CR151904

SIGMA CORPORATION
TECHNICAL NOTE

TN 76-116

(NASA-CR-151904) ROBOT DISPLAY AND CONTROL PROGRAM (ROBDAC) (Sigma Corp., Houston, Tex.) 33 p HC A03/MF A01 CSCL 09B

N79-17576

Unclas
G3/61 13936

ROBOT DISPLAY AND CONTROL PROGRAM (ROBDAC)

By: William A. Stewart

Prepared for:

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
Johnson Space Center
Houston, Texas 77058

July 1976

TABLE OF CONTENTS

SUBROUTINES DEFINITIONS

Main Display Routines

CONTRL	Generates ROBOT Control Menu
PARIN	ROBOT Parameter Input Selected in Control
ARRIN	ROBOT Array Input Menu
ARRAY	STD Array Edit Routine Called from ARRIN (used 34 times in ROBOT arguments - NAME,VALUE,NDP, DESC,NWD)
TABIN	ROBOT Table Input Menu
PLOTIN	Controls Input Data Plots
SARDAT	Store Plus Retrieve Data
TRAJDAT	Setup of Trajectory Plot Data
TRAJPLT	Display Plot of Trajectory Output Parameters
NUMTRJ	Display Trajectory Output Parameters and Values
NUMCON	Display Convergence Output Parameters and Values
CONVPLT	Display Plot of Convergence Output Parameters
PERFSUM	Display Performance Summary Parameters and Values
DATRAN	Data Transformation and Format

Utility Display Routines

DISTAB	Display Table of Values
DISPLT	Display Plot of ROBOT Input Parameters
DISSCL	Display Scale Factors
DISPOP	Display Plot Options

SUMMARY AND INTRODUCTION

This report is a preliminary design of the ROBOT Display and Control Program (ROBDAC) to be written for the Adage 340 computer. This program is designed to communicate with the Univac 1110 computer with the aid of the Graphic Support Communications Programs (GS COMM Package). The GS COMM Package allows data to be shipped and received between the Univac and Adage computers when the data meets input/output requirements necessary for a transfer. ROBDAC and the Adage computer act as an input/output device for the program ROBOT, which is run on the Univac 1110. At first, consideration was given to running the ROBOT Program on the Adage. However, due to the size of ROBOT, and the time and accuracy involved in its calculations, this idea was dropped. ROBDAC controls the input to ROBOT by displaying images of the inputs, providing an editing technique for these inputs, shipping the input data to the Univac, and then commanding ROBOT to start execution. The output is done in a similar manner after receiving the output data from the Univac. This data will be displayed numerically as well as plotted graphs. All of the input/output will be done interactively with practically "finger tip" control. A general program flow diagram of ROBDAC is shown in figure 1.

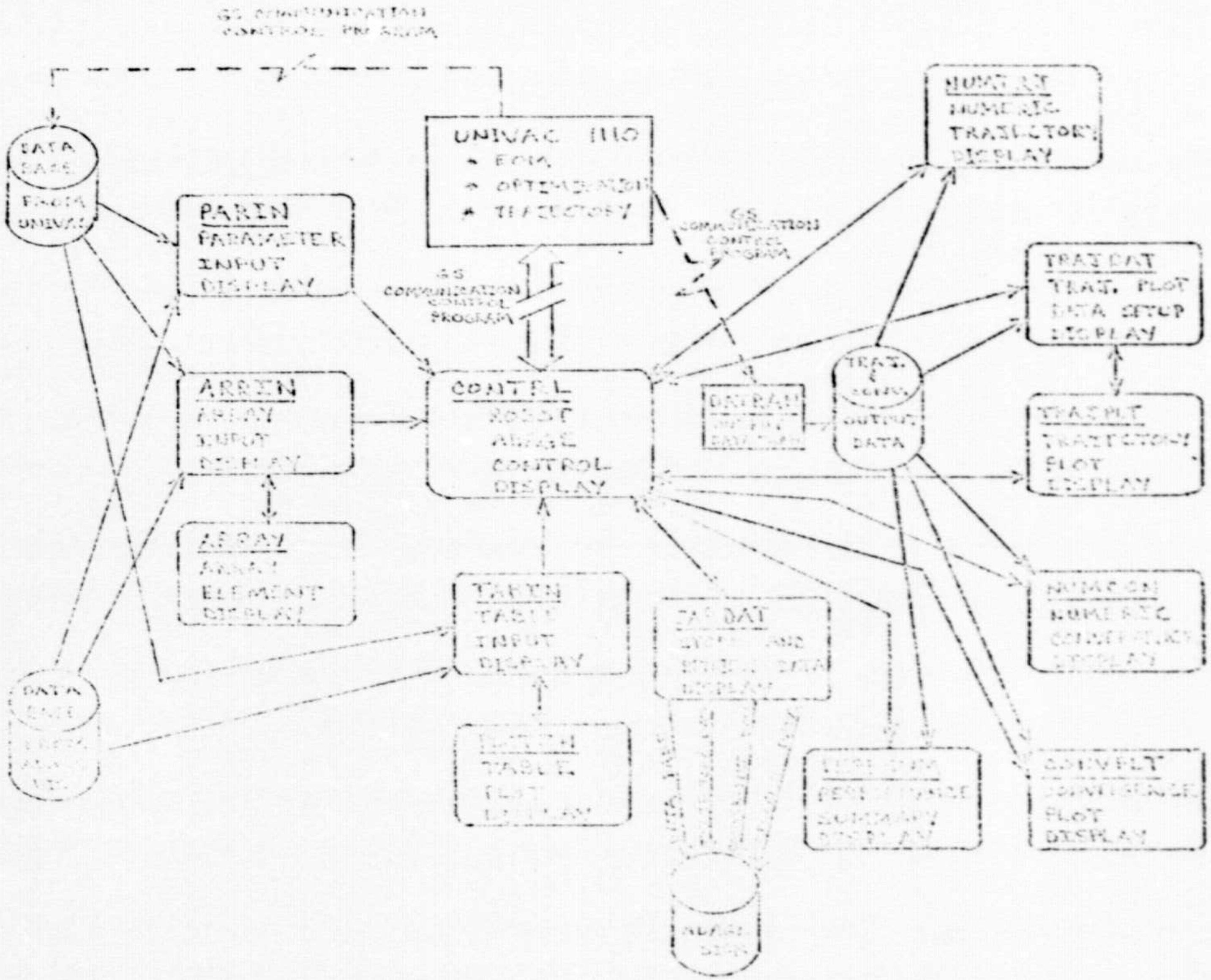


FIGURE 1. ROBOT DISPLAY AND CONTROL PROGRAM

ORIGINAL PAGE IS
OF POOR QUALITY

PROGRAM DESIGN

The intent of the ROBDAC Program is to interactively communicate from the Adage 340 with the ROBOT Program on the Univac 1110. The data transfer between the two computers is handled by the GS COMM package. This package is a series of programs on both computers that allows data to be sent, tested and received so that the main programs running on each computer are not interfered with or received erroneous data. A description of the GS COMM programs are in Appendix A. The ROBDAC Program consists of several images which aid in the analysis of input/output ROBOT data.

The first display image is the ROBOT CONTROL MENU. This image consists of input/output options and commands to start ROBOT on the Univac (refer to figure 2). Any option or command is selected by depressing its corresponding function switch. If an option is selected that needs additional input, the ANK is the input device. As noted in figure 2, there are eight input options, of which only the first four will be discussed at this time. The last four options include the same inputs as the first four, but these inputs are grouped in different array images to be designed later.

The first option is LOAD DATA which loads the desired data base from disk into Adage core. Additional input will be typed in from the ANK telling from what pack/volume and file name the data base is stored.

The PARAMETER INPUT option is a display of one element array inputs to ROBOT. Figure 3 shows an example format of the parameters. The parameters are listed alphabetically with their values, units and descriptions. All of the input parameters cannot be displayed at one time, therefore, function switch 31 will page the data for display of additional input parameters. Any parameter value may be changed by depressing the function switch associated with that parameter and typing in the change from the ANK. When all parameters are set properly, function switch 32 will return to the control menu.

The ARRAY INPUT option is a display shown in figure 4. This option displays all of the multiple element arrays, the number of elements in each array, and the array description. The commands at the bottom of the display are the same as described earlier. Upon depressing a function switch associated with an array name, a new display appears as shown in figure 5. This display is an example of the array PRINT with each element and its value listed. The description is the same for all elements. Any array element can be selected with its function switch and the array value changed via the ANK.

ROBOT CONTROL MENU

FNS	INPUT OPTIONS	FNS	OUTPUT OPTIONS
1	LOAD DATA	16	STORE AND RETRIEVE DATA
2	PARAMETER INPUT	17	TRAJECTORY PLOT SETUP
3	ARRAY INPUT	18	PLOT TRAJECTORY
4	TABLE INPUT	19	NUMERIC TRAJECTORY
5	THRUST EVENT DEP. INPUT	20	PLOT CONVERGENCE
6	OPTIMIZATION INPUT	21	NUMERIC CONVERGENCE
7	GRAVITATIONAL INPUT	22	PERFORMANCE SUMMARY
8	INITIAL CONDITIONS		

30	START TRAJECTORY RUN
31	CONTINUE TRAJECTORY RUN
32	STOPS PROGRAM

FIGURE 2 CONTROL DISPLAY.

ROBOT PARAMETER INPUT

FNS	NAME	VALUE	UNITS	DESCRIPTION
1	ALONGO	80.5649	DEG	LONGITUDE OF LAUNCH SITE
2	AYL	0.002		FORWARD INTEGRATION ERROR CHECK
3	AZ	90.	DEG	LAUNCH AZIMUTH

31 PAGE THE DATA
32 RETURN TO CONTROL MENU

(THIS SPACE AVAILABLE FOR ANK INPUT)

FIGURE 3 INPUT PARAMETER DISPLAY.

ROBOT ARRAY INPUT

FNC	NAME	NUM	DESCRIPTION
1	AE	15	ENGINE EXIT AREA FOR VEHICLE 1 (ORBITER)
2	AE2	15	ENGINE EXIT AREA FOR VEHICLE 2 (BOOSTER)
3	BSTEP	15	BACKWARD INTEGRATION STEP SIZE

31 PAGE THE DATA
32 RETURN TO CONTROL MENU

FIGURE 4 INPUT ARRAY DISPLAY.

ARRAY PRINT

FNS	ELEMENT	VALUE	DESCRIPTION
1	1	2.	PRINT TIME INCREMENT FOR EACH THRUST EVENT
2	2	8.	
3	3	0.	
4	4	0.	
5	5	0.	
6	6	0.	
7	7	0.	
8	8	0.	
9	9	0.	
10	10	0.	
11	11	0.	
12	12	0.	
13	13	0.	
14	14	0.	
15	15	0.	

31 PAGE THE DATA

32 RETURN TO ARRAY LIST

(THIS SPACE AVAILABLE FOR ANK INPUT)

FIGURE 5 ARRAY EXAMPLE DISPLAY.

TABLE INPUT is the last input method discussed and its display is shown in figure 6. The display lists the seven tables of ROBOT input with a brief description of each. The X-name is the independent variable plotted along the X-axis and there are as many as three Y-names which are the dependent variables plotted along the Y-axis. An example of selecting the second table, which has three dependent variables, is shown in figure 7. This display writes the name of the table at the top of the CRT and calls four subimages. The subimages are listed as utility display routines in the subroutines definitions. DISTAB displays a table of values in the upper right hand corner of the CRT. The example in figure 7 shows PNM and CAN values listed. The plot of PNM VS. CAN in the upper left hand corner is displayed by the image DISPLT. The scale for the plot defaults to equally segmented grids between the maximum and minimum values that are plotted. A user has the capability to change the scale with the image DISSCL displayed in the lower right hand corner. The maximum and minimum values and the number segments can be changed via the function switches. In the lower left hand corner DISPOP is displayed which is the options of this input technique. Either one of the other two dependent variables of this table can be selected and plotted via the function switches. The numeric values of the selected plot will change accordingly. The option GLOMMER allows a user to move a point on the plot to another location. The SCROLL option will move additional numeric data points into view if a table has more than fifteen (15) points. When all inputs are satisfied with the selected table, function switch 32 will return to the table input for another table selection.

The first option in the output is STORE AND RETRIEVE DATA shown in figure 8. The options here allow a user to store or retrieve specified data via the function switches. For example, a user might want to retrieve a data base EDIN0620. This would be accomplished by depressing function switch 2, then 3. A message at the lower right hand corner will appear asking for the name of the data base and the pack/volume where it is located. After the computer has time to read the data into core function switch 32 will return to the control menu.

The TRAJECTORY PLOT DATA lists all of the output parameters of ROBOT as well as their values, units and descriptions (refer to figure 9). Two parameters need to be selected via the function switches, then by depressing function switch 30, they will be plotted. The first parameter selected will be plotted along the X-axis and the second along the Y-axis. Function switch 29 is a reset option if a wrong parameter is selected and a user wishes to start over.

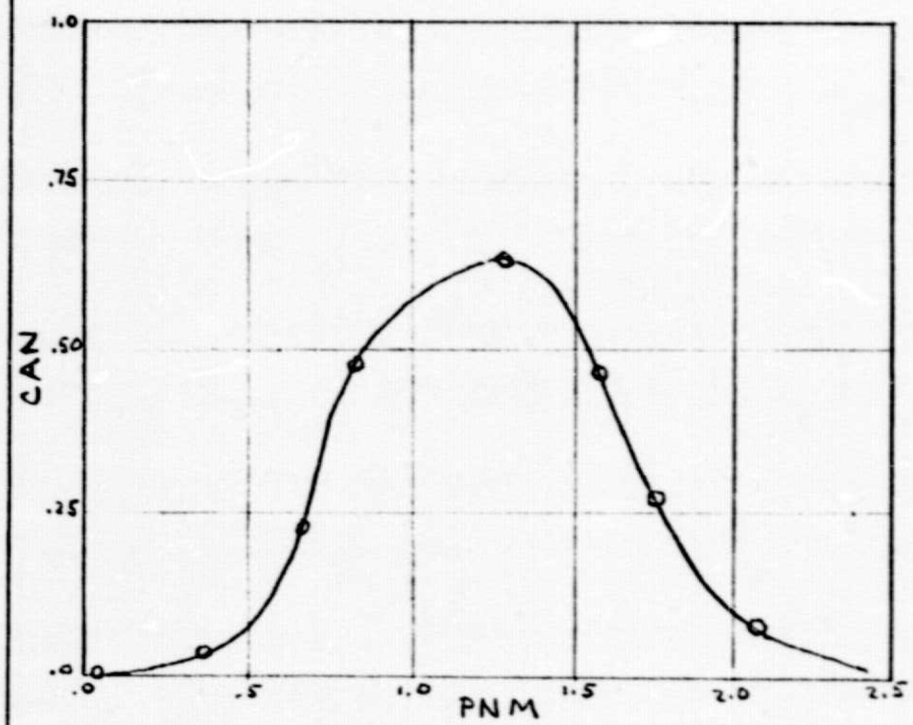
ROBOT TABLE INPUT

FNS	X-NAME	Y-NAME	Y-NAME	Y-NAME	DESCRIPTION
1	PFM	CAF	CNF	—	POWER OFF AERODYNAMICS = F(MACH)
2	PNM	CAN	CNOTAB	CNN	POWER ON AERODYNAMICS = F(MACH)
3	XMACHC	CMOTAB	CMATAB	—	PITCH AERODYNAMICS = F(MACH)
4	FABAL	FABTB	—	—	BASE DRAG = F(ALT)
5	TT1	THRTN	WPTN	—	THRUST AND FUEL FLOW = F(TIME)
6	WTTAB	XCGTAB	ZCGTAB	—	CG = F(WEIGHT)
7	TTBL	CPTBL	CYTBL	—	CONTROL TABLES = F(TIME)

32 RETURN TO CONTROL MENU

FIGURE 6 INPUT TABLE DISPLAY.

POWER-ON AERODYNAMICS = F(MACH)



FNS	PNM	CAN
1	0.0	0.0
2	0.5	~
3	0.7	~
4	0.8	~
5	0.9	~
6	1.0	~
7	1.05	~
8	1.10	~
9	1.25	~
10	1.35	~
11	1.45	~
12	1.50	~
13	1.75	~
14	2.00	~
15	2.50	1.0

	FNS	X-AXIS	FNS	Y-AXIS
MIN	16	~	17	~
MAX	18	~	19	~
SEG	20	~	21	~

FNS	OPTIONS	FNS	OPTIONS
22	PNM VS CAN	25	GLOMMER
23	PNM VS CNOTAB	26	SCROLL
24	PNM VS CNN	27	

32 RETURN TO TABLE INPUT

(THIS SPACE AVAILABLE FOR ANK INPUT)

FIGURE 7 TABLE EXAMPLE DISPLAY.

STORE AND RETRIEVE DATA

FNS	OPTIONS	PUV	FILE NAME
1	STORE		
2	RETRIEVE		
3	DATA BASE	222	EDIN0620
4	TRAJECTORY DATA	110	TRAJ0620
5	CONVERGENCE DATA	110	CONV0620

32 RETURN TO CONTROL MENU (THIS SPACE AVAILABLE FOR ANK INPUT)

FIGURE 8 STORE AND RETRIEVE DISPLAY.

TRAJECTORY PLOT DATA

FNS	NAME	VALUE	UNITS	DESCRIPTION
1	T	0.0	SEC	TRAJECTORY TIME
2	VEL	0.0	FT/SEC	VEHICLE INERTIAL VELOCITY
3	H	0.0	FT	VEHICLE ALTITUDE

29 RESET - CHANGE PARAMETERS TO BE PLOTTED
30 PLOT -- TWO PARAMETERS NEED TO BE SELECTED
31 PAGE TRAJECTORY DATA
32 RETURN TO CONTROL MENU

FIGURE 9 TRAJECTORY PLOT SETUP DISPLAY.

The TRAJECTORY PLOT display plots the two parameters selected in TRAJECTORY PLOT DATA, as shown in figure 10. The plotting format and method are the same as in the input table plot image.

The NUMERIC TRAJECTORY is a display of the ROBOT output parameters, values and units for specific observation points. Options at the bottom of the display steps or scrolls the values for other iteration points forward or backward. Figure 11 shows an example of this display.

The CONVERGENCE PLOT display plots the optimization parameter and its terminal constraints. Figure 12 shows an example with GLOW being the optimization parameter and the other four parameters are constraints. A series of plots of the parameters can be selected via the function switches in the lower right hand corner.

The NUMERIC CONVERGENCE displays the numeric values of the convergence parameters as shown in figure 13. Additional parameters are displayed to aid a user when a particular trajectory might not be converging. The commands at the bottom of the display have the same function as in the NUMERIC TRAJECTORY.

The last display is the PERFORMANCE SUMMARY in figure 14. This is a list of critical trajectory parameters and their values at the end of each stage that is available.

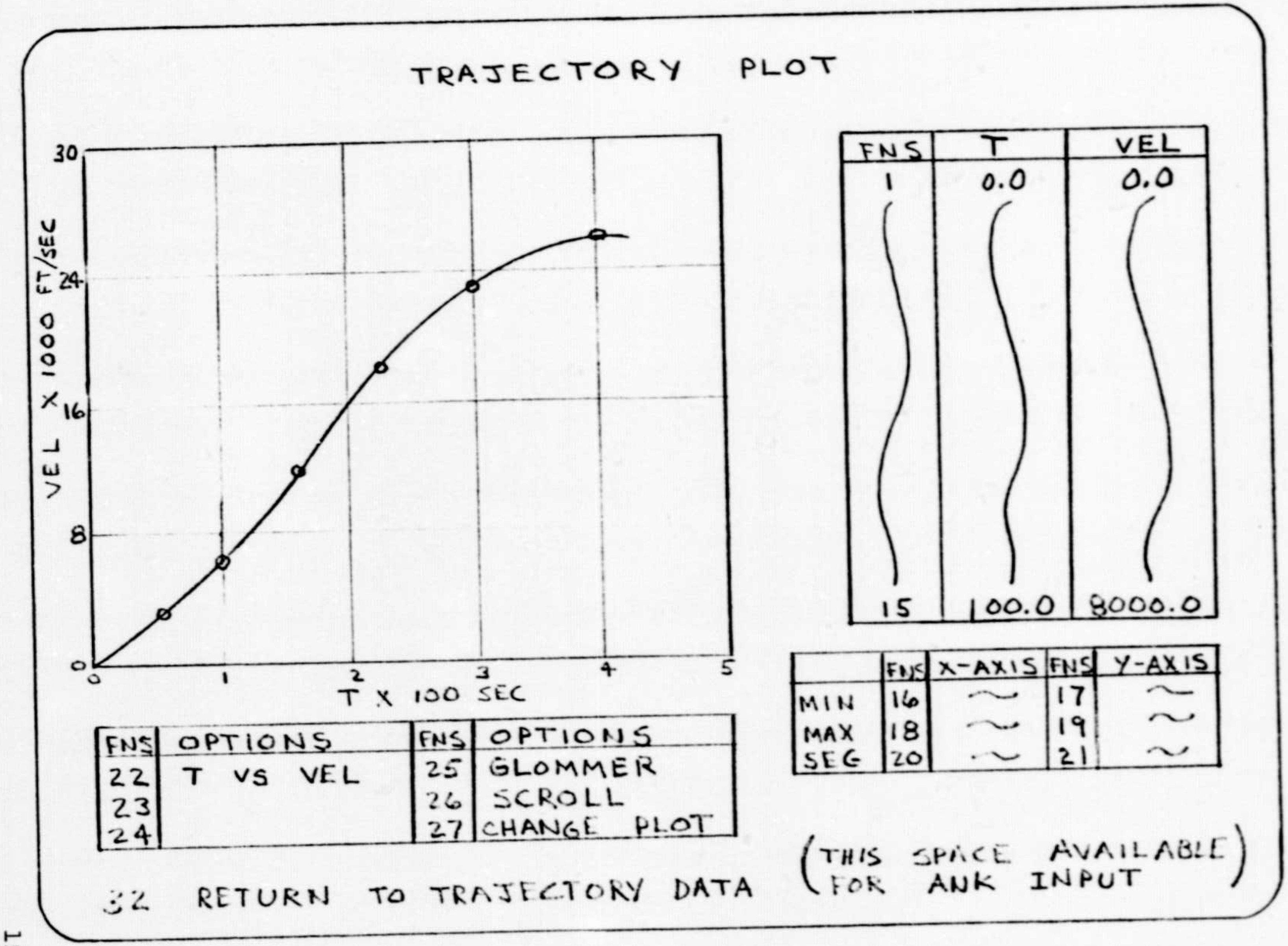


FIGURE 10 TRAJECTORY PLOT DISPLAY.

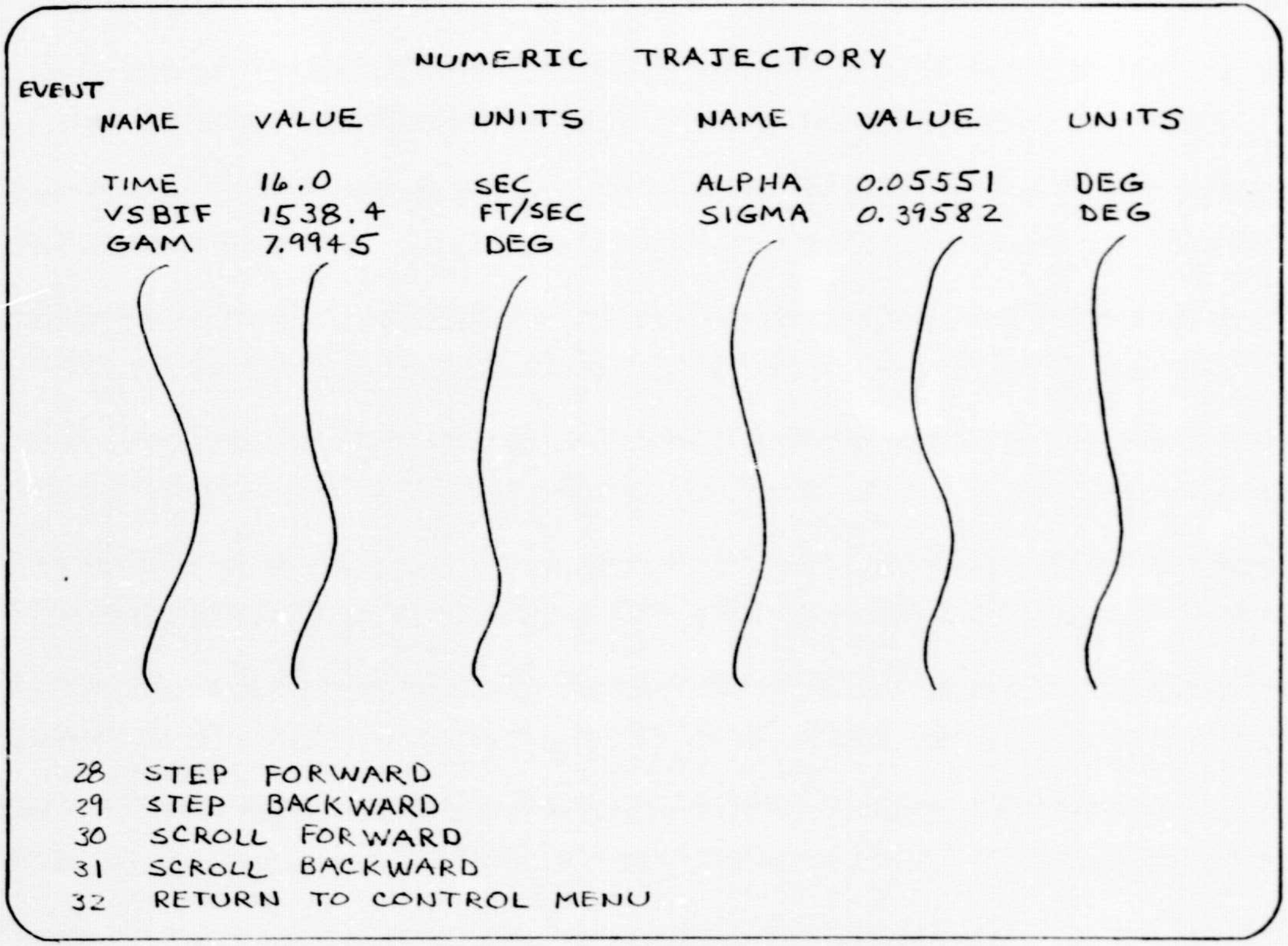


FIGURE 11 NUMERIC TRAJECTORY OUTPUT DISPLAY.

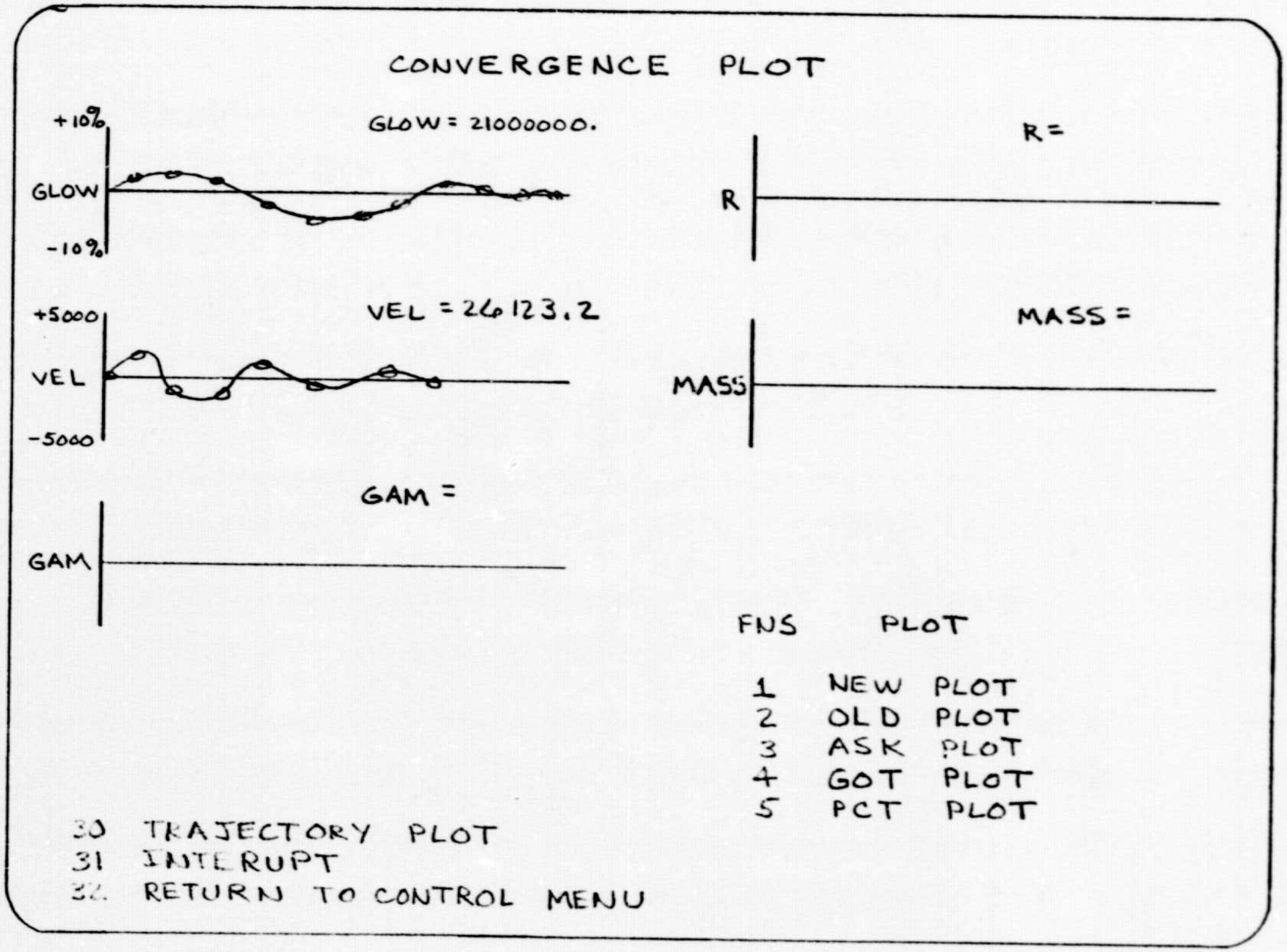


FIGURE 12 CONVERGENCE PLOT DISPLAY.

NUMERIC CONVERGENCE

	GLOW	VEL	GAM	R	MASS
OLD	.87809+06	-.59843+03	.26724+01	.38026+06	-.11031+04
NEW					
ASK					
GOT					
PCT					

DPAR	OLD PCON	NEW PCON	P SCALE
.36263504-01	.13493455-01	.17664223-02	.27370971+04

PSUBI	.47853699-02	.29480213+04	.82664805-11
WIBT	.10000000+01	.10000000+01	.10000000+01
X KAY	.10000000+01		
E1	.97337738+00		
DEL CHIP	.6504-02		

28 STEP FORWARD
29 STEP BACKWARD
30 SCROLL FORWARD
31 SCROLL BACKWARD
32 RETURN TO CONTROL MENU

FIGURE 13 NUMERIC CONVERGENCE OUTPUT DISPLAY.

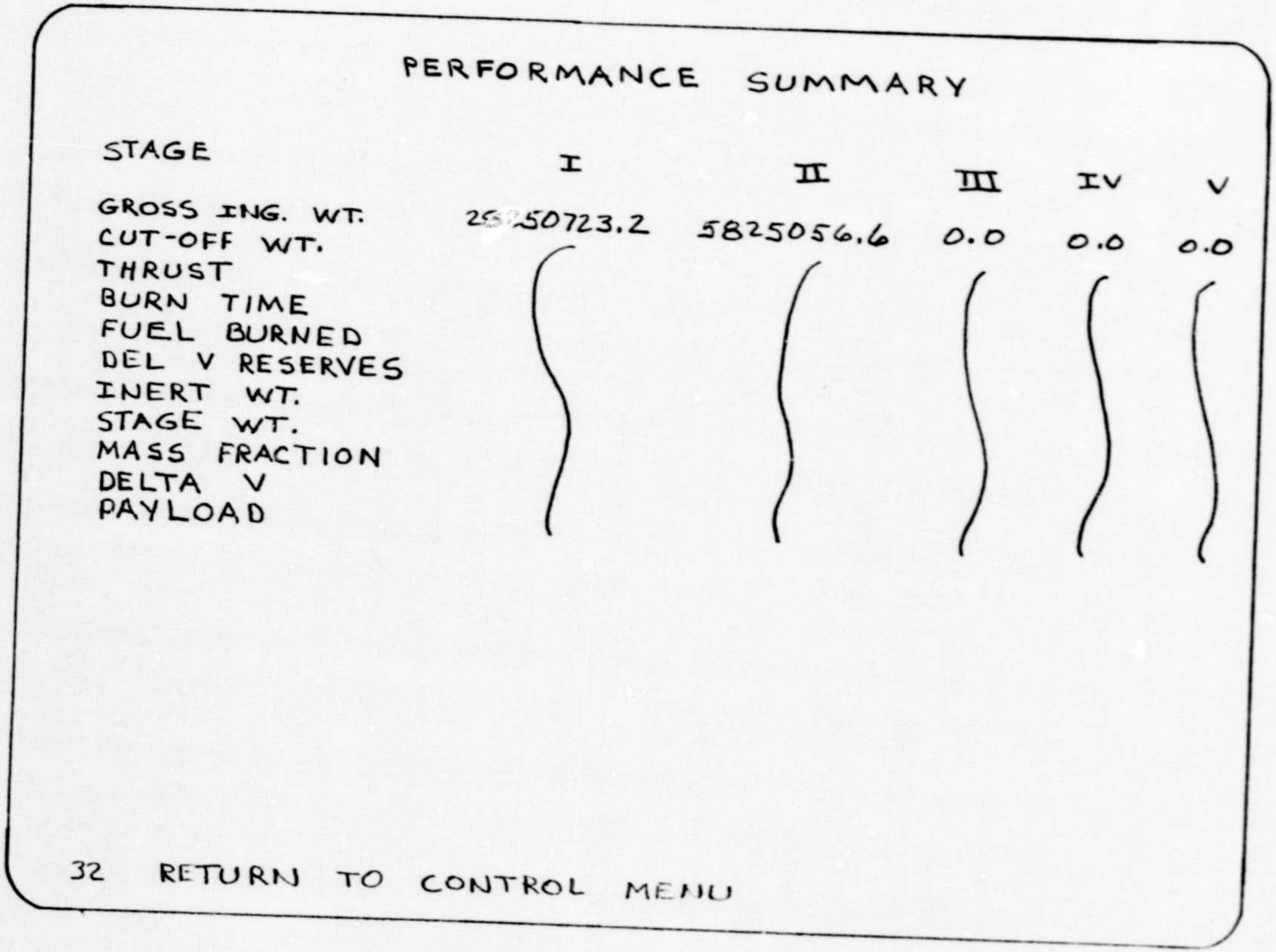


FIGURE 14 PERFORMANCE SUMMARY DISPLAY.

CONCLUSION

This preliminary design document serves as a foundation as to how the ROBDAC Program will be written. In no way should this document be taken as an official design criteria that can not be deviated from. As changes to this design are made or supplemental options added, they will be written in the form of a NASA memorandum so that a complete and final document can be written at the end of the project.

APPENDIX A - GRAPHIC SUPPORT COMMUNICATIONS PROGRAMS.

COMMUNICATION CONTROL SUBPROGRAMS

OUTLINE

The GS (Graphic Support) subprograms used for data transfer between application programs written on the host processor (1100 Series) and the display processor (AGS 100/300 Series) individually and used for program synchronization are described in this section.

The statuses of communication are classified as follows:

(1) The initiation of the communication is declared from the host and display processor sides respectively.

(2) The report data are sent from the host processor side to the display one or vice versa.

(3) The existence of communication information sent from the opposite side is checked.

(4) The report data sent from the opposite processor are received.

(5) The end of communication is declared in the host and display processor sides respectively.

GS subprograms used for communication are as follows:

(1) GSOPN ---- This declares the initiation of communication.

(2) GSCLS ---- This declares the end of communication.

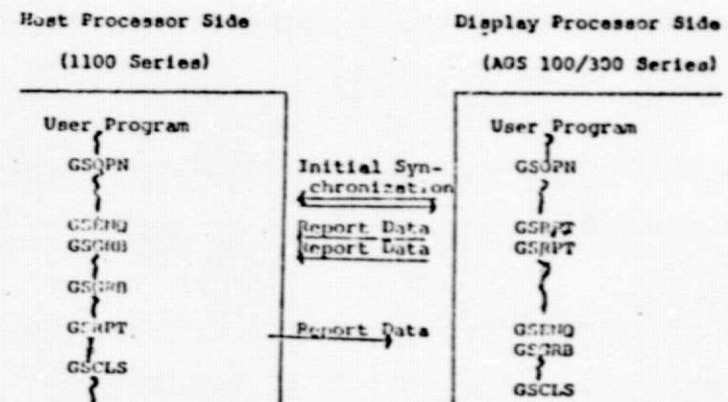
(3) GSRPT ---- This sends the report data.

(4) GSENQ ---- This checks the existence of the communication information sent from the opposite side.

(5) GSGRB ---- This receives the report data.

The way (specification) to use the GS subprogram on communication is common to both host and display processor sides.

The outline of communication is as follows:



Each GS subprogram is described in the text as follows:

(1) Purpose

The content of the job performed by each subprogram is described briefly.

Format

The general format to call each subprogram including arguments is described below:

(a) The underlined arguments indicate that values are returned to them by GS. Therefore, these arguments have to be variables or arrays.

(b) The arguments enclosed by brackets [ ] indicate that they can be omitted.

(c) The argument "$n" can be used only for the GS subprogram in the host processor side. The argument "$n" cannot be specified for the GS subprogram in the display processor side.

(3) General Description

The use of this job is described below.

INITIATION AND TERMINATION

GSOPN

(a) Purpose

This subprogram declares the initiation of the communication control GS subprograms.

(b) Format

CALL GSOPN ([$n,] istat)

where,

$n: When an error has occurred, the control is returned to the statement number specified here. When omitted, the control is returned to the next line.

istat: The error code caused in calling all GS subprograms is stored in this area. (Integer Variable)

(c) General Description

(i) This subprogram initializes the communication line and synchronizes the communication program with the opposite processor.

(ii) The user program must call GSOPN without fail prior to every use of the communication control GS subprograms (except for GSOPN itself.)

GSCLS

(1) Purpose

This subprogram declares the termination of the communication control GS subprograms.

(2) Format

CALL GSCLS ([$n])

(3) General Description

(a) This subprogram sends a termination message to the opposite processor and releases the communication line.

(b) After the completion of (a), the calling of other communication control GS subprograms than GSOPN cannot be accepted.

(c) The sent termination message is accepted when the opposite user program called the "GSENQ" or "GSCLS" subprogram.

GSRPT

(1) Purpose

This subprogram sends the report data.

(2) Format

CALL GSRPT ([$n,] last, data, count [, type])

where,

.last : Specify whether the data block sent now is the last one or not.

Ø : Data to send still remain.

1 : This data is the last.

.Data : Data to send. Constant, variable, or array.

.count: The amount of data to send.
(In case of numeric values, specify the number of words and in case of characters, specify the number of characters.)

.type: Specify the data type. (If omitted, the value Ø is assumed.)

Ø: Integer

1: Real number

2: Character

In case of integer, the absolute value must not be more than 536,870,911 $(= 2^{29} - 1)$.

(3) General Description

(a) The report data is transferred to the user program of the opposite processor in the sent order of the report data.

(b) Whether the report data is sent from the user program of the opposite processor or not is checked by the GSENQ subprogram.

(c) GSGRB is called to pick up the sent report data.

(d) One report data block is completed when the last data constructing the block ("last = 1) is transferred. Only completed report data are transferred to the user program.

GSENQ

(1) Purpose

This subprogram checks the existence of the communication information.

(2) Format

CALL GSENQ ([$a,] check, itype)

Where,

. check : Ø : When this value is specified, this subprogram waits until any information is sent from the opposite user program.

1 : When this specified, this subprogram examines whether the information is sent from the opposite user program.

. itype : Specify the integer variable indicating the sort of information sent from the opposite user program.

Ø : No information is being sent. (Only for "check" = 1)

1 : Report data

2 : The opposite user program terminates (The GSCLS subprogram was called.)

3 : The opposite user program is waiting (only on AGS side.)

(3) General Description

(a) This subprogram examines the sort of communication data sent by the opposite user program.

(b) When no communication data is sent:
If "check" = Ø, GS waits until the communication data is sent and it does not return control to the user program. If "check" = 1, GS returns the control to the user program immediately specifying "itype" = Ø.

(c) When the report data is sent:
" itype " is set to be 1. Then, the corresponding report data can be picked up by the GSGRB subprogram.

(d) When the call "check" = Ø was performed on the display processor side and the call was also performed on the host processor side, "itype" is set to 3 and immediately control is returned on the display processor side. The host processor continues to wait.

(e) After the report data (not the last, "last" = Ø) was sent, this subprogram cannot be called without sending the last report data ("last" = 1.)

(f) When both incomplete report data and "GSCLS" information sent from the opposite user program exist in calling this subprogram, only the latter is transferred, the former being missed.

(g) When "GSENQ" subprogram was called before picking out the report data sent from the opposite user program, the report data is lost.

(h) When the opposite user program called "GSCLS" subprogram,"itype" is set to be 2. When "itype" = 2 on the host processor side, the communication is automatically terminated, so that the user program does not have to call the "GSCLS" subprogram.

GSGRB

(1) Purpose

This subprogram receives report data.

(2) Format

CALL GSGRB ([$n,] array, size, icount, itype)

Where,

* array : The array which receives data.
* size : The size of the array.
* icount: The integer variable of the amount of data received(when this data is numeric, the number of words is returned and when characters, the number of characters is returned.)
  When the data overflows the array, the exact amount of data filling the array is stored, and a negative value is returned into "icount." The rest can be picked up by calling this subprogram again.
* itype: The integer variable of the data type.
  Ø : Integer
  1 : Real number
  2 : Characters
  3 : All report data has already been picked out.

(3) General Description

(a) Before picking out the report data by this subprogram, the user program has to make certain by the "GSENQ" subprogram that the report data has been sent.

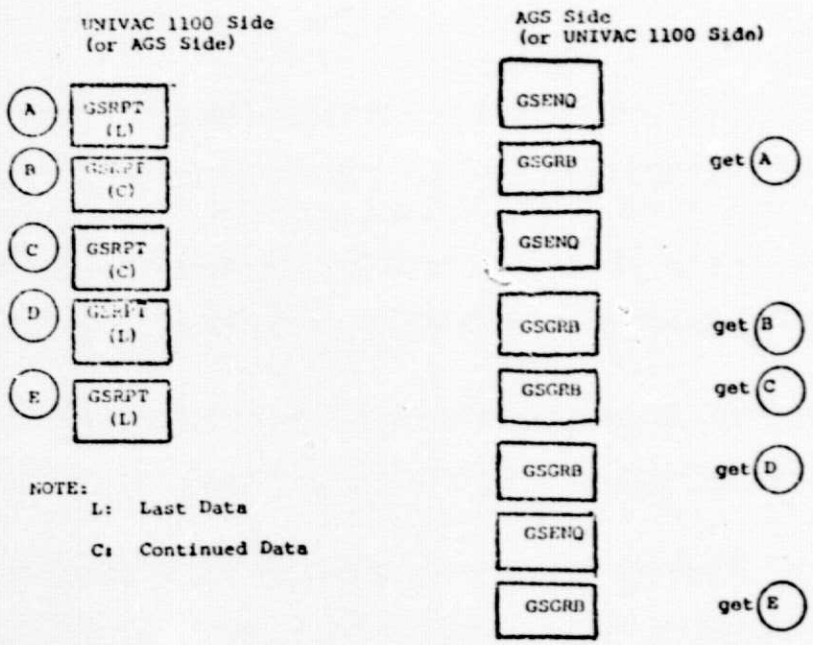
(b) When the "GSENQ" subprogram is called again without picking out data by "GSGRB" subprogram after making certain by the "GSENQ" subprogram that the report data has been sent, the report data will be missed.

(c) If the size of the array "array" presented is insufficient to store sent data (corresponding to GSRPT called once), the value of data amount with a minus sign actually stored in "array" will be returned to "icount". Then, the consecutive part of the previously picked up data can be picked up by calling this subprogram again.

SUPPLEMENT

Data Transfer

The relation of GSRPT, GSENQ, and GSGRB is illustrated and the transfer of communication data is indicated as follows:



NOTE:
L: Last Data
C: Continued Data

THE KINDS OF ERRORS AND ERROR PROCESSING

(1) AGS Side

The following two kinds of errors exist on the AGS Side.

(a) Status Error

When the user's calling condition is bad, this shall be called status error and a positive error code is given. (See the error code table in the next page.

(b) System Error

The error caused by trouble in the system or that generated depending on the communication processing shall be a system error.

* Status Error Processing

The error code being returned to the user, the GS CALL statement causing the error is skipped. This appears from the users viewpoint as if this GS subprogram was not called.

* System Error Processing

Following messages are outputted on the system console and the communication is terminated.

*** GSA SYSTEM ERROR ***

* ERROR ADDRESS = xxxxx

- Status Error Code Table (common to AGS and UNIVAC 1100 sides)

ERROR CODE	DESCRIPTION	ROUTINE NAME
4	GS has been already initiated.	GSOPN
10	There are too many arguments.	All Routines
11	Arguments are insufficient.	All Routines
36	The specification of the amount of data is not appropriate. (The amount of data $\leqq$ Ø)	GSRPT
63	The specification of the condition code is not appropriate. (The condition code $\neq$ Ø or 1)	GSRPT GSENQ
71	The specification of the size of the array to pick out data is not appropriate. (The size $\leqq$ 0)	GSGRB
72	There is no data to pick out. (all data have been already picked out.)	GSGRB
74	The specification of the data type is not appropriate. (Data type $\neq$ Ø, 1, or 2)	GSRPT
77	The report data sent previously was [illegible] completed. (The last report data [illegible] not sent.)	GSENQ

- Other Error Messages

(a) "GSOPN" is not called or resulted in an error, and any other GS subprogram was called.

PLEASE CALL GSOPN

(b) An error depending on the communication processing has occurred during the execution of the "GSOPN" subprogram.

' GSOPN ERROR ... GS CLOSE '

(2) UNIVAC 11ØØ Side

The following three kinds of errors exist on UNIVAC 11ØØ side.

(a) <u>Status Error</u>

When the user call condition is bad, this shall be the status error and the positive error code is given. (See the error code list.)

(b) <u>System Error</u>

This is the error caused by some trouble in the system or the error generated depending on the communication processing, and a negative error code is given.

(c) <u>Contingency Error</u>

An illegal hardware instruction or guard mode, etc.

shall cause a contingency error and a negative error code. (- 100) is given.

- Status Error Processing

  The error code is returned to the user and the GS subprogram causing the error is skipped. This seems to the user as if this GS subprogram was not called.

- System Error Processing

  The GS subprogram which caused the error is skipped and the "GSCLS" subprogram is called internally. (The user cannot call other subprograms than "GSOPN" hereafter.) Then, the error information in the following figure A is printed and the processing is continued.

- Contingency Error Processing

  The GS program which caused the error is skipped, "GSCLS" instruction is called internally. (The user cannot call other subprograms than "GSOPN" hereafter.) Then, the error information in the following figure B is printed and the processing is continued.

```
***  SYSTEM ERROR IN GSH        ***
       ERROR CODE       =  - BB
       COMMAND NAME     =  GSXXX
       W.B POINT        =  WWWW
* GSH *

     DUMP OF USER WORKING AREA
```

Figure A System Error Information

BB : System Error Code (Decimal Number)

GSXXX: GS Subprogram Name

WWWW : Walk Back Point (Octal Number)

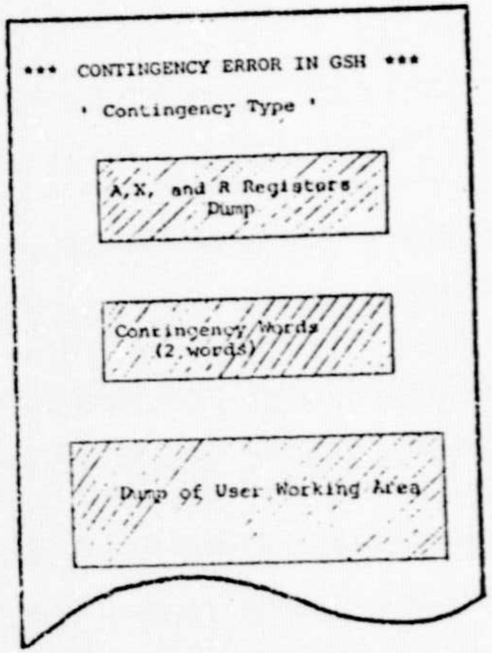


Figure B Contingency Error Information

'Contingency Type '

: Illegal Operation
: Guard Mode
: Floating Point Overflow
: Floating Point Underflow
: Divide Overflow
: Error Mode Entry

- System Error Code Table

CODE	DESCRIPTION
-1 ~ -63	The status error of IOG$ function
-64	The integer number overed 30 bits.
-67	An invalid message came from AGS side while GSOPN is executed.
-68	The address of the pool buffer is improper.
-69	An invalid message came from the AGS side while GSENQ is executed.
-70	Operation error of call queue stack
-71	Confliction of internal working parameters
-72	Initialization/termination error of IOG$
-100	Contingency error

- Other Error Messages

(1) Any other subprogram was called without calling "GSOPN."

** GSOPN HAS NOT BEEN CALLED **

(2) There is an error in "GSOPN" call.

** SYNTAX ERROR EXISTS IN GSOPN CALL **

(3) Printing for System/Contingency Error

(This will be printed just before printing each error information.)

--DEAD LOCK IN GSH : --BUT GRAPHICS --

--PROCESSING MAY BE RESUMED BY A --

--SUBSEQUENT CALL TO [GSOPN] --