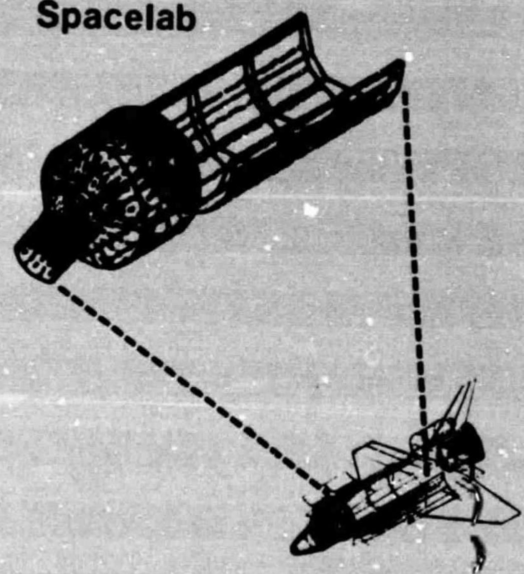
MCR-76-382
NAS8-31574

Final Report

September 1976

Payload/Orbiter Contamination Control Requirement Study

- COMPUTER INTERFACE STUDY



MARTIN MARIETTA

MCR-76-382
September 30, 1976

TECHNICAL REPORT

PAYLOAD/ORBITER CONTAMINATION CONTROL REQUIREMENT STUDY-COMPUTER INTERFACE STUDY

FINAL REPORT

CONTRACT NAS8-31574

AUTHORS

L. E. BAREISS
V. W. HOOPER
E. B. RESS
D. A. STRANGE

PREPARED FOR

GEORGE C. MARSHALL SPACE FLIGHT CENTER
MARSHALL SPACE FLIGHT CENTER, ALABAMA 35812

BY

MARTIN MARIETTA AEROSPACE, DENVER DIVISION
P. O. BOX 179, DENVER, COLORADO 80201

FOREWORD

This study was performed in order to determine the modifications necessary to convert the Martin Marietta Aerospace (MMA) Spacelab Contamination Computer Model, written for a CDC 6500 computer, for use on a Marshall Space Flight Center (MSFC) computer. Such a conversion is necessary if Spacelab contamination programs are to be processed locally on MSFC computers. In order to pursue the study, it was necessary to review the computer complement at MSFC and their computer usage processes to determine the most applicable and available computers for the purpose.

MSFC is presently in the process of revising their computer operations, access terminals, and control language and, because of the state of change, it is not presently possible to itemize completely all necessary details of the conversion requirements. However, no undue difficulties are foreseen. The new computer system at MSFC will be known as the Marshall Interactive Planning System (MIPS).

The UNIVAC 1108 computer was selected as the most applicable and available computer in the MSFC complement for processing the contamination program. Major differences between the CDC 6500 and UNIVAC 1108 are memory capacity, word length, and various specific functional capabilities. The CDC 6500 uses Fortran IV whereas, the UNIVAC 1108 uses Fortran V control language. None of the differences are such as to create any great difficulties in program conversion and a good programmer, familiar with the contamination program, should be able to process the necessary changes with minimum difficulty, knowing the differences.

The study includes a large table listing comparable characteristics of the two computers side by side so that a comprehensive comparison of computer capabilities can be made. From this table, a description and a smaller table of differences and program modification requirements has been developed. A small table listing some of the presently devised control language elements for the MIPS is also included in the study. The final section of the study lists the various Spacelab Contamination Computer Model routines and subroutines and program input support requirements.

MSFC plans to include approximately 50 of its most used programs on disc storage in the MIPS for rapid access when required. The contamination program is designed to accept input from a modified Thermal Radiation Analysis System (TRASYS) program which establishes the necessary geometrical configurations and locational relationships for contaminant interchange assessment. This modified program or one with similar output must be available and working in the MIPS in order to process all aspects of the Spacelab contamination program.

CONTENTS

1. INTRODUCTION

The purpose of this study effort is to conduct a preliminary assessment of the computer interface requirements of the Spacelab Configuration Contamination Computer Model to determine the compatibility of the program, as presently formatted, with the computer facilities at MSFC. This computer interface document sets forth the necessary Spacelab model modifications to be made if and when the program is transferred for utilization at MSFC.

This task required that the present computer complement at MSFC and future planning for it be determined and a comparison made of the programming differences between suitable MSFC computers and the CDC 6000 series computers for which the program is presently formatted at Martin Marietta Aerospace.

This report describes the MSFC computer facilities, and future plans for them, and discusses characteristics of the various computers as to availability and suitability for processing the contamination program. A listing of the CDC 6000 series and UNIVAC 1108 characteristics is presented so that programming requirements can be compared directly and differences noted.

This report is an update of interim report MCR 76-165 published in February 1976 under the same title and should be considered as current to date as based upon the available information on the MSFC computer facilities.

2. STUDY ACTIVITY STATUS

Time allotted to this study has been utilized in determining the present computer complement at MSFC, projected future changes, the suitability and availability of the various MSFC computers, and the basic differences between the applicable MSFC computers and the CDC 6000 computer for which the program is presently formatted. A summary of the findings of this study to date is presented in the following subsections.

2.1 Presently Available Computer Facilities at MSFC - Computers presently available at MSFC include two UNIVAC 1108's, one PDP 11/45, and several IBM 360's. In addition, lines are available to the Slidell facility where other UNIVAC 1108's and IBM models are located.

The basic MSFC computer for engineering problems is the UNIVAC 1108. The PDP 11/45 is also available for engineering problems but is a much smaller and slower computer. The IBM 360's are used primarily for Data Systems storage, processing, and retrieval and will not be available for engineering problem processing.

MSFC is presently in the process of making extensive changes in computer usage and operations. These changes are being incorporated to provide more effective use of the computers, more flexibility in peripheral instruments, and improved and time saving service to users. Previously, MSFC has operated in the Batch Processing Mode wherein programs were submitted and awaited their turn for processing by the computer operators. This was a time consuming process, requiring about a day from program submittal to the receipt of data. If programming errors were encountered, much longer delays were inherent. A Common Accessing Process was also used wherein a user requested routines or programs stored on tape reels in a storage vault. With this process, a computer operator was required to go to the vault, find the requested tape, and connect it into a computer before a program could proceed.

2.2 Improved System - The revised system is called the Marshall Interactive Planning System (MIPS). It involves the installation of multiple remote access terminals having graphic display capability, disc storage of programs within the computer, and linking of the separate computers into a common system.

The Techtronix 4014 will be used for all the remote access terminals. These units will permit direct access by programmers at many convenient locations. Their graphic display capability permits computed data to be plotted automatically at the remote terminal for those desiring such presentation. This facility eliminates the onerous task of plotting tabular data by hand and speeds up the processing of trial and error problems by permitting the terminal user to modify variables rapidly until a desired result is obtained.

It is planned to include approximately 50 of the most used programs in the disc storage system. This availability will eliminate much of the delay encountered with the previous Common Accessing Process.

Linking the computers into a common system will more nearly equalize computer usage and reduce access delays such as would be encountered when a programmer addresses a computer that is down for repairs or servicing.

2.3 Availability and Acceptability of MSFC Computers for Processing Spacelab Configuration Contamination Programs - Only the UNIVAC 1108's and the PDP 11/45 will be made available for processing programs such as that required for the Spacelab contamination investigation. The UNIVAC 1108 is completely suitable. The PDP 11/45 has limited capabilities.

2.3.1 PDP 11/45 Acceptability - Disadvantages of the PDP 11/45 for processing the Spacelab configuration contamination program are:

a. short word length;
b. small memory capacity;
c. low speed processing; and
d. reduced precision.

The normal word length used with PDP 11/45 programs consists of 16 binary bits. A double precision process can be used wherein 32 bit words are available. However, use of such a process would automatically reduce the memory capacity to one-half that available with 16 bit words. Capacity with 16 bit words is 124,000 words of addressable memory space. The Spacelab contamination program as presently formatted uses approximately 100,000 sixty bit words. Although it is possible to reformat the contamination program to fit within the limited PDP 11/45 capacity, the end result would be a much more simplified model with limited flexibility and a considerable reduction of accuracy. It would also require a considerable amount of effort and time. Furthermore, the program would require processing in a series of incremental steps, one step at a time. Add to this the fact that the speed of the PDP 11/45 is only about one-hundredth that of the UNIVAC 1108 and the indications are that the PDP 11/45 is not an acceptable system for processing the contamination program. The UNIVAC 1108 is comparable in speed to the CDC 6500.

2.3.2 UNIVAC 1108 Adaptation Requirements - In order to convert the existing CDC program for use on the UNIVAC, differences in source programs between the two computer systems must be considered. The two compilers accept source programs that differ in at least three ways. These are that:

a. there are differences in permissible number sizes;
b. there are some features of FORTRAN that are not available in both implementations; and
c. some implementations go beyond the standard features of FORTRAN.

Table I is a listing of the characteristics of the two systems. The comparable characteristics are presented side by side so that differences can be identified more easily. A program compatible with both systems can then be written by noting the similarities between the two systems or the CDC program can be maintained as is and a revised program can be written for the UNIVAC considering its specific characteristics when and if it is requested by MSFC. The latter option is probably the most logical approach from a convenience standpoint in that current programming methods for the ongoing modeling activities would not require modification to a new system.

As indicated by the comparison table, presently available cards and tapes generated for use with the CDC computer are not directly compatible with the UNIVAC 1108 requirements and reformatting will be necessary. However, no untoward problems have been indicated by the comparison and reformatting of the general programming processes should require only a relatively short time. The specific translational requirements are presented in sections 3 and 4. Plotting, on the other hand, will require a complete rewrite, taking into consideration the specific software packages and peripherals made available for that purpose at MSFC.

TABLE I CHARACTERISTICS OF CDC 6000 SERIES AND UNIVAC 1108 COMPUTERS

CDC 6000 SERIES	UNIVAC 1108
FORTRAN IV	FORTRAN V
1) <u>VARIABLE AND FUNCTION TYPES</u>	1) <u>VARIABLE AND FUNCTION TYPES</u>
EXPLICIT	EXPLICIT
INTEGER REAL DOUBLE PRECISION COMPLEX LOGICAL	EXPLICIT REAL DOUBLE PRECISION COMPLEX LOGICAL
DOUBLE	NA*
IMPLICIT	IMPLICIT
NA*	IMPLICIT type (a_1,a_2,...),..., type (a_1,a_2,...) type: INTEGER,REAL,COMPLEX,LOGICAL INTEGER (standard 4) REAL (standard 8) COMPLEX (standard 8) LOGICAL (standard 4) type can be double precision. a_1,a_2,... single alphabetic characters or a range of characters
	Example: IMPLICIT REAL (A-H,O-Z,$), INTEGER(I-N)

*NA = Not Applicable

TABLE I (continued)

CDC 6000 SERIES	UNIVAC 1108
2) <u>CONTROL STATEMENTS</u>	2) <u>CONTROL STATEMENTS</u>
GO TO n	GO TO n
ASSIGN n TO i	ASSIGN n TO i
GO TO i, $(m_1,\ldots,m_i)$	GO TO i, $(m_1,\ldots,m_i)$
GO TO i	GO TO i
GO TO $(m_1,m_2,\ldots,m_i)$,i	GO TO $(m_1,m_2,\ldots,m_i)$,i
IF (a) $m_1,m_2,\ldots,m_3$	IF (a) $M_1,m_2,\ldots,m_3$
IF (a) S	IF (a) S
IF (a) n_1,n_2 TWO BRANCH LOGICAL IF	NA
DO ni = m_1,m_2,m_3	DO n i = m_1,m_2,m_3
CONTINUE	CONTINUE
PAUSE	PAUSE
IF (a) m_1,m_2 TWO BRANCH ARITHMETIC IF	IF (a) m_1, m_2
STOP	STOP
END	END
Note: a is a logical variable	Note: a is a logical variable
Example:	Example:
ASSIGN 64 to J GO TO J GO TO J, (10,13,25,64,83) PAUSE n STOP n n is a string of 1 to 5 octal digits	ASSIGN 64 TO J GO TO J GO TO J,(10,13,25,64,83) PAUSE n STOP n n is a string of 1 to 6 alphanumeric characters
END name name is the name of the program or subprogram which it terminates, and is ignored by the compiler	NA
IF (1)n_1,n_2 two branch logical if.	
NA	i END i is a statement number; it may be referenced by a control statement.

TABLE I (continued)

CDC 6000 SERIES	UNIVAC 1108
3) SUBPROGRAM STATEMENTS	3) SUBPROGRAM STATEMENTS
Statement function: name $(p_1,\ldots,p_n)$=expression reference, name $(a_1,\ldots,a_n)$	Statement function: name $(p_1,\ldots,p_n)$=expression reference, name $(a_1,\ldots,a_n)$
Subroutine subprograms: SUBROUTINE name $(p_1,\ldots,p_n)$ SUBROUTINE name reference CALL SUBROUTINE name $(a_1,\ldots,a_n)$ CALL SUBROUTINE name	Subroutine subprograms: SUBROUTINE name $(p_1,\ldots,p_n)$ SUBROUTINE name reference CALL SUBROUTINE name $(a_1,\ldots,a_n)$ CALL SUBROUTINE name
Function subprograms: type FUNCTION name $(p_1,\ldots,p_n)$ type FUNCTION name reference, name $(p_1,\ldots,p_n)$ name	Function subprograms: type FUNCTION name $(p_1,\ldots,p_n)$ type FUNCTION name reference, name $(p_1,\ldots,p_n)$ name
Type is REAL, INTEGER, DOUBLE PRECISION COMPLEX, LOGICAL. When type is omitted, the mode is determined by the first character of name. EXTERNAL $name_1$, $name_2$,... RETURN	Type is REAL, INTEGER, DOUBLE PRECISION COMPLEX, LOGICAL. When type is omitted, the mode is determined by the first character of name. EXTERNAL $name_1$, $name_2$,... RETURN
PROGRAM name $(f_1,\ldots,f_n)$ The parameter f_i must be the names of all input/output files required by the main program and its subprograms.	NA
ENTRY name The formal parameters, if any, are the same as those with the FUNCTION or SUBROUTINE statement, and do not appear with the ENTRY statement.	ENTRY name $(p_1,\ldots,p_n)$ p_i are the arguments corresponding to an actual argument in a CALL statement or in a function reference. Compatible with IBM SYSTEM 360.
BLOCK DATA	BLOCK DATA In order to collect a block data subprogram as part of a program for execution, the block data subprogram must be referenced.

TABLE I (continued)

CDC 6000 SERIES	UNIVAC 1108
NA	DEFINE name = expression DEFINE NAME $(p_1,\ldots,p_n)$ = expression reference, name $(a_1,\ldots,a_n)$ a DEFINE procedure generates inline code when it is referenced. DEFINE is analogous to a statement function.
NA	RETURN k k is an integer constant, a parameter variable, or an integer variable. Example: Calling Program / Subprogram: Calling Program: . . . 10 CALL SUB (A,$ or & 30) 20 Y = A+B . . 30 Y = A+C . . Subprogram: SUBROUTINE SUB (X,$ or *) . . . 100 IF (M) 200, 300, 200 200 RETURN 300 RETURN 2 END
NA	Function Subprograms: Type FUNCTION name *S$(p_1,\ldots,p_n)$ Univac in order to be compatible with IBM SYSTEM 360, will accept the above FUNCTION statement
NA	ABNORMAL In order to produce correct and efficient code, the compiler recognizes common subexpressions. That is, a subexpression is

TABLE I (continued)

CDC 6000 SERIES	UNIVAC 1108
	evaluated only once if the variables contained in it are not altered. These subexpressions contain variables in common. If a COMMON variable is altered by a function, the function should be declared ABNORMAL in order for the compiler to generate correct code.
NA	INTERNAL FUNCTIONS AND SUBROUTINES Internal subprograms are compiled in conjunction with a main program, an external function subprogram, or an external subroutine subprogram. An internal subprogram may be referenced from any part of its program unit except from its own body. Example:

```
     .
     .
     .
Y=A+B
CALL SAM (X)
     .
     .
     .
RETURN
SUBROUTINE SAM (T)
Z = Y
T = Z+A
RETURN
END
```

TABLE I (continued)

CDC 6000 SERIES	UNIVAC 1108
4) <u>ALLOCATION STATEMENTS</u>	4) <u>ALLOCATION STATEMENTS</u>
COMPLEX list DOUBLE PRECISION list REAL list INTEGER list LOGICAL list DIMENSION $v_1,v_2,\ldots,v_n$ COMMON/l_1/$list_1$/l_2/$list_2$...	COMPLEX list DOUBLE PRECISION list REAL list INTEGER list LOGICAL list DIMENSION $v_1,v_2,\ldots,v_n$ COMMON/l_1/$list_1$/l_2/$list_2$...
/l_i/... represent optional names consisting of 1 - 6 alphanumeric characters, the first of which is alphabetic.	/l_i/...represent optional names consisting of 1 - 6 alphanumeric characters, the first of which is alphabetic.
EQUIVALENCE (a_1,b_1,..), ($a_2 b_2$,...),... DATA $list_1$/$a_1,\ldots,a_n$/, $list_2$/$b_1,\ldots,b_n$/,...	EQUIVALENCE (a_1,b_1,...), ($a_2 b_2$,...),... DATA $list_1$/$a_1,\ldots,a_n$/, $list_2$/$b_1,\ldots,b_n$/,...
COMMON/l_1/$list_1$/l_2/$list_2$ /l_i/... represent optional names consisting of 1 - 7 alphanumeric characters. They may be all numeric.	NA
DATA ($list_1$=c ,...,c_n), ($list_2$= $d_1,\ldots,d_n$),...	NA
DOUBLE list	
NA	Type v_1/l_1/v_2/l_2/... v_i represents a list of variables l_i represents a literal list.
	DIMENSION v_1/l_1/v_2/l_2... v_i represents a list of array declarations. l_i represents a literal list.

TABLE I (continued)

CDC 6000 SERIES	UNIVAC 1108
5) <u>REPLACEMENT STATEMENTS</u>	5) <u>REPLACEMENT STATMENTS</u>
a = Arithmetic expression l = Logical expression	a = Arithmetic expression l = Logical expression
m = Masking expression	NA
The masking expression is a generalized form of logical expression in which the variables may be types other than logical.	
Multiple Replacement Statement A=B=C=expression	
Multiple Statement Cards A=expression $ B=expression	
6) <u>FORMAT STATEMENT AND SPECIFICATIONS</u>	6) <u>FORMAT STATEMENT AND SPECIFICATIONS</u>
FORMAT ($spec_1$,...,$spec_n$) Where $spec_i$ =	FORMAT ($spec_1$,...,$spec_n$) Where $spec_i$ =
EW.d Single precision floating point with exponent	EW.d Single precision floating point with exponent
FW.d Single precision floating point without exponent	FW.d Single precision floating point without exponent
DW.d Double precision floating point with exponent	DW.d Double precision floating point with exponent
GW.d Single precision floating point with or without exponent	GW.d Single precision floating point with or without exponent
IW Decimal integer	IW Decimal integer
Aw Alphanumeric, left justified, with trailing blanks	Aw Alphanumeric, left justified, with trailing blanks
Lw Logical	Lw Logical
nP Scaling factor	nP Scaling factor
Complex values are converted by a pair of consecutive EW.D or FW.D.	Complex values are converted by a pair of consecutive EW.d or FW.d.
wX Intra-line spacing	wX Intra-line spacing
wH Transmits literal data	wH Transmits literal data

TABLE I (continued)

CDC 6000 SERIES		UNIVAC 1108	
Rw	Alphanumeric, right justified, leading zeros	Rw	Alphanumeric, right justified
...	Transmits literal data	'...'	Transmits literal data
Ow	Octal integer	NA	
NA		Tw	Indicates the position in a FORTRAN record where transfer of data is to start. Complex values are converted by a pair of E, F, or G format codes.
7)	<u>PRINTER CARRIAGE CONTROL</u>	7)	<u>PRINTER CARRIAGE CONTROL</u>
0	Double space after printing	0	Double space after printing
1	Eject page before printing	1	Eject page before printing
+	Suppress spacing before printing	+	Suppress spacing before printing
blank	Single space after printing	blank	Single space after printing
8)	<u>INPUT/OUTPUT AND DATA TRANSMISSION</u>	8)	<u>INPUT/OUTPUT AND DATA TRANSMISSION</u>
	READ n, list		READ n, list
	PRINT n, list		PRINT n, list
	PUNCH n, list		PUNCH n, list
	READ (i,n) list		READ (i,n) list
	WRITE (i,n) list		WRITE (i,n) list
	READ (i) list		READ (i) list
	WRITE (i) list		WRITE (i) list
	END FILE i		END FILE i
	REWIND i		REWIND i
	BACKSPACE i		BACKSPACE i
	NAMELIST/x/a,b,...,c/y/d,e, ...f...		NAMELIST/x/a,b,...,c/y/d,e, ...f...
	READ (i,x) - namelist read		READ (i,x) - namelist read
	WRITE (i,x)-namelist write		WRITE (i,x)-namelist write

TABLE I (continued)

CDC 6000 SERIES	UNIVAC 1108
BUFFER IN (i,m) list	READ(a,b,END=c,ERR=d) list END=c is optional, transfer to c encountering the end of the data set ERR=d is optional, transfer to d encountering error condition in data transfer
BUFFER OUT (i,m) list	WRITE (a,b,END=c,ERR=d) list
ENCODE (c,n,v) list DECODE (c,n,v) list	ENCODE (v,n) list DECODE (v,n) list

IF (EOF,i) n_1,n_2
IF(ENDFILE,i)n_1,n_2
IF (UNIT,i)n_1,n_2,n_3,n_4
	NA

NA	DEFINE FILE $a_1(m_1,r_1,f_1,v_1),\ldots,a_n(m_n,r_n,f_n,v_n)$ describes data set used during a direct access Input/out operation a - data set reference number. m - number of records in a. r - record size maximum. f - format control. v - associated variable.
NA	READ(a'r,b,ERR=d) list a - data set reference number followed by an apostrophe. r - integer, relative position of record in data set. b - format statement number, operational
NA	ERR = d, same as above,
NA	WRITE (a'r,b) list
NA	FIND(a'r) finds next input record while present record is being processed.

TABLE I (continued)

CDC 6000 SERIES	UNIVAC 1108
9) SUBSCRIPTS	9) SUBSCRIPTS
$A(i_1,\ldots,i_n)$ $1 \leq n \leq 3$ i may be: integer constant simple integer variable simple integer arithmetic expression	$a(i_1,\ldots,i_n)$ $1 \leq n \leq 3$ i may be: integer constant simple integer variable simple integer arithmetic expression
NA	$A(i_1,\ldots,i_n)$ $1 \leq n \leq 7$ i as above
10) FORTRAN DECK STRUCTURE	10) FORTRAN DECK STRUCTURE
(1) Program declaration statements (a) PROGRAM (b) FUNCTION (c) SUBROUTINE (d) BLOCK DATA (2) Type statements (3) DIMENSION statements (4) COMMON statements (5) EQUIVALENCE statements (6) DATA statements (7) NAMELIST statements (8) EXTERNAL statements (9) Executable statements (10) FORMAT statements (11) END	(1) Program declaration statements (a) PROGRAM (b) FUNCTION (c) SUBROUTINE (d) BLOCK DATA (2) Type statements (3) DIMENSION statements (4) COMMON statements (5) EQUIVALENCE statements (6) DATA statements (7) NAMELIST statements (8) EXTERNAL statements (9) Executable statements (10) FORMAT statements (11) END

TABLE I (continued)

11) FORTRAN CONSTANTS

	CDC 6000 SERIES	UNIVAC 1108	EXAMPLES
Integer	$n_1n_2..n_m$ $1 \leq m \leq 18$	$n_1n_2..n_m$ $1 \leq m \leq 11$	for 2, m=1 for 247, m=1
Octal	$On_1..n_m$ $6 \leq m \leq 20$ $n_1..n_mB$ $1 \leq m \leq 20$	$On_1..n_m$ $1 \leq m \leq 12$	0231461 777000B
Real	$n_1..n_mE\pm exp_{10}$ $1 \leq m \leq 15$	$n_1..n_mE\pm exp_{10}$ $1 \leq m \leq 9$	3.14 .0749 3.14E05
Double Precision	$n_1..n_mD\pm exp_{10}$ $1 \leq m \leq 18$	$n_1..n_mD\pm exp_{10}$ $1 \leq m \leq 18$	3.1415D0 -16.9D+19
Complex	(r_1,r_2) r_1=real r_2=imaginary	(r_1,r_2) r_1=real r_2=imaginary	(1.,6.55) (0.,-1.) (-15.,16.7)
Literal	$nHf_1..f_n$ replacement: $1 \leq n \leq 10$ Format: *GOGETIT* $1 \leq n \leq 136$ data statement $1 \leq n \leq 1307$,	$nHf_1..f_n$ all modes: $1 \leq n \leq$ unlimited $'f_1..f_n'$ $1 \leq n \leq$ unlimited	7HGOGETIT 'GOGETIT'
Logical	.TRUE..T. .FALSE..F.	.TRUE. .FALSE.	

TABLE I (continued)

12) FORTRAN VARIABLES

	CDC 6000 SERIES	UNIVAC 1108	EXAMPLES
Integer	$a_1a_2..a_m$ $1 \leqslant m \leqslant 7$ a_1 I to N	$a_1a_2..a_m$ $1 \leqslant m \leqslant 6$ a_1 I to N	N L2504 M58
	All a's past a_1 alphanumeric		
Real	$a_1a_2..a_m$ $1 \leqslant m \leqslant 7$	$a_1a_2..a_m$ $1 \leqslant m \leqslant 6$	VECTOR SPOILS
	a_1 alphabetic other than I to N all a's past a_1 alphanumeric		
	Variables defined by type declarations begin with any letter.		

TABLE I (continued)

13) FORTRAN FUNCTIONS

	CDC 6000 SERIES	UNIVAC 1108
Exponential	EXP(x)	EXP(x)
	DEXP(d)	DEXP(d)
	CEXP(c)	CEXP(c)
Natural	ALOG(x)	ALOG(x)
Logarithm	DLOG(d)	DLOG(d)
	CLOG(c)	CLOG(c)
Common Logarithm	ALOG10(x)	ALOG10(x)
	DLOG10(D)	DLOG10(d)
Arcsine	ASIN(x)	ASIN(x)
		DASIN(d)
Arccosine	ACOS(x)	ACOS(x)
		DACOS(d)
Arctangent	ATAN(x)	ATAN(x)
	ATAN2(x_1,x_2)	ATAN2(x_1,x_2)
	DATAN2(d_1,d_2)	DATAN2(d_1,d_2)
Sine	SIN(x)	SIN(x)
	DSIN(d)	DSIN(d)
	CSIN(c)	CSIN(c)

Note: in the arguments

i=integer
x=real
d=double precision
c=complex

TABLE I (continued)

	CDC 6000 SERIES	UNIVAC 1108
Cosine	COS(x) DCOS(d) CCOS(c)	COS(x) DCOS(d) CCOS(c)
Tangent	TAN(x)	TAN(x) DTAN(d) CTAN(c)
Square Root	SQRT(x) DSQRT(d) CSQRT(c)	SQRT(x) DSQRT(d) CSQRT(c)
Cube Root	NA	CBRT(x) DCBRT(d) CCBRT(c)
Hyperbolic Sine	NA	SINH(x) DSINH(d) CSINH(c)
Hyperbolic Cosine	NA	COSH(x) DCOSH(d) CCOSH(c)
Hyperbolic Tangent	TANH(x)	TANH(x) DTANH(d) CTANH(C)

TABLE I (continued)

	CDC 6000 SERIES	UNIVAC 1108
Modular Arithmetic	AMOD(x_1,x_2) MOD(i_1,i_2) DMOD(d_1,d_2)	AMOD(x_1,x_2) MOD(i_1,i_2) DMOD(d_1,d_2)
Absolute Value	ABS(x) DABS(d) CAGS(c) IABS(i)	ABS(x) DABS(d) CABS(c) IABS(i)
Truncation	AINT(x) INT(x) IDINT(d)	AINT(x) INT(x) IDINT(d) DINT(d)
Largest Value	AMAXO(i_1,i_2,...) AMAX1(x_1,x_2,...) MAXO(i_1,i_2,...) MAX1(x_1,x_2,...) DMAX1(d_1,d_2,...)	AMAXO(i_1,i_2,...) AMAX1(x_1,x_2,...) MAXO(i_1,i_2,...) MAX1(x_1,x_2,...) DMAX1(d_1,d_2,...)
Smallest Value	AMINO(i_1,i_2,...) AMIN1(x_1,x_2,...) MINO(i_1,i_2,...) MIN1(x_1,x_2,...) DMIN1(d_1,d_2,...)	AMINO(i_1,i_2,...) AMIN1(x_1,x_2,...) MINO(i_1,i_2,...) MIN1(x_1,x_2,...) DMIN1(d_1,d_2,...)
Float	FLOAT(i)	FLOAT(i)

TABLE I (continued)

	CDC 6000 SERIES	UNIVAC 1108
Fix	IFIX(x)	IFIX(x)
Transfer of Sign	SIGN(x_1,x_2) ISIGN(i_1,i_2) DSIGN(d_1,d_2)	SIGN(x_1,x_2) ISIGN(i_1,i_2) DSIGN(d_1,d_2)
Positive Difference	DIM(x_1,x_2) IDIM(i_1,i_2)	DIM(x_1,x_2) IDIM(i_1,i_2)
Significant Part of DP Argument	SNGL(d)	SNGL(d)
Real Part of Complex	REAL(c)	REAL(c)
Imaginary Part of Complex	AIMAG(c)	AIMAG(c)
Real to DP	DBLE(x)	DBLE(x)
Real to Complex	CMPLX(x_1,x_2)	CMPLX(x_1,x_2)
Conjugate	CONJG(c)	CONJG(c)
Logical Product	AND(x_1,...,x_n)	AND(x_1,x_2)
Logical Sum	OR(x_1,..,x_n)	OR(x_1,x_2)
Complement	COMPL(x)	COMPL(x)
Number of Words from Unit i	LENGTH(i)	NA

TABLE I (continued)

	CDC 6000 SERIES	UNIVAC 1108
Random Number	RANF(x)	NA
Time from Dead Start	SECOND(i)	NA

TABLE I (continued)

14) CHARACTER CODES

CHARACTER		COMPUTER CODE		PUNCH CARD	
BCD	EBCDIC	CDC (octal)	UNIVAC (octal)	CDC	UNIVAC
A	A	01	06	12-1	12-1
B	B	02	07	12-2	12-2
C	C	03	10	12-3	12-3
D	D	04	11	12-4	12-4
E	E	05	12	12-5	12-5
F	F	06	13	12-6	12-6
G	G	07	14	12-7	12-7
H	H	10	15	12-8	12-8
I	I	11	16	12-9	12-9
J	J	12	17	11-1	11-1
K	K	13	20	11-2	11-2
L	L	14	21	11-3	11-3
M	M	15	22	11-4	11-4
N	N	16	23	11-5	11-5
O	O	17	24	11-6	11-6
P	P	20	25	11-7	11-7
Q	Q	21	26	11-8	11-8
R	R	22	27	11-9	11-9
S	S	23	30	0-2	0-2
T	T	24	31	0-3	0-3
U	U	25	32	0-4	0-4
V	V	26	33	0-5	0-5
W	W	27	34	0-6	0-6
X	X	30	35	0-7	0-7
Y	Y	31	36	0-8	0-8
Z	Z	32	37	0-9	0-9
0	0	33	60	0	0
1	1	34	61	1	1
2	2	35	62	2	2
3	3	36	63	3	3
4	4	37	64	4	4
5	5	40	65	5	5
6	6	41	66	6	6
7	7	42	67	7	7
8	8	43	70	8	8
9	9	44	71	9	9
/	/	50	74	0-1	0-1
+	&	45	42	12	12
-		46	41	11	11
blank	blank	55	05	space	space

TABLE I (continued)

CHARACTER		COMPUTER CODE		PUNCH CARD	
BCD	EBCDIC	CDC (octal)	UNIVAC (octal)	CDC	UNIVAC
.	.	57	75	12-8-3	12-8-3
)	<	52	40	12-8-4	12-8-4
$	$	53	47	11-8-3	11-8-3
*	*	47	50	11-8-4	11-8-4
,	,	56	56	0-8-3	0-8-3
(	%	51	51	0-8-4	0-8-4
=	#	54	44	8-3	8-3
	¢	NA	NA	NA	NA
[	(	61	01	8-7	12-8-5
<	+	73	43	12-0	12-8-6
]	)	62	02	0-8-2	11-8-5
;	;	77	73	12-8-7	11-8-6
Δ	→	NA	04	NA	11-8-7
\	>	NA	57	NA	0-8-6
@	@	NA	00	NA	8-7
:	/	63	53	8-2	8-5
>	=	72	45	11-8-7	8-6
?		NA	54	NA	12-0
!		NA	55	NA	11-0
≠		NA	77	NA	0-8-2
&		NA	46	NA	8-2
/		NA	72	NA	8-4
#		NA	03	NA	12-8-7
≡		60	NA	0-8-6	NA
≠		64	NA	8-4	NA
→		65	NA	0-8-5	NA
∨		66	NA	11-0	NA
∧		67	NA	0-8-7	NA
↑		70	NA	11-8-[illegible]	NA
↓		71	NA	11-8-6	NA
≤		74	NA	8-5	NA
≥		75	NA	12-8-5	NA
¬		76	NA	12-8-6	NA

TABLE I (continued)

15) FLOATING POINT WORD STRUCTURE

CDC 6000 SERIES--

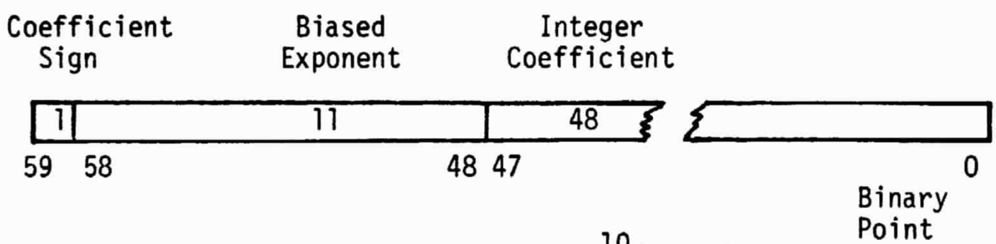
Floating point arithmetic takes advantage of the ability to express a number with the general expression KB^n, where:

K = coefficient

B = base number

n = exponent or power to which the base number is raised

The base number is constant (2) for binary-coded quantities and is not included in the general format. The 60-bit floating-point format is shown below. The binary point is considered to be to the right of the coefficient, thereby providing a 48-bit integer coefficient, the equivalent of about 14 decimal digits. The sign of the coefficient is carried in the highest order bit of the packed word. Negative numbers are represented in one's complement notation.



The 11-bit exponent carries a bias of $2^{10}(2000_8)$.

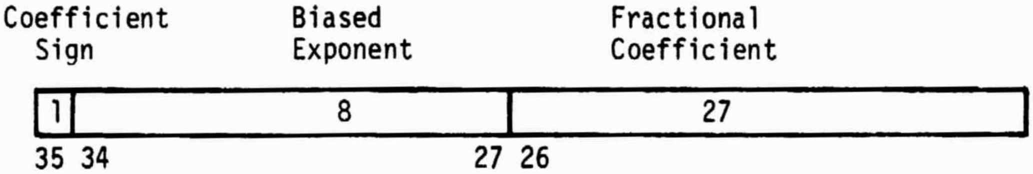
UNIVAC 1108--

The Univac 1108 processor can operate with two forms of floating point arithmetic: Single-precision and double-precision.

Single-precision instructions produce double-precision results, i.e., a two word results.

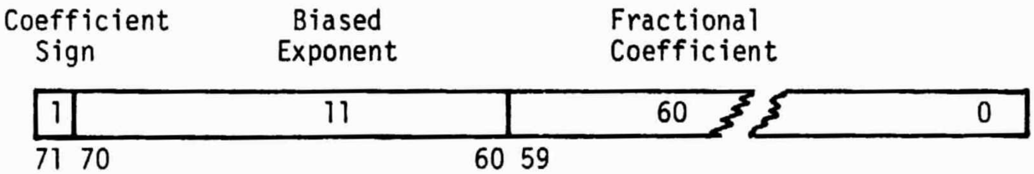
The 1108 requires that the coefficients and the exponents with their separate signs be provided in the following formats:

TABLE I (concluded)



Single-Precision Floating Point

The coefficient is the numerical value of the data, it is always a fractional value less than 1. However, the exponent is not the exponent of the coefficient; it is the exponent of the base.



Dougle-Precision Floating Point Number

For single-precision the 8 bit exponent carries a bias of $2^7(200_8)$.
For double precision the 11 bit exponent carries a bias of $2^{10}(2000_8)$.

16) FIXED POINT WORD STRUCTURE

CDC 6000--

Negative numbers are represented in one's complement notation and overflows are ignored. The sign bit is in the high-order bit position (bit 59) and the binary point is at the right of the low-order bit position (bit 0).

UNIVAC 1108---

The basic fixed-point number is a 36 bit binary word. The sign bit is in the high-order bit position (bit 35) and the binary point is at the right of the low-order bit position (bit 0).

3. PROGRAM CONVERSION REQUIREMENTS

3.1 Basic Programming Differences - Table II lists the differences resulting from the comparisons made in Table I. From these results, the following basic programming guidelines are indicated as essential for conversion of a FORTRAN IV CDC 6500 program to FORTRAN V for the UNIVAC 1108:

a. The 60 bit word of the CDC 6500 allows 10 Hollerith characters. The UNIVAC 36 bit word permits only six characters per word. Therefore, all words must be reduced to six or less characters.

b. The CDC 6500 FORTRAN IV compiler allows symbols with up to seven alphanumeric characters. UNIVAC 1108 FORTRAN V permits only six alphanumeric characters. Symbols must therefore be reduced to six or less alphanumeric characters.

c. CDC FORTRAN IV allows Hollerith characters in FORMAT statements to be specified by placing asterisks around the Hollerith information. UNIVAC uses this feature with apostrophes instead of asterisks.

d. CDC FORTRAN IV allows the DO loop index to be altered by a replacement statement within the loop. UNIVAC FORTRAN V does not permit DO indices to be altered within the iteration loop. Do not reset DO loop indices within the iteration loop.

e. CDC FORTRAN IV allows the DO loop index to be referenced outside the limits of the loop. UNIVAC does not have this feature. Therefore, the index value must be stored in a separate word.

f. The CDC loader allocates storage for common blocks according to their first appearance in the job string. The UNIVAC loader searches the referenced elements and allocates storage for the longest appearance of the common block. Therefore, common statements must be kept consistent throughout all elements.

g. The CDC loader creates an absolute element of all elements in the job string including BLOCK DATA subprograms. The UNIVAC loader loads only those elements that are

TABLE II. UNIVAC 1108/CDC 6500 Differences

ITEM	UNIVAC 1108	CONTROL DATA 6400/6500
MEMORY	$65K_{10}$ (Batch), $32K_{10}$ (Interact)	$132K_{10}$ (Batch), $52K_{10}$ (Interact)
INTERACTIVE TIME LIMIT	10 Minutes	5 Minutes
WORD LENGTH	36 Bits	60 Bits
FLOATING PT. BIAS	200_8	2000_8
DOUBLE PREC. BIAS	2000_8	2000_8
MANTISSA	FRACTION	INTEGER
REAL	$10^{-38} \leq R \leq 10^{38}$ (9 decimal digits)	$10^{-294} \leq R \leq 10^{322}$ (14 decimal digits)
DOUBLE PRECISION	$10^{-308} \leq D \leq 10^{308}$ (18 decimal digits)	$10^{-294} \leq D \leq 10^{322}$ (31 decimal digits)
INTEGER	$I \leq (2^{35} -1)$	$I \leq (2^{59} -1)$ DO index, $I \leq (2^{17} -2)$ $I -R$, $R \leq (2^{48} -1)$
CHARACTER	6 FIELDATA CHAR/WORD	10 Hollerith Char/Word
ARRAYS	UP TO 7 DIMENSIONS	UP TO 3 DIMENSIONS
WALK BACK	AVAILABLE	NA
FUNCTIONS: (1108 has many functions that are not available on CDC 6400/6500)	CBRT(X)	NA
	XØR(X_1, X_2, ..., X_n	NA
	BOOL	NA
	FLD	NA
ITEM EXECUTABLE	**UNIVAC 1108**	**CONTROL DATA 6400/6500**
DO n i = M_1, M_2, M_3	M_j may be signed constant or variable	M_j unsigned (variable may have negative value)
	M_j, i may not be altered	M_j, i may be changed during execution of DO
RETURN	RETURN k	RETURN

Table II. UNIVAC 1108/CDC 6500 Differences (cont)

ITEM EXECUTABLE	UNIVAC 1108	CONTROL DATA 6400/6500
STOP	STOP name	STOP 00000_8
PAUSE	PAUSE name	PAUSE 00000_8
READ	READ (u, f, ERR = c, END = d) list	READ (u, f) list
WRITE	WRITE (u, f, ERR = c, END = c)	WRITE (u, f) list
ENCODE	ENCODE (c, v, f) list v = block f - format c = no. of characters	ENCODE (c, f, v) list v - block f = format c = no. of characters in record
DECODE	DECODE (c, v, f) list	DECODE (c, f, v) list
BLOCK DATA	Reference in MAP	BLOCK DATA name
FORMAT	'hhh'	*hhh*
PARAMETER	PARAMETER	NA
INTEGER	INTEGER list /data/	INTEGER
REAL	REAL list /data/	REAL
DOUBLE PRECISION	DOUBLE PRECISION list /data/	DOUBLE PRECISION or DOUBLE
COMPLEX	COMPLEX list /data/	COMPLEX
LOGICAL	LOGICAL list /data/	LOGICAL
ABNORMAL	ABNORMAL	NA
DIMENSION	DIMENSION list /data/	DIMENSION
EQUIVALENCE	EQUIVALENCE (V(o), w)	EQUIVALENCE (X(n), w)
COMMON	Largest block loaded	First block loaded
DATA	Hollerith may be in 'hhh' nested implied DO's	Hollerith in nH field. Single variable subscript
END	n END n is statement number	END name name is element name
NAME LIST		

Table II. UNIVAC 1108/CDC 6500 Differences (concld)

ITEM NONEXECUTABLE	UNIVAC 1108	CONTROL DATA 6400/6500
ENTRY	ENTRY name (a_1, a_2, a_3) unique arguments not necessarily like parent	ENTRY name arguments defined in parent element
IMPLICIT	IMPLICIT	NA
DEFINE	DEFINE name (1) =	Name (1) =
COMPILER	COMPILER	NA
ITEM SOURCE CONTROL	**UNIVAC 1108**	**CONTROL DATA 6400/6500**
INCLUDE	INCLUDE	NA
DELETE	DELETE	NA
EDIT	EDIT	NA

referenced by the main program and those that are subsequently referenced. Therefore, BLOCK DATA subprograms must not be used. All elements to be loaded must be referenced - whether or not the CALL statement will be executed.

h. The following list indicates special CDC 6500 FORTRAN IV capabilities that are incompatible with UNIVAC 1108 FORTRAN V. An alternative compatible statement is shown where applicable.

CDC 6500 INCOMPATIBLE STATEMENT	COMPATIBLE ALTERNATIVE
DOUBLE LIST	DOUBLE PRECISION list
PAUSE 00000_8	PAUSE
STOP 00000_8	STOP
END name	END
IF (logical expression) N1, N2	IF (logical expression) GO TO
IF (UNIT, i) N1, N2, N3, N4	NTRAN
IF (EOF, i) N1, N2	NTRAN
IF (ENFILE i) N1, N2	NTRAN
BUFFER IN (i, m) list	NTRAN
BUFFER OUT (i, m) list	NTRAN

3.2 Control Language - Basic characteristic differences are to be expected between computers from different manufacturers having different capacities and capabilities. However, the differences do not end there. Unfortunately, no effective attempt has been made, up to the present time, to standardize control language. This has been left to the philosophy and desires of the managers or computer architects and will vary from one computer location to the next even on the same type computers. In order for programmers to use a program at a new location, they must learn and use the locally devised control language. Programming procedural variations will also be required by different peripheral input equipment. Apparently a complete control language has not yet been devised for the Marshall Interactive Planning System (MIPS). At this time, however, a description of some of the presently devised elements of the MIPS control language has been obtained and is contained in Table III.

TABLE III. MIPS Control Language Elements

ELEMENT	CONTROL PERFORMED
RN MOD	Effects the running of a system module mod
PT X	Tags X as a reference point
GO TO X	Effects a mandatory transfer of control to X
IF (A. OP. B) GO TO X	Effects a conditional transfer of control to X
SD	Delayed ST
SV MOD	Effects the saving of a command stack under the name MOD
DS MOD	Displays the module MOD
ER MOD	Erases the module MOD
UP	Effects the update of a module
GO	Effects the insertion of the command stack into the run stream
SP	Stops processing
ST MPS.A=C	Sets the value C under the name A in the specified data file MPS
PF MPS.A	Prints the value set under the specified name A in the specified data file MPS
-	Creates a data file. All required input is tutorial
EF MPS	Erases the specified data file MPS
PD	Prints system directories
READY MPS	Readies specified data file MPS

4. CONTAMINATION MODEL COMPUTER PROGRAM

4.1 Model Segments and Subroutines - The Spacelab Contamination Computer Model, which has basically been constructed and formatted for Spacelab design and development support, currently is comprised of a series of segregated elemental subroutines. These include unique programs which individually calculate the mass column density (MCD), number column density (NCD), return flux to a fixed geometry surface, and the resulting deposition for the major Spacelab contaminant sources. These elemental subroutines are also part of an integrated and more sophisticated program denoted as the VOLCAN (Vent, Outgassing and Leakage Contamination Analysis) Program. Although VOLCAN is still in the early development stage, the basic architecture has been established. Therefore, for an overview of the current modeling philosophy, a general description of the four basic segments of the VOLCAN Program from which the Spacelab Contamination Computer Model has been developed is presented herein.

The first segment of VOLCAN acts as the executive and coordinates the analysis by defining the problem and collecting the data required to perform the type of analysis the user requests. This executive segment in turn calls the three other segments if their functions are required.

Input data can be accessed under the following options:

a. user input cards;
b. previously generated permanent disk files or magnetic tapes; and
c. local files that are generated by the VOLCAN preprocessor by accessing the MSFC MIPS data files (the preprocessor is currently in the design stage so this option is not yet available).

Data is currently stored internally for analyzing the NASA Spacelab and Orbiter configurations, however data can be input to VOLCAN via cards or tape to analyze any satellite configuration.

A significant portion of the input data is geometric information assembled by a NASA sponsored radiation configuration program TRASYS. VOLCAN is formatted to accept output data from the MMC TRASYS program directly, but it could be easily modified to accept view factors, surface separation distances and angular relationships from other techniques that provide this information. Major subroutines contained in the first segment include:

COLLECT - controls input of data from various sources (i.e. cards, files, tapes)

BLOCK A - inserts block data for Spacelab/Orbiter surface identification numbers, material definition, location, surface area

AUDIT - defines a mass loss audit from all surface sources

MLOSSR - defines the mass loss rate of H_2O, N_2, CO_2, O_2, plus two outgassing species for each surface

LOADT - loads in temperatures from cards or permanent files depending on user options

If the user requests an evaluation of surface deposition, the executive calls the second segment which computes direct line-of-sight, transport of contaminants from all sources (distributed surfaces, engines, evaporator, leaks). Second surface sources for Spacelab are also considered in this segment by accessing a set of block data stored on a permanent file containing all body to body view factors. Mass arriving at a surface from other surfaces can be reflected or partially re-emitted and combined with the mass that originates with the surface itself.

Major subroutines contained in the second segment include:

DEPSIT - controls the type of sources that can deposit contaminants on the critical surface

ODRAP - computes outgassing/early desorption deposition

STICK - defines condensation coefficients for surfaces

PLUMES - computes mass flux to the surface from engines

VENT - computes mass flux to the surface from the evaporator

VELOC - calculates mean velocity of contaminants leaving a surface

Again depending on the type of analysis requested, the third segment can be called which contains the logic and mathematics for computing the mass or number column density through the cloud that surrounds the Spacelab/Orbiter. A capability now exists for computing MCD or NCD along generalized lines-of-sight with origins located at any station number.

Major subroutines used in this segment include:

PTSLCT - selects those precalculated view factors to points in the general cloud matrix that are required for a generalized line-of-sight

MCD - performs the integration along a line-of-sight

BUNCH - arranges the contributors in descending order of influence or in groups by material classification or in groups by location on the Spacelab/Orbiter

The fourth segment computes the flux of contaminants in the cloud surrounding the spacecraft that can return to critical surfaces thru interaction with the ambient or by self collision. The latest return flux model for contaminant self scattering has been refined and expanded for generalized 3 dimensional spacecraft and has been coded into the program. Major subroutines include:

RTFLX - controls computation of return flux

PTSLCT - defines those points in the general matrix required to encompass the field-of-view of the critical surface

4.2 Program Input Support - In order to determine the contamination characteristics for a particular mission or for a particular set of orbital operation conditions, specific initial conditions and mission operations data will be required to support the contamination model program. It is expected that much of these data will be made available from the MIPS data files through the Data Utility Module. Examples of such support data are the thermal profiles resulting from specific spacecraft orientations and orbit locations. Other input data requirements include:

a. experiment pointing directions;

b. experiment operational timelines;

c. reaction control motor operational timelines;

d. vent operation timelines;

e. shuttle attitude timeline;

f. spacecraft nonmetallic materials history (especially replacement and refurbishment history); and

g. initial outgassing and early desorption rates.

These data may be supplied either directly or indirectly from the MIPS data files when full capability is attained by that system.

The optimization of the Martin Marietta VOLCAN Program and the MIPS Data Bank could be realized through the development of a "preprocessor". Such a preprocessor would be a software package that has full knowledge of the MIPS data structure and the input data requirements of VOLCAN. The preprocessor would then build and format an updated input tape for each execution of the contamination model as current as the existing data banks in MIPS. This approach would save considerable engineering man hours by reducing the amount of labor involved in building such a tape by parts. The alternative would require programmers to learn the specific procedures required for accessing and interpreting the desired input data when they become available, formatting it accordingly, and manually constructing the input tapes.

5. SUMMARY

A summary of the results of the computer interface study leads to the following conclusions:

a. The UNIVAC 1108 is the most applicable computer in the MSFC computer complement for adaptation of the Spacelab Configuration Contamination Computer Model.

b. Language difference problems between the CDC 6500 and UNIVAC 1108 are minimal. However, plotting routines for different computer installations are unique, dependent upon the particular systems set up at particular locations, and will constitute a more involved problem than the language differences. Also, input/output formats and processes are unique, being a function of computer architect philosophy, and will require modification to the particular control language devised for MIPS.

c. The contamination program will require that a modified TRASYS or similar program be included in the MIPS so that modifications to geometrical parameters can be accommodated.

d. Optimization of the interface with the MIPS may require use of a special preprocessor program for extracting desired data from MIPS for inclusion in the VOLCAN Program or the Spacelab Contamination Computer Model.