALL TTOUT(0,11,37,'------',6,6,31)
          CALL TTOUT(0,14,25,'DO number = ',12,12,31)
          CALL PARTM(5,14,37,CHAR,LL,1)
          WRITE (9,100) LL,(CHAR(MM:MM),MM=1,LL)
          REWIND (9)
          READ (9,200) VALU1
          REWIND (9)

          IF (VALU1 .LT. 1.0 .OR. VALU1 .GT. 16.0) THEN
             CALL USER(30)
             GOTO 1000
          ENDIF

          VAL1 = VALU1 + 16


C      THEN PROMPT THE USER FOR THE VALUE, WHICH MUST BE A ZERO OR
C      A ONE.

          CALL TTOUT(0,15,25,'DO value  = ',12,12,31)
          CALL PARTM(5,15,37,CHAR,LL,1)
          WRITE (9,100) LL,(CHAR(MM:MM),MM=1,LL)
          REWIND (9)
          READ (9,200) VALU2
          REWIND (9)

          IF (VALU2 .NE. 0.0 .AND. VALU2 .NE. 1.0) THEN
             CALL USER(28)
             GOTO 1000
          ENDIF

          VAL2 = VALU2

          IF (VAL1 .EQ. 17) THEN
             IF (VAL2 .EQ. 1) THEN
               P17 = 1
             ELSE
               P17 = 0
             ENDIF
          ELSEIF (VAL1 .EQ. 18) THEN
```

```
   IF (VAL2 .EQ. 1)   THEN
     P18 = 1
   ELSE
     P18 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 19) THEN
   IF (VAL2 .EQ. 1)   THEN
     P19 = 1
   ELSE
     P19 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 20) THEN
   IF (VAL2 .EQ. 1)   THEN
     P20 = 1
   ELSE
     P20 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 21) THEN
   IF (VAL2 .EQ. 1)   THEN
     P21 = 1
   ELSE
     P21 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 22) THEN
   IF (VAL2 .EQ. 1)   THEN
     P22 = 1
   ELSE
     P22 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 23) THEN
   IF (VAL2 .EQ. 1)   THEN
     P23 = 1
   ELSE
     P23 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 24) THEN
   IF (VAL2 .EQ. 1)   THEN
     P24 = 1
   ELSE
     P24 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 25) THEN
   IF (VAL2 .EQ. 1)   THEN
     P25 = 1
   ELSE
     P25 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 26) THEN
   IF (VAL2 .EQ. 1)   THEN
     P26 = 1
   ELSE
     P26 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 27) THEN
   IF (VAL2 .EQ. 1)   THEN
     P27 = 1
   ELSE
     P27 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 28) THEN
   IF (VAL2 .EQ. 1)   THEN
     P28 = 1
   ELSE
     P28 = 0
   ENDIF
ELSEIF (VAL1 .EQ. 29) THEN
   IF (VAL2 .EQ. 1)   THEN
```

```
            P29 = 1
          ELSE
            P29 = 0
          ENDIF
       ELSEIF (VAL1 .EQ. 30) THEN
          IF (VAL2 .EQ. 1)   THEN
            P30 = 1
          ELSE
            P30 = 0
          ENDIF
       ELSEIF (VAL1 .EQ. 31) THEN
          IF (VAL2 .EQ. 1)   THEN
            P31 = 1
          ELSE
            P31 = 0
          ENDIF
       ELSEIF (VAL1 .EQ. 32) THEN
          IF (VAL2 .EQ. 1)   THEN
            P32 = 1
          ELSE
            P32 = 0
          ENDIF
       ENDIF


C      THE DO NUMBER, AND THE VALUE ARE TRANSMITTED TO THE
C      GRAPHICS DRIVER.

          CODE = 3.0
       CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)

       CALL DSWAP(0)

       CALL TTOUT(0,11,1,'Do You Want to Redraw the Screen (Y/N)',
      *38,38,31)
       READ(*,123) ANS
       IF (ANS .EQ. 'Y') THEN
       CODE = 5.0
       CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)
       CALL DSWAP(0)
       ENDIF


C      SETC

C      IF THE USER STRUCK THE F6 KEY, THE USER IS PROMPTED FOR THE
C      COUNTER NAME. THE NAME IS CHECKED FOR ITS EXISTENCE, AS
C      BEFORE.

       ELSEIF (KCOD2 .EQ. 64) THEN
298       CALL TTOUT(0,10,39,'SETC',4,4,31)
          CALL TTOUT(0,11,39,'----',4,4,31)
          CALL TTOUT(0,14,25,'Counter name  = ',16,16,31)
          CALL PARTM(6,14,41,CHAR,LL,2)
          WRITE (9,100) LL,(CHAR(MM:MM),MM=1,LL)
          REWIND (9)
299       READ (3,300,END=301) IC2,NNN,ICODE,CNT
          IF (ICODE .NE. 4) GOTO 299
          IF (CNT(1:NNN) .NE. CHAR(1:LL)) GOTO 299

          DO 303 I = 1,NUM3 - 1

             IF (IC2 .EQ. COUNT(I)) THEN
                L5 = I
                GOTO 302
             ENDIF
```

```
303       CONTINUE

301       CALL CSROFF
          CALL TTOUT(0,24,1,'Counter undefined -- Hit any key to resume',
     *    42,42,28)
          CALL INKEY(KYCOD1,KYCOD2)
          REWIND (3)
          GOTO 1000

C      THE USER IS THEN PROMPTED FOR THE VALUE. IF THE VALUE IS OUT OF
C      THE VALID RANGE, AN ERROR MESSAGE IS DISPLAYED.

302       CALL TTOUT(0,15,25,'Counter value = ',16,16,31)
          CALL PARTM(6,15,41,CHAR,LL,1)
          WRITE (9,100) LL,(CHAR(MM:MM),MM=1,LL)
          REWIND (9)
          READ (9,200) VALU1
          REWIND (9)

          IF (VALU1 .LT. -32767.0 .OR. VALU1 .GT. 32767.0) THEN
             CALL USER(31)
             REWIND (3)
             GOTO 1000
          ENDIF

C      THE COUNTER VALUE IS THEN SET TO THE VALUE KEYED IN BY THE USER.
C      THIS VALUE IS ALSO WRITTEN TO THE SYMBOL TABLE FILE.

          CNTVAL(L5) = VALU1
          WRITE(8,400,REC=IC2) IC2,VALU1
          REWIND (3)

       ENDIF

1000   CLOSE(9,STATUS='DELETE')

100    FORMAT(I2,1X,72A1)
200    FORMAT(3X,F8.2)
300    FORMAT(1X,I3,13X,I3,2X,I1,6X,A72)
400    FORMAT(1X,I3,2X,F8.2)

       RETURN
       END



C      SUBROUTINE PARTM

C      SUBROUTINE TO IDENTIFY CHARACTERS TYPED IN FOR VARIABLES PRIOR
C      TO PARTIAL EXECUTION

C      PARAMETER DEFINITION:

C           KCODE     -      CODE PASSED DEPENDING ON COMMAND TYPE
C           ROW       -      ROW NUMBER
C           COLM      -      COLUMN NUMBER
C           CHAR      -      CHARACTER TYPED BY USER
C           LL        -      INCREMNTAL COLUMN NUMBER
C           MM        -      INDICATES WHETHER INPUT SHOULD BE
C                            NUMERIC (VALUE IS 1) OR ALPHANUMERIC (VALUE IS 2)


C      SUBROUTINES CALLED BY THIS PROGRAM ARE :
C              CSROFF,CSRON,ID,INKEY,LOCATE,TTOUT.

       SUBROUTINE PARTM(KCODE,ROW,COLM,CHAR,LL,MM)
```

```
      INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
      INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100)
      INTEGER SS,SLPAY,PART1,PART2,VAL1,VAL2,RETVAL,BASC
      INTEGER * 2 KYCOD1,KYCOD2,J
      INTEGER ROW,COLM
      REAL THONI,THTWI,ZIN,RIN,CODE
      CHARACTER * 1 BLANK(80),CELNAM*10,AFILE*72,AEXT*1
      CHARACTER * 1 CHAR1,ALPHA(26),NUMER(10)
      CHARACTER * 72 CHAR

      DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
      DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
      DIMENSION XPNT(100),YPNT(100),ZPNT(100),RPNT(100)
      DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
      DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)

      COMMON/A/COUNT,CNTVAL,PALLET,PALPTS,PARTNO
      COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
      COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
      COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
      COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
      COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
      COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX

      DATA BLANK/80 * ' '/

      DATA ALPHA/'A','B','C','D','E','F','G','H','I','J','K','L','M',
     *'N','O','P','Q','R','S','T','U','V','W','X','Y','Z'/
      DATA NUMER/'0','1','2','3','4','5','6','7','8','9'/

1     DO 10 MMM = 1,72
         CHAR(MMM:MMM) = ' '
10    CONTINUE

C     LOCATE THE CURSOR AND READ USER'S INPUT AND CHECK FOR VALID KEYS.

      LL = 1
5     CALL LOCATE(0,ROW,COLM+LL)
      J = LL
      CALL INKEY(KYCOD1,KYCOD2)

      IF (KYCOD1 .EQ. 13) THEN
         GOTO 20
      ELSE
         CALL ID(KYCOD1,KYCOD2,CHAR1,J,ALPHA,NUMER)

C     IF USER'S INPUT IS INVLAID, DISPLAY ERROR MESSAGE.

         IF (J .EQ. 0) THEN
            CALL CSROFF
           CALL TTOUT(0,24,1,'INVALID CHARACTER - Hit any key to resume'
     *       ,41,41,28)
            CALL INKEY(KYCOD1,KYCOD2)
            CALL TTOUT(0,24,1,BLANK,80,80,31)
            CALL CSRON
            GOTO 5
         ENDIF

C     IF THE USER STRIKES THE BACKSPACE KEY, ERASE THE PREVIOUS
C     CHARACTER FROM THE SCREEN, OTHERWISE DISPLAY THE CHARACTER
C     THE USER TYPES.

         IF (J .LT. LL) THEN
            LL = J
            CALL TTOUT(0,ROW,COLM+J,BLANK(1),1,1,31)
```

```
            GOTO 5
         ELSE
            CALL TTOUT(0,ROW,COLM+LL,CHAR1,1,1,26)

C     CONCATENATE THE STRING OF CHARACTERS TYPED IN BY THE USER, AND
C     INCREMENT THE CURSOR POSITION.

            IF (J .NE. 1) THEN
CCC            CHAR(1:LL) = CHAR(1:LL-1)//CHAR1

               AFILE =  CHAR(1:LL-1)
               AEXT  =  CHAR1
               CALL CONCAD (AFILE,AEXT)
               CHAR(1:LL)  = AFILE
            ELSE
               CHAR(1:LL) = CHAR1
            ENDIF

            LL = LL + 1
            GOTO 5

         ENDIF

      ENDIF

C     IF THE INPUT IS NUMERIC, CHECK FOR A DECIMAL POINT. IF THERE
C     IS NO DECIMAL POINT, ADD ONE TO THE STRING OF CHARACTERS.

20    IF (MM .EQ. 1) THEN
         LL = LL - 1

         DO 30 MMM = 1,LL
            IF (CHAR(MMM:MMM) .EQ. '.') GOTO 40
30       CONTINUE

         LL = LL + 1
         CHAR(LL:LL) = '.'

C     CHECK EACH CHARACTER TO BE A NUMBER, OR A DECIMAL POINT, OR
C     THE MINUS SIGN. IF ANY CHARACTER IS NOT ONE OF THESE, THE
C     NUMBER IS INVALID, AND THE ERROR CODE, IER, IS SET TO 1.

40       IER = 0

         DO 43 MMM = 1,LL

            DO 44 MMMM = 1,10
               IF (CHAR(MMM:MMM) .EQ. NUMER(MMMM)) GOTO 43
               IF (CHAR(MMM:MMM) .EQ. '-' .AND. MMM .EQ. 1) GOTO 43
               IF (CHAR(MMM:MMM) .EQ. '.') GOTO 43
44          CONTINUE

            IER = 1
43       CONTINUE

         ISET = 0

C     IF THERE IS MORE THAN ONE DECIMAL POINT, IER IS SET TO 1.

         DO 59 MMM = 1,LL
            IF (CHAR(MMM:MMM) .EQ. '.') ISET = ISET + 1
            IF (ISET .GT. 1) IER = 1
59       CONTINUE

C     IF THE LENGTH OF CHARACTERS IN THE NUMBER IS GREATER THAN 8,
C     OR IF IER IS 1, DISPLAY ERROR MESSAGE. THEN BLANK OUT THE
C     THE USER'S INPUT, AND PROMPT AGAIN.
```

```
         IF (LL .GT. 8 .OR. IER .EQ. 1 ) THEN
            CALL CSROFF
            CALL TTOUT(0,24,1,'Invalid number -- Hit any key to resume',
     *      39,39,28)
            CALL INKEY(KYCOD1,KYCOD2)
            CALL TTOUT(0,24,1,BLANK,80,80,31)

            DO 52 MMM = 1,LL
               CALL TTOUT(0,ROW,COLM+MMM,' ',1,1,31)
52          CONTINUE

            IER = 0
            CALL CSRON
            GOTO 1
         ENDIF

C     IF THE INPUT IS ALPHANUMERIC, CHECK THAT THE FIRST CHARACTER
C     IS ALPHABETIC. IF NOT, DISPLAY ERROR MESSAGE AND PROMPT USER AGAIN.

      ELSE
         IER = 0
         LL = LL - 1

         DO 45 MMM = 1,26
            IF (CHAR(1:1) .EQ. ALPHA(MMM)) GOTO 46
45       CONTINUE

48       CALL CSROFF
         CALL TTOUT(0,24,1,'Invalid name -- Hit any key to resume',
     *   37,37,28)
         CALL INKEY(KYCOD1,KYCOD2)
         CALL TTOUT(0,24,1,BLANK,80,80,31)

         DO 53 MMM = 1,LL
            CALL TTOUT(0,ROW,COLM+MMM,' ',1,1,31)
53       CONTINUE

         IER = 0
         CALL CSRON
         GOTO 1

C     IF ANY CHARACTER IS THE DECIMAL POINT OR THE MINUS SIGN,
C     DISPLAY ERROR MESSAGE.

46       DO 47 MMM = 1,LL
            IF (CHAR(MMM:MMM) .EQ. '.') GOTO 48
            IF (CHAR(MMM:MMM) .EQ. '-') GOTO 48
47       CONTINUE

      ENDIF

      RETURN
      END

C     SUBROUTINE ID

C     THIS SUBROUTINE IDENTIFIES THE CHARACTER TYPED IN BY THE USER.
C     VALID CHARACTERS ARE ALL ALPHABETS, NUMBERS, THE BACKSPACE KEY,
C     THE ENTER KEY AND SOME SPECIAL CHARACTERS.

C     PARAMETER DEFINITION:

C          KYCOD1    -      ASCII CODE OF KEY
C          KYCOD2    -      EXTENDED CHARACTER CODE OF KEY
C          CHAR      -      CHARACTER IDENTIFIED
C          J         -      COLUMN POSITION
```

```
C          ALPHA     -       ARRAY OF ALPHABETS
C          NUMMER    -       ARRAY OF NUMBERS

      SUBROUTINE ID(KYCOD1,KYCOD2,CHAR1,J,ALPHA,NUMER)

      INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
      INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100)
      INTEGER SS,SLPAY,PART1,PART2,VAL1,VAL2,RETVAL,BASC
      INTEGER * 2 KYCOD1,KYCOD2,J
      REAL THONI,THTWI,ZIN,RIN,CODE
      CHARACTER * 1 CHAR1,ALPHA(26),NUMER(10),CELNAM*10

      DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
      DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
      DIMENSION XPNT(100),YPNT(100),ZPNT(100),RPNT(100)
      DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
      DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)

      COMMON/A/COUNT,CNTVAL,PALLET,PALPTS,PARTNO
      COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
      COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
      COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
      COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
      COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
      COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX


      IF (KYCOD1 .GE. 48 .AND. KYCOD1 .LE. 57) THEN
         CHAR1 = NUMER(KYCOD1-47)
      ELSEIF (KYCOD1 .GE. 65 .AND. KYCOD1 .LE. 90) THEN
         CHAR1 = ALPHA(KYCOD1-64)
      ELSEIF (KYCOD1 .GE. 97 .AND. KYCOD1 .LE. 122) THEN
         CHAR1 = ALPHA(KYCOD1-96)
      ELSEIF (KYCOD1 .EQ. 29) THEN
         J = J - 1
      ELSEIF (KYCOD1 .EQ. 8 .AND. KYCOD2 .EQ. 14) THEN
         J = J - 1
      ELSEIF (KYCOD1 .EQ. 127 .AND. KYCOD2 .EQ. 14) THEN
         J = J - 1
      ELSEIF (KYCOD1 .EQ. 0 .AND. KYCOD2 .EQ. 75) THEN
         J = J - 1
      ELSEIF (KYCOD1 .EQ. 45) THEN
         CHAR1 = '-'
      ELSEIF (KYCOD1 .EQ. 46) THEN
         CHAR1 = '.'
      ELSEIF (KYCOD1 .EQ. 95) THEN
         CHAR1 = '_'
      ELSE
         J = 0
      ENDIF

      RETURN
      END

C     SUBROUTINE GOSUB

C     THIS SUBROUTINE INTERACTIVELY EXECUTES A SUBROUTINE.

C     PARAMETER DEFINITION:

C          IER       -       ERROR CODE DUE TO USER INPUT
C                            0 WHEN NO ERROR DETECTED
C                            1 WHEN THERE IS AN ERROR
```

```
C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C             CLS,CSROFF,CSRON,LOCATE,INKEY,PARTM,TTOUT.

      SUBROUTINE GOSUB(IER)

      INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
      INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100)
      INTEGER SS,SLPAY,PART1,PART2,VAL1,VAL2,RETVAL,BASC
      INTEGER * 2 KYCOD1,KYCOD2
      INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      INTEGER P29,P30,P31,P32

      REAL THONI,THTWI,ZIN,RIN,CODE
      CHARACTER * 1 BLANK(80),C,CELNAM*10
      CHARACTER * 72 CHAR,CNT,CNT1(5)

      DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
      DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
      DIMENSION XPNT(100),YPNT(100),ZPNT(100),RPNT(100)
      DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
      DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)
      DIMENSION NNN1(5),IC21(5),VAL(4)

      COMMON/A/COUNT,CNTVAL,PALLET,PALPTS,PARTNO
      COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
      COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
      COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
      COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
      COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
      COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32


      DATA BLANK/80 * ' '/

      IER = 0
      OPEN (39,FILE='IRPS02',STATUS='NEW')
      CALL CLS(31)
      CALL CSRON

      DO 597 I = 1,5
         IC21(I) = 0
         NNN1(I) = 0
597   CONTINUE

C     PROMT THE USER FOR THE SUBROUTINE NAME. READ THE CHARACTERS
C     KEYED IN AND WRITE TO THE TEMPORARY FILE, IRPS00, AND READ
C     THE VARIABLES FILE AND COMPARE ANY SUBROUTINE NAME WITH
C     THE NAME TYPED IN BY THE USER.

598   CALL TTOUT(0,3,31,'SUBROUTINE SIMULATION',21,21,31)
      CALL TTOUT(0,4,31,'----------------------',21,21,31)
      CALL TTOUT(0,7,22,'Subroutine name = ',18,18,31)
      CALL PARTM(0,7,40,CHAR,LL,2)
      WRITE (39,100) LL,(CHAR(MM:MM),MM=1,LL)
      REWIND (39)

599   READ (3,300,END=601) IC2,IC3,NNN,ICODE,CNT
      IF (ICODE .NE. 6) GOTO 599
      IF (CNT(1:NNN) .NE. CHAR(1:LL)) GOTO 599
      GOTO 605
```

```
C     THE SUBROUTINE NAME DOES NOT MATCH ANY NAMES IN THE PROGRAM.
C     DISPLAY AN ERROR MESSAGE AND EXIT FROM THIS SUBROUTINE.

601   CALL CSROFF
      CALL TTOUT(0,24,1,'Name undefined -- Hit any key to resume',
     *39,39,28)
      CALL INKEY(KYCOD1,KYCOD2)
      REWIND (3)
      CALL CSRON
      IER = 1
      GOTO 1000

C     READ THE OBJECT FILE FOR THE FIRST AND LAST EXECUTABLE
C     STATEMENT NUMBERS FOR THIS SUBROUTINE AND ASSIGN THESE
C     STATEMENT NUMBERS TO VARIABLES PART1 AND PART2.

605   READ (7,400,REC=IC3) K1,K2
      PART1 = K2
      PART2 = K1

C     READ THE VARIABLES FILE TO CHECK FOR ANY FORMAL PARAMETERS,
C     AND IF THERE ARE ANY, ASSIGN THE VARIABLE NAMES TO THE
C     ARRAY CNT1. VARIABLE IC21(I) IS SET TO ZERO, IF THERE ARE
C     ONLY I-1 FORMAL PARAMETERS.

      DO 700 I = 1,5
606      READ (3,300,END=701) IC21(I),IC3,NNN1(I),ICODE,CNT1(I)
         IF (ICODE .EQ. 7) GOTO 700
         IC21(I) = 0
         GOTO 701
700   CONTINUE

701   REWIND(3)

C     PROMPT THE USER FOR VALUES FOR ANY FORMAL PARAMETERS.
C     THE USER CAN EITHER PASS A VARIABLE NAME TO THE PARAMETER,
C     OR CAN PASS A SET OF VALUES.

      DO 702 I = 1,5

         IF (IC21(I) .NE. 0) THEN
            CALL TTOUT(0,7,1,BLANK,80,80,31)
            CALL TTOUT(0,7,1,'Formal parameter :',18,18,31)
            CALL TTOUT(0,7,20,CNT1(I)(1:NNN1(I)),NNN1(I),NNN1(I),28)
            CALL TTOUT(0,9,1,BLANK,80,80,31)
            CALL TTOUT(0,10,1,BLANK,80,80,31)
            CALL TTOUT(0,12,1,BLANK,80,80,31)
            CALL TTOUT(0,13,1,BLANK,80,80,31)
            CALL TTOUT(0,14,1,BLANK,80,80,31)
            CALL TTOUT(0,15,1,BLANK,80,80,31)
            CALL TTOUT(0,9,10,'Do you want to pass a variable name',35,
     *      35,31)
            CALL TTOUT(0,10,10,'for this parameter (Y/N) ?',26,26,31)
            CALL LOCATE(0,10,46)
            CALL INKEY(KYCOD1,KYCOD2)

C     IF THE USER WANTS TO PASS A VARIABLE NAME, THE USER IS THEN
C     PROMPTED FOR THE VARIABLE NAME THAT NEEDS TO BE PASSED.
C     THE NAME TYPED IN IS READ, AND COMPARED WITH VARIABLE NAMES READ
C     FROM THE VARIABLES FILE.

            IF (KYCOD1 .EQ. 89 .OR. KYCOD1 .EQ. 121) THEN
               CALL TTOUT(0,9,1,BLANK,80,80,31)
               CALL TTOUT(0,10,1,BLANK,80,80,31)
709            CALL TTOUT(0,9,20,'Variable name = ',16,16,31)
               CALL PARTM(0,9,36,CHAR,LL,2)
704            READ (3,300,END=703) IC2,IC3,NNN,ICODE,CNT
```

```
               IF (CNT(1:NNN) .NE. CHAR(1:LL)) GOTO 704

C      IF THE VARIABLE NAME IS A CONSTANT OR A COUNTER, ITS VALUE IS READ
C      FROM THE SYMBOL TABLE FILE, AND WRITTEN TO THE LOCATION OF THE
C      FORMAL PARAMTER IN THE SYMBOL TABLE FILE.

               IF (ICODE .EQ. 2 .OR. ICODE .EQ. 4) THEN
                  READ (8,501,REC=IC2) CC1
                  WRITE (8,500,REC=IC21(I)) IC21(I),CC1

C      IF THE VARIABLE NAME IS A POINT OR AN AGGREGATE, THE FOUR VALUES
C      ARE READ, AND WRITTEN TO THE LOCATION OF THE FORMAL PARAMETER.

               ELSEIF (ICODE .EQ. 3 .OR. ICODE .EQ. 7 .OR. ICODE .EQ. 8)
     *         THEN
                  READ (8,501,REC=IC2) CC1,CC2,CC3,CC4
                  WRITE (8,500,REC=IC21(I)) IC21(I),CC1,CC2,CC3,CC4

C      IF THE VARIABLE NAME IS A POINT DEFINED AS A SET OF COUNTERS,
C      THE COUNTER NUMBERS ARE READ AND WRITTEN TO THE LOCATION OF THE
C      OF THE FORMAL PARAMETER.

               ELSEIF (ICODE .EQ. 9) THEN
                  READ (8,699,REC=IC2) NC1,NC2,NC3,NC4,NC5,NC6,NC7,NC8
                  WRITE (8,600,REC=IC21(I)) IC21(I),NC1,NC2,NC3,NC4,NC5,
     *            NC6,NC7,NC8
               ENDIF

               REWIND (3)
               GOTO 702

C      IF THE VARIABLE NAME IS NOT FOUND, AN ERROR MESSAGE IS DISPLAYED,
C      AND THE EXECUTIONS OPTIONS MENU IS DISPLAYED AGAIN.

703            CALL CSROFF
          CALL TTOUT(0,24,1,'Variable not defined -- Hit any ',32,32,28)
               CALL TTOUT(0,24,33,'key to resume....',17,17,28)
               CALL INKEY(KYCOD1,KYCOD2)
               REWIND (3)
               CALL CSRON
               IER = 1
               GOTO 1000
             ENDIF

C      IF THE USER WANTS TO PASS NUMERIC VALUES TO THE FORMAL
C      PARAMETER, THE PROGRAM PROMPTS THE USER FOR THE NUMBER OF
C      OF VALUES TO BE PASSED FOR IT. THE MAXIMUM NUMBER THAT CAN
C      BE PASSED IS FOUR, FOR EACH PARAMETER.

            CALL TTOUT(0,9,1,BLANK,80,80,31)
            CALL TTOUT(0,10,1,BLANK,80,80,31)
            CALL CSRON
            CALL TTOUT(0,9,1,'Please specify the number of values ',36,
     *      36,31)
            CALL TTOUT(0,9,37,'for this parameter and then please',34,
     *      34,31)
720        CALL TTOUT(0,10,1,'enter them in the proper order.',31,31,31)
            CALL PARTM(0,10,32,CHAR,LL,1)
            WRITE (39,100) LL,(CHAR(MM:MM),MM=1,LL)
            REWIND (39)
            READ (39,800) VALU1
            REWIND (39)
            IVAL1 = VALU1

            IF (IVAL1 .GT. 4) THEN
               CALL CSROFF
            CALL TTOUT(0,24,1,'Too may values -- Hit any key ',30,30,28)
```

```
            CALL TTOUT(0,24,31,'to resume....',13,13,28)
            CALL INKEY(KYCOD1,KYCOD2)
            CALL CSRON
            IER = 1
            GOTO 1000
         ENDIF

         IF (IVAL1 .LT. 1) THEN
            CALL CSROFF
         CALL TTOUT(0,24,1,'Too few values -- Hit any key ',30,30,28)
            CALL TTOUT(0,24,31,'to resume....',13,13,28)
            CALL INKEY(KYCOD1,KYCOD2)
            CALL CSRON
            IER = 1
            GOTO 1000
         ENDIF

C     THE PROGRAM THEN PROMPTS THE USER FOR EACH VALUE FOR EVERY PARAMETER.
C     THESE VALUES ARE THEN WRITTEN TO THE LOCATION OF THE FORMAL
C     PARAMETER IN THE SYMBOL TABLE FILE.

         DO 730 JJJJ = 1,IVAL1
            IF (JJJJ .EQ. 1) C(1:1) = '1'
            IF (JJJJ .EQ. 2) C(1:1) = '2'
            IF (JJJJ .EQ. 3) C(1:1) = '3'
            IF (JJJJ .EQ. 4) C(1:1) = '4'
            CALL TTOUT(0,11+JJJJ,25,'Value ',6,6,31)
            CALL TTOUT(0,11+JJJJ,31,C(1:1),1,1,31)
            CALL TTOUT(0,11+JJJJ,33,'= ',2,2,31)
            CALL PARTM(0,11+JJJJ,35,CHAR,LL,1)
            WRITE (39,100) LL,(CHAR(MM:MM),MM=1,LL)
            REWIND (39)
            READ (39,800) VAL(JJJJ)
            REWIND (39)

730      CONTINUE

         WRITE (8,500,REC=IC21(I)) IC21(I),(VAL(JJJJ),JJJJ=1,IVAL1)
         GOTO 702

      ENDIF

      GOTO 1000

702   CONTINUE

100   FORMAT(I2,1X,72A1)
300   FORMAT(1X,I3,7X,I4,2X,I3,2X,I1,6X,A72)
400   FORMAT(18X,I4,1X,I4)
500   FORMAT(1X,I3,4(2X,F8.2))
501   FORMAT(4X,4(2X,F8.2))
600   FORMAT(1X,I3,8(1X,I4))
699   FORMAT(4X,8(1X,I4))
800   FORMAT(3X,F8.2)

1000  CLOSE (39,STATUS='DELETE')

      RETURN
      END


C     SUBROUTINE TTOUT

C     THIS SUBROUTINE WRITES A STRING A CHARACTERS AT A SPECIFIED
C     ROW AND COLUMN OF THE DISPLAY.
```

```
C     SUBROUTINES CALLED BY THIS PROGRAM ARE :
C             LOCATE, MRCHAT.

      SUBROUTINE TTOUT(PAGE,ROW,COLUMN,BUF,LENG,SIZE,MRATT)

      INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
      INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100)
      INTEGER SS,SLPAY,PART1,PART2,VAL1,VAL2,RETVAL,BASC
      INTEGER * 2 PAGE,ROW,COLUMN,SIZE,MRATT,LENG
      INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      INTEGER P29,P30,P31,P32
      REAL THONI,THTWI,ZIN,RIN,CODE
      CHARACTER * 1 BUF(SIZE),CELNAM*10

      DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
      DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
      DIMENSION XPNT(100),YPNT(100),ZPNT(100),RPNT(100)
      DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
      DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)

      COMMON/A/COUNT,CNTVAL,PALLET,PALPTS,PARTNO
      COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
      COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
      COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
      COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
      COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
      COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32


      CALL LOCATE(PAGE,ROW,COLUMN)

      DO 10 J = 1,LENG
         CALL MRCHAT(PAGE,BUF(J),MRATT,1)
10    CONTINUE

      RETURN
      END


      SUBROUTINE  CONCA (AFILE,AEXT)

C      THIS SUBROUTINE CONCATENATES THE FILENAME TO AN EXTENSION
C
C
C          AFILE    EQUIVALENCED WITH FILE NAME
C          AFIX     3 CHARACTER FILE EXTENSION
C
C          AFILE    EXTENSION APPENDED AND UPPERCASE
C
      CHARACTER * 20 AFILE
      CHARACTER * 3 AEXT


100   DO 130 I = 1,20
       IF(AFILE(I:I) .EQ. '.' .OR. AFILE(I:I) .EQ. ' ') GO TO 140
130   CONTINUE


140   AFILE(I:I) = '.'
      DO 150 J =1,3
```

```
      AFILE(I+J:I+J) = AEXT(J:J)
150   CONTINUE


C     CAPITALIZE

      DO 200 K = 1,I+J
      IF(AFILE(K:K) .GE. 'a' .AND.  AFILE(K:K) .LE. 'z' ) then
          AFILE(K:K) = CHAR (ICHAR (AFILE(K:K)) -32 )
      ENDIF
200   CONTINUE

      RETURN
      END


      SUBROUTINE  CONCAD (AFILE,AEXT)

C      THIS SUBROUTINE CONCATENATES A CHARACTER TO A STRING
C
C
C          AFILE    ORIGINAL STRING
C          AFIX     1 CHARACTER
C

      CHARACTER *72 AFILE
      CHARACTER*1 AEXT

100   DO 130 I = 1,72
       IF(AFILE(I:I) .EQ. ' ') GO TO 140
130   CONTINUE

140   I = I - 1
      AFILE(I+1:I+1) = AEXT

      RETURN
      END
```

****

(2) DEPUTY1 - GRAPHICS DISPLAY PROGRAM

THIS PROGRAM 'DEPUTY1' IS CALLED BY THE PROGRAM MAIN AND THE PROGRAM ALTER. IT HANDLES CODES FROM 1 TO 9

```
C   VARIABLES USED IN THIS PROGRAM. SOME OF THESE ARE ALSO
C   IN THE PROGRAM MAIN.

C    BASC       =  INTEGER TO REPRESENT PLAN(80)/SIDE VIEW(80)
C    GRIPX      =  X COORDINATE OF GRIPPER ROTATION INDICATOR
C    GRIPY      =  Y COORDINATE OF GRIPPER ROTATION INDICATOR
C    NTTY       =  VALUE OF THE ENTITY IN THE DATA FILE
C    P1-P16     =  SIXTEEN INPUT PORTS
C    P16-P32    =  SIXTEEN OUTPORT PORTS
C    RIN        =  INITIAL VALUE OF R
C    SXMAX      =  UPPER LIMIT OF USERS X COORDINATE IN THE SIDE VIEW
C    SXMIN      =  LOWER LIMIT OF USERS X CORDINATE IN THE SIDE VIEW
C    SYMAX      =  UPPER LIMIT OF USERS Y COORDINATE IN THE SIDE VIEW
C    SYMIN      =  LOWER LIMIT OF USERS Y CORDINATE IN THE SIDE VIEW
C    THONI      =  INITIAL ANGLE FOR ARM ONE
C    THTWI      =  INITIAL ANGLE FOR ARM TWO
C    XARM(4)    =  X COORDINATES OF THE ARM
C    XBT        =  X COORDINATE OF ARM BASE CIRCLE TOP LINK POINT
C    XBB        =  X COORDINATE OF ARM BASE CIRCLE BOTTOM LINK POINT
C    XET        =  X COORDINATE OF ARM END CIRCLE TOP LINK POINT
C    XEB        =  X COORDINATE OF ARM END CIRCLE BOTTOM LINK POINT
```

```
C     XGRP       =  X COORDINATE  OF GRIPPER CIRCLE
C     XMAX       =  UPPER LIMIT OF USERS X COORDINATE IN THE PLAN VIEW
C     XMIN       =  LOWER LIMIT OF USERS X CORDINATE IN THE PLAN VIEW
C     XMT1       =  X COORDINATE OF ARM MIDDLE CIRCLE TOP LINK LEFT POINT
C     XMB1       =  X COORDINATE OF ARM MIDDLE CIRCLE BOTTOM LEFT LINK PT.
C     XMT2       =  X COORDINATE OF ARM MIDDLE CIRCLE TOP LINK RIGHT POINT
C     XMB2       =  X COORDINATE OF ARM MIDDLE CIRCLE BOTTOM RIGHT LINK PT.
C     XPLBA(10)  =  X COORDINATE OF ARM BASE CIRCLE
C     XPLED(10)  =  X COORDIANTE OF ARM END CIRCLE
C     XPLMD(10)  =  X COORDINATE OF ARM MIDDLE CIRCLE
C     XWORK(I)   =  X COORDINATE OF WORKSPACE POINTS IN PLAN VIEW
C     YARM(4)    =  Y COORDINATES OF THE ARM
C     YBT        =  Y COORDINATE OF ARM BASE CIRCLE TOP LINK POINT
C     YBB        =  Y COORDINATE OF ARM BASE CIRCLE BOTTOM LINK POINT
C     YET        =  Y COORDINATE OF ARM END CIRCLE TOP LINK POINT
C     YEB        =  Y COORDINATE OF ARM END CIRCLE BOTTOM LINK POINT
C     YGRP       =  Y COORDINATE OF GRIPPER CIRCLE
C     YMAX       =  UPPER LIMIT OF USERS Y COORDINATE IN THE PLAN VIEW
C     YMIN       =  LOWER LIMIT OF USERS Y CORDINATE IN THE PLAN VIEW
C     YMT1       =  Y COORDINATE OF ARM MIDDLE CIRCLE TOP LINK LEFT POINT
C     YMB1       =  Y COORDINATE OF ARM MIDDLE CIRCLE BOTTOM LEFT LINK PT.
C     YMT2       =  Y COORDINATE OF ARM MIDDLE CIRCLE TOP LINK RIGHT POINT
C     YMB2       =  Y COORDINATE OF ARM MIDDLE CIRCLE BOTTOM RIGHT LINK PT.
C     YPLBA(10)  =  Y COORDINATE OF ARM BASE CIRCLE
C     YPLED(10)  =  Y COORDIANTE OF ARM END CIRCLE
C     YPLMD(10)  =  Y COORDINATE OF ARM MIDDLE CIRCLE
C     YWORK(I)   =  Y COORDINATE OF WORKSPACE POINTS IN PLAN VIEW
C     ZARM(4)    =  Z COORDINATE OF THE ARM
C

C
C     SUBROUTINES CALLED BY THIS PROGRAM ARE
C        MOVE, ZOOM, REDRAW, PLANSIDE, DEAD, INPORT, OUTPORT,
C        FRAMES, STROUT, SWITCH, STANDARD, WKSPACET.


      SUBROUTINE DEPUTY1(XVAL,YVAL,ZVAL,RVAL)

      INTEGER ISCAN,IASC,NCHAR,KOUNT,BASC,VAL1,VA2
      INTEGER VALO,VALT,VALRET,RETVAL,VAL5
      INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      INTEGER P29,P30,P31,P32
      REAL XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      REAL XPLBA(11),YPLBA(11),XCIRC(11),YCIRC(11)
      REAL XPLMD(11),YPLMD(11),XPLED(11),YPLED(11)
      REAL THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2,ZIN,RIN
      REAL XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      REAL XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      REAL XARM(4), YARM(4), ZARM(4), THETA, PI
      REAL XWORK(675), YWORK(675),GRIPX,GRIPY
      CHARACTER*7 TYPE,CELNAM*10,CELNA*10,ANS*1

      COMMON/G/ SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/K/ XPLBA,YPLBA,XCIRC,YCIRC,XPLMD,YPLMD
      COMMON/L/ XPLED,YPLED,TH1,TH2,X,Y,Z,R,L1,L2
      COMMON/M/ XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      COMMON/N/ XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      COMMON/O/ XARM,YARM,ZARM,THETA,PI,GRIPX,GRIPY
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

      X     = XVAL
      Y     = YVAL
```

```
      Z      = ZVAL
      R      = RVAL

      THETA = 0.0
      PI    = 3.1416
      KOUNT = 1

      CALL DSWAP(2)


C     MOVE TO ANY CHOSEN POINT

      IF ( CODE .EQ. 1.0) THEN
      CALL MOVE

C     OPEN OR CLOSE INPORT

      ELSEIF ( CODE .EQ. 2.0 ) THEN
             CALL INPORT

C     OPEN OR CLOSE OUTPORT

      ELSEIF ( CODE .EQ. 3.0 ) THEN
             CALL OUTPORT

C     GO HOME, DISPLAY IN CURRENT COORDINATES

      ELSEIF ( CODE .EQ. 4.0) THEN

         XARM(1) = 0.0
         YARM(1) = 0.0
         ZARM(1) = 325.0

         XARM(2) = 400.0
         YARM(2) = 0.0
         ZARM(2) = 325.0

         XARM(3) = 650.0
         YARM(3) = 0.0
         ZARM(3) = 325.0

         XARM(4) = 650.0
         YARM(4) = 0.0
         ZARM(4) = 250.0

         GRIPX   = 175.0
         GRIPY   = 0.0

         CALL REDRAW

      ELSEIF ( CODE .EQ. 5.0) THEN
        CALL ZOOM
      ELSEIF ( CODE .EQ. 6.0) THEN
        CALL SWITCH
      ELSEIF ( CODE .EQ. 7.0) THEN
        BASC = 80
       CALL STANDARD
      ELSEIF ( CODE .EQ. 8.0 .OR. CODE .EQ. 9.0) THEN
       CALL REDRAW
      ENDIF

      RETURN
      END


C     TO MOVE THE ROBOT ARM TO THE NEW XYZR POSITION
```

```
      SUBROUTINE MOVE

      INTEGER ISCAN,IASC,NCHAR,KOUNT,COUNT,BASC,VAL1,VAL2,RETVAL
      REAL XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      REAL XPLBA(11),YPLBA(11),XCIRC(11),YCIRC(11)
      REAL XPLMD(11),YPLMD(11),XPLED(11),YPLED(11)
      REAL THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2 ,ZIN,RIN
      REAL XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      REAL XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      REAL XARM(4), YARM(4), ZARM(4), THETA, PI
      REAL XSTAND(2),YSTAND(2),ZSTAND(2),DUMMY
      REAL INC1,INC2,INCZ,DTR,GRIPX,GRIPY
      CHARACTER*7 TYPE,CELNAM*10,ANS*1

      COMMON/G/ SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/K/ XPLBA,YPLBA,XCIRC,YCIRC,XPLMD,YPLMD
      COMMON/L/ XPLED,YPLED,TH1,TH2,X,Y,Z,R,L1,L2
      COMMON/M/ XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      COMMON/N/ XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      COMMON/O/ XARM,YARM,ZARM,THETA,PI,GRIPX,GRIPY
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

      THETA = 0.0
      L1 = 400.0
      L2 = 250.0
      TH1 = 0.0
      TH2=0.0

C     TO COMPUTE TH THETAS THE TWO ARMS MUST MOVE TO REACH XYZ POSITION

      IF (X .EQ. -650.0 .AND. Y .EQ. 0.0)THEN
         THONF = 180.0
         THTWF = 0.0
         GOTO 100
      ENDIF

      RNUM = X**2 + Y**2 - L1**2 - L2**2
      RDEN = 2.0*L1*L2
      RDUM1 = RNUM/RDEN
      THTWF = ACOS(RDUM1)

      RNUM = Y*(L1 + L2*COS(THTWF)) - X*L2*SIN(THTWF)
      RDEN = X*(L1 + L2*COS(THTWF)) + Y*L2*SIN(THTWF)

      IF (RDEN .EQ. 0.0)THEN
         THONF = PI/2.
      ELSE
         THONF = ATAN(RNUM/RDEN)
      ENDIF

C     IF FINAL DESTINATION IS IN QUADRANT ONE
C

      IF (X .GE. 0. .AND. Y .GE. 0.) THEN
         IF (X .EQ. 0.0)THEN
            THINT = PI/2.
         ELSE
            THINT =ATAN(Y/X)
         ENDIF
      ENDIF

C     IF FINAL DESTINATION IS IN QUADRANT TWO
```

```
      IF (X .LT. 0. .AND. Y .GE. 0.) THEN
         IF (THONF .LT. 0) THEN
            THONF = THONF + PI
         ENDIF
         THINT = ATAN(Y/X) + PI
      ENDIF

C     IF FINAL DESTINATION IS IN QUADRANT THREE

      IF (X .LT. 0. .AND. Y .LT. 0.)  THEN
         THONF = THONF + PI
         THINT = ATAN(Y/X) + PI
      ENDIF

C     COMMON TO ALL THREE QUADRANTS

      THINT = THINT/PI*180.
      THONF = THONF/PI*180.
      THTWF = THTWF/PI*180.

      THDIF = THINT - THONF
      THTWP = - THTWF
      THONP = THINT + THDIF

C     CHECK WHETHER ARMS MEET PHYSICAL CONSTRAINTS

      IF (THTWF .GT. 135.0) THEN
      CALL STROUT(5,110,'ARM DOES NOT MEET PHYSICAL CONSTRAINTS ',38)
      CALL STROUT(5,110,'CHECK TOOL TIP CORDINATES              ',38)
         RETURN
      ENDIF

      IF (THTWF .LT. 20.0) THEN
         IF (THONF .GE. 0.0 .AND. THONF .LE. 200.0)  THEN
            IF (THONP .LT. 0.0 .OR. THONP .GT. 200.0)  THEN
               THTWF = THTWF
               THONF = THONF
               GOTO 100
            ENDIF
         ELSE
            IF (THONP .GE. 0.0 .AND. THONP .LE. 200.0)  THEN
               THTWF = THTWP
               THONF = THONP
               GOTO 100
            ELSE
      CALL STROUT(5,110,'ARM DOES NOT MEET PHYSICAL CONSTRAINTS ',38)
      CALL STROUT(5,110,'CHECK TOOL TIP CORDINATES              ',38)
               RETURN
            ENDIF
         ENDIF
      ELSE
         IF (THONF .GE. 0.0 .AND. THONF .LE. 200.0)  THEN
            THTWF = THTWF
            THONF = THONF
            GOTO 100
         ELSE
      CALL STROUT(5,110,'ARM DOES NOT MEET PHYSICAL CONSTRAINTS ',38)
      CALL STROUT(5,110,'CHECK TOOL TIP CORDINATES              ',38)

            RETURN
         ENDIF
      ENDIF

C     COMPUTE THE TOTAL ANGLE MOVED FOR CASE1 (FINAL ANGLES)

      DTHTWF = THTWF - THTWI
      DTHONF = THONF - THONI
```

```
      DTHF   =  ABS(DTHTWF) + ABS(DTHONF)

C     COMPUTE THE TOTAL ANGLE MOVED  FOR CASE2 (PRIME ANGLES)

      DTHTWP = THTWP - THTWI
      DTHONP = THONP - THONI
      DTHP   =  ABS(DTHTWP) + ABS(DTHONP)

C     COMPARE AND SELECT THE LEAST VALUE

      DUMMY = MIN (DTHF,DTHP)

      IF (DUMMY .NE. DTHF) THEN
         THTWF = THTWP
         THONF = THONP
      ENDIF

 100  THINT = THINT*PI/180.
      THONF = THONF*PI/180.
      THTWF = THTWF*PI/180.

C     FIND THE ANGLE TO BE MOVED FOR EACH ARM

      DT1 = THONF - THONI
      DT2 = THTWF - THTWI

     FIND INCREMENT FOR EACH ARM BASED ON 10 SEGMENTS FOR EACH MOVE

      INC1 = DT1/3.
      INC2 = DT2/3.

      XARM(1) = 0.0
      YARM(1) = 0.0

C  FIRST DRAW THE PLAN COMPLETELY

      CALL TCSIND(3)

C     FOR BASE CIRCLE

      DO 399  I = 1,11
           THETA = (I-1)*PI/5.
           XPLBA(I) = 45.*COS(THETA)
           YPLBA(I) = 45.*SIN(THETA)
 399  CONTINUE

C THE LOOP STARTS HERE TO FIND ALL THE NEW VALUES FOR THE CIRCLES & LINKS
C TO DRAW IN THE NEW POSITION.

        INCREM = 3
        DO 1100 J = 1,INCREM

      IF (BASC .EQ. 80) THEN
        CALL TWINDO(0,750,0,750)
        CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
      CALL TCSIND(6)
      CALL STROUT(610,750,'PLAN VIEW',9)
      ELSE
        CALL TWINDO(751,1023,400,672)
        CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
      CALL TCSIND(6)
```

```
      CALL STROUT(890,645,'PLAN VIEW',9)
      ENDIF


        TH1 = THONI + J*INC1
        TH2 = THTWI + J*INC2

        XARM(2) = L1*COS(TH1)
        YARM(2) = L1*SIN(TH1)

        XARM(3) = L2*COS(TH1+TH2) + XARM(2)
        YARM(3) = L2*SIN(TH1+TH2) + YARM(2)


C  FIRST THE TEMPLATE CIRCLE

      DO 200  I = 1,11
           THETA = (I-1)*PI/5.
           XCIRC(I) = 45.*COS(THETA)
           YCIRC(I) = 45.*SIN(THETA)
 200  CONTINUE

C    BASE CIRCLE DONE OUTSIDE THE LOOP ONCE FOR ALL

C    FOR MIDDLE CIRCLE

          DO  402 I = 1, 11
         XPLMD(I) =  XCIRC(I) + XARM(2)
         YPLMD(I) =  YCIRC(I) + YARM(2)
402       CONTINUE
C     LAST CIRCLE

          DO  403 I = 1, 11
         XPLED(I) =  XCIRC(I)  + XARM(3)
         YPLED(I) =  YCIRC(I)  + YARM(3)
403       CONTINUE

C      BASE CIRCLE

        XBT  = -45.*SIN(TH1)
        YBT  = +45.*COS(TH1)
        XBB  = +45.*SIN(TH1)
        YBB  = -45.*COS(TH1)


C MIDDLE CIRCLE - TOP AND BOTTOM - LEFT

       XMT1 = XARM(2) - 45.*SIN(TH1)
       YMT1 = YARM(2) + 45.*COS(TH1)
       XMB1 = XARM(2) + 45.*SIN(TH1)
       YMB1 = YARM(2) - 45.*COS(TH1)

C MIDDLE CIRCLE - TOP AND BOTTOM - RIGHT

       XMT2 = XARM(2) - 45.*SIN(TH2)
       YMT2 = YARM(2) + 45.*COS(TH2)
       XMB2 = XARM(2) + 45.*SIN(TH2)
       YMB2 = YARM(2) - 45.*COS(TH2)

C END CIRCLE - TOP AND BOTTOM

       XET = XARM(3) - 45.*SIN(TH2)
       YET = YARM(3) + 45.*COS(TH2)
       XEB = XARM(3) + 45.*SIN(TH2)
       YEB = YARM(3) - 45.*COS(TH2)

C   START DRAWING, THE CIRCLES FIRST.
```

```
C     THE BASE CIRCLE IS PART OF THE DRAW PLAN ROUTINE

C     DRAW THE MIDDLE CIRCLE

      CALL MOVEA(XPLMD(1),YPLMD(1))
      DO 720 I =1,11
      CALL DRAWA(XPLMD(I),YPLMD(I))
 720  CONTINUE

C     DRAW THE LAST CIRCLE

      CALL MOVEA(XPLED(1),YPLED(1))
      DO 730 I=1,11
      CALL DRAWA(XPLED(I),YPLED(I))
 730  CONTINUE

C    LINK 1

      CALL MOVEA(XBT,YBT)
      CALL DRAWA(XMT1,YMT1)
      CALL MOVEA(XBB,YBB)
      CALL DRAWA(XMB1,YMB1)

C    LINK 2

      CALL MOVEA(XMT2,YMT2)
      CALL DRAWA(XET,YET)
      CALL MOVEA(XMB2,YMB2)
      CALL DRAWA(XEB,YEB)
C
C
C       NOW DRAW THE SIDE VIEW
C
C
      IF (BASC .EQ. 80) THEN
        CALL TWINDO(751,1023,400,672)
        CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)
      ELSE
        CALL TWINDO(0,750,0,750)
        CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)
      ENDIF

C    START DRAWING, THE CIRCLES FIRST (IN SIDE VIEW, THEY ARE RECT.)

C      BASE CIRCLE

C     TOP LINE

      CALL MOVEA(YPLBA(1),500.)
      CALL DRAWA(YPLBA(1),500.)
      CALL DRAWA(YPLBA(5),500.)

C BOTTOM LINE

      CALL MOVEA(YPLBA(1),350.)
      CALL DRAWA(YPLBA(1),350.)
      CALL DRAWA(YPLBA(5),350.)

C LEFT LINE

      CALL MOVEA(YPLBA(1),350.)
      CALL DRAWA(YPLBA(1),350.)
      CALI DRAWA(YPLBA(1),500.)

C RIGHT LINE
```

```
      CALL MOVEA(YPLBA(5),350.)
      CALL DRAWA(YPLBA(5),350.)
      CALL DRAWA(YPLBA(5),500.)

C     DRAW THE MIDDLE CIRCLE

C     TOP LINE

      CALL MOVEA(YPLMD(1),500.)
      CALL DRAWA(YPLMD(1),500.)
      CALL DRAWA(YPLMD(5),500.)

C BOTTOM LINE

      CALL MOVEA(YPLMD(1),350.)
      CALL DRAWA(YPLMD(1),350.)
      CALL DRAWA(YPLMD(5),350.)

C LEFT LINE

      CALL MOVEA(YPLMD(1),350.)
      CALL DRAWA(YPLMD(1),350.)
      CALL DRAWA(YPLMD(1),500.)

C RIGHT LINE

      CALL MOVEA(YPLMD(5),350.)
      CALL DRAWA(YPLMD(5),350.)
      CALL DRAWA(YPLMD(5),500.)

C     DRAW THE LAST CIRCLE

C     TOP LINE

      CALL MOVEA(YPLED(1),500.)
      CALL DRAWA(YPLED(1),500.)
      CALL DRAWA(YPLED(5),500.)

C BOTTOM LINE

      CALL MOVEA(YPLED(1),350.)
      CALL DRAWA(YPLED(1),350.)
      CALL DRAWA(YPLED(5),350.)

C LEFT LINE

      CALL MOVEA(YPLED(1),350.)
      CALL DRAWA(YPLED(1),350.)
      CALL DRAWA(YPLED(1),500.)

C RIGHT LINE

      CALL MOVEA(YPLED(5),350.)
      CALL DRAWA(YPLED(5),350.)
      CALL DRAWA(YPLED(5),500.)

C      NOW DRAW THE LINKS

C   LINK 1

      CALL MOVEA(YPLBA(1),500.)
      CALL DRAWA(YPLMD(5),500.)
      CALL MOVEA(YPLBA(1),350.)
      CALL DRAWA(YPLMD(5),350.)

C   LINK 2
```

```
      CALL MOVEA(YPLMD(1),500.)
      CALL DRAWA(YPLED(5),500.)
      CALL MOVEA(YPLMD(1),350.)
      CALL DRAWA(YPLED(5),350.)

C     THE GRIPPER HEAD AND THE MOVING GRIPPER PART

C     THE GRIPPER HEAD

      DUMMY = YARM(3) + 25.
      CALL MOVEA (DUMMY,375.)
      CALL DRAWA(DUMMY,350.)
      DUMMY = YARM(3) - 25.
      CALL DRAWA(DUMMY,350.)
      CALL DRAWA(DUMMY,500.)
      DUMMY = YARM(3) + 25.
      CALL DRAWA(DUMMY,500.)
      CALL DRAWA(DUMMY,375.)

C  DRAW THE GRIPPER IN THE LAST POSITION FOR ALL THE REDRAWS

      DUMMY = YARM(3) + 10.
      CALL MOVEA(DUMMY,350.)
      CALL DRAWA(DUMMY,ZARM(4))

      DUMMY = YARM(3) - 10.
      CALL DRAWA(DUMMY,ZARM(4))
      CALL DRAWA(DUMMY,350.)

1100  CONTINUE

C     AFTER ALL THIS SIMULATE THE Z AXIS-GRIPPER MOVEMENT.


      DTZ = Z -ZIN
      INCZ = DTZ/5.0

      DO 1250 J = 1,INCREM

      ZARM(4) = ZIN + J*INCZ

      DUMMY = YARM(3) + 10.
      CALL MOVEA(DUMMY,350.)
      CALL DRAWA(DUMMY,ZARM(4))

      DUMMY = YARM(3) - 10.
      CALL DRAWA(DUMMY,ZARM(4))
      CALL DRAWA(DUMMY,350.)

 1250 CONTINUE

C     GRIPPER WINDOW

      DTR = R - RIN
      INCR = DTR/3.0

      CALL TWINDO(751,859,672,780)
      CALL DWINDO(-400.,400.,-400.,400.)

      DO 1300 J = 1, 3

      CALL MOVEA(0.,0.)

      TETA = J*INCR*PI/180.
      GRIPX = 175.*COS(TETA + RIN)
      GRIPY = 175.*SIN(TETA + RIN)
      CALL DRAWA(GRIPX,GRIPY)
```

```
 1300 CONTINUE

C     REINITIALIZE INITIAL ANGLE VALUES TO FINAL VALUES

       CALL TCSBEL
       THONI = THONF
       THTWI = THTWF
       ZIN = Z
       RIN = R


1200   RETURN
       END

C       FOR ZOOM

       SUBROUTINE ZOOM

       REAL XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
       REAL XPLBA(11),YPLBA(11),XCIRC(11),YCIRC(11)
       REAL XPLMD(11),YPLMD(11),XPLED(11),YPLED(11)
       REAL THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2,ZIN,RIN
       REAL XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
       REAL XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
       REAL XARM(4), YARM(4), ZARM(4), THETA, PI
       REAL GRIPX,GRIPY,COODE,ASPECT,DUMMY,XAVG
       REAL X1,X2,Y1,Y2,XC,YC,DELX,DELY,YAVG
       INTEGER BASC,VAL1,VAL2,RETVAL
       CHARACTER*7 TYPE,CELNAM*10,ANS*1

       COMMON/G/ SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
       COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
       COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
       COMMON/K/ XPLBA,YPLBA,XCIRC,YCIRC,XPLMD,YPLMD
       COMMON/L/ XPLED,YPLED,TH1,TH2,X,Y,Z,R,L1,L2
       COMMON/M/ XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
       COMMON/N/ XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
       COMMON/O/ XARM,YARM,ZARM,THETA,PI,GRIPX,GRIPY
       COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
       COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
       COMMON/R/ P29,P30,P31,P32

       IF (BASC .EQ. 80) THEN

       CALL TWINDO(0,750,0,750)
       CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
       CALL STROUT
      *     (115,65,'SELECT NEW WINDOW CORNER  PRESS SPACE BAR ',42)
       CALL VCURSR (ICHAR,X1,Y1)
       CALL STROUT
      *     (115,40,'SELECT OPPOSITE CORNER  PRESS SPACE BAR ',40)
       CALL VCURSR (ICHAR,X2,Y2)

       XMIN = X1
       IF (X2 .LT. X1)  XMIN=X2
       YMIN =Y1
       IF (Y2 .LT. Y1)  YMIN=Y2
       DELX = ABS(X2-X1)
       DELY = ABS(Y2-Y1)
       YAVG = (Y2+Y1)/2.0
       XAVG = (X2+X1)/2.0
       IF (DELX .EQ. 0.0 ) DELX =0.001
       IF (DELY .EQ. 0.0 ) DELY =0.001
       ASPECT = DELY /DELX
       IF (ASPECT .LT. 1.0) DELY = DELY *1.0/ASPECT
       IF (ASPECT .GT. 1.0) DELX = DELX *ASPECT /1.0
```

```
      YMIN =YAVG -DELY/2.0
      XMIN =XAVG -DELX/2.0
      XMAX = XMIN + DELX
      YMAX = YMIN + DELY

C     TO DRAW THE ARM IN HOME POSITION

      XARM(1) = 0.0
      YARM(1) = 0.0
      ZARM(1) = 325.0

      XARM(2) = 400.
      YARM(2) = 0.0
      ZARM(2) = 325.0

      XARM(3) = 650.0
      YARM(3) = 0.0
      ZARM(3) = 325.0

      XARM(4) = 650.0
      YARM(4) = 0.0
      ZARM(4) = 250.0

      OPEN (22,FILE='INFO',STATUS='NEW')
      WRITE(22,250)BASC,XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
 250  FORMAT(1X,I2,1X,F8.2,1X,F8.2,1X,F8.2,1X,F8.2,1X,F8.2,
     *1X,F8.2,1X,F8.2,1X,F8.2)
      CLOSE(22)

      CALL REDRAW

      ELSE

      CALL TWINDO(0,750,0,750)
      CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)
      CALL STROUT
     *     (115,65,'SELECT NEW WINDOW CORNER  PRESS SPACE BAR ',42)
      CALL VCURSR (ICHAR,X1,Y1)
      CALL STROUT
     *     (115,40,'SELECT OPPOSITE CORNER  PRESS SPACE BAR ',40)
      CALL VCURSR (ICHAR,X2,Y2)

      SXMIN = X1
      IF (X2 .LT. X1)  SXMIN=X2
      SYMIN =Y1
      IF (Y2 .LT. Y1)  SYMIN=Y2
      DELX = ABS(X2-X1)
      DELY = ABS(Y2-Y1)
      YAVG = (Y2+Y1)/2.0
      XAVG = (X2+X1)/2.0
      IF (DELX .EQ. 0.0 ) DELX =0.001
      IF (DELY .EQ. 0.0 ) DELY =0.001
      ASPECT = DELY /DELX
      IF (ASPECT .LT. 1.0) DELY = DELY *1.0/ASPECT
      IF (ASPECT .GT. 1.0) DELX = DELX *ASPECT /1.0
      SYMIN =YAVG -DELY/2.0
      SXMIN =XAVG -DELX/2.0
      SXMAX = SXMIN + DELX
      SYMAX = SYMIN + DELY

C     TO DRAW THE ARM IN HOME POSITION

      XARM(1) = 0.0
      YARM(1) = 0.0
      ZARM(1) = 325.0

      XARM(2) = 400.
```

```
      YARM(2) = 0.0
      ZARM(2) = 325.0

      XARM(3) = 650.0
      YARM(3) = 0.0
      ZARM(3) = 325.0

      XARM(4) = 650.0
      YARM(4) = 0.0
      ZARM(4) = 250.0

      OPEN (22,FILE='INFO',STATUS='NEW')
      WRITE(22,250)BASC,XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      CLOSE(22)

      CALL REDRAW

      ENDIF

C  TO PAN THE DRAWING

      IF (BASC .EQ. 80) THEN

      CALL TWINDO(0,750,0,750)
      CALL DWINDO(XMIN,XMAX,YMIN,YMAX)

      CALL STROUT(205,80,'DO YOU WANT TO PAN (Y/N)',24)
      CALL TCSKEY(ISCAN,IASC,NCHAR)
      IF (IASC .EQ. 89) THEN
      GOTO 100
      ELSE
      GOTO 200
      ENDIF

100   CALL STROUT(315,40,'INDICATE NEW CENTER, PRESS SPACE BAR',36)
      CALL VCURSR (ICHAR,XC,YC)
      IF (ICHAR .NE. 77 .AND. ICHAR .NE. 109) GOTO 15
  12  CALL STROUT
     *   (315,60,'ENTER NEW (X,Y) FOR CENTER OF MODEL --> ',40)
      CALL TSEND
      READ (*,*,END =12) XC,YC

  15  DELX =XMAX - XMIN
      DELY =YMAX - YMIN
      XMIN = XC - DELX/2.0
      YMIN = YC - DELY/2.0
      XMAX = XMIN + DELX
      YMAX = YMIN + DELY

      OPEN (22,FILE='INFO',STATUS='NEW')
      WRITE(22,250)BASC,XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      CLOSE(22)

      CALL REDRAW

      ELSE

      CALL TWINDO(0,750,0,750)
      CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)

      CALL STROUT(205,80,'DO YOU WANT TO PAN (Y/N)',24)
      CALL TCSKEY(ISCAN,IASC,NCHAR)
      IF (IASC .EQ. 89) THEN
      GOTO 300
      ELSE
      GOTO 200
      ENDIF
```

```
300   CALL STROUT(315,40,'INDICATE NEW CENTER, PRESS SPACE BAR',36)
      CALL VCURSR (ICHAR,XC,YC)
      IF (ICHAR .NE. 77 .AND. ICHAR .NE. 109) GOTO 25
  22  CALL STROUT
     *   (315,60,'ENTER NEW (X,Y) FOR CENTER OF MODEL --> ',40)
      CALL TSEND
      READ (*,*,END =22) XC,YC

  25  DELX =SXMAX - SXMIN
      DELY =SYMAX - SYMIN
      SXMIN = XC - DELX/2.0
      SYMIN = YC - DELY/2.0
      SXMAX = SXMIN + DELX
      SYMAX = SYMIN + DELY

      OPEN (22,FILE='INFO',STATUS='NEW')
      WRITE(22,250)BASC,XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      CLOSE(22)

      CALL REDRAW

      ENDIF

200   RETURN
      END

C     SUBROUTINE SWITCH TO SWITCH BETWEEN THE PLAN AND SIDE VIEW AS THE
C     MAJOR AND MINOR VIEW.

      SUBROUTINE SWITCH

      REAL XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      REAL XPLBA(11),YPLBA(11),XCIRC(11),YCIRC(11)
      REAL XPLMD(11),YPLMD(11),XPLED(11),YPLED(11)
      REAL THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2,ZIN,RIN
      REAL XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      REAL XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      REAL XARM(4), YARM(4), ZARM(4), THETA, PI
      REAL GRIPX,GRIPY,COODE
      REAL X1,X2,Y1,Y2,XC,YC,DELX,DELY,DUMMY,XAVG,YAVG
      INTEGER BASC,VAL1,VAL2,RETVAL
      CHARACTER*7 TYPE,CELNAM*10
      CHARACTER*1 ANS

      COMMON/G/ SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/K/ XPLBA,YPLBA,XCIRC,YCIRC,XPLMD,YPLMD
      COMMON/L/ XPLED,YPLED,TH1,TH2,X,Y,Z,R,L1,L2
      COMMON/M/ XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      COMMON/N/ XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      COMMON/O/ XARM,YARM,ZARM,THETA,PI,GRIPX,GRIPY
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

      KOUNT = 1
      GRIPX = 175.0
      GRIPY = 0.0

C      CHOOSING THE MAJOR VIEW


      CALL STROUT(5,20,'CHOOSE MAJOR VIEW: PLAN/SIDE VIEW (P/S) ',40)
      CALL TCSKEY(ISCAN,IASC,NCHAR)
      BASC=IASC
```

```
      OPEN (22,FILE='INFO',STATUS='NEW')
      WRITE(22,250)BASC,XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
  250 FORMAT(1X,I2,1X,F8.2,1X,F8.2,1X,F8.2,1X,F8.2,1X,F8.2,
     *1X,F8.2,1X,F8.2,1X,F8.2)
      CLOSE(22)

      CALL REDRAW

      RETURN
      END

C     SUBROUTINE STANDARD IS TO DRAW THE STANDARD VIEW WITHOUT ZOOM . IT
C     WILL DISPLAY THE WORKCELL ALSO.
C
      SUBROUTINE STANDARD

      REAL XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      REAL XPLBA(11),YPLBA(11),XCIRC(11),YCIRC(11)
      REAL XPLMD(11),YPLMD(11),XPLED(11),YPLED(11)
      REAL THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2,ZIN,RIN
      REAL XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      REAL XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      REAL XARM(4), YARM(4), ZARM(4), THETA, PI
      REAL GRIPX,GRIPY,DUMMY,XAVG,YAVG
      REAL X1,X2,Y1,Y2,XC,YC,DELX,DELY
      INTEGER BASC,VAL1,VAL2,RETVAL
      CHARACTER*7 TYPE,CELNAM*10,ANS*1

      COMMON/G/ SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/K/ XPLBA,YPLBA,XCIRC,YCIRC,XPLMD,YPLMD
      COMMON/L/ XPLED,YPLED,TH1,TH2,X,Y,Z,R,L1,L2
      COMMON/M/ XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      COMMON/N/ XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      COMMON/O/ XARM,YARM,ZARM,THETA,PI,GRIPX,GRIPY
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

      XMIN    = -800.0
      XMAX    =  800.0
      YMIN    = -650.0
      YMAX    =  950.0
      SXMIN   = -420.0
      SXMAX   =  684.0
      SYMIN   = -420.0
      SYMAX   =  684.0
      KOUNT   =  1
      GRIPX   =  175.0
      GRIPY   =  0.0
      BASC    =  80

      XARM(1) = 0.0
      YARM(1) = 0.0
      ZARM(1) = 325.0

      XARM(2) = 400.0
      YARM(2) = 0.0
      ZARM(2) = 325.0

      XARM(3) = 650.0
      YARM(3) = 0.0
      ZARM(3) = 325.0

      XARM(4) = 650.0
```

```
      YARM(4) = 0.0
      ZARM(4) = 250.0

      GRIPX   = 175.0
      GRIPY   = 0.0

      THONI   = 0.0
      THTWI   = 0.0
      ZIN     = 250.0

  200 OPEN (22,FILE='INFO',STATUS='NEW')
      WRITE(22,250)BASC,XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
  250 FORMAT(1X,I2,1X,F8.2,1X,F8.2,1X,F8.2,1X,F8.2,1X,F8.2,
     *1X,F8.2,1X,F8.2,1X,F8.2)
      CLOSE(22)

      CALL REDRAW

      RETURN
      END

C     TO REDRAW THE WHOLE PICTURE  - STANDARD VIEW AND WITH WORKCELL
      SUBROUTINE REDRAW

      REAL XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      REAL XPLBA(11),YPLBA(11),XCIRC(11),YCIRC(11)
      REAL XPLMD(11),YPLMD(11),XPLED(11),YPLED(11)
      REAL THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2,ZIN,RIN
      REAL XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      REAL XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      REAL XARM(4), YARM(4), ZARM(4), THETA, PI
      REAL GRIPX,GRIPY,D,E,F,ND,NE,NF
      REAL M(4),N(4),XWORK(675),YWORK(675)
      REAL CX(2), CY(2),CZ(2),NX(2),NY(2),NZ(2)
      INTEGER J(6),VAL1,VAL2,RETVAL,ICOM,ICOMP
      INTEGER ISCAN,IASC,NCHAR,BASC,KOUNT,Q
      CHARACTER*7 TYPE,CELNAM*10,ANS*1
      CHARACTER INFIL*1,NAME*20 ,AFILE*20,DNAME*5
      CHARACTER CHOICE*1,DUMMY*1,NTTY*3 ,AEXT*3
      DIMENSION INFIL(80)

      COMMON/G/ SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/K/ XPLBA,YPLBA,XCIRC,YCIRC,XPLMD,YPLMD
      COMMON/L/ XPLED,YPLED,TH1,TH2,X,Y,Z,R,L1,L2
      COMMON/M/ XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      COMMON/N/ XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      COMMON/O/ XARM,YARM,ZARM,THETA,PI,GRIPX,GRIPY
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

      CALL SETMON(1)

      OPEN (22,FILE='INFO',STATUS='OLD',ERR=110)
      READ(22,250)BASC,XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
  250 FORMAT(1X,I2,1X,F8.2,1X,F8.2,1X,F8.2,1X,F8.2,1X,F8.2,
     *1X,F8.2,1X,F8.2,1X,F8.2)
      CLOSE(22)

  110 CALL DEAD

      OPEN(17,FILE=CELNAM,STATUS='OLD',ERR=100)

    5 CONTINUE
      READ(17,10,END=200)(INFIL(I),I=1,80)
```

```
 10   FORMAT(80A1)
        READ (INFIL(1:3),55) NTTY
 55     FORMAT(A3)
        IF (NTTY .EQ.'110') THEN
          ICOM = 0
          DO 26 Q = 5,72
            IF (INFIL(Q:Q) .EQ. ',' ) THEN
              ICOM =ICOM + 1
              J(ICOM) = Q
             ENDIF


            IF (INFIL(Q:Q) .EQ. ';') THEN
              ICOM = ICOM + 1
              J(ICOM) = Q
              GOTO 27
            ENDIF
  26      CONTINUE

27        K = J(1) - 1
          READ(INFIL(5:K),*)CX(1)
          K = J(1) + 1
          L = J(2) - 1
          READ(INFIL(K:L),*)CY(1)
          K = J(2) + 1
          L = J(3) - 1
          READ(INFIL(K:L),*)CZ(1)
          K = J(3) + 1
          L = J(4) - 1
          READ(INFIL(K:L),*)CX(2)
          K = J(4) + 1
          L = J(5) - 1
          READ(INFIL(K:L),*)CY(2)
          K = J(5) + 1
          L = J(6) - 1
          READ(INFIL(K:L),*)CZ(2)

C       DRAW THE PLAN OF EQP. - EITHER AS THE MAJOR OR MINOR VIEW

        IF (BASC .EQ. 80) THEN
        CALL TWINDO(0,750,0,750)
        CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
        ELSE
        CALL TWINDO(751,1023,400,672)
        CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
        ENDIF

        CALL MOVEA(CX(1),CY(1))
        CALL DRAWA(CX(2),CY(2))

C       DRAW THE SIDEVIEW OF EQP. - EITHER AS THE MAJOR OR MINOR VIEW

        IF (BASC .EQ. 80) THEN
        CALL TWINDO(751,1023,400,672)
        CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)
        ELSE
        CALL TWINDO(0,750,0,750)
        CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)
        ENDIF

        CALL MOVEA(CY(1),CZ(1))
        CALL DRAWA(CY(2),CZ(2))

          GOTO 5

         ELSEIF (NTTY .EQ. '116') THEN
           ICOMP = 0
```

```
          DO 30 Q = 5,72
           IF (INFIL(Q:Q) .EQ. ',') THEN
            ICOMP =ICOMP + 1
            J(ICOMP) = Q
           ENDIF

          IF (INFIL(Q:Q) .EQ. ';') THEN
           ICOMP = ICOMP + 1
           J(ICOMP) = Q
           GOTO 40
          ENDIF
   30   CONTINUE

   40   K = J(1) - 1
        READ(INFIL(5:K),*)D

        K = J(1) + 1
        L = J(2) - 1
        READ(INFIL(K:L),*)E

        K = J(2) + 1
        L = J(3) - 1
        READ(INFIL(K:L),*)F

C       DRAW THE PLAN OF EQP. - EITHER AS THE MAJOR OR MINOR VIEW

        IF (BASC .EQ. 80) THEN
        CALL TWINDO(0,750,0,750)
        CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
        ELSE
        CALL TWINDO(751,1023,400,672)
        CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
        ENDIF

        CALL MOVEA(D,E)
        CALL DRAWA(D,E)

C       DRAW THE SIDEVIEW OF EQP. - EITHER AS THE MAJOR OR MINOR VIEW

        IF (BASC .EQ. 80) THEN
        CALL TWINDO(751,1023,400,672)
        CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)
        ELSE
        CALL TWINDO(0,750,0,750)
        CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)
        ENDIF

        CALL MOVEA(E,F)
        CALL DRAWA(E,F)

          GOTO 5
         ELSE
         GOTO 5
           ENDIF

  200   CLOSE(17)
        CLOSE(19,STATUS='DELETE')

  100   CONTINUE
        CALL PLANSIDE
        CALL FRAMES
        CALL PORTS

        RETURN
        END

C       DRAW THE PLANSIDE
```

```
      SUBROUTINE PLANSIDE

      INTEGER ISCAN,IASC,NCHAR,BASC,KOUNT,VAL1,VAL2,RETVAL
      REAL XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      REAL XPLBA(11),YPLBA(11),XCIRC(11),YCIRC(11)
      REAL XPLMD(11),YPLMD(11),XPLED(11),YPLED(11)
      REAL THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2,ZIN,RIN
      REAL XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      REAL XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      REAL XARM(4), YARM(4), ZARM(4), THETA, PI
      REAL XWORK(674),YWORK(674),GRIPX,GRIPY
      CHARACTER*7 TYPE,CELNAM*10,ANS*1

      COMMON/G/ SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/K/ XPLBA,YPLBA,XCIRC,YCIRC,XPLMD,YPLMD
      COMMON/L/ XPLED,YPLED,TH1,TH2,X,Y,Z,R,L1,L2
      COMMON/M/ XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      COMMON/N/ XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      COMMON/O/ XARM,YARM,ZARM,THETA,PI,GRIPX,GRIPY
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

      THETA = 0.0

C  FIRST DRAW THE PLAN COMPLETELY

      IF (BASC .EQ. 80) THEN
        CALL TWINDO(0,750,0,750)
        CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
        CALL TCSIND(6)
      CALL STROUT(610,750,'PLAN VIEW',9)
      ELSE
        CALL TWINDO(751,1023,400,672)
        CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
        CALL TCSIND(6)
      CALL STROUT(890,645,'PLAN VIEW',9)
      ENDIF

C     DRAW THE WORKSPACE

      CALL WKSPACE

C    FOR BASE CIRCLE

      DO 399  I = 1,11
           THETA = (I-1)*PI/5.
           XPLBA(I) = 45.*COS(THETA)
           YPLBA(I) = 45.*SIN(THETA)
 399  CONTINUE

C    FOR MIDDLE CIRCLE

      DO 389  I = 1,11
           THETA = (I-1)*PI/5.
           XPLMD(I) = 45.*COS(THETA) + XARM(2)
           YPLMD(I) = 45.*SIN(THETA) + YARM(2)
 389  CONTINUE

C    FOR END CIRCLE

      DO 379  I = 1,11
           THETA = (I-1)*PI/5.
           XPLED(I) = 45.*COS(THETA) + XARM(3)
```

```
          YPLED(I) = 45.*SIN(THETA) + YARM(3)
 379  CONTINUE

C       BASE CIRCLE

          XBT  = -45.*SIN(TH1)
          YBT  = +45.*COS(TH1)
          XBB  = +45.*SIN(TH1)
          YBB  = -45.*COS(TH1)


C MIDDLE CIRCLE - TOP AND BOTTOM - LEFT

        XMT1 = XARM(2) - 45.*SIN(TH1)
        YMT1 = YARM(2) + 45.*COS(TH1)
        XMB1 = XARM(2) + 45.*SIN(TH1)
        YMB1 = YARM(2) - 45.*COS(TH1)

C MIDDLE CIRCLE - TOP AND BOTTOM - RIGHT

        XMT2 = XARM(2) - 45.*SIN(TH2)
        YMT2 = YARM(2) + 45.*COS(TH2)
        XMB2 = XARM(2) + 45.*SIN(TH2)
        YMB2 = YARM(2) - 45.*COS(TH2)

C END CIRCLE - TOP AND BOTTOM

        XET = XARM(3) - 45.*SIN(TH2)
        YET = YARM(3) + 45.*COS(TH2)
        XEB = XARM(3) + 45.*SIN(TH2)
        YEB = YARM(3) - 45.*COS(TH2)

C       IN PLAN VIEW

        CALL TCSIND(7)

C
C       START DRAWING THE SQUARE BASE
C
        CALL MOVEA(55.,55.)
        CALL DRAWA(-55.,55.)
        CALL DRAWA(-55.,-55.)
        CALL DRAWA(55.,-55.)
        CALL DRAWA(55.,55.)
        CALL MOVEA(0.,0.)
        CALL DRAWA(0.,0.)


C     DRAW THE BASE CIRCLE

        CALL MOVEA(XPLBA(1),YPLBA(1))
        DO 710 I =1,11
        CALL DRAWA(XPLBA(I),YPLBA(I))
 710  CONTINUE

C     DRAW THE MIDDLE CIRCLE

        CALL TCSIND(14)

        CALL MOVEA(XPLMD(1),YPLMD(1))
        DO 720 I =1,11
        CALL DRAWA(XPLMD(I),YPLMD(I))
 720  CONTINUE

C     DRAW THE END CIRCLE

        CALL MOVEA(XPLED(1),YPLED(1))
```

```
      DO 730 I=1,11
      CALL DRAWA(XPLED(I),YPLED(I))
 730  CONTINUE


C   LINK 1

      CALL MOVEA(XBT,YBT)
      CALL DRAWA(XMT1,YMT1)
      CALL MOVEA(XBB,YBB)
      CALL DRAWA(XMB1,YMB1)

C   LINK 2

      CALL MOVEA(XMT2,YMT2)
      CALL DRAWA(XET,YET)
      CALL MOVEA(XMB2,YMB2)
      CALL DRAWA(XEB,YEB)

C      NOW DRAW THE SIDE VIEW

      IF (BASC .EQ. 80) THEN
        CALL TWINDO(751,1023,400,672)
        CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)
        CALL TCSIND(13)
      CALL STROUT(890,645,'SIDE VIEW',9)
      ELSE
        CALL TWINDO(0,750,0,750)
        CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)
        CALL TCSIND(13)
      CALL STROUT(610,750,'SIDE VIEW',9)
      ENDIF

C     WORKSPACE IN SIDEVIEW


      CALL TCSIND(4)
      CALL MOVEA(-420.,0.)
      CALL DRAWA(684.,0.)
      CALL TCSIND(4)
      CALL MOVEA(-386.81,0.)
      CALL DRAWA(-386.81,250.)
      CALL DRAWA(650.,250.)
      CALL DRAWA(650.,0.)
      CALL DRAWA(-386.81,0.)

C     ROBOT STAND

      CALL TCSIND(7)
      CALL MOVEA(-50.,0.)
      CALL DRAWA(-50.,20.)
      CALL DRAWA(50.,20.)
      CALL DRAWA(50.,0.)
      CALL MOVEA(-30.,20.)
      CALL DRAWA(-30.,500.)
      CALL DRAWA(30.,500.)
      CALL DRAWA(30.,20.)


C   START DRAWING, THE CIRCLES FIRST (IN SIDE VIEW, THEY ARE RECT.)

      CALL TCSIND(14)

C     BASE CIRCLE
```

```
C    TOP LINE

      CALL MOVEA(YPLBA(1),500.)
      CALL DRAWA(YPLBA(5),500.)

C BOTTOM LINE

      CALL MOVEA(YPLBA(1),350.)
      CALL DRAWA(YPLBA(1),350.)
      CALL DRAWA(YPLBA(5),350.)

C LEFT LINE

      CALL MOVEA(YPLBA(1),350.)
      CALL DRAWA(YPLBA(1),350.)
      CALL DRAWA(YPLBA(1),500.)

C RIGHT LINE

      CALL MOVEA(YPLBA(5),350.)
      CALL DRAWA(YPLBA(5),350.)
      CALL DRAWA(YPLBA(5),500.)
C
C    DRAW THE MIDDLE CIRCLE
C
C    TOP LINE
      CALL MOVEA(YPLMD(1),500.)
      CALL DRAWA(YPLMD(1),500.)
      CALL DRAWA(YPLMD(5),500.)

C BOTTOM LINE

      CALL MOVEA(YPLMD(1),350.)
      CALL DRAWA(YPLMD(1),350.)
      CALL DRAWA(YPLMD(5),350.)

C LEFT LINE

      CALL MOVEA(YPLMD(1),350.)
      CALL DRAWA(YPLMD(1),350.)
      CALL DRAWA(YPLMD(1),500.)

C RIGHT LINE

      CALL MOVEA(YPLMD(5),350.)
      CALL DRAWA(YPLMD(5),350.)
      CALL DRAWA(YPLMD(5),500.)

C    DRAW THE LAST CIRCLE

C    TOP LINE

      CALL MOVEA(YPLED(1),500.)
      CALL DRAWA(YPLED(1),500.)
      CALL DRAWA(YPLED(5),500.)

C BOTTOM LINE

      CALL MOVEA(YPLED(1),350.)
      CALL DRAWA(YPLED(1),350.)
      CALL DRAWA(YPLED(5),350.)
C LEFT LINE

      CALL MOVEA(YPLED(1),350.)

      CALL DRAWA(YPLED(1),350.)
      CALL DRAWA(YPLED(1),500.)
```

```
C RIGHT LINE

      CALL MOVEA(YPLED(5),350.)
      CALL DRAWA(YPLED(5),350.)
      CALL DRAWA(YPLED(5),500.)


C     NOW DRAW THE LINKS

C   LINK 1

      CALL MOVEA(YPLBA(1),500.)
      CALL DRAWA(YPLMD(5),500.)
      CALL MOVEA(YPLBA(1),350.)
      CALL DRAWA(YPLMD(5),350.)

C   LINK 2

      CALL MOVEA(YPLMD(1),500.)
      CALL DRAWA(YPLED(5),500.)
      CALL MOVEA(YPLMD(1),350.)
      CALL DRAWA(YPLED(5),350.)

C     THE GRIPPER HEAD AND THE MOVING GRIPPER PART

C     THE GRIPPER HEAD

      DUMMY = YARM(3) + 25.
      CALL MOVEA (DUMMY,375.)
      CALL DRAWA(DUMMY,350.)
      DUMMY = YARM(3) - 25.
      CALL DRAWA(DUMMY,350.)
      CALL DRAWA(DUMMY,500.)
      DUMMY = YARM(3) + 25.
      CALL DRAWA(DUMMY,500.)
      CALL DRAWA(DUMMY,375.)

C  DRAW THE GRIPPER IN THE LAST POSITION FOR ALL THE REDRAWS

      DUMMY = YARM(3) + 10.
      CALL MOVEA(DUMMY,350.)
      CALL DRAWA(DUMMY,ZARM(4))

      DUMMY = YARM(3) - 10.
      CALL DRAWA(DUMMY,ZARM(4))
      CALL DRAWA(DUMMY,350.)


      RETURN
      END


C     DRAW THE GRIPPER

      SUBROUTINE GRIPPER

C     DRAW THE DEAD WINDOW

      INTEGER ISCAN,IASC,NCHAR,BASC,KOUNT,VAL1,VAL2,RETVAL
      INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      INTEGER P29,P30,P31,P32
      REAL XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      REAL XPLBA(11),YPLBA(11),XCIRC(11),YCIRC(11)
      REAL XPLMD(11),YPLMD(11),XPLED(11),YPLED(11)
      REAL THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2,ZIN,RIN
```

```
      REAL XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      REAL XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      REAL XARM(4), YARM(4), ZARM(4), THETA, PI
      REAL XGRP(11),YGRP(11),GRIPX,GRIPY
      CHARACTER*7 TYPE,CELNAM*10,ANS*1

      COMMON/G/ SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/K/ XPLBA,YPLBA,XCIRC,YCIRC,XPLMD,YPLMD
      COMMON/L/ XPLED,YPLED,TH1,TH2,X,Y,Z,R,L1,L2
      COMMON/M/ XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      COMMON/N/ XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      COMMON/O/ XARM,YARM,ZARM,THETA,PI,GRIPX,GRIPY
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

      CALL TWINDO(751,859,672,780)
      CALL DWINDO(-400.,400.,-400.,400.)

      CALL TCSIND(3)
      CALL STROUT(920,750,'GRIPPER',7)

      IF (P18 .EQ. 1) THEN
      CALL STROUT(920,700,'      ',5)
      CALL STROUT(920,700,'CLOSE',5)

      ELSEIF (P18 .EQ. 0) THEN

      CALL STROUT(920,700,'      ',5)
      CALL STROUT(920,700,'OPEN ',5)
      ENDIF


C     CIRCUMFERENCE OF GRIPPER

      DO 100 I = 1,11
           THETA = (I-1)*PI/5.
          XGRP(I) =175.*COS(THETA)
          YGRP(I) =175.*SIN(THETA)
100   CONTINUE

      CALL MOVEA(XGRP(1),YGRP(1))
      DO 200 I=1,11
      CALL DRAWA(XGRP(I),YGRP(I))
200   CONTINUE

      CALL MOVEA(0.0,0.0)
      CALL DRAWA(GRIPX,GRIPY)

      RETURN
      END

C     THE PORTS

      SUBROUTINE PORTS

      INTEGER ISCAN,IASC,NCHAR,BASC,KOUNT,VAL1,VAL2,RETVAL
      INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      INTEGER P29,P30,P31,P32
      REAL XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      REAL XPLBA(11),YPLBA(11),XCIRC(11),YCIRC(11)
      REAL XPLMD(11),YPLMD(11),XPLED(11),YPLED(11)
      REAL THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2,ZIN,RIN
      REAL XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1,GRIPX
```

```
      REAL XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB,GRIPY
      REAL XARM(4), YARM(4), ZARM(4), THETA, PI
      CHARACTER*80 DUMMY
      CHARACTER*7 TYPE,CELNAM*10,ANS*1

      COMMON/G/ SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/K/ XPLBA,YPLBA,XCIRC,YCIRC,XPLMD,YPLMD
      COMMON/L/ XPLED,YPLED,TH1,TH2,X,Y,Z,R,L1,L2
      COMMON/M/ XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      COMMON/N/ XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      COMMON/O/ XARM,YARM,ZARM,THETA,PI,GRIPX,GRIPY
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

      CALL TWINDO(751,883,0,400)
      CALL DWINDO(0.,10.,0.,100.)

      CALL TCSIND(11)
      CALL MOVABS(750,370)
      CALL DRWABS(883,370)

      CALL MOVABS(750,0)
      CALL DRWABS(883,0)
      CALL DRWABS(883,400)
      CALL DRWABS(750,400)
      CALL DRWABS(750,0)

      CALL TCSIND(11)
      DO 20 I = 1,16
      WRITE(DUMMY,100)I
  100 FORMAT(I2)
      IY= (17-I)*22 - 7
      CALL STROUT(785,IY,DUMMY,2)
   20 CONTINUE

      CALL STROUT(785,385,'PORTS',5)
      CALL INPORT
      CALL OUTPORT

      RETURN
      END

C     WINDOW FOR THE INPORTS ONLY FOR LATER ON/OFF FACILITY

      SUBROUTINE INPORT

      INTEGER ISCAN,IASC,NCHAR,BASC,KOUNT,VAL1,VAL2,RETVAL
      INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      INTEGER P29,P30,P31,P32
      REAL XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      REAL XPLBA(11),YPLBA(11),XCIRC(11),YCIRC(11)
      REAL XPLMD(11),YPLMD(11),XPLED(11),YPLED(11)
      REAL THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2,ZIN,RIN
      REAL XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1,GRIPX
      REAL XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB,GRIPY
      REAL XARM(4), YARM(4), ZARM(4), THETA, PI
      CHARACTER*7 TYPE,CELNAM*10,ANS*1,DUMMY*80

      COMMON/G/ SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/K/ XPLBA,YPLBA,XCIRC,YCIRC,XPLMD,YPLMD
      COMMON/L/ XPLED,YPLED,TH1,TH2,X,Y,Z,R,L1,L2
```

```
COMMON/M/ XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
COMMON/N/ XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
COMMON/O/ XARM,YARM,ZARM,THETA,PI,GRIPX,GRIPY
COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
COMMON/R/ P29,P30,P31,P32


CALL TWINDO(883,958,0,400)

CALL TCSIND(11)
CALL MOVABS(883,370)
CALL DRWABS(958,370)

CALL TCSIND(4)
CALL STROUT(917,385,'DI',2)

CALL TCSIND(11)
CALL MOVABS(883,0)
CALL DRWABS(958,0)
CALL DRWABS(958,400)
CALL DRWABS(883,400)
CALL DRWABS(883,0)

CALL TCSIND(14)

IF (P1 .EQ. 1) THEN
CALL STROUT(917,345,'X',1)
ELSEIF (P1 .EQ. 0) THEN
CALL STROUT(917,345,' ',1)
ENDIF

IF (P2 .EQ. 1) THEN
CALL STROUT(917,323,'X',1)
ELSEIF (P2 .EQ. 0) THEN
CALL STROUT(917,323,' ',1)
ENDIF

IF (P3 .EQ. 1) THEN
CALL STROUT(917,301,'X',1)
ELSEIF (P3 .EQ. 0) THEN
CALL STROUT(917,301,' ',1)
ENDIF

IF (P4 .EQ. 1) THEN
CALL STROUT(917,279,'X',1)
ELSEIF (P4 .EQ. 0) THEN
CALL STROUT(917,279,' ',1)
ENDIF

IF (P5 .EQ. 1) THEN
CALL STROUT(917,257,'X',1)
ELSEIF (P5 .EQ. 0) THEN
CALL STROUT(917,257,' ',1)
ENDIF

IF (P6 .EQ. 1) THEN
CALL STROUT(917,235,'X',1)
ELSEIF (P6 .EQ. 0) THEN
CALL STROUT(917,235,' ',1)
ENDIF

IF (P7 .EQ. 1) THEN
CALL STROUT(917,213,'X',1)
ELSEIF (P7 .EQ. 0) THEN
CALL STROUT(917,213,' ',1)
ENDIF
```

```
      IF (P8 .EQ. 1) THEN
      CALL STROUT(917,191,'X',1)
      ELSEIF (P8 .EQ. 0) THEN
      CALL STROUT(917,191,' ',1)
      ENDIF

      IF (P9 .EQ. 1) THEN
      CALL STROUT(917,169,'X',1)
      ELSEIF (P9 .EQ. 0) THEN
      CALL STROUT(917,169,' ',1)
      ENDIF

      IF (P10 .EQ. 1) THEN
      CALL STROUT(917,147,'X',1)
      ELSEIF (P10 .EQ. 0) THEN
      CALL STROUT(917,147,' ',1)
      ENDIF

      IF (P11 .EQ. 1) THEN
      CALL STROUT(917,125,'X',1)
      ELSEIF (P11 .EQ. 0) THEN
      CALL STROUT(917,125,' ',1)
      ENDIF

      IF (P12 .EQ. 1) THEN
      CALL STROUT(917,103,'X',1)
      ELSEIF (P12 .EQ. 0) THEN
      CALL STROUT(917,103,' ',1)
      ENDIF

      IF (P13 .EQ. 1) THEN
      CALL STROUT(917,81,'X',1)
      ELSEIF (P13 .EQ. 0) THEN
      CALL STROUT(917,81,' ',1)
      ENDIF

      IF (P14 .EQ. 1) THEN
      CALL STROUT(917,59,'X',1)
      ELSEIF (P14 .EQ. 0) THEN
      CALL STROUT(917,59,' ',1)
      ENDIF

      IF (P15 .EQ. 1) THEN
      CALL STROUT(917,37,'X',1)
      ELSEIF (P15 .EQ. 0) THEN
      CALL STROUT(917,37,' ',1)
      ENDIF

      IF (P16 .EQ. 1) THEN
      CALL STROUT(917,15,'X',1)
      ELSEIF (P16 .EQ. 0) THEN
      CALL STROUT(917,15,' ',1)
      ENDIF

      RETURN
      END

C     WINDOW FOR THE OUTPORTS ONLY

      SUBROUTINE OUTPORT

      INTEGER ISCAN,IASC,NCHAR,BASC,KOUNT,VAL1,VAL2,RETVAL
      INTEGER P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      INTEGER P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      INTEGER P29,P30,P31,P32
      REAL XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
```

```
      REAL XPLBA(11),YPLBA(11),XCIRC(11),YCIRC(11)
      REAL XPLMD(11),YPLMD(11),XPLED(11),YPLED(11)
      REAL THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2,ZIN,RIN
      REAL XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1,GRIPX
      REAL XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB,GRIPY
      REAL XARM(4), YARM(4), ZARM(4), THETA, PI
      CHARACTER*7 TYPE,CELNAM*10

      COMMON/G/ SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/K/ XPLBA,YPLBA,XCIRC,YCIRC,XPLMD,YPLMD
      COMMON/L/ XPLED,YPLED,TH1,TH2,X,Y,Z,R,L1,L2
      COMMON/M/ XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      COMMON/N/ XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      COMMON/O/ XARM,YARM,ZARM,THETA,PI,GRIPX,GRIPY
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

      CALL TWINDO(958,1023,0,400)

      CALL TCSIND(11)
      CALL MOVABS(958,370)
      CALL DRWABS(1023,370)

      CALL TCSIND(4)
      CALL STROUT(975,385,'DO',2)

      CALL TCSIND(11)
      CALL MOVABS(958,0)
      CALL DRWABS(1023,0)
      CALL DRWABS(1023,400)
      CALL DRWABS(958,400)
      CALL DRWABS(958,0)

C
C     HERE FOR PORT TWO WHICH IS THE GRIPPER, THE GRIPPER WINDOW
C     IS CALLED AND THE GRIPPER STATUS IS INDICATED BY OPEN/CLOSE.
C
        CALL TCSIND(14)

      IF (P17 .EQ. 1) THEN
      CALL STROUT(975,345,'X',1)
      ELSEIF (P17 .EQ. 0) THEN
      CALL STROUT(975,345,' ',1)
      ENDIF

      IF (P18 .EQ. 1) THEN
      CALL STROUT(975,323,'X',1)
      CALL TWINDO(751,859,672,780)
      CALL DWINDO(-400.,400.,-400.,400.)
      CALL STROUT(920,700,'     ',5)
      CALL STROUT(920,700,'CLOSE',5)

      ELSEIF (P18 .EQ. 0) THEN

      CALL STROUT(975,323,' ',1)
      CALL TWINDO(751,859,672,780)
      CALL DWINDO(-400.,400.,-400.,400.)
      CALL STROUT(920,700,'     ',5)
      CALL STROUT(920,700,'OPEN ',5)
      ENDIF

      CALL TWINDO(958,1023,0,400)

      IF (P19 .EQ. 1) THEN
```

```
CALL STROUT(975,301,'X',1)
ELSEIF (P19 .EQ. 0) THEN
CALL STROUT(975,301,' ',1)
ENDIF

IF (P20 .EQ. 1) THEN
CALL STROUT(975,279,'X',1)
ELSEIF (P20 .EQ. 0) THEN
CALL STROUT(975,279,' ',1)
ENDIF

IF (P21 .EQ. 1) THEN
CALL STROUT(975,257,'X',1)
ELSEIF (P21 .EQ. 0) THEN
CALL STROUT(975,257,' ',1)
ENDIF

IF (P22 .EQ. 1) THEN
CALL STROUT(975,235,'X',1)
ELSEIF (P22 .EQ. 0) THEN
CALL STROUT(975,213,' ',1)
ENDIF

IF (P23 .EQ. 1) THEN
CALL STROUT(975,213,'X',1)
ELSEIF (P23 .EQ. 0) THEN
CALL STROUT(975,213,' ',1)
ENDIF

IF (P24 .EQ. 1) THEN
CALL STROUT(975,191,'X',1)
ELSEIF (P24 .EQ. 0) THEN
CALL STROUT(975,191,' ',1)
ENDIF

IF (P25 .EQ. 1) THEN
CALL STROUT(975,169,'X',1)
ELSEIF (P25 .EQ. 0) THEN
CALL STROUT(975,169,' ',1)
ENDIF

IF (P26 .EQ. 1) THEN
CALL STROUT(975,147,'X',1)
ELSEIF (P26 .EQ. 0) THEN
CALL STROUT(975,147,' ',1)
ENDIF

IF (P27 .EQ. 1) THEN
CALL STROUT(975,125,'X',1)
ELSEIF (P27 .EQ. 0) THEN
CALL STROUT(975,125,' ',1)
ENDIF

IF (P28 .EQ. 1) THEN
CALL STROUT(975,103,'X',1)
ELSEIF (P28 .EQ. 0) THEN
CALL STROUT(975,103,' ',1)
ENDIF

IF (P29 .EQ. 1) THEN
CALL STROUT(975,81,'X',1)
ELSEIF (P29 .EQ. 0) THEN
CALL STROUT(975,81,' ',1)
ENDIF

IF (P30 .EQ. 1) THEN
CALL STROUT(975,59,'X',1)
```

```
      ELSEIF (P30 .EQ. 0) THEN
      CALL STROUT(975,59,' ',1)
      ENDIF

      IF (P31 .EQ. 1) THEN
      CALL STROUT(975,37,'X',1)
      ELSEIF (P31 .EQ. 0) THEN
      CALL STROUT(975,37,' ',1)
      ENDIF

      IF (P32 .EQ. 1) THEN
      CALL STROUT(975,15,'X',1)
      ELSEIF (P32 .EQ. 0) THEN
      CALL STROUT(975,15,' ',1)
      ENDIF

      RETURN
      END

C     FOR LOCATING A STRING IN THE SCREEN

      SUBROUTINE STROUT (IX,IY,KSTR,NCHR)

      INTEGER ISCAN,IASC,NCHAR,BASC,KOUNT,VAL1,VAL2,RETVAL
      REAL XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      REAL XPLBA(11),YPLBA(11),XCIRC(11),YCIRC(11)
      REAL XPLMD(11),YPLMD(11),XPLED(11),YPLED(11)
      REAL THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2,ZIN,RIN
      REAL XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1,GRIPX
      REAL XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB,GRIPY
      REAL XARM(4), YARM(4), ZARM(4), THETA, PI
      DIMENSION KSTR(1),KAS(80)
      CHARACTER*7 TYPE,CELNAM*10,ANS*1

      COMMON/G/ SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/K/ XPLBA,YPLBA,XCIRC,YCIRC,XPLMD,YPLMD
      COMMON/L/ XPLED,YPLED,TH1,TH2,X,Y,Z,R,L1,L2
      COMMON/M/ XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      COMMON/N/ XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      COMMON/O/ XARM,YARM,ZARM,THETA,PI,GRIPX,GRIPY
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

      CALL KAM2AS (NCHR,KSTR,KAS)
      CALL MOVABS (IX,IY)
      CALL ANSTR  (NCHR,KAS)

      RETURN
      END

C     DRAWING THE FRAME OF THE THREE VIEWS

      SUBROUTINE FRAMES

      INTEGER ISCAN,IASC,NCHAR,BASC,KOUNT,VAL1,VAL2,RETVAL
      REAL XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      REAL XPLBA(11),YPLBA(11),XCIRC(11),YCIRC(11)
      REAL XPLMD(11),YPLMD(11),XPLED(11),YPLED(11)
      REAL THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2 ,ZIN,RIN
      REAL XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1,GRIPX
      REAL XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB,GRIPY
      REAL XARM(4), YARM(4), ZARM(4), THETA, PI
      DIMENSION KSTR(1),KAS(80)
      CHARACTER*7 TYPE,CELNAM*10,ANS*1
```

```
      COMMON/G/ SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/K/ XPLBA,YPLBA,XCIRC,YCIRC,XPLMD,YPLMD
      COMMON/L/ XPLED,YPLED,TH1,TH2,X,Y,Z,R,L1,L2
      COMMON/M/ XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      COMMON/N/ XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      COMMON/O/ XARM,YARM,ZARM,THETA,PI,GRIPX,GRIPY
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32

      CALL TCSIND(2)
      CALL MOVABS(0,0)
      CALL DRWABS(750,0)
      CALL DRWABS(750,777)
      CALL DRWABS(0,777)
      CALL DRWABS(0,0)

C     SIDEVIEW  WINDOW FRAME

      CALL TCSIND(10)
      CALL MOVABS(751,400)
      CALL DRWABS(1023,400)
      CALL DRWABS(1023,671)
      CALL DRWABS(751,671)
      CALL DRWABS(751,400)

C     DEAD SPACE AND GRIPPER WINDOW FRAME

      CALL MOVABS(750,672)
      CALL DRWABS(1023,672)
      CALL DRWABS(1023,777)
      CALL DRWABS(750,777)
      CALL DRWABS(750,672)


      RETURN
      END

C     CALCULATE THE POINTS FOR THE WORKSPACE IN PLAN VIEW

      SUBROUTINE WKSPACE

      INTEGER ISCAN,IASC,NCHAR,BASC,KOUNT,VAL1,VAL2,RETVAL
      REAL XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      REAL XPLBA(11),YPLBA(11),XCIRC(11),YCIRC(11)
      REAL XPLMD(11),YPLMD(11),XPLED(11),YPLED(11)
      REAL THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2,ZIN,RIN
      REAL XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1,GRIPX
      REAL XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB,GRIPY
      REAL XARM(4), YARM(4), ZARM(4), THETA, PI
      REAL XWORK(675),YWORK(675)
      CHARACTER*7 TYPE,CELNAM*10,ANS*1

      COMMON/G/ SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
      COMMON/H/ THONI,THTWI,ZIN,RIN,CODE,CELNAM,VAL5
      COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/K/ XPLBA,YPLBA,XCIRC,YCIRC,XPLMD,YPLMD
      COMMON/L/ XPLED,YPLED,TH1,TH2,X,Y,Z,R,L1,L2
      COMMON/M/ XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      COMMON/N/ XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      COMMON/O/ XARM,YARM,ZARM,THETA,PI,GRIPX,GRIPY
      COMMON/P/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
      COMMON/Q/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
      COMMON/R/ P29,P30,P31,P32
```

```
      THETA =0.0

C     CALCULATING THE 674 POINTS IN THE PLAN VIEW

      DO 300 I = 1,201
           THETA = (I-1)*PI/180.
           XWORK(I) = 650.*COS(THETA)
           YWORK(I) = 650.*SIN(THETA)
 300  CONTINUE

      DO 400 I = 200,335
           IC = I + 2
           THETA = I*PI/180.
           XWORK(IC) = 250.*COS(THETA) - 375.877
           YWORK(IC) = 250.*SIN(THETA) - 136.808
 400  CONTINUE

      DO 500 I = 238,38,-1
           IC = 576 - I
           THETA = I*PI/180.
           XWORK(IC) = 286.*COS(THETA)
           YWORK(IC) = 286.*SIN(THETA)
 500  CONTINUE

      DO 600 I = 135,1,-1
           IC = 674 - I

           THETA = I*PI/180.
           XWORK(IC) = 250.*COS(THETA) + 400.
           YWORK(IC) = 250.*SIN(THETA)
 600  CONTINUE

            XWORK(674) = XWORK(1)
            YWORK(674) = YWORK(1)

      CALL TCSIND(4)
      CALL MOVEA(650.0,0.)

      DO 800 I=2,674
      CALL DRAWA(XWORK(I),YWORK(I))
 800  CONTINUE

      RETURN
      END
```

Appendix D.

THIS SECTION CONTAINS THE FOLLOWING PROGRAMS:

(1) DEFEQP

(2) DEFSETU

(3) DISPSETU

(4) DIRECTORY

```
**************************************************************************
C     (1) DEFEEQP -  PROGRAM TO DEFINE EQUIPMENTS

      SUBROUTINE DEFEQP (BASC,NAME,POSX, POSY, POSZ)

      DIMENSION INFIL(80)
      CHARACTER INFIL*1, NAME*10,NTTY*3
      REAL X(2), Y(2), Z(2)
      REAL A,B,C,POSX,POSY,POSZ
      INTEGER ICOM,ICOMP
      INTEGER BASC,J(6)
      REAL XMIN,XMAX,YMIN,YMAX
      REAL SXMIN,SXMAX,SYMIN,SYMAX
      DATA INFIL/80*' '/
      DATA XMIN/-800.0/
      DATA XMAX/800.0/
      DATA YMIN/-650.0/
      DATA YMAX/950.0/
      DATA SXMIN/-420.0/
      DATA SXMAX/684.0/
      DATA SYMIN/-420.0/
      DATA SYMAX/684.0/


      CALL INITT(5)
      CALL SETMOD(0)
      CALL TCSMOD(1)
      CALL TCSIND(2)


      OPEN(7,FILE=NAME,STATUS='OLD')
  5   CONTINUE
      READ(7,10,END=500)(INFIL(I),I=1,80)
 10   FORMAT(80A1)
      IF (INFIL(73) .EQ. 'P') THEN
        READ (INFIL(1:3),55) NTTY
 55     FORMAT(A3)
```

```
C       IF THE DATA IS A LINE DATA - 110

        IF (NTTY .EQ.'110') THEN
          ICOM = 0
          DO 26 M = 5,72
            IF (INFIL(M:M) .EQ. ',' ) THEN
              ICOM =ICOM + 1
              J(ICOM) = M
            ENDIF

           IF (INFIL(M:M) .EQ. ';') THEN
             ICOM = ICOM + 1
             J(ICOM) = M
             GOTO 27
           ENDIF
 26      CONTINUE


 27      K = J(1) - 1
         READ(INFIL(5:K),*)X(1)
         K = J(1) + 1
         L = J(2) - 1
         READ(INFIL(K:L),*)Y(1)
         K = J(2) + 1
         L = J(3) - 1
         READ(INFIL(K:L),*)Z(1)
         K = J(3) + 1
         L = J(4) - 1
         READ(INFIL(K:L),*)X(2)
         K = J(4) + 1
         L = J(5) - 1
         READ(INFIL(K:L),*)Y(2)
         K = J(5) + 1
         L = J(6) - 1
         READ(INFIL(K:L),*)Z(2)

         X(1) = X(1) + POSX
         Y(1) = Y(1) + POSY
         Z(1) = Z(1) + POSZ
         X(2) = X(2) + POSX
         Y(2) = Y(2) + POSY
         Z(2) = Z(2) + POSZ

C      DRAW THE PLAN OF EQP. - EITHER AS THE MAJOR OR MINOR VIEW

       IF (BASC. EQ. 80)  THEN
       CALL TWINDO(0,750,0,750)
       CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
       ELSE
       CALL TWINDO(751,1023,400,672)
       CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
       ENDIF

       CALL MOVEA(X(1),Y(1))
       CALL DRAWA(X(2),Y(2))

C      DRAW THE SIDEVIEW OF EQP. - EITHER AS THE MAJOR OR MINOR VIEW

       IF (BASC. EQ. 80)  THEN
       CALL TWINDO(751,1023,400,672)
       CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)
       ELSE
       CALL TWINDO(0,750,0,750)
       CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)
       ENDIF
```

```
      CALL MOVEA(Y(1),Z(1))
      CALL DRAWA(Y(2),Z(2))

        GOTO 5

C     IF THE DATA IS POINT DATA - 116


      ELSEIF (NTTY .EQ. '116') THEN
        DO 30 M = 5,72
         IF (INFIL(M:M) .EQ. ',') THEN
          ICOMP =ICOMP + 1
          J(ICOMP) = M
         ENDIF

        IF (INFIL(M:M) .EQ. ';') THEN
         ICOMP = ICOMP + 1
         J(ICOMP) = M
         GOTO 40
        ENDIF
 30   CONTINUE

 40   K = J(1) - 1
      READ(INFIL(5:K),*)A

      K = J(1) + 1
      L = J(2) - 1
      READ(INFIL(K:L),*)B

      K = J(2) + 1
      L = J(3) - 1
      READ(INFIL(K:L),*)C

        A = A + POSX
        B = B + POSX
        C = C + POSX

C     DRAW THE PLAN OF EQP. - EITHER AS THE MAJOR OR MINOR VIEW

      IF (BASC. EQ. 80)  THEN
      CALL TWINDO(0,750,0,750)
      CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
      ELSE
      CALL TWINDO(751,1023,400,672)
      CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
      ENDIF

      CALL MOVEA(A,B)
      CALL DRAWA(A,B)

C     DRAW THE SIDEVIEW OF EQP. - EITHER AS THE MAJOR OR MINOR VIEW

      IF (BASC. EQ. 80)  THEN
      CALL TWINDO(751,1023,400,672)
      CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)
      ELSE
      CALL TWINDO(0,750,0,750)
      CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)
      ENDIF

      CALL MOVEA(B,C)
      CALL DRAWA(B,C)


       GOTO 5
      ELSE
      GOTO 5
```

```
        ENDIF

      ELSE
       GOTO 5
      ENDIF

500   CLOSE(7)

      RETURN
      END

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

C     (2) DEFSETU - PROGRAM TO ARRANGE WORKCELLS

C     THIS PROGRAM DEFINES THE WORKCELL LAYOUT BY CREATING THE DATA FILE
C     WITH THE EXTENSION .CEL

      PROGRAM DEFSETUP

      CHARACTER*10 FNAME,DNAME,INFIL*1,NTTY*3
      CHARACTER*2 C,AEXT*3,EQPN*10
      REAL X(2),Y(2),Z(2),POSX,POSY,POSZ,R,PI
      REAL NX(2),NY(2),NZ(2)
      DIMENSION INFIL(80)
      REAL D,E,F,J(7)
      INTEGER Q,T,M,ICOM,ICOMP,K,L
      DATA INFIL/80*' '/
      DATA ICOM/0/
      DATA ICOMP/0/
      DATA PI/3.1428/
      DATA AEXT/'DAT'/
      C = '  '

170   CALL CLS(31)
      CALL TTOUT(0,1,23,'SYSTEM FOR IBM 7545 ROBOT',25,25,
     *31)
      CALL TTOUT(0,3,12,
     *'GRAPHICAL SIMULATION AND INTERACTIVE PROGRAMMING',
     *48,48,31)
      CALL TTOUT(0,5,25,'ARRANGE WORKCELL MENU',21,21,27)
      CALL TTOUT(0,6,25,'--------------------',21,21,31)
      CALL TTOUT(0,9,15, 'ARRANGE WORKCELL = A ',21,21,31)
      CALL TTOUT(0,10,15,'PREVIOUS MENU    = P ',21,21,31)
      CALL TTOUT(0,12,15,'ENTER SELECTION (A/P)',21,21,31)
      CALL LOCATE(0,12,48)
      READ(*,31)CHOICE
  31  FORMAT(A1)

      IF (CHOICE .EQ. 'A') THEN
      GOTO 200
      ELSEIF (CHOICE .EQ. 'P') THEN
      GOTO 300
      ELSE
      CALL TTOUT(0,14,15,
     *'CHARACTER NOT DEFINED HIT ENTER KEY TO CONTINUE',47,47,31)
      READ(*,*)
      GOTO 170
      ENDIF


200   CALL CLS(31)
      CALL TTOUT(0,1,23,'SYSTEM FOR IBM 7545 ROBOT',25,25,
     *31)
      CALL TTOUT(0,3,12,
     *'GRAPHICAL SIMULATION AND INTERACTIVE PROGRAMMING',
     *48,48,31)
```

```
      CALL TTOUT(0,5,25,'ARRANGE WORKCELL',16,16,27)
      CALL TTOUT(0,6,25,'---------------',16,16,31)
      CALL TTOUT(0,8,10,
     *'ENTER NAME OF WORKCELL(.CEL IS ASSUMED)',39,39,31)
      CALL TTOUT(0,9,10,'(FIVE CHARACTERS ONLY)',22,22,31)
      CALL LOCATE(0,8,50)
      READ(*,25)FNAME

C     CONCATENATE THE WORKCELL NAME WITH .CEL AND CHECK IF IT EXISTS.

      AEXT = 'CEL'
      CALL CONCA(FNAME,AEXT)
      OPEN(11,FILE=FNAME,STATUS='OLD',ERR=190)
      CLOSE (11)
      CALL TTOUT(0,10,10,'WORKCELL NAME ALREADY EXISTS !!!',32,32,31)
      CALL TTOUT(0,11,10,'HIT ENTER KEY TO CONTINUE',25,25,31)
      CALL LOCATE(0,11,45)
      READ(*,*)
      GOTO 200

190   CLOSE (11)
      OPEN(8,FILE=FNAME,STATUS='NEW')

      CALL TTOUT(0,11,10,
     *'HOW MANY EQUIPMENTS ARE PLACED IN THIS WORKCELL?',48,48,31)
      CALL TTOUT(0,12,10,'(MAXIMUM OF EIGHT)',18,18,31)
      CALL LOCATE(0,11,60)

      READ(*,120)N
120   FORMAT(I2)

       DO 500 Q = 1,N

C      TO GET THE SPACING OF THE LINES ON THE SCREEN

110   IF (Q  .LE. 2) THEN
         T = 8+Q*5
         CALL TTOUT(0,T,10,
     *'GIVE NAME OF EQUIPMENT     (.EQP IS ASSUMED)',44,44,31)
         WRITE (C,'(I2)')Q
         CALL TTOUT(0,T,33,C,2,2,31)
         CALL LOCATE(0,T,56)
         READ(*,25)EQPN

         T=9+5*Q
         M=10+5*Q

      ELSEIF (Q .LE. 5) THEN

         IF (Q .EQ. 3)  THEN
      CALL CLS(31)
      CALL TTOUT(0,1,23,'SYSTEM FOR IBM 7545 ROBOT',25,25,
     *31)
      CALL TTOUT(0,3,12,
     *'GRAPHICAL SIMULATION AND INTERACTIVE PROGRAMMING',
     *48,48,31)
         ENDIF
         T = -8 + (Q*5)
         CALL TTOUT(0,T,10,
     *'GIVE NAME OF EQUIPMENT     (.EQP IS ASSUMED)',44,44,31)
         WRITE (C,'(I2)')Q
         CALL TTOUT(0,T,33,C,2,2,31)
         CALL LOCATE(0,T,56)
         READ(*,25)EQPN

         T=-6+5*Q
         M=-5+5*Q
```

```
      ELSE
      IF (Q .LE. 8) THEN

         IF (Q .EQ. 6) THEN
       CALL CLS(31)
       CALL TTOUT(0,1,23,'SYSTEM FOR IBM 7545 ROBOT',25,25,
      *31)
       CALL TTOUT(0,3,12,
      *'GRAPHICAL SIMULATION AND INTERACTIVE PROGRAMMING',
      *48,48,31)
           ENDIF
           T = -23 + (Q*5)
           CALL TTOUT(0,T,10,
      *'GIVE NAME OF EQUIPMENT     (.EQP IS ASSUMED)',44,44,31)
           WRITE (C,'(I2)')Q
           CALL TTOUT(0,T,33,C,2,2,31)
           CALL LOCATE(0,T,56)
           READ(*,25)EQPN

           T=-21+5*Q
           M=-20+5*Q
       ENDIF
       ENDIF

C      THE DATA IS READ IN FREE FORMAT.

       CALL TTOUT(0,T,10,'GIVE X, Y, Z, R VALUES',21,21,31)
       CALL TTOUT(0,M,10,
      *'(ALL VALUES IN ONE LINE SEPERATED BY A SPACE)',45,45,31)
       CALL TTOUT(0,M+1,10,'                                                ',
      *42,42,31)
       CALL LOCATE(0,T,35)
       READ(*,*)POSX,POSY,POSZ,R
       R = R*(PI/180.0)

       AEXT ='EQP'
       CALL CONCA(EQPN,AEXT)
       OPEN(7,FILE=EQPN,STATUS='OLD',ERR=100)
       GOTO 5

C      CHECKS WHETHER THE FILE EXISTS OR EXCEEDS FIVE CHARACTERS

100    CALL TTOUT(0,M+1,10,'FILE DOES NOT EXIST/EXCEEDS FIVE CHARACTERS',
      *42,42,31)
       CALL TTOUT(0,T,10,'                                                  ',
      *44,44,31)
       CALL TTOUT(0,M,10,
      *'                                                ',50,50,31)
       GOTO 110

  5    CONTINUE
       READ(7,10,END=500)(INFIL(I),I=1,80)
 10    FORMAT(80A1)

       IF (INFIL(73) .EQ. 'P') THEN
         READ (INFIL(1:3),55) NTTY
 55      FORMAT(A3)

C        IF THE DATA IS A LINE DATA - 110

         IF (NTTY .EQ.'110') THEN
           ICOM = 0
           DO 26 M = 5,72
             IF (INFIL(M:M) .EQ. ',' ) THEN
               ICOM =ICOM + 1
               J(ICOM) = M
```

```
            ENDIF

           IF (INFIL(M:M) .EQ. ';') THEN
             ICOM = ICOM + 1
             J(ICOM) = M
             GOTO 27
           ENDIF
 26      CONTINUE

 27      K = J(1) - 1
         READ(INFIL(5:K),*)X(1)
         K = J(1) + 1
         L = J(2) - 1
         READ(INFIL(K:L),*)Y(1)
         K = J(2) + 1
         L = J(3) - 1
         READ(INFIL(K:L),*)Z(1)
         K = J(3) + 1
         L = J(4) - 1
         READ(INFIL(K:L),*)X(2)
         K = J(4) + 1
         L = J(5) - 1
         READ(INFIL(K:L),*)Y(2)
         K = J(5) + 1
         L = J(6) - 1
         READ(INFIL(K:L),*)Z(2)

C  AFTER READING THE X Y Z VALUES FORM THE EQP DATA FILE THEY ARE
C  WRITTEN TO THE SETUP DATA FILE WITH THE NEW X Y Z  VALUES.

          IF ( R .EQ. 0.0) THEN

          DO 450  I=1,2
          X(I)  = X(I) + POSX
          Y(I)  = Y(I) + POSY
          Z(I)  = Z(I) + POSZ
450       CONTINUE

          ELSE

          DO  400 I = 1, 2

         NX(I) = X(I)*COS(R) + Y(I)*SIN(R)
         NY(I) = -X(I)*SIN(R) + Y(I)*COS(R)
         NZ(I) = Z(I)

          X(I)  = NX(I) + POSX
          Y(I)  = NY(I) + POSY
          Z(I)  = NZ(I) + POSZ

400       CONTINUE

          ENDIF

      WRITE(8,15)X(1),Y(1),Z(1),X(2),Y(2),Z(2)
15    FORMAT('110',1H,,F8.3,1H,,F8.3,1H,,F8.3,1H,,
     *F8.3,1H,,F8.3,1H,,F8.3,1H;)

      GOTO 5

C       IF THE DATA IS A POINT DATA - 116

       ELSEIF (NTTY .EQ. '116') THEN
         DO 80 M = 5,72
          IF (INFIL(M:M) .EQ. ',') THEN
           ICOMP =ICOMP + 1
           J(ICOMP) = M
```

```
         ENDIF

        IF (INFIL(M:M) .EQ. ';') THEN
         ICOMP = ICOMP + 1
         J(ICOMP) = M
         GOTO 40
        ENDIF
 80   CONTINUE

 40   K = J(1) - 1
      READ(INFIL(5:K),*)A

      K = J(1) + 1
      L = J(2) - 1
      READ(INFIL(K:L),*)B

      K = J(2) + 1
      L = J(3) - 1
      READ(INFIL(K:L),*)C

         IF ( R .EQ. 0) THEN

         D = D + POSX
         E = E + POSY
         F = F + POSZ

         ELSE

         D = D + POSX
         E = E + POSY
         F = F + POSZ

         D = D*COS(R) + E*SIN(R)
         E = -D*SIN(R) + E*COS(R)
         F =  F

         ENDIF

      WRITE(8,16)D,E,F
16    FORMAT(F8.3,1H,,F8.3,1H,,F8.3,1H;)

      ELSE
        GOTO 5
        ENDIF
         ELSEIF (INFIL(73) .EQ. 'T') THEN
              GOTO 500
         ELSE
           GOTO 5
        ENDIF

 500  CONTINUE
      CLOSE(7)
      CLOSE(8)

 25   FORMAT(A5)

 50   CALL CLS(0)

 300  END


C      TTOUT

      SUBROUTINE TTOUT(PAGE,ROW,COLUMN,BUF,LENGTH,SIZE,NRATT)

      INTEGER * 2 PAGE,ROW,COLUMN,SIZE,NRATT
```

```
      CHARACTER * 1 BUF(SIZE)
      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME,NKNAME
      INTEGER * 2 L
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
      COMMON /B/ L

      CALL LOCATE(PAGE,ROW,COLUMN)

      DO 10 I = 1,LENGTH
         CALL NRCHAT(PAGE,BUF(I),NRATT,1)
10    CONTINUE

      RETURN
      END

C      CONCA

      SUBROUTINE  CONCA (AFILE,AEXT)


      CHARACTER *20 AFILE
      CHARACTER*3 AEXT


100   DO 130 I = 1,20
       IF(AFILE(I:I) .EQ. ' ') GOTO 140
130   CONTINUE

140   AFILE(I:I) = '.'
      DO 150 J =1,3
      AFILE(I+J:I+J) = AEXT(J:J)
150   CONTINUE

C     CAPITALIZE

      DO 200 K = I,I+1
      IF(AFILE(K:K) .GE. 'a' .AND. AFILE(K:K) .LE. 'z' ) then
          AFILE(K:K) = CHAR (ICHAR (AFILE(K:K)) -32 )
      ENDIF
 200  CONTINUE

      RETURN
      END

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

C     (3) DISPSETU - PROGRAM TO DISPLAY WORKCELL SETUP.

C     THIS PROGRAM DISPSETU.FOR GETS THE NAME OF THE EQUIPMENT
C     TO BE DISPLAYED, AND CREATES THE BATCH FILE DISEQP.BAT


      PROGRAM DISPSETU

      REAL XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      REAL XPLBA(11),YPLBA(11),XCIRC(11),YCIRC(11)
      REAL XPLMD(11),YPLMD(11),XPLED(11),YPLED(11)
      REAL THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2
      REAL XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      REAL XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      REAL XARM(4), YARM(4), ZARM(4), THETA, PI,ASPECT
      REAL X1,X2,Y1,Y2,XC,YC,DELX,DELY,DUMMY,XAVG,YAVG
      INTEGER BASC
      CHARACTER*7 TYPE,CELNAM*10,ANS*1,AEXT*3

      COMMON/A/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
```

```
      COMMON/B/ XPLBA,YPLBA,XCIRC,YCIRC,XPLMD,YPLMD,XPLED,YPLED
      COMMON/C/ THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2
      COMMON/D/ XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      COMMON/E/ XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      COMMON/F/ XARM,YARM,ZARM,THETA,PI
      COMMON/H/ BASC,CELNAM

      XMIN  = -800.0
      XMAX  =  800.0
      YMIN  = -650.0
      YMAX  =  950.0
      SXMIN = -420.0
      SXMAX =  684.0
      SYMIN = -420.0
      SYMAX =  684.0
      KOUNT =  1
      XARM(1) = 0.0
      YARM(1) = 0.0
      ZARM(1) = 325.0
      XARM(2) = 400.
      YARM(2) = 0.0
      ZARM(2) = 325.0
      XARM(3) = 650.0
      YARM(3) = 0.0
      ZARM(3) = 325.0
      XARM(4) = 650.0
      YARM(4) = 0.0
      ZARM(4) = 250.0

      CALL SETMON(0)
      CALL TCSMOD(0)

130   CALL CLS(31)
      CALL TTOUT(0,1,23,'SYSTEM FOR IBM 7545 ROBOT',25,25,
     *31)
      CALL TTOUT(0,3,12,
     *'GRAPHICAL SIMULATION AND INTERACTIVE PROGRAMMING',
     *48,48,31)
      CALL TTOUT(0,5,25,'DISPLAY WORKCELL LAYOUT',23,23,27)
      CALL TTOUT(0,6,25,'-----------------------',23,23,31)
      CALL TTOUT(0,9,15, 'DISPLAY LAYOUT = D   ',21,21,31)
      CALL TTOUT(0,10,15,'PREVIOUS MENU  = P   ',21,21,31)
      CALL TTOUT(0,12,15,'ENTER SELECTION (D/P)',21,21,31)
      CALL LOCATE(0,12,48)
      READ(*,31)CHOICE
   31 FORMAT(A1)

      IF (CHOICE .EQ. 'D') THEN
      GOTO 200
      ELSEIF (CHOICE .EQ. 'P') THEN
      GOTO 300
      ELSE
      CALL TTOUT(0,14,15,
     *'CHARACTER NOT DEFINED HIT ENTER KEY TO CONTINUE',47,47,31)
      READ(*,*)
      GOTO 130
      ENDIF

C     THE CHOICE OF DISPLAY WORKCELL LAYOUT.

200   CALL CLS(31)
      CALL TTOUT(0,1,23,'SYSTEM FOR IBM 7545 ROBOT',25,25,
     *31)
      CALL TTOUT(0,3,12,
     *'GRAPHICAL SIMULATION AND INTERACTIVE PROGRAMMING',
     *48,48,31)
      CALL TTOUT(0,5,25,'DISPLAY WORKCELL LAYOUT',23,23,27)
```

```
      CALL TTOUT(0,6,25,'----------------------',23,23,31)
      CALL TCSTXT(10,15,1,2,44,
     *'ENTER THE NAME OF WORKCELL (.CEL IS ASSUMED)')
      CALL TCSTXT(11,15,1,2,25,'(MAXIMUN FIVE CHARACTERS)')
      CALL LOCATE(0,11,61)
      READ(*,35)CELNAM
   35 FORMAT(A5)

C     THE NAME OF THE WORKCELL IS CONCATENATED WITH THE
C     EXTENSION .CEL AND THE FILE IS OPENED.

      AEXT = 'CEL'
      CALL CONCA(CELNAM,AEXT)
      OPEN (4,FILE=CELNAM, STATUS='OLD',ERR=100)
      CLOSE(4)
      GOTO 170

C     IF THE FILE DOES NOT EXIST THE MESSAGE IS DISPLAYED.

100   CALL TCSTXT(14,15,1,2,24,'FILE DOES NOT EXIST !!! ')
      CALL TCSTXT(15,15,1,2,25,'HIT ENTER KEY TO CONTINUE')
      READ(*,*)
      GOTO 200

C      SELECTING BETWEEN THE TWO VIES FOR THE MAJOR VIEW

170   CALL TCSTXT(13,15,1,2,22,'CHOOSE MAJOR VIEW(P/S)')
      CALL TCSTXT(14,15,1,2,25,'(PLAN = P, SIDEVIEW = S )')
      CALL LOCATE(0,14,50)
      CALL TCSKEY(ISCAN,IASC,NCHAR)
      BASC = IASC

C     THE GRAPHICS DISPLAYED IS CALLED

      CALL DSWAP(2)
      CALL INITT(5)
      CALL REDRAW

C     FOR THE FIRST ZOOM

120   IF ( KOUNT .EQ. 1) THEN
      CALL LOCATE(0,13,48)
      CALL STROUT(190,100,'DO YOU WANT TO ZOOM (Y/N)',25)
      CALL TCSKEY(ISCAN,IASC,NCHAR)
      KOUNT =  KOUNT + 1
      GOTO 110
      ELSE

C     FOR SUBSEQUENT ZOOMS

      CALL LOCATE(0,13,48)
      CALL STROUT(190,100,'DO YOU WANT MORE DETAILS (Y/N)',30)
      CALL TCSKEY(ISCAN,IASC,NCHAR)
      GOTO 110
      ENDIF

 110  IF (IASC .EQ. 89) THEN
      CALL  ZOOM
      GOTO 120
      ELSE
      GOTO 600
      ENDIF

C     CALL TO THE TEXT SCREEN

600   CALL DSWAP(0)
```

```
300   END

C       ZOOM

      SUBROUTINE ZOOM

      REAL XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      REAL XPLBA(11),YPLBA(11),XCIRC(11),YCIRC(11)
      REAL XPLMD(11),YPLMD(11),XPLED(11),YPLED(11)
      REAL THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2
      REAL XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      REAL XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      REAL XARM(4), YARM(4), ZARM(4), THETA, PI,ASPECT
      REAL X1,X2,Y1,Y2,XC,YC,DELX,DELY,DUMMY,XAVG,YAVG
      INTEGER BASC
      CHARACTER*7 TYPE,CELNAM*10,ANS*1

      COMMON/A/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/B/ XPLBA,YPLBA,XCIRC,YCIRC,XPLMD,YPLMD,XPLED,YPLED
      COMMON/C/ THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2
      COMMON/D/ XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      COMMON/E/ XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      COMMON/F/ XARM,YARM,ZARM,THETA,PI
      COMMON/H/BASC,CELNAM

C     IF MAJOR VIEW IS PLAN VIEW

      IF (BASC .EQ. 80)
      CALL TWINDO(0,750,0,750)
      CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
      CALL STROUT
     *     (55,65,'SELECT NEW WINDOW CORNER PRESS SPACE BAR',40)
      CALL VCURSR (ICHAR,X1,Y1)
      CALL STROUT
     *     (55,40,'SELECT OPPOSITE CORNER   PRESS SPACE BAR ',40)
      CALL VCURSR (ICHAR,X2,Y2)

      XMIN = X1
      IF (X2 .LT. X1)  XMIN=X2
      YMIN =Y1
      IF (Y2 .LT. Y1)  YMIN=Y2
      DELX = ABS(X2-X1)
      DELY = ABS(Y2-Y1)
      YAVG = (Y2+Y1)/2.0
      XAVG = (X2+X1)/2.0
      IF (DELX .EQ. 0.0 ) DELX =0.001
      IF (DELY .EQ. 0.0 ) DELY =0.001
      ASPECT = DELY /DELX
      IF (ASPECT .LT. 1.0) DELY = DELY *1.0/ASPECT
      IF (ASPECT .GT. 1.0) DELX = DELX *ASPECT /1.0
      YMIN =YAVG -DELY/2.0
      XMIN =XAVG -DELX/2.0
      XMAX = XMIN + DELX
      YMAX = YMIN + DELY

      CALL REDRAW

      ELSE

C     IF MAJOR VIEW IS SIDE VIEW

      CALL TWINDO(0,750,0,750)
      CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)
      CALL STROUT
     *     (55,65,'SELECT NEW WINDOW CORNER PRESS SPACE BAR',40)
      CALL VCURSR (ICHAR,X1,Y1)
      CALL STROUT
```

```
*     (55,40,'SELECT OPPOSITE CORNER   PRESS SPACE BAR',40)
      CALL VCURSR (ICHAR,X2,Y2)

      SXMIN = X1
      IF (X2 .LT. X1)  SXMIN=X2
      SYMIN =Y1
      IF (Y2 .LT. Y1)  SYMIN=Y2
      DELX = ABS(X2-X1)
      DELY = ABS(Y2-Y1)
      YAVG = (Y2+Y1)/2.0
      XAVG = (X2+X1)/2.0
      IF (DELX .EQ. 0.0 ) DELX =0.001
      IF (DELY .EQ. 0.0 ) DELY =0.001
      ASPECT = DELY /DELX
      IF (ASPECT .LT. 1.0) DELY = DELY *1.0/ASPECT
      IF (ASPECT .GT. 1.0) DELX = DELX *ASPECT /1.0
      SYMIN =YAVG -DELY/2.0
      SXMIN =XAVG -DELX/2.0
      SXMAX = SXMIN + DELX
      SYMAX = SYMIN + DELY

      CALL REDRAW
      ENDIF

C     TO PAN THE DRAWING

C     IF MAJOR VIEW IS PLAN VIEW

      IF (BASC .EQ. 80) THEN
      CALL TWINDO(0,750,0,750)
      CALL DWINDO(XMIN,XMAX,YMIN,YMAX)

      CALL STROUT(205,80,'DO YOU WANT TO PAN (Y/N)',24)
      CALL TCSKEY(ISCAN,IASC,NCHAR)
      IF (IASC .EQ. 89) THEN
      GOTO 100
      ELSE
      GOTO 200
      ENDIF

100   CALL STROUT(55,40,'INDICATE NEW CENTER, PRESS SPACE BAR',36)
      CALL VCURSR (ICHAR,XC,YC)
      IF (ICHAR .NE. 77 .AND. ICHAR .NE. 109) GOTO 15
 12   CALL STROUT
     *(55,60,'ENTER NEW (X,Y) FOR CENTER OF MODEL --> ',40)
      CALL TSEND
      READ (*,*,END =12) XC,YC

 15   DELX =XMAX - XMIN
      DELY =YMAX - YMIN
      XMIN = XC - DELX/2.0
      YMIN = YC - DELY/2.0
      XMAX = XMIN + DELX
      YMAX = YMIN + DELY
      CALL REDRAW

      ELSE

C     IF MAJOR VIEW IS SIDE VIEW

      CALL TWINDO(0,750,0,750)
      CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)
      CALL STROUT(205,80,'DO YOU WANT TO PAN (Y/N)',24)
      CALL TCSKEY(ISCAN,IASC,NCHAR)
      IF (IASC .EQ. 89) THEN
      GOTO 300
      ELSE
```

```
      GOTO 200
      ENDIF

300   CALL STROUT(55,40,'INDICATE NEW CENTER, PRESS SPACE BAR',36)
      CALL VCURSR (ICHAR,XC,YC)
      IF (ICHAR .NE. 77 .AND. ICHAR .NE. 109) GOTO 25
 22   CALL STROUT
     *(55,60,'ENTER NEW (X,Y) FOR CENTER OF MODEL --> ',40)
      CALL TSEND
      READ (*,*,END =22) XC,YC

 25   DELX =SXMAX - SXMIN
      DELY =SYMAX - SYMIN
      SXMIN = XC - DELX/2.0
      SYMIN = YC - DELY/2.0
      SXMAX = SXMIN + DELX
      SYMAX = SYMIN + DELY

      CALL REDRAW

      ENDIF

 200  RETURN
      END


C      TO REDRAW THE WHOLE PICTURE  - STANDARD VIEW AND WITH WORKCELL

       SUBROUTINE REDRAW

      REAL XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      REAL XPLBA(11),YPLBA(11),XCIRC(11),YCIRC(11)
      REAL XPLMD(11),YPLMD(11),XPLED(11),YPLED(11)
      REAL THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2
      REAL XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      REAL XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      REAL XARM(4), YARM(4), ZARM(4), THETA, PI
      REAL M(4),N(4),XWORK(675),YWORK(675)
      REAL CX(2), CY(2),CZ(2),NX(2),NY(2),NZ(2)
      REAL D,E,F,ND,NE,NF
      INTEGER ICOM,ICOMP,Q,J(6)
      INTEGER ISCAN,IASC,NCHAR,BASC,KOUNT
      CHARACTER*7 TYPE,ANS*1
      CHARACTER INFIL*1,NAME*20 ,AFILE*20,CELNAM*10
      CHARACTER CHOICE*1,DUMMY*1,NTTY*3 ,AEXT*3
      DIMENSION INFIL(80)


      COMMON/A/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
      COMMON/B/ XPLBA,YPLBA,XCIRC,YCIRC,XPLMD,YPLMD,XPLED,YPLED
      COMMON/C/ THONI,THTWI,TH1,TH2,X,Y,Z,R,L1,L2
      COMMON/D/ XBT,YBT,XBB,YBB,XMT1,YMT1,XMB1,YMB1
      COMMON/E/ XMT2,YMT2,XMB2,YMB2,XET,YET,XEB,YEB
      COMMON/F/ XARM,YARM,ZARM,THETA,PI
      COMMON/H/BASC,CELNAM

      DATA INFIL/80*' '/
      CALL SETMON(1)
      CALL TCSMOD(1)
      CALL TCSIND(14)

C     CHECKING TO SEE IF A WORKCELL HAS TO BE DISPLAYED

      OPEN (7,FILE=CELNAM, STATUS='OLD',ERR=100)

5     CONTINUE
```

```
      READ(7,10,END=200)(INFIL(I),I=1,80)
10    FORMAT(80A1)
        READ (INFIL(1:3),55) NTTY
  55    FORMAT(A3)

C       IF THE DATA IS A LINE DATA - 110

        IF (NTTY .EQ.'110') THEN
          ICOM = 0
          DO 26 Q = 5,72
            IF (INFIL(Q:Q) .EQ. ',' ) THEN
              ICOM =ICOM + 1
              J(ICOM) = Q
            ENDIF

           IF (INFIL(Q:Q) .EQ. ';') THEN
             ICOM = ICOM + 1
             J(ICOM) = Q
             GOTO 27
           ENDIF
  26      CONTINUE


          K = J(1) - 1
          READ(INFIL(5:K),*)CX(1)
          K = J(1) + 1
          L = J(2) - 1
          READ(INFIL(K:L),*)CY(1)
          K = J(2) + 1
          L = J(3) - 1
          READ(INFIL(K:L),*)CZ(1)
          K = J(3) + 1
          L = J(4) - 1
          READ(INFIL(K:L),*)CX(2)
          K = J(4) + 1
          L = J(5) - 1
          READ(INFIL(K:L),*)CY(2)
          K = J(5) + 1
          L = J(6) - 1
          READ(INFIL(K:L),*)CZ(2)

C       DRAW THE PLAN OF EQP. - EITHER AS THE MAJOR OR MINOR VIEW

        IF (BASC .EQ. 80) THEN
        CALL TWINDO(0,750,0,750)
        CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
        ELSE
        CALL TWINDO(751,1023,400,672)
        CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
        ENDIF

        CALL MOVEA(CX(1),CY(1))
        CALL DRAWA(CX(2),CY(2))

C       DRAW THE SIDEVIEW OF EQP. - EITHER AS THE MAJOR OR MINOR VIEW

        IF (BASC .EQ. 80) THEN
        CALL TWINDO(751,1023,400,672)
        CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)
        ELSE
        CALL TWINDO(0,750,0,750)
        CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)
        ENDIF

        CALL MOVEA(CY(1),CZ(1))
        CALL DRAWA(CY(2),CZ(2))
```

```
        GOTO 5

C       IF THE DATA IS A POINT DATA - 116

       ELSEIF (NTTY .EQ. '116') THEN
         ICOMP = 0
         DO 30 Q = 5,72
          IF (INFIL(Q:Q) .EQ. ',') THEN
           ICOMP =ICOMP + 1
           J(ICOMP) = Q
          ENDIF

         IF (INFIL(Q:Q) .EQ. ';') THEN
          ICOMP = ICOMP + 1
          J(ICOMP) = Q
          GOTO 40
         ENDIF
  30   CONTINUE

  40   K = J(1) - 1
       READ(INFIL(5:K),*)D

       K = J(1) + 1
       L = J(2) - 1
       READ(INFIL(K:L),*)E

       K = J(2) + 1
       L = J(3) - 1
       READ(INFIL(K:L),*)F

C      DRAW THE PLAN OF EQP. - EITHER AS THE MAJOR OR MINOR VIEW

       IF (BASC .EQ. 80) THEN
       CALL TWINDO(0,750,0,750)
       CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
       ELSE
       CALL TWINDO(751,1023,400,672)
       CALL DWINDO(XMIN,XMAX,YMIN,YMAX)
       ENDIF

       CALL MOVEA(D,E)
       CALL DRAWA(D,E)

C      DRAW THE SIDEVIEW OF EQP. - EITHER AS THE MAJOR OR MINOR VIEW

       IF (BASC .EQ. 80) THEN
       CALL TWINDO(751,1023,400,672)
       CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)
       ELSE
       CALL TWINDO(0,750,0,750)
       CALL DWINDO(SXMIN,SXMAX,SYMIN,SYMAX)
       ENDIF

       CALL MOVEA(E,F)
       CALL DRAWA(E,F)

         GOTO 5
        ELSE
        GOTO 5
          ENDIF

 200   CLOSE(7)

 100   CALL PLANSIDE

       CALL FRAMES
       RETURN
```

```
      END


C
C     SUBROUTINE PLANSIDE, FRAMES, AND MKSPACE ARE THE NEXT THREE
C     SUBROUTINES IN THIS PROGRAM. THEY ARE THE SAME SUBROUTINES
C     IN THE PROGRAM DEPUTY1 AND HENCE NOT REPEATED IN THIS PRINTOUT.
C

C     TTOUT

      SUBROUTINE TTOUT(PAGE,ROW,COLUMN,BUF,LENGTH,SIZE,MRATT)

      INTEGER * 2 PAGE,ROW,COLUMN,SIZE,MRATT
      CHARACTER * 1 BUF(SIZE)
      CHARACTER * 1 BLANK(80)
      CHARACTER * 20 NAME,MKNAME
      INTEGER * 2 L

      CALL LOCATE(PAGE,ROW,COLUMN)

      DO 10 I = 1,LENGTH
         CALL MRCHAT(PAGE,BUF(I),MRATT,1)
10    CONTINUE

      RETURN
      END


C     FOR LOCATING A STRING IN THE SCREEN

      SUBROUTINE STROUT (IX,IY,KSTR,NCHR)

      DIMENSION   KSTR(1),KAS(80)

      CALL KAM2AS (NCHR,KSTR,KAS)
      CALL MOVABS (IX,IY)
      CALL ANSTR    (NCHR,KAS)

      RETURN
      END


C      CONCA


      SUBROUTINE  CONCA (AFILE,AEXT)


      CHARACTER *20 AFILE
      CHARACTER*3 AEXT


 100  DO 130 I = 1,20
       IF(AFILE(I:I) .EQ. ' ') GOTO 140
 130  CONTINUE

140   AFILE(I:I) = '.'
      DO 150 J =1,3
      AFILE(I+J:I+J) = AEXT(J:J)
 150  CONTINUE

C     CAPITALIZE

      DO 200 K = I,I+1
      IF(AFILE(K:K) .GE. 'a' .AND. AFILE(K:K) .LE. 'z' ) then
```

```
          AFILE(K:K) = CHAR (ICHAR (AFILE(K:K)) -32 )
        ENDIF
 200    CONTINUE

        RETURN
        END

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

C       (4) DIRECTOR - PROGRAM TO DISPLAY THE DIRECTORY

C       PROGRAM DIRECTOR GIVES THE DIRECTORY LISTING OF ALL THE
C       EQUIPMENT AND WORKCELL DATA FILES AVAILABLE.


        PROGRAM DIRECTOR

        CHARACTER *1 CHOICE,DNAME*9


 250    CALL CLS(31)
        CALL CSRON
        CALL CLRBUF
        CALL TTOUT(0,1,23,'SYSTEM FOR IBM 7545 ROBOT',25,25,
       *31)
        CALL TTOUT(0,3,12,
       *'GRAPHICAL SIMULATION AND INTERACTIVE PROGRAMMING',
       *48,48,31)
        CALL TTOUT(0,6,25,'DIRECTORY LISTING',17,17,27)
        CALL TTOUT(0,7,25,'-----------------',17,17,31)
        CALL TTOUT(0,11,21,'Equipment Directory = E',23,23,31)
        CALL TTOUT(0,12,21,'Workcell Directory  = W',23,23,31)
        CALL TTOUT(0,15,24,'Select Option (E/W)',19,19,27)
        CALL LOCATE(0,15,45)
        READ(*,31)CHOICE
   31   FORMAT(A1)

        IF (CHOICE .EQ. 'E') THEN
        GOTO 200
        ELSEIF (CHOICE .EQ. 'W') THEN
        GOTO 300
        ELSE
        CALL TTOUT(0,17,17,
       *'CHARACTER NOT DEFINED HIT ENTER KEY TO CONTINUE',47,47,31)
        READ(*,*)
        GOTO 250
        ENDIF

C       CREATES THE FLAG DIRCT1 AND CONTROL RETURNS TO THE
C       BATCH FILE WHICH CAUSES THE DIRECTORY LISTING OF ALL
C       EQUIPMENTS TO BE DISPLAYED.

  200   OPEN(10,FILE='DIRCT1.FLG',STATUS='NEW')
        CLOSE(10)
        GOTO 1000

C       THE SAME AS ABOVE, THE LIST OF WORKCELLS IS DISPLAYED.

  300   OPEN(11,FILE='DIRCT2.FLG',STATUS='NEW')
        CLOSE(11)
        GOTO 1000

 1000   STOP
        END


C       SUBROUTINE TTOUT
```

```
      SUBROUTINE TTOUT(PAGE,ROW,COLUMN,BUF,LENGTH,SIZE,WRATT)

      INTEGER * 2 PAGE,ROW,COLUMN,SIZE,WRATT
      CHARACTER * 1 BUF(SIZE)
      CHARACTER * 1 BLANK(80),C1,C2,C3,C4
      CHARACTER * 20 NAME,WKNAME
      INTEGER * 2 L
      COMMON /A/ NAME,BLANK,C1,C2,C3,C4
      COMMON /B/ L

      CALL LOCATE(PAGE,ROW,COLUMN)

      DO 10 I = 1,LENGTH
         CALL WRCHAT(PAGE,BUF(I),WRATT,1)
10    CONTINUE

      RETURN
      END
```

The vita has been removed from the scanned document