UNCLASSIFIED

AD NUMBER

ADB028910

LIMITATION CHANGES

TO:
Approved for public release; distribution is unlimited. Document partially illegible.

FROM:
Distribution authorized to U.S. Gov't. agencies only; Test and Evaluation; JUN 1978. Other requests shall be referred to Air Force Rome Air Development Center, IRAE, Griffiss AFB, NY 13441. Document partially illegible.

AUTHORITY

RADC, USAF ltr, 26 sep 1980

THIS PAGE IS UNCLASSIFIED
THIS REPORT HAS BEEN DELIMITED AND CLEARED FOR PUBLIC RELEASE UNDER DOD DIRECTIVE 5200.20 AND NO RESTRICTIONS ARE IMPOSED UPON ITS USE AND DISCLOSURE.

DISTRIBUTION STATEMENT A
APPROVED FOR PUBLIC RELEASE;
DISTRIBUTION UNLIMITED.
INTERACTIVE PROGRAMMING AND ANALYSIS AIDS (IPAA)

Mr. D. W. Johnson
Harris Corporation ESD

Distribution limited to U.S. Government agencies only; test and evaluation; June 1978. Other requests for this document must be referred to RADC (IRAE), Griffiss AFB NY 13441.
DISCLAIMER NOTICE

THIS DOCUMENT IS BEST QUALITY PRACTICABLE. THE COPY FURNISHED TO DDC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.
RADC-TR-78-143 has been reviewed and is approved for publication.

APPROVED:  

JOSEPH CAMERA  
Project Engineer

APPROVED:  

HOWARD DAVIS  
Technical Director  
Intelligence and Reconnaissance Division

FOR THE COMMANDER:  

JOHN P. HUSS  
Acting Chief, Plans Office

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (IRAЕ), Griffiss AFB NY 13441. This will assist us in maintaining a current mailing list.

Do not return this copy. Retain or destroy.
**Title:** INTERACTIVE PROGRAMMING AND ANALYSIS AIDS (IPAA)

**Performing Organization:** Harris Corporation ESD

**Address:** P.O. Box 37, Melbourne FL 32901

**Report Date:** Sep 77 - Apr 78

**Final Technical Report, June 1978.**

**Distribution Statement:** Distribution limited to U.S. Government agencies only; test and evaluation; June 1978. Other requests for this document must be referred to RADC (IRAE), Griffiss AFB NY 13441.

**Keywords:**
- Computer Graphics
- Diagram
- Interactive Graphics
- Modeling
- Analog Simulation
- Interactive Analysis

**Abstract:** A sophisticated, previously-developed concept of an Interactive Analysis Automated Aid has been successfully installed on a Univac 1110 Computer System. This aid is capable of significantly enhancing the analyst's response by provision of a highly interactive means for:
- (a) modeling new analysis problems as diagrams,
- (b) executing the analysis diagrams, and
- (c) evaluating the execution results.

Analysis diagrams can be created, edited, modified, stored, and evaluated.
retrieved, annotated, and executed at the analyst's graphics station. The diagrams themselves are mathematical in nature to provide the widest flexibility and applicability. Available special features include: handling multipage diagrams, parallel analysis processes, nonlinear analysis and automated input and output of sampled data.
This report presents the results of the Interactive Programming and Analysis Aid Implementation. The report addresses the specific capabilities of an interactive computerized graphics system which merges the strength of an analyst with that of the computer to perform analysis of a broad class of problems.

This report was written and the design and implementation of the software system was done under the direction of Mr. D. W. Johnson, Harris ESD, Melbourne, Florida.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Paragraph</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1</td>
<td>Objective</td>
<td>1</td>
</tr>
<tr>
<td>1.2</td>
<td>General Nature of Analysis</td>
<td>1</td>
</tr>
<tr>
<td>2.0</td>
<td>CONSIDERATIONS IN THE DESIGN OF ITAS</td>
<td>4</td>
</tr>
<tr>
<td>2.1</td>
<td>Analyst</td>
<td>4</td>
</tr>
<tr>
<td>2.2</td>
<td>Basic Algorithms</td>
<td>4</td>
</tr>
<tr>
<td>2.3</td>
<td>Diagram Formation</td>
<td>4</td>
</tr>
<tr>
<td>2.4</td>
<td>Diagram Construction Errors</td>
<td>5</td>
</tr>
<tr>
<td>2.5</td>
<td>Man/Machine Interaction</td>
<td>5</td>
</tr>
<tr>
<td>3.0</td>
<td>DESIGN OF THE IPAA SYSTEM</td>
<td>6</td>
</tr>
<tr>
<td>3.1</td>
<td>IPAA and Its FTD Environment</td>
<td>6</td>
</tr>
<tr>
<td>3.2</td>
<td>Data Files</td>
<td>8</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Display Data File</td>
<td>8</td>
</tr>
<tr>
<td>3.2.2</td>
<td>PIF Data File</td>
<td>8</td>
</tr>
<tr>
<td>3.2.3</td>
<td>User Data Files</td>
<td>8</td>
</tr>
<tr>
<td>3.3</td>
<td>Building Blocks</td>
<td>9</td>
</tr>
<tr>
<td>3.4</td>
<td>Graphics Module Functions</td>
<td>9</td>
</tr>
<tr>
<td>3.4.1</td>
<td>Diagram Editing</td>
<td>9</td>
</tr>
<tr>
<td>3.4.1.1</td>
<td>Add Block</td>
<td>9</td>
</tr>
<tr>
<td>3.4.1.2</td>
<td>Block Terminus Points</td>
<td>19</td>
</tr>
<tr>
<td>3.4.1.3</td>
<td>Block Connection</td>
<td>19</td>
</tr>
<tr>
<td>3.4.1.4</td>
<td>Add Text</td>
<td>19</td>
</tr>
<tr>
<td>3.4.1.5</td>
<td>Modify Parameters</td>
<td>19</td>
</tr>
<tr>
<td>3.4.1.6</td>
<td>Erase Pin Connection</td>
<td>20</td>
</tr>
<tr>
<td>3.4.1.7</td>
<td>Erase Network</td>
<td>20</td>
</tr>
<tr>
<td>3.4.1.8</td>
<td>Erase Block</td>
<td>21</td>
</tr>
<tr>
<td>3.4.1.9</td>
<td>Erase Text</td>
<td>21</td>
</tr>
<tr>
<td>3.4.2</td>
<td>Diagram Storage and Retrieval</td>
<td>21</td>
</tr>
<tr>
<td>3.4.2.1</td>
<td>Save Diagram</td>
<td>22</td>
</tr>
</tbody>
</table>
## TABLE OF CONTENTS (Continued)

<table>
<thead>
<tr>
<th>Paragraph</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.4.2.2</td>
<td>Recall Diagram</td>
<td>22</td>
</tr>
<tr>
<td>3.4.3</td>
<td>Paging</td>
<td>23</td>
</tr>
<tr>
<td>3.4.4</td>
<td>Zooming</td>
<td>23</td>
</tr>
<tr>
<td>3.5</td>
<td>Analysis Execution Functions</td>
<td>25</td>
</tr>
<tr>
<td>3.5.1</td>
<td>Process Control Parameter Modification</td>
<td>25</td>
</tr>
<tr>
<td>3.5.2</td>
<td>IPAA Data Base</td>
<td>28</td>
</tr>
<tr>
<td>3.5.3</td>
<td>Block Ordering</td>
<td>28</td>
</tr>
<tr>
<td>3.5.4</td>
<td>Interpolation</td>
<td>31</td>
</tr>
<tr>
<td>4.0</td>
<td>COMPARATIVE ALGORITHMS</td>
<td>32</td>
</tr>
<tr>
<td>4.1</td>
<td>Algorithm Number One</td>
<td>32</td>
</tr>
<tr>
<td>4.1.1</td>
<td>Algorithm No. 1 - Inputs and Mathematical Formulation</td>
<td>33</td>
</tr>
<tr>
<td>4.1.2</td>
<td>Algorithm No. 1 - Comparison</td>
<td>35</td>
</tr>
<tr>
<td>4.2</td>
<td>Algorithm Number Two</td>
<td>36</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Algorithm No. 2 - Mathematical Formulation</td>
<td>36</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Algorithm No. 2 - Comparison</td>
<td>40</td>
</tr>
<tr>
<td>5.0</td>
<td>CONCLUSIONS</td>
<td>43</td>
</tr>
<tr>
<td>6.0</td>
<td>RECOMMENDATIONS</td>
<td>44</td>
</tr>
<tr>
<td>6.1</td>
<td>Macro Capability</td>
<td>45</td>
</tr>
<tr>
<td>6.1.1</td>
<td>Macro Definition</td>
<td>45</td>
</tr>
<tr>
<td>6.1.2</td>
<td>Macro Implementation</td>
<td>45</td>
</tr>
<tr>
<td>6.2</td>
<td>Graphic Display of Output Data</td>
<td>49</td>
</tr>
<tr>
<td>6.3</td>
<td>Current System Modifications</td>
<td>49</td>
</tr>
<tr>
<td>6.3.1</td>
<td>Uniscope Execution</td>
<td>49</td>
</tr>
<tr>
<td>6.3.2</td>
<td>Addition of Functional Blocks</td>
<td>49</td>
</tr>
<tr>
<td>7.0</td>
<td>REFERENCES</td>
<td>51</td>
</tr>
<tr>
<td>APPENDIX A</td>
<td>UTILIZATION OF LINES AS BLOCK CONNECTORS</td>
<td>A-1</td>
</tr>
<tr>
<td>APPENDIX B</td>
<td>ALGORITHMS LISTINGS AND DIAGRAMS</td>
<td>B-1</td>
</tr>
</tbody>
</table>
EVALUATION

This report describes a unique approach to solving the problem of inadequate man-machine interaction achieved by the conventional approach to coding scientific computer algorithms. The report examines the utilization of a graphical block diagram schematic approach to programming which enables a user to solve complex engineering problems interactively with the aid of a computer. This approach was evaluated through specific examples and proved to be a very powerful tool in the hands of trained analysts.

In addition, this report describes recommended modifications/additions to the IPAA Software System for future investigation.

JOSEPH CAMARA
Project Engineer
1.0 INTRODUCTION

This report describes the implementation of the Itas concept on the Univac 1110 Computing System at FTD, Wright Patterson AFB. This particular implementation of the Itas concept will be referred to as IPAA in the report. Background information on the Itas concept is also included for completeness.

1.1 Objective

The contract objective is to implement the Itas concept on the Univac 1110 Computing System which has Tektronix 4014 graphics terminals. This general-purpose computing system supports a wide range of analysis activities, and as such, the analysts have no standard data or data format. Therefore, a secondary contract objective is to allow the analyst to access a variety of data formats (both input and output) within the IPAA software system.

1.2 General Nature of Analysis

Most analysis activities can be based upon description of the problem under investigation through the use of a schematic diagram. Examples of such diagrams are plentiful; logic networks, flow charts, electronic schematics, mechanical diagrams, system diagrams, etc. Each of these schematic diagrams are basically topological graphs which are drawn to be readable and suggestive to the analyst trained in the particular discipline. As a result, the diagrams serve as primary vehicles for analyses, communication, instruction and reporting of results. Not only do these diagrams conveniently describe the problems to a trained analyst, but they are interpretable by the computer as they are drawn, thereby providing a high level man/machine interface "language".

For the broad class of analysis problems considered above, the general methodology of analysis can be given in terms of the six phases of activity shown in Figure 1. Although only the primary flow is shown, other flows are also possible as it is common practice for an analyst to
Figure 1.
The Six Phases of Analysis Methodology

START

PHASE I
MODELING OF PROBLEM

PHASE II
EXECUTION OF ANALYSIS

PHASE III
EVALUATION OF RESULTS

PHASE IV
DEFINITION OF NEW MODELING BLOCKS

PHASE V
DEFINITION OF NEW ANALYSIS PROCEDURES

PHASE VI
DEFINITION OF NEW DATA PRESENTATION FORMATS

END
backtrack to an earlier phase of activity to change the nature of the problem analysis in some fashion.

The general nature of analysis methodology is normally based on three consistently used phases:

I. Modeling of Problem
II. Execution of Analysis
III. Evaluation of Result

Less consistently utilized are the three support phases:

IV. Definition of new modeling block
V. Definition of new analysis procedures
VI. Definition of new data presentation formats

In order to understand the flow shown in Figure 1, assume that the analysis is electrical in nature. The analyst will then think of modeling his problem in terms of electrical circuits (Phase I). He will normally use the basic modeling blocks (in modeling theory called connectives, primitives, symbols or tools) of resistors, capacitors, inductors, current sources and voltage sources. Occasionally, he might be forced to create new modeling blocks (Phase IV) such as a nonlinear element. Once the problem has been modeled as a schematic circuit diagram, the analyst will analyze it (Phase II). This analysis will likely involve calculating the response to a particular input signal. In the process, the analyst may be required to develop new analysis procedures (Phase V) such as sensitivity analyses, transient analyses, controllability analyses, etc. Upon completion of analysis calculations, the analyst will evaluate the results of the analysis (Phase III). To do this, he will normally display the results in some manner such as waveform plots. On other occasions, he may devise other data presentation formats (Phase VI) such as Bode plots, histograms, etc., prior to evaluating the results.
2.0 CONSIDERATIONS IN THE DESIGN OF ITAS

Prior to the design of Itas certain desirable features were innumerated and carefully considered. Also taken into account in the Itas design was the important fact that Itas-aided analysis would be performed by specialized users in a restricted environment.

2.1 Analyst

In general, an analyst will not be a trained computer programmer though, most probably, he will have had significant exposure to computer programs. Thus the procedures of diagram generation and analysis execution should not be dependent on knowledge of computer techniques, computer language or computer jargon.

2.2 Basic Algorithms

The set of basic algorithms or blocks should encompass all the fundamental functions that an analyst might reasonably require in the constructions of analytic processes relating to the analysis of telemetry data. On the other hand, the set of primitive blocks should be small enough to be displayed on a small fixed portion of the CRT screen. Also the identification of the primitives on the CRT screen should be readily recognizable and there should be no ambiguity in the meaning or use of the blocks.

2.3 Diagram Formation

The interconnecting of the primitive blocks to form a processor should be a fairly simply and logical procedure. It is also reasonable to expect the analyst will wish to make changes in the diagram after he has examined the results of the analysis. Therefore, relatively minor changes should not require a complete redoing of the diagram. Should the user wish to make diagram changes while he is in the process of generating the diagram, the changes should be simple to incorporate.
2.4 **Diagram Construction Errors**

Inconsistencies in the construction of a diagram should be flagged and rejected by the program. The user should be able to continue after an error without undue interruption and with no loss in that portion of the diagram already completed.

2.5 **Man/Machine Interaction**

A most significant feature of Itas should be the merger of man's peculiar capabilities with those of a digital computer. For instance, a man can think in broad terms, can reason intuitively, can remember previous trials and benefit from them. On the other hand, the computer has great speed and accuracy and can do complex mathematical computations.
3.0 DESIGN OF THE IPAA SYSTEM

The IPAA Software System is designed to functionally adhere to the Itas concept; i.e., to provide basic tools to the analyst which he may use to create unique analysis sequences to process real or simulated data, existing in a variety of formats.

IPAA is made up of two functional subsystems; a set of graphics routines which the analyst uses to compose processing algorithms and a set of analysis execution routines which process the data in accordance with the previously generated algorithms.

3.1 IPAA and Its FTD Environment

IPAA software exists as a stand-alone program within the Univac 1110 executive operating system. IPAA is accessed by performing the instructions displayed in Figure 2.

More specifically, the IPAA System possesses the following individual features which are designed to improve the Itas concept in the Univac 1110 FTD working environment:

1) IPAA is designed to operate on the Univac 1110 host computer and operate within the constraints of the executive time-sharing oriented operating system.

2) The IPAA System makes full use of the Tektronix 4014 graphic display terminal as the prime means of analyst interaction. The IPAA software interfaces with the Tektronix 4014 terminal through the use of the TCS graphics display software package.

3) The IPAA software is coded in the Fortran V compiler language. The use of Fortran has two important benefits:
STEP 1 - SIGN ON TEKTRONIX TERMINAL WITH NORMAL FTD SIGN ON PROCEDURES.

STEP 2 - TTY LOCK

STEP 3 - @XQT FTD$*IPAA.IPAA

Figure 2. ACCESSING THE IPAA SOFTWARE
a) The software will be easy to maintain and/or modify.
b) The programs are self documented by the generous use of comment cards inserted in a uniform and precise manner.

4) The IPAA software is generally the result of a structured top/down system design. The resulting software is modularized according to specific functions and/or services. The software developed is thus easier to maintain.

3.2 Data Files

There are three types of data files closely associated with the IPAA System. In general, storage capability is provided for the diagram, processing instructions, and the input/output data.

3.2.1 Display Data File

All information associated with a diagram will be stored within a user-named data file. This includes all blocks, function codes, connections, and parameters of the nine full pages of display in a single data file.

3.2.2 PIF Data File

All user instructions pertaining to the format of the input and output data files and of the execution control parameters will be stored in the processor information file (PIF).

3.2.3 User Data Files

The specified user data files may or may not have been generated by the IPAA software system. The IPAA software system, however, does provide for the input and output of data in a great variety of formats.
3.3 Building Blocks

The IPAA diagram is based on interconnecting a set of primitive blocks most of which are mathematical operators. This mathematical structure being universal has direct and immediate utility for a variety of analysis disciplines. An IPAA graphics menu as shown in Figure 3 has a list of all of the primitive blocks in the upper right portion of the display page. Figure 4 is a description of each block including its inputs, outputs and parameters.

3.4 Graphics Module Functions

The graphics module provides the capability for an analyst to build data-processing diagrams on an interactive basis with the computer. The module maintains all diagram-associated data in a sophisticated database which is transferred to the analysis execution module when the diagram is to be executed. The module also saves diagrams for subsequent execution and/or modification.

3.4.1 Diagram Editing

The generation of a complete diagram is the process of combining the various diagram elements (blocks, lines, parameters, and text) into an executable entity. The editing function as considered here includes not only the addition of elements to a partially constructed diagram but also the deletion of unwanted elements that are a part of the diagram. The various editing functions are discussed in the following paragraphs under Section 3.4.1.

3.4.1.1 Add Block

The addition of a block requires, in general, two distinct operator actions; selection of the add block function and selection of the block's operational function. Selection of the add block function in IPAA automatically specifies screen position. While not necessarily aesthetically appealing, each block was chosen to be the same in size and shape and to be distinguishable only by an up to 6-character mnemonic
Figure 3. IPAA Graphics Menu
<table>
<thead>
<tr>
<th>BLOCK TYPE</th>
<th>BLOCK FUNCTION</th>
<th>NO. OF PARAMETERS</th>
<th>NOTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPUT</td>
<td>1. Enter data from a data file or</td>
<td>1</td>
<td>1. When used to bring data from a file, the parameter value must lie between +1 and +4 for default format, and between +1 and +6 for all other formats.</td>
</tr>
<tr>
<td></td>
<td>2. Used as a diagram page connector.</td>
<td></td>
<td>2. When used as a page connector, the parameter value must be negative and equal to the output block to which it is connected.</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>1. Writes data to a designated output file.</td>
<td>1</td>
<td>1. When used to output a file of data, the parameter value must lie between +1 and +4 default format, and between +1 and +6 for all other formats.</td>
</tr>
<tr>
<td></td>
<td>2. Used as a diagram page connector.</td>
<td></td>
<td>2. When used as a page connector, the parameter value must be negative.</td>
</tr>
<tr>
<td>TIME</td>
<td>Output the current time</td>
<td>0</td>
<td>The start time, stop time and time increment is specified during the Analysis Execution phase.</td>
</tr>
</tbody>
</table>

Figure 4. FUNCTIONAL BLOCK DEFINITIONS
<table>
<thead>
<tr>
<th>BLOCK TYPE</th>
<th>BLOCK FUNCTION</th>
<th>NO. OF PARAMETERS</th>
<th>NOTES</th>
</tr>
</thead>
</table>
| A          | MOD            | 1                | Parameter 1 = modulus  
|            |                |                  | $Y = \text{AMOD} \left[(A+B) + (C+D), P1\right]$ |
| A          | O HOLD         | 1                | Parameter 1 = fixed hold value  
|            |                |                  | $A=0 \quad Y=0$  
|            |                |                  | $A=1 \quad Y=P1$  
|            |                |                  | $A=0 \quad Y=B$  |
| A          | SIMP           | 3                | $X_n = A + B + C + D$  
|            |                |                  | $P_1 = \text{INITIAL VALUE} = Y_0$ or $Y_{n-1}$  
|            |                |                  | $P_2 = X_{n-2}$  
|            |                |                  | $P_3 = X_{n-1}$  
|            |                |                  | $Y_n = Y_{n-1} + \Delta t \left[X_{n-2} + 4X_{n-1} + X_n\right]/6$ |

Figure 4 (Continued)
<table>
<thead>
<tr>
<th>BLOCK TYPE</th>
<th>BLOCK FUNCTION</th>
<th>NO. OF PARAMETERS</th>
<th>NOTES</th>
</tr>
</thead>
</table>
| DELAY      | Provide a one cycle delay | 1 | P = Initial value  
|            |                 |                 | Y = P  
|            |                 |                 | P = X |
| CON        | Output a constant value | 1 | Y = C where C is entered as the parameter |
| GAIN       | Multiplies the sum of a series of inputs by a constant | 1 | Y = G * (A+B+C+D) where the parameter is the gain G |
| INTG1      | Trapezoidal integrator | 2 | Parameter 1 = initial value out of integrator (Y_{n-1})  
|            |                 |                 | Parameter 2 is X_{n-1} where X_n=A_n+B_n+C_n+D_n  
|            |                 |                 | Y_0 = Parameter 1  
|            |                 |                 | X_0 = Parameter 2  
|            |                 |                 | Y_n = Y_{n-1} + \Delta t/2(X_n + X_{n-1})  
|            |                 |                 | where \Delta t = time between successive points |

Figure 4 (Continued).
<table>
<thead>
<tr>
<th>BLOCK TYPE</th>
<th>BLOCK FUNCTION</th>
<th>NO. OF PARAMETERS</th>
<th>NOTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>A B C D</td>
<td>SINE</td>
<td>0</td>
<td>$Y = \sin [(A\times B) + (C\times D)]$</td>
</tr>
<tr>
<td>A B C D</td>
<td>COS</td>
<td>0</td>
<td>$Y = \cos [(A\times B) + (C\times D)]$</td>
</tr>
<tr>
<td>A B C D</td>
<td>ATAN</td>
<td>0</td>
<td>$Y = \tan^{-1} [(A\times B) + (C\times D)]$</td>
</tr>
<tr>
<td>A B C D</td>
<td>EXP</td>
<td>0</td>
<td>$Y = \exp [(A\times B) + (C\times D)]$</td>
</tr>
<tr>
<td>A B C D</td>
<td>ABS</td>
<td>0</td>
<td>$Y = \left</td>
</tr>
</tbody>
</table>

Figure 4 (Continued).
<table>
<thead>
<tr>
<th>BLOCK TYPE</th>
<th>BLOCK FUNCTION</th>
<th>NO. OF PARAMETERS</th>
<th>NOTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTG2</td>
<td>Not implemented</td>
<td></td>
<td></td>
</tr>
<tr>
<td>+</td>
<td>Adder</td>
<td>0</td>
<td>(Y = A + B + C + D)</td>
</tr>
<tr>
<td>-</td>
<td>Subtractor</td>
<td>0</td>
<td>(Y = A - B - C - D)</td>
</tr>
<tr>
<td>(\times)</td>
<td>Multiplier</td>
<td>0</td>
<td>(Y = A \times B \times C \times D)</td>
</tr>
<tr>
<td>/</td>
<td>Divider</td>
<td>0</td>
<td>(Y = A/B/C/D)</td>
</tr>
</tbody>
</table>

Figure 4 (Continued).
<table>
<thead>
<tr>
<th>BLOCK TYPE</th>
<th>BLOCK FUNCTION</th>
<th>NO. OF PARAMETERS</th>
<th>NOTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>FGEN1</td>
<td>11</td>
<td>P1 = number of pairs. Parameters 2-11 are used in pairs to define the interpolator boundaries.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Y = f(A, P2---P11) where</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X1 = P2  Y1 = P3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X2 = P4  Y2 = P5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X3 = P6  Y3 = P7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X4 = P8  Y4 = P9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X5 = P10 Y5 = P11</td>
</tr>
</tbody>
</table>

| A          | DISPLAY        | 0                 | Display the output value, A, of a block. |

| A B C D    | LOG 10         | 0                 | Computes the common log | Y = Log_{10} [(A*B) + (C*D)] |

Figure 4 (Continued).
<table>
<thead>
<tr>
<th>BLOCK TYPE</th>
<th>BLOCK FUNCTION</th>
<th>NO. OF PARAMETERS</th>
<th>NOTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>A B C D FAN</td>
<td>Computes the tangent function</td>
<td>0</td>
<td>$Y = \tan \left( (A<em>B) + (C</em>D) \right)$</td>
</tr>
<tr>
<td>A B C D SQRT</td>
<td>Computes the square root</td>
<td>0</td>
<td>$Y = \sqrt{(A<em>B) + (C</em>D)}$</td>
</tr>
<tr>
<td>A B C D POWER</td>
<td>Computes a real number raised to a power</td>
<td>1</td>
<td>$Y = \left( (A<em>B) + (C</em>D) \right)^{p1}$</td>
</tr>
<tr>
<td>A B C D LOGE</td>
<td>Computes the natural log</td>
<td>0</td>
<td>$Y = \log_e \left( (A<em>B) + (C</em>D) \right)$</td>
</tr>
</tbody>
</table>

Figure 4 (Continued).
<table>
<thead>
<tr>
<th>BLOCK TYPE</th>
<th>BLOCK FUNCTION</th>
<th>NO. OF PARAMETERS</th>
<th>NOTES</th>
</tr>
</thead>
</table>
| A          | Performs elementary arithmetic functions based on parameter values | 3                | Parameter 1 defines $f(A,B)$  
Parameter 2 defines $G[f(A,B),C]$  
Parameter 3 defines $H[G[f(A,B),C],D]$  
When the Parameter value is:  
1 the operation is addition  
2 the operation is subtraction  
3 the operation is multiplication  
4 the operation is division. |
| B          | ASIN                                    | 0                | $Y = \arcsin \left[(A*1) + (C*1)\right]$                           |
| C          | ACOS                                    | 0                | $Y = \arccos \left[(A*1) + (C*1)\right]$                           |

Figure 4 (Continued).
enclosed in the block. The number of input stubs and parameters are determined by the program from the block's function.

3.4.1.2 **Block Terminus Points**

Line terminus points on the blocks are critical forms of data items in a diagram and are used in subsequent machine calculations. Terminus points or input/output line stubs are generated on the block schematic when it is displayed. The number of input termini is determined by the block's function. A terminus becomes active in a usable part of the diagram only if a line is connected to it.

3.4.1.3 **Block Connection**

Functionally, the analyst need only specify the output of a particular block be connected to one of the input pins of the connecting block. At this point, the IPAA software system will generate the connecting line between the two blocks. A detailed discussion of the method of graphically drawing this connection is contained in Appendix A.

3.4.1.4 **Add Text**

Text may be appended to a block during the construction of a diagram at which time it becomes an integral part of the graphics database. The text serves two purposes; 1) automatically it becomes a significant part of the analysis documentation, and 2) upon subsequent recall of the diagram, it may aid the user in understanding the purpose and function of the diagram. A detailed description of the procedure for adding text is given in Section 3.3.7 in Reference 3.

3.4.1.5 **Modify Parameters**

Figure 4 shows which of the blocks require parameters and their meaning. Those blocks which require parameters have them all initialized to zero. The procedure for modifying parameters is given in Section 3.3.9 in Reference 3. Although parameters could be permanently displayed
in a diagram, it was felt that a picture would become too busy and so lose its effectiveness. The user may quickly and easily check the current value of the parameter by using the methods described in Section 3.3.8 in Reference 3.

3.4.1.6 **Erase Pin Connection**

Any input pin connection may be deleted from a finished diagram or from a partially completed diagram. After the user selects the input pin which he wishes to be deleted, the program proceeds as follows:

1) The designed input pin pointer is checked to determine if it has been used. If not, an error is printed for the analyst.

2) If the pin has been used, the software crosses out the input connection on the screen.

3) The input pin pointer is then flagged as having been deleted but still displayed. (Note: should the analyst decide at a later time to connect another block to this particular input pin, the software will allow the connection; however, it will immediately erase the screen and redisplay that page.)

3.4.1.7 **Erase Network**

Any connection network (where a network consists of all the line connections that eventually connect to a common block output stub) may be deleted from a completed or partially completed diagram. After the user has selected the block from which the output is to be deleted, the program proceeds as follows:

1) The graphics data base is searched to determine which input pin connection pointers are referencing the designated output block.
2) As each pin is determined to have that particular reference, the software then proceeds to perform the functions referenced in Items (2) and (3) in Section 3.4.1.6.

3.4.1.8  **Erase Block**

Any block may be deleted from a completed or partially completed diagram. The user selects the block to be deleted according to the method described in Section 3.3.10 of Reference 3. The program then proceeds as follows:

1) The indicated block is crossed out by the software on the display.

2) All input pointers and the parameter pointer is zeroed.

3) The software then proceeds to perform the functions referenced in Items (1) and (2) in Section 3.4.1.7.

3.4.1.9  **Erase Text**

Text may be erased from a diagram during construction, upon completion or after the diagram has been recalled and the editing is desired. A detailed description of the procedure necessary to accomplish this is contained in Section 3.3.13 in Reference 3. The program proceeds as follows:

1) The block location is determined from the screen coordinates sent to the software by the cursor position of the terminal.

2) The indicated text array at this point is then set to blank characters.

3.4.2  **Diagram Storage and Retrieval**

In order to accomplish the storage or retrieval of a diagram, a transfer of information to or from a disc file is necessary. This
information includes the graphics database and any indicated parameters or text associated with that database.

The IPAA System has a paging capability in which 1-9 pages may comprise a single diagram. The total of nine pages is stored as one disc file.

In keeping with the existing protocol for operating within the Univac 1110 environment, the analyst may assign a 12-character alphanumeric name to a file.

3.4.2.1 Save Diagram

The purpose of the Save Diagram function is to give the analyst the capability of storing diagrams for future use. This storage is accomplished nine pages at a time. The detailed procedure necessary to save a diagram is given in Reference 3, Section 3.3.14. Program interaction is briefly described below:

1) Upon command, the software will request the name of the desired file from the analyst.

2) The file is then created within the Univac 1110 system.

3) The necessary graphics database is then written to that file.

4) The file is then catalogued and is available for later recall by the analyst.

3.4.2.2 Recall Diagram

This function allows the analyst to recall to the screen any diagram he has previously stored. Once a diagram has been recalled, the analyst can either modify the diagram or run the analysis execution function. The procedure necessary to recall a diagram is given in Reference 3, Section 3.3.15. Program interaction is briefly described below:
1) Upon selection of the recall diagram function, the analyst must specify the file name of the diagram to be recalled.

2) The correct file is then located and the information read into the program memory.

3.4.3 Paging

A diagram of the IPAA System may be made up of as many as nine separate display pages. The pages are connected by referring a specially designated output block of one page to a similarly designated input block of another page. This special usage of output and input blocks may be seen in Figure 5. For every diagram created and stored, there will be one data file generated.

The present configuration of IPAA provides for no more than 180 blocks in a complete diagram. When a diagram is continued from one page to another it is necessary to have an output block and an input block as connectors. These blocks are not formally a part of the processing function of the diagram but they are counted as part of the 180 allowable blocks. The inclusion of these blocks was made necessary by the logic of the analysis ordering algorithm as described in Section 3.5.3. Further user-related details of paging are discussed in Section 3.3.1 of Reference 3.

3.4.4 Zooming

In connection with the paging capability a zooming feature has been incorporated into IPAA. Zooming allows the user to display all nine pages of the diagram at once. Because of the size of the blocks in trying to display 180 blocks on a single display, it will be noted that only the block number is displayed. There is no function code and no parameters. There is, however, an additional paging number (absolute value of the parameter for input and output blocks) which is printed to the left or right of input and output blocks in order to easily reference the interpage connections.
3.5 Analysis Execution Functions

An IPAA diagram as constructed by the graphics function will not be in an executable state. Execution of the diagram requires that the order of operating the various functions of a diagram be established. For example, it would be disastrous to run a subroutine before its current inputs were available. There are basically two methods of controlling diagram execution, (1) a syntax compiler and (2) a table-driven compiler.

A syntax compiler (not chosen for IPAA) produces a unique coding sequence in some language syntax, such as FORTRAN. The main program would consist of a series of calls to subroutines that are a part of the main program. The program would be required to establish the order of the calls and assure that proper parameters were available to each called subroutine. These calling statements are built in source format and input to a compiler program for final compilation, linking and loading. The output of this compiler is a unique problem-oriented program. A change to the analysis diagram requires an iteration of compiling, linking and loading. This type of system is not suited to diagrams of the complexity processed by IPAA.

The table-driven compiler as used in IPAA consists of a main program and all the subroutines and functions that might possibly be used, not just those required for the particular diagram to be executed. The preprocessor portion of the analysis execution function generates a table that determines the order of execution of the various mathematical subroutines. The run-time portion of the analysis functions scans the run table and calls the appropriate mathematical functions in the proper order.

3.5.1 Process Control Parameter Modification

To execute the analysis of a diagram, the following process control parameters are required:

Start Time
Stop Time
Time Increment
Diagram
Input Interpolation
The default values for these parameters are given in Figure 6 which is the analysis execution menu. The analyst may modify any of these parameters according to the procedures given in Reference 3, Section 3.4. The purpose of these parameters is discussed below:

Diagram: The name of the diagram may be provided by the analyst or it may be the default value as seen in Figure 6.

Start Time: This time refers to the start time in a time block of a diagram. Default time is 0.0 which may be changed to any time desired by the analyst.

Stop Time: This value refers to the time at which the analyst wishes to end the analysis run. Default stop time is 10.0 and may be changed to any desired time.

Time Increment: This value is the delta time used in the analysis and may be changed to any delta time increment desired.

Input Interpolation Degree: A Lagrange interpolation function is provided in which polynomials up to a ninth degree may be evaluated. The default value is 1 which would provide a linear interpolation. If the analyst desires a polynomial interpolator, then the degree of the polynomial is entered.

Display Ratio: The display blocks entered in a diagram provides the analyst with the means to view immediate results on the CRT screen. If the analyst does not desire to see these immediate results at every delta time period in the analysis, the display ratio may be modified to any desired time increment the analyst wishes to view.
** IPAA PROCESSOR INFORMATION **

1- PROCESSING DESCRIPTION:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2- Diagnosis Name</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3- Start Time</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>4- Stop Time</td>
<td>10.000</td>
<td></td>
</tr>
<tr>
<td>5- Time Increment</td>
<td>0.100</td>
<td></td>
</tr>
<tr>
<td>6- Input Interpolation Degree</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>7- Display Ratio</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>8- Time Option</td>
<td>TIME INPUT PROVIDED</td>
<td></td>
</tr>
</tbody>
</table>

INPUTS:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>9- Input Format</td>
<td>DEFAULT FORMAT (POT)</td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10- Input File No. 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11- Input File No. 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12- Input File No. 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13- Input File No. 4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

OUTPUT

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>14- Output Format</td>
<td>DEFAULT FORMAT (POT)</td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>15- Output File No. 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16- Output File No. 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17- Output File No. 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18- Output File No. 4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20- Save PIF in File</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21- Restore PIF from File</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30- Execute</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 6. Default Execution Menu.
Inputs: IPAA provides for a variety of input formats. In addition to the PUT-type of input file, the system will accept binary, formatted and free-field inputs. These three options allow mixtures of floating point and integer variables. See Section 3.4.9 and Appendix C in Reference 3 for a complete discussion.

Outputs: IPAA provides essentially the same format variety on output as on input with the exception that the free-field option is not allowed. See Section 3.4.9 and Appendix C in Reference 3 for a complete discussion.

3.5.2 IPAA Data Base

A complete functional description of an IPAA processing block is contained in six words (See Figure 7). This includes function code, input pointers, and parameter list pointer. Thus the complete diagram (180 blocks) may be functionally described in an array of length 6x180. This array is called the block table.

The IPAA data base consists of the block table and a parameter array. The length of the parameter array is fixed at 300, which is deemed to be sufficient for the current software system.

3.5.3 Block Ordering

The ordering algorithm is the heart of the analysis execution function. In addition to the block table and parameter array, the ordering algorithm makes use of a software table in the IPAA system. This table consists of a flag for each available function denoting whether or not it has an output available initially. The current implementation flags only delay and integrator blocks as having outputs initially available. The procedure will now be discussed in a step-by-step manner.

Step 1. Cycle through the block table and remove the off-page connections. This is performed by redirecting the pointers to an input page connector to the block which is input to the output page connectors.
In memory, each block is represented by a 6-word table entry:

\[ \text{KFUN}(I, J, K) \] where 
- \( I = 1 \) to 6 (the table entries) 
- \( J = 1 \) to 20 (the block number on the page) 
- \( K = 1 \) to 9 (the page number)

Special codes in these table entries are shown below.

<table>
<thead>
<tr>
<th>FUNCTION CODE</th>
<th>PIN A</th>
<th>PIN B</th>
<th>PIN C</th>
<th>PIN D</th>
<th>PARAMETER POINTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>A positive integer M where M varies from 1 to N, and is the index in the table of defined function codes (MATCH). 0 if function is not defined. A negative integer indicates that the block has been used but the function has not yet been defined.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>An integer entry containing the block number (1 to 180) from which the input to this pin comes.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>For all blocks: Contains an integer pointing to the first entry (for this block) in the parameter table.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>For Input/Output block parameters: A positive integer indicates the number of the file (cross-referenced in the Processor Information File - PIF) to be used in I/O operations. A negative integer indicates an off-page connection. I/O blocks with this same negative number are logically connected during execution.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 7. BLOCK FUNCTIONAL DESCRIPTOR
Step 2. Cycle through the block table and put into the ordering table pointers to those blocks that have outputs available and which may be ordered immediately. This set consists of all constant, input, and time blocks. Then set, in tables parallel to the ordering table, a flag indicating that the block is in the ordering table and another flag indicating the block has its output available.

Step 3. Cycle through the block table and set the appropriate output available flag for each function. This information is derived from the software table.

Step 4. Set the block table pointer to the first entry in the table.

Step 5. Has this entry been ordered? If so, go to Step 8. If not go to Step 6.

Step 6. Do all the blocks referenced by this block's input pointers have outputs available? If yes, go to Step 7. If not go to Step 8.

Step 7. Order the block and flag its output available.

Step 8. Increment the block table pointer. If the table has been exhausted go to Step 9. If not, go to Step 5.

Step 9. Are there any blocks which haven't been ordered? If not, the ordering process is finished. If there are blocks remaining to be ordered and at least one was ordered on the last pass through the table, go to Step 4. If none were ordered on the last pass and there are still blocks remaining to be ordered, then terminate process with error flag.
3.5.4 Interpolation

Provision was made within the Itas concept to handle non-uniformly sampled telemetry formats. A Lagrangian interpolating polynomial algorithm with degree \( n \) is provided to handle such non-uniform data. The degree of interpolation (linear, quadratic, etc.) is under analyst control.

The general form of Lagrange's interpolating polynomial is given by

\[
P_n(x) = \sum_{i=0}^{n} L_i(x) f(x_i)
\]

where

\[
L_i(x) = \prod_{\substack{j=0 \atop j \neq i}}^{n} \frac{(x-x_j)}{(x_i-x_j)}, \quad i=0,1,...,n
\]

Each functional value, \( f(x_i) \), included in the polynomial fit is multiplied by \( L_i \), an \( n^{th} \) degree polynomial in \( x \) (since there are \( n \) factors \( x-x_j \)). For a more detailed mathematical discussion of the Lagrange interpolation polynomial, the reader is referred to References 1 and 2.
4.0 COMPARATIVE ALGORITHMS

Two algorithms were chosen by the customer for comparison of conventional programming techniques versus an IPAA approach.

Although it is felt that for certain types of processing the IPAA approach may be faster and somewhat less error-prone than the conventional FORTRAN program, it is difficult at best to attempt a comparison of coding/debug time (conventional) versus diagram building (IPAA). Such a comparison would apply only to the individuals involved and any conclusions would not necessarily apply to other users. For this reason, the comparisons in this section will be concerned with capability of the different approaches and the computer system knowledge required for implementation.

Common to both algorithm comparisons is the prerequisite that the conventional approach requires a knowledge of the FORTRAN programming language and a programmer's familiarity with the Univac Operating System and file-handling abilities as opposed to the superficial computer-system knowledge required by the IPAA-approach. This means that some analysts may not be able to make use of the computer system except through IPAA due to a lack of training and knowledge in the software disciplines.

4.1 Algorithm Number One

This algorithm takes inertially fixed orthogonal sensed velocities through an Euler angle transformation (as defined by the Gimbal angles and order) to obtain orthogonal sensed velocity in the vehicle body coordinate frame. The RSS of the orthogonal vehicle velocities are computed to obtain total velocity and the orientation of this velocity with respect to the vehicle body is calculated. In addition, differences of the input sensed velocities are used to calculate the orthogonal sensed acceleration in the vehicle body coordinate frame, and the total acceleration and orientation with respect to the vehicle body. (See program listing in Appendix L.)
Algorithm No. 1 - Inputs and Mathematical Formulation

The input variables defined for algorithm one are as follows:

- $T_i$: Time
- $\text{IG}_i$: Inner Gimbal Angle
- $\text{MG}_i$: Middle Gimbal Angle
- $\text{OG}_i$: Outer Gimbal Angle
- $\text{VV}_i$: Vertical Sensed Velocity
- $\text{HV}_i$: Horizontal Sensed Velocity
- $\text{LV}_i$: Lateral Sensed Velocity

Figure 8 illustrates the test case used in the comparison. Let $\dot{X}, \dot{Y}, \dot{Z} = \dot{X}, \dot{Y}, \dot{Z}$ velocities

- $\text{MAG} = \text{Total Velocity Magnitude}$
- $\phi = \text{Angle of total velocity relative to vertical longitudinal plane.}$
- $\gamma = \text{Angle of total velocity relative to horizontal plane.}$

Then

\[
\dot{X}_i = \left[ \text{HV}_i \cdot \cos(\text{IG}_i) - \text{LV}_i \cdot \sin(\text{IG}_i) \right] \cdot \cos(\text{OG}_i)
+ \left[ \text{HV}_i \cdot \sin(\text{IG}_i) + \text{LV}_i \cdot \cos(\text{IG}_i) \right] \cdot \sin(\text{OG}_i)
\]

\[
\dot{Y}_i = \left[ \text{HV}_i \cdot \sin(\text{IG}_i) + \text{LV}_i \cdot \cos(\text{IG}_i) \right] \cdot \cos(\text{MG}_i)
- \text{VV}_i \cdot \sin(\text{MG}_i)
\]

\[
\dot{Z}_i = \left[ \text{HV}_i \cdot \sin(\text{IG}_i) + \text{LV}_i \cdot \cos(\text{IG}_i) \right] \cdot \sin(\text{MG}_i)
+ \text{VV}_i \cdot \cos(\text{MG}_i) \cdot \cos(\text{OG}_i)
\]

\[
\text{MAG}_i = \sqrt{\dot{X}_i^2 + \dot{Y}_i^2 + \dot{Z}_i^2}
\]
Figure 8. Algorithm #1 Input Data
\[
\dot{\psi}_i = \tan^{-1}\left(\frac{-\dot{y}_i}{\dot{x}_i}\right), \text{ in proper quadrant}
\]

\[
\dot{\gamma} = \tan^{-1}\left(\frac{\dot{x}_i^2 + \dot{y}_i^2}{\dot{z}_i}\right), \text{ in proper quadrant}
\]

Replacement with the following values will yield the acceleration components when used in the above formulae:

let \( \text{FT} = \text{Frame Time} \)

\[
SC_i = (T_i - T_{i-1}) \cdot FT
\]

then \( \dot{V}_V_i = (\dot{V}_V_i - \dot{V}_{V_{i-1}}) / SC_i \)

\( \dot{H}_V_i = (\dot{H}_V_i - \dot{H}_{V_{i-1}}) / SC_i \)

\( \dot{L}_V_i = (\dot{L}_V_i - \dot{L}_{V_{i-1}}) / SC_i \)

4.1.2 Algorithm No. 1 - Comparison

- **Input** - Both approaches utilize the same data input for the test case (see Figure B). Inherent in the conventional approach, however, is a capability of selectively ignoring certain input values (see Appendix B listing) which have been flagged as bad measurements. In order to duplicate this particular capability in the IPAA approach, the data file would have to have these records deleted.

- **Implementation** - A simple replacement of three velocity variables in the conventional approach allow the same basic algorithm to be utilized for acceleration computations. In addition, subroutine QUAD (see Appendix B) is utilized for four of the 12 computed values.

The limitation on the number of blocks available in a diagram (180) forces the IPAA user to build two separate diagrams (see Appendix B). Also, the functional blocks representing the QUAD subroutine must be duplicated each time a conventional approach would invoke it.
• **Execution** - Scaling variables which may be specified during execution in the conventional approach entail a diagram modification of constant blocks prior to execution in the IPAA approach (e.g., Frame Time). Differences in execution time between the two approaches is not noticeable in the Univac time-sharing environment.

• **Output** - The output of both approaches for the test case are essentially the same (see Figures 9 and 10). Note, however, that due to the limited format associated with DISP blocks in the IPAA System, two of the output variables (X,Y) had to be scaled in order to print meaningful values. The computations for time 1.000 are not performed in the conventional approach, but must be in the IPAA approach in order that succeeding acceleration values be correct.

### 4.2 Algorithm Number Two

This algorithm is used in the analysis of rocket engine performance characteristics. Specifically, the algorithm establishes the relationship between the engine (overall) propellant mixture ratio and the gas generator (for turbopump power) propellant mixture ratio. The entire amount of one of the propellants, fuel or oxidizer, is supplied to the gas generator. The gas-generator portion of the remaining propellant constituent is also calculated.

#### 4.2.1 Algorithm No. 2 - Mathematical Formulation

Let

\[ T_i = \text{TIME} \]
\[ \text{EWF} = 1/(T_i+1) \]
\[ \text{EWO} = T_i \cdot \text{EWF} \]
<table>
<thead>
<tr>
<th>OUTPUT VARIABLE (COLUMN) IDENTIFICATION:</th>
<th>E-ANGLE OF TOTAL VELOCITY VECTOR RELATIVE TO (1+ LEFT OF) VEHICLE VERTICAL LONGITUDINAL PLANE (PMI)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-PSEUDO TIME</td>
<td>7-ANGLE OF TOTAL VELOCITY VECTOR RELATIVE TO VEHICLE HORIZONTAL PLANE (GAMMA)</td>
</tr>
<tr>
<td>2.3.4-VEHICLE XDOT+ YDOT+ ZDOT</td>
<td>8-13 SAME AS 2-7 EXCEPT FOR ACCELERATION</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VELOCITY SCALE FACTOR IS</th>
<th>1.000 CTS/METER/SEC</th>
<th>FRAME TIME IS 6800000 SEC</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.000 -673.9</td>
<td>-41.79</td>
<td>1367.1</td>
</tr>
<tr>
<td>3.000 -933.7</td>
<td>-49.61</td>
<td>1364.1</td>
</tr>
<tr>
<td>4.000 -991.5</td>
<td>-53.29</td>
<td>1367.1</td>
</tr>
<tr>
<td>5.000 -1050</td>
<td>-47.16</td>
<td>1367.1</td>
</tr>
<tr>
<td>6.000 -1116</td>
<td>-47.51</td>
<td>1367.1</td>
</tr>
<tr>
<td>7.000 -1175</td>
<td>-55.60</td>
<td>1364.1</td>
</tr>
<tr>
<td>8.000 -1236</td>
<td>-51.42</td>
<td>1364.1</td>
</tr>
<tr>
<td>9.000 -1300</td>
<td>-55.41</td>
<td>1367.1</td>
</tr>
<tr>
<td>10.000 -1363</td>
<td>-49.61</td>
<td>1367.1</td>
</tr>
<tr>
<td>11.000 -1425</td>
<td>-57.66</td>
<td>1364.1</td>
</tr>
<tr>
<td>12.000 -1488</td>
<td>-51.13</td>
<td>1367.1</td>
</tr>
<tr>
<td>13.000 -1552</td>
<td>-62.47</td>
<td>1367.1</td>
</tr>
<tr>
<td>14.000 -1615</td>
<td>-58.40</td>
<td>1367.1</td>
</tr>
<tr>
<td>15.000 -1676</td>
<td>-127.5</td>
<td>1370.1</td>
</tr>
<tr>
<td>16.000 -1748</td>
<td>-198.3</td>
<td>1427.1</td>
</tr>
<tr>
<td>17.000 -1808</td>
<td>-223.1</td>
<td>1491.1</td>
</tr>
<tr>
<td>18.000 -1872</td>
<td>-222.0</td>
<td>1496.1</td>
</tr>
<tr>
<td>19.000 -1950</td>
<td>-215.6</td>
<td>1462.1</td>
</tr>
<tr>
<td>20.000 -2037</td>
<td>-211.7</td>
<td>1446.1</td>
</tr>
<tr>
<td>21.000 -2100</td>
<td>-251.1</td>
<td>1366.1</td>
</tr>
<tr>
<td>22.000 -1979</td>
<td>-105.6</td>
<td>1307.1</td>
</tr>
<tr>
<td>23.000 -1728</td>
<td>-1552.0</td>
<td>1229.1</td>
</tr>
<tr>
<td>24.000 -1516</td>
<td>-1553.0</td>
<td>1224.1</td>
</tr>
<tr>
<td>25.000 -1176</td>
<td>-212.8</td>
<td>1379.1</td>
</tr>
<tr>
<td>26.000 -699.6</td>
<td>-220.4</td>
<td>1462.1</td>
</tr>
<tr>
<td>27.000 -551.3</td>
<td>-222.2</td>
<td>1572.1</td>
</tr>
<tr>
<td>28.000 -729.7</td>
<td>-232.3</td>
<td>1514.1</td>
</tr>
<tr>
<td>29.000 -1992</td>
<td>-230.0</td>
<td>1393.1</td>
</tr>
<tr>
<td>30.000 -884.9</td>
<td>-275.1</td>
<td>1467.1</td>
</tr>
<tr>
<td>31.000 -958.4</td>
<td>-279.0</td>
<td>1465.1</td>
</tr>
<tr>
<td>32.000 -1032</td>
<td>-282.5</td>
<td>1468.1</td>
</tr>
<tr>
<td>33.000 -1106</td>
<td>-229.5</td>
<td>2839.1</td>
</tr>
<tr>
<td>34.000 -1179</td>
<td>-228.7</td>
<td>2872.1</td>
</tr>
<tr>
<td>35.000 -1252</td>
<td>-239.3</td>
<td>1392.1</td>
</tr>
<tr>
<td>36.000 -1326</td>
<td>-239.4</td>
<td>1490.1</td>
</tr>
<tr>
<td>37.000 -1393</td>
<td>-239.4</td>
<td>1497.1</td>
</tr>
<tr>
<td>38.000 -200</td>
<td>-239.3</td>
<td>3111.1</td>
</tr>
<tr>
<td>39.000 -1552</td>
<td>-230.7</td>
<td>1493.1</td>
</tr>
<tr>
<td>40.000 -1625</td>
<td>-230.7</td>
<td>1497.1</td>
</tr>
<tr>
<td>41.000 -1697</td>
<td>-234.4</td>
<td>1503.1</td>
</tr>
</tbody>
</table>

Figure 9. Algorithm #1 Output (Conventional)
**Algorithm = 1 Output (IPAA)**

<table>
<thead>
<tr>
<th>TIME</th>
<th>9/3</th>
<th>9/10</th>
<th>9/12</th>
<th>9/14</th>
<th>9/17</th>
<th>9/19</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-81 263</td>
<td>-3 947</td>
<td>1363 276</td>
<td>1587 590</td>
<td>177 219</td>
<td>59 178</td>
</tr>
<tr>
<td>0</td>
<td>-87391</td>
<td>-4 179</td>
<td>1364 443</td>
<td>1517 486</td>
<td>177 262</td>
<td>67 255</td>
</tr>
<tr>
<td>0</td>
<td>-93367</td>
<td>-4 661</td>
<td>1365 514</td>
<td>1653 491</td>
<td>177 325</td>
<td>59 586</td>
</tr>
<tr>
<td>0</td>
<td>-100145</td>
<td>-4 329</td>
<td>1366 544</td>
<td>1686 639</td>
<td>177 400</td>
<td>53 999</td>
</tr>
<tr>
<td>0</td>
<td>-104 963</td>
<td>-4 316</td>
<td>1367 430</td>
<td>1724 977</td>
<td>177 646</td>
<td>62 447</td>
</tr>
<tr>
<td>0</td>
<td>-111 813</td>
<td>-4 751</td>
<td>1357 219</td>
<td>1744 766</td>
<td>177 562</td>
<td>58 781</td>
</tr>
<tr>
<td>0</td>
<td>-117 497</td>
<td>-4 540</td>
<td>1371 211</td>
<td>1806 179</td>
<td>177 777</td>
<td>49 385</td>
</tr>
<tr>
<td>0</td>
<td>-123 553</td>
<td>-4 182</td>
<td>1369 299</td>
<td>1844 704</td>
<td>178 061</td>
<td>47 603</td>
</tr>
<tr>
<td>0</td>
<td>-129 970</td>
<td>-4 561</td>
<td>1367 464</td>
<td>1827 688</td>
<td>178 990</td>
<td>46 437</td>
</tr>
<tr>
<td>0</td>
<td>-136 315</td>
<td>-4 981</td>
<td>1372 614</td>
<td>1935 355</td>
<td>178 967</td>
<td>45 386</td>
</tr>
<tr>
<td>0</td>
<td>-142 543</td>
<td>-5 256</td>
<td>1364 161</td>
<td>1973 713</td>
<td>178 884</td>
<td>43 722</td>
</tr>
<tr>
<td>0</td>
<td>-148 004</td>
<td>-5 312</td>
<td>1370 597</td>
<td>2023 779</td>
<td>177 956</td>
<td>42 609</td>
</tr>
<tr>
<td>0</td>
<td>-155 222</td>
<td>-5 228</td>
<td>1376 505</td>
<td>2077 645</td>
<td>176 491</td>
<td>41 460</td>
</tr>
<tr>
<td>0</td>
<td>-162 682</td>
<td>-5 610</td>
<td>1361 097</td>
<td>2151 953</td>
<td>178 085</td>
<td>39 695</td>
</tr>
<tr>
<td>0</td>
<td>-171 586</td>
<td>-12 752</td>
<td>1369 832</td>
<td>2150 956</td>
<td>175 750</td>
<td>38 519</td>
</tr>
<tr>
<td>0</td>
<td>-174 823</td>
<td>-19 832</td>
<td>1427 245</td>
<td>2226 639</td>
<td>173 528</td>
<td>38 409</td>
</tr>
<tr>
<td>0</td>
<td>-180 340</td>
<td>-22 347</td>
<td>1447 878</td>
<td>2318 488</td>
<td>172 949</td>
<td>38 417</td>
</tr>
<tr>
<td>0</td>
<td>-186 339</td>
<td>-22 204</td>
<td>1444 310</td>
<td>2377 118</td>
<td>173 241</td>
<td>37 476</td>
</tr>
<tr>
<td>0</td>
<td>-194 592</td>
<td>-21 557</td>
<td>1442 477</td>
<td>2446 898</td>
<td>173 693</td>
<td>38 763</td>
</tr>
<tr>
<td>0</td>
<td>-200 161</td>
<td>-33 156</td>
<td>1457 538</td>
<td>2493 036</td>
<td>176 624</td>
<td>36 415</td>
</tr>
<tr>
<td>0</td>
<td>-203 780</td>
<td>- 62 359</td>
<td>1395 394</td>
<td>2547 273</td>
<td>162 885</td>
<td>33 216</td>
</tr>
<tr>
<td>0</td>
<td>-197 927</td>
<td>-194 965</td>
<td>1397 865</td>
<td>2593 824</td>
<td>162 665</td>
<td>30 260</td>
</tr>
<tr>
<td>0</td>
<td>-172 766</td>
<td>-155 199</td>
<td>1328 811</td>
<td>2587 817</td>
<td>158 468</td>
<td>27 869</td>
</tr>
<tr>
<td>0</td>
<td>-131 567</td>
<td>-195 263</td>
<td>1241 603</td>
<td>2561 937</td>
<td>153 971</td>
<td>27 883</td>
</tr>
<tr>
<td>0</td>
<td>-131 567</td>
<td>-195 263</td>
<td>1241 603</td>
<td>2561 937</td>
<td>153 971</td>
<td>27 883</td>
</tr>
<tr>
<td>0</td>
<td>-120 459</td>
<td>-166 781</td>
<td>2527 281</td>
<td>112 135</td>
<td>23 627</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>-95 835</td>
<td>-228 643</td>
<td>2378 630</td>
<td>113 705</td>
<td>28 659</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>-183 196</td>
<td>-228 681</td>
<td>2378 417</td>
<td>114 308</td>
<td>38 344</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>-100 412</td>
<td>-228 490</td>
<td>2378 831</td>
<td>112 189</td>
<td>24 247</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>-117 875</td>
<td>-228 336</td>
<td>2378 630</td>
<td>117 304</td>
<td>30 153</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>-125 217</td>
<td>-228 982</td>
<td>2378 433</td>
<td>118 839</td>
<td>29 771</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>-132 577</td>
<td>-228 411</td>
<td>2378 039</td>
<td>120 424</td>
<td>29 358</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>-139 301</td>
<td>-230 007</td>
<td>2378 416</td>
<td>121 281</td>
<td>29 644</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>-147 865</td>
<td>-230 438</td>
<td>2378 763</td>
<td>3118 740</td>
<td>122 582</td>
<td>28 531</td>
</tr>
<tr>
<td>0</td>
<td>-153 299</td>
<td>-230 719</td>
<td>2379 088</td>
<td>3104 213</td>
<td>123 936</td>
<td>28 231</td>
</tr>
<tr>
<td>0</td>
<td>-162 639</td>
<td>-230 652</td>
<td>2368 881</td>
<td>3193 644</td>
<td>128 190</td>
<td>27 847</td>
</tr>
</tbody>
</table>

**Figure 10. Algorithm = 1 Output (IPAA)**
### IPAA Analysis Routine

**Algorithm Number One Analysis-Acceleration Section**

**Execution Results**

<table>
<thead>
<tr>
<th>TIME</th>
<th>902</th>
<th>905</th>
<th>912</th>
<th>914</th>
<th>917</th>
<th>919</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.000</td>
<td>111111111</td>
<td>-78.935</td>
<td>2726.650</td>
<td>3175.181</td>
<td>177.219</td>
<td>50.172</td>
</tr>
<tr>
<td>2.000</td>
<td>-122.463</td>
<td>-721</td>
<td>-5.569</td>
<td>122.690</td>
<td>179.862</td>
<td>-2.575</td>
</tr>
<tr>
<td>3.000</td>
<td>-125.960</td>
<td>299</td>
<td>4.414</td>
<td>126.624</td>
<td>179.884</td>
<td>1.125</td>
</tr>
<tr>
<td>4.000</td>
<td>-121.597</td>
<td>-1.864</td>
<td>-5.380</td>
<td>121.721</td>
<td>179.899</td>
<td>-2.533</td>
</tr>
<tr>
<td>5.000</td>
<td>-128.318</td>
<td>282</td>
<td>2.860</td>
<td>122.762</td>
<td>179.877</td>
<td>1.344</td>
</tr>
<tr>
<td>6.000</td>
<td>-124.960</td>
<td>-2.638</td>
<td>9.668</td>
<td>126.840</td>
<td>179.866</td>
<td>1.810</td>
</tr>
<tr>
<td>7.000</td>
<td>-125.467</td>
<td>282</td>
<td>3.395</td>
<td>125.499</td>
<td>179.899</td>
<td>1.477</td>
</tr>
<tr>
<td>8.000</td>
<td>-129.864</td>
<td>431</td>
<td>-3.364</td>
<td>129.851</td>
<td>179.819</td>
<td>1.595</td>
</tr>
<tr>
<td>9.000</td>
<td>-128.813</td>
<td>-1.74</td>
<td>-4.006</td>
<td>128.835</td>
<td>179.823</td>
<td>-1.781</td>
</tr>
<tr>
<td>10.000</td>
<td>-127.413</td>
<td>-7.78</td>
<td>10.638</td>
<td>127.875</td>
<td>179.659</td>
<td>4.862</td>
</tr>
<tr>
<td>11.000</td>
<td>-119.205</td>
<td>-7.11</td>
<td>-12.856</td>
<td>119.812</td>
<td>179.658</td>
<td>-8.764</td>
</tr>
<tr>
<td>12.000</td>
<td>-125.266</td>
<td>-9.24</td>
<td>-12.872</td>
<td>125.260</td>
<td>179.578</td>
<td>5.867</td>
</tr>
<tr>
<td>13.000</td>
<td>-125.690</td>
<td>-144.013</td>
<td>10.468</td>
<td>102.754</td>
<td>131.562</td>
<td>3.113</td>
</tr>
<tr>
<td>15.000</td>
<td>-134.782</td>
<td>-9.154</td>
<td>-11.643</td>
<td>131.864</td>
<td>175.694</td>
<td>-6.519</td>
</tr>
<tr>
<td>17.000</td>
<td>-14.311</td>
<td>-12.314</td>
<td>-6.663</td>
<td>144.489</td>
<td>175.186</td>
<td>-2.627</td>
</tr>
<tr>
<td>18.000</td>
<td>-144.456</td>
<td>-7.832</td>
<td>2.431</td>
<td>144.718</td>
<td>176.356</td>
<td>-9.652</td>
</tr>
<tr>
<td>19.000</td>
<td>-153.861</td>
<td>12.943</td>
<td>32.214</td>
<td>156.949</td>
<td>175.156</td>
<td>11.644</td>
</tr>
<tr>
<td>20.000</td>
<td>-134.965</td>
<td>17.195</td>
<td>-5.666</td>
<td>136.558</td>
<td>172.777</td>
<td>-2.343</td>
</tr>
<tr>
<td>21.000</td>
<td>-128.575</td>
<td>13.265</td>
<td>-7.74</td>
<td>129.929</td>
<td>174.118</td>
<td>-3.17</td>
</tr>
<tr>
<td>22.000</td>
<td>-118.945</td>
<td>-14.479</td>
<td>9.806</td>
<td>119.436</td>
<td>177.559</td>
<td>4.709</td>
</tr>
<tr>
<td>23.000</td>
<td>-91.213</td>
<td>-5.789</td>
<td>7.185</td>
<td>91.870</td>
<td>176.368</td>
<td>4.495</td>
</tr>
<tr>
<td>25.000</td>
<td>-127.113</td>
<td>-42.276</td>
<td>-2.180</td>
<td>133.059</td>
<td>161.684</td>
<td>-5.623</td>
</tr>
<tr>
<td>26.000</td>
<td>-137.821</td>
<td>-46.609</td>
<td>10.664</td>
<td>144.156</td>
<td>183.471</td>
<td>4.242</td>
</tr>
<tr>
<td>27.000</td>
<td>-146.119</td>
<td>-37.128</td>
<td>13.981</td>
<td>151.497</td>
<td>185.746</td>
<td>5.298</td>
</tr>
<tr>
<td>28.000</td>
<td>-144.962</td>
<td>-19.946</td>
<td>-5.296</td>
<td>146.962</td>
<td>173.723</td>
<td>-2.879</td>
</tr>
<tr>
<td>29.000</td>
<td>-147.879</td>
<td>-1.771</td>
<td>14.779</td>
<td>147.475</td>
<td>177.514</td>
<td>5.833</td>
</tr>
<tr>
<td>30.000</td>
<td>-146.456</td>
<td>-7.110</td>
<td>6.262</td>
<td>146.762</td>
<td>177.221</td>
<td>2.445</td>
</tr>
<tr>
<td>31.000</td>
<td>-146.437</td>
<td>-4.424</td>
<td>-8.627</td>
<td>146.728</td>
<td>178.272</td>
<td>-1.685</td>
</tr>
<tr>
<td>32.000</td>
<td>-149.068</td>
<td>-8.745</td>
<td>14.648</td>
<td>149.066</td>
<td>178.457</td>
<td>5.840</td>
</tr>
<tr>
<td>33.000</td>
<td>-147.492</td>
<td>-7.254</td>
<td>15.145</td>
<td>147.842</td>
<td>177.192</td>
<td>5.840</td>
</tr>
<tr>
<td>34.000</td>
<td>-146.670</td>
<td>-8.826</td>
<td>4.882</td>
<td>147.617</td>
<td>176.556</td>
<td>1.995</td>
</tr>
<tr>
<td>35.000</td>
<td>-146.235</td>
<td>-11.314</td>
<td>5.710</td>
<td>147.262</td>
<td>175.856</td>
<td>-5.262</td>
</tr>
<tr>
<td>36.000</td>
<td>-146.923</td>
<td>-9.842</td>
<td>5.910</td>
<td>147.692</td>
<td>178.582</td>
<td>2.383</td>
</tr>
<tr>
<td>37.000</td>
<td>-146.373</td>
<td>-6.499</td>
<td>5.997</td>
<td>146.548</td>
<td>177.456</td>
<td>2.346</td>
</tr>
<tr>
<td>38.000</td>
<td>-147.182</td>
<td>-3.513</td>
<td>-3.213</td>
<td>147.431</td>
<td>178.947</td>
<td>-1.240</td>
</tr>
<tr>
<td>39.000</td>
<td>-147.871</td>
<td>-11.532</td>
<td>14.871</td>
<td>147.871</td>
<td>178.582</td>
<td>5.718</td>
</tr>
<tr>
<td>40.000</td>
<td>-146.000</td>
<td>8.375</td>
<td>7.370</td>
<td>146.290</td>
<td>178.989</td>
<td>2.889</td>
</tr>
<tr>
<td>41.000</td>
<td>-148.833</td>
<td>-15.472</td>
<td>6.037</td>
<td>149.807</td>
<td>174.065</td>
<td>2.318</td>
</tr>
</tbody>
</table>

**End of Analysis**

Figure 10. Algorithm #1 Output (IPAA) (Continued)
PBWF = .75-EWO
OPBMR = EWO/PBWF
PCTPBF = PBWF/EWF
PBWO = .75-EWF
FPBMR = PBWO/EWF
PCTPBO = PBWO/EWO
RPBMR = 1/FPBMR

4.2.2 Algorithm No. 2 - Comparison

• **Input** - No input required for this algorithm.

• **Implementation** - The implementation of this algorithm is straightforward whether the conventional or IPAA approach is taken.

  This particular algorithm, however, had the approaches implemented in a parallel fashion. The one feature that stood out was the ability of an analyst unfamiliar with the Operating System to readily implement the IPAA approach.

• **Output** - The conventional output has the advantage of providing a column heading to identify the variable on the printout (see Figure 11). The IPAA output has the feature of isolating when and where (block number) an error occurred (see Figure 12).
IPAA ANALYSIS ROUTINE
EXECUTION RESULTS

<table>
<thead>
<tr>
<th>TIME</th>
<th>EMMR</th>
<th>QPBMR</th>
<th>PCIPBF</th>
<th>RPBM</th>
<th>FPBM</th>
<th>PCIPBO</th>
</tr>
</thead>
<tbody>
<tr>
<td>.000</td>
<td>.000</td>
<td>.750</td>
<td>-4.000</td>
<td>-.250</td>
<td>.000</td>
<td></td>
</tr>
<tr>
<td>.100</td>
<td>.138</td>
<td>.725</td>
<td>-.574</td>
<td>-.175</td>
<td>-1.750</td>
<td></td>
</tr>
<tr>
<td>.200</td>
<td>.256</td>
<td>.700</td>
<td>-10.000</td>
<td>-.100</td>
<td>-.500</td>
<td></td>
</tr>
<tr>
<td>.300</td>
<td>.444</td>
<td>.675</td>
<td>-40.000</td>
<td>-.025</td>
<td>-.083</td>
<td></td>
</tr>
<tr>
<td>.400</td>
<td>.615</td>
<td>.650</td>
<td>20.000</td>
<td>.050</td>
<td>.125</td>
<td></td>
</tr>
<tr>
<td>.500</td>
<td>.800</td>
<td>.625</td>
<td>8.000</td>
<td>.125</td>
<td>.250</td>
<td></td>
</tr>
<tr>
<td>.600</td>
<td>1.000</td>
<td>.600</td>
<td>5.000</td>
<td>.200</td>
<td>.333</td>
<td></td>
</tr>
<tr>
<td>.700</td>
<td>1.217</td>
<td>.575</td>
<td>3.636</td>
<td>.275</td>
<td>.393</td>
<td></td>
</tr>
<tr>
<td>.800</td>
<td>1.455</td>
<td>.550</td>
<td>2.857</td>
<td>.350</td>
<td>.438</td>
<td></td>
</tr>
<tr>
<td>.900</td>
<td>1.714</td>
<td>.525</td>
<td>2.353</td>
<td>.425</td>
<td>.472</td>
<td></td>
</tr>
<tr>
<td>1.000</td>
<td>2.000</td>
<td>.500</td>
<td>2.000</td>
<td>.500</td>
<td>.500</td>
<td></td>
</tr>
<tr>
<td>1.100</td>
<td>2.316</td>
<td>.475</td>
<td>1.739</td>
<td>.575</td>
<td>.523</td>
<td></td>
</tr>
<tr>
<td>1.200</td>
<td>2.667</td>
<td>.450</td>
<td>1.538</td>
<td>.650</td>
<td>.542</td>
<td></td>
</tr>
<tr>
<td>1.300</td>
<td>3.059</td>
<td>.425</td>
<td>1.379</td>
<td>.725</td>
<td>.558</td>
<td></td>
</tr>
<tr>
<td>1.400</td>
<td>3.500</td>
<td>.400</td>
<td>1.250</td>
<td>.800</td>
<td>.571</td>
<td></td>
</tr>
<tr>
<td>1.500</td>
<td>4.000</td>
<td>.375</td>
<td>1.143</td>
<td>.875</td>
<td>.583</td>
<td></td>
</tr>
<tr>
<td>1.600</td>
<td>4.571</td>
<td>.350</td>
<td>1.053</td>
<td>.950</td>
<td>.594</td>
<td></td>
</tr>
<tr>
<td>1.700</td>
<td>5.231</td>
<td>.325</td>
<td>.976</td>
<td>1.025</td>
<td>.603</td>
<td></td>
</tr>
<tr>
<td>1.800</td>
<td>6.000</td>
<td>.300</td>
<td>.909</td>
<td>1.100</td>
<td>.611</td>
<td></td>
</tr>
<tr>
<td>1.900</td>
<td>6.909</td>
<td>.275</td>
<td>.851</td>
<td>1.175</td>
<td>.618</td>
<td></td>
</tr>
<tr>
<td>2.000</td>
<td>8.000</td>
<td>.250</td>
<td>.800</td>
<td>1.250</td>
<td>.625</td>
<td></td>
</tr>
<tr>
<td>2.100</td>
<td>9.333</td>
<td>.225</td>
<td>.755</td>
<td>1.325</td>
<td>.631</td>
<td></td>
</tr>
<tr>
<td>2.200</td>
<td>11.000</td>
<td>.200</td>
<td>.714</td>
<td>1.400</td>
<td>.636</td>
<td></td>
</tr>
<tr>
<td>2.300</td>
<td>13.143</td>
<td>.175</td>
<td>.678</td>
<td>1.475</td>
<td>.641</td>
<td></td>
</tr>
<tr>
<td>2.400</td>
<td>16.000</td>
<td>.150</td>
<td>.645</td>
<td>1.550</td>
<td>.646</td>
<td></td>
</tr>
<tr>
<td>2.500</td>
<td>20.000</td>
<td>.125</td>
<td>.615</td>
<td>1.625</td>
<td>.650</td>
<td></td>
</tr>
<tr>
<td>2.600</td>
<td>26.000</td>
<td>.100</td>
<td>.588</td>
<td>1.700</td>
<td>.654</td>
<td></td>
</tr>
<tr>
<td>2.700</td>
<td>36.000</td>
<td>.075</td>
<td>.563</td>
<td>1.775</td>
<td>.657</td>
<td></td>
</tr>
<tr>
<td>2.800</td>
<td>56.000</td>
<td>.050</td>
<td>.541</td>
<td>1.850</td>
<td>.661</td>
<td></td>
</tr>
<tr>
<td>2.900</td>
<td>116.000</td>
<td>.025</td>
<td>.519</td>
<td>1.925</td>
<td>.664</td>
<td></td>
</tr>
<tr>
<td>3.000</td>
<td>******</td>
<td>.000</td>
<td>.500</td>
<td>2.000</td>
<td>.667</td>
<td></td>
</tr>
</tbody>
</table>

Figure 12. Algorithm #2 Output (IPAA)
5.0 CONCLUSIONS

A sophisticated interactive analysis aid has been successfully installed on the Univac 1110. This aid is capable of significantly enhancing the analyst's response by provision of a highly interactive menu for (a) modeling new analysis problems as diagrams, (b) executing the analysis diagrams, and (c) evaluating the execution results.

Analysis diagrams can be created, edited, modified, stored, retrieved, annotated, and executed at the analyst's graphics station. The diagrams themselves are mathematical in nature to provide the widest flexibility and applicability. Available special features include handling multi-page diagrams, parallel analysis processes, nonlinear analysis and automated input and output of sampled data.

The recommended development of macros within the software system will allow system modeling at a modular or functional level as opposed to the current primitive level.
6.C RECOMMENDATIONS

The following subsections describe recommended modifications/additions to the IPAA software system. Some are logical extensions of the software design, while others are the result of analyst interviews, feedback from user classes, and general discussions with the customer.

Before pursuing the individual, specific recommendations, one over-all suggestion is presented. Interactive software systems, besides possessing good system design which lends itself to modification and expansion, must go through a period of "operator-proofing." Ideally, an interactive software system will not allow the analyst to "bomb" during its execution due to an operator-error. This ideal is seldom met in practice, however, due primarily to the inability of the software system to determine the intent of the analyst at all times, and secondarily the inability of the analyst to communicate directly with the software through the terminal, e.g., the bi-directional communication of the analyst and software system generally passes through a system device handler which may intercept certain commands and attempt executive system functions.

Additionally, the initial implementation of an interactive system is rarely entirely pleasing to the user. Even assuming the software system will perform all the functions the analyst desires, the method of implementation may make its use less desirable. As an example, consider the entry of parameters in the IPAA system. The designer may consider the following alternatives:

1. Have the user enter all parameters.

2. Have the user enter each parameter until a carriage return only is detected. This signals end of input and the remaining parameters are to remain unchanged.

3. Have the user enter carriage return only if he wishes the parameter to remain unchanged. Otherwise enter the new parameter. This method forces the user to exhaust the list of parameters, whether they are changed or not.
4. Have the operator enter the parameter number of the parameter he wishes to change.

Even through any of the methods will accomplish the task, the method chosen by the software designer may not be as desirable as another method for the user. Frequently this will not be discovered until actual use on real problems by the analyst. Many times these "fielder's choice" options do not require extensive re-design of the software system, only fairly straightforward modification. Should they have a serious effect on the software design, a suitable compromise can usually be worked out. Therefore, in the interest of providing useful, well-designed, and error-free software, it is recommended that future software modifications of the IPAA software system be implemented in an incremental manner, i.e., as each module or modification is completed, this latest version of IPAA be turned over to a select group of analysts for testing, and, if necessary, recommendations for alteration.

6.1 Macro Capability

The IPAA software system was designed with the eventual inclusion of macros in mind. To the analyst, it means modeling at a system level as opposed to the present primitive level.

6.1.1 Macro Definition

A macro, in IPAA terms, is a method of referencing a previously created diagram by the use of a single functional block. As an example, the page representing the QUAD subroutine in the Algorithm No. One diagram (diagram page 6) would be referenced by a block with the name QUAD for function.

6.1.2 Macro Implementation

The macro capability must have certain constraints in order for it to work within the present framework of the IPAA system.
A top-level view of a macro implementation/modification to the IPAA software system may be best understood by presenting an example and discussing the design considerations involved.

Figure 13 depicts the QUAD subroutine in diagram form. This particular "page" may be created with the current IPAA system; however, a method of storing this "page" in a library must be developed. Inclusion of the block depicted in Figure 14 within an IPAA diagram tells us that a method of replacing that block prior to execution with the stored "page" must be developed. From this simple macro example, the following design criteria may be inferred:

1. The maximum length of a macro must be established. It is reasonable to allow the full nine pages.

2. The not-so-obvious constraints; certain current blocks will not be allowed in macros, e.g., input and output blocks except as page connectors.

3. The maximum number of inputs to a macro will be four, for compatibility with the current software. (In the example there are two.)

4. Although this example would not require parameter modification prior to execution, it is possible that a method to do so would be desirable for other macros.

In addition, the available resources of the computer system will have an effect on the design. For example, the maximum number of blocks allowed for any execution (both diagram and macros) would be constrained by the amount of memory available at execution time. The method of storing and recalling a library of macros will be defined in large part by the capabilities and peculiarities of the operating system.

In conclusion, it may be stated that the inclusion of a macroing capability in the IPAA software is not only feasible, but may be implemented in a number of ways.
Figure 13. Quad Macro
Figure 14. QUAD MACRO REFERENCE
6.2 Graphic Display of Output Data

Analysis of processed data is less error prone and faster when displayed to the analyst in a graphic form as opposed to a columnar tabulation.

Inherent in a discussion of interactive displays is the capability for the analyst to expand displays both horizontally and vertically in order to view portions of data in more detail.

Experience has shown that many times analysts who have interactive displays available will develop measurement techniques which they wish to be included as options in the interactive display. The proper design of an interactive display will make possible its evolution into an interactive analysis position. As an example, an analyst may wish to measure the time between two events on his display. He should be able to simply mark the events with his cursor and get a rapid printout of this time difference.

6.3 Current System Modifications

Several suggestions have been made by analysts for modifications to the current IPAA software.

6.3.1 Uniscope Execution

It is possible to modify the software so that execution from a Uniscope terminal is possible. There was more interest in the initial stages of the contract for this capability than there was toward the end, so discussions with the analysts may show this to be of marginal value.

6.3.2 Addition of Functional Blocks

The capability of adding functions to the IPAA system exists and will continue whether or not a macroing capability is introduced. Assuming there was a macroing capability on the IPAA system, there would still exist a need for additional functions, since all models do not lend themselves to macroing.
Specific requests are:

- Random number generator block.

- Expansion of FGEN1 capability to allow storage of up to 200 table pair values.

- Labeling capability for the tabular output generated by DISP blocks.

- Capability to produce a greater range of values in the tabular output generated by DISP blocks.
7.0 REFERENCES


APPENDIX A

UTILIZATION OF LINES /'S BLOCK CONNECTORS

A discussion of an algorithmic approach to the problem of selectively connecting displayed blocks without superimposition of lines is presented.

Consider Figure 1. A user area is depicted having a possible 20 blocks which may be defined (shaded areas). (These shaded areas, whether defined or not, will not have lines drawn through them.)

It is helpful to view the shaded blocks as a 4x5 array, while the entire user area may be viewed as a 9x11 array.

In order to implement this on a digital computer, an array of length 9x11x2 may be set up in core, i.e., there are two words representing each block in the 9x11 array. One word will be a counter for the number of horizontal lines existing within that space, while the other will be a counter for the number of vertical lines in that space.

Now, if this array is initialized to zero and the words representing the shaded blocks are flagged so they won't be used, the algorithm may be defined.

Definition of variables used:

\[ \text{DX} = \text{distance between vertical lines} \]
\[ \text{DY} = \text{distance between horizontal lines} \]
\[ R_1, C_1 = \text{output block location in array} \]
\[ R_2, C_2 = \text{input block location in array} \]

Note that all outputs are from the right while inputs are into the left of the respective blocks.

The determination of path is, of course, dependent on the location of the input block relative to the output block (above, below, left, right) in the array.

In general, however, a vertical line is drawn one DX greater than the maximum of the vertical line counter for the spaces it will pass through, and similarly for the horizontal lines.
Figure 1 shows an example of two lines being drawn with this algorithm. Note that all lines proceed from output pin to input pin and all lines crossing should be ignored while tracing a path.
SECTION B.1.

ALGORITHM No. 1 CONVENTIONAL CODE LISTING
MOD-SUPG*G(1).MAIN:

1    
2    
3    
4    
5    
6    
7    
8    
9    
10   
11   
12   
13   
14   
15   
16   
17   
18   
19   
20   
21   
22   
23   
24   
25   

COMPILER(DIAG=3)

WRITE(6,20)

READ(5,10)IOFT

GO TO (1,2,3,4,5,6),IOFT

CALL VEHICLE

STOP

CALL VEL

STOP

CALL GIMBAL

STOP

CALL ANGLE

STOP

CALL SENSE

STOP

CALL MATH

STOP

FORMAT()

FORMAT(*3,2,ENTER OPTION CODE: */)

*3,1 = VEHICLE */

*3,2 = TOTAL VELOCITY */

*3,3 = EULER ANGLE TRANSFORMATIONS */

*3,4 = INSTRUMENT MOUNTING ANGLES */

*3,5 = SENSED VELOCITY FROM INSTRUMENTS */

*3,6 = MATH FUNCTIONS *)

END
PROGRAM CALLS SUBROUTINES PBV, THRU, IMSRT, QUAU

M=6

WRITE (*, 1000)
REAL (C)*=33333330
IF (LOGICAL('B',*')) WRITE(6,1001)
CALL PBV
CALL PBV
STOP

10 FORMAT(4)
11 FORMAT(A11)
1000 FORMAT(*'READY PROGRAM DESCRIPTION?'
1001 FORMAT(*'VERTICAL, HORIZONTAL AND LATERAL SENSED?'
1061 FORMAT(*'VERTICAL PROJECTED ONTO VEHICLE REFERENCE AXES. VEHICLE'
1062 FORMAT(*'VERTICAL COMPONENTS AND TOTAL VELOCITY ARE COMPUTED.
1063 FORMAT(*'SIGNED ACCELERATIONS ARE COMPUTED SIMILARLY. ENTER DATA',
1064 FORMAT(*'SIGNED AS FOLLOWS?'
1065 FORMAT(*'INCH GYMBAL COUNTS?'
1066 FORMAT(*'DEGREE GYMBAL COUNTS?'
1067 FORMAT(*'DEGREE GYMBAL COUNTS?'
1068 FORMAT(*'VERTICAL SENSED VELOCITY?'
1069 FORMAT(*'HORIZONTAL SENSED VELOCITY?'
1070 FORMAT(*'LATERAL SENSED VELOCITY?'
END
MDR=SCRM&G(1).TRMD

1      COMPLER(DIAG=3)
2      SUBROUTINE TRMD
3      PARAMETER IPRA=12,IPRB=2500,IPRC=IPRA/2
4      COMMON S(IPHA,IPRB),K,KOUNT
5      COMMON /A/UT(IPRC)
6      DIMENSION FHT(20),V(30),T(IPRC)
7      DIMENSION CHAN1(IPRC),CHAN2(IPRC)

C      VALUE OF K (NO. OF FILES) MUST BE SET BY CALLING
11      C      PROGRAM. SET K=0 IF NUMBER OF FILES IS TO BE
12      C      READ IN AT PROGRAM EXECUTION TIME.

C      COUNT=0 DATA POINT COUNT OF REFERENCE FILE
14      ICODE=5
15      E11=1*N
16      IF(K.EQ.0)GO TO 3
17      WRITE(6,101)
18      CNTINUE
19      READ(5,IO1)K
20      IFILE=IPRC
21      IF(K.GT.IFILE)WRITE(6,102)IFILE
22      IF(K.GT.IFILE)GO TO 4
23      CNTINUE
24      FORMAT='Y'
25      IF(K.GT.1)WRITE(6,103)
26      IF(K.GT.1)READ(5,II)FORMAT
27      DO 1 J1=1,K
28      IF(J1.EQ.1.OR.,FORMAT.NE.'Y')WRITE(6,104)
29      IF(J1.EQ.1.OR.,FORMAT.NE.'Y')READ(5,10)IF10M
30      IF(J1.GT.1.AND.,FORMAT.EQ.'Y')GO TO 5
31      IF(IFORM.LT.3)WRITE(6,105)
32      IF(IFORM.LT.3)READ(5,11)EDIT
33      IF(EDIT.EQ.'Y')ICODE=0
34      IF(IFORM.GT.2)WRITE(6,109)
35      IF(IFORM.GT.2)READ(5,10)N
IF (IFORM.EQ.3) WRITE(6,110)
IF (IFORM.EQ.3) READ(5,100) (FMT(I):I=1,20)
IF (IFORM.EQ.4) WRITE(6,111)
IF (IFORM.EQ.4) READ(5,10) IF
      5 CONTINUE
IF (IFORM.GT.2) WRITE(6,112)
IF (IFORM.GT.2) RF = 0(5,10) IX, IY
J = 0
IF (IFORM.LT.4) WRITE(6,106) J1
   11 = 2*J1-1
   12 = I1+1
   16 CONTINUE
   17 = J+1
   17 CONTINUE
IF (IFORM.EQ.1) READ(5,107) ENDC=19, ERR=17) IEDIT: S(I1,J),
* S(I2,J)
   31 IF (IFORM.EQ.2) READ(5,108) ENDC=19, ERR=17) S(I1,J), S(I2,J),
   33 * I EDIT
   34 IF (IFORM.EQ.3) READ(5,FMT) ENDC=19, ERR=17) (V(N2):N2=1,N)
   55 IF (IFORM.EQ.4) READ(IF,END=19, ERR=17) (V(N1):N1=1,N)
   56 IF (IFORM.GT.2) S(I1,J) = V(IX)
   57 IF (IFORM.GT.2) S(I2,J) = V(IY)
   58 IF (IFORM.GT.2) GO TO 20
   59 IF (IEDIT.GT.1, ICODE) GO TO 17
   60 IF (S(I1,J).LE.0) GO TO 17
   62 IF (S(I1,J).GT.99999999) GO TO 17
   63 IF (U.GT.1) GO TO 18
   64 IF (S(I1,J).GT) A1 = IT
   65 T(J1) = S(J1)-AT
   66 LT(J1) = T(J1)-T(1)
   68 16 CONTINUE
   69 GO TO 16
CONTINUE
KOUNT=J=1
WRITE(F,11),U1,KOUNT
IF(U1,EL,E),KOUNT=KOUNT+1
IF(IF=0,E,4),REAL(UU,1)
CONTINUE
KOUNT=KOUNT+1
CALL I120R

PRINT OUT SOURCE DATA

WRITE(E,14),KOUNT,S(1),S(2),KOUNT1
WRITE(E,15)
REAL(5,11)PR1
IF(IPRT.EQ.0),PRINT TO 24
IA=1
IB=KOUNT
IF(IPRT.EQ.1),WRITE(6,115)KOUNT
IF(IPRT.EQ.2),WRITE(6,116)
IPRT=IPRT+1
IF(IPRT.EQ.3),WRITE(6,117)
IPRT=IPRT+1
IF(IPRT.EQ.4),WRITE(5,118)KOUNT
IPRT=IPRT+1
IF(IPRT.EQ.5),WRITE(5,119)
IF(IPRT.EQ.6),WRITE(5,120)
IF(IPRT.EQ.7),WRITE(5,121)
IF(IPRT.EQ.8),WRITE(5,122)
IF(IPRT.EQ.9),WRITE(5,123)
IF(IPRT.EQ.10),WRITE(5,124)
IF(IPRT.EQ.11),WRITE(5,125)
IF(IPRT.EQ.12),WRITE(5,126)
IF(IPRT.EQ.13),WRITE(5,127)
IF(IPRT.EQ.14),WRITE(5,128)
IF(IPRT.EQ.15),WRITE(5,129)
IF(IPRT.EQ.16),WRITE(5,130)
IF(IPRT.EQ.17),WRITE(5,131)
IF(IPRT.EQ.18),WRITE(5,132)
IF(IPRT.EQ.19),WRITE(5,133)
IF(IPRT.EQ.20),WRITE(5,134)
IF(IPRT.EQ.21),WRITE(5,135)
IF(IPRT.EQ.22),WRITE(5,136)
IF(IPRT.EQ.23),WRITE(5,137)
IF(IPRT.EQ.24),WRITE(5,138)
CONTINUE

IF(IF2 .NE. 6)CALL CLOSE(1F2,1)
17  IF(IF2 .NE. 6)WRITE(6,122)IF2
10 CONTINUE
11 OPEN
12 FORMAT()
13 FORMAT(*A1)
14 FORMAT(PL/6)
15 FORMAT(X,'ENTER NUMBER OF DATA FILES TO BE USED')
16 FORMAT(3X,' (NO TITLE)')
17 FORMAT(3X,' LIMIT, RE-ENTER NO. OF FILES)
18 FORMAT(-X,'SAFETY INPUT FORMAT FOR ALL FILES')
19 FORMAT(-X,'ENTER INPUT DATA FILE FORMATTING')
20 FORMAT(X,' REAL VARIABLES & WRITE IN FORMAT')
21 FORMAT(X,' REAL ONLY EDITED POINTS')
22 FORMAT(X,' ALL DATA FILE DESIGNATION AND DECK')
23 FORMAT(X,' FOR FILE ')
24 FORMAT(3X,'0.00')
25 FORMAT(3X,'P')
26 FORMAT(3X,'')
27 FORMAT(3X,'P')
28 FORMAT(3X,'')
29 FORMAT(3X,'P')
30 FORMAT(3X,'')
31 FORMAT(3X,'P')
32 FORMAT(3X,'')
33 FORMAT(3X,'P')
34 FORMAT(3X,'')
35 FORMAT(3X,'P')
36 FORMAT(3X,'')
37 FORMAT(3X,'P')
38 FORMAT(3X,'')
39 FORMAT(3X,'P')
40 FORMAT(3X,'')
41 FORMAT(3X,'P')
42 FORMAT(3X,'')
43 FORMAT(3X,'P')
44 FORMAT(3X,'')
45 FORMAT(3X,'P')
46 FORMAT(3X,'')
47 FORMAT(3X,'P')
48 FORMAT(3X,'')
49 FORMAT(3X,'P')
50 FORMAT(3X,'')
51 FORMAT(3X,'P')
52 FORMAT(3X,'')
53 FORMAT(3X,'P')
54 FORMAT(3X,'')
55 FORMAT(3X,'P')
56 FORMAT(3X,'')
57 FORMAT(3X,'P')
58 FORMAT(3X,'')
59 FORMAT(3X,'P')
60 FORMAT(3X,'')
61 FORMAT(3X,'P')
62 FORMAT(3X,'')
63 FORMAT(3X,'P')
64 FORMAT(3X,'')
65 FORMAT(3X,'P')
66 FORMAT(3X,'')
67 FORMAT(3X,'P')
68 FORMAT(3X,'')
69 FORMAT(3X,'P')
70 FORMAT(3X,'')
71 FORMAT(3X,'P')
72 FORMAT(3X,'')
73 FORMAT(3X,'P')
74 FORMAT(3X,'')
75 FORMAT(3X,'P')
76 FORMAT(3X,'')
77 FORMAT(3X,'P')
78 FORMAT(3X,'')
79 FORMAT(3X,'P')
80 FORMAT(3X,'')
81 FORMAT(3X,'P')
82 FORMAT(3X,'')
83 FORMAT(3X,'P')
84 FORMAT(3X,'')
85 FORMAT(3X,'P')
86 FORMAT(3X,'')
87 FORMAT(3X,'P')
88 FORMAT(3X,'')
89 FORMAT(3X,'P')
90 FORMAT(3X,'')
91 FORMAT(3X,'P')
92 FORMAT(3X,'')
93 FORMAT(3X,'P')
94 FORMAT(3X,'')
95 FORMAT(3X,'P')
96 FORMAT(3X,'')
97 FORMAT(3X,'P')
98 FORMAT(3X,'')
99 FORMAT(3X,'P')
100 FORMAT(3X,'')
101 FORMAT(3X,'')
102 FORMAT(3X,'')
103 FORMAT(3X,'')
104 FORMAT(3X,'')
105 FORMAT(3X,'')
106 FORMAT(3X,'')
107 FORMAT(3X,'')
108 FORMAT(3X,'')
109 FORMAT(3X,'')
110 FORMAT(3X,'')
111 FORMAT(3X,'')
112 FORMAT(3X,'')
```plaintext
141 119 FORMAT(1X,'ENTER "TMKE" OUTPUT FILE DESIGNATER (NO DEC)')
142 120 FORMAT(*14/,'PSEUDO TIME',5X,5D(2A6))
143 121 FORMAT(1X,F12.3,5DF12.0)
144 122 FORMAT(1X,'OUTPUT IN FILE',15/,'FORMAT(1X,F12.3,1NF12.0)')
145 123 FORMAT(1X,'PRINT OUT SORTED DATA?')
146 END
```
MDR=SDR*F(1)*Puv

COMPILE (DIAG=3)
SUBROUTINE PUV

PARAMETER IPRC=12, IPRB=2500, IPRC=IPRC/2
DOUBLE PRECISION SPH,CPH,SHS,CPS,ST,CT,X4,Y4,Z4
DOUBLE PRECISION X3,Y3,Z3,X2,Y2,Z2,X1,Y1,Z1
DOUBLE PRECISION AX4,AY4,A24
COMMON (MPRA,IPRB)*K,KUANT

DIMENSION V(128)

COMPUTER BODY RELATED VELOCITY AND ACCELERATION DATA

CALL (4)KIND(6,*FREE 7,*8)
WRITE(*,1001)
REAL(5,J=1,IC=GT-1)WRITE(6,I=1001)
REAL(J=1,IC=GT-1)READ(S,10)SH,SY,SP
IF(J=1,IC=1,IC=GT-1)GO TO 20
X=0
Y=0
Z=0
1000 CONTINUE
WRITE(*,1000)
READ(S,10)AVSF A SCALE FACTOR "UNITS" DESIGNATOR
WRITE(*,1001)
REAL(S,J=1,IC=GT-1)WRITE(S,1001)
REAL(S,J=1,IC=GT-1)WRITE(S,1004)
REAL(S,J=1,IC=GT-1)GET AFFAME TIME
U=3.14159*6/180
DO 200 'U'='KUANT
PH=S(2,J)*U/GF
PS=S(4,J)*G/GY
L=3(6,J)*U/GP
T1=M=3(1,J)
SPH=DSR*F(PH)
35  CPH=DCOS(PH)
36  SPS=USIN(PS)
37  CPS=DCOS(PS)
38  ST=USIN(T)
39  CT=DCOS(T)
40  X4=S(10,I)/GV
41  Y4=S(12,I)/GV
42  Z4=S(6,I)/GV
43  I1=I-1
44  GT1=(S(1,I)-S(1,I1))*G1
45  AX4=((S(10,I)-S(10,I1))/GV)/GT1
46  AY4=((S(12,I)-S(12,I1))/GV)/GT1
47  AZ4=((S(6,I)-S(6,I1))/GV)/GT1
48  DO 100 J1=1,2
49
50  102  CONTINUE
51      X4=AX4
52      Y4=AY4
53      Z4=AZ4
54  101  CONTINUE
55      X3=X4*CPH-Y4*SPS
56      Y3=Y4*SPS+Y4*CPH
57      Z3=Z4
58      X2=X3
59      Y2=Y3*CPS+Z3*SPS
60      Z2=Y3*SPS+Z3*CPS
61      X1=X2*CT+Z2*ST
62      Y1=Y2
63      Z1=-X2*ST+Z2*CT
64      AMAG=SWRT(X1**2+Y1**2+Z1**2)
65      Y11=-Y1
66      X11=X1
67      Z11=Z1
68      CALL QUAD(X11,Y11,PHI)
69      XX=SORT(X11**2+Y11**2)
CALL QUAD(XX*Z11*GAMMA)
GO TO(103,104),J1
103 CONTINUE
M=1
V(1)=TIME
104 CONTINUE
M1=M+1
M2=M1+1
M3=M2+1
M4=M3+1
M5=M4+1
M6=M5+1
M=M6
V(M1)=X1
V(M2)=Y1
V(M3)=Z1
V(M4)=AMAG
V(M5)=PHI
V(M6)=GAMMA
100 CONTINUE
WRITE(7)(V(I2),I2=1,13)
200 CONTINUE
CALL CLOSE(7,1)
C WRITE OUT HEADINGS, DATA, AND PLOT FILES
C
WRITE(6,1005)
READ(5,10)IPB
IF(IPB.EQ.0.OR.IPB.LT.0)GO TO 121
IF(IPB.GT.KOUNT)IPB=KOUNT
100 IF(IPB.GT.0)WRITE(6,1006)
101 IF(IPB.GT.0)READ(5,10)IF2
102 IF(IPB.GT.0)WRITE(IF2,1008)
103 IF(IVSF.EQ.1)WRITE(IF2,1009)GV,GT
104 IF(IVSF.EQ.2)WRITE(IF2,1010)GV,GT
105 CONTINUE
106 READ(7,EN=120)(V(I3),I3=1,13)
107 KT=KT+1
109 IF(KT.GT.1P8)GO TO 120
110 WRITE(IF2,1007)(V(I4),I4=1,13)
111 GO TO 119
112 CONTINUE
113 IF(IF2.EQ.6)GO TO 121
114 CALL CLOSE(IF2,1)
115 RETURN
116 WRITE(6,1161)
117 IF(IVSF.EQ.1)WRITE(6,1009)GV,GT
118 IF(IVSF.EQ.2)WRITE(6,1010)GV,GT
119 RETURN
120 CONTINUE
121 RETURN
122 FORMAT(/)
124 FORMAT('INPUT GIMBAL DATA IN COUNTS OR',*DEGREES)< COUNTS=1, DEGREES=0')
126 FORMAT('ENTER INNER, MIDDLE, OUTER GIMBAL SCALE',*SCALE FACTOR, CTS/DEG. (INCL. DEC.))
128 FORMAT('ENTER VELOCITY SCALE FACTOR, INCL. DEC.')
130 FORMAT('ENTER FRAME TIME')
132 FORMAT('ENTER OUTPUT? ENTER NO. OF PTS. (NO DEC.))'
134 FORMAT('ENTER OUTPUT FILE DESIGNATER (NO DEC.)')
136 FORMAT('OUTPUT VARIABLE (COLUMN) IDENTIFICATION: ',*)
138 #x, '1'-PSEUDO TIME'*/
139 #x, '2'-VEHICLE XDOT, YDOT, ZDOT'*/
140 #x, '5'-TOTAL VELOCITY MAGNITUDE'*/
139
140
141
142
143
144
145
146
147
148
149

\*5X, \*6-ANGLE OF TOTAL VELOCITY VECTOR RELATIVE /\,
\*10X, \*6 LEFT OF VEHICLE VERTICAL LONGITUDINAL PLANE (PHI) /\,
\*5X, \*7-ANGLE OF TOTAL VELOCITY VECTOR RELATIVE /\,
\*10X, \*7 VEHICLE HORIZONTAL PLANE (GAMMA) /\,
\*5X, \*8-9 SAME AS 2-7 EXCEPT FOR ACCELERATION /\
1009 FORMAT('X*VELOCITY SCALE FACTOR IS *F10.3, CTS/METER/SEC *',
    \"FRAME IS *F10.6, SEC *", /\)
1010 FORMAT('X*VELOCITY SCALE FACTOR IS *F10.3, CTS/FT/SEC * *',
    \"FRAME IS *F10.6, SEC *", /\)
1011 FORMAT('X*OUTPUT IN FILE *', /\)
\*4,
1012 FORMAT('X*UNFORMATTED PLOT OUTPUT ON UNIT ? *')
END
SUBROUTINE QUAD(X,Y,C)

PI=3.1415926535897932/180.
V=SQRT(X**2+Y**2)
A=X/V
B=Y/V
IF(A>10,1,31)

31 C=ASIN(-V)/PI
GO TO 34

30 IF(B>10,33,33)

33 C=180.+SIN(B)/PI
GO TO 34

32 C=-180.-ASIN(B)/PI
34 RETURN

END
SECTION B.2.

ALGORITHM NO. 1 IPAA DIAGRAMS
## IPMA Processor Information

### Processing Description:

1. **Diagram Name**
   - ALGOU.
2. **Start Time**
   - 1.000
3. **Stop Time**
   - 41.000
4. **Time Increment**
   - 1.000
5. **Input Interpolation Degree**
   - 1
6. **Display Ratio**
   - 1
7. **Time Option**
   - TIME INPUT PROVIDED

### Inputs:

8. **Input Format**
   - FORMATTED DATA

9. **Input File No. 1**
   - FILE NAME: GIMSALE
   - START TIME: 0.000

10. **Number of Fields**
    - 7

11. **Input Reference**
    - 1 2 3 4 5 6 7

12. **Field Width**
    - 1 2 3 4 5 6 7
    - 15151515151515

### Output:

13. **Output Format**
    - DEFAULT FORMAT(POT)

14. **Output File No. 1**
15. **Output File No. 2**
16. **Output File No. 3**
17. **Output File No. 4**
18. **Save PIF in File**
19. **Restore PIF from File**
20. **Return to Diagram**
21. **Execute**
## Output Algo 81 Acceleration

### PROCESSING DESCRIPTION

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Start Time</td>
<td>1.000</td>
</tr>
<tr>
<td>4</td>
<td>Stop Time</td>
<td>41.000</td>
</tr>
<tr>
<td>5</td>
<td>Time Increment</td>
<td>1.000</td>
</tr>
<tr>
<td>6</td>
<td>Input Interpolation Degree</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>Display Ratio</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>Time Option</td>
<td>TIME INPUT PROVIDED</td>
</tr>
</tbody>
</table>

### INPUTS

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>Input Format</td>
<td>FORMATTED DATA</td>
</tr>
<tr>
<td>10</td>
<td>Input File No. 1</td>
<td>FILE NAME</td>
</tr>
<tr>
<td></td>
<td>Number of Fields</td>
<td>7</td>
</tr>
<tr>
<td>12</td>
<td>Input Reference</td>
<td>1 2 3 4 5 6 7</td>
</tr>
<tr>
<td>13</td>
<td>Field Width</td>
<td>1 2 3 4 5 6 7 151515151515</td>
</tr>
</tbody>
</table>

### OUTPUT

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>Output Format</td>
<td>DEFAULT FORMAT</td>
</tr>
<tr>
<td>15</td>
<td>Output File No. 1</td>
<td>FILE NAME</td>
</tr>
<tr>
<td>16</td>
<td>Output File No. 2</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Output File No. 3</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Output File No. 4</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Save PIF in File</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Restore PIF from File</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>Return to Diagram</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>Execute</td>
<td></td>
</tr>
</tbody>
</table>
SECTION B.3
ALGORITHM NO. 2 CONVENTIONAL CODE LISTING
JAG='SU' P#PASSRAT(1) PROG1

1       LU1=6
2       LU2=9
3       LU4=10
4       WRITE (=*10)
5       FORMAT (* ENTER START, STOP, INCREMENT*)
6       REAL (5,11) EMRR,EMRF,EMRI
7       J FORMAT (1)
8       ICT=(EMMK=EMPRF)/EMRI+1
9       EMRK (*=*=100)
10      WRITE (*=*=100)
11      L00      FORMAT (*7*,EMMK=11X,UPBMRR=10X,*PCTPBF=10X,
12      =*FPLBMRR=10X,*RPBMRR=10X,*PCTPBO*)
13      UO 100 U 1=1,ICT
14      EKF=1.0/(EMRK+1.0)
15      LQO = EMRR*EFK
16      E K 6 X RICH PRE-TURNER
17      PWF = .75 = FLO
18      UPBMRR=E00/PLRF
19      PLTRBF=PPRF/EFK
20      C      FUEL RICH PRE-BURNER
21      PWU = .75 - EWF
22      PFR=K=PPRF/EFK
23      PLTPBF=PPRF/EWF
24      KN=PK=1.0/PPBMRR
25      C      WRITE (LUN=1,300) EMRR,OPBMRR
26      300      FORMAT (1X,E14.6*,*E14.6)
27      300      WRITE (LUN=2,300) EMRK,FPBMRR
28      WRITE (LUN=3,300) EMRR,RPBMRR
29      WRITE (LUN=3,300) EMRF,RPBMRR
30      C      WRITE (LUN=3,300) EMRR,OPBMRR,PCTPBF,FPBMRR,RPBMRR,PCTPBO
31      300      FORMAT (1X,9(2X,E14.9))
32      300      EMRK = EMRR + EMRI
33      300      END FILE LUN1
SECTION B.4

ALGORITHM NO. 2 IPAA DIAGRAMS
### IP豪华处理器信息###

1. **处理描述**

2. **图名称**

3. **开始时间** 0.000

4. **结束时间** 3.000

5. **时间增量** 0.100

6. **输入内插度** 0

7. **显示比例** 1

8. **时间选项** TIME INPUT PROVIDED

---

**输入**

9. **输入格式** DEFAULT FORMATTED

10. **输入文件号 1**

11. **输入文件号 2**

12. **输入文件号 3**

13. **输入文件号 4**

---

**输出**

14. **输出格式** DEFAULT FORMATTED

15. **输出文件号 1**

16. **输出文件号 2**

17. **输出文件号 3**

18. **输出文件号 4**

19. **保存PIF到文件**

20. **从文件恢复PIF**

21. **返回到图表**

22. **执行**

START TIME 0.000 0.000 0.000 0.020
MISSION
of
Rome Air Development Center

RADC plans and conducts research, exploratory and advanced development programs in command, control, and communications (C³) activities, and in the C³ areas of information sciences and intelligence. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.