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
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

1.0 Chapter 1. Introduction ................................................ 1  
  1.1 Background ........................................................ 1  
  1.2 Simulators ......................................................... 2  
  1.3 Problem Statement ................................................... 3  
  1.4 Objective .......................................................... 5  

2.0 Chapter 2. Literature Review ............................................ 7  
  2.1 Introduction ........................................................ 7  
  2.2 Commercial Systems .................................................. 7  
  2.3 Classification ....................................................... 12  
    2.3.1 Simulation Objective ........................................... 13  
    2.3.2 Display Capability ............................................. 14  
    2.3.3 Simulation Capability ......................................... 14  
  2.4 Summary ......................................................... 15  

3.0 Chapter 3. Methodology .............................................. 16  
  3.1 Introduction ....................................................... 16
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.2</td>
<td>System Hardware/Software</td>
<td>17</td>
</tr>
<tr>
<td>3.3</td>
<td>System Specifications</td>
<td>21</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Menus and Batch file.</td>
<td>21</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Graphic Display Modes</td>
<td>23</td>
</tr>
<tr>
<td>3.3.3</td>
<td>Define/Display Equipment and Workcell</td>
<td>26</td>
</tr>
<tr>
<td>3.3.4</td>
<td>Graphics Display</td>
<td>27</td>
</tr>
<tr>
<td>4.0</td>
<td>Chapter 4. Development of the Graphic Simulator</td>
<td>30</td>
</tr>
<tr>
<td>4.1</td>
<td>Introduction</td>
<td>30</td>
</tr>
<tr>
<td>4.2</td>
<td>Menu Programs</td>
<td>31</td>
</tr>
<tr>
<td>4.3</td>
<td>Display Programs</td>
<td>33</td>
</tr>
<tr>
<td>4.4</td>
<td>Definition Programs</td>
<td>41</td>
</tr>
<tr>
<td>4.5</td>
<td>Assembly Routines</td>
<td>43</td>
</tr>
<tr>
<td>5.0</td>
<td>Chapter 5. Analysis of the Simulator</td>
<td>44</td>
</tr>
<tr>
<td>5.1</td>
<td>Introduction</td>
<td>44</td>
</tr>
<tr>
<td>5.2</td>
<td>Assumptions and Limitations</td>
<td>44</td>
</tr>
<tr>
<td>5.2.1</td>
<td>Pseudo Compiler</td>
<td>44</td>
</tr>
<tr>
<td>5.2.2</td>
<td>Graphics Driver</td>
<td>45</td>
</tr>
<tr>
<td>5.2.3</td>
<td>Data Files</td>
<td>46</td>
</tr>
<tr>
<td>5.3</td>
<td>Test Programs</td>
<td>47</td>
</tr>
<tr>
<td>6.0</td>
<td>Chapter 6. Conclusions and Recommendations</td>
<td>49</td>
</tr>
<tr>
<td>7.0</td>
<td>Bibliography</td>
<td>51</td>
</tr>
<tr>
<td>Appendix A.</td>
<td></td>
<td>54</td>
</tr>
</tbody>
</table>

Table of Contents
List of Illustrations

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>This is the Menus Chart</td>
<td>18</td>
</tr>
<tr>
<td>2</td>
<td>List of Files developed for the Simulator.</td>
<td>19</td>
</tr>
<tr>
<td>3</td>
<td>This is the Main Menu</td>
<td>22</td>
</tr>
<tr>
<td>4</td>
<td>This is the Main Simulation Menu and Graphical Simulation Menu</td>
<td>24</td>
</tr>
<tr>
<td>5</td>
<td>This is the Views Menu and Workcell Layout Menu.</td>
<td>25</td>
</tr>
<tr>
<td>6</td>
<td>Function of codes used in the Graphics Display Program</td>
<td>36</td>
</tr>
<tr>
<td>7</td>
<td>Codes corresponding to the AML/E commands</td>
<td>38</td>
</tr>
</tbody>
</table>
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 capabilities, 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 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.

<table>
<thead>
<tr>
<th>Present Simulator</th>
<th>Proposed Simulator</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Shows plan view only.</td>
<td>1. Shows both plan and side view, with options to switch between the two as major/minor view.</td>
</tr>
</tbody>
</table>
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

Chapter 2. Literature Review
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 beginning 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 access 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. Derby's GRASP developed at Rensselaer 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 capabilities 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 wireframe 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.
SYSTEM FOR IBM 7545 ROBOT
GRAPHICAL SIMULATION AND PROGRAMMING

MAIN MENU

MAIN SIMULATION MENU

+ ........................................ +

GRAPHICAL SIMULATION MENU WORKCELL LAYOUT MENU

+ ............. +

VIEWS MENU

SIMULATION

Figure 1. This is the Menus Chart
<table>
<thead>
<tr>
<th>FILE</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>GSAPP.BAT</td>
<td>Initialize system, transfer control between programs, and quit system.</td>
</tr>
<tr>
<td>FIRST.EXE</td>
<td>Displays Main Menu, and writes the flags to direct control to appropriate programs.</td>
</tr>
<tr>
<td>BETWEEN.EXE</td>
<td>Displays the Main Simulation Menu.</td>
</tr>
<tr>
<td>MENU2.EXE</td>
<td>Displays Graphical Simulation Menu and passes control to the program MAIN or the Alter Views menu as chosen.</td>
</tr>
<tr>
<td>MENU3.EXE</td>
<td>Displays Workcell Layout Menu and directs to define/display equipment, define/display workcell setup, or display directory as chosen.</td>
</tr>
<tr>
<td>ALTER.EXE.</td>
<td>Displays Views Menu, and directs to/Zoom/ /Switch/Normal/Simulation.</td>
</tr>
<tr>
<td>DISPEQP.EXE</td>
<td>Define/Display equipment.</td>
</tr>
<tr>
<td>DEFSETUP.EXE</td>
<td>Arrange Workcell layout.</td>
</tr>
<tr>
<td>DISPSETU.EXE</td>
<td>Display Workcell layout.</td>
</tr>
<tr>
<td>DIRECTORY.EXE</td>
<td>Display directory of equipment and workcells.</td>
</tr>
<tr>
<td>MAIN.EXE</td>
<td>Graphics driver program that controls the graphics display program.</td>
</tr>
<tr>
<td>DEPUTY1.EXE</td>
<td>Graphics display program that does all the computations and displays on the graphics screen.</td>
</tr>
<tr>
<td>MOVE</td>
<td>A subroutine in DEPUTY1 that computes the final locations of the arms of the robot in plan and side view.</td>
</tr>
<tr>
<td>PLANSIDE</td>
<td>A subroutine in DEPUTY1 that does the actual draw on the graphics screen.</td>
</tr>
<tr>
<td>INPORT/OUTPORT</td>
<td>A subroutine in DEPUTY1 that controls the display of the input and output port on the screen</td>
</tr>
</tbody>
</table>

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
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 take 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 go 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 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 chosen 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 Fig-
ure 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 pro-
gram 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 pa-
rameters 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 in-
dicated 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 irrespec-
tive of whether the plan view or side view is displayed there. The user is prompted to
<table>
<thead>
<tr>
<th>CODE</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Calls the subroutine MOVE which computes the values of the intermediate and final coordinates. It then calls PLANSIDE to draw these positions.</td>
</tr>
<tr>
<td>2</td>
<td>Calls the subroutine INPORT.</td>
</tr>
<tr>
<td>3</td>
<td>Calls the subroutine OUTPORT. This also causes the gripper status to be displayed in the gripper window.</td>
</tr>
<tr>
<td>4</td>
<td>Calls the subroutine HOME and draws the robot and workcell in the home position in the view and coordinates selected by the user.</td>
</tr>
<tr>
<td>5</td>
<td>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.</td>
</tr>
<tr>
<td>6</td>
<td>Calls the subroutine SWITCH to switch between the plan and side view in the standard coordinates.</td>
</tr>
<tr>
<td>7</td>
<td>Calls the subroutine STANDARD to display the standard view which is the plan in the major view using the standard coordinates.</td>
</tr>
<tr>
<td>8</td>
<td>Calls the subroutine REDRAW which redraws the entire screen in the current location of robot using the current coordinates and view.</td>
</tr>
</tbody>
</table>

*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
calculate 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 subse-
quent 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.
<table>
<thead>
<tr>
<th>AML/E Command</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>DELAY</td>
<td>1</td>
</tr>
<tr>
<td>DPMOVE</td>
<td>1</td>
</tr>
<tr>
<td>GRASP</td>
<td>3</td>
</tr>
<tr>
<td>PMOVE</td>
<td>1</td>
</tr>
<tr>
<td>RELEASE</td>
<td>3</td>
</tr>
<tr>
<td>ZMOVE</td>
<td>1</td>
</tr>
<tr>
<td>LINEAR</td>
<td>-</td>
</tr>
<tr>
<td>PAYLOAD</td>
<td>-</td>
</tr>
<tr>
<td>ZONE</td>
<td>-</td>
</tr>
<tr>
<td>WAITI</td>
<td>2</td>
</tr>
<tr>
<td>WRITEO</td>
<td>2</td>
</tr>
<tr>
<td>BRANCH</td>
<td>-</td>
</tr>
<tr>
<td>TESTC</td>
<td>-</td>
</tr>
<tr>
<td>TESTI</td>
<td>2</td>
</tr>
<tr>
<td>TESTP</td>
<td>-</td>
</tr>
<tr>
<td>BREAKPOINT</td>
<td>2</td>
</tr>
<tr>
<td>COMPC</td>
<td>-</td>
</tr>
<tr>
<td>DECR</td>
<td>-</td>
</tr>
<tr>
<td>INCR</td>
<td>-</td>
</tr>
<tr>
<td>SETC</td>
<td>-</td>
</tr>
<tr>
<td>GETPART</td>
<td>1</td>
</tr>
<tr>
<td>NEXTPART</td>
<td>-</td>
</tr>
<tr>
<td>PREVPART</td>
<td>-</td>
</tr>
<tr>
<td>SETPART</td>
<td>-</td>
</tr>
</tbody>
</table>

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 from 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 input 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 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.
Chapter 5. Analysis of the Simulator

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.

Assumptions and Limitations

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
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


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 LOAD.BAT GOTO LOAD
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 M/CONTROLLER
:COMA
COMMAND /C COMA.BAT
DEL COMA.BAT
GOTO FIRST
REM: TO PASS CONTROL TO THE BATCH FILE FOR GENERATE CROSS LISTING
:REF
COMMAND /C REF.BAT
DEL REF.BAT
GOTO FIRST
REM: TO DISPLAY MAIN SIMULATION MENU
:BETNEEN
IF EXIST WORK DEL WORK
IF EXIST BETNEEN.FLG DEL BETNEEN.FLG
BETNEEN
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 MENUTH.FLG GOTO MENUTH
IF EXIST MENUTH.FLG GOTO MENUTH
REM: TO DISPLAY GRAPHICAL SIMULATION MENU
:MENUTH
IF EXIST MENUTH.FLG DEL MENUTH.FLG
IF EXIST SIM.FLG DEL SIM.FLG
IF EXIST SIM1.FLG DEL SIM1.FLG
MENUTH
IF EXIST BETNEEN.FLG GOTO BETNEEN
IF EXIST SIM.FLG GOTO SIM
IF EXIST ALT.FLG GOTO ALT
REM: TO DISPLAY SIMULATION MENU
:SIM
IF EXIST SIM.FLG DEL INFO
COMMAND
BOSS1
REM: AFTER SIMULATING WITH NORMAL VIEW GO BACK TO MENU 2
IF EXIST SIM.FLG GOTO MENUTH
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 MENUTH.FLG GOTO MENUTH
IF EXIST SIM1.FLG GOTO SIM
REM: TO DISPLAY WORKCELL LAYOUT MENU
:MENUTH

Appendix A. 55
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

Appendix A. 56
C (2) DSMAp.ASM - THE ASSEMBLY LANGUAGE PROGRAM TO SWITCH MONITORS.

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

title dsmap.asm
page 132

comment /

usage call dsmap (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 doesn'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 dsmap

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 ds100
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

Appendix A. 57
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, 00010000B
  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 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, CSRFF, CSRON, INKEY,
C INTER, MENU, TTOUT, DSNAP.

CHARACTER * 1 BLANK, 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, A4)
10 REWIND (1)
CLOSE (1)
CALL DSHAP(D)
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(0,6,16,'Filename (.AML assumed):',24,24,26)
   CALL TTOUT(0,7,21,'Compiler options:',17,17,26)
   J = 0
   DO 1 I = 22-L,26
      J = J + 1
      CALL TTOUT(0,6,38+J,NAME(I:13),L-1,L,26)
   CONTINUE
   IF (C1 .NE. ' ' .AND. C2 .NE. ' ') THEN
      CALL TTOUT(0,7,39,C1,1,1,26)
      CALL TTOUT(0,7,4D,C2,1,1,26)
   ENDIF
   IF (C3 .NE. ' ' .AND. C4 .NE. ' ') THEN
      CALL TTOUT(0,7,|1,CS;1,1,26]
      CALL TTOUT(U,7,14Z,C'|1,1,1,26]
   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(0,15,21,'FN2 - Edit/Teach a Program',27,27,31)
CALL TTOUT(0,16,21,'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
GOTO 100

C F10 IS PRESSED - SIMULATE PROGRAM GRAPHICALLY

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

Appendix B.
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 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(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 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,A20,4'A1')
100 FORMAT(I2,1X,A20,4'A1')
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.

Appendix B.
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('COMAID')

CLOSE(2)
RETURN
END

C SUBROUTINE AMLCOMA (THRU-F6)
C
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 AMLCOMA

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
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

Appendix B.
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,IO,INKEY,TTOUT.

SUBROUTINE MENU

CHARACTER * 1 CHAR,ALPHA(26),NUMER(10)
INTEGER * 2 J,KYCOD1,KYCOD2
CHARACTER * 1 BLANK(80),C1,C2,C5,C4
CHARACTER * 20 NAME,AFILE
INTEGER A E 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','
*','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(0,4,24,'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)
RENVIND (1)

2 FORMAT(1,2)
C INITIALIZE NAME TO BLANKS.
DO 45 I = 1,20

Appendix B. 65
NAME(I:I) = ' '  
CONTINUE
C1 = ' '  
C2 = ' '  
C3 = ' '  
C4 = ' '  
L = 0
LOCATE CURSOR, AND DISPLAY VALID INPUTS ON THE SCREEN AND
CONCATENATE THE NAME.
DO 10 I = 1,200
L = L + 1
IF (L .GE. 21) GOTO 20
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 TTDUT(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)
ENDIF
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
CONTINUE
CALL CLRBUF
POSITION CURSOR TO READ COMPILER OPTIONS.
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'
  CALL CLRBUF
  CALL INKEY(KYCOD1,KYCOD2)
  CALL CLRBUF
  CALL TTOUT(0,24,1,BLANK,80,80,31)
  CALL CSRGN
  GOTO 34
ELSE
  C4 = CHAR
  CALL TTOUT(0,5,42,C4,1,1,26)
ENDIF

36 IF (C2 .EQ. C4) THEN
  C5 = '
  C6 = ':'
ENDIF

WRITE THE PROGRAM NAME AND COMPILER OPTIONS INTO THE FILE IRPSO.

WRITE (1,30) L,NAM(!L-1),C1,C2,C3,C4
REIND (1)
CLOSE(1)

30 FORMAT(I2,1X,A20,4A1)

RETURN
END

SUBROUTINE INTER (THRU-F9)

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

Appendix B.
C BLOCK DATA
C THIS BLOCK DEFINES THE ARRAY BLANK, WHICH IS AN ARRAY OF BLANK 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 ROW AND COLUMN OF THE DISPLAY.
C
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, WRCHAR.

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 WRCHAR(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
C PARAMETER DEFINITION:
C KYCOD1 - ASCII CODE OF KEY
C KYCOD2 - EXTENDED CHARACTER CODE OF KEY

Appendix B.
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 = 29) THEN
  J = J - 1
ELSEIF (KYCOD1 = 8 .AND. KYCOD2 = 14) THEN
  J = J - 1
ELSEIF (KYCOD1 = 127 .AND. KYCOD2 = 14) THEN
  J = J - 1
ELSEIF (KYCOD1 = 0 .AND. KYCOD2 = 75) THEN
  J = J - 1
ELSEIF (KYCOD1 = 33) THEN
  CHAR = '!' 
ELSEIF (KYCOD1 = 64) THEN
  CHAR = '"'
ELSEIF (KYCOD1 = 35) THEN
  CHAR = '#' 
ELSEIF (KYCOD1 = 36) THEN
  CHAR = '$'
ELSEIF (KYCOD1 = 37) THEN
  CHAR = '%'
ELSEIF (KYCOD1 = 94) THEN
  CHAR = '
'
ELSEIF (KYCOD1 = 38) THEN
  CHAR = '
'
ELSEIF (KYCOD1 = 40) THEN
  CHAR = '
'
ELSEIF (KYCOD1 = 41) THEN
  CHAR = '
'
ELSEIF (KYCOD1 = 95) THEN
  CHAR = '
'
ELSEIF (KYCOD1 = 45) THEN
  CHAR = '
'
ELSEIF (KYCOD1 = 123) THEN
  CHAR = '
'
ELSEIF (KYCOD1 = 125) THEN
  CHAR = '
'
ELSEIF (KYCOD1 = 39) THEN
  CHAR = '
'
ELSEIF (KYCOD1 = 50) THEN
  CHAR = '
'
ELSEIF (KYCOD1 = 47) THEN
  CHAR = '
'
ELSEIF (KYCOD1 = 47) THEN
  CHAR = '
'
ELSEIF (KYCOD1 = 92) THEN
  CHAR = '
'
ELSE
  J = 0
ENDIF
SUBROUTINE IDDSTG(AFILE,AEXT)

CHARACTER*20 AFILE
CHARACTER*1 AEXT

100 DO 130 I = 1,20
   IF(AFILE(I:1) .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 = 1,آ+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 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/WORKCELL.
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,DSMAP,INKEY,TTOUT

CHARACTER*1 BLANK(80),C1,C2,C3,C4
CHARACTER*20 NAME,HNAME
INTEGER*2 J,L,KYCOD1,KYCOD2,BASC
COMMON /A/ NAME,BLANK,C1,C2,C3,C4
COMMON /B/ L
COMMON /J/ THONI,THONII,ZIN,RIN,CODE,CELNAM,VAL
COMMON /J/ XMIN,XMAX,YMIN,YMAX,SMIN,SMAX,SYMIN,SYMAX
COMMON /O/ XARM,YARM,ZARM,THETA,PI,GRIpx,GRIpy
COMMON /G/ SS,SLPAY,PAR1,PART2,PAYO1D,VAL1,VAL2,RETI1,BSAC

XMIN = -800.0
XMAX = 800.0
YMIN = -650.0
YMAX = 950.0
CALL TOUT(0,12,24,"All selections using alphabets",30,30,28)
CALL TOUT(0,14,24,"must be in CAPITAL LETTERS",26,26,28)
CALL TOUT(0,16,24,"Press any key to Continue....",29,29,28)
CALL INKEY(KYCOD1,KYCOD2)
CALL CLS(31)
CALL CSRON
CALL CLRBUF

CALL TOUT(0,1,23,"SYSTEM FOR IBM 7545 ROBOT",25,25,31)
CALL TOUT(0,3,12,"GRAPHICAL SIMULATION AND INTERACTIVE PROGRAMMING",48,48,31)
CALL TOUT(0,5,25,"MAIN SIMULATION MENU",20,18,27)
CALL TOUT(0,6,25,"---------------------",20,18,31)
CALL TOUT(0,8,24,"",1,1,31)
CALL TOUT(0,9,24,"Select a function:",19,19,27)
CALL TOUT(0,12,21,"F01 - Previous Menu",19,19,31)
CALL TOUT(0,13,21,"F02 - Simulate Program",22,22,31)
CALL TOUT(0,14,23,"F03 - Define Equipment/Arrange Workcell",39,39,31)
CALL TOUT(0,20,24,"Select Option==>",17,17,27)
CALL LOCATE(0,20,41)
CALL CLRBUF
CALL INKEY(KYCOD1,KYCOD2)
IF (KYCOD1 .NE. 0) THEN
  CALL CSROFF
  CALL TOUT(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 TOUT(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
F2 IS SELECTED - CALL MAIN SIMULATION MENU. THE FLAG MENUTH.FLG IS CREATED FOR THE BATCH FILE GASP TO TRANSFER CONTROL TO THAT 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 DSNAP(0)
   GO TO 100

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

BLOCK DATA

THIS BLOCK DEFINES THE ARRAY BLANK, WHICH IS AN ARRAY OF BLANK 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

SUBROUTINE TTOUT

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

SUBROUTINES CALLED BY THIS PROGRAM ARE :
LOCATE, WRCHAT.
SUBROUTINE TTOUT(PAGE,ROM,COLUMN,BUF,LENGTH,SIZE,MRAIIT)

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

CALL LOCATE(PAGE,ROM,COLUMN)

DO 10 I = 1,LENGTH
   CALL MRCHAT(PAGE,BUF(I),MRAIIT)
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,CSROR,INKEY,SETMEN,CONCA,MODSTG

CHARACTER * 1 BLANK(80)
CHARACTER * 20 NAME,CELNAME*10
INTEGER * 2 J,L,KYCOD1,KYCOD2,BASC
COMMON /A/ NAME,BLANK
COMMON /B/ L
COMMON /H/ THONI,THM1,ZIN,ZIN,CODE,CELNAME,VALS
COMMON /J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SMAX,SXMIN,SYM1X,SYM1X
COMMON /O/ XARM,YARM,ZARM,THETA,PI,GRIPE,GRIPE
COMMON /G/ SS,SLPAY,PAR1,PART2,PAYOLD,VAL1,VAL2,RIVAL,BASC

XMIN = -800.0
XMAX = 800.0
YMIN = -650.0
YMAX = 950.0
SXMIN = -420.0
SXMAX = 684.0
SYMIN = -420.0
SYM1X = 684.0
KOUT = 1
GRIPE = 175.0
GRIPE = 0.0
L = 0
XVAL = 650.0
YVAL = 0.0
ZVAL = 0.0
RIVAL = 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,CELNA

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

66  CALL DSWAP(0)
   CALL CSRON
   CALL CLRBUF

C DISPLAY MAIN MENU IF NOFILENAME EXISTS IN THE FILE "MORK", C
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(0,4,25,'--------------------------',25,31)

C CHECK FILE CELNAM FOR WORKCELL NAME. IF NO NAME THEN DISPLAY C
C THE OPTIONS ONLY. IF NAME EXISTS THEN SHOW NAME OF WORKCELL NAME C
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 workcall does not exist',28,28,27)
   GOTO 65
802  CODE = 4.0
   CALL DSWAP(0)
   ENDF

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(0,4,25,'--------------------------',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

Appendix B.

75
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)
GOTO 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 :
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

Appendix B. 76
COMMON /B/ L
COMMON /H/ THONI,THWIN,ZIN,RIN,CODE,CELNAM,VAL5
COMMON /J/ XMIN,YMIN,XMAX,YMAX,SYMIX,SYMIX,SYMIX,SYMIX,SYMIX
COMMON /0/ XARM,YARM,ZARM,THETA,PI,GRIPX,GRIPY
COMMON /G/ SS,SLPAY,PAR1,PAR2,PAYOLO,VAL1,VAL2,RETVAL,BASC

DATA NUMER/ '0', '1', '2', '3', '4', '5', '6', '7', '8', '9'/

800 CALL CLS(31)
CALL CSDON
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 THO',8,8,31)
   CALL TTOUT(0,6,30,'--------',8,8,31)
   CALL TTOUT(0,8,14,'Horkce11 Name (.CEL assumed)',28,28,26)
   CALL TTOUT(0,10,24,'Select e 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 Paremeters',30,30,31)
   CALL TTOUT(0,20,24,'Select Option===>',17,17,27)
OPEN (31,FILE='HORK',STATUS='NEH')
WRITE (31,2)
REHIND (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
   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 CSDOFF
         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)
      END IF
   END IF
10 CONTINUE

Appendix B.  77
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)
ENDIF

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

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(12,1X,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 )
200 CONTINUE
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, CELNAME*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:

LOCATE, WRECHAT.

SUBROUTINE TTOUT(PAGE, ROM, COLUMN, BUF, LENGTH, SIZE, MRATT)

INTEGER *2 PAGE, ROM, COLUMN, SIZE, MRATT, L
CHARACTER * 1 BUF(SIZE), BLANK(80)
CHARACTER * 20 NAME, CELNAME*10
COMMON /A/ NAME, BLANK
COMMON /B/ L

CALL LOCATE(PAGE, ROM, COLUMN)

DO 10 I = 1, LENGTH
   CALL WRECHAT(PAGE, BUF(I), MRATT, L)
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, CELNAME*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

Appendix B. 79

Appendix B. 79
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. 94) THEN
    CHAR = '
ELSEIF (KYCOD1 .EQ. 38) THEN
    CHAR = '&'
ELSEIF (KYCOD1 .EQ. 40) THEN
    CHAR = '('
ELSEIF (KYCOD1 .EQ. 41) 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

SUBROUTINE MODSTG

SUBROUTINE MODSTG (AFILE, AEXT)

C THIS SUBROUTINE CONCATENATES TWO STRINGS
C AFILE FIRST STRING
C AFIX SECOND STRING
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 = I,1

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

C CAPITALIZE

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

RETURN
ENDC

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

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,CSR0N,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(0,15,21,'FN2 - Define/Display Individual Equipment',40,40,31)
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(0,20,31)
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',*
  40,40,31)
  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='NEH')
CLOSE(24)
GO TO 100

C  F2 IS PRESSED - DISPLAY INDIVIDUAL EQUIPMENTS

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

C  F3 IS PRESSED - ARRANGE WORKCELL

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

C  F4 IS PRESSED - DISPLAY WORKCELL

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

C  F5 IS PRESSED - DIRECTORY OF AVAILABLE EQUIPMENTS

ELSEIF (KYCOD2 .EQ. 63) THEN
CALL CLRBUF
OPEN(28,FILE='DIR.FLG',STATUS='NEH')
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 CSROFF
CALL TTOUT(0,24,1,BLANK,80,80,31)
GOTO 11
ENDIF

100 CLOSE(1)
CLOSE(2)
CALL CLS(0)
CALL CSROFF
STOP
END

C  BLOCK DATA
C THIS BLOCK DEFINES THE ARRAY BLANK, WHICH IS AN ARRAY OF BLANK 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 ROW AND COLUMN OF THE DISPLAY.

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

SUBROUTINE TTOUT(PAGE,ROM,COLUMN,BUF,LENGTH,SIZE,NRATT)

INTEGER * 2 PAGE,ROM,COLUMN,SIZE,NRATT
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,ROM,COLUMN)
DO 10 I = 1,LENGTH
   CALL WRCHAT(PAGE,BUF(I),NRATT,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 SELECTION. THIS PROGRAM IS CALLED BY THE BATCH FILE AFTER THE ALT.FLG HAS BEEN FLAGGED. THIS PROGRAM PUTS UP THE VIEWS MENU FOR THE USER TO CHOOSE THE DESIRED VIEW AND SIZE, AND SIMULATE. THIS PROGRAM IS LINKED WITH THE PROGRAM DEPUTY1 FOR STANDARD VIEW TO BE REDRAWN, FOR SWITCH VIEW, AND ZOOM.

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

CHARACTER * 1 BLANK(80)
CHARACTER * 20 NAME,CELNAM=10
INTEGER *2 J,L,KYCOD1,KYCOD2
INTEGER BASC
REAL XVAL,YVAL,ZVAL,RVAL,CODE,THONI,THTMI,ZIN,RIN
REAL XMIN,XMAX,YMIN,YMAX,XMIN,SYM,XMIN,SYM
COMMON /H/ THONI,THTMI,ZIN,RIN,CODE,CELNAM,VALS
COMMON /J/ XMIN,XMAX,YMIN,YMAX,XMIN,SYM,XMIN,SYM
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
SMIN = -420.0
SMAX = 684.0
SYMIN = -420.0
SYMAX = 684.0
BASC = 80

OPEN(25,FILE='HORK',STATUS='OLD',ERR=104)
READI25,170)CELNAM
170 FORMAT(3X,A10)
CLOSE(24)

104 OPEN(22,FILE='INFO',STATUS='NEH')WRITE(22,250)IBASC,XMIN,XMAX,YMIN,YMAX,SMIN,SMAX,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)
CLOSE(22)

CODE = 4.0
CALL DEPUTY1(XVAL•YVAL,ZVAL,RVAL)
CALL D$NAP(0)
103 CALL D$NAP(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 Play/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 CRSOFF
  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='MENUTN.FLG',STATUS='NEM')
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 DSMAPO
   GO TO 103

C F3 IS PRESSED- SWITCH VIEW
ELSEIF (KYCOD2 .EQ. 61) THEN
   CALL CLRBUF
   CODE = 6.0
   CALL DEPUTY1(XVAL,YVAL,ZVAL,RVAL)
   CALL DSMAPO
   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 DSMAPO
   GO TO 103

C F5 IS PRESSED- SIMULATE PROGRAM
ELSEIF (KYCOD2 .EQ. 63) THEN
   CALL CLRBUF
   OPEN(15,FILE='SIM1.FLG',STATUS='NEM')
   CLOSE(15)
go to 100
ELSE
   CALL CSROFF
   CALL TTOUT(0,40,1,'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,60,80,31)
go to 11
ENDIF
100 CALL CLS
   CALL CSRON
   END

C BLOCK DATA
C THIS BLOCK DEFINES THE ARRAY BLANK, WHICH IS AN ARRAY OF BLANK CHARACTERS.
BLOCK DATA
CHARACTER * 1 BLANK (80)

Appendix B.
Character * 20 Name, CELNAM * 10
Integer * 2 L

Common /H/ THONI, THTHI, 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, HRCHAT.

SUBROUTINE TTOUT(PAGE, ROM, COLUMN, BUF, LENGTH, SIZE, HRATT)
Integer * 2 PAGE, ROM, COLUMN, SIZE, HRATT
Character * 1 BUF(SIZE)
Character * 1 BLANK(80)
Character * 20 NAME, CELNAM * 10
Integer * 2 L

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

Call LOCATE(PAGE, ROM, COLUMN)

Do 10 I = 1, LENGTH
   Call HRCHAT(PAGE, BUF(I), HRATT, I)
10 Continue

Return

End

Appendix B.
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 CONVAL(I) = Array containing constant number
C CONVVAL(I) = Array containing constant value
C COUNT(I) = Array containing counter number
C IONUM(I) = Array containing DI/DD value
C LAST = Last statement number to be executed
C LASTSUB = 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 PALLEL(I) = Array containing pallet number
C PALPITS(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
THE MAIN PROGRAM OPEN ALL FILES AND CALLS SUBROUTINE SYS TO START INTERACTIVE SIMULATION.

SUBROUTINES CALLED BY THIS PROGRAM ARE:

CLS,CSROFF,FINI,INKEY,SYS,TTOUT.

THE USER'S AM/E PROGRAM NAME IS READ FROM THE FILE IRPS00.
THIS FILE IS CREATED BY THE PSEUDO-COMPILER.
THE PROGRAM THEN CONCATENATES THE EXTENSIONS FOR THE OTHER FILES CREATED BY THE COMPILER, AND OPENS THEM UP.
CLOSE (24) 
CONTINUE

101  

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, XMINT, XMAX, YMINT, YMAX, SXINT, SXMAX, SYINT, SYMAX

250 FORMAT (1X, I2, 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

Appendix C. 89
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) = NAMCON(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 .CNI, 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,F0.2,5U(' '))
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=NA2OB(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 DSMAP(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(KYC0D1,KYC0D2)

CALL FINI
CALL CLS(0)
CALL CSROF
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 :
CLRBUF,CLS,CSRON.

SUBROUTINE FINI

INTEGER COUNT(50),CNYVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100),VAL5
INTEGER SS,SLPAY,PAR1,PAR2,VAL1,VAL2,RETPAY, 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,TTHNI,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,PAR1,PAR2,RETPAY,VAL1,VAL2, RETPAY, BASC
COMMON/H/ THONI,TTHNI,ZIN,RIN, CODE, CELNAM, VAL5
COMMON/J/ XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYM3IN,SYM3AX,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 NAZAML - 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, HOT, PAL, PALL, PART, SENS, SLOW, SUB1, SUB2, TTOUT, WHERE, WRITER.

SUBROUTINE SYS(NAZAML, 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, BACS
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
CHARACTER* 24 NAZAML, CELNAM*10
CHARACTER* 1 BLANK(80)

DIMENSION LASIBI(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, XPAR, YPAR, ZPAR, XPAR
COMMON/E/ XPNT, YPNT, ZPNT, RPNT, XAGG, YAGG, ZAGG, RAGG
COMMON/F/NUNI, NUM2, NUM3, NUM4, NUM5, NUM6, XPAR, YPAR, ZPAR, RPAR
COMMON/G/ SS, SLPAY, PART1, PART2, PAYOLD, VAL1, VAL2, RETVAL, BACS
COMMON/H/ THONI, THTMI, ZIN, RIN, CODE, CELNAM, VAL5
COMMON/J/ XMIN, XMAX, YMIN, YMAX, XMIN, XMAX, YMIN, YMAX
COMMON/K/ 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
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
  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 OBTAINED 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=MM) 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 (8,500,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(NUM5) = 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)

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

ELSEIF (ICODE .EQ. 6) THEN
  NN = 0
  NUM = NUM + 1
  NN = NN + 1
  READ (3,100,END=1000) ICI,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
    MM5 = MM5 + 1
    READ (7,700,REC=ICI) L1,L2,L3,L4,L5,L6,(L(I),I=1,5)
    L(NN) = NUM
    WRITE (7,700,REC=ICI) L1,L2,L3,L4,L5,L6,(L(I),I=1,5)
    GO TO 12
  ENDIF
  GOTO 11

IF THE VARIABLE IS AN AGGREGATE, READ IN THE FOUR VALUES, AND
ASSIGN THEM TO THE ARRAYS XAGG,YAGG,ZAGG,RAGG.
ELSEIF (ICODE .EQ. 8) THEN
  READ (8,300,REC=NUM) X,Y,Z,R
  XAGG(NUM4) = NUM
  YAGG(NUM4) = X
  ZAGG(NUM4) = Y
  RAGG(NUM4) = Z
  NUM4 = NUM4 + 1
ENDIF

INCREMENT THE VARIABLE NUMBER AND READ THE NEXT ONE.
NUM = NUM + 1
GOTO 10

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)

CODE TO RETURN TO HOME POSITION
CODE = 4.0

CALL THE GRAPHICS DISPLAY PROGRAM
CALL DEPUTY1(XVAL,YVAL,ZVAL,RVAL)

PROMPT THE USER FOR SINGLE-STEPPING OR CONTINUOUS OPERATION.
IF THE USER OPTS FOR SINGLE-STEPPING SET SS TO 1.
CALL DSHAP(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

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

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

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

CALL PART

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

DISPLAY NAME OF PROGRAM BEING EXECUTED, AND THE USE OF FUNCTION 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)

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

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

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
FORMAT(*80)
CALL TTOUT(0,12,1,STAT:80,80,31)

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

IF (SS .EQ. 1) THEN

Appendix C.
CALL CSRN
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)
CALL INEY(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

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 DSSMAP(0)
IF (K1 .EQ. 1) THEN

Appendix C. 96
IF ((K2 .EQ. 2) .OR. (K2 .EQ. 3) .OR. (K2 .EQ. 4)) 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)',
857,37,31)
CALL TCSCKEY (ISCAN,IASC,NCHAR)
IF (IASC .EQ. 'Y') THEN
  CODE = 8.0
  CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)
ENDIF
C GO READ THE NEXT STATEMENT TO BE EXECUTED.
125 GOTO 20

123 FORMAT(A1)
100 FORMAT(1X,I4,7X,I1)
200 FORMAT(6X,F8.2)
300 FORMAT(4X,I4,2X,F8.2)
400 FORMAT(4X,I4,2X,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),AGGI(20),PAR(50),PNTVAL(100),VAL5
INTEGER SS,SLPAY,PAR1,PAR2,VAL1,VAL2,RIVAL,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
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 XPTN(100),YPTN(100),ZPTN(100),RPTN(100)
DIMENSION AGGI(20),ZAGGI(20),RAGGI(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

Appendix C. 97
COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
COMMON/F/NL13,NL14,NL5,NL45,NL6,NP1,NP2,NP3,NP4,NP5,NP6,NP7,NP8,NP9,NP10,NP11,NP12,NP13,NP14,NP15
COMMON/G/SS,SLPAY,PAR1,PAR2,PAR3,PAR4,PAR5,PAR6,PAR7,PAR8,PAR9,PAR10,PAR11,PAR12,PAR13,PAR14,PAR15,PAR16,PAR17,PAR18,PAR19,PAR20,PAR21,PAR22,PAR23,PAR24,PAR25,PAR26,PAR27,PAR28
COMMON/H/THMIN,THMAX,THMIN,THMAX,THMIN,THMAX,THMIN,THMAX
COMMON/J/YMIN,YMAX,YMIN,YMAX,YMIN,YMAX,YMIN,YMAX,SYMIN,SYMAX,SYMIN,SYMAX,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 TABLE FILE.
READ (8,200,REC=LZ) 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 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 TABLE 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 ON THE COORDINATES OF THE CORNER POINTS, NUMBER OF PALLET POINTS, 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
ENDIF
XPAL(NUM6,1) = X1
YPAL(NUM6,1) = Y1
K = 1
DO 10 I = 1, PALPTS(NUM6), IPPR
XPAL(NUM6,I) = XPAL(NUM6,1) + (XINC * (K - 1))
YPAL(NUM6,I) = YPAL(NUM6,1) + (YINC * (K - 1))
10 CONTINUE
DO 20 J = 1, IPPR
XPAL(NUM6,I+J-1) = XPAL(NUM6,I) + ((J - 1) * XINC)
YPAL(NUM6,I+J-1) = YPAL(NUM6,I) + ((J - 1) * YINC)
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
RPAU(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(I4, I4))
200 FORMAT(4X, 4(2X, F8.2))
300 FORMAT(22X, F8.2)
RETURN
END
C SUBROUTINE MT
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 MT(K2)
INTEGER COUNT(50), CNTVAL(50), PALLET(10), PALPTS(10), PARTNO(10)
INTEGER CONS(100), ADDG(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, THTMX, ZIN, RIN, CODE

Appendix C.
CHARACTER *10 CELNAM

DIMENSION LASLG(10),IOM(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,PTVAL,AGG,NREC
COMMON/C/LASLG,IONL,LAST,NEXT,ALIN,PAY,ZON,IC
COMMON/D/XPAL,YPAL,ZPAL,RPAL,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,THIN,TIN,RIN,CODE,CELNAME,VAL5
COMMON/J/XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAY
COMMON/P/P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
COMMON/Q/Q16,Q17,Q18,Q19,Q20,Q21,Q22,Q23,Q24,Q25,Q26,Q27,Q28
COMMON/R/R29,R30,R31,R32

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,NUM5-1
      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
C DELAY

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

Appendix C. 101
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 (6,500,REC=L3) Y
  READ (8,500,REC=L4) Z
  READ (8,500,REC=L5) R
  XVAL = XVAL + X
  YVAL = YVAL + YZVAL = 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
VALS = 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 ARE 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 MOVE
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. CONV(I)) THEN
      ZVAL = COMVAL(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,NUMS-1
   IF (L2 .EQ. PAR(I)) THEN
      READ (8,500,REC=L2) ZVAL
      GOTO 61
   ENDIF
69 CONTINUE

Appendix C. 104
C IF THE VALUE IS A NUMBER, ITS VALUE IS READ FROM THE CONSTANTS 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, 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 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  ENDF

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:

K2 - SUBTYPE OF COMMAND
  1 - MAII
  2 - WRITEO

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

SUBROUTINE SENS(K2)

INTEGER COUNT(50),CNTVAL(50),PALLEY(10),PALPTS(10),PARTNO(10)
INTEGER CONS(100),AGG(20),PAR(50),PMTVAL(100),VAL5
INTEGER SS,SLPAY,PART1,PAR2,VAL1,VAL2,RETVAL,BASC

Appendix C.  105
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,THTHI,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,PAL,PNTVAL,AGG,NREC
COMMON/C/LASUB,IONUM,LAST,NEXT,ALIN,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,THTHI,ZIN,RIN,CODE,CELNAM,VALS
COMMON/J/ XMIN,YMIN,YMAX,SYM1,SYM2,SYM3,SYM4,SYM5,SYM6
COMMON/K/ 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, 500, 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, 500, REC=L4) T
L4 = T
GOTO 14

Appendix C. 107
THE VARIABLE NAME IS COMPARED AGAINST THE NAMES OF CONSTANTS, COUNTERS
AND PARAMETERS IN THE PROGRAM, AND ONCE THE MATCH HAS BEEN FOUND, ITS
VALUE IS ASSIGNED TO THE VARIABLE TIME.

IF (L5 .EQ. 1) THEN
   DO 16 I = 1,NUM1-1
      IF (L6 .EQ. CONV(I)) THEN
         TIME = CONV(I)
      GOTO 17
   ENDF
16 CONTINUE
   DO 18 I = 1,NUM3-1
      IF (L6 .EQ. COUNT(I)) THEN
         TIME = COUNT(I)
      GOTO 17
   ENDF
18 CONTINUE
   DO 39 I = 1,NUM5-1
      IF (L6 .EQ. PAR(I)) THEN
         READ (8,500,REC=L6) T
         TIME = T
      GOTO 17
   ENDF
39 CONTINUE

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

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

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

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

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. 6.0 .OR. TIME .GT. 25.5) CALL USER(1)

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

CALL TTOUT(0,16,1,'IS THE PORT ON/OFF (1/O) ?',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 number and the state of the port the corresponding port is given the value 0 or 1
C
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
    ENDIF
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

CALL THE GRAPHICS DISPLAY PROGRAM
CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)

Depending on the value of VAL2 being equal to L4 the next line
or if there is a label to branch to, is decided. Control transfers
to the statement number of the label. Otherwise, an error message is
displayed.

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

WRITEO

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

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

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

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

IF (L1 .EQ. 1) THEN
  DO 20 I = 1,NUM1-1
    IF (L2 .EQ. CON(1)) THEN
      L2 = CONVAL(I)
      GOTO 21
    ENDF
  20 CONTINUE

  DO 22 I = 1,NUM3-1
    IF (L2 .EQ. COUNT(I)) THEN
      L2 = CNTVAL(I)
      GOTO 21
    ENDF
  22 CONTINUE

  DO 49 I = 1,NUM5-1
    IF (L2 .EQ. PAR(I)) THEN
      READ (8,500,REC=L2) T
      L2 = T
      GOTO 21
    ENDF
  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

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

21 IF (L3 .EQ. 1) THEN
  DO 23 I = 1,NUM1-1
    IF (L4 .EQ. CON(1)) THEN
      L4 = CONVAL(I)
      GOTO 25
    ENDF
  23 CONTINUE

DO 24 I = 1,NUM5-1

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

24 CONTINUE

DO 59 I = 1,NUMS-1

IF (L4 .LE. 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 CORRESPONDING 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

Appendix C.
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,I11,2(I2,X,I8))
500 FORMAT(4X,4(2X,F8.2))
900 FORMAT(2X,I11,2X,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
  1 - BRANCH
  2 - TESTC
  3 - TESTI
  4 - TESTP
  5 - BREAKPOINT

C SUBROUTINES CALLED BY THIS PROGRAM ARE:
CHECK,ERROR,READ,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,PAR1,PAR2,VAL1,VAL2,RETV,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,THI,ZIN,RIN,CODE
CHARACTER *10 CELNAH
DIMENSICN LASIB! 10 J GAN ! 32 I a CONVAL! 100 I a L! 5 I a M! 10 I a P! 4 I
DIMENSICN XPAL! 10 a 50 I a YPAL! 10 a 50 I a ZPAL! 10 a 50 I a RPAL! 10 a 50 I
DIMENSICN XPAR! 50 I • YPAR(50 I a ZPAR! 50 I a RPAR! 50)
DIMENSICN XPNT! 100 I a YPNT! 100 I a ZPNT! 100 I a RPNT! 100 I
DIMENSICN XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
DIMENSICN 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,YVAL,YVAL2,ZVAL,RVAL
COMMON/E/XPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
COMMON/F/NUMI,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
COMMON/G/SS,SLPAY,PAR1,PAR2,PAYOLO,VAL1,VAL2,RETV,BASC
COMMON/H/ THONI,THTHI,ZIN,RIN,CODE,CELNAH,VALS
COMMON/J/ XMNX,YMAX,YMIN,XMIN,SXMIN,SYMNX,SYMNX,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

Appendix C.
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. C0NT(I)) THEN
L2 = CNTVAL(I)
GOTO 21
ENDIF
20 CONTINUE

DO 27 I = 1,NUMS-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.

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

Appendix C.
DO 30 I = 1,NUMI-1
   IF (L2 .EQ. CONS(I)) THEN
      L2 = CONVAL(I)
      GOTO 31
   ENDIF
30    CONTINUE

DO 37 I = 1,NUMS-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,NUMI-1
      IF (L4 .EQ. CONS(I)) THEN
         L4 = CONVAL(I)
         GOTO 33
      ENDIF
32    CONTINUE

   DO 34 I = 1,NUMS-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
ENDIF
CONTINUE
C IF THE VALUE IS A NUMBER, ITS VALUE IS READ FROM THE CONSTANTS 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 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
  \text{P6} = 0
ENDIF
ELSEIF (\text{VAL1} \text{ .EQ. 7}) \text{ THEN}
  \text{IF (VAL2 .EQ. 1) THEN}
    \text{P7} = 1
  ELSE
    \text{P7} = 0
  ENDIF
ELSEIF (\text{VAL1} \text{ .EQ. 8}) \text{ THEN}
  \text{IF (VAL2 .EQ. 1) THEN}
    \text{P8} = 1
  ELSE
    \text{P8} = 0
  ENDIF
ELSEIF (\text{VAL1} \text{ .EQ. 9}) \text{ THEN}
  \text{IF (VAL2 .EQ. 1) THEN}
    \text{P9} = 1
  ELSE
    \text{P9} = 0
  ENDIF
ELSEIF (\text{VAL1} \text{ .EQ. 10}) \text{ THEN}
  \text{IF (VAL2 .EQ. 1) THEN}
    \text{P10} = 1
  ELSE
    \text{P10} = 0
  ENDIF
ELSEIF (\text{VAL1} \text{ .EQ. 11}) \text{ THEN}
  \text{IF (VAL2 .EQ. 1) THEN}
    \text{P11} = 1
  ELSE
    \text{P11} = 0
  ENDIF
ELSEIF (\text{VAL1} \text{ .EQ. 12}) \text{ THEN}
  \text{IF (VAL2 .EQ. 1) THEN}
    \text{P12} = 1
  ELSE
    \text{P12} = 0
  ENDIF
ELSEIF (\text{VAL1} \text{ .EQ. 13}) \text{ THEN}
  \text{IF (VAL2 .EQ. 1) THEN}
    \text{P13} = 1
  ELSE
    \text{P13} = 0
  ENDIF
ELSEIF (\text{VAL1} \text{ .EQ. 14}) \text{ THEN}
  \text{IF (VAL2 .EQ. 1) THEN}
    \text{P14} = 1
  ELSE
    \text{P14} = 0
  ENDIF
ELSEIF (\text{VAL1} \text{ .EQ. 15}) \text{ THEN}
  \text{IF (VAL2 .EQ. 1) THEN}
    \text{P15} = 1
  ELSE
    \text{P15} = 0
  ENDIF
ELSEIF (\text{VAL1} \text{ .EQ. 16}) \text{ THEN}
  \text{IF (VAL2 .EQ. 1) THEN}
    \text{P16} = 1
  ELSE
    \text{P16} = 0
  ENDIF
ENDIF

\text{C VAL1 IS THE PORT AND VAL2 IS THE STATE OF THE PORT (L6=LABEL)}

Appendix C.
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(I)

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. PALLE(T(I))) THEN
      L2 = PART(NO(I))
      GOTO 41
    EDFIF
  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
  42 CONTINUE
ENDIF

CONTINUE

DO 44 I = 1, NUMS-1

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

CONTINUE

DO 49 I = 1, NUMS-1

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

CONTINUE

C IF THE VALUE IS A NUMBER, ITS VALUE IS READ FROM THE CONSTANTS FILE, AND ASSIGNED TO L4.
C
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 ENCOUNTERED, A MESSAGE IS SEND TO THE USER
C INFORMING HIM OF THE OCCURRENCE OF A BREAKPOINT AND TO HIT ANY KEY
C TO CONTINUE WITH THE PROGRAM SIMULATION.
C
ELSEIF (K2 .EQ. 5) THEN

CALL USER(12)
    NEXT = NEXT + 1

ENDIF

C SUBROUTINE COUN
C
C THIS SUBROUTINE HANDLES ALL THE COUNTER COMMANDS.
PARAMETER DEFINITION:

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

SUBROUTINES CALLED BY THIS PROGRAM ARE:
USER.

SUBROUTINE COUNT(K2)

INTEGER COUNT(50),CNTVAL(50),PALLE(10),PALPTS(10),PARNO(10)
INTEGER CONS(100),AGGI(20),PAR(50),PNTVAL(100),VALS
INTEGER S1,SLPAY,PART1,PAR2,VAL1,VAL2,RETV,ASC
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,THTHI,ZIN,RIN,CODE
CHARACTER *10 CELNAM

DIMENSION LASUB(10),IONUM(32),CONVAL(100),L(5),I(10),P(4)
DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
DIMENSION XNIT(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/ACOUNT,CNTVAL,PALLE,PALPTS,PARTNO
COMMON/B/CONS,CONVAL,PALLE,AGGI,REDC
COMMON/C/LASUB,IONUM,LAST,INDEX,ALIN,PAY,ZZON,IC
COMMON/D/XPAL,YPAL,ZPAL,RPAL,XYAL,XYAL,ZVAL,VAL
COMMON/E/XNIT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
COMMON/F/NUMI,NUM2,NUM3,NUM4,NUM5,NUM6,NUM7,NUM8,NUM9,NUM10,
COMMON/G/SS,SLPAY,PART1,PAR2,PAYOLD,VAL1,VAL2,RETV,ASC
COMMON/H/THONI,THTHI,ZIN,RIN,CODE,CELNAM,VALS
COMMON/J/XHIN,XHIN,XHIN,YHIN,YHIN,SMAX,SMAX,SMAX,SMAX,SMAX
COMMON/K/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
COMMON/L/ P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
COMMON/M/ P29,P30,P31,P32

COMP

READ THE COMMAND PARAMETER NUMBERS. L1 IS THE VARIABLE NUMBER OF
OF THE COUNTER. L2 IS THE CONDITION IN THE STATEMENT. L3 IS 2
IF THE VALUE IS A NUMBER AND 1 IF THE VALUE IS SPECIFIED AS A
COUNTER OR A CONSTANT. L5 IS THE LABEL NUMBER. THE SYMBOL TABLE
FILE IS READ AT RECORD L5, TO READ THE STATEMENT NUMBER OF THE
LABEL, AND L6 IS ASSIGNED THE STATEMENT NUMBER.
L1 REPRESENTS COUNTER NAME, L2 REPRESENTS THE CONDITION,
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 (6,310,REC=L5) CC
L6 = CC

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

DO 10 I = 1,NUMS-1
   IF (L1 .EQ. COUNT(I)) THEN
      J1 = CNTVAL(I)

Appendix C.
GOTO 11
ENDIF

10 CONTINUE
DO 17 I = 1,NUMS-1
  IF (L1 .EQ. PAR(I)) THEN
    READ (8,600,REC=L1) T
    J1 = T
    GOTO 11
  ENDIF
CONTINUE

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

11 IF (L3 .EQ. 1) THEN
  DO 14 I = 1,NUMS-1
    IF (L4 .EQ. C(N$(I)) THEN
      J2 = CNTVAL(I)
      GOTO 13
    ENDIF
  CONTINUE
DO 12 I = 1,NUMS-1
  IF (L4 .EQ. CNT(I)) THEN
    J2 = CNTVAL(I)
    GOTO 13
  ENDIF

12 IF (L3 .EQ. 1) THEN
  DO 18 I = 1,NUMS-1
    IF (L4 .EQ. PAR(I)) THEN
      READ (8,600,REC=L4) T
      J2 = T
      GOTO 13
    ENDIF
  CONTINUE
DO 18 I = 1,NUMS-1
  IF (L4 .EQ. PAR(I)) THEN
    READ (8,600,REC=L4) T
    J2 = T
    GOTO 13
  ENDIF
ELSEIF (L3 .EQ. 2) THEN
  READ (4,400,REC=L4) T
  J2 = T
  GOTO 13
ENDIF

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

Appendix C.
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,NUMS-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,NUMS-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  ETC
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.

Appendix C. 124
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 HAS 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
      GO TO 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. CONST(I)) THEN
      CNTVAL(L5) = CONVAL(I)
      NEXT = NEXT + 1
      GO TO 990
   ENDIF
41 CONTINUE
DO 42 I = 1,NUM3-1
   IF (L4 .EQ. COUNT(I)) THEN
      CNTVAL(L5) = CNTVAL(I)
      NEXT = NEXT + 1
      GO TO 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
      GO TO 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.
C SLBRGJTINE 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, THTHI, ZIN, RIN, CODE
CHARACTER *10 CELNAM

DIMENSION LASUB(10), IONUM(32), CONVAL(100), L(10), P(4)
DIMENSION XPAL(10,50), YPAL(10,50), ZPAL(10,50), RPAL(10,50)
DIMENSION XPTNT(100), YPTNT(100), ZPTNT(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/CNS, CONVAL, PAR, PNTVAL, AGG, NREC
COMMON/C/LASUB, IONUM, LAST, NEXT, ALIN, PAY, ZON, IC
COMMON/D/XPAL, YPAL, ZPAL, RPAL, KVAL, ZVAL, ZVAL, RVAL
COMMON/E/XPTNT, YPTNT, ZPTNT, RPNT, XAGG, YAGG, ZAGG, RAGG
COMMON/F/NUM1, NUM2, NUM3, NUM4, NUM5, NUM6, XPAR, YPAR, ZPAR, RPAR
COMMON/G/SS, SLPAY, PART1, PART2, PARYLD, VAL1, VAL2, RETVAL, BASC
COMMON/H/ THONI, THTHI, ZIN, RIN, CODE, CELNAM, VAL5
COMMON/J/ XMIN, XMAX, YMIN, YMAX, S0HIN, S0HAX, 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 TO 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.
       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) - 1ELSE
                    PARTNO(I) = PALPTS(I)
                ENDF
                ZPAL(I,PARTNO(I)) = ZVAL
                GOTO 900
            ENDF
        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)
                ENS
            40 CONTINUE
            DO 42 I = NUM3-1
                IF (L4 .EQ. Count(I)) THEN
                    L4 = CNTVAL(I)
                ENS
            42 CONTINUE
            DO 43 I = 1,NUM5-1
                IF (L4 .EQ. Par(I)) THEN
                    READ (8,200,REC=L4) T
                    L4 = T
                ENS
            43 CONTINUE
        ENDIF
    ELSEIF (K2 .EQ. 5) THEN
        C READ THE COMMAND PARAMETERS.
        ...
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 IF 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.
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
   ENDIF
   CONTINUE
44 CONTINUE
ENDIF
99 FORMAT(2X,I1,4(2X,F8.2))
100 FORMAT(17X,7(1X,I4))
200 FORMAT(6X,F8.2)
300 FORMAT(22X9F8.2)
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
C SUBROUTINE SUB1(K2)
INTEGER COUNT(50),CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
INTEGER CONS(100),AGGI(20),PAR(50),PNTVAL(100),VAL5
INTEGER SS,SLPAY,PAR1,PAR2,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
IF THE COMMAND IS "END", READ THE OBJECT FILE TO FIGURE OUT THE NEXT STATEMENT TO BE EXECUTED, AND ASSIGN IT TO THE VARIABLE, NEXT.

IF (K2 .LE. 2) THEN
    READ (7,100,REC=NEXT), L2
    NEXT = L2
ENDIF

99 FORMAT(2X,I1,2X,I6)
100 FORMAT(23X,14)

RETURN
END

SUBROUTINE SUB2

THIS SUBROUTINE HANDLES CALLS TO SUBROUTINES.
THIS HANDLES A MAXIMUM OF 5 FORMAL PARAMETERS.

PARAMETER DEFINITION:

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,RETYVAL,BASC
REAL THON,THTM,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 XPN(100),YPN(100),ZPN(100),RPN(100)
DIMENSION XAG(20),YAG(20),ZAG(20),RAG(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,XDAL,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,RETYVAL,BASC
COMMON/H/THON,THTM,ZIN,RIN,CODE,CELNAM,VAL5
COMMON/J/XMIN,XMAX,YMIN,YMAX,XMIN,YMIN,YMAX,XMIN,YMAX
COMMON/P/P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
COMMON/G/P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
COMMON/R/P29,P30,P31,P32
COMMON/H/ THONI,THTMI,ZIN,RIN,CODE,CELNAM,VALS
COMMON/J/ XM IN,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(I) THROUGH L(S).
READ (7,100,REC=KZ) 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 COMPLETE.
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
  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.

10 IF (M(I) .EQ. 1) THEN
  DO 11 J = 1,NUM1-1
    IF (M(I+1) .EQ. CONS(J)) THEN
      P(1) = CONSVAL(J)
      NN = 1
      GOTO 9
    ENDIF
  CONTINUE
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
CONTINUE

12 CONTINUE

DO 13 J = 1,NUM3-1
  IF (M(I+1) .EQ. COUNT(J)) THEN
    P(1) = CNTVAL(J)
  CONTINUE
ENDIF
NN = 1
GOTO 9
ENDIF

13 CONTINUE

DO 14 J = 1,NUM-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 (8,600,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,13(I2,I4))
120 FORMAT(17X,10(I2,I4))
300 FORMAT(11X,4,7X,F8.2)
400 FORMAT(1X,5S,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,PAR1,PAR2,VAL1,VAL2,RETTY,RETYBAS
REAL THMA,THMB,ZIN,RIN,CODE
CHARACTER *10 CELNAM
DIMENSION LASUB(10),IONUM(10),CONVAL(100),L(5),M(10),P(4)
THE PROGRAM READS THE VARIABLE NUMBER OF THE COMMAND PARAMETER.
L1 AND L2 REPRESENT THE VARIABLE NAME.

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

IF (L1 .EQ. 1) THEN

CHECK IF THE VARIABLE IS A POINT. IF SO, SET THE CURRENT VALUE
OF THE COORDINATES OF THE ROBOT ARM TO THE ARRAYS, XPNT,YPNT,
ZPNT AND RPNT.

DO 10 I = 1,NUM1-1

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
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

INCREMENT THE STATEMENT NUMBER TO BE EXECUTED, AND WRITE THE
CURRENT VALUE OF THE ROBOT ARM COORDINATES INTO THE LOCATION
OF THE VARIABLE IN THE SYMBOL TABLE FILE.

NEXT = NEXT + 1
WRITE (8,200,REC=L2) L2,XVAL,YVAL,ZVAL,RVAL

Appendix C.
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
C PARAMETER DEFINITION:
C
C NNNN - ERROR NUMBER
C
C SUBROUTINES CALLED BY THIS PROGRAM ARE :
C CSROFF,CSRON,FINI,INKEY,TTOUT.

SUBROUTINE USER(NNNN)

INTEGER CNTVAL(50),PALLET(10),PALPTS(10),PARTNO(10)
INTEGER CONS(100),AGG(20),PAR(50),NTVAL(100),VAL5
INTEGER SS,SLPAY,PAR1,PAR2,VAL1,VAL2,RETVAL,BASC
INTEGER KYO1,KYO2
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),CELNAME(10)
REAL THONI,THHNI,ZIN,RIN,CODE

DIMENSION LSUB(10),IONUM(32),CONVAL(100),L(5),P(4)
DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
DIMENSION XNPT(100),YNPT(100),ZNPT(100),RNPT(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,NTVAL,AGG,NREC
COMMON/C/LSUB,IONUM,LAST,NEXT,ALIN,PAY,ZON,IC
COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
COMMON/E/XPNT,YPNT,ZPNT,RNPT,XAGG,YAGG,ZAGG,RAGG
COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
COMMON/G/SS,SLPAY,PAR1,PAR2,PAYOLD,VAL1,VAL2,RETVAL,BASC
COMMON/H/THONI,THHNI,ZIN,RIN,CODE,CELNAME,VAL5
COMMON/J/ XMIN,YMIN,XMAX,YMAX,SMIN,SMAX,SYMIX,SYMAY
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 = ' '/

THE ERROR NUMBERS ARE LESS THAN 20 FOR ERRORS IN THE USER'S
PROGRAM. THE SAME ERRORS DURING USER INPUT FOR INTERACTIVE
SIMULATION HAVE A VALUE OF 20 ADDED TO THEM. IF THE ERROR
IS GREATER THAN 20, ISET IS SET TO 1, AND PROGRAM SIMULATION
DOES NOT TERMINATE. HOWEVER, IF THE ERROR IS IN THE USER'S
PROGRAM, PROGRAM SIMULATION TERMINATES.

CALL CSROFF
ISET = 0

IF (NNNN .GT. 20) THEN
  ISET = 1
  NN = NNNN - 20
ELSE
  NN = NNNN
ENDIF
MESSAGE IS DISPLAYED, AND PROGRAM SIMULATION DOES NOT TERMINATE WHEN
THE COMMAND "BREAKPOINT" IS FOUND IN THE USER'S PROGRAM.

IF (IN .EQ. 12) THEN
CALL TTOUT(0,20,1,'BREAKPOINT found. Hit any key to resume...'

CALL INKEY(KYCOD1,KYCOD2)
CALL TTOUT(0,20,1,BLANK,80,80,31)
RETURN
ENDIF

IF (IN .EQ. 1) THEN
CALL TTOUT(0,20,1,'Time value out of range. Simulation aborted.'

ELSEIF (IN .EQ. 2) THEN
CALL TTOUT(0,20,1,'R value out of range. Simulation aborted.'

ELSEIF (IN .EQ. 3) THEN
CALL TTOUT(0,20,1,'Z value out of range. Simulation aborted.'

ELSEIF (IN .EQ. 4) THEN
CALL TTOUT(0,20,1,'X value out of range. Simulation aborted.'

ELSEIF (IN .EQ. 5) THEN
CALL TTOUT(0,20,1,'Y value out of range. Simulation aborted.'

ELSEIF (IN .EQ. 7) THEN
CALL TTOUT(0,20,1,'DI port out of range. Simulation aborted.'

ELSEIF (IN .EQ. 8) THEN
CALL TTOUT(0,20,1,'DI/DO value out of range. Simulation aborted.'

ELSEIF (IN .EQ. 9) THEN
CALL TTOUT(0,20,1,'Timeout error in WAITI. Simulation aborted.'

ELSEIF (IN .EQ. 10) THEN
CALL TTOUT(0,20,1,'DO port out of range. Simulation aborted.'

ELSEIF (IN .EQ. 11) THEN
CALL TTOUT(0,20,1,'Counter value out of range. Simulation aborted.'

ELSEIF (IN .EQ. 13) THEN
CALL TTOUT(0,20,1,'Palette value out of range. Simulation aborted.'

ENDIF

IF (ISET .EQ. 1) THEN
CALL TTOUT(0,25,1,'Error in value...................

CALL INKEY(KYCOD1,KYCOD2)
CALL TTOUT(0,20,1,BLANK,80,80,31)
CALL TTOUT(0,24,1,BLANK,80,80,31)
CALL CSTRN
RETURN

ELSE
CALL TTOUT(0,24,1,'Hit any key to terminate Simulation...',
141,41,28)
CALL INKEY(KYCOD1,KYCOD2)
CALL CSROF
CALL FIN

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 BLAN((80),C(4),CELNAME10
REAL THON1,THTHI,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,KVAL,YVAL,ZVAL,RVAL
COMMON/E/XPNT,YPNT,ZPNT,RPNT,XAGG,YAGG,ZAGG,RAGG
COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,NUM7,NUM8,NUM9,NUM10,NUM11,
COMMON/G/SS,SLPAY,PART1,PART2,PAYOLD,VAL1,VAL2,RETVAL,BASC
COMMON/H/THON1,THTHI,ZIN,RIN,CODE,CELNAME,VALS
COMMON/I/XIN,XMAX,YIN,YMAX,SQIN,SQMAX,SQYIN,SQMAX
COMMON/P/11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28
COMMON/R/P29,P30,P31,P32

DATA BLANK/80 '*' /

DISPLAY THE EXECUTION OPTIONS MENU AND READ USER'S INPUT.

33 CALL CLS(31)
CALL CSROF
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,'FM1 - Complete program Simulation',35,35,31)
CALL TTOUT(0,10,26,'FM2 - Partial program Simulation',34,34,31)
CALL TTOUT(0,11,26,'FM3 - Subroutine Simulation',29,29,31)
CALL TTOUT(0,12,26,'FM4 - Abort Simulation',20,24,31)
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 (KICODE1 .NE. 0) THEN
  CALL CSROFF
  CALL TTOUT(0,24,1,'KEY NOT DEFINED -- Hit any key to resume',
  * 40,40,28)
  CALL INKEY(KICODE1,KICODE2)
  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 (KICODE2 .EQ. 59) THEN
  RETURN
ELSEIF (KICODE2 .EQ. 60) THEN
  GOTO 37
C IF THE USER STRIKES THE F2 KEY, PROMPT THE USER FOR THE LINE
C NUMBERS AT WHICH SIMULATION SHOULD START AND END.

ELSEIF (KICODE2 .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
ELSEIF (KICODE2 .EQ. 62) THEN
C DISPLAY ERROR MESSAGE IF INVALID KEY IS STRUCK.

  CALL CSROFF
  CALL TTOUT(0,24,1,'KEY NOT DEFINED -- Hit any key to resume',
  * 40,40,28)
  CALL INKEY(KICODE1,KICODE2)
  CALL CSRON
  CALL TTOUT(0,24,1,BLANK,80,80,31)
  GOTO 36
ENDIF

C IF THE USER STRIKES THE F3 KEY, CALL THE SUBROUTINE GOSUB.

C IF THE USER STRIKES THE F4 KEY, CALL THE SUBROUTINE FINI.

C DISPLAY ERROR MESSAGE IF INVALID KEY IS STRUCK.

ELSE

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.

  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
  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)

Appendix C. 138
CALL INKEY(KYCOD1,KYCOD2)
CALL TTOUT(0,24,1,BLANK,80,80,31)
CALL CSROH
GOTO 34
ENDIF

JJJ = JJJ + 1
IF (JJJ .LE. 4) GOTO 34
PART1 = JJJ(1) * 1000 + JJJ(2) * 100 + JJJ(3) * 10 + JJJ(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(0,10,35,'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, JJJ(1), 
C JJJ(2), JJJ(3), AND JJJ(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 = JJJ(1)
GOTO 1001
ELSEIF (KYCOD1 .EQ. 13 .AND. JJJ .EQ. 3) THEN 
PART2 = JJJ(1) * 10 + JJJ(2)
GOTO 1001
ELSEIF (KYCOD1 .EQ. 13 .AND. JJJ .EQ. 4) THEN 
PART2 = JJJ(1) * 100 + JJJ(2) * 10 + JJJ(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

Appendix C.
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')
  CALL INKEY(Kycod1,Kycod2)
  CALL TTOUT(0,24,1,41,41,28)
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(0,24,1,'starting line number',21,21,28)
  CALL TTOUT(0,24,52,'Press any key to continue..',27,27,28)
  CALL INKEY(Kycod1,Kycod2)
GOTO 35
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>1X.I4)
101 FORMAT(6X,I4,6X,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
Appendix C. 140
IF THE STATEMENT NUMBER IS GREATER THAN THE LAST
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

THE PROGRAM SIMULATION STARTS AT THE STATEMENT NUMBER DENOTED BY JJJ.
PART1 = JJJ

THIS SECTION CHECKS FOR THE LAST STATEMENT NUMBER.

JJJ = PART1

IF THE USER DOES NOT SPECIFY ANY NUMBER FOR THE END LINE, PART2 IS
EQUAL TO 1000, AND PROGRAM SIMULATION ENDS AT THE LAST STATEMENT
NUMBER, K2.

IF (PART2 .EQ. 1000) THEN
   PART2 = K1
GOTO 1110
ENDIF

THE OBJECT FILE IS READ FOR THE LINE NUMBERS, AND COMPARED TO THE
USER'S INPUT. IF THE LINE NUMBER IS LESS THAN PART2, THE NEXT LINE
IN THE OBJECT FILE IS READ.

READ (7,101,REC=JJJ) JJJJ

IF (JJJJ .LT. PART2) THEN
   JJJ = JJJ + 1
ENDIF

IF THE LAST STATEMENT NUMBER IS REACHED, AN ERROR MESSAGE IS
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(0,24,38,'Press any key to continue....',29,29,28)
CALL INKEY(KYCOD1,KYCOD2)
GOTO 33
ENDIF

GOTO 1003

ENDIF

IF THE END STATEMENT NUMBER IS THE LAST STATEMENT OF THE PROGRAM,
PART2 IS SET TO THE LAST STATEMENT NUMBER, OTHERWISE, IT IS SET
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 CSRFON
CALL TTOUT(0,7,1,'Please move the robot arm to desired position',
#65,45,31)
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 CSRFON
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 CSRFON
CALL TTOUT(0,24,1,'Key not defined -- Hit any key to resume',
#40,40,28)
CALL INKEY(KYCOD1,KYCOD2)
CALL CSRFON
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.
PARAMETER DEFINITION:

KC0D2 - FUNCTION KEY SELECTED BY USER

SUBROUTINES CALLED BY THIS PROGRAM ARE:
CHECK,CLS,CSROFF,CSRON,INKEY,LOCATE,PARTM,TTOUT,
USER,WRITEP,WRITER.

SUBROUTINE PARTN(KCOD2)

INTEGER COUNT(50),CNVAL(50),PALLEI(10),PALPTS(10),PARTNO(10)
INTEGER CONS(100),AGG(20),PRT(50),PNTVAL(100)
INTEGER SS,SLPAY,PART1,PART2,VAL1,VAL2,RETPAL BASC
INTEGER = 2 KC0D1,KC0D2,00D2,H,LL
INTEGER P12,P5,P4,P3,P6,P1,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,THTNN,Z1N,REI CODE,VAL1,VAL2,VAL3,VAL4
REAL XVAL,YVAL,ZVAL,RVAL
CHARACTER * 1 BLANK(80),CELNAME100
CHARACTER * 72 CHAR,CNT "
DIMENSION LASUB(10),JOHN(32),CUNAL(100),L(5),M(10),P(4)
DIMENSION XPAL(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
DIMENSION XPTS(100),YPTS(100),ZPTS(100),RPNT(100)
DIMENSION XAGG(20),YAGG(20),ZAGG(20),RAGG(20)
DIMENSION XPAR(50),YPAR(50),ZPAR(50),RPAR(50)

COMMON/A/COUNT,CNVAL,PALLEI,PALPTS,PARTNO
COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
COMMON/C/LASUB,IONNUM,52,CONVAL(100),L(5),M(10),P(4)
COMMON/D/XPAL,YPAL,ZPAL,RPAL,XVAL,YVAL,ZVAL,RVAL
COMMON/E/XPTS,YPTS,ZPTS,RPNT,XAGG,YAGG,ZAGG,RAGG
COMMON/F/NUM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,ZPAR,RPAR
COMMON/G/SS,SLPAY,PART1,PART2,VAL0D,VAL1,VAL2,RETPAL,BASC
COMMON/H/THONI,THTNI,ZIN,REI CODE,CELNAME10
COMMON/J/XMIN,XMAX,YMIN,YMAX,SXMIN,SXMAX,SYMIN,SYMAX
COMMON/K/P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15
COMMON/L/P16,P17,P18,P19,P20,P21,P22,P23,P24,P25,P26,P27,P28
COMMON/R/R1,P50,P31,P32

DATA BLANK/80 ** /

VAL1 = 0.0
VAL2 = 0.0
VAL3 = 0.0
VAL4 = 0.0

OPEN A TEMPORARY FILE TO WRITE AND READ THE USER'S INPUT.

OPEN (9,FILE='IRPS01',STATUS='NEW')
CALL CLS(31)
CALL CSRON

SETPART

IF THE USER HAD STRUCK F1 KEY, PROMPT THE USER FOR THE PALLEI
NAME. READ THE INPUT, AND WRITE THE NAME INTO THE FILE IRPS01.
COMPARE THIS NAME TO THE VARIABLE NAMES IN THE VARIABLES FILE.
IF NAME DOES NOT EXIST, DISPLAY ERROR MESSAGE. IF PALLEI NAME
HAS BEEN FOUND, FIND THE INDEX NUMBER FOR THE ARRAY "PALLEI".

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)
REMARK (9)
199 READ (3,500,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,'Pa11•f undafinad ·- Hit any key to resume',
41,41,28)
CALL INKEY(KYCOD1,KYCOD2)
REMARK (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)
REMARK (9)
READ (9,200) VALU1
REMARK (9)
IF (VALU1 .GT. PALPTS(L5) .0R. VALU1 .LT. 1.0) THEN
   PARTNO(L5) = 1
   CALL USER(33)
   REMAIN (3)
   GOTO 1000
else
   PARTNO(L5) = VALU1
   ZPAL(L5,PARTNO(L5)) = ZVAL
ENDIF
REMARK (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.

498 ELSEIF (KCOD2 .EQ. 60) THEN
   CALL TTOUT(0,10,37,'GETPART',7,7,31)
   CALL TTOUT(0,11,37,--------',7,7,31)
   CALL TTOUT(0,14,28,'Pallet name = ',15,15,31)
   CALL PARTM(2,14,40,CHAR,LL,2)
   WRITE (9,100) LL,(CHAR(MM:MM),MM=1,LL)
   REMAIN (9)
499 READ (3,500,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
Appendix C. 144
ENDIF

CONTINUE

CALL CSROFF
CALL TTOUT(0,24,1,'Pallet undefined -- Hit any key to resume',
* 43,41,28)
CALL INKEY(KYCOD1,KYCOD2)
REHIND (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)
    REMIND (3)
    CALL DSNAPI(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 DSNAPI(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(0,10,30,'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(0,16,25,'Point name = ',13,13,31)
    CALL PARTN(3,14,38,CHAR,LL,2)
    WRITE (9,100) LL,(CHAR(MM:MM),MM=1,LL)
    REMIND (9)

599   READ (3,300,END=601) IC2,NIT,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
    GO TO 602
  ENDIF
END CONTINUE

601 CALL CSROFF
  CALL TTOUT(0, 24, 1, 'Point undefined -- Hit any key to resume',
  40, 40, 28)
  CALL INKEY(KYCOD1, KYCOD2)
  REMIND (3)
  GO TO 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)
  GO TO 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, 16, 25, 'X value = ', 10, 10, 31)
  CALL PARTM(3, 14, 35, CHAR, LL, 1)
  WRITE (9, 100) LL, (CHAR(MM:MM), MM=1, LL)
  REMIND (9)
  IF (LL .LE. 1) GO TO 1
  READ (9, 200) VALU1
  REMIND (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)
  REMIND (9)
  IF (LL .LE. 1) GO TO 2
  READ (9, 200) VALU2
  REMIND (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)
  REMIND (9)
  IF (LL .LE. 1) GO TO 3
  READ (9, 200) VAL3
  REMIND (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)
  REMIND (9)
  IF (LL .LE. 1) GO TO 607
  READ (9, 200) VAL4
  REMIND (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)
  REMIND (3)
  GO TO 1000
ENDIF

IF (VAL3 .LT. -250.0 .OR. VAL3 .GT. 0.0) THEN
CALL USER(23)
REIND (3)
GOTO 1000
ENDIF

ENDIF

IF (VALU1 .LT. -650.0 .OR. VALU1 .GT. 650.0) THEN
CALL USER(24)
REIND (3)
GOTO 1000
ENDIF

ENDIF

IF (VALU2 .LT. -387.0 .OR. VALU3 .GT. 650.0) THEN
CALL USER(25)
REIND (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)

REIND (3)
CALL DSNAP(0)

CALL TTOUT(0,11,1,'Do You Want to Redraw the Screen (Y/N)',
#58,58,31)
READ(*,123) ANS
IF (ANS .EQ. 'Y') THEN
CODE = 8.0
CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)
CALL DSNAP(0)
ENDIF

C ZMOVE

C IF THE USER STRUCK THE F4 KEY, PROMPT THE USER 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(4,16,35;CHAR,LL,1)
WRITE (9,100) LL,(CHAR(MM,MM),MM=1,LL)
REIND (9)
READ (9,200) VALU1
REIND (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 DSHAP(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 DSHAP(0)
ENDIF

C WRITEO
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 (KCDO2 .EQ. 63) THEN
CALL TTOUT(0,10,37,'WRITEO',6,6,31)
CALL TTOUT(0,11,37,----------',6,6,31)
CALL TTOUT(0,14,25,'DO number = ',12,12,31)
CALL PARTH(5,14,37,CHAR,LL,1)
WRITE (9,100) LL,(CHAR(MM:MM)),MM=1,LL
REMIND (9)
READ (9,200) VALU1
REMIN (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 PARTH(5,15,37,CHAR,LL,1)
WRITE (9,100) LL,(CHAR(MM:MM)),MM=1,LL
REMIND (9)
READ (9,200) VALU2
REMIN (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
PI7 = 1
ELSE
PI7 = 0
ENDIF
ELSEIF (VAL1 .EQ. 18) THEN

Appendix C.  148
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

Appendix C. 149
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 DSHAP(0)

CALL TTOUT(0,11,1,'Do You Want to Redraw the Screen (Y/N)',
            38,36,31)
READ(*,123) ANS
IF (ANS .EQ. 'Y') THEN
    CODE = 5.0
    CALL DEPUTY1 (XVAL,YVAL,ZVAL,RVAL)
    CALL DSHAP(0)
ENDIF

C SETC

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 (KC002 .EQ. 64) THEN

    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)
    WRITR (9,100) LL,(CHAR(MM:MM),MM=1,LL)
    REMIND (9)

    READ (3,500,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,NUMS - 1

        IF (IC2 .EQ. COUNT(I)) THEN
            L5 = I
            GOTO 302
        ENDIF

Appendix C. 150
303 CONTINUE

301 CALL CSROFF
   CALL TTOUT(0,24,1,'Counter undefined -- Hit any key to resume',
   * 42,42,28)
   CALL INKEY(KYCOD1,KYCOD2)
   REMIND (3)
   GOTO 1000

THE USER IS THEN PROMPTED FOR THE VALUE. IF THE VALUE IS OUT OF
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)
   REMIND (9)
   READ (9,200) VALU1
   REMIND (9)
   IF (VALU1 .LT. -32767.0 .OR. VALU1 .GT. 32767.0) THEN
      CALL USER(31)
      REMIND (3)
      GOTO 1000
   ENDIF

THE COUNTER VALUE IS THEN SET TO THE VALUE KEYED IN BY THE USER.
THIS VALUE IS ALSO WRITTEN TO THE SYMBOL TABLE FILE.

   CNTVAL(L5) = VALU1
   WRITE(8,400,REC=IC2) IC2,VALU1
   REMIND (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 ROM - ROM NUMBER
C COLM - COLUMN NUMBER
C CHAR - CHARACTER TYPED BY USER
C LL - INCREMNTAL COLUMN NUMBER
C MM - INDICATES WHETHER INPUT SHGJLD BE
C NUMERIC (VALUE IS 1) OR ALPHANUMERIC (VALUE IS 2)

SUBROUTINES CALLED BY THIS PROGRAM ARE :
CSROFF,CSROF,INKEY,LOCATE,TTOUT.
SUBROUTINE PARTM(KCODE,ROM,COLM,CHAR,LL,MM)
INTEGER COUNT(50),CNTVAL(50),PALLE(100),PALPTS(100),PARTNO(10)
INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100)
INTEGER S$SLPAY,PAR1,PAR2,VAL$,VAL2,RETAIL,BASC
INTEGER KYC001,KYCOD2,J
INTEGER ROM,COIIM
REAL THONI,THTEI,ZIN,INCODE
CHARACTER BLAN$(80),CELNAME$,AFI$,NEXT$1
CHARACTER CHAR$(1),ALPHA$(26),NUMER$(10)
CHARACTER CHAR$(72)
DIMENSION LASUB$(10),I0MUM$(32),CONVAL$(100),L$(15),H$(10),P$(4)
DIMENSION XPAL$(10,50),YPAL$(10,50),ZPAL$(10,50),RPAL$(10,50)
DIMENSION XNT$(100),YNT$(100),ZNT$(100),RT$(100)
DIMENSION XG$(10),YG$(10),ZG$(10),RG$(10)
DIMENSION XPAR$(50),YPAR$(50),ZPAR$(50),RPAR$(50)
COMMON/A/COUNT,CNTVAL,PALLE,PALPTS,PARTNO
COMMON/B/CONS,CONVAL,PAR,PNTVAL,AGG,NREC
COMMON/C/LASUB,IOMUM,LAST,NEXT,ALIM,PAY,DOM,IC
COMMON/D/XPAL,YPAL,ZPAL,RPAL,ZVAL,ZVAL,RVAl
COMMON/E/XNT,YNNT,ZNT,RT,AGG,YAGG,ZAGG,RAGG
COMMON/F/NM1,NUM2,NUM3,NUM4,NUM5,NUM6,XPAR,YPAR,RPAR,PAR
COMMON/G/S$SLPAY,PAR1,PAR2,PAYOLD,VAL1,VAL2,RETAIL,BASC
COMMON/H/THONI,THTHI,ZIN,INCODE,CELNAME,VAL5
COMMON/J/XMIN,XMAX,YMIN,YMAX,SMIN,SMAX,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
   CALL LOCATE(0,ROM,COIM+LL)
   J = LL
   CALL IMKEY(KYCOD1,KYCOD2)
   IF (KYCOD1 .EQ. 13) THEN
      GOTO 20
   ELSE
      CALL ID(KYCOD1,KYCOD2,CHAR1,J,ALPHA,NUMER)
   ENDIF
C IF USER'S INPUT IS INVALID, DISPLAY ERROR MESSAGE.
   IF (J .EQ. 0) THEN
      CALL CSROFF
      CALL TTOUT(0,24,1,'INVALID CHARACTER - Hit any key to resume'
*        ,41,41,20)
      CALL IMKEY(KYCOD1,KYCOD2)
      CALL TTOUT(0,24,1,BLAN$,80,80,31)
      CALL CSRON
      GOTO 5
   ENDIF
C IF THE USER STRIKES THE BACKSPACE KEY, ERASE THE PREVIOUS
C CHARACTER FROM THE SCREEN, OTHERWISE DISPLAY THE CHARACTER
C THE USER TYPES.
   IF (J .LT. LL) THEN
      LL = J
      CALL TTOUT(0,ROM,COIM+J,BLAN$,1,1,31)

GOTO 5
ELSE
   CALL TTOUT(0,ROM,COLM,LL,CHAR1,1,1,26)
   CALL CONCATENATE THE STRING OF CHARACTERS TYPED IN BY THE USER, AND
   INCREMENT THE CURSOR POSITION.
   IF (J .NE. 1) THEN
      CHAR1:LL) = CHAR1:LL-1/CHAR1
      AFILE = CHAR1:LL-1
      AEXT = AFILE
      CALL CONCAT (AFILE,AEXT)
      CHAR1:LL) = AFILE
   ELSE
      CHAR1: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:MNM) .EQ. '.') GOTO 40
   CONTINUE
   LL = LL + 1
   CHAR1(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 MMM = 1,10
         IF (CHAR(MMM:MNM) .EQ. NUMER(MMM)) GOTO 43
         IF (CHAR(MMM:MNM) .EQ. '-' .AND. MMM .EQ. 1) GOTO 43
         IF (CHAR(MMM:MNM) .EQ. '.') GOTO 43
      CONTINUE
      IER = 1
   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:MNM) .EQ. '.') ISET = ISET + 1
      IF (ISET .GT. 1) IER = 1
   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,ROM,COM+MMM,' ',1,1,31)
    52 CONTINUE
    IER = 0
    CALL CSRGN
    GOTO 1
ENDIF

C IF THE INPUT IS ALPHANUMERIC, CHECK THAT THE FIRST CHARACTER 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
    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,ROM,COM+MMM,' ',1,1,31)
    53 CONTINUE
    IER = 0
    CALL CSRGN
    GOTO 1
ENDIF

C IF ANY CHARACTER IS THE DECIMAL POINT OR THE MINUS SIGN, 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

Appendix C. 154
C ALPHA - ARRAY OF ALPHABETS
C NUMER - ARRAY OF NUMBERS

SUBROUTINE ID(KYCOD1,KYCOD2,CHAR1,J,ALPHA,NUMER)

INTEGER COUNT(50),CNTVAL(50),PALLE(10),PALPTS(10),PARTNO(10)
INTEGER CONS(100),AGG(20),PAR(50),PNTVAL(100)
INTEGER SS,SLPAY,PAR1,PAR2,VAL1,VAL2,RETVAL,BASC
INTEGER * 2 KYCOD1,KYCOD2,J
REAL THONI,THTNI,ZIN,RIN,CODE
CHARACTER * 1 CHAR1,ALPHAI26),NUMERI10),CELNAM*10

DIMENSION LASUBI10),IONUM(32),CONVAL(100),L(5),M(10),P(4)
DIMENSION XPAI(10,50),YPAL(10,50),ZPAL(10,50),RPAL(10,50)
DIMENSION XPNTI100),YPNTI100),ZPNTI100),RPNTI100)
DIMENSION XAGGI20),YAGGI20),ZAGGI20),RAGGI20)
DIMENSION XPARI50),YPARI50),ZPARI50),RPARI50)

COMMON/A/COUNT,CNTVAL,PALLE,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,KVAL,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,PAR1,PAR2,PAYOLD,VAL1,VAL2,RETVAL,BASC
COMMON/H/THONI,THTNI,ZIN,ZIN,CODE,CELNAM,VALS
COMMON/J/XMIN,XMAX,YMIN,YMAX,XMIN,XMAX,YMIN,YMAX

IF (KYCOD1 .GE. 48 .AND. KYCOD1 .LE. 57) THEN
   CHAR1 = NUMER(KYCOD1-47)
ELSEIF (KYCOD1 .GE. 65 .AND. KYCOD1 .LE. 90) THEN
   CHAR1 = ALPHAIKYCOD1-64)
ELSEIF (KYCOD1 .GE. 97 .AND. KYCOD1 .LE. 122) THEN
   CHAR1 = ALPHAIKYCOD1-96)
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

Appendix C.
SUBROUTINES CALLED BY THIS PROGRAM ARE:

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 KYC0D1, KYC0D2
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 THGI, THNI, ZIN, RIN, CODE
CHARACTER = 1 BLANK (80), C, CELNAM * 10
CHARACTER = 72 CHAR, CNT, CNT1 * 50

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 XPARI (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, XPAR, ZPAR, ZPAR
COMMON/G/SS, SLPAY, PART1, PART2, PAYOLD, VAL1, VAL2, RETVAL, BASC
COMMON/H/ THONI, THNI, ZIN, RIN, CODE, CELNAM, VAL5
COMMON/J/ XMIN, XMAX, YMIN, YMAX, SXMIN, SXMAX, SYMIN, SYMAX
COMMON/K/ P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P11, P12, P13, P14, P15
COMMON/L/ 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

IER = 0

CALL TTOUT (0, 3, 51, '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 = 1, LL)
RE气温 (39)

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
THE SUBROUTINE NAME DOES NOT MATCH ANY NAMES IN THE PROGRAM.
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)
REMIND (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 1-1 FORMAL PARAMETERS.

DO 700 I = 1,5
606 READ (3,500,END=701) IC21(I),IC3,MN1(I),ICODE,CNT1(I)
IF (ICODE .EQ. 7) GOTO 700
IC21(I) = 0
GOTO 701
700 CONTINUE
701 REMIND(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,'For a formal parameter :',18,18,31)
        CALL TTOUT(0,7,20,CNT1(I)(1:MN1(I)),MN1(I),MN1(I),28)
        CALL TTOUT(0,9,1,BLANK,80,80,31)
        CALL TTOUT(0,9,1,BLANK,80,80,31)
        CALL TTOUT(0,12,1,BLANK,80,80,31)
        CALL TTOUT(0,15,1,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 LOCATE(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.

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,MN,ICODE,CNT

Appendix C. 157
IF (CNT(1:NN) .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 PARAMETER 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
ENDIF

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
ENDIF

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

REMIND (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)
  REMIND (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 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 PARTH(0,10,32,CHAR,LL,1)
WRITE (39,100) LL,(CHAR(HH:HH),HH=1,LL)
REMIND (39)
READ (39,800) VALUI
REMIND (39)
IVAL1 = VALUI

IF (IVAL1 .GT. 4) THEN
  CALL CSROFF
  CALL TTOUT(0,24,1,'Too many values -- Hit any key ',30,30,28)
ENDIF
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(0,24,31,'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

THE PROGRAM THEN PROMPTS THE USER FOR EACH VALUE FOR EVERY PARAMETER.
THESE VALUES ARE THEN WRITTEN TO THE LOCATION OF THE FORMAL 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(MH:HM),HM=1,LL)
    REMIND (39)
    READ (39,600) VAL(JJJJ)
    REMIND (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,7ZA1)
300 FORMAT(I1X,I5,7X,I4,2X,I5,2X,I16,A72)
400 FORMAT(18X,I4,1X,I4)
500 FORMAT(I1X,I5,6(2X,F8.2))
501 FORMAT(4X,4(2X,F8.2))
600 FORMAT(I1X,I5,6(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 ROW AND COLUMN OF THE DISPLAY.
SUBROUTINES CALLED BY THIS PROGRAM ARE:
  LOCATE, HRCHAT.

SUBROUTINE TTOUT(PAGE, ROM, 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, ROM, 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, THTHI, ZIN, R1N, 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, THTHI, ZIN, R1N, CODE, CELNAM, VAL5
COMMON/J/ XMIN, XMAX, YMIN, YMAX, SXM1N, SXMAX, SYM1N, 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, ROM, COLUMN)

DO 10 J = 1, LENG
  CALL HRCHAT(PAGE, BUF(J), MRATT, 1)
10  CONTINUE

RETURN
END

SUBROUTINE CONCA (AFILE, AEXT)

THIS SUBROUTINE CONCATENATES THE FILENAME TO AN EXTENSION

AFILE  EQUIVALENCED WITH FILE NAME
AFX   3 CHARACTER FILE EXTENSION

AFILE  EXTENSION APPENDED AND UPPERCASE

CHARACTER * 20 AFILE
CHARACTER * 3 AEXT

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

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

CONTINUE

C CAPITALIZE

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

200 CONTINUE

RETURN

END

SUBROUTINE CONCAT (AFILE, AEXT)

C THIS SUBROUTINE CONCATENATES A CHARACTER TO A STRING

C

C AFILE = ORIGINAL STRING
C AFX = 1 CHARACTER

C

CHARACTER *72 AFILE
CHARACTER*1 AEXT

100 DO 130 I = 1,72
IF(AFFILE(I:I) .EQ. ' ') GO TO 140
130 CONTINUE
140 I = I - 1
AFFILE(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 OUTPUT PORTS
C RIN = INITIAL VALUE OF R
C SMAX = UPPER LIMIT OF USERS X COORDINATE IN THE SIDE VIEW
C SMIN = LOWER LIMIT OF USERS X COORDINATE IN THE SIDE VIEW
C SYMAX = UPPER LIMIT OF USERS Y COORDINATE IN THE SIDE VIEW
C SYMIN = LOWER LIMIT OF USERS Y COORDINATE IN THE SIDE VIEW
C THONI = INITIAL ANGLE FOR ARM ONE
C THXMI = 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

Appendix C.
C XGR = X COORDINATE OF GRIPPER CIRCLE
C XMAX = UPPER LIMIT OF USERS X COORDINATE IN THE PLAN VIEW
C XM1 = LOWER LIMIT OF USERS X COORDINATE IN THE PLAN VIEW
C XMT1 = X COORDINATE OF ARM MIDDLE CIRCLE TOP LINK LEFT POINT
C XM1 = X COORDINATE OF ARM MIDDLE CIRCLE BOTTOM LEFT LINK PT.
C XM2 = X COORDINATE OF ARM MIDDLE CIRCLE TOP LINK RIGHT POINT
C XM2 = X COORDINATE OF ARM MIDDLE CIRCLE BOTTOM RIGHT LINK PT.
C XPLBA(10) = X COORDINATE OF ARM BASE CIRCLE
C XPLED(10) = X COORDINATE 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 COORDINATE 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 YGR = Y COORDINATE OF GRIPPER CIRCLE
C YMAX = UPPER LIMIT OF USERS Y COORDINATE IN THE PLAN VIEW
C YMIN = LOWER LIMIT OF USERS Y COORDINATE 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 COORDINATE 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

SUBROUTINES CALLED BY THIS PROGRAM ARE

MOVE, ZOOM, REDRAW, PLANSIDE, DEAD, IMPORT, EXPORT,
FRAMES, STROUT, SWITCH, STANDARD, MKSPACET.

SUBROUTINE DEPUTY(XVAL,VAL1,VAL2,VAL3,VAL4)

INTEGER ISCAN, IASC, NCHAR, KOUNT, BASC, VAL1, VA2
INTEGER VAL0, VAL1, VAL2, 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, TH1, TH2, TH3, TH4, TH5, TH6, TH7, TH8, TH9, TH10, TH11, TH12
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, TH1, TH2, 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 DMAP(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(61), YPLBA(61), XCIRC(61), YCIRC(61)
REAL XPLMD(61), YPLMD(61), XPLED(61), YPLED(61)
REAL THON1, THTH1, 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, INC3, DTR, GRIPX, GRIPY
CHARACTER*7 TYPE, CELNAM*10, ANS*1

COMMON/G/ SS, SLPAY, PART1, PART2, PAYOLD, VAL1, VAL2, RETVAL, BASC
COMMON/H/ THON1, THTH1, ZIN, RIN, CODE, CELNAM, VALS
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

TO COMPUTE THE THETAS THE TWO ARMS MUST MOVE TO REACH XYZ POSITION

IF (X .EQ. -650.0 .AND. Y .EQ. 0.0) THEN
  THONF = 180.0
  TTHTF = 0.0
  GOTO 100
ENDIF

RNUM = X**2 + Y**2 - L1**2 - L2**2
RDEN = 2.0*L1*L2
RDUM1 = RNUM/RDEN
TTHTF = ACOS(RDUM1)

RNUM = Y*(L1 + L2*COS(TTHTF)) - X*L2*SIN(TTHTF)
RDEN = X*(L1 + L2*COS(TTHTF)) + Y*L2*SIN(TTHTF)

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. 0.0 .AND. Y .GE. 0.0) THEN
IF (X .EQ. 0.0) THEN
  THINT = PI/2.
ELSE
  THINT = ATAN(Y/X)
ENDIF

C IF FINAL DESTINATION IS IN QUADRANT TWO

Appendix C.
IF (X .LT. 0. .AND. Y .GE. 0.) THEN
IF (THONF .LT. 0) THEN
  THINF = THINF + 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
  THGIF = THONF - PI
  THINT = ATAN(Y/X) + PI
ENDIF

C COMMON TO ALL THREE QUADRANTS
THINT = THINT/PI*180.
THONF = THONF/PI*180.
THTHF = THTHF/PI*180.

THDIF = THINT - THONF
THTHP = -THTMF
THONP = THINT + THDIF
C CHECK WHETHER ARMS MEET PHYSICAL CONSTRAINTS
IF (THTHF .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 (THTHF .LT. 20.0) THEN
  IF (THONF .GE. 0.0 .AND. THINF .LE. 200.0) THEN
    IF (THONP .GE. 0.0 .OR. THONP .GT. 200.0) THEN
      THINF = THTHF
      THONF = THINF
      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
  ELSE
    CALL STROUT(5,110,'ARM DOES NOT MEET PHYSICAL CONSTRAINTS ',38)
    CALL STROUT(5,110,'CHECK TOOL TIP CORDINATES ',38)
    RETURN
  ENDIF
ELSE
  IF (THONF .GE. 0.0 .AND. THONF .LE. 200.0) THEN
    THTHF = THTHF
    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)
DTHTHF = THTHF - THTMF
DTTHONF = THONF - THONP

Appendix C
DTHF = ABS(DTHTMF) + ABS(DTHONF)

C COMPUTE THE TOTAL ANGLE MOVED FOR CASE2 (PRIME ANGLES)

DTHTMP = THTMP - THTMI
DTHONP = THONP - THONI
DTHP = ABS(DTHTMP) + ABS(DTHONP)

C COMPARE AND SELECT THE LEAST VALUE

DUMMY = MIN (DTHF, DTHP)

IF (DUMMY .NE. DTHF) THEN
    THTMF = THTHP
    THONF = THONP
ENDIF

100 THINT = THINT*PI/180.
    THONF = THONF*PI/180.
    THTMF = THTMF*PI/180.

C FIND THE ANGLE TO BE MOVED FOR EACH ARM

DT1 = THWF - THONI
DT2 = THTMF - THTMI

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 DRAM 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 (BASE .EQ. 80) THEN
    CALL TMINDO(0,750,0,750)
    CALL DMINDO(XMIN,XMAX,YMIN,YMAX)
    CALL TCSIND(6)
    CALL STROUT(610,750,'PLAN VIEW',9)
ELSE
    CALL TMINDO(751,1023,400,672)
    CALL DMINDO(XMIN,XMAX,YMIN,YMAX)
    CALL TCSIND(6)
ENDIF

Appendix C. 166
CALL STROUT(890,645,'PLAN VIEW',9)

ENDIF

TH1 = TH0 + J*INC1
TH2 = TH0 + 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 DRAM PLAN ROUTINE

C DRAM THE MIDDLE CIRCLE

CALL MOVEA(XPLMD(I),YPLMD(I))
DO 720 I=1,11
    CALL DRAMA(XPLMD(I),YPLMD(I))
720 CONTINUE

C DRAM THE LAST CIRCLE

CALL MOVEA(XPLED(I),YPLED(I))
DO 730 I=1,11
    CALL DRAMA(XPLED(I),YPLED(I))
730 CONTINUE

C LINK 1

CALL MOVEA(XBT,YBT)
CALL DRAMA(XMT1,YMT1)
CALL MOVEA(XBB,YBB)
CALL DRAMA(XMB1,YMB1)

C LINK 2

CALL MOVEA(XMT2,YMT2)
CALL DRAMA(XET,YET)
CALL MOVEA(XMB2,YMB2)
CALL DRAMA(XEB,YEB)

C NOW DRAM THE SIDE VIEW

C IF (IBASC .EQ. 80) THEN
    CALL TMINDO(751,1023,400,672)
    CALL DMINDO(ISXMIN,ISXMAX,ISYMIN,ISYMAX)
ELSE
    CALL TMINDO(0,750,0,750)
    CALL DMINDO(ISXMIN,ISXMAX,ISYMIN,ISYMAX)
ENDIF

C START DRAMING, THE CIRCLES FIRST (IN SIDE VIEW, THEY ARE RECT.)

C BASE CIRCLE

C TOP LINE

CALL MOVEA(YPLBA(I),500.)
CALL DRAMA(YPLBA(I),500.)
CALL DRAMA(YPLBA(S),500.)

C BOTTOM LINE

CALL MOVEA(YPLBA(I),350.)
CALL DRAMA(YPLBA(I),350.)
CALL DRAMA(YPLBA(S),350.)

C LEFT LINE

CALL MOVEA(YPLBA(I),350.)
CALL DRAMA(YPLBA(I),350.)
CALL DRAMA(YPLBA(S),350.)

C RIGHT LINE

Appendix C.
CALL MOVEA(YPLBA(5), 350.)
CALL DRAMA(YPLBA(5), 350.)
CALL DRAMA(YPLBA(5), 500.)

C DRAM THE MIDDLE CIRCLE
C TOP LINE
CALL MOVEA(YPLMD(1), 500.)
CALL DRAMA(YPLMD(1), 500.)
CALL DRAMA(YPLMD(5), 500.)

C BOTTOM LINE
CALL MOVEA(YPLMD(1), 350.)
CALL DRAMA(YPLMD(1), 350.)
CALL DRAMA(YPLMD(5), 350.)

C LEFT LINE
CALL MOVEA(YPLMD(1), 350.)
CALL DRAMA(YPLMD(1), 350.)
CALL DRAMA(YPLMD(5), 350.)

C RIGHT LINE
CALL MOVEA(YPLMD(5), 350.)
CALL DRAMA(YPLMD(5), 350.)
CALL DRAMA(YPLMD(5), 500.)

C DRAM THE LAST CIRCLE
C TOP LINE
CALL MOVEA(YPLED(1), 500.)
CALL DRAMA(YPLED(1), 500.)
CALL DRAMA(YPLED(5), 500.)

C BOTTOM LINE
CALL MOVEA(YPLED(1), 350.)
CALL DRAMA(YPLED(1), 350.)
CALL DRAMA(YPLED(5), 350.)

C LEFT LINE
CALL MOVEA(YPLED(1), 350.)
CALL DRAMA(YPLED(1), 350.)
CALL DRAMA(YPLED(5), 350.)

C RIGHT LINE
CALL MOVEA(YPLED(5), 350.)
CALL DRAMA(YPLED(5), 350.)
CALL DRAMA(YPLED(5), 500.)

C NUM DRAM THE LINKS
C LINK 1
CALL MOVEA(YPLBA(1), 500.)
CALL DRAMA(YPLBA(5), 500.)
CALL MOVEA(YPLBA(1), 350.)
CALL DRAMA(YPLMD(5), 350.)
C LINK 2
CALL MOVEA(YPLMD(1),500.)
CALL DRAMA(YPLED(5),500.)
CALL MOVEA(YPLMD(1),350.)
CALL DRAMA(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 DRAMA(DUMMY,350.)
DUMMY = YARM(3) - 25.
CALL DRAMA(DUMMY,350.)
DUMMY = YARM(3) + 25.
CALL DRAMA(DUMMY,500.)
DUMMY = YARM(3) - 25.
CALL DRAMA(DUMMY,500.)

C DRAM THE GRIPPER IN THE LAST POSITION FOR ALL THE REDRAMS

DUMMY = YARM(3) + 10.
CALL MOVEA(DUMMY,350.)
CALL DRAMA(DUMMY,ZARM(4))
DUMMY = YARM(3) - 10.
CALL DRAMA(DUMMY,ZARM(4))
CALL DRAMA(DUMMY,350.)

1100 CONTINUE

C AFTER ALL THIS SIMULATE THE Z AXIS-GRIPPER MOVEMENT.

DTZ = Z - ZIN
INCR = DTZ/5.0
DO 1250 J = 1, INCREM
ZARM(4) = ZIN + J*INCR
DUMMY = YARM(3) + 10.
CALL MOVEA(DUMMY,350.)
CALL DRAMA(DUMMY,ZARM(4))
DUMMY = YARM(3) - 10.
CALL DRAMA(DUMMY,ZARM(4))
CALL DRAMA(DUMMY,350.)

1250 CONTINUE

C GRIPPER WINDOM

DTR = R - RIN
INCR = DTR/3.0
CALL TMIND0(751,859,672,780)
CALL DMINDO(-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 DRAMA(GRIPX,GRIPY)

Appendix C.
1300 CONTINUE

C REINITIALIZE INITIAL ANGLE VALUES TO FINAL VALUES

CALL TCSBEL
THONI = THONF
THINIT = THNMF
ZIN = Z
RIN = R

1200 RETURN
END

C FOR ZOOM

SUBROUTINE ZOOM

REAL XMIN, XMAX, YMIN, YMAX, XMIN, XMID, YMIN, YMAX
REAL XPLBA(11), YPLBA(11), XCIRC(11), YCIRC(11)
REAL XPLMD(11), YPLMD(11), XPLED(11), YPLED(11)
REAL THONI, THINIT, TH1, TH2, X, Y, Z, R, L1, L2, ZIN, RIN
REAL XBT, YBT, XBB, YBB, XMT1, YMT1, XMB1, YMB1
REAL XMTZ, YMTZ, XBTB, YBB, XMT1, YMT1, XMB1, YMB1
REAL XARM(4), YARM(4), ZARM(4), THETA, PI
REAL GRIPX, GRIPY, CODE, ASPECT, DUMMY, XAVG
REAL X1, X2, Y1, Y2, X, Y, DELX, DELY, YAVG
INTEGER BASC, VAL1, VAL2, RETVAL
LOGICAL*7 TYPE, CELNA, VAL5

COMMON/G/ SS, SPLAY, PART1, PART2, PAYOLD, VAL1, VAL2, RETVAL, BASC
COMMON/H/ THONI, THINIT, ZIN, RIN, CODE, CELNA, VAL5
COMMON/J/ XMIN, XMAX, YMIN, YMAX, XMIN, XMID, YMIN, YMAX
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/ XMTZ, YMTZ, XBTB, YBB, XMT1, YMT1, XMB1, YMB1
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 .LE. 80) THEN
CALL THINDO(0, 750, 0, 750)
CALL DHINDO(XMIN, XMAX, YMIN, YMAX)
CALL STROUT *
(115, 65, 'SELECT NEW CORNER CONNER PRESS SPACE BAR ')
CALL VCUSR (ICHAR, X1, Y1)
CALL STROUT *
(115, 40, 'SELECT OPPOSITE CORNER PRESS SPACE BAR ')
CALL VCUSR (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) DELX = DELX * 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) 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)
CLOSE (22)

CALL REDRAM
ELSE
CALL TMINDO (0, 750, 0, 750)
CALL DMINDO (SXMIN, SXMAX, SYMIN, SYMAX)
CALL STROUT
C (115, 65, 'SELECT NEW MINDOM CORNER PRESS SPACE BAR ', 42)
CALL VCURSR (ICHAR, X1, Y1)
CALL STROUT
C (115, 40, 'SELECTOPPOSITE 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,XMIN,XMAX,YMIN,YMAX,XMIN,YMAX
CLOSE(22)

CALL REDRAM
ENDIF

C TO PAN THE DRAWING

IF (BASC .EQ. 80) THEN
CALL TMINDO(0,750,0,750)
CALL DMINDO(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,XMIN,XMAX,YMIN,YMAX,XMIN,YMAX
CLOSE(22)

CALL REDRAM
ELSE

CALL TMINDO(0,750,0,750)
CALL DMINDO(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 300
ELSE
GOTO 200
ENDIF
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

CALL STROUT
* (315,60,'ENTER NEW (X,Y) FOR CENTER OF MODEL --> ',40)
CALL TSEND
READ (*,*,END =22) XC, YC

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, XHIN, YHIN, XHAX, YHAX, SWIN, SWIN, XWIN, YWIN
CLOSE(22)

CALL REDRAW
ENDIF

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, YM1N, YMAX, SXMIN, SXMAX, SYMIN, SYMAX
REAL XLPBA(11), YLPBA(11), XCIRC(11), YCIRC(11)
REAL XLPMD(11), YLPMD(11), XPLED(11), YPLED(11)
REAL THON, THN, THI, 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, CODE
REAL XI, X2, Y1, YC, XC, DELX, DELY, DUMMY, XAVG, YAVG
INTEGER BASC, VAL1, VAL2, RETVAL
CHARACTER*7 TYPE, CELNAME*10
CHARACTER*1 ANS

COMMON/G/ SS, SLPAY, PART1, PART2, PAYOLD, VAL1, VAL2, RETVAL, BASC
COMMON/H/ THON, THN, THI, ZIN, RIN, CODE, CELNAME, VAL5
COMMON/J/ XMIN, XMAX, YM1N, YMAX, SXMIN, SXMAX, SYMIN, SYMAX
COMMON/K/ XLPBA, YLPBA, XCIRC, YCIRC, XLPMD, YLPMD
COMMON/L/ XPLED, YPLED, THI, TH2, X, Y, Z, R, L1, L2
COMMON/M/ XBT, YBT, XBB, YBB, XMT1, YMT1, XMB1, YMB1
COMMON/O/ XMT2, YMT2, XMB2, YMB2, XET, YET, XEB, YEB
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='NEN')
WRITE (22, 250) IBASC, XMIN, XMAX, YMIN, YMAX, SXMIN, SXMAX, SYMIN, SYMAX
CLOSE (22)
CALL REDRAN
RETURN
END

SUBROUTINE STANDARD IS TO DRAW THE STANDARD VIEW WITHOUT ZOOM. IT WILL DISPLAY THE WORKCELL ALSO.

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, THTHI, TH2, X, Y, Z, R, L1, L2, ZIN, RIN
REAL XBT, YBT, XBB, YBB, XMTI, YMTI, XMBI, YMBI
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, DELX, DELY
INTEGER BASC, VAL1, VAL2, RETVAL
CHARACTER*7 TYPE, CELNAM*10, ANS*1

COMMON/G/ SS, SLPAY, PART1, PART2, PAYO1D, VAL1, VAL2, RETVAL, BASC
COMMON/H/ THONI, THTHI, ZIN, RIN, CODE, CELNAM, VALS
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/H/ XBT, YBT, XBB, YBB, XMTI, YMTI, XMBI, YMBI
COMMON/H/ 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

Appendix C.
YARM(4) = 0.0
ZARM(4) = 250.0
GRIpx = 175.0
GRIpy = 0.0
THONI = 0.0
THTNI = 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)
CLOSE (22)

CALL REDRAN
RETURN
END

C TO REDRAN THE WHOLE PICTURE - STANDARD VIEW AND WITH WORKCELL
SUBROUTINE REDRAN

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, THTNI, 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), XMORK(675), YMORK(675)
REAL CX(2), CY(2), CZ(2), NX(2), NY(2), NZ(2)
INTEGER J(6), VAL1, VAL2, RETVAL, ICON, ICOMP
INTEGER ISCANN, IASC, ICHAR, BASC, KOUNT, Q
CHARACTER*7 TYPE, CELNAM*10, ANS*1
CHARACTER INFILE*1, NAME*20, AFILE*20, DNAME*5
CHARACTER CHOME*1, DUMMY*1, NTTRY*3, AEXT*3
DIMENSION INFILE(80)

COMMON/G/ SS, SLPAY, PART1, PART2, PAYOLD, VAL1, VAL2, RETVAL, BASC
COMMON/H/ THONI, THTNI, 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)
CLOSE (22)

110 CALL DEAD

OPEN (17, FILE=CELNAM, STATUS='OLD', ERR=100)
5 CONTINUE
READ (17, 10, END=200) (INFILE(I), I=1, 80)

Appendix C.
10 FORMAT(85A1)
     READ (INFIL(1:3),55) NTY

55 FORMAT(A5)
     IF (NTY .EQ. '110') THEN
         ICOM = 0
         DO 26 Q = 5,72
             IF (INFIL(Q:Q) .EQ. ',') THEN
                 ICOM = ICOM + 1
                 J(ICOM) = Q
             ENDIF
         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 TMINDO(0,750,0,750)
         CALL DMINDO(XMIN,XMAX,YMIN,YMAX)
     ELSE
         CALL TMINDO(751,1023,400,672)
         CALL DMINDO(XMIN,XMAX,YMIN,YMAX)
     ENDIF

     CALL MOVEA(CX(1),CY(1))
     CALL DRAMA(CX(2),CY(2))

C DRAW THE SIDEVIEW OF EQP. - EITHER AS THE MAJOR OR MINOR VIEW

     IF (BASC .EQ. 80) THEN
         CALL TMINDO(751,1023,400,672)
         CALL DMINDO(SMIN,SMAX,SYMIN,SYMXY)
     ELSE
         CALL TMINDO(0,750,0,750)
         CALL DMINDO(SMIN,SMAX,SYMIN,SYMXY)
     ENDIF

     CALL MOVEA(CY(1),CZ(1))
     CALL DRAMA(CY(2),CZ(2))
     GOTO 5

     ELSEIF (NTY .EQ. '116') THEN
         ICOMP = 0

Appendix C.
DO 30 Q = 5,72
   IF (INFILE(Q:Q) .EQ. ',') THEN
      ICOMP = ICOMP + 1
      J(ICOMP) = Q
   ENDIF
   IF (INFILE(Q:Q) .EQ. ';') THEN
      ICOMP = ICOMP + 1
      J(ICOMP) = Q
   GOTO 40
   ENDIF
30 CONTINUE

40 K = J(1) - 1
    READ(INFILE(5:K),*)D

   K = J(1) + 1
   L = J(2) - 1
   READ(INFILE(K:L),*)E

   K = J(2) + 1
   L = J(3) - 1
   READ(INFILE(K:L),*)F

C DRAW THE PLAN OF EQP. — EITHER AS THE MAJOR OR MINOR VIEW

   IF (BASC .EQ. 80) THEN
      CALL TMINDO(0,750,0,750)
      CALL DMINDO(XMIN,MAX,YMIN,YMAX)
   ELSE
      CALL TMINDO(751,1023,400,672)
      CALL DMINDO(XMIN,MAX,YMIN,YMAX)
   ENDIF
   CALL MOVEA(D,E)
   CALL DRAMA(D,E)

C DRAW THE SIDEVIEW OF EQP. — EITHER AS THE MAJOR OR MINOR VIEW

   IF (BASC .EQ. 80) THEN
      CALL TMINDO(751,1023,400,672)
      CALL DMINDO(SXMIN,MAX,SYMIN,SYMAX)
   ELSE
      CALL TMINDO(0,750,0,750)
      CALL DMINDO(SXMIN,MAX,SYMIN,SYMAX)
   ENDIF
   CALL MOVEA(E,F)
   CALL DRAMA(E,F)

   GOTO 5
   ELSE
   GOTO 5
   ENDIF

200 CLOSE(17)
   CLOSE(19,STATUS='DELETE')

100 CONTINUE
   CALL PLANSIDE
   CALL FRAMES
   CALL PORTS
   RETURN

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(ll), YPLBA(ll), XCIRC(ll), YCIRC(ll)
REAL XPLMD(ll), YPLMD(ll), XPLED(ll), YPLED(ll)
REAL THONI, THTHI, TH1, TH2, X, Y, Z, R, L1, L2, ZIN, RIN
REAL XBT, YBT, XB2, YB2, XMT1, YMT1, XM2, YM2, XET, YET, XEB, YEB
REAL XARM(4), YARM(4), ZARM(4), THETA, PI
REAL XHORK(674), YHORK(674), GRIPX, GRIPY
CHARACTER*10 TYPE, CELNAM*10, ANS*1

COMMON/G/ SS, SLPAY, PART1, PART2, PAYOLD, VAL1, VAL2, RETVAL, BASC
COMMON/H/ THONI, THTHI, 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, XB2, YB2, XMT1, YMT1, XM2, YM2, 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 THINDOI(0.750, 0.750)
   CALL DHINDO(XMIN, XMAX, YMIN, YMAX)
   CALL TCSIND(6)
   CALL STROUT(610, 750, 'PLAN VIEW', 9)
ELSE
   CALL THINDO(751, 1023, 400, 672)
   CALL DHINDO(XMIN, XMAX, YMIN, YMAX)
   CALL TCSIND(6)
   CALL STROUT(690, 645, 'PLAN VIEW', 9)
ENDIF

C DRAW THE WORKSPACE

CALL WSPACE

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(I)

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

XHT1 = XARM(2) - 45. * SIN(TH1)
YHT1 = YARM(2) + 45. * COS(TH1)
XHB1 = XARM(2) + 45. * SIN(TH1)
YHB1 = YARM(2) - 45. * COS(TH1)

C MIDDLE CIRCLE - TOP AND BOTTOM - RIGHT

XHT2 = XARM(2) - 45. * SIN(TH2)
YHT2 = YARM(2) + 45. * COS(TH2)
XHB2 = XARM(2) + 45. * SIN(TH2)
YHB2 = 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 START DRAWING THE SQUARE BASE

CALL MOVEA(55.,55.)
CALL DRAMA(-55.,55.)
CALL DRAMA(55.,-55.)
CALL DRAMA(55.,55.)
CALL DRAMA(0.,0.)
CALL DRAMA(0.,0.)

C DRAW THE BASE CIRCLE

CALL MOVEA(XPLBA(1),YPLBA(1))
DO 710 I = 1,11
CALL DRAMA(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 DRAMA(XPLMD(I),YPLMD(I))
720 CONTINUE

C DRAW THE END CIRCLE

CALL MOVEA(XPLED(1),YPLED(1))
DO 730 I=1,11
    CALL DRAHA(XPLED(I),YPLED(I))
  730 CONTINUE

C LINK 1
    CALL MOVEA(XBT,YBT)
    CALL DRAHA(XMT1,YMT1)
    CALL MOVEA(XBB,YBB)
    CALL DRAHA(XMB1,YMB1)

C LINK 2
    CALL MOVEA(XMT2,YMT2)
    CALL DRAHA(XET,YET)
    CALL MOVEA(XMB2,YMB2)
    CALL DRAHA(XEB,YEB)

C NOW DRAW THE SIDE VIEW
    IF (BASC .EQ. 80) THEN
        CALL THINDO(751,1023,400,672)
        CALL DMINDO(SXMIN,SXMAX,SYMIN,SYMAX)
        CALL TCSIND(13)
        CALL STROUT(690,645,'SIDE VIEW',9)
    ELSE
        CALL THINDO(0,750,0,750)
        CALL DMINDO(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 DRAHA(484,0.)
    CALL TCSIND(4)
    CALL MOVEA(-386.81,0.)
    CALL DRAHA(-386.81,250.)
    CALL DRAHA(650,250.)
    CALL DRAHA(650,0.)
    CALL DRAHA(-386.81,0.)

C ROBOT STAND
    CALL TCSIND(7)
    CALL MOVEA(-50,0.)
    CALL DRAHA(-50,20.)
    CALL DRAHA(50,20.)
    CALL DRAHA(50,0.)
    CALL MOVEA(-30,20.)
    CALL DRAHA(-30,500.)
    CALL DRAHA(30,500.)
    CALL DRAHA(30,20.)

C START DRAWING, THE CIRCLES FIRST (IN SIDE VIEW, THEY ARE RECT.)
    CALL TCSIND(14)

C BASE CIRCLE
CALL MOVEA(YPLBA(1), 500.)
CALL DRAWA(YPLBA(5), 500.)

CALL MOVEA(YPLBA(1), 350.)
CALL DRAWA(YPLBA(1), 350.)
CALL DRAWA(YPLBA(5), 350.)

CALL MOVEA(YPLBA(1), 350.)
CALL DRAWA(YPLBA(1), 350.)
CALL DRAWA(YPLBA(1), 500.)

CALL MOVEA(YPLBA(5), 350.)
CALL DRAWA(YPLBA(5), 350.)
CALL DRAWA(YPLBA(5), 500.)

CALL MOVEA(YPLMD(1), 500.)
CALL DRAWA(YPLMD(1), 500.)
CALL DRAWA(YPLMD(5), 500.)

CALL MOVEA(YPLMD(1), 350.)
CALL DRAWA(YPLMD(1), 350.)
CALL DRAWA(YPLMD(1), 500.)

CALL MOVEA(YPLMD(5), 350.)
CALL DRAWA(YPLMD(5), 350.)
CALL DRAWA(YPLMD(5), 500.)

CALL MOVEA(YPLED(1), 500.)
CALL DRAWA(YPLED(1), 500.)
CALL DRAWA(YPLED(5), 500.)

CALL MOVEA(YPLED(1), 350.)
CALL DRAWA(YPLED(1), 350.)
CALL DRAWA(YPLED(5), 350.)

CALL MOVEA(YPLED(5), 350.)
CALL DRAWA(YPLED(5), 500.)
C  RIGHT  LINE
    CALL  MOVEA(YPLED(5),550.)
    CALL  DRAHA(YPLED(5),550.)
    CALL  DRAHA(YPLED(5),500.)
    
    C  NOW  DRAW  THE  LINKS
    C  LINK  1
    CALL  MOVEA(YPLBA(1),500.)
    CALL  DRAHA(YPLMD(5),500.)
    CALL  MOVEA(YPLBA(1),350.)
    CALL  DRAHA(YPLMD(5),350.)
    
    C  LINK  2
    CALL  MOVEA(YPLMD(1),500.)
    CALL  DRAHA(YPLED(5),500.)
    CALL  MOVEA(YPLMD(1),350.)
    CALL  DRAHA(YPLED(5),350.)
    
    C  THE  GRIPPER  HEAD  AND  THE  MOVING  GRIPPER  PART
    C  THE  GRIPPER  HEAD
    DUMMY  =  YARM(3)  +  25.
    CALL  MOVEA(DUMMY,575.)
    CALL  DRAHA(DUMMY,350.)
    DUMMY  =  YARM(3)  -  25.
    CALL  DRAHA(DUMMY,350.)
    CALL  DRAHA(DUMMY,500.)
    CALL  DRAHA(DUMMY,375.)
    
    C  DRAW  THE  GRIPPER  IN  THE  LAST  POSITION  FOR  ALL  THE  REDRMS
    DUMMY  =  YARM(3)  +  10.
    CALL  MOVEA(DUMMY,350.)
    CALL  DRAHA(DUMMY,ZARM(4))
    DUMMY  =  YARM(3)  -  10.
    CALL  DRAHA(DUMMY,ZARM(4))
    CALL  DRAHA(DUMMY,350.)
    
    RETURN
    END
    
    C  DRAW  THE  GRIPPER
    SUBROUTINE  GRIPPER
    
    C  DRAW  THE  DEAD  WINDOW
    INTEGER  ISCAN,ISAC,NCHAR,BASE,KOUNT,VAL1,VAL2,RETV
    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,XMIN2,XMAX2,YMIN2,YMAX2
    REAL  XPLBA(11),YPLBA(11),XERC(11),YERC(11)
    REAL  XPLMD(11),YPLMD(11),XPLED(11),YPLED(11)
    REAL  THONI,THTWX,TH1,TH2,X,Y,Z,R,L1,L2,ZIN,RIN

Appendix  C.  183
REAL XBT, YBT, XBB, YBB, XMT1, YMT1, XMB1, YMB1
REAL XMT2, YMT2, XMB2, YMB2, XET, YET, XEB, YEB
REAL XARM(4), YARM(4), ZARM, THETA, PI
REAL XGRP(11), YGRP(11), GRIPX, GRIPY
REAL XARM(4), YARM(4), ZARM, THETA, PI, GRIPX, GRIPY
REAL XARM(4), YARM(4), ZARM, THETA, PI, GRIPX, GRIPY
CHARACTER*7 TYPE, CELNAM*10, ANS*1
CHARACTER*7 TYPE, CELNAM*10, ANS*1
COMMON/G/ SS, SLPAY, PART1, PART2, PAYOLD, VAL1, VAL2, RETVAL, BASC
COMMON/H/ THONI, THINT, ZIN, RIN, CODE, CELNAM, VAL5
COMMON/J/ XMIN, XM0, YMIN, YMAX, XM1, XM2, YM1, YM2, 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 THINDO(751, 859, 672, 780)
CALL DMINO(-400., 400., -400., 400.)
CALL TCSINDO(3)
CALL STROUT(920, 750, 'GRIPPER', 7)
CALL STROUT(920, 750, 'OPEN', 5)
CALL STROUT(920, 750, 'CLOSE', 5)
ELSEIF (P18 .LE. 0) THEN
CALL STROUT(920, 750, ' ', 5)
CALL STROUT(920, 750, ' ', 5)
ENDIF

C CIRCUMFERENCE OF GRIPPER

DO 100 I = 1, 11
   THETA = (I-1)*PI/5.
   XGRPI(I) = 175.*COS(THETA)
   YGRPI(I) = 175.*SIN(THETA)
100 CONTINUE
CALL MOVEA(XGRPI(1), YGRPI(1))
CALL MOVEA(0.0, 0.0)
C 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, XM0, YMIN, YMAX, XM1, XM2, YM1, YM2, SYMIN, SYMAX
REAL XPLBA(11), YPLBA(11), XCIRC(11), YCIRC(11)
REAL XPLMD(11), YPLMD(11), XPLED(11), YPLED(11)
REAL THONI, THINT, TH1, TH2, X, Y, Z, R, L1, L2, ZIN, RIN
REAL XBT, YBT, XBB, YBB, XMT1, YMT1, XMB1, YMB1, GRIPX

Appendix C.
REAL XMT2, YMT2, XMB2, YMB2, XET, YET, XEB, YEB, GRIPY
REAL XARM4, YARM4, ZARM4, THETA, PI
CHARACTER*7 TYPE, CELNAM=10, ANSW=1

COMMON/G/ SS, SLPAY, PART1, PART2, PAYOLD, VAL1, VAL2, RETVAL, BASC
COMMON/H/ THONI, THMTHI, 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/H/ XBT, YBT, XBB, YBB, XMT1, YMT1, XMB1, YMB1
COMMON/J/ 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/G/ P16, P17, P18, P19, P20, P21, P22, P23, P24, P25, P26, P27, P28
COMMON/R/ P29, P30, P31, P32

CALL THINDO(751, 883, 0, 400)
CALL DMINDO(0, 10, 0, 100)

CALL TCSIND(11)
CALL MOVABS(750, 370)
CALL DRMABS(883, 370)
CALL MOVABS(750, 0)
CALL DRMABS(883, 0)
CALL DRMABS(883, 400)
CALL DRMABS(750, 400)
CALL DRMABS(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 IMPORT
CALL OUTPUT
RETURN
END

C WINDOW FOR THE IMPORTS ONLY FOR LATER ON/OFF FACILITY

SUBROUTINE IMPORT

INTEGER ISCAN, IASC, MCHAR, 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, THMTHI, 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 XARM4, YARM4, ZARM4, THETA, PI
CHARACTER*7 TYPE, CELNAM=10, ANSW=1, DUMMY=60

COMMON/G/ SS, SLPAY, PART1, PART2, PAYOLD, VAL1, VAL2, RETVAL, BASC
COMMON/H/ THONI, THMTHI, 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

Appendix C.
CALL THINDO(883, 958, 0, 400)
CALL TCSIND(11)
CALL MOVABS(883, 370)
CALL DRAABS(958, 370)
CALL TCSIND(4)
CALL STROUT(917, 385, 'DI', 2)
CALL TCSIND(11)
CALL MOVABS(883, 0)
CALL DRAABS(958, 0)
CALL DRAABS(958, 400)
CALL DRAABS(883, 400)
CALL DRAABS(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 .50. 1) THEN
CALL $TROUT(917,191,'X',1)
ELSEIF (P8 .50. 0) THEN
CALL STROUT(917,191,'X',1)
ENDIF

IF (P9 .50. 1) THEN
CALL $TROUT(917,169,'X',1)
ELSEIF (P9 .50. 0) THEN 
CALL STROUT(917,169,'X',1)
ENDIF

IF (P10 .50. 1) THEN 
CALL $TROUT(917,147,'X',1)
ELSEIF (P10 .50. 0) THEN 
CALL STROUT(917,147,'X',1)
ENDIF

IF (P11 .50. 1) THEN 
CALL $TROUT(917,125,'X',1)
ELSEIF (P11 .50. 0) THEN 
CALL STROUT(917,125,'X',1)
ENDIF

IF (P12 .50. 1) THEN 
CALL $TROUT(917,103,'X',1)
ELSEIF (P12 .50. 0) THEN 
CALL STROUT(917,103,'X',1)
ENDIF

IF (P13 .50. 1) THEN 
CALL $TROUT(917,81,'X',1)
ELSEIF (P13 .50. 0) THEN 
CALL STROUT(917,81,'X',1)
ENDIF

IF (P14 .50. 1) THEN 
CALL $TROUT(917,59,'X',1)
ELSEIF (P14 .50. 0) THEN 
CALL STROUT(917,59,'X',1)
ENDIF

IF (P15 .50. 1) THEN 
CALL $TROUT(917,37,'X',1)
ELSEIF (P15 .50. 0) THEN 
CALL STROUT(917,37,'X',1)
ENDIF

IF (P16 .50. 1) THEN 
CALL $TROUT(917,15,'X',1)
ELSEIF (P16 .50. 0) THEN 
CALL STROUT(917,15,'X',1)
ENDIF

RETURN
REAL XPLBA(11), YPLBA(11), XCIRC(11), YCIRC(11)
REAL XPLMD(11), YPLMD(11), XPLED(11), YPLED(11)
REAL THONI, THTNI, 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, THTNI, ZIN, RIN, CODE, CELNAM, VALS
COMMON/J/ XMIN, XMAX, YMIN, YMAX, XMIN, XMAX, YMIN, SYMAX
COMMON/K/ XPLBA, YPLBA, XCIRC, YCIRC, XPLMD, YPLMD
COMMON/L/ XPLED, YPLED, TH1, TH2, X, Y, R, L1, L2
COMMON/M/ XBT, YBT, XBB, YBB, XMT1, YMT1, XMB1, YMB1
COMMON/U/ XMT2, YMT2, XMB2, YMB2, XET, YET, XEB, YEB
COMMON/G/ 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/G/ P16, P17, P18, P19, P20, P21, P22, P23, P24, P25, P26, P27, P28
COMMON/R/ P29, P30, P31, P32

CALL THINDO(958, 1023, 0, 400)
CALL TCSIND(11)
CALL MOVABS(958, 370)
CALL DRHABS(1023, 370)

CALL TCSIND(4)
CALL STROUT(975, 385, 'DO', 2)

CALL TCSIND(11)
CALL MOVABS(958, 0)
CALL DRHABS(1023, 0)
CALL DRHABS(1023, 400)
CALL DRHABS(958, 400)
CALL DRHABS(958, 0)

HERE FOR PORT TWO WHICH IS THE GRIPPER, THE GRIPPER WINDOW IS CALLED AND THE GRIPPER STATUS IS INDICATED BY OPEN/CLOSE.

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, 325, 'X', 1)
CALL THINDO(751, 059, 672, 780)
CALL DNINDO(-400., 400., -400., 400.)
CALL STROUT(920, 700, ' ', 5)
CALL STROUT(920, 700, 'CLOSE', 5)
ELSEIF (P18 .EQ. 0) THEN
CALL STROUT(975, 325, ' ', 1)
CALL THINDO(751, 059, 672, 780)
CALL DNINDO(-400., 400., -400., 400.)
CALL STROUT(920, 700, ' ', 5)
CALL STROUT(920, 700, 'OPEN ', 5)
ENDIF

CALL THINDO(958, 1023, 0, 400)

IF (P19 .EQ. 1) THEN

Appendix C.
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(975,235,' ',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,KCHR)

INTEGER ISCAN, IASC, NCHAR, BASC, KOUNT, VAL1, VAL2, RETVAL
REAL XMIN, XMNO, XMMA, XMNN, XMMA, XMNN, XMMA, XMNN
REAL XPLBA(11), YPBLA(11), XCIRC(11), YCIRC(11)
REAL XPLMD(11), YPBLMD(11), XPLED(11), YPLED(11)
REAL THON1, THHI, THZ, 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

DIMENSION KSTR(1), KAS(80)
CHARACTER*7 TYPE, CELNAM*10, ANSW*1

COMMON/G/ SS, SLPAY, PART1, PART2, PAYOLD, VAL1, VAL2, RETVAL, BASC
COMMON/H/ THON1, THHI, THZ, XIN, XMAX, XMNO, XMMA, XMNN, XMMA, XMNN, XMMA, XMNN
COMMON/L/ XPLED, YPLED, TH1, THZ, X, Y, Z, R, L1, L2
COMMON/M/ XBT, YBT, XBB, YBB, XMT1, YMT1, XMB1, YMB1
COMMON/M/ XMT2, YMT2, XMB2, YMB2, XET, YET, XEB, YEB
COMMON/O/ XARM, YARM, ZARM, THETA, PI, GRIPX, GRIPY
COMMON/O/ P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P11, P12, P13, P14, P15
COMMON/G/ P16, P17, P18, P19, P20, P21, P22, P23, P24, P25, P26, P27, P28
COMMON/R/ P29, P30, P31, P32

CALL KAMZAS (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, XMNO, XMMA, XMNN, XMMA, XMNN, XMMA, XMNN
REAL XPLBA(11), YPBLA(11), XCIRC(11), YCIRC(11)
REAL XPLMD(11), YPBLMD(11), XPLED(11), YPLED(11)
REAL THON1, THHI, THZ, 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

DIMENSION KSTR(1), KAS(80)
CHARACTER*7 TYPE, CELNAM*10, ANSW*1

Appendix C.
CALL TCSIND(2)
CALL MOVABS(0,0)
CALL DRNABS(750,0)
CALL DRNABS(750,777)
CALL DRNABS(0,777)
CALL DRNABS(0,0)

C SIDEVIEW WINDOW FRAME

CALL TCSIND(10)
CALL MOVABS(751,400)
CALL DRNABS(1025,400)
CALL DRNABS(1025,671)
CALL DRNABS(751,671)
CALL DRNABS(751,400)

C DEAD SPACE AND GRIPPER WINDOW FRAME

CALL MOVABS(750,672)
CALL DRNABS(1025,672)
CALL DRNABS(1025,777)
CALL DRNABS(750,777)
CALL DRNABS(750,672)

RETURN

END

C CALCULATE THE POINTS FOR THE WORKSPACE IN PLAN VIEW

SUBROUTINE MKSPACE

INTEGER ISCAN, IASC, NCHAR, NCHAR, KOUNT, VALI, VAL2, RETVAL
REAL XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX, XMIN, YMIN, ZMIN
REAL XPLBA(11), YPLBA(11), XCIRC(11), YCIRC(11)
REAL XPLMD(11), YPLMD(11), XPLED(11), YPLED(11)
REAL THONI, THTHI, 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 GRIPX
REAL XARM(4), YARM(4), ZARM(4), THETA, PI
REAL XWORK(675), YWORK(675)
CHARACTER TYPE, CELNAM*10, ANS*1

COMMON/G/ SS, SLPAY, PART1, PART2, PAYOLD, VAL1, VAL2, RETVAL, BASC
COMMON/H/ THONI, THTHI, ZIN, RIN, CODE, CELNAM, VALS
COMMON/J/ XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX, XMIN, YMIN, ZMIN
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) - 375.077
   YWORK(IC) = 250. * SIN(THETA) - 136.808
400 CONTINUE

DO 500 I = 238, 385
   IC = 576 - I
   THETA = I * PI / 180.
   XWORK(IC) = 286. * COS(THETA)
   YWORK(IC) = 286. * SIN(THETA)
500 CONTINUE

DO 600 I = 135, 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 DRAMA(XWORK(I), YWORK(I))
800 CONTINUE

RETURN
END
Appendix D.

THIS SECTION CONTAINS THE FOLLOWING PROGRAMS:

(1) DEFEPQ
(2) DEFSETU
(3) DISPSETU
(4) DIRECTORY

----------------------------------------------------------------------------------------------------------------------------------------
C (1) DEFEPQ - PROGRAM TO DEFINE EQUIPMENTS

SUBROUTINE DEFEQP (BASC, NAME, POSX, POSY, POSZ)

DIMENSION INFIL(80)
CHARACTER INFIL*1, NAME*10, NTY*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 SETEM0D(0)
CALL TCSMOD(1)
CALL TCSIND(2)

OPEN(7, FILE=NAME, STATUS='OLD')
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) NTY
55 FORMAT(1S)
C IF THE DATA IS A LINE DATA - 110

IF (NTTY .EQ. '110') THEN
  ICOM = 0
  DO 26 M = 5, 72
  IF (INFL(M:M) .EQ. ',' ) THEN
    ICOM = ICOM + 1
    J(ICOM) = M
    ENDFI
  IF (INFL(M:M) .EQ. ';') THEN
    ICOM = ICOM + 1
    J(ICOM) = M
    GOTO 27
  ENDFI
  26 CONTINUE

  27 K = J(1) - 1
  READ(INFL(5:K), *, X(1))
  K = J(1) + 1
  L = J(2) - 1
  READ(INFL(K:L), *, Y(1))
  K = J(2) + 1
  L = J(3) - 1
  READ(INFL(K:L), *, Z(1))
  K = J(3) + 1
  L = J(4) - 1
  READ(INFL(K:L), *, X(2))
  K = J(4) + 1
  L = J(5) - 1
  READ(INFL(K:L), *, Y(2))
  K = J(5) + 1
  L = J(6) - 1
  READ(INFL(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 DRAM THE PLAN OF EGP. - EITHER AS THE MAJOR OR MINOR VIEW

IF (BASC. EQ. 80) THEN
  CALL TMIND010, 750, 0, 750)
  CALL DMIND01XMIN, XMAX, YMIN, YMAX)
ELSE
  CALL TMIND01751, 1023, 400, 672)
  CALL DMIND01XMIN, XMAX, YMIN, YMAX)
ENDIF

  CALL MOVEA1(X11), Y11)
  CALL DRAMA1(X12), Y12)

C DRAM THE SIDEVIEW OF EGP. - EITHER AS THE MAJOR OR MINOR VIEW

IF (BASC. EQ. 80) THEN
  CALL TMIND01751, 1023, 400, 672)
  CALL DMIND01XMIN, XMAX, YMIN, YMAX)
ELSE
  CALL TMIND010, 750, 0, 750)
  CALL DMIND01XMIN, XMAX, YMIN, YMAX)
ENDIF

Appendix D.
CALL MOVEA(Y(1),Z(1))
CALL DRANA(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
        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 TNINDO(0,750,0,750)
        CALL DHINDO(XMIN,XMAX,YMIN,YMAX)
    ELSE
        CALL TNINDO(751,1023,400,672)
        CALL DHINDO(XMIN,XMAX,YMIN,YMAX)
    ENDIF
    CALL MOVDA(A,B)
    CALL DRANA(A,B)

C  DRAW THE SIDEVIEW OF EQP. - EITHER AS THE MAJOR OR MINOR VIEW

    IF (BASC. EQ. 80) THEN
        CALL TNINDO(751,1023,400,672)
        CALL DHINDO(SXMIN,SXMAX,SYMIN,SYMAX)
    ELSE
        CALL TNINDO(0,750,0,750)
        CALL DHINDO(SXMIN,SXMAX,SYMIN,SYMAX)
    ENDIF
    CALL MOYEA(B,C)
    CALL DRANA(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  
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,EGPN*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 G,T,M,ICOM,ICOMP,K,L
DATA INFIL/80·¤·' '/
DATA ICOM/0/ 
DATA ICOMP/0/ 
DATA PI/3.1428/ _
DATA AEXT/'DAT'/
C = I I '

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 CONCAT(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?',68,68,31)
CALL TTOUT(0,12,10,'(MAXIMUM OF EIGHT)',18,18,31)
CALL LOCATE(0,11,60)

READ(*,120 IN

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,35,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
      ENS
ELSE
IF (Q .LE. 0) 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(*,25)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 52 M = 5,72
    IF (INFIL(M:M) .EQ. ',') THEN
        ICOM =ICOM + 1
        J(ICOM) = M
    ENDIF
ENDDO

Appendix D.
ENDIF

IF (INFL(M:M) .EQ. ' ') THEN
  ICOM = ICOM + 1
  J(ICOM) = M
ENDIF
GOTO 27
ENDIF

26 CONTINUE

27 K = J(1) - 1
READ(INFL(5:K),*1)X(1)
K = J(1) + 1
L = J(2) - 1
READ(INFL(K:L),*1)Y(1)
K = J(2) + 1
L = J(3) - 1
READ(INFL(K:L),*1)Z(1)
K = J(3) + 1
L = J(4) - 1
READ(INFL(K:L),*1)X(2)
K = J(4) + 1
L = J(5) - 1
READ(INFL(K:L),*1)Y(2)
K = J(5) + 1
L = J(6) - 1
READ(INFL(K:L),*1)Z(2)

C AFTER READING THE X Y Z VALUES FORM THE EGP 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(6,15)X(1),Y(1),Z(1),X(2),Y(2),Z(2)
15 FORMAT('110',1H,,F8.1H,,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 (INFL(M:M) .EQ. ' ') THEN
    ICOMP = ICOMP + 1
    J(ICOMP) = M
  80 CONTINUE
ENDIF

IF (INFIL(M:M) .EQ. '1') 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,ROM,COLUMN,BUF,LENGTH,SIZE,HRATT)

INTEGER = 2 PAGE,ROM,COLUMN,SIZE,HRATT

Appendix D. 200
CALL LOCATE(PAGE,ROM,COLUMN)
DO 10 I = 1,LENGTH
   CALL NRCHAT(PAGE,BUF(I),MRATT,1)
10   CONTINUE
   RETURN
END

SUBROUTINE CONCA (AFILE, 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
   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
C THIS PROGRAM DISPSETU.FOR GETS THE NAME OF THE EQUIPMENT
C TO BE DISPLAYED, AND CREATES THE BATCH FILE DISSEQP.BAT

PROGRAM DISPSETU

REAL XMIN,XMAX,YMIN,YMAX,SXMIN,SYM MIN,SYM MAX
REAL XPLBA(11),YPLBA(11),XCI R(C11),YCIRC(11)
REAL XPLMD(11),YPLMD(11),XP LED(11),YPLED(11)
REAL THON,THM1,THM2,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,SYM MAX,SYM MAX

Appendix D. 201
COMMON/B/ XPLBA,YPLBA,XCIRC,YCIRC,XPLMD,YPLMD,XPLED,YPLED
COMMON/C/ THONI,THTHI,TH1,TH2,X,Y,Z,R,L1,L2
COMMON/D/ XBT,YBT,XBB,YBB,XB1,YB1,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
SYMAY = 684.0
KOUNT = 1
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

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',23,23,31)
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(10,6,25,'---------------------------',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,'(MAXIMUM FIVE CHARACTERS)'
CALL LOCATE(0,11,61)
READ(*,35)CELNAM
35 FORMAT(A$)

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(*,*) A
GOTO 200

C SELECTING BETWEEN THE TWO VIEWS 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(IASC,ISCAN,NCHAR)
BASC = IASC

C THE GRAPHICS DISPLAYED IS CALLED
CALL DSMPAL(2)
CALL INITT(5)
CALL REDRAM

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(IASC,ISCAN,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(IASC,ISCAN,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 DSMA(0)

Appendix D. 203
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, THTNI, TH1, THZ, 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, THETA, PI, ASPECT
REAL X1, X2, Y1, Y2, X, Y, Z, R, L1, L2
REAL XMTZ, YMTZ, XMBZ, YMBZ, XET, YET, XEB, YEB
REAL XARM(4), YARM(4), ZARM
REAL X1, XZ, Y1, YZ, XC, YC, DELX, DELY, DXHY, XAVG, YAVG
INTEGER BASC
CHARACTER*7 TYPE, CELNAM*10, ANS*1

COMMON /B/ XPLBA, YPLBA, XCIRC, YCIRC, XPLMD, YPLMD, XPLED, YPLED
COMMON /C/ THONI, THTNI, TH1, THZ, 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 THINDO(0.750, 0.750)
CALL DHINDO(XMIN, XMAX, YMIN, YMAX)
CALL STROUT
* (55, 65, 'SELECT NEW MINDON CORNER PRESS SPACE BAR', 40)
CALL VCURSR (ICHAR, X1, Y1)
CALL STROUT
* (55, 60, '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
DELEX = ABS(XZ-X1)
DELEY = ABS(Y2-Y1)
XAVG = (X2+Y1)/2.0
YAVG = (X2+Y1)/2.0
IF (DELEX .EQ. 0.0) DELX = 0.001
IF (DELEY .EQ. 0.0) DELY = 0.001
ASPECT = DELX/DELEY
IF (ASPECT .LT. 1.0) DELX = DELX * 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 REDRAN

ELSE

C IF MAJOR VIEW IS SIDE VIEW

CALL THINDO(0.750, 0.750)
CALL DHINDO(XMIN, XMAX, YMIN, YMAX)
CALL STROUT
* (55, 65, 'SELECT NEW MINDON CORNER PRESS SPACE BAR', 40)
CALL VCURSR (ICHAR, X1, Y1)
CALL STROUT

Appendix D. 204
CALL VCURSR (ICHAR,X2,Y2)

SXMIN = X1
IF (X2 .LT. X1) SXMIN=X2
SYMIN =Y1
IF (Y2 .LT. Y1) SYMIN=Y2
DELF = ABS(X2-X1)
DELY = ABS(Y2-Y1)
YAVG = (Y2+Y1)/2.0
XAVG = (X2+X1)/2.0
IF (DELF .EQ. 0.0 ) DELF =0.001
IF (DELY .EQ. 0.0 ) DELY =0.001
ASPECT = DELY /DELF
IF (ASPECT .LT. 1.0) DELY = DELY *1.0/ASPECT
IF (ASPECT .GT. 1.0) DELF = DELF *ASPECT /1.0
SYMIN =YAVG -DELY/2.0
SXMIN =XAVG -DELF/2.0
SXMAX = SXMIN + DELF
SYMAX = SYMIN + DELY
CALL REDRAN
ENDIF

C TO PAN THE DRAWING
C IF MAJOR VIEW IS PLAN VIEW

IF (BASC .EQ. 80) THEN
CALL TMINDO(0,750,0,750)
CALL DMINDO(SXMIN,SXMAX,SYMIN,SYMAX)
CALL STROUT(205,80,'DO YOU WANT TO PAN (Y/N)',24)
CALL TC$KEY(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 REDRAN
ELSE
C IF MAJOR VIEW IS SIDE VIEW

CALL TMINDO(0,750,0,750)
CALL DMINDO(SXMIN,SXMAX,SYMIN,SYMAX)
CALL STROUT(205,80,'DO YOU WANT TO PAN (Y/N)',24)
CALL TC$KEY(ISCAN,IASC,NCHAR)
IF (IASC .EQ. 89) THEN
GOTO 300
ELSE

Appendix D.  205
GOTO 200
ENDIF

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

CALL STROUT
#(55,60,'ENTER NEW (X,Y) FOR CENTER OF MODEL --> ',40)
CALL TSEND
READ (*,*, END = 22) XC, YC

DELX = SXMAX - SXMIN
DELY = SYMAX - SYMIN
SXMIN = XC - DELX/2.0
SYMIN = YC - DELY/2.0
SXMAX = SXMIN + DELX
SYMAX = SYMIN + DELY

CALL REDRAN
ENDIF

200 RETURN
END

C TO REDRAN THE WHOLE PICTURE - STANDARD VIEW AND WITH WORKCELL

SUBROUTINE REDRAN

REAL XMIN,XMAX,YMIN,YMAX,XMIN,YMIN,SXMIN,SXMAX,YMIN,SYMAX
REAL XPLBA(I1),YPLBA(I1),XCIRC(I1),YCIRC(I1)
REAL XPLMD(I1),YPLMD(I1),XPLED(I1),YPLED(I1)
REAL THONI,THTNI,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 CK(2), CZ(2), N(2), NY(2), NZ(2)
REAL D,E,F,ND,NE,NF
INTEGER ICOM,ICOMP,J(6)
INTEGER ISCANN,IASC,JCHAR,BASC,KOUNT
CHARACTER*7 TYPE,ANS*I
CHARACTER INFIL*I, NAME*20, AFILE*I, CELNAM*I
CHARACTER C*1, D*1, NTTY*3, AEXT*3
DIMENSION INFIL(80)

COMMON/A/ XMIN,XMAX,YMIN,YMAX,XMIN,YMIN,SXMIN,SXMAX,YMIN,SYMAX
COMMON/B/ XPLBA,YPLBA,XCIRC,YCIRC,XPLMD,YPLMD,XPLED,YPLED
COMMON/C/ THONI,THTNI,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 SETFHM
CALL TCSFID
CALL TCSIND(14)

C CHECKING TO SEE IF A WORKCELL HAS TO BE DISPLAYED

OPEN (7, FILE=CELNAM, STATUS='OLD', ERR=100)

5 CONTINUE
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
      ENDIF
   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 TMINDO(0,750,0,750)
   CALL DMINDO(XMIN,XMAX,YMIN,YMAX)
ELSE
   CALL TMINDO(751,1023,400,672)
   CALL DMINDO(XMIN,XMAX,YMIN,YMAX)
ENDIF
CALL MOVEA(CX(1),CY(1))
CALL DRAMA(CX(2),CY(2))
C DRAW THE SIDEVIEW OF EQP. - EITHER AS THE MAJOR OR MINOR VIEW

IF (BASC .EQ. 80) THEN
   CALL TMINDO(751,1023,400,672)
   CALL DMINDO(XMIN,XMAX,YMIN,YMAX)
ELSE
   CALL TMINDO(0,750,0,750)
   CALL DMINDO(XMIN,XMAX,YMIN,YMAX)
ENDIF
CALL MOVEA(CY(1),CZ(1))
CALL DRAMA(CY(2),CZ(2))

Appendix D. 207
GOTO 5

C IF THE DATA IS A POINT DATA - 116

ELSEIF INTTY .EQ. '116') THEN
   ICOMP = 0
   DO 30 Q = 5, 72
      IF (INFILE(Q:Q) .EQ. ', ') THEN
         ICOMP = ICOMP + 1
         J(ICOMP) = Q
      ENDIF
      IF (INFILE(Q:Q) .EQ. 'S') THEN
         ICOMP = ICOMP + 1
         J(ICOMP) = Q
         GOTO 40
      ENDIF
   30 CONTINUE

K = J(1) - 1
READ(INFILE(5:K), *) D
K = J(1) + 1
L = J(2) - 1
READ(INFILE(K:L), *) E
K = J(2) + 1
L = J(3) - 1
READ(INFILE(K:L), *) F

C DRAH THE PLAN OF EQP. - EITHER AS THE MAJOR OR MINOR VIEW

IF (BASIC .EQ. 80) THEN
   CALL THINDO(0, 750, 0, 750)
   CALL DMINDO(XMIN, XMAX, YMIN, YMAX)
ELSE
   CALL THINDO(751, 1023, 400, 672)
   CALL DMINDO(XMIN, XMAX, YMIN, YMAX)
ENDIF

CALL MOVEA(D, E)
CALL DRAMA(D, E)

C DRAH THE SIDE VIEW OF EQP. - EITHER AS THE MAJOR OR MINOR VIEW

IF (BASIC .EQ. 80) THEN
   CALL THINDO(751, 1023, 400, 672)
   CALL DMINDO(SYMIN, SYMAX, SYMIN, SYMAX)
ELSE
   CALL THINDO(0, 750, 0, 750)
   CALL DMINDO(SYMIN, SYMAX, SYMIN, SYMAX)
ENDIF

CALL MOVEA(E, F)
CALL DRAMA(E, F)

GOTO 5
ELSE
   GOTO 5
ENDIF

200 CLOSE(7)

100 CALL PLANSIDE

   CALL FRAME RETURN

Appendix D.  208
SUBROUTINE PLANSIDE, FRAMES, AND MSPACE ARE THE NEXT THREE
SUBROUTINES IN THIS PROGRAM. THEY ARE THE SAME SUBROUTINES
IN THE PROGRAM DEPUTY1 AND HENCE NOT REPEATED IN THIS PRINTOUT.

SUBROUTINE TTOUT

INTEGER PAGE,ROM,COLUMN,BUF,LENGTH,SIZE,MNAME
CHARACTER NAME(20)
CHARACTER BLANK

CALL LOCATE(PAGE,ROM,COLUMN)

DO 10 I = 1,LENGTH
   CALL NRCHAT(PAGE,BUF(I),MNAME,1)
10 CONTINUE
RETURN
END

FOR LOCATING A STRING IN THE SCREEN

SUBROUTINE STROUT(INX,INY,KSTR,NCHR)

DIMENSION KSTR(1),KAS(80)

CALL KAM2AS(NCHR,KSTR,KAS)
CALL MOVABS(INX,INY)
CALL ANSTR(NCHR,KAS)
RETURN
END

CONCA

SUBROUTINE CONCA(AFILN,AEXT)

CHARACTER NAME AFILN
CHARACTER AEXT

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

140 AFILN(I:I) = '.!

DO 150 J = 1,3
   AFILN(I+J:I+J) = AEXT(J:J)
150 CONTINUE

CAPITALIZE

DO 200 K = I,I+1
   IF(AFILN(K:K) .GE. 'a' .AND. AFILN(K:K) .LE. 'z') then
```
AFILE(K:K) = CHAR (ICHAR (AFILE(K:K)) -32 )
END
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',
#48,68,31)
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
```

Appendix D. 210
SUBROUTINE TTOUT(PAGE, ROM, COLUMN, BUF, LENGTH, SIZE, WRATT)

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

CALL LOCATE(PAGE, ROM, COLUMN)

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

RETURN
END
The vita has been removed from the scanned document.